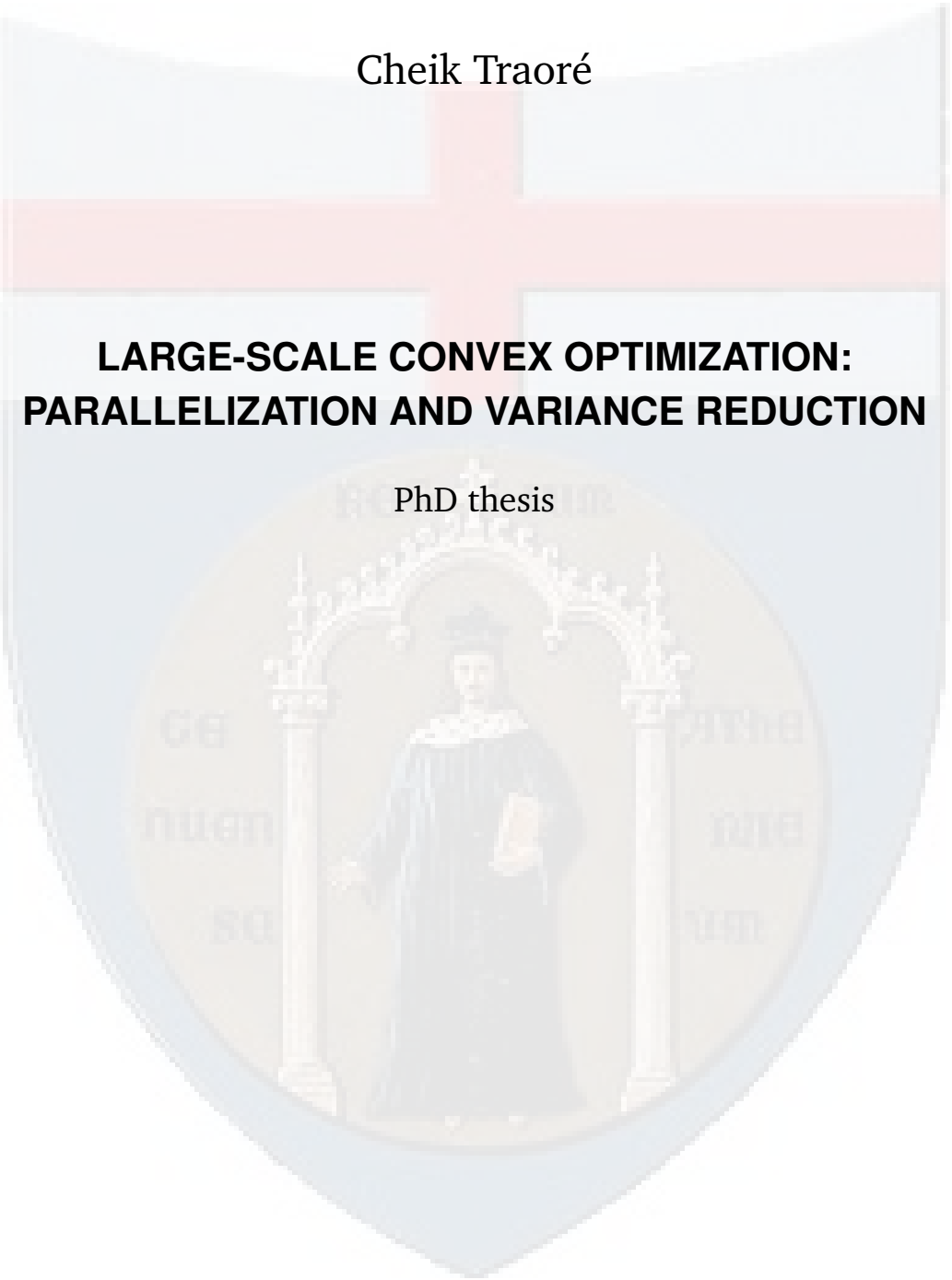
Genuense Athenaeum

Cheik Traoré

LARGE-SCALE CONVEX OPTIMIZATION: PARALLELIZATION AND VARIANCE REDUCTION

PhD thesis



Department of Mathematics
March 2024

University of Genoa

PhD Program in Mathematics and Applications

Mathematical Methods for Data Science Curriculum (8228)

Large-Scale Convex Optimization: Parallelization and Variance Reduction

by

Cheik Traoré

Thesis submitted for the degree of *Doctor of Philosophy* (Cycle XXXVI)

on March 11, 2024

Silvia Villa, *University of Genoa* — Thesis supervisor

Saverio Salzo, *Sapienza University of Rome* — Thesis co-supervisor

Stefano Vigni, *University of Genoa* — Head of the PhD program

Department of Mathematics (DIMA)

To my mom, my late dad, and my late sisters.

Abstract

Large-Scale Convex Optimization: Parallelization and Variance Reduction

Cheik Traoré

In this work, we investigate two aspects of large-scale optimization for convex functions defined on an infinite-dimensional separable Hilbert space: parallelized methods and incremental methods. These methods are used to efficiently solve problems that arise in data science, especially in machine learning and inverse problems.

In parallelized optimization methods, the computational load of running the algorithm is distributed among several workers. For example, if the algorithm comprises a gradient computation, one can give each worker a coordinate of the gradient to compute, and then put everything back together. A parallelized algorithm is called synchronous if there is a synchronization phase where local information of all workers are updated. It is called asynchronous if there is no such phase. In practice, asynchronous implementations are preferred to synchronous ones. However, their analysis has to account for delayed information, which is modeled by a delay vector. In this document, we study an asynchronous version of random block coordinate descent, where only one randomly selected coordinate is used at each iteration. We consider a version in which the selection probability of the coordinates is arbitrary, in contrast to what is done in the literature for asynchronous algorithms. We also allow coordinate-wise stepsize rule. Under convexity assumption, we prove weak convergence of the iterates and sublinear convergence rate. Assuming an additional error bound condition, we prove a linear convergence rate and strong convergence of the iterates. In both cases, the dependence on the delay vector is linear.

Incremental optimization methods are iterative algorithms used to minimize a function defined as a finite sum of functions. The function is then minimized by using one summand at each iteration instead of the whole function. We are interested in the case where the choice of the summand is random. This leads to stochastic algorithms such as stochastic gradient descent (SGD) or stochastic proximal point algorithm (SPPA). While they are cheaper to implement in terms of computation and memory than their deterministic counterparts that use the entire function, stochastic methods suffer from a drop in convergence rates. This drop is mainly due to the variance introduced by the stochasticity. Therefore, variance reduction techniques have been used in the literature to successfully recover the rates of deterministic algorithms. These techniques were first applied to stochastic gradient methods. In our work, we are, instead, concerned with the stochastic proximal point algorithm (SPPA). This method has recently been studied and has been shown to be more stable compared to stochastic gradient descent (SGD). Our work focuses on the application of variance reduction methods to SPPA. In particular, we introduce a general variance reduction scheme for SPPA. Many variance-reduced SPPA-based algorithms can be recovered from this scheme, mimicking those that already exist for SGD (SVRG, SAGA, etc.). We recover standard sublinear (respectively linear) convergence rates of the proximal point algorithm (PPA) when the stepsize is constant and the function is convex (respectively

convex plus satisfying the Polyak-Łojasiewicz condition).

Keywords. Convex optimization, asynchronous algorithms, randomized block-coordinate descent, error bounds, stochastic quasi-Fejér sequences, forward-backward algorithm, convergence rates, stochastic optimization, proximal point algorithm, variance reduction techniques, SVRG, SAGA.

Acknowledgements

First and foremost, I am grateful to Silvia Villa for offering me this fantastic opportunity and for the direction of research I followed. I also thank her and Saverio Salzo for their abundant and useful advice, and their great patience and comprehension throughout my time as their student. Saverio, thanks for the time and efforts invested in collaborating with us.

I would also like to thanks Radu Ioan Boţ and François Glineur for reviewing my thesis and for their recommendations to improve it.

Then I thank all MalGa and DIMA (professors, students, administrative staffs, visitors, and collaborators), for the great time I spent in the lab, at the department, on the football pitch, and in the city of Genova. I would like to single out and wholeheartedly thanks Giulia Casu and Nathalie Baxs for their tremendous help with the administrative duties that I encountered during my PhD and beyond.

Last but not least, I acknowledge that my PhD work has received funding from the European Union's Horizon 2020 research and innovation programme under the Marie Skłodowska-Curie grant agreement No 861137.

Table of Contents

CHAPTER 1

Introduction

1.1 General context and motivation

The impact that data science has on everyday life is incommensurable and continuously expanding. Imaging, language translation, self-driving cars, ChatGPT are some examples among the applications of this discipline. One of the major tools in data science is optimization, more precisely, numerical optimization. For example, in machine learning, the core of most "intelligent" systems, the training phase consists of minimizing a loss function that measures the discrepancy between a predicted value and the true value [94]. To achieve all the aforementioned prowesses, a huge amount of data is required more often than not. This poses many challenges to the field of optimization. Indeed, one expects to minimize a function not only as quickly and accurately[1], as possible, but also efficiently both in terms of computation and memory. The concepts of speed, accuracy and efficiency are of paramount importance in today's optimization theory and are at the core of various techniques used. The primary focus of this piece of work is on those aspects, specifically on the efficient implementation of algorithms for solving large-scale problems.

In continuous optimization, the iterative algorithms used to minimize a function need a direction in which to move at each iteration. This requires some idea of the variation of the objective function to be minimized. If the function has a Gâteaux differential or an appropriate generalized differential, the first variation is an excellent candidate. Indeed, the gradient captures the local variational behavior of a function at a point: it is the direction of maximum increase at any point where it is defined. Algorithms that use the derivative are called first-order methods. Consequently, those that use a higher-order derivative are called higher-order methods. Because of the cost of computing higher-order derivatives, first-order methods are the most popular methods even though they are sometimes slower.

The first-order methods that we are concerned with are the (sub)gradient descent (GD) [77]

$$(\forall k \in \mathbb{N}) \quad \mathbf{x}^{k+1} = \mathbf{x}^k - \gamma_k \mathbf{v}^k, \tag{GD}$$

and the proximal point algorithm (PPA), first introduced in [72]

$$(\forall k \in \mathbb{N}) \quad \mathbf{x}^{k+1} = \mathbf{x}^k - \gamma_k \mathbf{v}^{k+1}, \tag{PPA}$$

where $\mathbf{v}^k$ is the gradient or a subgradient of F at $\mathbf{x}^k$ for all $k \in \mathbb{N}$. The computation of $\mathbf{v}^k$ can be very expensive or impossible to perform. There are few ways to go around those difficulties. In this document, we shall concentrate our attention on two of them.

[1] By accurately, we mean closely enough to a (local) minimum.

Parallelization. A solution, when computation is expensive, is to distribute the load among many workers (machines, cores, computers, etc.), i.e., to parallelize the system. To achieve this, the algorithm should support parallelization. For instance, the gradient in (GD) can be divided into blocks of coordinates; each worker computes a block, and everything is then assembled. In this case, you should have as many workers as blocks. A suitable variant of GD for parallelization is therefore coordinate descent, defined for a differentiable function $F\colon \mathbb{R}^d \to \mathbb{R}$, by

$$(\forall k \in \mathbb{N}) \quad \mathsf{x}_{i_k}^{k+1} = \mathsf{x}_{i_k}^{k} - \gamma_{i_k}^k \nabla_{i_k} F(\mathbf{x}^k), \tag{CD}$$

where $i_k \in \{1, 2, \cdots, d\}$, $\nabla_{i_k} F(\mathbf{x}^k)$ is the ${i_k}^{th}$ partial derivative of F at $\mathbf{x}^k$ and the blocks are composed of one coordinate. Generally, there are two approaches to parallelization. First, it can be done synchronously. In such a system, the computation workload and the data are spread across the workers. Then there are synchronization steps where all the local information is consolidated to make progress [2, 34, 37, 42]. For example, in the case of CD and with d machines, if the synchronization is done at each iteration, it will correspond to GD with the computation of the gradient spread across the machines. The synchronization steps introduce additional overhead, and during those phases, the algorithm is as slow as the slowest worker because it has to wait for all the workers to provide their local information. This is avoided by performing all steps asynchronously (no synchronization phase), known as asynchronous parallelization. In an asynchronous system, workers can progress freely without waiting for each other in a synchronization phase.

In our work, we focus on asynchronous algorithms. We specifically investigate the convergence properties of a version of coordinate descent CD with asynchronous parallelization, defined here AsyncBlockProxGrad. We will dive deeper into this later in this document.

Approximation of the (sub)gradient. A second solution and an efficient alternative is to replace the (sub)gradient with a sufficiently good approximation that is less expensive to compute or store. This approximation should be good enough in the sense that speed and accuracy are not significantly compromised. Various methods can be employed to approximate the (sub)gradient depending on the property or structure of the function F. When only the function values are available, a finite difference approximation can be used, as seen in some zeroth-order optimization methods [78]. Another popular option is to use a stochastic estimator to approximate the true gradient. This is common in the realm of stochastic approximation [88], where the function to minimize is the expectation of a random function. In this situation, the gradient of a realization of the random function is used instead of the gradient of the expected value. A widespread example in machine learning, which we address later in this work, is empirical risk minimization obtained by using an empirical expectation. Unfortunately, when compared to their deterministic counterparts, stochastic methods, such as Stochastic Gradient Descent (SGD) in the differentiable case and Stochastic Proximal Point Algorithm (SPPA), lag in terms of accuracy when a constant stepsize is used. That is to say that, assuming convexity, they converge to a ball around the minimum rather than the minimum itself. This behavior is primarily attributed to the variance introduced by the stochasticity; for more details, see point (i) of Remark 2.15. To have them to converge to the minimum, a vanishing stepsize is needed to cancel out the variance. But, in that case, when it comes to convergence rate (speed), stochastic methods lose to the deterministic ones. That is due to the fact that the latter methods do not need a vanishing stepsize for convergence. A vanishing stepsize always slows down algorithms, in this instance stochastic algorithms [12]. In a subsequent section, we will explore various techniques that have been successfully introduced to recover both the convergence rate

and accuracy by algorithmically reducing the variance instead of the vanishing stepsize trick.

The contribution of our document on that topic is to study these variance reduction techniques for the stochastic version, SPPA, of PPA. To be more specific, we introduced a unified way of studying several variance reduction techniques for SPPA mirroring what has been done for SGD [44].

1.2 Thesis outline and contributions

This thesis revolves around two aspects of large-scale optimization: parallelization and variance reduction which will be treated separately in two different chapters, namely Chapter 3 and Chapter 4 respectively. Our study is carried out in infinite dimensional and separable real Hilbert spaces, unless stated otherwise. The following is the outline of the rest of the thesis.

Chapter 2: This chapter's objective is to recall some background information that is relevant to the next two chapters. We first explain the forward-backward algorithm and its asynchronous implementation. Then, we briefly present stochastic gradient descent SGD and compare it to its deterministic counterpart GD. Subsequently, we show the main variance reduction techniques introduced in the literature to make SGD "as fast as" GD.

Chapter 3: Associated with the publication:

TRAORÉ, C., SALZO, S., AND VILLA, S. (2023). "Convergence of an Asynchronous Block-Coordinate Forward-Backward Algorithm for Convex Composite Optimization". *Computational Optimization and Applications*, 86(1), 303-344.

In this chapter, we focus on an asynchronous implementation of the random coordinate forward-backward algorithm. In contrast to the literature of asynchronous forward-backward algorithms, we use an arbitrary probability of coordinates selection, not necessarily uniform. We also introduce a coordinate-dependent stepsize rule. Considering convex and (smooth + nonsmooth) composite functions, we

- prove almost sure (a.s.) weak convergence of the iterates to a random variable taking values in the set of minimizers.
- provide a convergence rate of $o(1/k)$ in expectation.
- have the state-of-the-art dependence on the delay, which is linear for both stepsizes and convergence constants.

Assuming an error bound condition on top of the previous assumptions, we

- prove almost sure (a.s.) strong convergence of the iterates to a random variable taking values in the set of minimizers.
- prove, in expectation, a linear convergence rate $O(\varepsilon^k)$ with $0 < \varepsilon < 1$.

Chapter 4: Based on the following submitted preprint:

TRAORÉ, C., APIDOPOULOS, V., SALZO, S., AND VILLA, S. (2023). "Variance reduction techniques for stochastic proximal point algorithms". *arXiv preprint arXiv:2308.09310*.

In this chapter, we perform a unified study of variance reduction techniques for stochastic proximal point algorithms (SPPA).

- First, we define a generic variance-reduced SPPA. For convex and smooth functions, a sublinear convergence rate is provided for this generic algorithm. For convex and smooth functions that satisfy the Polyak-Łojasiewicz condition, we prove a linear convergence rate for it.
- When specified, this generic algorithm recovers, along with their convergence rates, proximal variance-reduced algorithms such as SVRP, L-SVRP, and SAPA, which are proximal versions of SVRG, L-SVRG [60], and SAGA, respectively.

While these variance-reduced algorithms for SPPA have been individually studied in the literature with the stronger assumption of strong convexity, non-constant stepsizes, or, in the case of SAPA, a slight variation of the algorithm [29, 55, 73], our work represents the first unified study of variance reduction techniques for SPPA. Our generic analysis might be specialized to even more examples than the ones cited above.

Paper not included in this thesis:

TRAORÉ, C., AND PAUWELS, E. (2021). Sequential convergence of AdaGrad algorithm for smooth convex optimization. *Operations Research Letters*, 49(4), 452-458.

1.3 Preliminaries

We assume that the reader is familiar with basics of point topology and differential calculus. If not, he or she can refer to [8] for additional insights.

We first introduce the notations and recall a few basic notions about convex analysis that will be needed throughout this document.

1.3.1 Notations

We denote by $\mathbb{N}$ the set of natural numbers (including zero), $\mathbb{R}_+ = [0, +\infty[$ and $\mathbb{R}_{++} =]0, +\infty[$. For every integer $\ell \geq 1$ we define $[\ell] = \{1, \dots, \ell\}$. H always denotes an arbitrary separable real Hilbert space endowed with a scalar product $\langle \cdot, \cdot \rangle$ and its induced norm $\|\cdot\|$. $\mathbf{H}$ is, instead, the direct sum of m separable real Hilbert spaces $(\mathsf{H}_i)_{1 \leq i \leq m}$, i.e. $\mathbf{H} = \bigoplus_{i=1}^m \mathsf{H}_i$. For all $i \in [m]$, we denote indifferently the scalar products of $\mathbf{H}$ and H_i by $\langle \cdot, \cdot \rangle$ and:

$$(\forall \mathbf{x} = (\mathsf{x}_1, \cdots, \mathsf{x}_m), \mathbf{y} = (\mathsf{y}_1, \cdots, \mathsf{y}_m) \in \mathbf{H}) \quad \langle \mathbf{x}, \mathbf{y} \rangle = \sum_{i=1}^m \langle \mathsf{x}_i, \mathsf{y}_i \rangle.$$

$\|\cdot\|$ and $|\cdot|$ represent the norms associated to their scalar product in $\mathbf{H}$ and in any of H_i respectively. We also consider the canonical embedding, for all $i = 1, 2, \cdots, m$, $\mathsf{J}_i \colon \mathsf{H}_i \to \mathbf{H}$, $\mathsf{x}_i \mapsto (0, \cdots, 0, \mathsf{x}_i, 0, \cdots, 0)$, with x_i in the i^{th} position.

The default font is used for random variables while sans serif font is used for their realizations or deterministic variables. The probability space underlying random variables is denoted by $(\Omega, \mathfrak{A}, \mathsf{P})$. For every random variable $\boldsymbol{x}$, $\mathsf{E}[\boldsymbol{x}]$ denotes its expectation, while if $\mathfrak{F} \subset \mathfrak{A}$ is a sub σ-algebra; we denote by $\mathsf{E}[\boldsymbol{x} \,|\, \mathfrak{F}]$ the conditional expectation of $\boldsymbol{x}$ given $\mathfrak{F}$. Also, $\sigma(\boldsymbol{x})$ represents the σ-algebra generated by the random variable $\boldsymbol{x}$.

Let $(\alpha_i)_{1 \leq i \leq m} \in \mathbb{R}^m_{++}$. The direct sum operator $\mathsf{A} = \bigoplus_{i=1}^m \alpha_i \mathsf{Id}_i$, where Id_i is the identity operator on H_i, is

$$\begin{aligned} \mathsf{A} \colon \mathbf{H} &\to \mathbf{H} \\ \mathbf{x} = (\mathsf{x}_i)_{1 \leq i \leq m} &\mapsto (\alpha_i \mathsf{x}_i)_{1 \leq i \leq m} \end{aligned}$$

This operator defines an equivalent scalar product on $\mathbf{H}$ as follows

$$(\forall \mathbf{x} \in \mathbf{H})(\forall \mathbf{y} \in \mathbf{H}) \qquad \langle \mathbf{x}, \mathbf{y} \rangle_{\mathsf{A}} = \langle \mathsf{A}\mathbf{x}, \mathbf{y} \rangle = \sum_{i=1}^{m} \alpha_i \langle \mathsf{x}_i, \mathsf{y}_i \rangle,$$

which gives the norm $\|\mathbf{x}\|_{\mathsf{A}}^2 = \sum_{i=1}^{m} \alpha_i |\mathsf{x}_i|^2$. We let

$$\mathsf{V} = \bigoplus_{i=1}^{m} \mathsf{p}_i \mathsf{Id}_i, \quad \Gamma^{-1} = \bigoplus_{i=1}^{m} \frac{1}{\gamma_i} \mathsf{Id}_i, \quad \text{and} \quad \mathsf{W} = \bigoplus_{i=1}^{m} \frac{1}{\gamma_i \mathsf{p}_i} \mathsf{Id}_i,$$

where for all $i \in [m]$, $\gamma_i > 0$ and $0 < \mathsf{p}_i < 1$. We set $\mathsf{p}_{\max} := \max_{1 \le i \le m} \mathsf{p}_i$ and $\mathsf{p}_{\min} := \min_{1 \le i \le m} \mathsf{p}_i$.

Let $\varphi \colon \mathsf{H} \to]-\infty, +\infty]$ be an extended real valued function. Arithmetic rules are extended to $-\infty$ and $+\infty$ as in [8]. The (effective) domain of φ is $\operatorname{dom} \varphi := \{\mathsf{x} \in \mathsf{H} \,|\, \varphi(\mathsf{x}) < +\infty\}$ and the set of minimizers of φ is $\operatorname{argmin} \varphi := \{\mathsf{x} \in \mathsf{H} \,|\, \varphi(\mathsf{x}) = \inf \varphi\}$. If $\inf \varphi$ is finite, it is represented by φ_*. When φ is differentiable $\nabla\varphi$ denotes the gradient of φ. The notation x_* means x is a minimizer of φ.

Now let $\varphi \colon \mathbf{H} \to]-\infty, +\infty]$. For all $\mathbf{u}, \mathbf{x} \in \mathbf{H}$ and any symmetric positive definite operator A, we have $\langle \nabla^{\mathsf{A}} \varphi(\mathbf{x}), \mathbf{u} \rangle_{\mathsf{A}} = \langle \nabla \varphi(\mathbf{x}), \mathbf{u} \rangle$, where ∇^{A} denotes the gradient operator in the norm $\|\cdot\|_{\mathsf{A}}$. Indeed

$$\lim_{t \to 0} \frac{\varphi(\mathbf{x} + t\mathbf{u}) - \varphi(\mathbf{x})}{t} = \langle \nabla \varphi(\mathbf{x}), \mathbf{u} \rangle = \langle \nabla^{\mathsf{A}} \varphi(\mathbf{x}), \mathbf{u} \rangle_{\mathsf{A}}.$$

This is true because the limit is independent of the norm on $\mathbf{H}$.

If $\mathsf{S} \subset \mathsf{H}$ is convex, closed and non empty, and $\mathsf{x} \in \mathsf{H}$, we set $\operatorname{dist}_{\mathsf{A}}(\mathsf{x}, \mathsf{S}) = \inf_{\mathsf{z} \in \mathsf{S}} \|\mathsf{x} - \mathsf{z}\|_{\mathsf{A}}$. The projection of x onto S is denoted by $P_{\mathsf{S}}(\mathsf{x}) := \inf_{\mathsf{z} \in \mathsf{S}} \|\mathsf{x} - \mathsf{z}\|$ and $P_{\mathsf{S}}^{\mathsf{A}}(\mathsf{x}) := \inf_{\mathsf{z} \in \mathsf{S}} \|\mathsf{x} - \mathsf{z}\|_{\mathsf{A}}$. $\forall\, \mathsf{S} \subset \mathsf{H}$, $\operatorname{int}(\mathsf{S})$ means interior of S.

In this work, ℓ^1 represents the space of real summable sequences and ℓ^2 the space of real sequences which are square summable.

The strong convergence (respectively weak convergence) of a sequence $(\mathsf{x}_n)_{n \in \mathbb{N}}$ to x is represented by $\mathsf{x}_n \xrightarrow{n} \mathsf{x}$ (respectively $\mathsf{x}_n \overset{n}{\rightharpoonup} \mathsf{x}$.).

1.3.2 Elements of convex analysis

In this section, we merely present the convex analysis tools that are used in this document. The interested reader can look at [8, 91] for additional results and proofs. Unless stated otherwise, $\varphi \colon \mathsf{H} \to]-\infty, +\infty]$ will always represent an extended real valued function.

We start by defining the fundamental geometrical property of convexity for sets.

Definition 1.1: A subset $C \subset \mathsf{H}$ is said to be convex if

$$(\forall\, \lambda \in [0,1]) \;\; (\forall\, \mathsf{x}, \mathsf{y} \in C) \quad \lambda \mathsf{x} + (1 - \lambda)\mathsf{y} \in C.$$

A set is convex if it contains the line segment joining any two points belonging to it. Below are few examples of convex sets; see Figure 1.1 for an illustration.

Example 1.2:

(i) The empty set, H, and the singletons $(\{\mathsf{x}\}, \mathsf{x} \in \mathsf{H})$ are convex.

(ii) For all $(\mathsf{x}_0, r) \in \mathsf{H} \times \mathbb{R}$, $\mathcal{B}(\mathsf{x}_0, r) := \{\mathsf{x} : \|\mathsf{x} - \mathsf{x}_0\| < r\}$ and $\mathcal{B}_c(\mathsf{x}_0, r) := \{\mathsf{x} : \|\mathsf{x} - \mathsf{x}_0\| \le r\}$ are convex.

The following one to one connection between functions and sets makes it possible to have also a notion of convexity for functions.

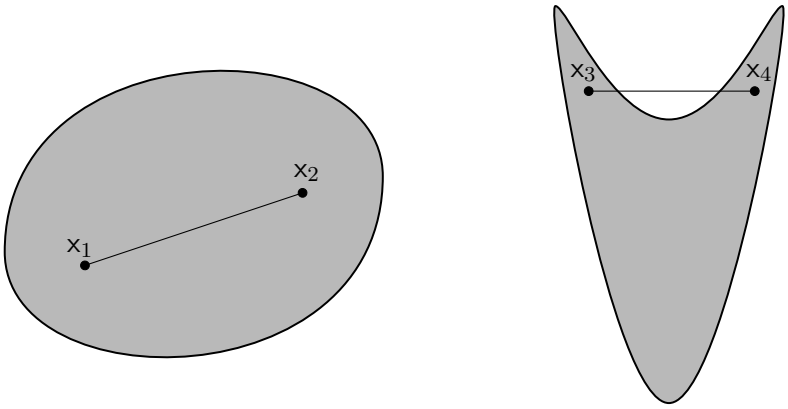


Figure 1.1: A convex set on the left and a nonconvex set on the right.

Definition 1.3: The epigraph of φ, denoted $\operatorname{epi}\varphi$, is

$$\operatorname{epi}\varphi := \{(x,t) \in H \times \mathbb{R} : \varphi(x) \leq t\}.$$

See Figure 1.2 for an illustration.

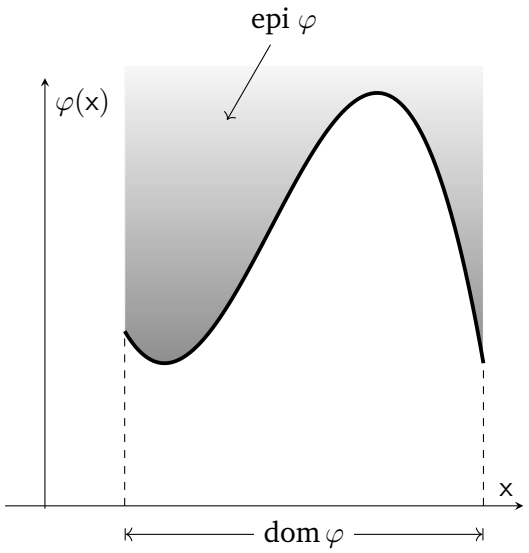


Figure 1.2: Representation of the epigraph of φ.

A convex function is then defined by:

Definition 1.4 (Convex functions): The function φ is convex if $\operatorname{epi}\varphi$ is a convex set in $H \times \mathbb{R}$. See Figure 1.3 for an illustration.

This can be equivalently formulated as follows.

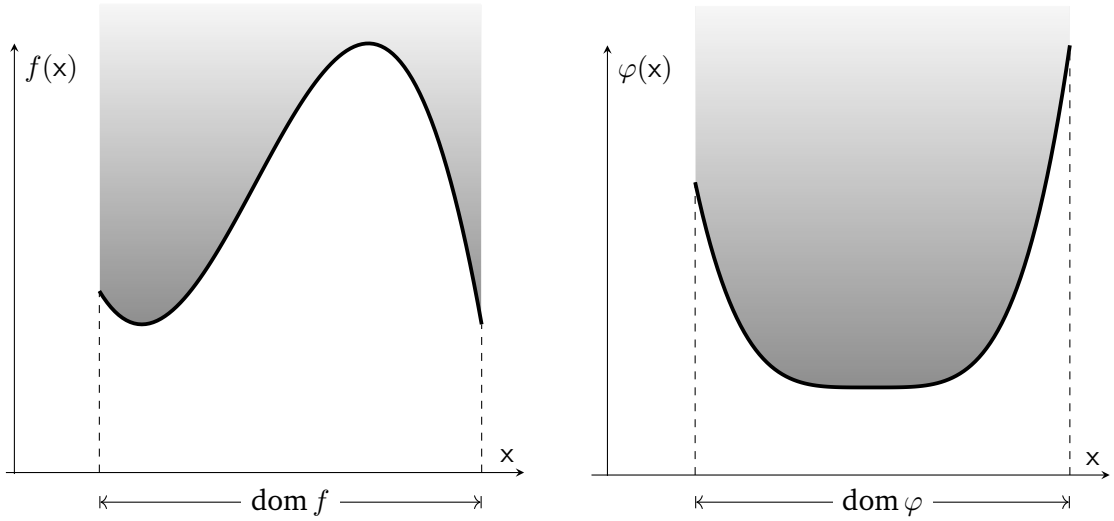


Figure 1.3: A nonconvex function f on the left and a convex function φ on the right.

Proposition 1.5

The following statements are equivalent:

(i) The function φ is convex.

(ii) For all $x, y \in \operatorname{dom}\varphi$ and $\lambda \in [0,1]$, the secant inequality

$$\varphi(\lambda x + (1-\lambda)y) \leq \lambda\varphi(x) + (1-\lambda)\varphi(y) \tag{1.3.1}$$

holds true; see Figure 1.4.

Assuming $\operatorname{dom}\varphi \neq \emptyset$ is convex and open, and φ is differentiable on $\operatorname{dom}\varphi$, the previous two statements are equivalent to the following one

$$(\forall\, x, y \in \operatorname{dom}\varphi) \quad \varphi(y) \geq \varphi(x) + \langle \nabla\varphi(x), y - x\rangle. \tag{1.3.2}$$

Definition 1.6: Suppose that there exists $x \in \operatorname{dom}\varphi$. Let $\mu \geq 0$. We say that φ is $\mu-$strongly convex if $\varphi - \frac{\mu}{2}\|\cdot\|^2$ is convex or equivalently, for all $\lambda \in [0,1]$ and $x, y \in \operatorname{dom}\varphi$,

$$\varphi(\lambda x + (1-\lambda)y) \leq \lambda\varphi(x) + (1-\lambda)\varphi(y) - \lambda(1-\lambda)\frac{\mu}{2}\|x-y\|^2.$$

Definition 1.7:

(i) φ is proper if $\operatorname{dom}\varphi \neq \emptyset$, i.e. there exists $x \in H$ such that $\varphi(x) < +\infty$.

(ii) φ is lower semicontinuous, abbreviated as l.s.c., if for all $(x_n)_{n\in\mathbb{N}} \subset H$, $\liminf_n \varphi(x_n) \geq \varphi(x)$ whenever $x_n \xrightarrow{n} x$.

(iii) φ is weakly lower semicontinuous, if for all $(x_n)_{n\in\mathbb{N}} \subset H$, $\liminf_n \varphi(x_n) \geq \varphi(x)$ whenever $x_n \xrightarrow{n} x$.

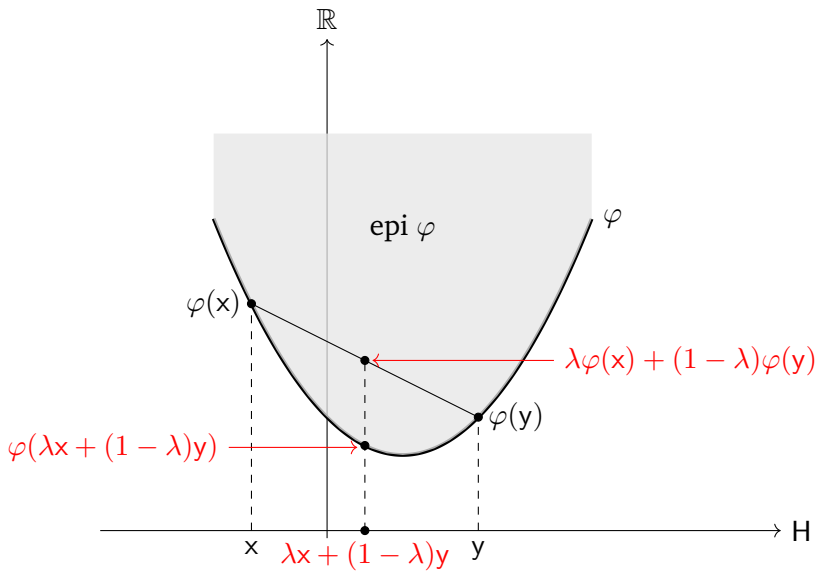


Figure 1.4: Secant inequality.

(iv) Let φ be proper. A global minimizer, or simply a minimizer, of φ is a point $x_* \in H$ such that

$$(\forall x \in H) \quad \varphi(x_*) \leq \varphi(x);$$

a local minimizer of φ is a point $x_* \in H$ such that

$$(\exists\, r > 0)(\forall x \in \mathcal{B}(x_*, r)) \quad \varphi(x_*) \leq \varphi(x).$$

Thanks to the next proposition, for convex function, we will only talk about lower semicontinuity.

Proposition 1.8

Let φ be convex. Then φ is weakly lower semicontinuous (w.l.s.c.) if and only if it is lower semicontinuous (l.s.c.).

We now give some definitions related to the notion of generalized gradient for convex functions.

Definition 1.9:

(i) Let φ be proper and convex, and $x \in \operatorname{dom}\varphi$. $u \in H$ is called a subgradient of φ at x if

$$(\forall y \in H) \quad \varphi(y) \geq \varphi(x) + \langle u, y - x\rangle, \tag{1.3.3}$$

and the subdifferential $\partial\varphi(x)$ of φ at x is the set $\{u \in H\colon (1.3.3) \text{ holds}\}$. The function φ is subdifferentiable at x if $\partial\varphi(x) \neq \emptyset$, i.e. if $x \in \operatorname{dom}\partial\varphi$. Whenever $x \notin \operatorname{dom}\varphi$, we set $\partial\varphi(x) = \emptyset$.

(ii) x is a critical point of φ if $0 \in \partial\varphi(x)$.

Proposition 1.10

Let φ be proper and convex.

(i) The domain of the subdifferential of φ is dense in the domain of φ if φ is l.s.c.

(ii) Let $\psi\colon \mathsf{H} \to]-\infty, +\infty]$ be convex and proper. If ψ is (Gâteaux) differentiable at $x \in \operatorname{int}(\operatorname{dom}\psi)$, then $\partial\psi(x) = \{\nabla\psi(x)\}$.

(iii) Assume (ii) with the same x. Then $\partial(\psi(x) + \varphi(x)) = \nabla\psi(x) + \partial\varphi(x)$.

(iv) If φ is l.s.c., the graph of the subdifferential of φ, Graph $\partial\varphi := \{(x, u) \in \mathsf{H} \times \mathsf{H}\colon u \in \partial\varphi(x)\}$, is closed in $\mathsf{H}^{\text{weak}} \times \mathsf{H}^{\text{strong}}$.

We have next an important property of convex functions that makes them extremely interesting in optimization.

Proposition 1.11

Let φ be proper and convex and let $x_* \in \operatorname{dom}\varphi$. These statements are equivalent

(i) x_* is a local minimizer of φ.

(ii) x_* is a (global) minimizer of φ.

(iii) x_* is a critical point of φ.

Next, we will give sufficient conditions for a function to have a minimizer.

Definition 1.12: Let φ be proper. φ is said coercive if

$$\lim_{\|x\|\to+\infty} \varphi(x) = +\infty,$$

which is equivalent to saying that, for every $\lambda \in \mathbb{R}$, $\{x \in \mathsf{H} : \varphi(x) \leq \lambda\}$ is bounded.

Proposition 1.13

Let φ be proper, weakly lower semicontinuous, and coercive. Then φ admits a minimizer.

Corollary 1.14: Let φ be proper, lower semicontinuous, and strongly convex. Then φ admits a unique minimizer.

Before moving forward, we introduce a notion that is not derived from convexity but will be needed in the thesis.

Definition 1.15: The function $\psi\colon \mathsf{H} \to \mathbb{R}$ is $L-$smooth if it is differentiable on $\mathsf{H} = \operatorname{dom}\psi$ and its gradient is $L-$Lipschitz on H for $L \geq 0$, i.e.

$$(\forall x, y \in \mathsf{H}) \quad \|\nabla\psi(x) - \nabla\psi(y)\| \leq L\|x - y\|.$$

Then we have the celebrated Descent Lemma.

Lemma 1.16 (Descent Lemma): Let $\psi\colon \mathsf{H} \to \mathbb{R}$ be $L-$smooth. Then for all $\mathsf{x}, \mathsf{y} \in \mathsf{H}$,

$$\psi(\mathsf{y}) - \psi(\mathsf{x}) - \langle \nabla\psi(\mathsf{x}), \mathsf{y} - \mathsf{x}\rangle \leq \frac{L}{2}\|\mathsf{x} - \mathsf{y}\|^2.$$

Finally, we introduce the notion of duality.

Definition 1.17: The Fenchel conjugate $\varphi^*\colon \mathsf{H} \to [-\infty, +\infty]$ of φ is the function

$$\mathsf{H} \ni \mathsf{u} \mapsto \sup_{\mathsf{x}\in\mathsf{H}} \langle \mathsf{u}, \mathsf{x}\rangle - \varphi(\mathsf{x}).$$

See Figure 1.5 for an illustration.

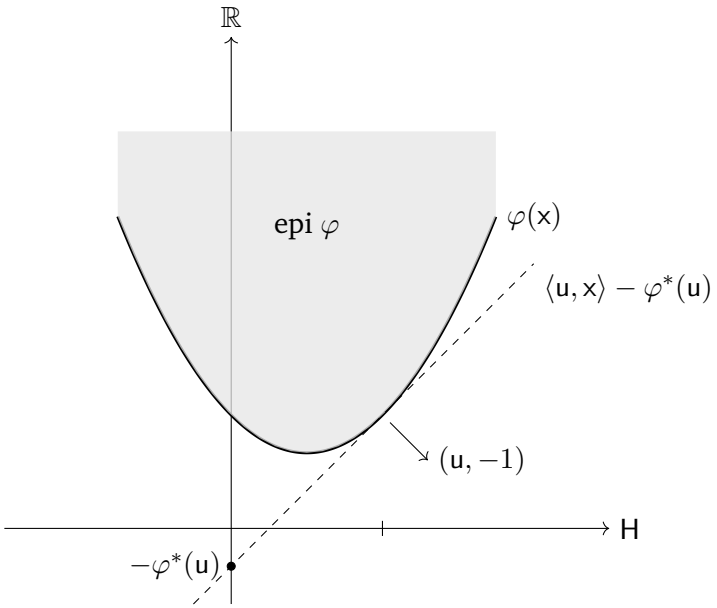


Figure 1.5: An explanation of the conjugate φ^* of φ.

Example 1.18:

(i) Let $\varphi(\mathsf{x}) = e^{\mathsf{x}}$. Then

$$\varphi^*(\mathsf{u}) = \begin{cases} \mathsf{u}\log\mathsf{u} - \mathsf{u} & \text{if } \mathsf{u} > 0 \\ 0 & \text{if } \mathsf{u} = 0 \\ +\infty & \text{if } \mathsf{u} < 0 \end{cases}$$

(ii) For every $\mathsf{u} \in \mathsf{H}$,

$$\left(\frac{1}{2}\|\cdot\|^2\right)^*(\mathsf{u}) = \langle \mathsf{u}, \mathsf{u}\rangle - \frac{1}{2}\|\mathsf{u}\|^2 = \frac{1}{2}\|\mathsf{u}\|^2.$$

Proposition 1.19

Let φ be proper. Then the following hold.

(i) $\varphi^* : \mathsf{H} \to]-\infty, +\infty]$ is convex and lower semicontinuous.

(ii) Let $\mu > 0$. Suppose that φ is convex and l.s.c. Then φ is $\mu-$strongly convex if and only if φ^* is $(1/\mu)-$smooth.

CHAPTER 2

Algorithmic background

In this chapter, we introduce some fundamental algorithms and techniques that are useful for the rest of the document. Throughout the chapter, $F\colon \mathbf{H} \longrightarrow]-\infty, +\infty]$ is supposed to be proper and bounded from below; so $F_* := \inf F > -\infty$.

Remark 2.1: For disclaimer, the rates of convergence of the different algorithms presented in this chapter are not necessarily the tightest ones in terms of the constants involved. What is relevant to the rest of the document and the goal of this chapter is to present and compare the order of the rates of convergence of those algorithms rather than the tightest constants of said rates. The interested reader can look in these references [99, 100, 101, 102], in most of which Performance Estimation technique is used.

We give one last definition before diving into the chapter.

Definition 2.2: A rate of convergence is called ergodic if it is given in terms of an average of the iterates generated by the algorithm, e.g. $\bar{\mathbf{x}}^k = \sum_{t=0}^{k-1} \alpha_t \mathbf{x}^t$ with $k \in \mathbb{N}$, $\sum_{t=0}^{k-1} \alpha_t = 1$ and $\mathbf{x}^t$ is the iterate generated at iteration t.

2.1 Forward-backward algorithm and its asynchronous version

2.1.1 Forward-backward algorithm

When $F\colon \mathbf{H} \longrightarrow]-\infty, +\infty]$ is convex, proper, l.s.c and differentiable on dom F, assumed open, the GD iteration can be rewritten equivalently as

$$(\forall k \in \mathbb{N}) \quad \mathbf{x}^{k+1} = \operatorname*{argmin}_{\mathbf{x}} F(\mathbf{x}^k) + \langle \nabla F(\mathbf{x}^k), \mathbf{x} - \mathbf{x}^k \rangle + \frac{1}{2\gamma_k} \|\mathbf{x} - \mathbf{x}^k\|^2. \tag{GD}$$

GD is well-defined since F is convex, proper and lower semicontinuous. In that case the function $\mathbf{y} \mapsto F(\mathbf{x}) + \langle \nabla F(\mathbf{x}), \mathbf{y} - \mathbf{x} \rangle + \frac{1}{2\gamma} \|\mathbf{y} - \mathbf{x}\|^2$ is strongly convex, proper and lower semicontinuous. So the minimizer exists and is unique; see Corollary 1.14. Equation GD shows that, at each iteration k, we are minimizing the first order Taylor approximation of F at $\mathbf{x}^k$ with the condition that the minimizer $\mathbf{x}^{k+1}$ is close to $\mathbf{x}^k$. This latter condition is imposed by the regularization term $\frac{1}{2\gamma_k} \|\mathbf{x} - \mathbf{x}^k\|^2$ and the stepsize γ_k controls how close $\mathbf{x}^{k+1}$ is to $\mathbf{x}^k$.

The PPA iteration can also be written equivalently, even (and especially) if F is not

differentiable, as

$$(\forall k \in \mathbb{N}) \quad \mathbf{x}^{k+1} = \operatorname{prox}_{\gamma_k F}(\mathbf{x}^k) := \operatorname*{argmin}_{\mathbf{x}} F(\mathbf{x}) + \frac{1}{2\gamma_k}\|\mathbf{x} - \mathbf{x}^k\|^2. \tag{PPA}$$

For the same reason as GD, PPA is well-defined since F is convex, proper and lower semicontinuous. In the case of PPA we are trying to find the minimizer $\mathbf{x}^{k+1}$ of F in the vicinity of $\mathbf{x}^k$, instead of the minimizer of its Taylor approximation like in GD.

Remark 2.3: Let A be a symmetric positive definite operator. We denote by $\operatorname{prox}^{\mathsf{A}}_{\gamma_k F}$ the proximal operator of F in the norm $\|\cdot\|_{\mathsf{A}}$.

Remark 2.4: Computing the proximity operator $\operatorname{prox}_{\gamma F}$ of a function F is in general difficult except for examples like the ℓ_1 norm $\mathbf{x} \mapsto \|\mathbf{x}\|_1$. Actually knowing how to compute the proximity operator of a function equates to knowing how to minimize that function. We just have to make γ tends to infinity to get the minimizer.
However, when the computation of the proximity operator is tractable and the function is nonsmooth, PPA is preferable compared to GD with subgradients. Indeed, their convergence rates for the function values are $O(1/k)$ with constant stepsize and ergodic $O(1/\sqrt{k})$ with vanishing stepsize respectively, for convex, proper and semicontinuous functions; see Theorem 2.5 for PPA and [86] for GD. This convergence property of PPA is very useful in composite or structured optimization, an example is given later in this section.

In this section, we will only present convergence rates for PPA. For GD, see Theorem 2.13 in Section 2.2.1.

Theorem 2.5: PPA rates [48, 91]

Let F be proper, convex and lower semicontinuous. Let $0 < \gamma_k = \gamma$ for every $k \in \mathbb{N}$.

(i) Suppose that $\operatorname{argmin} F \neq \emptyset$. Then $\forall\, k \in \mathbb{N}$,

$$F(\mathbf{x}^k) - F_* \leq \frac{\operatorname{dist}\left(\mathbf{x}^0, \operatorname{argmin} F\right)^2}{2\gamma}\frac{1}{k}.$$

(ii) If F is $\mu-$strongly convex with $\mu > 0$, let $\{\mathbf{x}_*\} = \operatorname{argmin} F$. Then $\forall\, k \in \mathbb{N}$,

$$\|\mathbf{x}^k - \mathbf{x}_*\| \leq \left(\frac{1}{1+\gamma\mu}\right)^k \|\mathbf{x}^0 - \mathbf{x}_*\|.$$

Remark 2.6: Now suppose that F is smooth and consider the following gradient flow

$$\dot{\mathbf{x}}(t) = -\nabla F(\mathbf{x}(t)).$$

The forward Euler discretization gives

$$\frac{\mathbf{x}_{k+1} - \mathbf{x}_k}{\gamma} = -\nabla F(\mathbf{x}_k),$$

which is exactly GD. In the same way, PPA is obtained by the following backward Euler discretization

$$\frac{\mathbf{x}_{k+1} - \mathbf{x}_k}{\gamma} = -\nabla F(\mathbf{x}_{k+1}).$$

Those denominations explain why the next algorithm we are going to present is called forward-backward.

Vanilla forward-backward algorithm

Sometimes, especially in data science, F is a composite function, i.e. $F = f + g$. For example in machine learning, the goal is to minimize a loss function f; but to prevent the model to overfit, a regularization term g is added to the loss. In this case, the problem considered is

$$\underset{\mathbf{x}\in\mathbf{H}}{\text{minimize}} \; F(\mathbf{x}) = f(\mathbf{x}) + g(\mathbf{x}).$$

In general F is not smooth. But sometimes, $f\colon \mathbf{H} \longrightarrow \mathbb{R}$ is smooth and $g\colon \mathbf{H} \longrightarrow]-\infty, +\infty]$, even though nonsmooth, has an easy-to-compute proximity operator. Instead of applying directly PPA to F, which can be difficult, we can take advantage of the smoothness of f and the fact that the proximity operator of g is easy to compute. So we split the algorithm by doing GD on f and PPA on g, hence the name splitting method. This gives the forward-backward (or proximal gradient descent) algorithm.

Algorithm 2.1 (Forward-Backward):
Let $x^0 = \mathbf{x}^0 \in \mathbf{H}$.

for $k = 0, 1, \ldots$

$$\left\lfloor \begin{aligned} \mathbf{x}^{k+1} &= \operatorname{argmin}_{\mathbf{x}\in\mathbf{H}} f(\mathbf{x}^k) + \langle \nabla f(\mathbf{x}^k), \mathbf{x} - \mathbf{x}^k\rangle + g(\mathbf{x}) + \tfrac{1}{2\gamma_k}\|\mathbf{x} - \mathbf{x}^k\|^2 \\ &= \operatorname{prox}_{\gamma_k g_k}\left(\mathbf{x}^k - \gamma_k \nabla f(\mathbf{x}^k)\right) \end{aligned} \right.$$

(ProxGrad)

Example 2.7: An example of a composite minimization problem is the constrained minimization over a closed convex set $\mathsf{C} \in \mathbf{H}$

$$\underset{\mathbf{x}\in\mathsf{C}}{\text{minimize}} \; f(\mathbf{x}),$$

which can be written as

$$\underset{\mathbf{x}\in\mathbf{H}}{\text{minimize}} \; F(\mathbf{x}) = f(\mathbf{x}) + g(\mathbf{x}),$$

where

$$g(\mathbf{x}) = \iota_{\mathsf{C}}(\mathbf{x}) \overset{\text{def}}{=} \begin{cases} 0 & \mathbf{x} \in \mathsf{C} \\ +\infty & \mathbf{x} \notin \mathsf{C} \end{cases}.$$

In this particular example, the proximity operator of g is the projection onto $\mathcal{C}$. The forward-backward algorithm applied to it is then called in the literature projected gradient descent.

Theorem 2.8: ProxGrad rates [91]

Let $f : \mathbf{H} \to \mathbb{R}$ an $L-$smooth convex function with $L > 0$, and g a proper, convex and lower semicontinuous function. Let $0 < \gamma_k = \gamma < 2/L$ and $\left(\mathbf{x}^k\right)_{k\in\mathbb{N}}$ be generated by Algorithm ProxGrad. Then the following statements hold

(i) Suppose that $\operatorname{argmin} F \neq \emptyset$. Then

$$F(\mathbf{x}^{k+1}) - F_* = o(1/(k+1)) \text{ and, for all } k \in \mathbb{N},$$

$$F(\mathbf{x}^{k+1}) - F_* \leq \frac{\operatorname{dist}\left(\mathbf{x}^0, \operatorname{argmin} F\right)^2}{k+1} \times \begin{cases} \frac{1}{2\gamma} & \text{if } \gamma \leq 1/L \\ \frac{L}{2}\frac{1}{2-\gamma L} & \text{if } 1/L < \gamma < 2/L. \end{cases}$$

(ii) Suppose there exists $\mu_f > 0, \mu_g \geq 0$ such that f is μ_f-strongly convex and g is μ_g-strongly convex. Let $\{\mathbf{x}_*\} = \operatorname{argmin} F$, and $0 < \gamma_k = \gamma < \frac{2}{L+\mu_f}$. Then

$$(\forall\, k \in \mathbb{N}) \quad \|\mathbf{x}^k - \mathbf{x}_*\|^2 \leq \left(\frac{1}{1+\gamma\mu_g}\right)\left(1 - \frac{2\gamma\mu_f L}{L+\mu_f}\right)^k \|\mathbf{x}^0 - \mathbf{x}_*\|^2.$$

(Random) Block-coordinate forward-backward algorithm

It can happen that g has more structure allowing for the more efficient block-coordinate proximal gradient descent. Let $\mathbf{H} = \bigoplus_{i=1}^m \mathsf{H}_i$. In addition, we have that g is separable, i.e. $g(\mathbf{x}) := \sum_{i=1}^m g_i(\mathsf{x}_i)$ for all $\mathbf{x} \in \mathbf{H}$. In that case we want to solve the following composite problem

$$\underset{\mathbf{x}\in\mathbf{H}}{\text{minimize}}\; F(\mathbf{x}) := f(\mathbf{x}) + g(\mathbf{x}), \qquad g(\mathbf{x}) := \sum_{i=1}^m g_i(\mathsf{x}_i). \tag{2.1.1}$$

The (random) block-coordinate forward-backward algorithm to solve Problem 2.1.1 is

Algorithm 2.2 (Block-coordinate forward-backward):
Let $\boldsymbol{x}^0 = \mathbf{x}^0 \in \mathbf{H}$.
for $k = 0, 1, \ldots$
  Choose i_k (randomly) in $[m]$. Then
  for $i = 1, \ldots, m$
    $$x_i^{k+1} = \begin{cases} \mathsf{prox}_{\gamma_k g_i}\left(x_i^k - \gamma_k \nabla_i f(\boldsymbol{x}^k)\right) & \text{if } i = i_k \\ x_i^k & \text{if } i \neq i_k, \end{cases}$$

(BlockProxGrad)

Example 2.9: An example of this type of problem is the Lasso problem. Given $\mathsf{A} \in \mathbb{R}^{n\times m}$ and $\mathsf{b} \in \mathbb{R}^n$, one wants to solve the problem

$$\underset{\mathbf{x}\in\mathbb{R}^m}{\text{minimize}}\; \frac{1}{2}\|\mathsf{A}\mathbf{x} - \mathsf{b}\|_2^2 + \lambda\|\mathbf{x}\|_1 \qquad (\lambda > 0)\,. \tag{2.1.2}$$

It is obvious that the Lasso problem corresponds to the template explained above with $f(\mathbf{x}) = (1/2)\|\mathsf{A}\mathbf{x} - \mathsf{b}\|_2^2$, $g(\mathbf{x}) = \lambda\|\mathbf{x}\|_1$ and $g_i(\mathsf{x}_i) = \lambda|\mathsf{x}_i|$. The proximity operator of $\mathsf{x}_i \mapsto \lambda|\mathsf{x}_i|$, for all $i \in [m]$, is called the soft thresholding operator

$$\mathsf{prox}_{\lambda|\cdot|}(\mathsf{x}_i) = \begin{cases} \mathsf{x}_i - \lambda & \text{if } \mathsf{x}_i > \lambda \\ 0 & \text{if } |\mathsf{x}_i| \leq \lambda \\ \mathsf{x}_i + \lambda & \text{if } \mathsf{x}_i < -\lambda \end{cases}.$$

The proximity operator of $\mathbf{x} \mapsto \lambda\|\mathbf{x}\|_1$ is a vector in $\mathbb{R}^m$ with each component i given by $\mathsf{prox}_{\lambda|\cdot|}(\mathsf{x}_i)$.

Theorem 2.10: BlockProxGrad rates [69, 90]

Let, $\forall\, i \in [m]$, f_i be convex and $L-$smooth and, g_i be proper, convex and lower semicontinuous. $\forall\, k \in \mathbb{N}$, i_k is chosen uniformly at random. Let $0 < \gamma_k = \gamma < 2/L$ and $(\boldsymbol{x}^k)_{k\in\mathbb{N}}$ be generated by BlockProxGrad.

(i) Suppose that argmin $F \neq \emptyset$. Then, for all $k \in \mathbb{N}$, $\mathsf{E}[F(\boldsymbol{x}^k)] - F_* = o(1/k)$ and

$$\mathsf{E}[F(\boldsymbol{x}^k)] - F_* \leq \left[\frac{\operatorname{dist}\left(\mathbf{x}^0, \operatorname{argmin} F\right)^2}{2\gamma} + \left(\max\left\{1, (2-\gamma L)^{-1}\right\} - \frac{1}{m} \right) \left(F(\mathbf{x}^0) - F_*\right) \right] \frac{m}{k}$$

(ii) Let $L > 0$. If there exists $\mu_f, \mu_g \geq 0$ such that f is μ_f-strongly convex and g is μ_g-strongly convex with $\mu_f + \mu_g > 0$ and $0 < \gamma_k = \gamma = 1/L$, then, for all $k \in \mathbb{N}$,

$$\mathsf{E}[F(\boldsymbol{x}^k)] - F_* \leq \left(1 - \frac{2(\mu_f + \mu_g)}{m(1 + \mu_f + 2\mu_g)}\right)^k \times \left(\frac{L(1+\mu_g)}{2} \operatorname{dist}\left(\mathbf{x}^0, \operatorname{argmin} F\right)^2 + F(\mathbf{x}^0) - F_* \right)$$

Thanks to its structure where updates are done coordinate-wise, BlockProxGrad allows for parallelization. We will explain next how this can be performed asynchronously.

2.1.2 Asynchronous algorithms

Mathematical modeling

In this section we discuss an example of an asynchronous parallel computational model, occurring in shared-memory system architectures, which can be covered by the proposed algorithm in Chapter 3. Consider a situation where we have a machine with multiple cores and the algorithm makes updates coordinate-wise. All the cores have access to a shared data $\boldsymbol{x} = (x_1, \ldots, x_m)$ and each core updates a block-coordinate x_i, $i \in [m]$, asynchronously without waiting for the others. The iteration's counter k is increased any time a component of $\boldsymbol{x}$ is updated. The value of the common data after k updates is written $\boldsymbol{x}^k$. When a core is given a coordinate to update, it has to read from the shared memory and compute a partial gradient. While performing these two operations, the data $\boldsymbol{x}$ may have been updated by other cores. So, when the core is updating its assigned coordinate at iteration k, the gradient might no longer be up-to-date. This phenomenon is modelled mathematically by using a delay vector $\mathbf{d}^k$ and using the partial gradient at $\boldsymbol{x}^{k-\mathbf{d}^k}$ for the update as in Algorithm 3.1. This is illustrated in Figure 2.1. Each component of the delay vector reflects how many times the data $\boldsymbol{x}$ have been updated since the core has read this particular coordinate from the shared memory. Note that different delays among the coordinates may arise since the shared data may be updated during the reading phase, so that the partial gradient ultimately is computed at a point which may not be consistent with any past instance of the shared data. This situation is called *inconsistent read* [13] and, in practice, allows a reading phase without any lock; see Figure 2.2. By contrast, in a *consistent read* model [67, 87], a lock is put during the reading phase and the delay originates only while computing the partial gradient. The delay is the same for all the block-coordinates, so that the value read by any core is a past instance of the shared data.

Asynchronous forward-backward algorithm

Under the consideration of the previous paragraph, the asynchronous version of Block-ProxGrad is given by the following algorithm

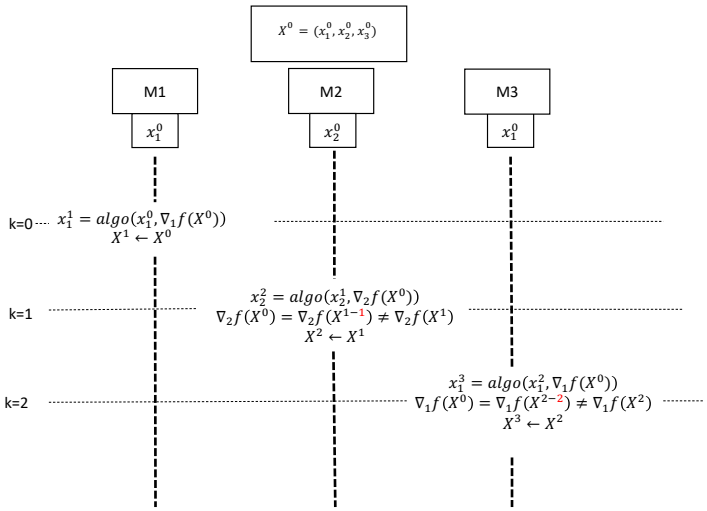


Figure 2.1: Delay representation. In this case we have 3 machines (M1, M2, M3) and a shared data X^0 at initialization. M1 and M3 have been assigned the same coordinate to update. M1 is faster to do is local partial gradient computation. So it makes the first update and counter move to 1. When M2 is ready to make its update at iteration 1, it has the partial gradient at 0 instead of 1. So it has a delay of 1. The counter moves to 2 after M2 makes its update. And M3 when doing its update has a delay of 2 because its partial gradient is still at 0.

Algorithm 2.3:
Let $\boldsymbol{x}^0 = \mathbf{x}^0 \in \mathbf{H}$.

for $k = 0, 1, \ldots$
  Choose i_k (randomly) in $[m]$. Then
  for $i = 1, \ldots, m$

$$x_i^{k+1} = \begin{cases} \mathsf{prox}_{\gamma_k g_i}\left(x_i^k - \gamma_k \nabla_i f(\boldsymbol{x}^{k-\mathbf{d}^k})\right) & \text{if } i = i_k \\ x_i^k & \text{if } i \neq i_k, \end{cases} \qquad \text{(AsyncBlockProxGrad)}$$

where $\boldsymbol{x}^{k-\mathbf{d}^k} = (x_1^{k-\mathsf{d}_1^k}, \ldots, x_m^{k-\mathsf{d}_m^k})$. This delay vector allows for an inconsistent read.

The convergence properties of AsyncBlockProxGrad will be investigated in Chapter 3.

2.2 Stochastic algorithms and variance reduction

The objective is, often in application, to solve the following finite-sum optimization problem

$$\underset{\mathbf{x}\in\mathsf{H}}{\text{minimize}} \;\; F(\mathbf{x}) = \frac{1}{n}\sum_{i=1}^{n} f_i(\mathbf{x}), \tag{2.2.1}$$

where H is a separable Hilbert space and for all $i \in \{1, 2, \cdots, n\}$, $f_i : \mathsf{H} \to \mathbb{R}$.

Several problems can be expressed as in (2.2.1). The most popular example is the Empirical Risk Minimization (ERM) problem in machine learning [96, Section 2.2]. In

Figure 2.2: Inconsistent read representation. On the left we have the local memory of M1. We can see that while it's reading the data coordinate by coordinate, machines M2 and M3 are updating the data. In the 4th iteration, M1 has locally a data with coordinates whose counters are different.

that setting, n is the number of data points, $\mathbf{x} \in \mathbb{R}^d$ includes the parameters of a machine learning model (e.g. linear functions, neural networks, etc.), and the function f_i is the loss of the model $\mathbf{x}$ on the i-th data point. Due to the large scale of data points used in machine learning/deep learning, leveraging gradient descent (GD) for problem (2.2.1) can be excessively costly both in terms of computational power and storage. To overcome these issues, several stochastic variants of gradient descent have been proposed in the last years.

2.2.1 Stochastic Gradient Descent (SGD)

The most common version of stochastic approximation [88] applied to (2.2.1) is the one where at each step the full gradient is replaced by ∇f_i, the gradient of a function f_i, with i sampled uniformly among $\{1, \cdots, n\}$. This procedure yields stochastic gradient descent (SGD), often referred to as incremental SGD.

Algorithm 2.4 (SGD):
Let $(i_k)_{k\in\mathbb{N}}$ be a sequence of i.i.d. random variables uniformly distributed on $\{1, \ldots, n\}$. Let $\gamma_k > 0$ for all $k \in \mathbb{N}$ and set the initial point $\boldsymbol{x}^0 = \mathbf{x}^0 \in \mathsf{H}$. Define

$$\begin{array}{l} \text{for } k = 0, 1, \ldots \\ \quad \boldsymbol{x}^{k+1} = \boldsymbol{x}^k - \gamma_k \nabla f_{i_k}(\boldsymbol{x}^k). \end{array} \qquad \text{(SGD)}$$

In its vanilla version or modified ones (AdaGrad [33], ADAM [59], etc.), SGD is ubiquitous in modern machine learning and deep learning. Stochastic approximation [88] provides the appropriate framework to study the theoretical properties of the SGD algorithm, which are nowadays well-understood [17, 18, 56, 74]. The theoretical analysis shows that SGD has a worse convergence rate compared to its deterministic counterpart GD. Indeed, SGD has the following rates [46]

Theorem 2.11: SGD rates

Assume that f_i is convex and $L-$smooth for all $i \in [n]$. Let $\mu > 0$. Suppose $0 < \gamma_k < \frac{1}{2L}$ and $\operatorname{argmin} F \neq \emptyset$, and set

$$\sigma^2 = \sup_{\mathbf{x} \in \operatorname{argmin} F} \mathsf{E}_i \|\nabla f_i(\mathbf{x})\|^2 < +\infty.$$

Then for every $k \in \mathbb{N}$,

(i) we have

$$\mathsf{E}[F(\bar{\boldsymbol{x}}^k)] - F_* \leq \frac{\operatorname{dist}\left(\mathbf{x}^0, \operatorname{argmin} F\right)^2}{2\sum_{t=0}^{k-1} \gamma_t(1-2\gamma_t L)} + \frac{2\sigma^2 \sum_{t=0}^{k-1} \gamma_t^2}{\sum_{t=0}^{k-1} \gamma_t(1-2\gamma_t L)},$$

where $\bar{\boldsymbol{x}}^k = \sum_{t=0}^{k-1} \frac{\gamma_t(1-2\gamma_t L)}{\sum_{t=0}^{k-1} \gamma_t(1-2\gamma_t L)} \boldsymbol{x}^t,$

(ii) if F is $\mu-$strongly convex and $\gamma_k = \gamma$, with $\{\mathbf{x}_*\} = \operatorname{argmin} F$,

$$\mathsf{E}\|\boldsymbol{x}^k - \mathbf{x}_*\|^2 \leq (1-\gamma\mu)^k \|\mathbf{x}^0 - \mathbf{x}_*\|^2 + \frac{2\gamma}{\mu}\sigma^2,$$

which implies

$$\mathsf{E}[F(\boldsymbol{x}^k)] - F_* \leq \frac{L}{2}(1-\gamma\mu)^k \|\mathbf{x}^0 - \mathbf{x}_*\|^2 + \frac{\gamma L}{\mu}\sigma^2.$$

Corollary 2.12 (Complexity of SGD). *We consider the general setting of Theorem 2.11. Let $\epsilon > 0$. Suppose $L > 0$. Then the following hold.*

(i) *Set $\gamma_k = \frac{1}{2L\sqrt{k}}$. Then for $k \geq 4$,*

$$k \geq \frac{1}{\epsilon^2}\left(2L\|\mathbf{x}^0 - \mathbf{x}_*\|^2 + \frac{\sigma^2}{L}\right)^2 \implies \mathsf{E}[F(\bar{\boldsymbol{x}}^k)] - F_* \leq \epsilon. \tag{2.2.2}$$

(ii) *if F is $\mu-$strongly convex and $\gamma_k = \gamma = \min\left\{\frac{\epsilon}{2}\frac{\mu}{L\sigma^2}, \frac{1}{2L}\right\}$, then*

$$k \geq \max\left\{\frac{1}{\epsilon}\frac{2L\sigma^2}{\mu^2}, \frac{2L}{\mu}\right\} \log\left(\frac{L\|\mathbf{x}^0 - \mathbf{x}_*\|^2}{\epsilon}\right) \implies \mathsf{E}[F(\boldsymbol{x}^k)] - F_* \leq \epsilon. \tag{2.2.3}$$

Whereas GD has these rates [46]

Theorem 2.13: GD rates

Let $\mu > 0$. Assume that f_i is convex and $L-$smooth for all $i \in [n]$ with $L > 0$. For every $k \in \mathbb{N}$,

(i) assuming that $\operatorname{argmin} F \neq \emptyset$, if $0 < \gamma_k = \gamma \leq \frac{1}{L}$,

$$F(\mathbf{x}^k) - F_* \leq \frac{\operatorname{dist}\left(\mathbf{x}^0, \operatorname{argmin} F\right)^2}{2\gamma k},$$

(ii) if F is $\mu-$strongly convex and $0 < \gamma_k = \gamma \leq \frac{1}{L}$, with $\{\mathbf{x}_*\} = \operatorname{argmin} F$,

$$\|\mathbf{x}^k - \mathbf{x}_*\|^2 \leq (1-\gamma\mu)^k \|\mathbf{x}^0 - \mathbf{x}_*\|^2,$$

which implies

$$F(\mathbf{x}^k) - F_* \leq \frac{L}{2}(1-\gamma\mu)^k \|\mathbf{x}^0 - \mathbf{x}_*\|^2.$$

Corollary 2.14 (Complexity of GD). *We consider the general setting of Theorem 2.13. Let $\epsilon > 0$. Then the following hold.*

(i) *If $0 < \gamma_k = \gamma \leq \frac{1}{L}$, then*

$$k \geq \frac{\|\mathbf{x}^0 - \mathbf{x}_*\|^2}{2\gamma\epsilon} \implies \mathsf{E}[F(\mathbf{x}^k)] - F_* \leq \epsilon. \tag{2.2.4}$$

(ii) *if F is $\mu-$strongly convex and $0 < \gamma_k = \gamma \leq \frac{1}{L}$, then*

$$k \geq \frac{1}{\mu\gamma}\log\left(\frac{L\|\mathbf{x}^0 - \mathbf{x}_*\|^2}{2\epsilon}\right) \implies \mathsf{E}[F(\mathbf{x}^k)] - F_* \leq \epsilon. \tag{2.2.5}$$

Remark 2.15:

(i) As exhibited in Theorem 2.11 (i) in the convex case, we have convergence for SGD if we make the usual assumption that $(\gamma_k)_{k\in\mathbb{N}} \in \ell_2 \setminus \ell_1$. Setting instead a constant stepsize γ yields

$$\mathsf{E}[F(\bar{\boldsymbol{x}}^k)] - F_* \leq \frac{\operatorname{dist}\left(\mathbf{x}^0, \operatorname{argmin} F\right)^2}{2\gamma(1-2\gamma L)k} + \frac{2\gamma}{(1-2\gamma L)}\sigma^2. \tag{2.2.6}$$

From Equation (2.2.6) above, in the convex case, when the stepsize is constant, we observe that, asymptotically, the function value optimality gap of SGD is bounded by $\frac{2\gamma}{(1-2\gamma L)}\sigma^2$, in expectation. But it does not necessarily go to 0 as for GD in Theorem 2.13. In the same way, for the strongly convex case, we can see, from Theorem 2.11 (ii), that, asymptotically, the iterates approach linearly, in expectation, a ball centered at x_* with radius $\frac{\gamma L}{\mu}\sigma^2$. They also do not necessarily go to x_* as for GD in Theorem 2.13. In both cases, this is due to the "variance" σ^2 at the minimum that is factored in those terms. To address this issue, it suffices to impose the stepsize to go appropriately to zero, which will eventually nullify the problematic terms; hence making the optimality gap tends to zero. In practice, this requirement is hard to follow; and the appropriate choice of the stepsize is one of the major issues in SGD implementations. In particular, if the initial stepsize is too big, SGD blows up even if the sequence of stepsizes satisfies the suitable decrease requirement, see, e.g., [74]. Therefore, the stepsize needs to be tuned by hand, and for solving problem (2.2.1), this is typically time-consuming.

(ii) GD exhibits complexities for the function values of order $O(1/\epsilon)$ for convex functions and $O(\log 1/\epsilon)$ for strongly convex functions. Meanwhile, SGD has worse complexities of the function values varying from $O(1/\epsilon^2)$ to $O(1/\epsilon)$ respectively.

2.2.2 Variance reduction methods for SGD

As observed above, the convergence rates for SGD are not only worse with respect to the ones of their deterministic counterpart, but also they are obtained for a vanishing stepsize. This is due to the non-vanishing variance of the stochastic estimator of the true gradient. In order to circumvent this issue, starting from SVRG [3, 53], a new wave of SGD algorithms were developed with the aim of reducing the variance and recovering standard GD rates with a constant stepsize. Different methods were developed, and all share a $O(1/k)$ and a $O(e^{-C_\mu k})$ convergence rate for function values for convex and μ-strongly convex objective functions, respectively. In the convex case the convergence is ergodic. Apart from SVRG, the other methods share the idea of reducing the variance by aggregating different stochastic estimates of the gradient. Among them, we mention SAG [92] and SAGA [30]. In subsequent years, a plethora of papers came out on variance reduction techniques, see for example [36, 60, 79, 111]. The paper [44] gave a unified study of variance reduction techniques for SGD that encompasses many of them. The latter work [44] has inspired our unified study of variance reduction for SPPA in Chapter 4.

The key to reducing the variance of a random variable is to use the following trick [30]. Let X be a random variable. Say we want to estimate E[X] and our initial estimator is X. Given another random variable Z whose expectation is easier to compute, we can construct a random variable X_Z with the same expectation as X by setting:

$$X_Z = X - Z + \mathsf{E}[Z].$$

It follows that variance of X_Z is

$$\mathrm{Var}(X_Z) = \mathrm{Var}(X) + \mathrm{Var}(Z) - 2\mathrm{Cov}(X, Z).$$

So if X and Y are correlated enough, i.e. $\mathrm{Cov}(X, Z)$ is big enough, we get that the variance of X_Z is less than variance of X; to this aim, it is sufficient to have $\mathrm{Cov}(X, Z) > \frac{1}{2}\mathrm{Var}(Z)$. In that case, not only X_Z is an unbiased estimator of $\mathsf{E}[X]$, but also it has less variance than X.

In the context of SGD, at each iteration k, $X = \nabla f_{i_k}(\mathbf{x}^k)$ and we use the conditional expectation $\mathsf{E}[X|\mathbf{x}^k] = \nabla F(\mathbf{x}^k)$. The goal of variance reduction techniques is to find a good Z and replace $X = \nabla f_{i_k}(\mathbf{x}^k)$ by X_Z. Depending on the choice of Z, we get different algorithms.

Now we will delve into more details about two relevant examples of variance reduction for SGD. We will present only two variance reduction techniques for conciseness purposes, although there are many more examples that appeared in the literature. The interested reader may refer to [36, 44, 60, 79, 111] for further information.

Stochastic variance reduced gradient (SVRG)

The first example we present is SVRG [53]. At iteration k, we set $Z = \nabla f_{i_k}(\tilde{\mathbf{x}})$, where $\tilde{\mathbf{x}}$ is some iterate in the past that is reused for a number of iterations.

Algorithm 2.5 (SVRG):
Let $m \in \mathbb{N}$, with $m \geq 1$, and $(\xi_s)_{s\in\mathbb{N}}$, $(i_t)_{t\in\mathbb{N}}$ be two independent sequences of i.i.d. random variables uniformly distributed on $\{0, 1, \ldots, m-1\}$ and $\{1, \ldots, n\}$ respectively. Let $\gamma > 0$ and set the initial point $\tilde{\boldsymbol{x}}^0 \equiv \tilde{\mathbf{x}}^0 \in \mathsf{H}$. Then

$$\begin{array}{l} \text{for } s = 0, 1, \ldots \\ \quad \boldsymbol{x}^0 = \tilde{\boldsymbol{x}}^s \\ \quad \text{for } k = 0, \ldots, m-1 \\ \quad\quad \boldsymbol{x}^{k+1} = \boldsymbol{x}^k - \gamma \left(\nabla f_{i_{sm+k}}(\boldsymbol{x}^s) - \nabla f_{i_{sm+k}}(\tilde{\boldsymbol{x}}^s) + \nabla F(\tilde{\boldsymbol{x}}^s) \right) \\ \quad \tilde{\boldsymbol{x}}^{s+1} = \sum_{k=0}^{m-1} \delta_{k,\xi_s} \boldsymbol{x}^k, \end{array} \tag{SVRG}$$

where $\delta_{k,h}$ is the Kronecker symbol.

Theorem 2.16: [53]

Assume that all f_i are convex and F is $\mu-$strongly convex with $\mu > 0$. Assume also that $0 < \gamma < \frac{1}{4L}$ and m is sufficiently large so that

$$\theta = \frac{1}{\mu\gamma(1-2L\gamma)m} + \frac{2L\gamma}{1-2L\gamma} < 1,$$

then we have geometric convergence in expectation for SVRG:

$$\mathsf{E}[F(\tilde{\boldsymbol{x}}^s)] - F_* \leq \theta^s \left(F(\tilde{\mathbf{x}}^0) - F_* \right)$$

Corollary 2.17. *Under the same assumptions as in Theorem 2.16, SVRG has an iteration complexity of order* $O\left((n + \frac{L}{\mu}) \log(\frac{1}{\varepsilon})\right)$ *[60].*

Stochastic average gradient algorithm

We can also set, at each iteration k, $X = \nabla f_{i_k}(\boldsymbol{\phi}_{i_k}^k)$, where $\boldsymbol{\phi}_{i_k}^k$ is some iterate in the past. But contrary to SVRG, it is used only once.

Algorithm 2.6 (SAGA):
Let $(i_k)_{k\in\mathbb{N}}$ be a sequence of i.i.d. random variables uniformly distributed on $\{1, \ldots, n\}$. Let $\gamma > 0$. Set the initial point $\boldsymbol{x}^0 \equiv \mathbf{x}^0 \in \mathsf{H}$ and, for every $i \in [n]$, $\boldsymbol{\phi}_i^0 = \mathbf{x}^0$. Then

$$\begin{array}{l} \text{for } k = 0, 1, \ldots \\ \quad \boldsymbol{x}^{k+1} = \boldsymbol{x}^k - \gamma \left(\nabla f_{i_k}(\boldsymbol{x}^k) - \nabla f_{i_k}(\boldsymbol{\phi}_{i_k}^k) + \frac{1}{n} \sum_{i=1}^n \nabla f_i(\boldsymbol{\phi}_i^k) \right), \\ \quad \forall\, i \in [n]\colon\ \boldsymbol{\phi}_i^{k+1} = \boldsymbol{\phi}_i^k + \delta_{i,i_k}(\boldsymbol{x}^k - \boldsymbol{\phi}_i^k), \end{array} \tag{SAGA}$$

where $\delta_{i,j}$ is the Kronecker symbol.

Theorem 2.18: [30]

Let $L, \mu > 0$. Let f_i be convex and $L-$smooth and $(x^k)_{k\in\mathbb{N}}$ be generated by SAGA, with $\gamma = \frac{1}{3L}$. The following hold.

(i) Assuming that argmin $F \neq \emptyset$, then, for every $k \in \mathbb{N}$,

$$\mathsf{E}[F(\bar{\boldsymbol{x}}^k)] - F_* \leq \frac{8L\,\mathrm{dist}(\mathbf{x}^0, \operatorname{argmin} F)^2 + 4n(F(\mathbf{x}^0) - F_*)}{k},$$

where $\bar{\boldsymbol{x}}^k = \dfrac{1}{k}\displaystyle\sum_{t=1}^{k} \boldsymbol{x}^t$.

(ii) If f_i is $\mu-$strongly convex and $\{\mathbf{x}_*\} = \operatorname{argmin} F$, then, for every $k \in \mathbb{N}$,

$$\mathsf{E}[||\boldsymbol{x}^k - \mathbf{x}_*||^2] \leq \theta^k \left(||\mathbf{x}^0 - \mathbf{x}_*||^2 + \frac{2}{3L}(F(\mathbf{x}^0) - F_*) \right),$$

which implies

$$\mathsf{E}[F(\boldsymbol{x}^k)] - F_* \leq \theta^k \frac{L}{2} \left(||\mathbf{x}^0 - \mathbf{x}_*||^2 + \frac{2}{3L}(F(\mathbf{x}^0) - F_*) \right),$$

with $\theta = 1 - \min\left\{\frac{1}{4n}; \frac{\mu}{3L}\right\}$.

Corollary 2.19. *Let $\varepsilon > 0$. Under the same assumptions in Theorem 2.18, we have the following iteration complexity*

(i) $k \geq O\left(\dfrac{n+L}{\varepsilon}\right) \implies \mathsf{E}[F(\bar{\boldsymbol{x}}^k)] - F_* < \varepsilon.$

(ii) $k \geq O\Big((n + L/\mu)\log(1/\varepsilon)\Big) \implies \mathsf{E}[F(\boldsymbol{x}^k)] - F_* < \varepsilon$, *if each f_i is $\mu-$strongly convex.*

Remark 2.20:

(i) Both SVRG and SAGA can recover GD rates without sacrificing too much computation efficiency of SGD.

(ii) SVRG needs a computation of the full gradient every m iterations instead of every iteration for GD. In that sense it is more costly that SGD but roughly m times less expensive than GD.

(iii) SAGA, on the other end, needs only one computation of the full gradient at the beginning of the algorithm. Basically it has the same computational cost as SGD. But a table of n stochastic gradients should be stored. Making it less memory efficient [30, 47].

(iv) As a general rule of thumb, when memory is a concern it is better to use SVRG. When it is not, SAGA may be a better choice.

2.2.3 Stochastic proximal point algorithm (SPPA)

In the past few years, stochastic proximal point algorithm (SPPA) has emerged in the literature. Just like for the gradient cases, SPPA uses at each iteration one summand instead of the whole function F. The algorithm is as follows

Algorithm 2.7:
Let $(i_k)_{k\in\mathbb{N}}$ be a sequence of i.i.d. random variables uniformly distributed on $\{1,\dots,n\}$. Let $\gamma_k > 0$ for all $k \in \mathbb{N}$ and set the initial point $\boldsymbol{x}^0 \equiv \mathbf{x}^0 =\in \mathsf{H}$. Define

$$\begin{array}{l}\text{for } k = 0, 1, \dots \\ \quad \boldsymbol{x}^{k+1} = \mathsf{prox}_{\gamma_k f_{i_k}}(\boldsymbol{x}^k).\end{array} \tag{SPPA}$$

Remark 2.21:

(i) Recent works, in particular [5, 57, 89], showed that SPPA is more robust to the stepsize choice compared to SGD.

(ii) On top of this, the convergence rates are the same as the ones of SGD, in various settings [5, 12, 81], possibly including a momentum term [57, 104, 105, 106]. That's also means that the convergence rates are worse than those of deterministic PPA. This is due to the variance of SPPA.

The goal of Chapter 4 of this document will be to study variance reduction techniques for SPPA in order to recover the standard rates of SPPA.

CHAPTER 3

Convergence of an Asynchronous Block-Coordinate Forward-Backward Algorithm for Convex Composite Optimization

In this chapter, we study the convergence properties of a randomized block-coordinate descent algorithm for the minimization of a composite convex objective function, where the block-coordinates are updated asynchronously and randomly according to an arbitrary probability distribution. We prove that the iterates generated by the algorithm form a stochastic quasi-Fejér sequence and thus converge almost surely to a minimizer of the objective function. Moreover, we prove a general sublinear rate of convergence in expectation for the function values and a linear rate of convergence in expectation under an error bound condition of Tseng type. Under the same condition strong convergence of the iterates is provided as well as their linear convergence rate.

3.1 Introduction

We recall the composite minimization problem

$$\underset{\mathbf{x}\in\mathbf{H}}{\text{minimize}}\; F(\mathbf{x}) := f(\mathbf{x}) + g(\mathbf{x}), \qquad g(\mathbf{x}) := \sum_{i=1}^{m} g_i(\mathsf{x}_i), \tag{3.1.1}$$

where $\mathbf{H}$ is the direct sum of m separable real Hilbert spaces $(\mathsf{H}_i)_{1\leq i\leq m}$, that is, $\mathbf{H} = \bigoplus_{i=1}^{m} \mathsf{H}_i$ and the following assumptions are satisfied unless stated otherwise.

Assumptions 3.1:

A1 $f\colon \mathbf{H} \to \mathbb{R}$ is convex and differentiable.

A2 For every $i \in \{1, \cdots, m\}$, $g_i\colon \mathsf{H}_i \to]-\infty, +\infty]$ is proper convex and lower semicontinuous.

A3 The map $\nabla f(\mathsf{x}_1, \ldots, \mathsf{x}_{i-1}, \cdot, \mathsf{x}_{i+1}, \ldots, \mathsf{x}_m)\colon \mathsf{H}_i \to \mathbf{H}$ is Lipschitz continuous with constant $L_{\text{res}} > 0$ and the map $\nabla_i f(\mathsf{x}_1, \ldots, \mathsf{x}_{i-1}, \cdot, \mathsf{x}_{i+1}, \ldots, \mathsf{x}_m)\colon \mathsf{H}_i \to \mathsf{H}_i$ is Lipschitz continuous with constant L_i, for all $\mathbf{x} \in \mathbf{H}$ and $i \in \{1, \cdots, m\}$. Note that $L_{\max} := \max_i L_i \leq L_{\text{res}}$ and $L_{\min} := \min_i L_i$.

A4 F attains its minimum $F_* := \min F$ on $\mathbf{H}$.

To solve problem 3.1.1, we use the following asynchronous block-coordinate descent algorithm. It is an extension of the parallel block-coordinate proximal gradient method

considered in [90] to the asynchronous setting, where an *inconsistent* delayed gradient vector may be processed at each iteration.

Algorithm 3.1:
Let $(i_k)_{k\in\mathbb{N}}$ be a sequence of i.i.d. random variables with values in $[m] := \{1,\dots,m\}$ and p_i be the probability of the event $\{i_k = i\}$, for every $i \in [m]$. Let $(\mathbf{d}^k)_{k\in\mathbb{N}}$ be a sequence of integer delay vectors, $\mathbf{d}^k = (\mathsf{d}_1^k,\dots,\mathsf{d}_m^k) \in \mathbb{N}^m$ such that $\max_{1\le i\le m} \mathsf{d}_i^k \le \min\{k,\tau\}$ for some $\tau \in \mathbb{N}$. This delay vector is deterministic and independent of the block coordinates selection process $(i_k)_{k\in\mathbb{N}}$. Let $(\gamma_i)_{1\le i\le m} \in \mathbb{R}^m_{++}$ and $\boldsymbol{x}^0 = (x_1^0,\dots,x_m^0) \in \mathbf{H}$ be a constant random variable. Iterate

$$\begin{array}{l}\text{for } k = 0,1,\dots \\ \quad \text{for } i = 1,\dots,m \\ \quad\quad x_i^{k+1} = \begin{cases} \mathsf{prox}_{\gamma_{i_k} g_{i_k}}\left(x_{i_k}^k - \gamma_{i_k}\nabla_{i_k} f(\boldsymbol{x}^{k-\mathbf{d}^k})\right) & \text{if } i = i_k \\ x_i^k & \text{if } i \ne i_k, \end{cases}\end{array} \tag{3.1.2}$$

where $\boldsymbol{x}^{k-\mathbf{d}^k} = (x_1^{k-\mathsf{d}_1^k},\dots,x_m^{k-\mathsf{d}_m^k})$.

In this work, we assume the following stepsize rule

$$(\forall\, i \in [m]) \quad \gamma_i(L_i + 2\tau L_{\mathrm{res}}\mathsf{p}_{\max}/\sqrt{\mathsf{p}_{\min}}) < 2, \tag{3.1.3}$$

where $\mathsf{p}_{\max} := \max_{1\le i\le m}\mathsf{p}_i$ and $\mathsf{p}_{\min} := \min_{1\le i\le m}\mathsf{p}_i$. If there is no delay, namely $\tau = 0$, the usual stepsize rule $\gamma_i < 2/L_i$ is obtained [26, 91].

The presence of the delay vectors in the above algorithm allows to describe a parallel computational model on multiple cores, as we explained in the introduction. We remark that, in our setting, for all $k \in \mathbb{N}$, the delay vector $\mathbf{d}^k$ is considered to be a parameter that does not dependent on the random variable i_k, similarly to the works [28, 49, 65, 67]. In this way, the stochastic attribute of the sequence $(\boldsymbol{x}^k)_{k\in\mathbb{N}}$ is not determined by the delay, but it only comes from the stochastic selection of the block-coordinates. Some papers consider the case where the delay vector is a stochastic variable that may depend on i_k [19, 97] or that it is unbounded [49, 97]. Those setting are natural extensions to our work that we are considering for future work. Finally, a completely deterministic model, both in the block's selection and delays is studied in [24].

Even though our study accounts for an inconsistent read paradigm, the theoretical results do not make any difference with the consistent read setting, because in the end only the maximum delay matters. In the literature other papers consider the inconsistent read and the results do not also make any difference with the consistent read, see [19, 28, 65].

3.1.1 Related work

The topic on parallel asynchronous algorithm is not a recent one. In 1969, Chazan and Miranker [20] presented an asynchronous method for solving linear equations. Later on, Bertsekas and Tsitsiklis [13] proposed an *inconsistent read* model of asynchronous computation. Due to the availability of large amount of data and the importance of large scale optimization, in recent years we have witnessed a surge of interest in asynchronous algorithms. They have been studied and adapted to many optimization problems and methods such as stochastic gradient descent [1, 38, 63, 80, 87], randomized Kaczmarz algorithm [66], and stochastic coordinate descent [6, 67, 82, 97, 110].

In general, stochastic algorithms can be divided in two classes. The first one is when the function f is an expectation i.e., $f(\mathsf{x}) = \mathsf{E}[h(\mathsf{x};\xi)]$. At each iteration k only a stochastic gradient $\nabla h(\cdot;\xi_k)$ is computed based on the current sample ξ_k. In this setting, many

asynchronous versions have been proposed, where delayed stochastic gradients are considered, see [7, 22, 38, 64, 71, 75]. The second class, which is the one we studied, is that of randomized block-coordinate methods. Below we describe the related literature.

The work [65] studied a problem and a model of asynchronicity which is similar to ours, but the proposed algorithm AsySPCD requires that the random variables $(i_k)_{k\in\mathbb{N}}$ are uniformly distributed (i.e, $p_i = 1/m$) and that the stepsize is the same for all the block-coordinates. This latter assumption is an important limitation, since it does not exploit the possibility of adapting the stepsizes to the block-Lipschitz constants of the partial gradients, hence allowing longer steps along block-coordinates. A linear rate of convergence is also obtained by exploiting a quadratic growth condition which is essentially equivalent to our error bound condition [32]. For a discussion on the limitations of [65] and the improvements we bring, see Remark 3.12 point (vi) and Section 3.6 on numerical experiments.

In the nonconvex case, [28] considers an asynchronous algorithm which may select the blocks both in an almost cyclic manner or randomly with a uniform probability. In the latter case, it is proved that the cluster points of the sequence of the iterates are almost surely stationary points of the objective function. However, the convergence of the whole sequence is not provided, nor is given any rate of convergence for the function values. Moreover, under the Kurdyka-Łojasiewicz (KL) condition [16, 32], linear convergence is also derived, but it is restricted to the deterministic case.

To conclude, we note that our results, when specialized to the case of zero delays, fully recover the ones given in [90].

3.1.2 Contributions

The main contributions of this work are summarized below:

- We first prove the almost sure weak convergence of the iterates $(\boldsymbol{x}^k)_{k\in\mathbb{N}}$, generated by Algorithm 3.1, to a random variable $\boldsymbol{x}_*$ taking values in $\operatorname{argmin} F$. At the same time, we prove a sublinear rate of convergence of the function values in expectation, i.e, $\mathsf{E}[F(\boldsymbol{x}^k)] - \min F = o(1/k)$. We also provide for the same quantity an explicit rate of $\mathcal{O}(1/k)$, see Theorem 3.11.

- Under an error bound condition of Luo-Tseng type, on top of the strong convergence a.s of the iterates, we prove linear convergence in expectation of the function values and in mean of the iterates, see Theorem 3.21.

We improve the state-of-the-art under several aspects: we consider an arbitrary probability for the selection of the blocks; the adopted stepsize rule improves over the existing ones, and coincides with the one in [28] in the special case of uniform selection of the blocks — in particular, it allows for larger stepsizes when the number of blocks grows; the almost sure convergence of the iterates in the convex and stochastic setting is new and relies on a stochastic quasi-Fejerian analysis; linear convergence under an error bound condition is also new in the asynchronous stochastic scenario.

The rest of the chapter is organized as follows. In the next subsection we set up basic notation. In Section 3.2 we recall few facts and we provide some preliminary results. The general convergence analysis is given in Section 3.3 where the main Theorem 3.11 is presented. Section 3.4 contains the convergence theory under an additional error bound condition, while applications are discussed in Section 3.5. The majority of proofs are postponed to Appendices 3.7 and 3.8.

3.2 Preliminaries

In this section we present basic definitions and facts that are used in the rest of the chapter. Most of them are already known, and we include them for clarity.

In the rest of the chapter, we extend the definition of $\boldsymbol{x}^k$ by setting $\boldsymbol{x}^k = \boldsymbol{x}^0$ for every $k \in \{-\tau, \dots, -1\}$. Using the notation of Algorithm 3.1, we also set, for any $k \in \mathbb{N}$

$$\begin{cases} \hat{\boldsymbol{x}}^k = \boldsymbol{x}^{k-\mathbf{d}^k} \\ \bar{x}_i^{k+1} = \mathsf{prox}_{\gamma_i g_i}\left(x_i^k - \gamma_i \nabla_i f(\hat{\boldsymbol{x}}^k)\right) \text{ for all } i \in [m] \\ \boldsymbol{x}^{k+1} = \boldsymbol{x}^k + \mathsf{J}_{i_k}\left[\mathsf{prox}_{\gamma_{i_k} g_{i_k}}\left(x_{i_k}^k - \gamma_{i_k} \nabla_{i_k} f(\hat{\boldsymbol{x}}^k)\right) - x_{i_k}^k\right] \\ \boldsymbol{\Delta}^k = \boldsymbol{x}^k - \bar{\boldsymbol{x}}^{k+1}. \end{cases} \tag{3.2.1}$$

With this notation, we have

$$\bar{x}_{i_k}^{k+1} = \mathsf{prox}_{\gamma_{i_k} g_{i_k}}\left(x_{i_k}^k - \gamma_{i_k} \nabla_{i_k} f(\hat{\boldsymbol{x}}^k)\right) = x_{i_k}^{k+1}; \qquad \Delta_{i_k}^k = x_{i_k}^k - x_{i_k}^{k+1}. \tag{3.2.2}$$

We remark that the random variables $\boldsymbol{x}^k$ and $\bar{\boldsymbol{x}}^{k+1}$ depend on the previously selected blocks, and related delays. More precisely, we have

$$\begin{aligned} \boldsymbol{x}^k &= \boldsymbol{x}^k(i_0, \dots, i_{k-1}, \mathbf{d}^0, \dots, \mathbf{d}^{k-1}) \\ \bar{\boldsymbol{x}}^{k+1} &= \bar{\boldsymbol{x}}^{k+1}(i_0, \dots, i_{k-1}, \mathbf{d}^0, \dots, \mathbf{d}^k). \end{aligned} \tag{3.2.3}$$

From (3.2.1) and (3.2.2), we derive

$$\frac{x_{i_k}^k - x_{i_k}^{k+1}}{\gamma_{i_k}} - \nabla_{i_k} f(\hat{\boldsymbol{x}}^k) \in \partial g_{i_k}(x_{i_k}^{k+1}) \quad \text{and} \quad \frac{x_i^k - \bar{x}_i^{k+1}}{\gamma_i} - \nabla_i f(\hat{\boldsymbol{x}}^k) \in \partial g_i(\bar{x}_i^{k+1}) \tag{3.2.4}$$

and therefore, for every $\mathsf{x} \in \mathbf{H}$

$$\langle \nabla_{i_k} f(\hat{\boldsymbol{x}}^k) - \frac{\Delta_{i_k}^k}{\gamma_{i_k}}, x_{i_k}^{k+1} - \mathsf{x}_{i_k} \rangle + g_{i_k}(x_{i_k}^{k+1}) - g_{i_k}(\mathsf{x}_{i_k}) \leq 0. \tag{3.2.5}$$

Suppose that x and x' in $\mathbf{H}$ differ only for one component, say that of index i, then it follows from Assumption A3 and the Descent Lemma [77, Lemma 1.2.3], that

$$\begin{aligned} f(\mathsf{x}') &= f(\mathsf{x}_1, \dots, \mathsf{x}_{i-1}, \mathsf{x}_i', \mathsf{x}_{i+1}, \cdots, \mathsf{x}_m) \\ &\leq f(\mathsf{x}) + \langle \nabla_i f(\mathsf{x}), \mathsf{x}_i' - \mathsf{x}_i \rangle + \frac{L_i}{2} |\mathsf{x}_i' - \mathsf{x}_i|^2 \qquad (3.2.6) \\ &\leq f(\mathsf{x}) + \langle \nabla f(\mathsf{x}), \mathsf{x}' - \mathsf{x} \rangle + \frac{L_{\max}}{2} \|\mathsf{x}' - \mathsf{x}\|^2. \qquad (3.2.7) \end{aligned}$$

We finally need the following results on the convergence of stochastic quasi-Fejér sequences and monotone summable positives sequences.

Fact 3.2 ([25], Proposition 2.3). *Let S be a nonempty closed subset of a separable real Hilbert space $\mathbf{H}$. Let $\mathscr{F} = (\mathcal{F}_n)_{n\in\mathbb{N}}$ be a sequence of sub-sigma algebras of $\mathcal{F}$ such that $(\forall n \in \mathbb{N})\ \mathcal{F}_n \subset \mathcal{F}_{n+1}$. We denote by $\ell_+(\mathscr{F})$ the set of sequences of $\mathbb{R}_+$-valued random variables $(\xi_n)_{n\in\mathbb{N}}$ such that, for every $n \in \mathbb{N}$, ξ_n is $\mathcal{F}_n$-measurable. We set*

$$\ell_+^1(\mathscr{F}) = \left\{ (\xi_n)_{n\in\mathbb{N}} \in \ell_+(\mathscr{F}) \,\middle|\, \sum_{n\in\mathbb{N}} \xi_n < +\infty \quad \mathsf{P}\text{-a.s.} \right\}.$$

Given a sequence $(x_n)_{n\in\mathbb{N}}$ *of* **H***-valued random variables, we define*

$$\mathscr{X} = (X_n)_{n\in\mathbb{N}}, \quad \text{where } (\forall n \in \mathbb{N}) \quad X_n = \sigma(x_0, \dots, x_n).$$

Let $(x_n)_{n\in\mathbb{N}}$ *be a sequence of* **H***-valued random variables. Suppose that, for every* $\mathsf{z} \in \mathsf{S}$*, there exist* $(\chi_n(\mathsf{z}))_{n\in\mathbb{N}} \in \ell^1_+(\mathscr{X})$*,* $(\vartheta_n(\mathsf{z}))_{n\in\mathbb{N}} \in \ell_+(\mathscr{X})$*, and* $(\eta_n(\mathsf{z}))_{n\in\mathbb{N}} \in \ell^1_+(\mathscr{X})$ *such that the stochastic quasi-Féjer property is satisfied* P*-a.s.:*

$$(\forall n \in \mathbb{N}) \quad \mathsf{E}\big[\|x_{n+1} - \mathsf{z}\|^2 \mid \mathcal{F}_n\big] + \vartheta_n(\mathsf{z}) \leqslant (1 + \chi_n(\mathsf{z}))\,\|x_n - \mathsf{z}\|^2 + \eta_n(\mathsf{z}).$$

Then the following hold:

(i) $(x_n)_{n\in\mathbb{N}}$ *is bounded* P*-a.s.*

(ii) *Suppose that the set of weak cluster points of the sequence* $(x_n)_{n\in\mathbb{N}}$ *is* P*-a.s. contained in* S*. Then* $(x_n)_{n\in\mathbb{N}}$ *weakly converges* P*-a.s. to an* S*-valued random variable.*

Fact 3.3 ([35, Example 5.1.5]). *Let* ζ_1 *and* ζ_2 *be independent random variables with values in the measurable spaces* $\mathcal{Z}_1$ *and* $\mathcal{Z}_2$ *respectively. Let* $\varphi\colon \mathcal{Z}_1 \times \mathcal{Z}_2 \to \mathbb{R}$ *be measurable and suppose that* $\mathsf{E}[|\varphi(\zeta_1, \zeta_2)|] < +\infty$*. Then* $\mathsf{E}[\varphi(\zeta_1, \zeta_2) \,|\, \zeta_1] = \psi(\zeta_1)$*, where for all* $z_1 \in \mathcal{Z}_1$*,* $\psi(z_1) = \mathsf{E}[\varphi(z_1, \zeta_2)]$*.*

Fact 3.4. *Let* $(a_k)_{k\in\mathbb{N}} \in \mathbb{R}^{\mathbb{N}}_+$ *be a decreasing sequence of positive numbers and let* $b \in \mathbb{R}_+$ *such that* $\sum_{k\in\mathbb{N}} a_k \leq b < +\infty$*. Then* $a_k = o(1/(k+1))$ *and for every* $k \in \mathbb{N}$*,* $a_k \leq b/(k+1)$*.*

Fact 3.5. *Let* $(a_k)_{k\in\mathbb{N}} \in \mathbb{R}^{\mathbb{N}}_+$ *be a sequence of positive numbers.* $(\forall\, n, k \in \mathbb{Z}, k \geq n)$*,*

$$\sum_{h=n}^{k-1} a_h = \sum_{h=n}^{k-1} (h - n + 1)a_h - \sum_{h=n+1}^{k} (h - n)a_h + (k - n)a_k.$$

3.2.1 Auxiliary lemmas

Here we collect technical lemmas needed for our analysis, using the notation given in (3.2.1). For reader's convenience, we provide all the proofs in Appendix 3.7.

The following result appears in [65, page 357].

Lemma 3.6. *Let* $(\boldsymbol{x}^k)_{k\in\mathbb{N}}$ *be the sequence generated by Algorithm 3.1. We have*

$$(\forall\, k \in \mathbb{N}) \quad \boldsymbol{x}^k = \hat{\boldsymbol{x}}^k - \sum_{h\in J(k)} (\boldsymbol{x}^h - \boldsymbol{x}^{h+1}), \tag{3.2.8}$$

where $J(k) \subset \{k - \tau, \dots, k - 1\}$ *is a random set.*

The next lemma bounds the difference between the delayed and the current gradient in terms of the steps along the block coordinates, see [65, equation A.7].

Lemma 3.7. *Let* $(\boldsymbol{x}^k)_{k\in\mathbb{N}}$ *be the sequence generated by Algorithm 3.1. It follows*

$$(\forall\, k \in \mathbb{N}) \quad \|\nabla f(\boldsymbol{x}^k) - \nabla f(\hat{\boldsymbol{x}}^k)\| \leq L_{\mathrm{res}} \sum_{h\in J(k)} \|\boldsymbol{x}^{h+1} - \boldsymbol{x}^h\|.$$

Remark 3.8: Since $\|\cdot\|_{\mathsf{V}}^2 \leq \mathsf{p}_{\max}\|\cdot\|^2$ and $\|\cdot\|^2 \leq \mathsf{p}_{\min}^{-1}\|\cdot\|_{\mathsf{V}}^2$, Lemma 3.7 yields

$$\begin{aligned}\|\nabla f(\boldsymbol{x}^k) - \nabla f(\hat{\boldsymbol{x}}^k)\|_{\mathsf{V}} &\leq \sqrt{\mathsf{p}_{\max}}\|\nabla f(\boldsymbol{x}^k) - \nabla f(\hat{\boldsymbol{x}}^k)\| \\ &\leq L_{\mathrm{res}}\sqrt{\mathsf{p}_{\max}} \sum_{h\in J(k)} \|\boldsymbol{x}^{h+1} - \boldsymbol{x}^h\| \\ &\leq L_{\mathrm{res}}\frac{\sqrt{\mathsf{p}_{\max}}}{\sqrt{\mathsf{p}_{\min}}} \sum_{h\in J(k)} \|\boldsymbol{x}^{h+1} - \boldsymbol{x}^h\|_{\mathsf{V}}.\end{aligned}$$

We set $L_{\mathrm{res}}^{\mathsf{V}} = L_{\mathrm{res}}\frac{\sqrt{\mathsf{p}_{\max}}}{\sqrt{\mathsf{p}_{\min}}}$.

The result below yields a kind of inexact convexity inequality due to the presence of the delayed gradient vector. It is our variant of [65, Equation A.20]:

$$\begin{aligned}\frac{2\gamma}{L_{\max}}\mathsf{E}\left[\langle (P_{\operatorname{argmin} F}(\boldsymbol{x}^k) - \boldsymbol{x}^k)_{i_k}, \nabla_{i_k} f(\hat{\boldsymbol{x}}^k)\rangle\right] \leq \frac{1}{n}\mathsf{E}\left(f(P_{\operatorname{argmin} F}(\boldsymbol{x}^k)) - f(\boldsymbol{x}_k)\right) \\ + \frac{L_{\max}\tau\theta'}{2n^2}\mathsf{E}\|\boldsymbol{x}_k - \bar{\boldsymbol{x}}_{k+1}\|^2.\end{aligned}$$

Lemma 3.9. *Let $(\boldsymbol{x}^k)_{k\in\mathbb{N}}$ be a sequence generated by Algorithm 3.1. Then, for every $k \in \mathbb{N}$,*

$$(\forall\, \mathbf{x} \in \mathbf{H}) \quad \langle \nabla f(\hat{\boldsymbol{x}}^k), \mathbf{x} - \boldsymbol{x}^k\rangle \leq f(\mathbf{x}) - f(\boldsymbol{x}^k) + \frac{\tau L_{\mathrm{res}}}{2}\sum_{h\in J(k)} \|\boldsymbol{x}^h - \boldsymbol{x}^{h+1}\|^2.$$

The result below generalizes to the asynchronous case Lemma 4.3 in [90].

Lemma 3.10. *Let $\mathbf{H}$ be a real Hilbert space. Let $\varphi\colon \mathbf{H} \to \mathbb{R}$ be differentiable and convex, and $\psi\colon \mathbf{H} \to]-\infty, +\infty]$ be proper, lower semicontinuous and convex. Let $\mathbf{x}, \hat{\mathbf{x}} \in \mathbf{H}$ and set $\mathbf{x}^+ = \operatorname{prox}_{\psi}(\mathbf{x} - \nabla\varphi(\hat{\mathbf{x}}))$. Then, for every $\mathbf{z} \in \mathbf{H}$,*

$$\begin{aligned}\left\langle \mathbf{x} - \mathbf{x}^+, \mathbf{z} - \mathbf{x}\right\rangle \leq \psi(\mathbf{z}) - \psi(\mathbf{x}) + \langle \nabla\varphi(\hat{\mathbf{x}}), \mathbf{z} - \mathbf{x}\rangle \\ + \psi(\mathbf{x}) - \psi\left(\mathbf{x}^+\right) + \left\langle \nabla\varphi(\hat{\mathbf{x}}), \mathbf{x} - \mathbf{x}^+\right\rangle - \|\mathbf{x} - \mathbf{x}^+\|^2.\end{aligned}$$

3.3 Convergence analysis

In this section we assume just convexity of the objective function, and we provide worst case convergence rate as well as almost sure weak convergence of the iterates.

Throughout the section we set

$$\delta = \max_{i\in[m]}\left(L_i\gamma_i + 2\gamma_i\tau L_{\mathrm{res}}^{\mathsf{V}}\sqrt{\mathsf{p}_{\max}}\right) = \max_{i\in[m]}\left(L_i\gamma_i + 2\gamma_i\tau L_{\mathrm{res}}\frac{\mathsf{p}_{\max}}{\sqrt{\mathsf{p}_{\min}}}\right), \tag{3.3.1}$$

where the constants L_i's and L_{res} are defined in Assumption A3 and the constant $L_{\mathrm{res}}^{\mathsf{V}}$ is defined in Remark 3.8. The main convergence theorem is as follows.

Theorem 3.11

Let $(\boldsymbol{x}^k)_{k\in\mathbb{N}}$ be the sequence generated by Algorithm 3.1 and suppose that $\delta < 2$. Then the following hold.

(i) The sequence $(\boldsymbol{x}^k)_{k\in\mathbb{N}}$ weakly converges P-a.s. to a random variable that takes

values in argmin F.

(ii) $\mathsf{E}[F(\boldsymbol{x}^k)] - F_* = o(1/k)$. Furthermore, for every integer $k \geq 1$,

$$\mathsf{E}[F(\boldsymbol{x}^k)] - F_* \leq \frac{1}{k}\left(\frac{\mathrm{dist}_{\mathsf{W}}^2(\boldsymbol{x}^0, \operatorname{argmin} F)}{2} + C\left(F(\boldsymbol{x}^0) - F_*\right)\right),$$

where $C = \dfrac{\max\left\{1, (2-\delta)^{-1}\right\}}{\mathsf{p}_{\min}} - 1 + \tau\dfrac{1}{\sqrt{\mathsf{p}_{\min}}(2-\delta)}\left(1 + \dfrac{\mathsf{p}_{\max}}{\sqrt{\mathsf{p}_{\min}}}\right)$.

Remark 3.12:

(i) Theorem 3.11 extends classical results about the forward-backward algorithm to the asynchronous and stochastic block-coordinate setting. See [91] and reference therein. Moreover, we note that the above results, when specialized to the synchronous case, that is, $\tau = 0$, yield exactly [90, Theorem 4.9]. The $o(1/k)$ was also proven in [62].

(ii) The almost sure weak convergence of the iterates for the asynchronous stochastic forward-backward algorithm is new. In general only convergence in value is provided or, in the nonconvex case, cluster points of the sequence of the iterates are proven to be almost surely stationary points [19, 28].

(iii) As it can be readily seen from statement (ii) in Theorem 3.11, our results depend only on the maximum possible delay, and therefore apply in the same way to the consistent and inconsistent read model.

(iv) If we suppose that the random variables $(i_k)_{k\in\mathbb{N}}$ are uniformly distributed over $[m]$, the stepsize rule reduces to $\gamma_i < 2/(L_i + 2\tau L_{\mathrm{res}}/\sqrt{m})$, which agrees with that given in [28] and gets better when the number of blocks m increases. In this case, we see that the effect of the delay on the stepsize rule is mitigated by the number of blocks. In [19] the stepsize is not adapted to the blockwise Lipschitz constants L_i's, but it is chosen for each block as $\gamma < 2/(2L_f + \tau^2 L_f)$ with $L_f \geq L_{\mathrm{res}}$, leading, in general, to smaller stepsizes. In addition, this rule has a worse dependence on the delay τ and lacks of any dependence on the number of blocks.

(v) The framework of [19] is nonconvex and considers more general types of algorithms, in the flavor of majorization-minimization approaches [52]. On the other hand the assumptions are stronger (in particular, they assume F to be coercive) and the rate of convergence is given with respect to $\|\boldsymbol{x}^k - \mathsf{prox}_g(\boldsymbol{x}^k - \nabla f(\boldsymbol{x}^k))\|^2$, a quantity which is hard to relate to $F(\boldsymbol{x}^k) - F_*$. They also prove that the cluster points of the sequence of the iterates are almost surely stationary points.

(vi) The work [65] was among the first to study an asynchronous version of the randomized coordinate gradient descent method. There, the coordinates were selected at random with uniform probability and the stepsize was chosen the same for every coordinate. However, the stepsize was chosen to depend exponentially on τ, i.e. as $\mathcal{O}(1/\rho^\tau)$ with $\rho > 1$, which is much worse than our $\mathcal{O}(1/\tau)$. The same problem affects the constant in front of the bound of the rate of convergence which indeed is of the form $\mathcal{O}(\rho^\tau)$.

To circumvent these limitations above they put a condition in [65, Corollary 4.2] that bounds how big the maximum delay τ can be:

$$4e\Lambda(\tau+1)^2 \leq \sqrt{m}, \quad \Lambda = \frac{L_{\text{res}}}{L_{\max}}, \tag{3.3.2}$$

where m is the dimension of the space. However, this inequality is never satisfied if $\Lambda > \sqrt{m}/(4e)$, since this would imply

$$(\tau+1)^2 < 1,$$

contradicting the fact that τ is a non-negative integer. An example where this happens is when we are dealing with a quadratic function with positive semidefinite Hessian $\mathsf{Q} \in \mathbb{R}^{n\times n}$. In this case

$$L_{\text{res}} = \max_i \|\mathsf{Q}_{\cdot i}\|_2 \text{ and } L_{\max} = \max_i \|\mathsf{Q}_{\cdot i}\|_\infty \text{ with } \mathsf{Q}_{\cdot i} \text{ the } i\text{th column of } \mathsf{Q}.$$

Say one column of Q has constant entries equal to $p > 0$, while the absolute value of all the other entries of Q are less than p. Then,

$$\Lambda = \frac{p\sqrt{m}}{p} = \sqrt{m} > \frac{\sqrt{m}}{4e}.$$

In Section 3.6, we show two experiments on real datasets for which condition (3.3.2) is not verified.

Before giving the proof of Theorem 3.11, we present few preliminary results. The first one is a proposition showing that the function values are decreasing in expectation. The proof of this proposition, as well as those of the next intermediate results, are given in Appendix 3.8.

Proposition 3.13

Assume that $\delta < 2$ and let $(\boldsymbol{x}^k)_{k\in\mathbb{N}}$ be the sequence generated by Algorithm 3.1. Then, for every $k \in \mathbb{N}$,

$$\begin{aligned}(2-\delta)\frac{\mathsf{p}_{\min}}{2}\|\bar{\boldsymbol{x}}^{k+1} - \boldsymbol{x}^k\|^2_{\mathsf{\Gamma}^{-1}} \\ \leq F(\boldsymbol{x}^k) + \alpha_k - \mathsf{E}\big[F(\boldsymbol{x}^{k+1}) + \alpha_{k+1} \,\big|\, i_0, \ldots, i_{k-1}\big] \ \text{P-a.s.},\end{aligned} \tag{3.3.3}$$

where $\alpha_k = \dfrac{L^{\mathsf{V}}_{\text{res}}}{2\sqrt{\mathsf{p}_{\max}}} \displaystyle\sum_{h=k-\tau}^{k-1} (h-(k-\tau)+1)\|\boldsymbol{x}^{h+1} - \boldsymbol{x}^h\|^2_{\mathsf{V}}$.

Lemma 3.14. *Let $(\boldsymbol{x}^k)_{k\in\mathbb{N}}$ be the sequence generated by Algorithm 3.1. Then for every $k \in \mathbb{N}$, we have*

$$\begin{aligned}\langle \nabla f(\boldsymbol{x}^k) - \nabla f(\hat{\boldsymbol{x}}^k), \bar{\boldsymbol{x}}^{k+1} - \boldsymbol{x}^k\rangle_{\mathsf{V}} \\ \leq \tau L^{\mathsf{V}}_{\text{res}}\sqrt{\mathsf{p}_{\max}} \sum_{i=0}^{m} \mathsf{p}_i |\bar{x}_i^{k+1} - x_i^k|^2 + \alpha_k - \mathsf{E}\big[\alpha_{k+1} \,\big|\, i_0, \ldots, i_{k-1}\big],\end{aligned}$$

where α_k is defined in Proposition 3.13.

The next two results extend [90, Proposition 4.4, Proposition 4.5] to our more general

setting.

Lemma 3.15. *Let $(\boldsymbol{x}^k)_{k\in\mathbb{N}}$ be a sequence generated by Algorithm 3.1. Let $k \in \mathbb{N}$ and let $\boldsymbol{x}$ be an $\mathbf{H}$-valued random variable which is measurable w.r.t. $i_1, \dots, i_{k-1}$. Then,*

$$\mathsf{E}[\|\boldsymbol{x}^{k+1} - \boldsymbol{x}\|_{\mathsf{W}}^2 \,|\, i_0, \dots, i_{k-1}] - \|\boldsymbol{x}^k - \boldsymbol{x}\|_{\mathsf{W}}^2 = \|\bar{\boldsymbol{x}}^{k+1} - \boldsymbol{x}\|_{\mathsf{\Gamma}^{-1}}^2 - \|\boldsymbol{x}^k - \boldsymbol{x}\|_{\mathsf{\Gamma}^{-1}}^2 \quad (3.3.4)$$

and $\mathsf{E}[\|\boldsymbol{x}^{k+1} - \boldsymbol{x}^k\|_{\mathsf{W}}^2 \,|\, i_0, \dots, i_{k-1}] = \|\bar{\boldsymbol{x}}^{k+1} - \boldsymbol{x}^k\|_{\mathsf{\Gamma}^{-1}}^2$.

Proposition 3.16. *Let $(\boldsymbol{x}^k)_{k\in\mathbb{N}}$ be a sequence generated by Algorithm 3.1 and suppose that $\delta < 2$. Let $(\bar{\boldsymbol{x}}^k)_{k\in\mathbb{N}}$ and $(\alpha_k)_{k\in\mathbb{N}}$ be defined as in (3.2.1) and in Proposition 3.13 respectively. Then, for every $k \in \mathbb{N}$,*

$$\begin{aligned}(\forall \mathbf{x} \in \mathbf{H}) \quad \langle \boldsymbol{x}^k - \bar{\boldsymbol{x}}^{k+1}, \mathbf{x} - \boldsymbol{x}^k \rangle_{\mathsf{\Gamma}^{-1}} &\\ \leq \frac{1}{\mathsf{p}_{\min}} \Big(F(\boldsymbol{x}^k) + \alpha_k - \mathsf{E}\big[F(\boldsymbol{x}^{k+1}) + \alpha_{k+1} \,|\, i_0, \dots, i_{k-1}\big]\Big) &\\ + F(\mathbf{x}) - F(\boldsymbol{x}^k) + \frac{\tau L_{\mathrm{res}}}{2} \sum_{h \in J(k)} \|\boldsymbol{x}^h - \boldsymbol{x}^{h+1}\|^2 &\\ + \frac{\delta - 2}{2} \|\boldsymbol{x}^k - \bar{\boldsymbol{x}}^{k+1}\|_{\mathsf{\Gamma}^{-1}}^2. &\end{aligned}$$

Next we state a proposition that we will use throughout the rest of this chapter. It corresponds to [90, Proposition 4.6].

Proposition 3.17. *Let $(\boldsymbol{x}^k)_{k\in\mathbb{N}}$ be a sequence generated by Algorithm 3.1 and suppose that $\delta < 2$. Let $(\alpha_k)_{k\in\mathbb{N}}$ be defined as in Proposition 3.13. Then, for every $k \in \mathbb{N}$,*

$$\begin{aligned}(\forall \mathbf{x} \in \mathbf{H}) \quad \mathsf{E}\big[\|\boldsymbol{x}^{k+1} - \mathbf{x}\|_{\mathsf{W}}^2 \,|\, i_0, \dots, i_{k-1}\big] &\\ \leq \|\boldsymbol{x}^k - \mathbf{x}\|_{\mathsf{W}}^2 &\\ + \frac{2}{\mathsf{p}_{\min}} \left(\frac{(\delta - 1)_+}{2 - \delta} + 1 \right) \Big(F(\boldsymbol{x}^k) + \alpha_k &\\ - \mathsf{E}\big[F(\boldsymbol{x}^{k+1}) + \alpha_{k+1} \,|\, i_0, \dots, i_{k-1}\big] \Big) &\\ + \tau L_{\mathrm{res}} \sum_{h \in J(k)} \|\boldsymbol{x}^h - \boldsymbol{x}^{h+1}\|^2 &\\ + 2(F(\mathbf{x}) - F(\boldsymbol{x}^k)). \quad (3.3.5) &\end{aligned}$$

In the following, we show a general inequality from which we derive simultaneously the convergence of the iterates and the rate of convergence in expectation of the function values.

Proposition 3.18

Let $(\boldsymbol{x}^k)_{k\in\mathbb{N}}$ be a sequence generated by Algorithm 3.1 and suppose that $\delta < 2$. Let $(\alpha_k)_{k\in\mathbb{N}}$ be defined as in Proposition 3.13. Then, for all $\mathbf{x} \in \mathbf{H}$,

$$\mathsf{E}\big[\|\boldsymbol{x}^{k+1} - \mathbf{x}\|_{\mathsf{W}}^2 \,|\, i_0, \dots, i_{k-1}\big] \leq \|\boldsymbol{x}^k - \mathbf{x}\|_{\mathsf{W}}^2 + 2\big(F(\mathbf{x}) - \mathsf{E}\big[F(\boldsymbol{x}^{k+1}) + \alpha_{k+1} \,|\, i_0, \dots, i_{k-1}\big]\big) + \xi_k,$$

where $(\xi^k)_{k\in\mathbb{N}}$ is a sequence of positive random variables such that

$$\sum_{k\in\mathbb{N}} \mathsf{E}[\xi_k] \leq 2C(F(\boldsymbol{x}^0) - F_*), \quad (3.3.6)$$

with $C = \dfrac{\max\{1,(2-\delta)^{-1}\}}{\mathsf{p}_{\min}} - 1 + \dfrac{\tau}{\sqrt{\mathsf{p}_{\min}}(2-\delta)}\left(1+\dfrac{\mathsf{p}_{\max}}{\sqrt{\mathsf{p}_{\min}}}\right)$.

Proposition 3.19. *Let $(\boldsymbol{x}^k)_{k\in\mathbb{N}}$ be a sequence generated by Algorithm 3.1 and suppose that $\delta < 2$. Let $(\bar{\boldsymbol{x}}^k)_{k\in\mathbb{N}}$ be defined as in (3.2.1). Then there exists a sequence of $\mathbf{H}$-valued random variables $(\boldsymbol{v}^k)_{k\in\mathbb{N}}$ such that the following assertions hold:*

(i) $\forall\, k\in\mathbb{N}\colon\ \boldsymbol{v}^k \in \partial F(\bar{\boldsymbol{x}}^{k+1})$ P-*a.s.*

(ii) $\boldsymbol{v}^k \to 0$ *and* $\boldsymbol{x}^k - \bar{\boldsymbol{x}}^{k+1} \to 0$ P-*a.s., as* $k\to+\infty$.

We are now ready to prove the main theorem.

Proof of Theorem 3.11.

(i): It follows from Proposition 3.18 that

$$(\forall\, \mathbf{x}\in \operatorname{argmin} F)\quad \mathsf{E}\big[\|\boldsymbol{x}^{k+1}-\mathbf{x}\|_{\mathsf{W}}^2 \,|\, i_0,\dots,i_{k-1}\big] \le \|\boldsymbol{x}^k-\mathbf{x}\|_{\mathsf{W}}^2 + \xi_k,$$

where $(\xi_k)_{k\in\mathbb{N}}$ is a sequence of positive random variable which is P-a.s. summable. Thus, the sequence $(\boldsymbol{x}^k)_{k\in\mathbb{N}}$ is stochastic quasi-Fejér with respect to $\operatorname{argmin} F$ in the norm $\|\cdot\|_{\mathsf{W}}$ (which is equivalent to $\|\cdot\|$). Then according to Fact 3.2 it is bounded P-a.s. We now prove that $\operatorname{argmin} F$ contains the weak cluster points of $(\boldsymbol{x}^k)_{k\in\mathbb{N}}$ P-a.s. Indeed, let $\Omega_1\subset\Omega$ with $\mathsf{P}(\Omega\setminus\Omega_1)=0$ be such that items (i) and (ii) of Proposition 3.19 hold. Let $\omega\in\Omega_1$ and let $\mathbf{x}$ be a weak cluster point of $(\boldsymbol{x}^k(\omega))_{k\in\mathbb{N}}$. There exists a subsequence $(\boldsymbol{x}^{k_q}(\omega))_{q\in\mathbb{N}}$ which weakly converges to $\mathbf{x}$. By Proposition 3.19, we have $\bar{\boldsymbol{x}}^{k_q+1}(\omega)\rightharpoonup\mathbf{x}$, $\boldsymbol{v}^{k_q+1}(\omega)\to 0$, and $\boldsymbol{v}^{k_q+1}(\omega)\in\partial(f+g)(\bar{\boldsymbol{x}}^{k_q+1}(\omega))$. Thus, 1.10 ((iv)) (demiclosedness of the graph of the subgradient) yields $0\in\partial F(\mathbf{x})$ and hence $\mathbf{x}\in\operatorname{argmin} F$. Therefore, again by Fact 3.2 we conclude that the sequence $(\boldsymbol{x}^k)_{k\in\mathbb{N}}$ weakly converges to a random variable that takes value in $\operatorname{argmin} F$ P-a.s.

(ii): Choose $\mathbf{x}\in\operatorname{argmin} F$ in Proposition 3.18 and then take the expectation. Then we get

$$\mathsf{E}[F(\boldsymbol{x}^{k+1})+\alpha_{k+1}] - F_* \le \frac{1}{2}\big(\mathsf{E}[\|\boldsymbol{x}^k-\mathbf{x}\|_{\mathsf{W}}^2] - \mathsf{E}[\|\boldsymbol{x}^{k+1}-\mathbf{x}\|_{\mathsf{W}}^2]\big) + \frac{1}{2}\mathsf{E}[\xi_k].$$

Since $\sum_{k\in\mathbb{N}}(\mathsf{E}[\|\boldsymbol{x}^k-\mathbf{x}\|_{\mathsf{W}}^2] - \mathsf{E}[\|\boldsymbol{x}^{k+1}-\mathbf{x}\|_{\mathsf{W}}^2]) \le \|\boldsymbol{x}^0-\mathbf{x}\|_{\mathsf{W}}^2$, and recalling the bound on $\sum_{k\in\mathbb{N}}\mathsf{E}[\xi_k]$ in (3.3.6), we have

$$\sum_{k\in\mathbb{N}}\big(\mathsf{E}[F(\boldsymbol{x}^{k+1})+\alpha_{k+1}] - F_*\big) \le \frac{\|\boldsymbol{x}^0-\mathbf{x}\|_{\mathsf{W}}^2}{2} + C(F(\boldsymbol{x}^0)-F_*).$$

Thus, since, in virtue of equation 3.3.3, $(\mathsf{E}[F(\boldsymbol{x}^{k+1})+\alpha_{k+1}]-F_*)_{k\in\mathbb{N}}$ is decreasing, the statement follows from Fact 3.4, considering that $\alpha_k\ge 0$. □

3.4 Linear convergence under error bound condition

In the previous section we get a sublinear rate of convergence. Here we show that with an additional assumption we can get a better convergence rate. Also, we derive a strong convergence of the iterates, improving the weak convergence proved in Theorem 3.11.

We will assume that the following *Luo-Tseng error bound condition* [70] holds on a subset $\mathsf{X}\subset\mathbf{H}$ (containing the iterates $\boldsymbol{x}^k$).

$$(\forall \mathbf{x}\in\mathsf{X})\quad \operatorname{dist}_{\Gamma^{-1}}(\mathbf{x},\operatorname{argmin} F) \le C_{\mathsf{X},\Gamma^{-1}}\big\|\mathbf{x}-\operatorname{prox}_g^{\Gamma^{-1}}\big(\mathbf{x}-\nabla^{\Gamma^{-1}}f(\mathbf{x})\big)\big\|_{\Gamma^{-1}}. \tag{3.4.1}$$

Remark 3.20: We recall that the condition above is equivalent to the Kurdyka-Lojasiewicz property and the quadratic growth condition [16, 32, 90]. Any of these conditions can be used to prove linear convergence rates for various algorithms.

The following theorem is the main result of this section. Here, linear convergence of the function values and strong convergence of the iterates are ensured.

Theorem 3.21

Let $(\boldsymbol{x}^k)_{k\in\mathbb{N}}$ be generated by Algorithm 3.1 and suppose $\delta < 2$ and that the error bound condition (3.4.1) holds with $\mathsf{X} \supset \{\boldsymbol{x}^k \,|\, k \in \mathbb{N}\}$ P-a.s. for some $C_{\mathsf{X},\Gamma^{-1}} > 0$. Then for all $k \in \mathbb{N}$,

(i) $$\mathsf{E}\big[F(\boldsymbol{x}^{k+1}) - F_*\big] \leq \left(1 - \frac{\mathsf{p}_{\min}}{\kappa+\theta}\right)^{\lfloor \frac{k+1}{\tau+1} \rfloor} \mathsf{E}\big[F(\boldsymbol{x}^0) - F_*\big],$$

where

$$\kappa = 1 + \frac{(2C_{\mathsf{X},\Gamma^{-1}} + \delta - 2)_+}{2-\delta} = \max\left\{1, \frac{2C_{\mathsf{X},\Gamma^{-1}}}{2-\delta}\right\}$$
$$\theta = \frac{\tau L_{\mathrm{res}} \gamma_{\max}}{2-\delta}\left(\frac{\mathsf{p}_{\max}^2}{\sqrt{\mathsf{p}_{\min}}} + 1\right) \leq \frac{\sqrt{\mathsf{p}_{\min}}}{\mathsf{p}_{\max}(2-\delta)}\left(\frac{\mathsf{p}_{\max}^2}{\sqrt{\mathsf{p}_{\min}}} + 1\right).$$

(ii) The sequence $(\boldsymbol{x}^k)_{k\in\mathbb{N}}$ converges strongly P-a.s. to a random variable $\boldsymbol{x}_*$ that takes values in $\operatorname{argmin} F$ and $\mathsf{E}\big[\|\boldsymbol{x}^k - \boldsymbol{x}_*\|_{\Gamma^{-1}}\big] = \mathcal{O}\big((1 - \mathsf{p}_{\min}/(\kappa+\theta))^{\lfloor \frac{k}{\tau+1} \rfloor/2}\big)$.

Proof.

(i): From Proposition 3.16 we have

$$\begin{aligned}
&\frac{1}{\mathsf{p}_{\min}}\mathsf{E}\big[F(\boldsymbol{x}^{k+1}) + \alpha_{k+1} - F(\boldsymbol{x}^k) - \alpha_k \,|\, i_0, \dots, i_{k-1}\big] \\
&\quad\leq \|\boldsymbol{x}^k - \bar{\boldsymbol{x}}^{k+1}\|_{\Gamma^{-1}}\|\boldsymbol{x}^k - \mathbf{x}\|_{\Gamma^{-1}} \\
&\qquad + F(\mathbf{x}) - F(\boldsymbol{x}^k) + \frac{\tau L_{\mathrm{res}}}{2}\sum_{h\in J(k)} \|\boldsymbol{x}^h - \boldsymbol{x}^{h+1}\|^2 \\
&\qquad + \frac{\delta-2}{2}\|\boldsymbol{x}^k - \bar{\boldsymbol{x}}^{k+1}\|^2_{\Gamma^{-1}},
\end{aligned}$$

where $\alpha_k = (L_{\mathrm{res}}/(2\sqrt{\mathsf{p}_{\min}}))\sum_{h=k-\tau}^{k-1}(h-(k-\tau)+1)\|\boldsymbol{x}^{h+1} - \boldsymbol{x}^h\|^2_{\mathsf{V}}$. Now, taking $\mathbf{x} \in \operatorname{argmin} F$ and using the error bound condition 3.4.1 and equation 3.3.3, we obtain

$$\begin{aligned}
&\frac{1}{\mathsf{p}_{\min}}\mathsf{E}\big[F(\boldsymbol{x}^{k+1}) + \alpha_{k+1} - F(\boldsymbol{x}^k)) - \alpha_k \,|\, i_0, \dots, i_{k-1}\big] \\
&\quad\leq \left(C_{\mathsf{X},\Gamma^{-1}} + \frac{\delta-2}{2}\right)\|\boldsymbol{x}^k - \bar{\boldsymbol{x}}^{k+1}\|^2_{\Gamma^{-1}} \\
&\qquad - (F(\boldsymbol{x}^k) - F_*) + \frac{\tau L_{\mathrm{res}}}{2}\sum_{h=k-\tau}^{k-1} \|\boldsymbol{x}^h - \boldsymbol{x}^{h+1}\|^2 \\
&\quad\leq \frac{(2C_{\mathsf{X},\Gamma^{-1}} + \delta - 2)_+}{(2-\delta)\mathsf{p}_{\min}}\mathsf{E}\big[F(\boldsymbol{x}^k) + \alpha_k - F(\boldsymbol{x}^{k+1}) - \alpha_{k+1} \,|\, i_0, \dots, i_{k-1}\big] \\
&\qquad - (F(\boldsymbol{x}^k) - F_*) + \frac{\tau L_{\mathrm{res}}}{2}\sum_{h=k-\tau}^{k-1} \|\boldsymbol{x}^h - \bar{\boldsymbol{x}}^{h+1}\|^2, \qquad (3.4.2)
\end{aligned}$$

Adding and removing F_* in both expectation yield

$$\begin{aligned}\kappa\mathsf{E}\big[F(\boldsymbol{x}^{k+1}) + \alpha_{k+1} - F_* \,|\, i_0, \dots, i_{k-1}\big] \\ \leq \kappa\mathsf{E}\big[F(\boldsymbol{x}^{k}) + \alpha_{k} - F_* \,|\, i_0, \dots, i_{k-1}\big] \\ + \frac{\tau L_{\mathrm{res}}\gamma_{\max}\mathsf{p}_{\min}}{2}\sum_{h=k-\tau}^{k-1}\|\boldsymbol{x}^h - \bar{\boldsymbol{x}}^{h+1}\|^2_{\mathsf{\Gamma}^{-1}} \\ - \mathsf{p}_{\min}(F(\boldsymbol{x}^k) + \alpha_k - F_*) + \mathsf{p}_{\min}\alpha_k,\end{aligned} \tag{3.4.3}$$

where $\kappa = 1 + (2C_{\mathsf{X},\mathsf{\Gamma}^{-1}} + \delta - 2)_+/(2-\delta)$. Now, since $\|\cdot\|^2_{\mathsf{V}} \leq \gamma_{\max}\mathsf{p}^2_{\max}\|\cdot\|^2_{\mathsf{W}}$ we have

$$\begin{aligned}\mathsf{E}[\alpha_k] &\leq \frac{\tau L_{\mathrm{res}}\gamma_{\max}\mathsf{p}^2_{\max}}{2\sqrt{\mathsf{p}_{\min}}}\sum_{h=k-\tau}^{k-1}\mathsf{E}[\|\boldsymbol{x}^{h+1} - \boldsymbol{x}^h\|^2_{\mathsf{W}}] \\ &= \frac{\tau L_{\mathrm{res}}\gamma_{\max}\mathsf{p}^2_{\max}}{2\sqrt{\mathsf{p}_{\min}}}\sum_{h=k-\tau}^{k-1}\mathsf{E}[\|\bar{\boldsymbol{x}}^{h+1} - \boldsymbol{x}^h\|^2_{\mathsf{\Gamma}^{-1}}],\end{aligned} \tag{3.4.4}$$

where in the last equality we used Lemma 3.15. From (3.3.3), we have, for k such that $k - \tau \geq 0$,

$$\begin{aligned}&\sum_{h=k-\tau}^{k-1}\mathsf{E}[\|\bar{\boldsymbol{x}}^{h+1} - \boldsymbol{x}^h\|^2_{\mathsf{\Gamma}^{-1}}] \\ &\quad\leq \frac{2}{(2-\delta)\mathsf{p}_{\min}}\sum_{h=k-\tau}^{k-1}\mathsf{E}\big[F(\boldsymbol{x}^h) + \alpha_h\big] - \mathsf{E}\big[F(\boldsymbol{x}^{h+1}) + \alpha_{h+1}\big] \\ &\quad= \frac{2}{(2-\delta)\mathsf{p}_{\min}}\Big(\mathsf{E}\big[F(\boldsymbol{x}^{k-\tau}) + \alpha_{k-\tau}\big] - \mathsf{E}\big[F(\boldsymbol{x}^{k}) + \alpha_{k}\big]\Big) \\ &\quad\leq \frac{2}{(2-\delta)\mathsf{p}_{\min}}\Big(\mathsf{E}\big[F(\boldsymbol{x}^{k-\tau}) + \alpha_{k-\tau}\big] - \mathsf{E}\big[F(\boldsymbol{x}^{k+1}) + \alpha_{k+1}\big]\Big) \\ &\quad= \frac{2}{(2-\delta)\mathsf{p}_{\min}}\Big(\mathsf{E}\big[F(\boldsymbol{x}^{k-\tau}) + \alpha_{k-\tau} - F_*\big] - \mathsf{E}\big[F(\boldsymbol{x}^{k+1}) + \alpha_{k+1} - F_*\big]\Big).\end{aligned} \tag{3.4.5}$$

Because the sequence $\big(\mathsf{E}\big[F(\boldsymbol{x}^k) + \alpha_k\big]\big)_{k\in\mathbb{N}}$ is decreasing, the transition from the second line to the third one is allowed. Using (3.4.4) and (3.4.5) in (3.4.3) with total expectation, we obtain

$$\begin{aligned}(\kappa+\theta)\mathsf{E}\big[F(\boldsymbol{x}^{k+1}) + \alpha_{k+1} - F_*\big] &\leq (\kappa - \mathsf{p}_{\min})\mathsf{E}\big[F(\boldsymbol{x}^k) + \alpha_k - F_*\big] \\ &\quad + \theta\mathsf{E}\big[F(\boldsymbol{x}^{k-\tau}) + \alpha_{k-\tau} - F_*\big] \\ &\leq (\kappa - \mathsf{p}_{\min})\mathsf{E}\big[F(\boldsymbol{x}^{k-\tau}) + \alpha_{k-\tau} - F_*\big] \\ &\quad + \theta\mathsf{E}\big[F(\boldsymbol{x}^{k-\tau}) + \alpha_{k-\tau} - F_*\big] \\ &= (\kappa + \theta - \mathsf{p}_{\min})\mathsf{E}\big[F(\boldsymbol{x}^{k-\tau}) + \alpha_{k-\tau} - F_*\big],\end{aligned} \tag{3.4.6}$$

where

$$\begin{aligned}\theta &= (2-\delta)^{-1}\left(\frac{\tau L_{\mathrm{res}}\gamma_{\max}\mathsf{p}^2_{\max}}{\sqrt{\mathsf{p}_{\min}}} + \tau L_{\mathrm{res}}\gamma_{\max}\right) \\ &= \tau L_{\mathrm{res}}\gamma_{\max}(2-\delta)^{-1}\left(\frac{\mathsf{p}^2_{\max}}{\sqrt{\mathsf{p}_{\min}}} + 1\right).\end{aligned}$$

That means

$$\begin{aligned}\mathsf{E}\big[F(\boldsymbol{x}^{k+1})+\alpha_{k+1}-F_*\big] &\le \left(1-\frac{\mathsf{p}_{\min}}{\kappa+\theta}\right)\mathsf{E}\big[F(\boldsymbol{x}^{k-\tau})+\alpha_{k-\tau}-F_*\big]\\ &\le \left(1-\frac{\mathsf{p}_{\min}}{\kappa+\theta}\right)^{\lfloor\frac{k+1}{\tau+1}\rfloor}\mathsf{E}\big[F(\boldsymbol{x}^{0})+\alpha_{0}-F_*\big]. \end{aligned}\tag{3.4.7}$$

Now for $k<\tau$, $\lfloor\frac{k+1}{\tau+1}\rfloor=0$. Because $\big(\mathsf{E}\big[F(\boldsymbol{x}^k)+\alpha_k\big]\big)_{k\in\mathbb{N}}$ is decreasing, we know that

$$\begin{aligned}\mathsf{E}\big[F(\boldsymbol{x}^{k+1})+\alpha_{k+1}-F_*\big] &\le \mathsf{E}\big[F(\boldsymbol{x}^{0})+\alpha_{0}-F_*\big]\\ &= \left(1-\frac{\mathsf{p}_{\min}}{\kappa+\theta}\right)^{\lfloor\frac{k+1}{\tau+1}\rfloor}\mathsf{E}\big[F(\boldsymbol{x}^{0})+\alpha_{0}-F_*\big].\end{aligned}$$

So (3.4.7) remains true. Also, from (3.8.10), we have

$$\theta \le \frac{\sqrt{\mathsf{p}_{\min}}}{\mathsf{p}_{\max}}(2-\delta)^{-1}\left(\frac{\mathsf{p}_{\max}^2}{\sqrt{\mathsf{p}_{\min}}}+1\right).$$

(ii): From Jensen inequality, (3.3.3) and (3.4.7), we have

$$\begin{aligned}\mathsf{E}\big[\|\boldsymbol{x}^{k+1}-\boldsymbol{x}^k\|_{\Gamma^{-1}}\big] &\le \sqrt{\mathsf{E}\big[\|\boldsymbol{x}^{k+1}-\boldsymbol{x}^k\|^2_{\Gamma^{-1}}\big]}\\ &\le \sqrt{\mathsf{E}\big[\|\bar{\boldsymbol{x}}^{k+1}-\boldsymbol{x}^k\|^2_{\Gamma^{-1}}\big]}\\ &\le \sqrt{\frac{2}{\mathsf{p}_{\min}(2-\delta)}\mathsf{E}\big[F(\boldsymbol{x}^k)+\alpha_k-F_*\big]}\\ &\le \sqrt{\frac{2}{\mathsf{p}_{\min}(2-\delta)}\left(1-\frac{\mathsf{p}_{\min}}{\kappa+\theta}\right)^{\lfloor\frac{k}{\tau+1}\rfloor}\mathsf{E}\big[F(\boldsymbol{x}^{0})+\alpha_{0}-F_*\big]}.\end{aligned}\tag{3.4.8}$$

Since $1-\mathsf{p}_{\min}/(\kappa+\theta)<1$,

$$\mathsf{E}\left[\sum_{k\in\mathbb{N}}\|\boldsymbol{x}^{k+1}-\boldsymbol{x}^k\|_{\Gamma^{-1}}\right]=\sum_{k\in\mathbb{N}}\mathsf{E}\big[\|\boldsymbol{x}^{k+1}-\boldsymbol{x}^k\|_{\Gamma^{-1}}\big]<\infty.$$

Therefore, $\sum_{k\in\mathbb{N}}\|\boldsymbol{x}^{k+1}-\boldsymbol{x}^k\|_{\Gamma^{-1}}<\infty$ P-a.s. This means the sequence $(\boldsymbol{x}^k)_{k\in\mathbb{N}}$ is a Cauchy sequence P-a.s. By Theorem 3.11 (i), this sequence has accumulation points that take values in $\operatorname{argmin}F$. So it converges strongly P-a.s. to a random variable that takes values in $\operatorname{argmin}F$.

Now let $\rho=1-\mathsf{p}_{\min}/(\kappa+\theta)$. For all $n\in\mathbb{N}$,

$$\|\boldsymbol{x}^{k+n}-\boldsymbol{x}^k\|_{\Gamma^{-1}}\le\sum_{i=0}^{n-1}\|\boldsymbol{x}^{k+i+1}-\boldsymbol{x}^{k+i}\|_{\Gamma^{-1}}\le\sum_{i=0}^{\infty}\|\boldsymbol{x}^{k+i+1}-\boldsymbol{x}^{k+i}\|_{\Gamma^{-1}}.$$

Letting $n\to\infty$ and using (3.4.8), we get

$$\begin{aligned}\mathsf{E}\big[\|\boldsymbol{x}^{k}-\boldsymbol{x}_*\|_{\Gamma^{-1}}\big] &\le \left(\frac{2}{\mathsf{p}_{\min}(2-\delta)}\mathsf{E}\big[F(\boldsymbol{x}^{0})+\alpha_{0}-F_*\big]\right)^{1/2}\sum_{i=0}^{\infty}\rho^{\lfloor\frac{k+i}{\tau+1}\rfloor/2}\\ &\le \left(\frac{2}{\mathsf{p}_{\min}(2-\delta)}\mathsf{E}\big[F(\boldsymbol{x}^{0})+\alpha_{0}-F_*\big]\right)^{1/2}\rho^{\lfloor\frac{k}{\tau+1}\rfloor/2}\sum_{i=0}^{\infty}\rho^{\lfloor\frac{i}{\tau+1}\rfloor/2}\\ &= \rho^{\lfloor\frac{k}{\tau+1}\rfloor/2}\left(\frac{2}{\mathsf{p}_{\min}(2-\delta)}\mathsf{E}\big[F(\boldsymbol{x}^{0})+\alpha_{0}-F_*\big]\right)^{1/2}\frac{\tau+1}{1-\rho^{1/2}}.\end{aligned}$$

□

Remark 3.22:

(i) A linear convergence rate is also given in [65, Theorem 4.1] by assuming a quadratic growth condition instead of the error bound condition (3.4.1). Their rate depend on the stepsize which in general can be very small, as explained earlier in point (vi) of Remark 3.12.

(ii) The error bound condition (3.4.1) is sometimes satisfied globally, meaning on $\mathsf{X} = \operatorname{dom} F$, so that the condition $\mathsf{X} \supset \{\boldsymbol{x}^k \,|\, k \in \mathbb{N}\}$ P-a.s. required in Theorem 3.21 is clearly fulfilled. This is the case when F is strongly convex or when f is quadratic and g is the indicator function of a polytope (see Remark 4.17(iv) in [90]). More often, for general convex objectives, the error bound condition (3.4.1) is satisfied on sublevel sets of F (see [90, Remark 4.18]). Therefore, it is important to find conditions ensuring that the sequence $(\boldsymbol{x}^k)_{k\in\mathbb{N}}$ remains in a sublevel set. The next results address this issue.

We first give an analogue of Lemma 3.14.

Lemma 3.23. *Let $(\boldsymbol{x}^k)_{k\in\mathbb{N}}$ be the sequence generated by Algorithm 3.1. Then, for every $k \in \mathbb{N}$,*

$$\langle \nabla f(\boldsymbol{x}^k) - \nabla f(\hat{\boldsymbol{x}}^k), x^{k+1} - x^k \rangle \leq \tau L_{\mathrm{res}} \|\boldsymbol{x}^{k+1} - \boldsymbol{x}^k\|^2 + \tilde{\alpha}_k - \tilde{\alpha}_{k+1},$$

with $\tilde{\alpha}_k = (L_{\mathrm{res}}/2) \sum_{h=k-\tau}^{k-1} (h - (k-\tau) + 1) \|\boldsymbol{x}^{h+1} - \boldsymbol{x}^h\|^2$.

Proof. Let $k \in \mathbb{N}$. We have, from Cauchy-Schwarz inequality, the Young inequality and Lemma 3.2.5, that

$$\begin{aligned}
&\langle \nabla f(\boldsymbol{x}^k) - \nabla f(\hat{\boldsymbol{x}}^k), \boldsymbol{x}^{k+1} - \boldsymbol{x}^k \rangle \\
&\qquad \leq L_{\mathrm{res}} \sum_{h\in J(k)} \|\boldsymbol{x}^{h+1} - \boldsymbol{x}^h\| \|\boldsymbol{x}^{k+1} - \boldsymbol{x}^k\| \\
&\qquad \leq \frac{1}{2} \left[\frac{L_{\mathrm{res}}^2}{s} \left(\sum_{h\in J(k)} \|\boldsymbol{x}^{h+1} - \boldsymbol{x}^h\| \right)^2 + s \|\boldsymbol{x}^{k+1} - \boldsymbol{x}^k\|^2 \right] \\
&\qquad \leq \frac{1}{2} \left[\frac{\tau L_{\mathrm{res}}^2}{s} \left(\sum_{h=k-\tau}^{k-1} \|\boldsymbol{x}^{h+1} - \boldsymbol{x}^h\|^2 \right) + s \|\boldsymbol{x}^{k+1} - \boldsymbol{x}^k\|^2 \right] \\
&\qquad = \frac{s}{2} \|\boldsymbol{x}^{k+1} - \boldsymbol{x}^k\|^2 + \frac{\tau L_{\mathrm{res}}^2}{2s} \sum_{h=k-\tau}^{k-1} \|\boldsymbol{x}^{h+1} - \boldsymbol{x}^h\|^2.
\end{aligned}$$

Using the same decomposition of the last term as in Lemma 3.14, we get

$$\begin{aligned}
&\langle \nabla f(\boldsymbol{x}^k) - \nabla f(\hat{\boldsymbol{x}}^k), \boldsymbol{x}^{k+1} - \boldsymbol{x}^k \rangle \\
&\qquad \leq \frac{s}{2} \|\boldsymbol{x}^{k+1} - \boldsymbol{x}^k\|^2 + \frac{\tau L_{\mathrm{res}}^2}{2s} \sum_{h=k-\tau}^{k-1} (h - (k-\tau) + 1) \|\boldsymbol{x}^{h+1} - \boldsymbol{x}^h\|^2 \\
&\qquad\quad - \frac{\tau L_{\mathrm{res}}^2}{2s} \sum_{h=k-\tau+1}^{k} (h - (k-\tau)) \|\boldsymbol{x}^{h+1} - \boldsymbol{x}^h\|^2 \\
&\qquad\quad + \frac{\tau^2 L_{\mathrm{res}}^2}{2s} \|\boldsymbol{x}^{k+1} - \boldsymbol{x}^k\|^2.
\end{aligned}$$

So taking

$$\tilde{\alpha}_k = \frac{\tau L_{\text{res}}^2}{2s} \sum_{h=k-\tau}^{k-1} (h-(k-\tau)+1)\|\boldsymbol{x}^{h+1}-\boldsymbol{x}^h\|^2,$$

we get

$$\langle \nabla f(\boldsymbol{x}^k) - \nabla f(\hat{\boldsymbol{x}}^k), \bar{x}^{k+1} - x^k\rangle \leq \left(\frac{s}{2} + \frac{\tau^2 L_{\text{res}}^2}{2s}\right)\|\bar{\boldsymbol{x}}^{k+1}-\boldsymbol{x}^k\|^2 + \tilde{\alpha}_k - \tilde{\alpha}_{k+1}.$$

By minimizing $s \mapsto (s/2 + \tau^2 L_{\text{res}}^2/(2s))$, we find $s = \tau L_{\text{res}}$. We then obtain

$$\langle \nabla f(\boldsymbol{x}^k) - \nabla f(\hat{\boldsymbol{x}}^k), x^{k+1} - x^k\rangle \leq \tau L_{\text{res}}\|\boldsymbol{x}^{k+1}-\boldsymbol{x}^k\|^2 + \tilde{\alpha}_k - \tilde{\alpha}_{k+1},$$

and the statement follows. □

Proposition 3.24

Let $(\boldsymbol{x}^k)_{k\in\mathbb{N}}$ be the sequence generated by Algorithm 3.1. Then, for every $k \in \mathbb{N}$,

$$\left(\frac{1}{\gamma_{i_k}} - \frac{L_{i_k}}{2} - \tau L_{\text{res}}\right)\|\boldsymbol{x}^{k+1}-\boldsymbol{x}^k\|^2 \leq F(\boldsymbol{x}^k)+\tilde{\alpha}_k-\big(F(\boldsymbol{x}^{k+1})+\tilde{\alpha}_{k+1}\big) \qquad \text{P-a.s.,} \quad (3.4.9)$$

where $\tilde{\alpha}_k = (L_{\text{res}}/2)\sum_{h=k-\tau}^{k-1}(h-(k-\tau)+1)\|\boldsymbol{x}^{h+1}-\boldsymbol{x}^h\|^2$.

Proof. Using Lemma 3.23 in equation (3.8.3), we have

$$\begin{aligned}
F(\boldsymbol{x}^{k+1}) &\leq F(\boldsymbol{x}^k) + \langle \nabla_{i_k} f(\boldsymbol{x}^k) - \nabla_{i_k} f(\hat{\boldsymbol{x}}^k), \bar{x}_{i_k}^{k+1} - x_{i_k}^k\rangle \\
&\quad - \left(\frac{1}{\gamma_{i_k}} - \frac{L_{i_k}}{2}\right)|\bar{x}_{i_k}^{k+1} - x_{i_k}^k|^2 \\
&= F(\boldsymbol{x}^k) + \langle \nabla f(\boldsymbol{x}^k) - \nabla f(\hat{\boldsymbol{x}}^k), \boldsymbol{x}^{k+1} - \boldsymbol{x}^k\rangle \\
&\quad - \left(\frac{1}{\gamma_{i_k}} - \frac{L_{i_k}}{2}\right)\|\boldsymbol{x}^{k+1} - \boldsymbol{x}^k\|^2 \\
&\leq F(\boldsymbol{x}^k) + \tilde{\alpha}_k - \tilde{\alpha}_{k+1} - \left(\frac{1}{\gamma_{i_k}} - \frac{L_{i_k}}{2} - \tau L_{\text{res}}\right)\|\boldsymbol{x}^{k+1} - \boldsymbol{x}^k\|^2.
\end{aligned}$$

So the statement follows. □

Corollary 3.25: Let $(\boldsymbol{x}^k)_{k\in\mathbb{N}}$ be generated by Algorithm 3.1 with the γ_i's satisfying the following stepsize rule

$$(\forall\, i \in [m]) \quad \gamma_i < \frac{2}{L_i + 2\tau L_{\text{res}}}. \qquad (3.4.10)$$

Then

$$(\forall\, k \in \mathbb{N}) \quad F(\boldsymbol{x}^k) \leq F(\mathbf{x}^0) \quad \text{P-a.s.} \qquad (3.4.11)$$

So if the error bound condition (3.4.1) holds on the sublevel set $\mathsf{X} = \{F \leq F(\mathbf{x}^0)\}$, then the assumptions of Theorem 3.21 are met.

Proof. The left-hand side in (3.4.9) is positive and hence $(F(\boldsymbol{x}^k) + \tilde{\alpha}_k)_{k\in\mathbb{N}}$ is decreasing P-a.s. Therefore, we have, for every $k \in \mathbb{N}$

$$F(\boldsymbol{x}^k) \leq F(\boldsymbol{x}^k) + \tilde{\alpha}_k \leq F(\boldsymbol{x}^0) + \tilde{\alpha}_0 = F(\mathbf{x}^0).$$ □

Remark 3.26: The rule (3.4.10) yields stepsizes possibly smaller than the ones given in Theorem 3.11, which requires $\gamma_i < 2/(L_i + 2\tau L_{\text{res}}\mathsf{p}_{\max}/\sqrt{\mathsf{p}_{\min}})$. Indeed, this happens when $\mathsf{p}_{\max}/\sqrt{\mathsf{p}_{\min}} < 1$. For instance if the distribution is uniform, we have $\mathsf{p}_{\max}/\sqrt{\mathsf{p}_{\min}} = 1/\sqrt{m} < 1$ whenever $m \geq 2$. On the bright side, there exists distributions for which $\mathsf{p}_{\max}/\sqrt{\mathsf{p}_{\min}} > 1$. For example, in the case of two coordinates, if the selection probability follows a Bernoulli distribution with $p = 0.7$. In that case, $\mathsf{p}_{\max}/\sqrt{\mathsf{p}_{\min}} = 0.7/\sqrt{0.3} > 1.278 > 1$. When $m > 2$, another example is the multinoulli distribution with $\mathsf{p}_{\max} > 0.1$, $\mathsf{p}_{\min} = 0.01$ and $\sum_{i=1}^{m} \mathsf{p}_i = 1$.

3.5 Applications

Here we present two problems where Algorithm 3.1 can be useful.

3.5.1 The Lasso problem

We start with the Lasso problem [103], also known as basis pursuit [21]. It is a least-squares regression problem with an ℓ_1 regularizer which favors sparse solutions. More precisely, given $\mathsf{A} \in \mathbb{R}^{n\times m}$ and $\mathsf{b} \in \mathbb{R}^n$, one aims at solving the following problem

$$\underset{\mathsf{x}\in\mathbb{R}^m}{\text{minimize}}\ \frac{1}{2}\|\mathsf{A}\mathsf{x} - \mathsf{b}\|_2^2 + \lambda\|\mathsf{x}\|_1 \qquad (\lambda > 0)\,. \tag{3.5.1}$$

We clearly fall in the framework of problem (3.1.1) with $f(\mathsf{x}) = (1/2)\|\mathsf{A}\mathsf{x} - \mathsf{b}\|_2^2$ and $g_i(\mathsf{x}_i) = \lambda|\mathsf{x}_i|$. The assumptions A1, A2, A3 and A4 are also satisfied. In particular, here $L_i = \|a_i\|^2$, where a_i is the i-th column of A, $L_{\text{res}} = \max_i \|\mathsf{A}^\intercal\mathsf{A}_{\cdot i}\|_2$, with $\mathsf{A}^\intercal\mathsf{A}_{\cdot i}$ the i-th column of $\mathsf{A}^\intercal\mathsf{A}$, and $F = f + g$ attains its minimum.

The Lasso technique is used in many fields, especially for high-dimensional problems – among others it is worth mentioning statistics, signal processing, and inverse problems; see [9, 11, 31, 58, 98, 107] and references therein. Since there is no closed form solution for this problem, many iterative algorithms have been proposed to solve it: forward-backward, accelerated (proximal) gradient descent, (proximal) block coordinate descent, etc. [9, 27, 39, 40, 76, 109]. In the same vein, applying Algorithm 3.1 to the Lasso problem (3.5.1) yields the iterative scheme:

$$\begin{array}{l} \text{for } n = 0, 1, \ldots \\ \quad \text{for } i = 1, \ldots, m \\ \quad\quad x_i^{k+1} = \begin{cases} \mathsf{soft}_{\lambda\gamma_i}\big(x_i^k - \gamma_i a_i^\intercal(\mathsf{A}\boldsymbol{x}^{k-\mathbf{d}^k} - \mathsf{b})\big) & \text{if } i = i_k \\ x_i^k & \text{if } i \neq i_k, \end{cases} \end{array} \tag{3.5.2}$$

where, for every $\rho > 0$, $\mathsf{soft}_\rho \colon \mathbb{R} \to \mathbb{R}$ is the soft thresholding operator (with threshold ρ) [91]. Thanks to Theorem 3.11 we know that the iterates $(\boldsymbol{x}^k)_{k\in\mathbb{N}}$ generated are weakly convergent and the function values have a convergence rate of $o(1/k)$. On top of that the cost function of the Lasso problem (3.5.1) satisfies the error bound condition (3.4.1) on its sublevel sets [108, Theorem 2]. So, following Corollary 3.25 and Theorem 3.21, the iterates converge strongly (a.s.) and linearly in mean, whenever $\gamma_i < 2/\left(L_i + 2\tau L_{\text{res}}\right)$, for all $i \in [m]$.

3.5.2 Linear convergence of dual proximal gradient method

We consider the problem

$$\underset{\mathsf{x}\in\mathsf{H}}{\text{minimize}}\ \sum_{i=1}^{m} \phi_i\left(\mathsf{A}_i\mathsf{x}\right) + h(\mathsf{x}), \tag{3.5.3}$$

where, for all $i \in [m]$, $\mathsf{A}_i \colon \mathsf{H} \to \mathsf{G}_i$ is a linear operator between Hilbert spaces, $\phi_i \colon \mathsf{G}_i \to]-\infty, +\infty]$ is proper convex and lower semicontinuous, and $h \colon \mathsf{H} \to]-\infty, +\infty]$ is proper lower semicontinuous and σ-strongly convex ($\sigma > 0$). The first term of the objective function may represent the empirical data loss and the second term the regularizer. This problem arises in many applications in machine learning, signal processing and statistical estimation, and is commonly called regularized empirical risk minimization [95]. It includes, for instance, ridge regression and (soft margin) support vector machines [95], more generally Tikhonov regularization [61, Section 5.3].

In the following we apply Algorithm 3.1 to the dual of problem (3.5.3). Below we provide details. Set $\mathbf{G} = \bigoplus_{i=1}^m \mathsf{G}_i$ and $\mathbf{u} = (\mathsf{u}_1, \mathsf{u}_2, \dots, \mathsf{u}_m)$. Then, the dual of problem (3.5.3) is

$$\operatorname*{minimize}_{\mathbf{u} \in \mathbf{G}} F(\mathbf{u}) = h^*\Big(-\sum_{i=1}^m \mathsf{A}_i^* \mathsf{u}_i \Big) + \sum_{i=1}^m \phi_i^*(\mathsf{u}_i), \tag{3.5.4}$$

where, A_i^* is the adjoint operator of A_i h^* and ϕ_i^* are the Fenchel conjugates of h and ϕ_i respectively. The link between the dual variable $\mathbf{u}$ and the primal variable x is given by the rule $\mathbf{u} \mapsto \nabla h^*(-\sum_{i=1}^m \mathsf{A}_i^* \mathsf{u}_i)$. Since h^* is $(1/\sigma)$-Lipschitz smooth, see Proposition 1.19, the dual problem above is in the form of problem (3.1.1). Thus, Algorithm 3.1 applied to the dual problem (3.5.4) gives

$$\begin{array}{l} \text{for } k = 0, 1, \dots \\ \quad \text{for } i = 1, \dots, m \\ \quad\quad u_i^{k+1} = \begin{cases} \mathsf{prox}_{\gamma_{i_k} \phi_{i_k}^*}\big(u_{i_k}^k + \gamma_{i_k} \mathsf{A}_{i_k} \nabla h^*(-\sum_{j=1}^m \mathsf{A}_j^* u_j^{k - \mathsf{d}_j^k})\big) & \text{if } i = i_k \\ u_i^k & \text{if } i \neq i_k, \end{cases} \end{array} \tag{3.5.5}$$

Suppose that $\nabla h^* = \mathsf{B}$ is a linear operator and that the delay vector $\mathbf{d}^k = (\mathsf{d}_1^k, \cdots, \mathsf{d}_m^k)$ is uniform, that is, $\mathsf{d}_i^k = \mathsf{d}_j^k = \mathsf{d}^k \in \mathbb{N}$. Then, using the primal variable, the KKT condition $x^k = \nabla h^*(-\sum_{j=1}^m \mathsf{A}_j^* u_j^k) = -\sum_{j=1}^m \mathsf{B}\mathsf{A}_j^* u_j^k$, and the fact that $\boldsymbol{u}^{k+1}$ and $\boldsymbol{u}^k$ differ only on the i_k-component, the algorithm becomes

$$\begin{array}{l} \text{for } k = 0, 1, \dots \\ \quad \text{for } i = 1, \dots, m \\ \quad\quad u_i^{k+1} = \begin{cases} \mathsf{prox}_{\gamma_{i_k} \phi_{i_k}^*}\big(u_{i_k}^k + \gamma_{i_k} \mathsf{A}_{i_k} \boldsymbol{x}^{k - \mathsf{d}^k}\big) & \text{if } i = i_k \\ u_i^k & \text{if } i \neq i_k. \end{cases} \\ \\ \quad\quad \boldsymbol{x}^{k+1} = \boldsymbol{x}^k - \mathsf{B}\mathsf{A}_{i_k}^*(u_{i_k}^{k+1} - u_{i_k}^k). \end{array} \tag{3.5.6}$$

The above algorithm requires a lock during the update of the primal variable x. On the contrary, the update of the dual variable $\mathbf{u}$ is completely asynchronous without any lock as in the setting we studied in this chapter. To get a better understanding of this aspect, we will expose a concrete example: the ridge regression.

Example: Ridge regression

The ridge regression is the following regularized least squares problem.

$$\operatorname*{minimize}_{\mathsf{w} \in \mathsf{H}} \frac{1}{\lambda m} \sum_{i=1}^m (\mathsf{y}_i - \langle \mathsf{w}, \mathsf{x}_i \rangle)^2 + \frac{1}{2} \|\mathsf{w}\|^2. \tag{3.5.7}$$

Its dual problem is

$$\operatorname*{minimize}_{\mathbf{u} \in \mathbb{R}^m} \frac{1}{2} \langle (\mathsf{K} + \lambda m \mathsf{Id}_m)\mathbf{u}, \mathbf{u} \rangle - \langle \mathbf{y}, \mathbf{u} \rangle,$$

where $\mathsf{K} = \mathsf{X}\mathsf{X}^*$ and $\mathsf{X}\colon \mathsf{H} \to \mathbb{R}^m$, with $\mathsf{X}\mathsf{w} = (\langle \mathsf{w}, \mathsf{x}_i \rangle)_{1 \leq i \leq m}$. We remark that, in this situation, $A_i = \langle \cdot, \mathsf{x}_i \rangle$, $A_i^* = \mathsf{x}_i$ and $B = \mathsf{Id}$. Let $\mathbf{d}^k = (\mathsf{d}^k, \mathsf{d}^k, \cdots, \mathsf{d}^k)$. With $\mathsf{w}^k = \mathsf{X}^* \mathbf{u}^k$ and considering that the non smooth part g is null, the algorithm is given by

$$
\begin{array}{l}
\text{for } k = 0, 1, \ldots \\
\quad \text{for } i = 1, \ldots, m \\
\quad\quad u_i^{k+1} = \begin{cases} u_{i_k}^k - \gamma_{i_k} \big(\langle \mathsf{x}_{i_k}, \boldsymbol{w}^{k - \mathsf{d}^k} \rangle + \lambda m u_{i_k}^{k - \mathsf{d}^k} - \mathsf{y}_{i_k} \big) & \text{if } i = i_k \\ u_i^k & \text{if } i \neq i_k. \end{cases} \\
\quad\quad \boldsymbol{w}^{k+1} = \boldsymbol{w}^k - \gamma_{i_k} \mathsf{x}_{i_k} \big(\mathsf{u}_{i_k}^{k+1} - \mathsf{u}_{i_k}^k \big).
\end{array}
\tag{3.5.8}
$$

Remark 3.27: Now we will compare the above dual asynchronous algorithm to the asynchronous stochastic gradient descent (ASGD) [1, 87]. We note that (3.5.8) yields

$$
\begin{aligned}
\boldsymbol{w}^{k+1} &= \boldsymbol{w}^k - \gamma_{i_k} \mathsf{x}_{i_k} \big(u_{i_k}^{k+1} - u_{i_k}^k \big) \\
&= \boldsymbol{w}^k - \gamma_{i_k} \big(\langle \mathsf{x}_{i_k}, \boldsymbol{w}^{k - \mathsf{d}^k} \rangle \mathsf{x}_{i_k} + \lambda m u_{i_k}^{k - \mathsf{d}^k} \mathsf{x}_{i_k} - \mathsf{y}_{i_k} \mathsf{x}_{i_k} \big).
\end{aligned}
$$

Instead, applying asynchronous SGD to the primal problem (3.5.7) multiply by λm, we get

$$
\boldsymbol{w}^{k+1} = \boldsymbol{w}^k - \gamma_k' \big(\langle \mathsf{x}_{i_k}, \boldsymbol{w}^{k - \mathsf{d}^k} \rangle \mathsf{x}_{i_k} + \lambda m \boldsymbol{w}^{k - \mathsf{d}^k} - \mathsf{y}_{i_k} \mathsf{x}_{i_k} \big).
$$

We see that the only difference is the second term inside the parentheses in both updates. Indeed, the term $\boldsymbol{w}^{k - \mathsf{d}^k} = \mathsf{X}^* \boldsymbol{u}^{k - \mathsf{d}^k} = \sum_{i=1}^m u_i^{k - \mathsf{d}^k} \mathsf{x}_i$ in ASGD is replaced by only one summand $u_{i_k}^{k - \mathsf{d}^k} \mathsf{x}_{i_k}$ in our algorithm. However, a major difference between the two approaches lies in the way the stepsize is set. Indeed, in ASGD, the stepsize γ_k' is chosen with respect to the operator norm of $\mathsf{K} + \lambda m \mathsf{Id}$ i.e., the Lipschitz constant of the full gradient of the primal objective function, see [1, Theorem 1]. By contrast, in algorithm (3.5.8), for all $i \in [m]$, the stepsizes γ_i^k are chosen with respect to the Lipschitz constant of the partial derivatives of the dual objective function i.e., $\mathsf{K}_{i,i} + \lambda m$. Not only the latter are easier to compute, they also allow for possibly longer steps along the coordinates.

3.6 Experiments

In this section, we will present some experiments with the purpose of assessing our theoretical findings and making comparison with related results in the literature. All the codes are available on GitHub[1].

We coded the mathematical model of asynchronicity in (3.1.2). At each iteration we compute the forward step using gradients that are possibly outdated. The delay vector components are a priori chosen according to a uniform distribution on $\{0, 1, \ldots, \tau\}$. The block coordinates are updated with a uniform distribution independent of the delay vector. We considered three kinds of experiments: in the first one we did a speedup test for our algorithm on the Lasso problem. This allows to check whether the speed of convergence increases linearly with the number of machines used. Then, we considered a comparison with the synchronous version of the algorithm in order to show the advantage of the asynchronous implementation. Finally, in the third group of experiments we compared our algorithm with those by Liu et al. [65] and Cannelli et al. [19].

[1] https://github.com/cheiktraore/Codes_Paper_Asc_Coord_Desc

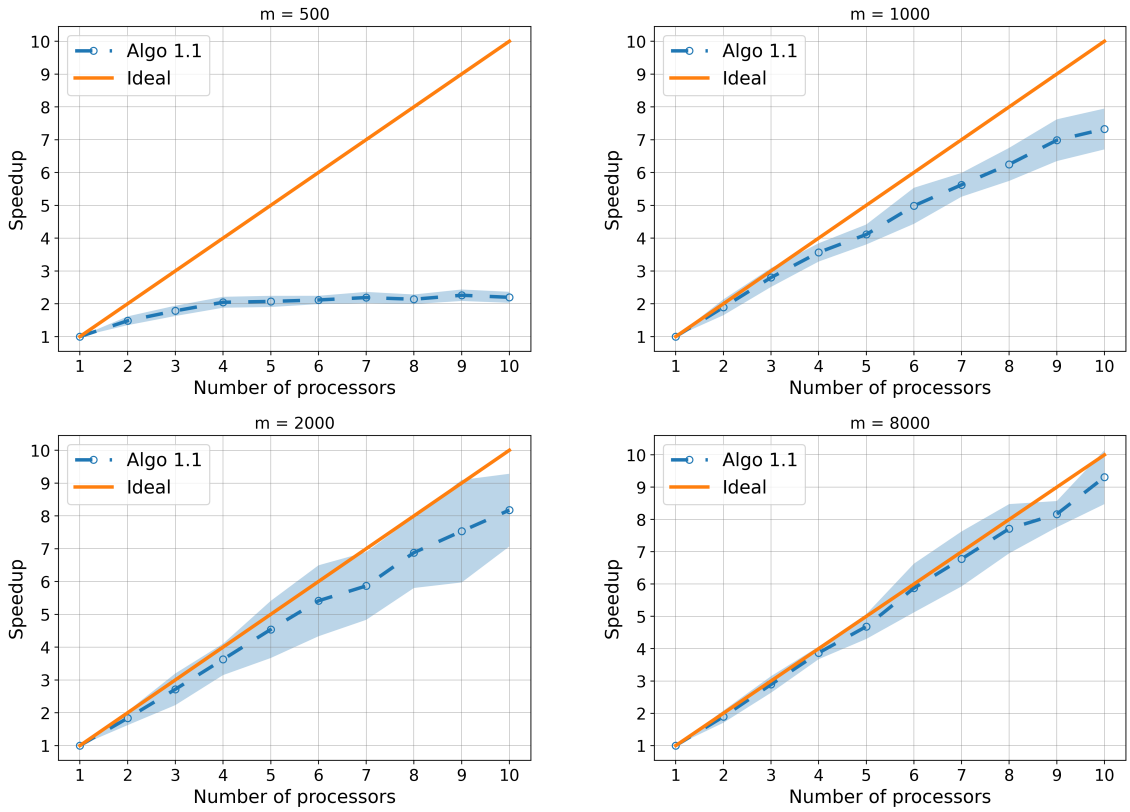


Figure 3.1: The plots showed the speedup obtain by Algorithm 1.1 compared to the ideal speedup for different number of blocks. The shaded zones illustrate the standard deviation of the results over 10 trials.

3.6.1 Speedup test

In this section we consider the Lasso problem (3.5.1) with $m \in \{500, 1000, 2000, 8000\}$ and $n = 100$. λ is chosen small enough so that the minimizer $\boldsymbol{x}_*$ has non-zero components. For more flexibility, we used synthetic data, which were generated using the function MAKE_CORRELATED_DATA of the python library CELER. This function creates a matrix A with columns generated according to the Autoregressive (AR) model[2]. Then b is generated as $\mathrm{b} = \mathrm{A}\boldsymbol{w} + \epsilon$, where ϵ is a Gaussian random vector, with zero mean and variance equal to the identity, such that the signal to noise ration (SNR) is 3 and $\boldsymbol{w}$ is a vector with 1% of nonzero entries. The nonzero blocks of $\boldsymbol{w}$ are chosen uniformly and their entries are generated according to the standard normal distribution. As in [19, 65], we make the assumption that τ is proportional to the number of machines. Since we use 10 cores, we fix $\tau = 10$ like in [64]. For a fixed data, we run the algorithm 10 times and average it. Similarly to [19, 65], in our experiment the speedup gets better when we increase the number of blocks, see Figure 3.1. This can be explained by the fact that the algorithm has to run long enough in order to minimize the cost of parallelization — the initialization cost, the mandatory locks in order to avoid data racing, etc. Also, if there are more blocks, the probability of two machines having to write to the same block at the same time is reduced and so is the number of locks. All these observations align with the known fact that the more there are cores, the more the problem should be complex to see good speedup.

[2]The code is available at https://github.com/mathurinm/celer/blob/501788e/celer/datasets/simulated.py#L10

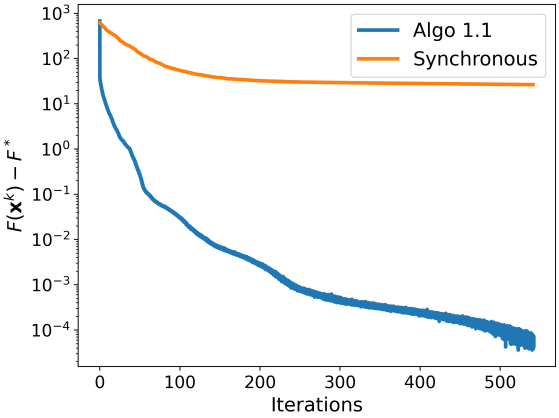


Figure 3.2: Comparison of Algorithm 3.1 to its synchronous counterpart. Both algorithms were run for a fix amount of seconds. The iteration count in the figure is the normalized one.

3.6.2 Comparison with the synchronous version

We compared Algorithm 3.1 to its synchronous counterpart in the Lasso case. The data, as well as the parameters, is generated as in the speedup experiment. The stepsize of the synchronous algorithm is set as suggested in [90] for a non-sparse matrix A. We run both algorithms for 120 seconds and compare the distances of their function values to the minimum. As expected, Algorithm 3.1 is faster; see Figure 3.2.

3.6.3 Comparison with other asynchronous algorithms

In this section we illustrate the results of the comparison with the algorithms proposed in [65] and [19]. As for [19], we set (in the notation of the chapter) the relaxation parameter $\gamma = 1$ and $c_{\tilde{f}} = 2\beta$ so that

$$x_i^{k+1} = \begin{cases} \mathsf{prox}_{(1/2\beta)g_i}\left(x_i^{k-d_i^k} - (1/2\beta)\nabla_i f(\boldsymbol{x}^{k-\mathbf{d}^k})\right) & \text{if } i = i_k \\ x_i^k & \text{if } i \neq i_k. \end{cases}$$

Then, according to Theorem 1 in [19], we choose $2\beta > L_f(1 + \delta^2/2)$ where $\delta = \tau$ is the maximum delay. We note that this model is slightly different from ours since the delay is present not only in the gradient.

In [65], the same algorithm as (3.1.2) is considered, but with a stepsize γ which is the same for all the blocks. In our comparisons, we choose the step according to the conditions required by the main Theorem 4.1 in [65], since the hypotheses of Corollary 4.2 are not satisfied for our datasets[3], see the discussion in Remark 3.12 (vi). If τ is the maximum delay, Theorem 4.1 in [65] requires the following conditions on the stepsize:

$$\gamma < \frac{\sqrt{n}(1-\rho^{-1}) - 4}{4(1+\theta)L_{\text{res}}/L'_{\max}} \quad \text{with } \theta = \frac{\rho^{(\tau+1)/2} - \rho^{1/2}}{\rho^{1/2} - 1},$$

which only makes sense if the right hand side is strictly positive, so when $n > 16$ and $\rho > \frac{1+4/\sqrt{n}}{1-16/n}$ (instead of $\rho > 1 + 4/\sqrt{n}$ as claimed in [65]). So, in the experiments, we set $\rho > \frac{1+4/\sqrt{n}}{1-16/n}$. This leads in general to very small stepsizes, as we will further discuss in the next section.

[3]For the two datasets we used, YEARPREDICTIONMSD.T and SPLICE.T, we have that $\sqrt{m}/(4e\Lambda)$ is equal to 0.62084123 0.00459623 respectively, so that condition (3.3.2) is never satisfied by any nonnegative integer τ.

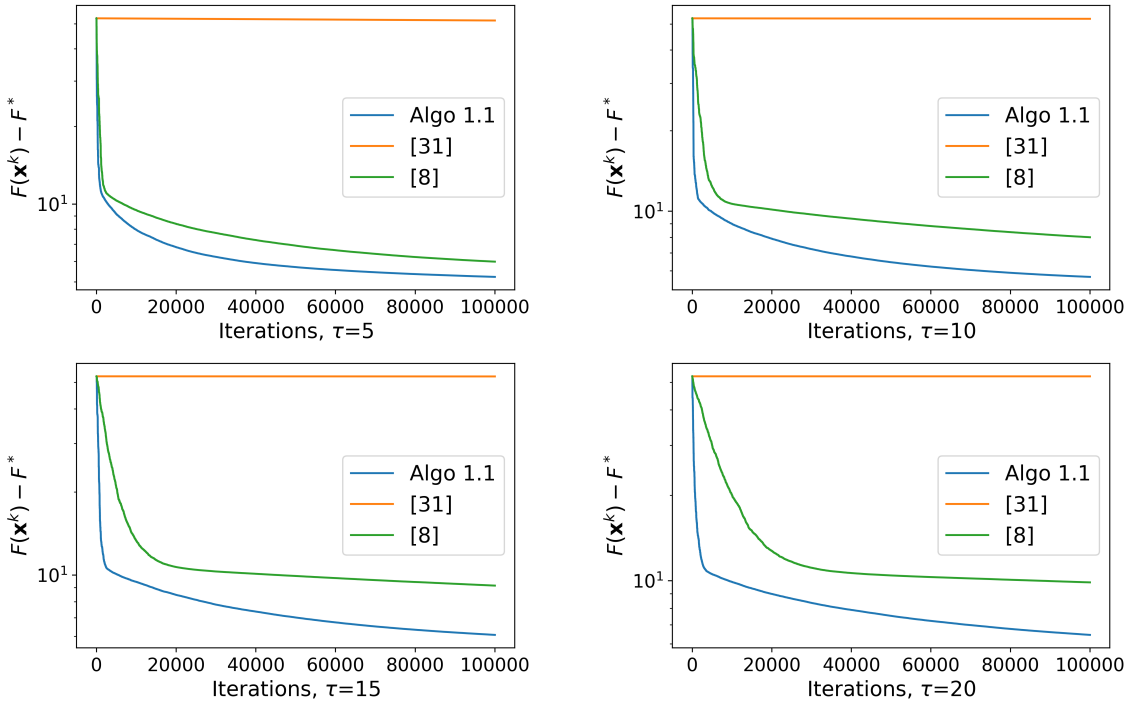


Figure 3.3: The plots show the behavior of $F(\boldsymbol{x}^k) - F_*$ for the 3 algorithms applied to a lasso loss with different values of τ: 5, 10, 15, 20.

Lasso problem

In this section we consider the Lasso problem (3.5.1) with $m = 90$, $n = 51630$, and $\lambda = 0.01$. We use the data YEARPREDICTIONMSD.T from LIBSVM[4] to generate the matrix A. Before showing the results, we briefly comment on the experimental set-up. As shown in Section 3.5.1, in this case $L_i = \|\mathsf{A}_{\cdot i}\|_2^2$ and $L_{\text{res}} = \max_i \|\mathsf{A}^\intercal \mathsf{A}_{\cdot i}\|_2$. In [19], $L_f = L_{\text{res}}$ and in [65] $L'_{\max} = \max_i \|\mathsf{A}^\intercal \mathsf{A}_{\cdot i}\|_\infty$.

Looking at the results, we see that our algorithm outperforms those in [65] and [19], see Figure 3.3. This difference is due to the fact that our stepsize is bigger than the other two. Indeed, in [65] and [19] the stepsizes have a worse dependence on the maximum delay τ (inverse quadratically in [19] and exponentially in [65]), which ultimately shorten the stepsizes. Also, in both [65] and [19] the stepsize is the same for all the blocks, so the algorithm is more sensitive to the conditioning of the problem. An overall comparison of the effect of τ on the stepsize is shown in Figure 3.4.

Logistic regression

For another comparison, next we consider the ℓ_1 regularized logistic loss:

$$F(x) = \frac{1}{n}\sum_{i=1}^{n} \log(1 + \exp\{-b_i \langle a_i, \boldsymbol{x}\rangle\}) + \lambda \|\boldsymbol{x}\|_1. \tag{3.6.1}$$

For this experiment we use the data SPLICE.T from LIBSVM[5] with $m = 60$, $n = 2175$, and $\lambda = 0.01$. Let $\mathsf{A} \in \mathbb{R}^{m\times n}$ be the matrix with columns the a_i's ($i \in [n]$). We denote by $\|\cdot\|$, $\|\cdot\|_\infty$, $\|\cdot\|_F$, the spectral norm, the infinity norm, and the Frobenius norm of matrices, respectively. The relevant constants for the stepsizes are

[4] https://www.csie.ntu.edu.tw/~cjlin/libsvmtools/datasets/

[5] https://www.csie.ntu.edu.tw/~cjlin/libsvmtools/datasets/

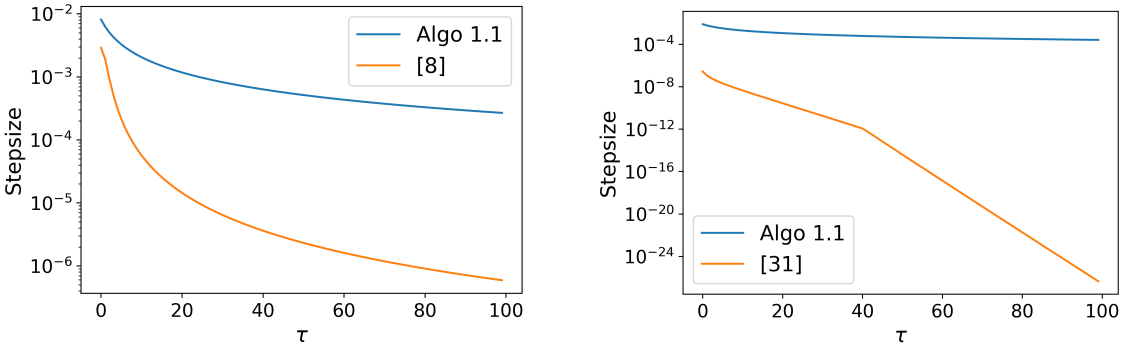


Figure 3.4: This figure shows how the minimum of our stepsizes fares against the two others when τ increases on a lasso problem.

- $L_{\text{res}} = \frac{1}{n}\|\mathsf{A}\| \max_j \|\mathsf{A}_{j\cdot}\|_2$ for our algorithm and [65],
- $L'_{\max} = \frac{1}{n}\|\mathsf{A}\|_\infty \max_j \|\mathsf{A}_{j\cdot}\|_\infty$ for [65],
- $L_j = \frac{1}{n}\|\mathsf{A}_{j\cdot}\|_2^2$, $j \in [m]$, for our algorithm, where $\mathsf{A}_{j\cdot}$ is the j-th row of A.
- $L_f = \frac{1}{n}\|\mathsf{A}\|_F \max_j \|\mathsf{A}_{j\cdot}\|_2$ for [19].

So, the stepsizes range from about $1.1191 * 10^{-3}$ to $7.5164 * 10^{-3}$ for [19], $5.6537 * 10^{-8}$ to $2.1571 * 10^{-10}$ for [65], and $2.2605 * 10^{-2}$ to $6.1590 * 10^{-3}$ for our algorithm. The results show the same trend as in the Lasso case, actually with even larger differences, see Figure 3.5.

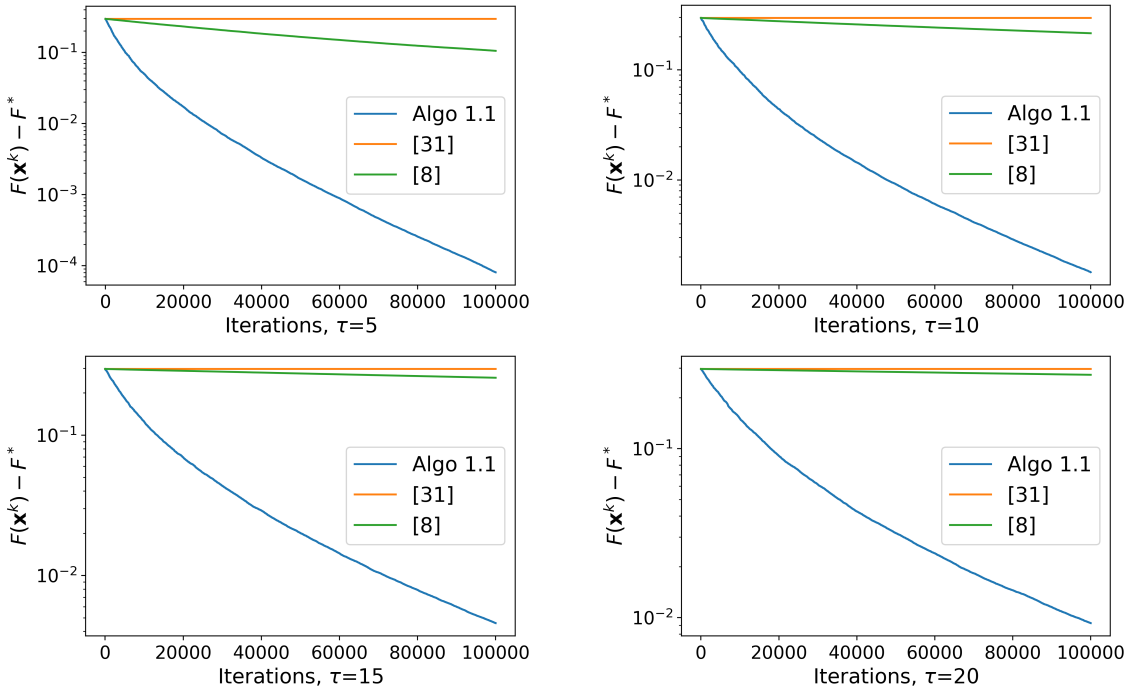


Figure 3.5: The plots show the behavior of $F(\boldsymbol{x}^k) - F_*$ for the 3 algorithms applied to a regularized logistic loss for different values of τ: 5, 10, 15, 20.

Appendix for Chapter 3

3.7 Proofs of the auxiliary Lemmas in Section 3.2

In this section, for reader's convenience, we provide detailed proofs of the Lemmas presented in Section 3.2, even though they are mostly not original. They are adapted from or can be found, e.g., in [65, 90].

Proof of Lemma 3.6. Let $k \in \mathbb{N}$. Since, for every $i \in [m]$, $\mathsf{d}_i^k \leq \min\{k, \tau\}$, we have

$$\begin{aligned}
\boldsymbol{x}^{k-\mathbf{d}^k} - \boldsymbol{x}^k &= \sum_{i=1}^{m} \mathsf{J}_i (x_i^{k-\mathsf{d}_i^k} - x_i^k) \\
&= \sum_{i=1}^{m} \mathsf{J}_i \Bigg(\sum_{h=k-\mathsf{d}_i^k}^{k-1} (x_i^h - x_i^{h+1}) \Bigg) \\
&= \sum_{i=1}^{m} \mathsf{J}_i \Bigg(\sum_{h=k-\tau}^{k-1} \delta_{h,i} (x_i^h - x_i^{h+1}) \Bigg) \\
&= \sum_{h=k-\tau}^{k-1} \sum_{i=1}^{m} \mathsf{J}_i \big(\delta_{h,i} (x_i^h - x_i^{h+1}) \big). \qquad (3.7.1)
\end{aligned}$$

where $\delta_{h,i} = 1$ if $h \geq k - \mathsf{d}_i^k$ and $\delta_{h,i} = 0$ if $h < k - \mathsf{d}_i^k$. Note that for any $h \in \{k-\tau, \dots, k-1\}$, in the sum

$$\sum_{i=1}^{m} \mathsf{J}_i \big(\delta_{h,i} (x_i^h - x_i^{h+1}) \big)$$

at most one summand is different from zero, because the difference between $\boldsymbol{x}^h$ and $\boldsymbol{x}^{h+1}$ is only in the i_h-th component. So

$$\sum_{i=1}^{m} \mathsf{J}_i \big(\delta_{h,i} (x_i^h - x_i^{h+1}) \big) = \begin{cases} \mathsf{J}_{i_h} (x_{i_h}^h - x_{i_h}^{h+1}) = \boldsymbol{x}^h - \boldsymbol{x}^{h+1} & \text{if } h \geq k - \mathsf{d}_{i_h}^k \\ 0 & \text{if } h < k - \mathsf{d}_{i_h}^k . \end{cases}$$

Therefore setting $J(k) = \big\{ h \in \{k-\tau, \dots, k-1\} \,|\, h \geq k - \mathsf{d}_{i_h}^k \big\}$, (3.7.1) yields (3.2.8). Note that, since i_h is a random variable, $J(k)$ is a random set in the sense that $J(k)(\omega) = \big\{ h \in \{k-\tau, \dots, k-1\} \,|\, h \geq k - \mathsf{d}_{i_h(\omega)}^k \big\}$. □

Proof of Lemma 3.7. Let $k \in \mathbb{N}$, let $p = \operatorname{card}(J(k))$, and let $(h_j)_{1 \leq j \leq p}$ be the elements of $J(k)$ ordered in (strictly) increasing order. Then, from Lemma 3.6 we have

$$\boldsymbol{x}^k - \hat{\boldsymbol{x}}^k = \sum_{j=1}^{p} (\boldsymbol{x}^{h_j+1} - \boldsymbol{x}^{h_j}). \qquad (3.7.2)$$

Let's set, for each $t \in \{0, \dots, p\}$

$$\hat{\boldsymbol{x}}^{k,t} = \hat{\boldsymbol{x}}^k + \sum_{j=1}^{t} (\boldsymbol{x}^{h_j+1} - \boldsymbol{x}^{h_j}).$$

Then it follows

$$\hat{\boldsymbol{x}}^{k,0} = \hat{\boldsymbol{x}}^k, \quad \hat{\boldsymbol{x}}^{k,p} = \boldsymbol{x}^k, \quad \text{and} \quad \forall\, t \geq 1 \quad \hat{\boldsymbol{x}}^{k,t} - \hat{\boldsymbol{x}}^{k,t-1} = \boldsymbol{x}^{h_t+1} - \boldsymbol{x}^{h_t}.$$

Therefore

$$\boldsymbol{x}^k - \hat{\boldsymbol{x}}^k = \sum_{t=1}^{p} (\hat{\boldsymbol{x}}^{k,t} - \hat{\boldsymbol{x}}^{k,t-1})$$

and $\hat{\boldsymbol{x}}^{k,t}, \hat{\boldsymbol{x}}^{k,t-1}$ differ only in the value of a component. Thus

$$\begin{aligned}
\|\nabla f(\boldsymbol{x}^k) - \nabla f(\hat{\boldsymbol{x}}^k)\| &= \Big\| \sum_{t=1}^{p} \nabla f(\hat{\boldsymbol{x}}^{k,t}) - \nabla f(\hat{\boldsymbol{x}}^{k,t-1}) \Big\| \\
&\leq \sum_{t=1}^{p} \|\nabla f(\hat{\boldsymbol{x}}^{k,t}) - \nabla f(\hat{\boldsymbol{x}}^{k,t-1})\| \\
&\leq L_{\mathrm{res}} \sum_{t=1}^{p} \|\hat{\boldsymbol{x}}^{k,t} - \hat{\boldsymbol{x}}^{k,t-1}\| \\
&= L_{\mathrm{res}} \sum_{t=1}^{p} \|\boldsymbol{x}^{h_t+1} - \boldsymbol{x}^{h_t}\| \\
&= L_{\mathrm{res}} \sum_{h \in J(k)} \|\boldsymbol{x}^{h+1} - \boldsymbol{x}^{h}\|.
\end{aligned}$$

from which the result follows. □

Proof of Lemma 3.9. Let $k \in \mathbb{N}$ and $\mathbf{x} \in \mathbf{H}$. Then

$$\begin{aligned}
\langle \nabla f(\hat{\boldsymbol{x}}^k), \mathbf{x} - \boldsymbol{x}^k \rangle &= \langle \nabla f(\hat{\boldsymbol{x}}^k), \mathbf{x} - \hat{\boldsymbol{x}}^k \rangle + \langle \nabla f(\hat{\boldsymbol{x}}^k), \hat{\boldsymbol{x}}^k - \boldsymbol{x}^k \rangle \\
&= \langle \nabla f(\hat{\boldsymbol{x}}^k), \mathbf{x} - \hat{\boldsymbol{x}}^k \rangle + \sum_{t=0}^{p-1} \langle \nabla f(\hat{\boldsymbol{x}}^k), \hat{\boldsymbol{x}}^{k,t} - \hat{\boldsymbol{x}}^{k,t+1} \rangle \\
&= \langle \nabla f(\hat{\boldsymbol{x}}^k), \mathbf{x} - \hat{\boldsymbol{x}}^k \rangle \\
&\quad + \sum_{t=0}^{p-1} \langle \nabla f(\hat{\boldsymbol{x}}^{k,t}), \hat{\boldsymbol{x}}^{k,t} - \hat{\boldsymbol{x}}^{k,t+1} \rangle + \langle \nabla f(\hat{\boldsymbol{x}}^k) - \nabla f(\hat{\boldsymbol{x}}^{k,t}), \hat{\boldsymbol{x}}^{k,t} - \hat{\boldsymbol{x}}^{k,t+1} \rangle.
\end{aligned}$$

Thanks to the convexity of f and (3.2.7), it follows

$$\begin{aligned}
\langle \nabla f(\hat{\boldsymbol{x}}^k), \mathbf{x} - \boldsymbol{x}^k \rangle &\leq f(\mathbf{x}) - f(\hat{\boldsymbol{x}}^k) + \sum_{t=0}^{p-1} f(\hat{\boldsymbol{x}}^{k,t}) - f(\hat{\boldsymbol{x}}^{k,t+1}) + \frac{L_{\max}}{2} \|\hat{\boldsymbol{x}}^{k,t} - \hat{\boldsymbol{x}}^{k,t+1}\|^2 \\
&\quad + \sum_{t=0}^{p-1} \langle \nabla f(\hat{\boldsymbol{x}}^k) - \nabla f(\hat{\boldsymbol{x}}^{k,t}), \hat{\boldsymbol{x}}^{k,t} - \hat{\boldsymbol{x}}^{k,t+1} \rangle \\
&= f(\mathbf{x}) - f(\boldsymbol{x}^k) + \frac{L_{\max}}{2} \sum_{t=0}^{p-1} \|\hat{\boldsymbol{x}}^{k,t} - \hat{\boldsymbol{x}}^{k,t+1}\|^2
\end{aligned}$$

$$
\begin{aligned}
&+\sum_{t=0}^{p-1}\sum_{s=0}^{t-1}\langle\nabla f(\hat{\boldsymbol{x}}^{k,s})-\nabla f(\hat{\boldsymbol{x}}^{k,s+1}),\hat{\boldsymbol{x}}^{k,t}-\hat{\boldsymbol{x}}^{k,t+1}\rangle\\
&\leq f(\mathbf{x})-f(\boldsymbol{x}^k)+\frac{L_{\max}}{2}\sum_{t=0}^{p-1}\|\hat{\boldsymbol{x}}^{k,t}-\hat{\boldsymbol{x}}^{k,t+1}\|^2\\
&\quad+L_{\mathrm{res}}\sum_{t=0}^{p-1}\sum_{s=0}^{t-1}\|\hat{\boldsymbol{x}}^{k,s}-\hat{\boldsymbol{x}}^{k,s+1}\|\|\hat{\boldsymbol{x}}^{k,t}-\hat{\boldsymbol{x}}^{k,t+1}\|.
\end{aligned}
$$

Using the equality of the square of sum, Holder inequality and $L_{\max}\leq L_{\mathrm{res}}$, we finally get

$$
\begin{aligned}
\langle\nabla f(\hat{\boldsymbol{x}}^k),\mathbf{x}-\boldsymbol{x}^k\rangle &\leq f(\mathbf{x})-f(\boldsymbol{x}^k)+\frac{L_{\max}}{2}\sum_{t=0}^{p-1}\|\hat{\boldsymbol{x}}^{k,t}-\hat{\boldsymbol{x}}^{k,t+1}\|^2\\
&\quad+\frac{L_{\mathrm{res}}}{2}\left[\left(\sum_{t=0}^{p-1}\|\hat{\boldsymbol{x}}^{k,t}-\hat{\boldsymbol{x}}^{k,t+1}\|\right)^2-\sum_{t=0}^{p-1}\|\hat{\boldsymbol{x}}^{k,t}-\hat{\boldsymbol{x}}^{k,t+1}\|^2\right]\\
&=f(\mathbf{x})-f(\boldsymbol{x}^k)+\frac{L_{\mathrm{res}}}{2}\left(\sum_{t=0}^{p-1}\|\hat{\boldsymbol{x}}^{k,t}-\hat{\boldsymbol{x}}^{k,t+1}\|\right)^2\\
&\quad+\left(\frac{L_{\max}}{2}-\frac{L_{\mathrm{res}}}{2}\right)\sum_{t=0}^{p-1}\|\hat{\boldsymbol{x}}^{k,t}-\hat{\boldsymbol{x}}^{k,t+1}\|^2\\
&\leq f(\mathbf{x})-f(\boldsymbol{x}^k)+\frac{\tau L_{\mathrm{res}}}{2}\sum_{h\in J(k)}\|\boldsymbol{x}^h-\boldsymbol{x}^{h+1}\|^2.
\end{aligned}
$$

The statement follows. □

Proof of Lemma 3.10. Let $\mathbf{z}\in\mathbf{H}$. It follows from the definition of $\mathbf{x}^+$ that $\mathbf{x}-\mathbf{x}^+-\nabla\varphi(\hat{\mathbf{x}})\in\partial\psi\left(\mathbf{x}^+\right)$. Therefore, $\psi(\mathbf{z})\geq\psi\left(\mathbf{x}^+\right)+\langle\mathbf{x}-\mathbf{x}^+-\nabla\varphi(\hat{\mathbf{x}}),\mathbf{z}-\mathbf{x}^+\rangle$, hence

$$
\left\langle\mathbf{x}-\mathbf{x}^+,\mathbf{z}-\mathbf{x}^+\right\rangle\leq\psi(\mathbf{z})-\psi\left(\mathbf{x}^+\right)+\left\langle\nabla\varphi(\hat{\mathbf{x}}),\mathbf{z}-\mathbf{x}^+\right\rangle.
$$

Then,

$$
\langle\mathbf{x}-\mathbf{x}^+,\mathbf{z}-\mathbf{x}\rangle+\langle\mathbf{x}-\mathbf{x}^+,\mathbf{x}-\mathbf{x}^+\rangle\leq\psi(\mathbf{z})-\psi\left(\mathbf{x}^+\right)+\langle\nabla\varphi(\hat{\mathbf{x}}),\mathbf{z}-\mathbf{x}\rangle+\langle\nabla\varphi(\hat{\mathbf{x}}),\mathbf{x}-\mathbf{x}^+\rangle.
$$

Rearranging the terms the statement follows. □

3.8 Proofs of Section 3.3

Proof of Lemma 3.14. Let $k\in\mathbb{N}$. We have, from Cauchy-Schwarz inequality, the Young inequality and Remark 3.8, that

$$
\begin{aligned}
&\langle\nabla f(\boldsymbol{x}^k)-\nabla f(\hat{\boldsymbol{x}}^k),\bar{\boldsymbol{x}}^{k+1}-\boldsymbol{x}^k\rangle_{\mathsf{V}}\\
&\quad\leq L_{\mathrm{res}}^{\mathsf{V}}\sum_{h\in J(k)}\|\boldsymbol{x}^{h+1}-\boldsymbol{x}^h\|_{\mathsf{V}}\|\bar{\boldsymbol{x}}^{k+1}-\boldsymbol{x}^k\|_{\mathsf{V}}\\
&\quad\leq\frac{1}{2}\left[\frac{(L_{\mathrm{res}}^{\mathsf{V}})^2}{s}\left(\sum_{h\in J(k)}\|\boldsymbol{x}^{h+1}-\boldsymbol{x}^h\|_{\mathsf{V}}\right)^2+s\|\bar{\boldsymbol{x}}^{k+1}-\boldsymbol{x}^k\|_{\mathsf{V}}^2\right]\\
&\quad\leq\frac{1}{2}\left[\frac{\tau(L_{\mathrm{res}}^{\mathsf{V}})^2}{s}\left(\sum_{h=k-\tau}^{k-1}\|\boldsymbol{x}^{h+1}-\boldsymbol{x}^h\|_{\mathsf{V}}^2\right)+s\|\bar{\boldsymbol{x}}^{k+1}-\boldsymbol{x}^k\|_{\mathsf{V}}^2\right]
\end{aligned}
$$

$$= \frac{s}{2}\|\bar{\boldsymbol{x}}^{k+1} - \boldsymbol{x}^k\|_{\mathsf{V}}^2 + \frac{\tau(L_{\mathrm{res}}^{\mathsf{V}})^2}{2s}\sum_{h=k-\tau}^{k-1}\|\boldsymbol{x}^{h+1} - \boldsymbol{x}^h\|_{\mathsf{V}}^2,$$

Now, thanks to a decomposition of the last term by Fact 3.5, we obtain

$$\begin{aligned}
&\langle \nabla f(\boldsymbol{x}^k) - \nabla f(\hat{\boldsymbol{x}}^k), \bar{\boldsymbol{x}}^{k+1} - \boldsymbol{x}^k\rangle_{\mathsf{V}} \\
&\qquad \leq \frac{s}{2}\|\bar{\boldsymbol{x}}^{k+1} - \boldsymbol{x}^k\|_{\mathsf{V}}^2 + \frac{\tau(L_{\mathrm{res}}^{\mathsf{V}})^2}{2s}\sum_{h=k-\tau}^{k-1}(h-(k-\tau)+1)\|\boldsymbol{x}^{h+1} - \boldsymbol{x}^h\|_{\mathsf{V}}^2 \\
&\qquad\quad - \frac{\tau(L_{\mathrm{res}}^{\mathsf{V}})^2}{2s}\sum_{h=k-\tau+1}^{k}(h-(k-\tau))\|\boldsymbol{x}^{h+1} - \boldsymbol{x}^h\|_{\mathsf{V}}^2 \\
&\qquad\quad + \frac{\tau^2(L_{\mathrm{res}}^{\mathsf{V}})^2}{2s}\|\boldsymbol{x}^{k+1} - \boldsymbol{x}^k\|_{\mathsf{V}}^2.
\end{aligned}$$

We recall that $\|\boldsymbol{x}^{k+1} - \boldsymbol{x}^k\|_{\mathsf{V}}^2 = \mathsf{p}_{i_k}|\bar{x}_{i_k}^{k+1} - x_{i_k}^k|^2$. So taking

$$\alpha_k = \frac{\tau(L_{\mathrm{res}}^{\mathsf{V}})^2}{2s}\sum_{h=k-\tau}^{k-1}(h-(k-\tau)+1)\|\boldsymbol{x}^{h+1} - \boldsymbol{x}^h\|_{\mathsf{V}}^2,$$

we get

$$\begin{aligned}
&\mathsf{E}\big[\langle \nabla f(\boldsymbol{x}^k) - \nabla f(\hat{\boldsymbol{x}}^k), \bar{x}^{k+1} - x^k\rangle_{\mathsf{V}} \,\big|\, i_0,\dots,i_{k-1}\big] \\
&\qquad \leq \frac{s}{2}\|\bar{\boldsymbol{x}}^{k+1} - \boldsymbol{x}^k\|_{\mathsf{V}}^2 + \frac{\tau^2(L_{\mathrm{res}}^{\mathsf{V}})^2}{2s}\sum_{i=0}^{m}\mathsf{p}_i^2|\bar{x}_i{}^{k+1} - x_i^k|^2 + \alpha_k - \mathsf{E}\big[\alpha_{k+1}\,\big|\, i_0,\dots,i_{k-1}\big]
\end{aligned}$$

Meaning

$$\begin{aligned}
&\langle \nabla f(\boldsymbol{x}^k) - \nabla f(\hat{\boldsymbol{x}}^k), \bar{x}^{k+1} - x^k\rangle_{\mathsf{V}} \\
&\qquad \leq \sum_{i=0}^{m}\mathsf{p}_i\left(\frac{s}{2} + \frac{\tau^2(L_{\mathrm{res}}^{\mathsf{V}})^2}{2s}\mathsf{p}_i\right)|\bar{x}_i{}^{k+1} - x_i^k|^2 + \alpha_k - \mathsf{E}\big[\alpha_{k+1}\,\big|\, i_0,\dots,i_{k-1}\big] \\
&\qquad \leq \sum_{i=0}^{m}\mathsf{p}_i\left(\frac{s}{2} + \frac{\tau^2(L_{\mathrm{res}}^{\mathsf{V}})^2}{2s}\mathsf{p}_{\max}\right)|\bar{x}_i{}^{k+1} - x_i^k|^2 + \alpha_k - \mathsf{E}\big[\alpha_{k+1}\,\big|\, i_0,\dots,i_{k-1}\big].
\end{aligned}$$

By minimizing $s \mapsto \left(\frac{s}{2} + \frac{\tau^2(L_{\mathrm{res}}^{\mathsf{V}})^2}{2s}\mathsf{p}_{\max}\right)$, we find $s = \tau L_{\mathrm{res}}^{\mathsf{V}}\sqrt{\mathsf{p}_{\max}}$. We then get

$$\begin{aligned}
&\langle \nabla f(\boldsymbol{x}^k) - \nabla f(\hat{\boldsymbol{x}}^k), \bar{x}^{k+1} - x^k\rangle_{\mathsf{V}} \\
&\qquad \leq \tau L_{\mathrm{res}}^{\mathsf{V}}\sqrt{\mathsf{p}_{\max}}\sum_{i=0}^{m}\mathsf{p}_i|\bar{x}_i{}^{k+1} - x_i^k|^2 + \alpha_k - \mathsf{E}\big[\alpha_{k+1}\,\big|\, i_0,\dots,i_{k-1}\big],
\end{aligned}$$

and $\alpha_k = \dfrac{L_{\mathrm{res}}^{\mathsf{V}}}{2\sqrt{\mathsf{p}_{\max}}}\displaystyle\sum_{h=k-\tau}^{k-1}(h-(k-\tau)+1)\|\boldsymbol{x}^{h+1} - \boldsymbol{x}^h\|_{\mathsf{V}}^2$. □

Proof of Lemma 3.15. We have

$$\|\boldsymbol{x}^{k+1} - \boldsymbol{x}\|_{\mathsf{W}}^2 = \sum_{i=1}^{m}\frac{1}{\mathsf{p}_i\gamma_i}|x_i^{k+1} - x_i|^2 = \frac{1}{\mathsf{p}_{i_k}\gamma_{i_k}}|\bar{x}_{i_k}^{k+1} - x_{i_k}|^2 + \|\boldsymbol{x}^k - \boldsymbol{x}\|_{\mathsf{W}}^2 - \frac{1}{\mathsf{p}_{i_k}\gamma_{i_k}}|x_{i_k}^k - x_{i_k}|^2. \tag{3.8.1}$$

Thus, taking the conditional expectation we have

$$\mathsf{E}[\|\boldsymbol{x}^{k+1}-\boldsymbol{x}\|_{\mathsf{W}}^2 \,|\, i_0,\dots,i_{k-1}] = \|\bar{\boldsymbol{x}}^{k+1}-\boldsymbol{x}\|_{\Gamma^{-1}}^2 + \|\boldsymbol{x}^k-\boldsymbol{x}\|_{\mathsf{W}}^2 - \|\boldsymbol{x}^k-\boldsymbol{x}\|_{\Gamma^{-1}}^2 \quad (3.8.2)$$

and (3.3.4) follows. The second equation follows from (3.3.4), by choosing $\boldsymbol{x}=\boldsymbol{x}^k$. □

Proof of Proposition 3.13. Let $k\in\mathbb{N}$. We have from the descent lemma along the i_k-th block-coordinate,

$$\begin{aligned}
F(\boldsymbol{x}^{k+1}) &\le f(\boldsymbol{x}^k) + \langle \nabla_{i_k} f(\boldsymbol{x}^k), \bar{x}_{i_k}^{k+1} - x_{i_k}^k\rangle + \frac{L_{i_k}}{2}|\bar{x}_{i_k}^{k+1}-x_{i_k}^k|^2 + \sum_{i=1}^n g_i(x_i^{k+1}) \\
&= f(\boldsymbol{x}^k) + \langle \nabla_{i_k} f(\boldsymbol{x}^k), \bar{x}_{i_k}^{k+1} - x_{i_k}^k\rangle + \frac{L_{i_k}}{2}|\bar{x}_{i_k}^{k+1}-x_{i_k}^k|^2 + \Big(g_{i_k}(x_{i_k}^{k+1}) + \sum_{i\neq i_k}^n g_i(x_i^k)\Big) \\
&= f(\boldsymbol{x}^k) + \langle \nabla_{i_k} f(\boldsymbol{x}^k), \bar{x}_{i_k}^{k+1} - x_{i_k}^k\rangle + \frac{L_{i_k}}{2}|\bar{x}_{i_k}^{k+1}-x_{i_k}^k|^2 \\
&\quad + \Big(g_{i_k}(x_{i_k}^{k+1}) - g_{i_k}(x_{i_k}^k) + g(\boldsymbol{x}^k)\Big) \\
&= F(\boldsymbol{x}^k) + \langle \nabla_{i_k} f(\boldsymbol{x}^k), \bar{x}_{i_k}^{k+1} - x_{i_k}^k\rangle + \frac{L_{i_k}}{2}|\bar{x}_{i_k}^{k+1}-x_{i_k}^k|^2 + \Big(g_{i_k}(\bar{x}_{i_k}^{k+1}) - g_{i_k}(x_{i_k}^k)\Big) \\
&= F(\boldsymbol{x}^k) + \langle \nabla_{i_k} f(\boldsymbol{x}^k) - \nabla_{i_k} f(\hat{\boldsymbol{x}}^k), \bar{x}_{i_k}^{k+1} - x_{i_k}^k\rangle + \frac{L_{i_k}}{2}|\bar{x}_{i_k}^{k+1}-x_{i_k}^k|^2 \\
&\quad + \Big(\langle \nabla_{i_k} f(\hat{\boldsymbol{x}}^k), \bar{x}_{i_k}^{k+1} - x_{i_k}^k\rangle + g_{i_k}(\bar{x}_{i_k}^{k+1}) - g_{i_k}(x_{i_k}^k)\Big).
\end{aligned}$$

From (3.2.5), we can write that

$$F(\boldsymbol{x}^{k+1}) \le F(\boldsymbol{x}^k) + \langle \nabla_{i_k} f(\boldsymbol{x}^k) - \nabla_{i_k} f(\hat{\boldsymbol{x}}^k), \bar{x}_{i_k}^{k+1} - x_{i_k}^k\rangle - \left(\frac{1}{\gamma_{i_k}} - \frac{L_{i_k}}{2}\right)|\bar{x}_{i_k}^{k+1}-x_{i_k}^k|^2 \quad (3.8.3)$$

By taking the conditional expectation and using Fact 3.3, it follows:

$$\begin{aligned}
&\mathsf{E}\big[F(\boldsymbol{x}^{k+1}) \,\big|\, i_0,\dots,i_{k-1}\big] \\
&\quad\le F(\boldsymbol{x}^k) + \mathsf{E}\big[\langle \nabla_{i_k} f(\boldsymbol{x}^k) - \nabla_{i_k} f(\hat{\boldsymbol{x}}^k), \bar{x}_{i_k}^{k+1} - x_{i_k}^k\rangle \,\big|\, i_0,\dots,i_{k-1}\big] \\
&\qquad - \sum_{i=0}^m \mathsf{p}_i\Big(\frac{1}{\gamma_i} - \frac{L_i}{2}\Big)|\bar{x}_i^{k+1}-x_i^k|^2 \\
&\quad= F(\boldsymbol{x}^k) + \sum_{i=0}^m \mathsf{p}_i\langle \nabla_i f(\boldsymbol{x}^k) - \nabla_i f(\hat{\boldsymbol{x}}^k), \bar{x}_i^{k+1} - x_i^k\rangle \\
&\qquad - \sum_{i=0}^m \mathsf{p}_i\Big(\frac{1}{\gamma_i} - \frac{L_i}{2}\Big)|\bar{x}_i^{k+1}-x_i^k|^2 \\
&\quad= F(\boldsymbol{x}^k) + \langle \nabla f(\boldsymbol{x}^k) - \nabla f(\hat{\boldsymbol{x}}^k), \bar{\boldsymbol{x}}^{k+1} - \boldsymbol{x}^k\rangle_{\mathsf{V}} \\
&\qquad - \sum_{i=0}^m \mathsf{p}_i\Big(\frac{1}{\gamma_i} - \frac{L_i}{2}\Big)|\bar{x}_i^{k+1}-x_i^k|^2.
\end{aligned} \quad (3.8.4)$$

From Lemma 3.14, we have

$$\begin{aligned}
&\langle \nabla f(\boldsymbol{x}^k) - \nabla f(\hat{\boldsymbol{x}}^k), \bar{x}^{k+1} - x^k\rangle_{\mathsf{V}} \\
&\qquad\le \tau L_{\mathrm{res}}^{\mathsf{V}}\sqrt{\mathsf{p}_{\max}}\sum_{i=0}^m \mathsf{p}_i|\bar{x_i}^{k+1} - x_i^k|^2 + \alpha_k - \mathsf{E}\big[\alpha_{k+1} \,\big|\, i_0,\dots,i_{k-1}\big],
\end{aligned}$$

with $\alpha_k = \dfrac{L^{\mathsf{V}}_{\mathrm{res}}}{2\sqrt{\mathsf{p}_{\max}}} \displaystyle\sum_{h=k-\tau}^{k-1} (h-(k-\tau)+1)\|\boldsymbol{x}^{h+1}-\boldsymbol{x}^h\|^2_{\mathsf{V}}$. We then plug this result in (3.8.4) obtaining

$$\sum_{i=0}^{m} \mathsf{p}_i\Big(\frac{1}{\gamma_i}-\frac{L_i}{2}\Big)|\bar{x}_i^{k+1}-x_i^k|^2 \le F(x^k)+\alpha_k+\tau L^{\mathsf{V}}_{\mathrm{res}}\sqrt{\mathsf{p}_{\max}}\sum_{i=0}^{m}\mathsf{p}_i|\bar{x}_i{}^{k+1}-x_i^k|^2 \\ -\mathsf{E}\big[F(\boldsymbol{x}^{k+1})+\alpha_{k+1}\,\big|\,i_0,\dots,i_{k-1}\big].$$

Hence

$$\sum_{i=0}^{m}\mathsf{p}_i\left(\frac{1}{\gamma_i}-\frac{L_i}{2}-\tau L^{\mathsf{V}}_{\mathrm{res}}\sqrt{\mathsf{p}_{\max}}\right)|\bar{x}_i^{k+1}-x_i^k|^2 \le F(\boldsymbol{x}^k)+\alpha_k-\mathsf{E}\big[F(\boldsymbol{x}^{k+1})+\alpha_{k+1}\,\big|\,i_0,\dots,i_{k-1}\big].$$

Since $\delta<2$, recalling (3.3.1), we have, for all $i\in[m]$,

$$\left(\frac{1}{\gamma_i}-\frac{L_i}{2}-\tau L^{\mathsf{V}}_{\mathrm{res}}\sqrt{\mathsf{p}_{\max}}\right)=\frac{1}{2\gamma_i}(2-L_i\gamma_i-2\gamma_i\tau L^{\mathsf{V}}_{\mathrm{res}}\sqrt{\mathsf{p}_{\max}})\ge\frac{1}{2\gamma_i}(2-\delta)>0.$$

Therefore the statement follows. □

Proof of Proposition 3.16. Let $k\in\mathbb{N}$ and $\mathbf{x}\in\mathbf{H}$. Since $\langle\nabla f(\hat{\boldsymbol{x}}^k),\mathbf{x}-\boldsymbol{x}^k\rangle=\langle\nabla^{\Gamma^{-1}}f(\hat{\boldsymbol{x}}^k),\mathbf{x}-\boldsymbol{x}^k\rangle_{\Gamma^{-1}}$ and $\bar{\boldsymbol{x}}^{k+1}=\mathrm{prox}_g^{\Gamma^{-1}}\big(\boldsymbol{x}^k-\nabla^{\Gamma^{-1}}f(\hat{\boldsymbol{x}}^k)\big)$, we derive from Lemma 3.10 above written in weighted norm that

$$\langle\boldsymbol{x}^k-\bar{\boldsymbol{x}}^{k+1},\mathbf{x}-\boldsymbol{x}^k\rangle_{\Gamma^{-1}}\le g(\mathbf{x})-g(\boldsymbol{x}^k)+\langle\nabla f(\hat{\boldsymbol{x}}^k),\mathbf{x}-\boldsymbol{x}^k\rangle \\ +g(\boldsymbol{x}^k)-g(\bar{\boldsymbol{x}}^{k+1})+\langle\nabla f(\hat{\boldsymbol{x}}^k),\boldsymbol{x}^k-\bar{\boldsymbol{x}}^{k+1}\rangle \\ -\|\boldsymbol{x}^k-\bar{\boldsymbol{x}}^{k+1}\|^2_{\Gamma^{-1}}. \tag{3.8.5}$$

From Lemma 3.9, we have

$$\langle\nabla f(\hat{\boldsymbol{x}}^k),\mathbf{x}-\boldsymbol{x}^k\rangle\le f(\mathbf{x})-f(\boldsymbol{x}^k)+\frac{\tau L_{\mathrm{res}}}{2}\sum_{h\in J(k)}\|\boldsymbol{x}^h-\boldsymbol{x}^{h+1}\|^2.$$

So (3.8.5) becomes

$$\langle\boldsymbol{x}^k-\bar{\boldsymbol{x}}^{k+1},\mathbf{x}-\boldsymbol{x}^k\rangle_{\Gamma^{-1}}\le F(\mathbf{x})-F(\boldsymbol{x}^k)+\frac{\tau L_{\mathrm{res}}}{2}\sum_{h\in J(k)}\|\boldsymbol{x}^h-\boldsymbol{x}^{h+1}\|^2 \\ +g(\boldsymbol{x}^k)-g(\bar{\boldsymbol{x}}^{k+1})+\langle\nabla f(\hat{\boldsymbol{x}}^k),\boldsymbol{x}^k-\bar{\boldsymbol{x}}^{k+1}\rangle \\ -\|\boldsymbol{x}^k-\bar{\boldsymbol{x}}^{k+1}\|^2_{\Gamma^{-1}}. \tag{3.8.6}$$

Next, recalling that x^k and x^{k+1} differs only in the i_k-th component, we have

$$g(\boldsymbol{x}^k)-g(\bar{\boldsymbol{x}}^{k+1})+\langle\nabla f(\hat{\boldsymbol{x}}^k),\boldsymbol{x}^k-\bar{\boldsymbol{x}}^{k+1}\rangle \\ =\mathsf{E}\left[\sum_{i=1}^{m}\frac{1}{\mathsf{p}_i}\big(g_i(x_i^k)-g_i(x_i^{k+1})+\langle\nabla_i f(\hat{\boldsymbol{x}}^k),x_i^k-x_i^{k+1}\rangle\big)\,|\,i_0,\dots,i_{k-1}\right]$$

Moreover,

$$\begin{aligned}
&\sum_{i=1}^{m}\frac{1}{\mathsf{p}_i}\big(g_i(x_i^k)-g_i(x_i^{k+1})+\langle\nabla_i f(\hat{\boldsymbol{x}}^k),x_i^k-x_i^{k+1}\rangle\big)\\
&\quad=\frac{1}{\mathsf{p}_{\min}}\big(g(\boldsymbol{x}^k)-g(\boldsymbol{x}^{k+1})+\langle\nabla f(\hat{\boldsymbol{x}}^k),\boldsymbol{x}^k-\boldsymbol{x}^{k+1}\rangle\big)\\
&\qquad-\sum_{i=1}^{m}\Big(\underbrace{\frac{1}{\mathsf{p}_{\min}}-\frac{1}{\mathsf{p}_i}}_{\geq 0}\Big)\big(g_i(x_i^k)-g_i(x_i^{k+1})+\langle\nabla_i f(\hat{\boldsymbol{x}}^k),x_i^k-x_i^{k+1}\rangle\big)\\
&\quad\leq\frac{1}{\mathsf{p}_{\min}}\big(g(\boldsymbol{x}^k)-g(\boldsymbol{x}^{k+1})+\langle\nabla f(\hat{\boldsymbol{x}}^k),\boldsymbol{x}^k-\boldsymbol{x}^{k+1}\rangle\big)\\
&\qquad-\left(\frac{1}{\mathsf{p}_{\min}}-\frac{1}{\mathsf{p}_{i_k}}\right)\frac{1}{\gamma_{i_k}}|\Delta_{i_k}^k|^2
\end{aligned}$$

where in the last inequality we used that

$$-\Big(g_{i_k}(x_{i_k}^k)-g_{i_k}(x_{i_k}^{k+1})+\langle\nabla_{i_k}f(\hat{\boldsymbol{x}}^k),x_{i_k}^k-x_{i_k}^{k+1}\rangle\Big)\leq-\frac{1}{\gamma_{i_k}}|\Delta_{i_k}^k|^2,$$

which was derived from (3.2.5). So

$$\begin{aligned}
&g(\boldsymbol{x}^k)-g(\bar{\boldsymbol{x}}^{k+1})+\langle\nabla f(\hat{\boldsymbol{x}}^k),\boldsymbol{x}^k-\bar{\boldsymbol{x}}^{k+1}\rangle\\
&\qquad\leq\frac{1}{\mathsf{p}_{\min}}\mathsf{E}\big[g(\boldsymbol{x}^k)-g(\boldsymbol{x}^{k+1})+\langle\nabla f(\hat{\boldsymbol{x}}^k),\boldsymbol{x}^k-\boldsymbol{x}^{k+1}\rangle\,\big|\,i_0,\dots,i_{k-1}\big]\\
&\qquad\quad-\frac{1}{\mathsf{p}_{\min}}\sum_{i=1}^{m}\frac{\mathsf{p}_i}{\gamma_i}|\Delta_i^k|^2+\|\boldsymbol{x}^k-\bar{\boldsymbol{x}}^{k+1}\|_{\mathsf{\Gamma}^{-1}}^2.
\end{aligned}$$

Now, by Lemma 3.14 and the block-coordinate descent lemma (3.2.6), we have

$$\begin{aligned}
&\mathsf{E}[\langle\nabla f(\hat{\boldsymbol{x}}^k),\boldsymbol{x}^k-\boldsymbol{x}^{k+1}\rangle\,|\,i_0,\dots,i_{k-1}]\\
&\quad\leq\mathsf{E}\big[\langle\nabla f(\hat{\boldsymbol{x}}^k)-\nabla f(\boldsymbol{x}^k),\boldsymbol{x}^k-\boldsymbol{x}^{k+1}\rangle\,\big|\,i_0,\dots,i_{k-1}\big]+\mathsf{E}\big[\langle\nabla f(\boldsymbol{x}^k),\boldsymbol{x}^k-\boldsymbol{x}^{k+1}\rangle\,\big|\,i_0,\dots,i_{k-1}\big]\\
&\quad=\langle\nabla f(\hat{\boldsymbol{x}}^k)-\nabla f(\boldsymbol{x}^k),\boldsymbol{x}^k-\bar{\boldsymbol{x}}^{k+1}\rangle_{\mathsf{V}}+\mathsf{E}[\langle\nabla f(\boldsymbol{x}^k),\boldsymbol{x}^k-\boldsymbol{x}^{k+1}\rangle\,|\,i_0,\dots,i_{k-1}]\\
&\quad\leq\tau L_{\text{res}}^{\mathsf{V}}\sqrt{\mathsf{p}_{\max}}\sum_{i=0}^{m}\mathsf{p}_i|\bar{x}_i^{k+1}-x_i^k|^2+\alpha_k-\mathsf{E}\big[\alpha_{k+1}\,\big|\,i_0,\dots,i_{k-1}\big]\\
&\qquad+\mathsf{E}\Big[f(\boldsymbol{x}^k)-f(\boldsymbol{x}^{k+1})+\frac{L_{i_k}}{2}|\Delta_{i_k}^k|^2\,\Big|\,i_0,\dots,i_{k-1}\Big],
\end{aligned}$$

where $\alpha_k=L_{\text{res}}^{\mathsf{V}}/(2\sqrt{\mathsf{p}_{\max}})\sum_{h=k-\tau}^{k-1}(h-(k-\tau)+1)\|\boldsymbol{x}^{h+1}-\boldsymbol{x}^h\|_{\mathsf{V}}^2$ for all $k\in\mathbb{N}$. Therefore

$$\begin{aligned}
&g(\boldsymbol{x}^k)-g(\bar{\boldsymbol{x}}^{k+1})+\langle\nabla f(\hat{\boldsymbol{x}}^k),\boldsymbol{x}^k-\bar{\boldsymbol{x}}^{k+1}\rangle\\
&\qquad\leq\frac{1}{\mathsf{p}_{\min}}\mathsf{E}[F(\boldsymbol{x}^k)+\alpha_k-F(x^{k+1})-\alpha_{k+1}\,|\,i_0,\dots,i_{k-1}]\\
&\qquad\quad+\frac{1}{\mathsf{p}_{\min}}\sum_{i=1}^{m}\mathsf{p}_i\left(\frac{L_i}{2}+\tau L_{\text{res}}^{\mathsf{V}}\sqrt{\mathsf{p}_{\max}}-\frac{1}{\gamma_i}\right)|\Delta_i^k|^2+\|\boldsymbol{x}^k-\bar{\boldsymbol{x}}^{k+1}\|_{\mathsf{\Gamma}^{-1}}^2.
\end{aligned}\tag{3.8.7}$$

Since $\gamma_iL_i+2\gamma_i\tau L_{\text{res}}^{\mathsf{V}}\sqrt{\mathsf{p}_{\max}}\leq\delta<2$, we have

$$\frac{L_i}{2}+\tau L_{\text{res}}^{\mathsf{V}}\sqrt{\mathsf{p}_{\max}}-\frac{1}{\gamma_i}=\frac{1}{2\gamma_i}(\gamma_iL_i+2\gamma_i\tau L_{\text{res}}^{\mathsf{V}}\sqrt{\mathsf{p}_{\max}}-2)<0,$$

and hence (3.8.7) yields

$$\begin{aligned} g(\boldsymbol{x}^k) - g(\bar{\boldsymbol{x}}^{k+1}) + \langle \nabla f(\hat{\boldsymbol{x}}^k), \boldsymbol{x}^k - \bar{\boldsymbol{x}}^{k+1} \rangle \\ \leq \frac{1}{\mathsf{p}_{\min}} \mathsf{E}[F(\boldsymbol{x}^k) + \alpha_k - F(x^{k+1}) - \alpha_{k+1} \,|\, i_0, \dots, i_{k-1}] \\ + \frac{\delta - 2}{2} \sum_{i=1}^{m} \frac{1}{\gamma_i} |\Delta_i^k|^2 + \|\boldsymbol{x}^k - \bar{\boldsymbol{x}}^{k+1}\|^2_{\mathsf{\Gamma}^{-1}}. \end{aligned}$$

The statement follows from (3.8.6). □

Proof of Proposition 3.17. We know that

$$\|\boldsymbol{x}^k - \mathbf{x}\|^2_{\mathsf{\Gamma}^{-1}} - \|\bar{\boldsymbol{x}}^{k+1} - \mathbf{x}\|^2_{\mathsf{\Gamma}^{-1}} = -\|\boldsymbol{x}^k - \bar{\boldsymbol{x}}^{k+1}\|^2_{\mathsf{\Gamma}^{-1}} + 2\langle \boldsymbol{x}^k - \bar{\boldsymbol{x}}^{k+1}, \boldsymbol{x}^k - \mathbf{x} \rangle_{\mathsf{\Gamma}^{-1}}.$$

We derive from Proposition 3.16, multiplied by 2, that

$$\begin{aligned} \|\bar{\boldsymbol{x}}^{k+1} - \mathbf{x}\|^2_{\mathsf{\Gamma}^{-1}} \leq \|\boldsymbol{x}^k - \mathbf{x}\|^2_{\mathsf{\Gamma}^{-1}} \\ + \frac{2}{\mathsf{p}_{\min}} \mathsf{E}\big[F(\boldsymbol{x}^k) + \alpha_k - F(\boldsymbol{x}^{k+1}) - \alpha_{k+1} \,|\, i_0, \dots, i_{k-1}\big] \\ + 2(F(\mathbf{x}) - F(\boldsymbol{x}^k)) + \tau L_{\mathrm{res}} \sum_{h \in J(k)} \|\boldsymbol{x}^h - \boldsymbol{x}^{h+1}\|^2 \\ (\delta - 1)\|\boldsymbol{x}^k - \bar{\boldsymbol{x}}^{k+1}\|^2_{\mathsf{\Gamma}^{-1}}. \end{aligned} \tag{3.8.8}$$

where $\alpha_k = L^{\mathsf{V}}_{\mathrm{res}}/(2\sqrt{\mathsf{p}_{\max}}) \sum_{h=k-\tau}^{k-1} (h - (k-\tau) + 1)\|\boldsymbol{x}^{h+1} - \boldsymbol{x}^h\|^2_{\mathsf{V}}$. It follows from Lemma 3.15 that

$$\begin{aligned} \mathsf{E}\big[\|\boldsymbol{x}^{k+1} - \mathbf{x}\|^2_{\mathsf{W}} \,|\, i_0, \dots, i_{k-1}\big] \\ \leq \|\boldsymbol{x}^k - \mathbf{x}\|^2_{\mathsf{W}} \\ + (\delta - 1)\|\boldsymbol{x}^k - \bar{\boldsymbol{x}}^{k+1}\|^2_{\mathsf{\Gamma}^{-1}} \\ + \frac{2}{\mathsf{p}_{\min}} \mathsf{E}\big[F(\boldsymbol{x}^k) + \alpha_k - F(\boldsymbol{x}^{k+1}) - \alpha_{k+1} \,|\, i_0, \dots, i_{k-1}\big] \\ + 2(F(\mathbf{x}) - F(\boldsymbol{x}^k)) + \tau L_{\mathrm{res}} \sum_{h \in J(k)} \|\boldsymbol{x}^h - \boldsymbol{x}^{h+1}\|^2. \end{aligned} \tag{3.8.9}$$

Plugging (3.3.3) in (3.8.9) the statement follows. □

Proof of Proposition 3.18. Let $k \in \mathbb{N}$ and $\mathbf{x} \in \mathbf{H}$. From Proposition 3.17, we have

$$\begin{aligned} \mathsf{E}\big[\|\boldsymbol{x}^{k+1} - \mathbf{x}\|^2_{\mathsf{W}} \,|\, i_0, \dots, i_{k-1}\big] \\ \leq \|\boldsymbol{x}^k - \mathbf{x}\|^2_{\mathsf{W}} \\ + \frac{2}{\mathsf{p}_{\min}} \left(\frac{(\delta - 1)_+}{2 - \delta} + 1 \right) \mathsf{E}\big[F(\boldsymbol{x}^k) + \alpha_k - F(\boldsymbol{x}^{k+1}) - \alpha_{k+1} \,|\, i_0, \dots, i_{k-1}\big] \\ + \tau L_{\mathrm{res}} \sum_{h \in J(k)} \|\boldsymbol{x}^h - \boldsymbol{x}^{h+1}\|^2 \\ + 2(F(\mathbf{x}) - \mathsf{E}\big[F(\boldsymbol{x}^{k+1}) + \alpha_{k+1} \,|\, i_0, \dots, i_{k-1}\big]). \\ - 2(\mathsf{E}\big[F(\boldsymbol{x}^k) + \alpha_k - F(\boldsymbol{x}^{k+1}) - \alpha_{k+1} \,|\, i_0, \dots, i_{k-1}\big]) + 2\alpha_k \end{aligned}$$

Set for all $k \in \mathbb{N}$,

$$\begin{aligned} \xi_k = 2 \left(\frac{\max\{1, (2-\delta)^{-1}\}}{\mathsf{p}_{\min}} - 1 \right) \mathsf{E}\big[F(\boldsymbol{x}^k) + \alpha_k - F(\boldsymbol{x}^{k+1}) - \alpha_{k+1} \,|\, i_0, \dots, i_{k-1}\big] \\ + \tau L_{\mathrm{res}} \sum_{h \in J(k)} \|\boldsymbol{x}^h - \boldsymbol{x}^{h+1}\|^2 + 2\alpha_k. \end{aligned}$$

Now, on the one hand, recalling (3.8.14), (3.3.3) and Lemma 3.15, we have

$$\begin{aligned}\mathsf{E}\Big[\sum_{k\in\mathbb{N}}\sum_{h\in J(k)}\|\boldsymbol{x}^h-\boldsymbol{x}^{h+1}\|^2\Big] &\leq \tau\gamma_{\max}\mathsf{p}_{\max}\sum_{k\in\mathbb{N}}\mathsf{E}[\|\boldsymbol{x}^k-\boldsymbol{x}^{k+1}\|_{\mathsf{W}}^2]\\ &\leq \frac{2\tau\gamma_{\max}\mathsf{p}_{\max}}{(2-\delta)\mathsf{p}_{\min}}\sum_{k\in\mathbb{N}}\big(\mathsf{E}[F(\boldsymbol{x}^k)+\alpha_k]-\mathsf{E}[F(\boldsymbol{x}^{k+1})-\alpha_{k+1}]\big)\\ &\leq \frac{2\tau\gamma_{\max}\mathsf{p}_{\max}}{(2-\delta)\mathsf{p}_{\min}}(F(\boldsymbol{x}^0)+\alpha_0-F_*)<+\infty\end{aligned}$$

Recalling the definition of α_k in Proposition 3.16 and of $L_{\mathrm{res}}^{\mathsf{V}}$ in Remark 3.8, this also yields

$$\begin{aligned}\mathsf{E}\Big[\sum_{k\in\mathbb{N}}\alpha_k\Big] &\leq \frac{\tau L_{\mathrm{res}}^{\mathsf{V}}}{2\sqrt{\mathsf{p}_{\max}}}\mathsf{E}\Big[\sum_{k\in\mathbb{N}}\sum_{h=k-\tau}^{k-1}\|\boldsymbol{x}^h-\boldsymbol{x}^{h+1}\|_{\mathsf{V}}^2\Big]\\ &\leq \frac{\tau L_{\mathrm{res}}^{\mathsf{V}}\mathsf{p}_{\max}}{2\sqrt{\mathsf{p}_{\max}}}\mathsf{E}\Big[\sum_{k\in\mathbb{N}}\sum_{h=k-\tau}^{k-1}\|\boldsymbol{x}^h-\boldsymbol{x}^{h+1}\|^2\Big]\\ &\leq \frac{\tau L_{\mathrm{res}}\mathsf{p}_{\max}^2}{\sqrt{\mathsf{p}_{\min}}}\frac{\tau\gamma_{\max}}{(2-\delta)\mathsf{p}_{\min}}(F(\boldsymbol{x}^0)+\alpha_0-F_*).\end{aligned}$$

On the other hand, setting $\eta_k=F(\boldsymbol{x}^k)+\alpha_k-\mathsf{E}\big[F(\boldsymbol{x}^{k+1})-\alpha_{k+1}\,|\,i_0,\ldots,i_{k-1}\big]$, which in virtue of (3.3.3) is positive P-a.s., we have

$$\mathsf{E}\Big[\sum_{k\in\mathbb{N}}\eta_k\Big]=\sum_{k\in\mathbb{N}}\mathsf{E}[\eta_k]=\sup_{n\in\mathbb{N}}\sum_{k=0}^{n}\mathsf{E}[F(\boldsymbol{x}^k)+\alpha_k]-\mathsf{E}[F(\boldsymbol{x}^{k+1})-\alpha_{k+1}]\leq F(\boldsymbol{x}^0)+\alpha_0-F_*<+\infty.$$

Let $C=\dfrac{\max\{1,(2-\delta)^{-1}\}}{\mathsf{p}_{\min}}-1+\tau^2\dfrac{L_{\mathrm{res}}\gamma_{\max}\mathsf{p}_{\max}}{\mathsf{p}_{\min}(2-\delta)}\left(1+\dfrac{\mathsf{p}_{\max}}{\sqrt{\mathsf{p}_{\min}}}\right)$. We then get

$$\sum_{k\in\mathbb{N}}\mathsf{E}[\xi_k]\leq 2C(F(\boldsymbol{x}^0)-F_*).$$

We remark that $(\forall\, i\in[m])\quad \gamma_i(L_i+2\tau L_{\mathrm{res}}\mathsf{p}_{\max}/\sqrt{\mathsf{p}_{\min}})<2$. So $\gamma_i\tau L_{\mathrm{res}}<\frac{2-\gamma_iL_i}{2}\frac{\sqrt{\mathsf{p}_{\min}}}{\mathsf{p}_{\max}}$. This implies $\tau\gamma_{\max}L_{\mathrm{res}}<\frac{2-\gamma_{\max}L_{i_0}}{2}\frac{\sqrt{\mathsf{p}_{\min}}}{\mathsf{p}_{\max}}$, where $i_0\in[m]$ such that $\gamma_{i_0}=\gamma_{\max}$. Thus

$$\tau\gamma_{\max}L_{\mathrm{res}}<\frac{2-\gamma_{\max}L_{\min}}{2}\frac{\sqrt{\mathsf{p}_{\min}}}{\mathsf{p}_{\max}}. \tag{3.8.10}$$

Using this in C, we get

$$\begin{aligned}C&\leq\frac{\max\{1,(2-\delta)^{-1}\}}{\mathsf{p}_{\min}}-1+\tau\frac{2-\gamma_{\max}L_{\min}}{2\sqrt{\mathsf{p}_{\min}}(2-\delta)}\left(1+\frac{\mathsf{p}_{\max}}{\sqrt{\mathsf{p}_{\min}}}\right)\\ &\leq\frac{\max\{1,(2-\delta)^{-1}\}}{\mathsf{p}_{\min}}-1+\tau\frac{1}{\sqrt{\mathsf{p}_{\min}}(2-\delta)}\left(1+\frac{\mathsf{p}_{\max}}{\sqrt{\mathsf{p}_{\min}}}\right).\end{aligned}$$

The statement follows. □

Proof of Proposition 3.19. It follows from (3.3.3) that

$$(2-\delta)\frac{\mathsf{p}_{\min}}{2}\mathsf{E}\big[\|\bar{\boldsymbol{x}}^{k+1}-\boldsymbol{x}^k\|_{\mathsf{\Gamma}^{-1}}^2\big]\leq\mathsf{E}\big[F(\boldsymbol{x}^k)+\alpha_k\big]-\mathsf{E}\big[F(\boldsymbol{x}^{k+1})+\alpha_{k+1}\big].$$

This means that $\big(\mathsf{E}[F(\boldsymbol{x}^k)+\alpha_k]\big)_{k\in\mathbb{N}}$ is a nonincreasing sequence and

$$\begin{aligned}(2-\delta)\frac{\mathsf{p}_{\min}}{2}\mathsf{E}\bigg[\sum_{k\in\mathbb{N}}\|\bar{\boldsymbol{x}}^{k+1}-\boldsymbol{x}^k\|^2_{\boldsymbol{\Gamma}^{-1}}\bigg] &= (2-\delta)\frac{\mathsf{p}_{\min}}{2}\sup_{k\in\mathbb{N}}\sum_{h=0}^{k}\mathsf{E}\big[\|\bar{\boldsymbol{x}}^{h+1}-\boldsymbol{x}^h\|^2_{\boldsymbol{\Gamma}^{-1}}\big]\\ &\le \sup_{k\in\mathbb{N}}\mathsf{E}\big[F(\boldsymbol{x}^0)+\alpha_0\big]-\mathsf{E}\big[F(\boldsymbol{x}^{k+1})+\alpha_{k+1}\big]\\ &\le F(\boldsymbol{x}^0)+\alpha_0-F_*<+\infty.\end{aligned}$$

Therefore, since $\|\cdot\|^2\le(\max_i\gamma_i)\|\cdot\|^2_{\boldsymbol{\Gamma}^{-1}}$, we derive that

$$\sum_{k\in\mathbb{N}}\|\bar{\boldsymbol{x}}^{k+1}-\boldsymbol{x}^k\|^2<\infty\quad\mathsf{P}\text{-a.s.}\tag{3.8.11}$$

So, it follows that

$$\|\bar{\boldsymbol{x}}^{k+1}-\boldsymbol{x}^k\|\to 0\quad\mathsf{P}\text{-a.s},\tag{3.8.12}$$

and, since $\|\boldsymbol{x}^{k+1}-\boldsymbol{x}^k\|\le\|\bar{\boldsymbol{x}}^{k+1}-\boldsymbol{x}^k\|$ for all $k\in\mathbb{N}$, we have also

$$\sum_{k\in\mathbb{N}}\|\boldsymbol{x}^{k+1}-\boldsymbol{x}^k\|^2<\infty\text{ and }\|\boldsymbol{x}^{k+1}-\boldsymbol{x}^k\|\to 0\quad\mathsf{P}\text{-a.s.}\tag{3.8.13}$$

Now, by Lemma 3.6, we have $\|\hat{\boldsymbol{x}}^k-\boldsymbol{x}^k\|^2\le\tau\sum_{h\in J(k)}\|\boldsymbol{x}^h-\boldsymbol{x}^{h+1}\|^2$ and, moreover,

$$\sum_{k\in\mathbb{N}}\sum_{h\in J(k)}\|\boldsymbol{x}^h-\boldsymbol{x}^{h+1}\|^2\le\sum_{k\in\mathbb{N}}\tau\|\boldsymbol{x}^k-\boldsymbol{x}^{k+1}\|^2<\infty\ \ \mathsf{P}\text{-a.s.},\tag{3.8.14}$$

so that

$$\|\bar{\boldsymbol{x}}^{k+1}-\hat{\boldsymbol{x}}^k\|\le\|\bar{\boldsymbol{x}}^{k+1}-\boldsymbol{x}^k\|+\|\boldsymbol{x}^k-\hat{\boldsymbol{x}}^k\|\to 0\ \ \mathsf{P}\text{-a.s.}\tag{3.8.15}$$

Define, for all $i\in[m]$,

$$v_i^k=\nabla_i f(\bar{\boldsymbol{x}}^{k+1})-\nabla_i f(\hat{\boldsymbol{x}}^k)+\frac{\Delta_i^k}{\gamma_i}.\tag{3.8.16}$$

Then, thanks to the second equation in (3.2.4), we have

$$\boldsymbol{v}^k=(v_1^k,\cdots,v_m^k)\in\nabla f(\bar{\boldsymbol{x}}^{k+1})+\partial g(\bar{\boldsymbol{x}}^{k+1})=\partial\,(f+g)\,(\bar{\boldsymbol{x}}^{k+1}).\tag{3.8.17}$$

Moreover, since ∇f is Lipschitz continuous, definition (3.8.16) and equations (3.8.12), (3.8.15) yield $\boldsymbol{v}^k\to 0$ P-a.s. □

CHAPTER 4

Variance reduction techniques for stochastic proximal point algorithms

In the context of finite sums minimization, variance reduction techniques are widely used to improve the performance of state-of-the-art stochastic gradient methods. Their practical impact is clear, as well as their theoretical properties. Stochastic proximal point algorithms have been studied as an alternative to stochastic gradient algorithms since they are more stable with respect to the choice of the stepsize but their variance reduced versions are not as studied as the gradient ones. In this work, we propose the first unified study of variance reduction techniques for stochastic proximal point algorithms. We introduce a generic stochastic proximal algorithm that can be specified to give the proximal version of SVRG, SAGA, and some of their variants for smooth and convex functions. We provide several convergence results for the iterates and the objective function values. In addition, under the Polyak-Łojasiewicz (PL) condition, we obtain linear convergence rates for the iterates and the function values. Our numerical experiments demonstrate the advantages of the proximal variance reduction methods over their gradient counterparts, especially about the stability with respect to the choice of the stepsize and the complexity of the problem.

4.1 Introduction

The objective of this chapter is to solve the finite-sum optimization problem 2.2.1, that is

$$\underset{\mathbf{x}\in\mathsf{H}}{\text{minimize}} \;\; F(\mathbf{x}) = \frac{1}{n}\sum_{i=1}^{n} f_i(\mathbf{x}), \tag{4.1.1}$$

where H is a separable Hilbert space and for all $i \in \{1, 2, \cdots, n\}$, $f_i : \mathsf{H} \to \mathbb{R}$.

Stochastic Proximal Point Algorithm. Whenever the computation of the proximity operator of f_i, prox_{f_i} (for a definition see notations Section 1.3.1), is tractable, an alternative to SGD is the stochastic proximal point algorithm (SPPA). Instead of the gradient ∇f_{i_k}, the proximity operator of an f_{i_k}, chosen randomly, is used at each iteration k:

$$\boldsymbol{x}^{k+1} = \mathsf{prox}_{\gamma_k f_{i_k}}(\boldsymbol{x}^k). \tag{SPPA}$$

As explained in Section 2.2.3, the convergence rates of SPPA are worse than those of deterministic PPA. The culprit, as with gradient algorithms, is the variance introduced by the stochasticity. The goal of this chapter is to study a variance reduction methods for SPPA, as for gradient case. Even though we work with the proximity operator, our analysis required $L-$smoothness. To the best of our knowledge, all variance reduced stochastic proximal point studies that exist in the literature also require at least smoothness (differentiability) in order to exhibit an improved rate. We present those related results in the next paragraph.

Variance reduced Stochastic Proximal Point Algorithm. Some variance reduced versions of SPPA have been proposed in the literature. In this section, we briefly describe them.

The first one is point-SAGA [29], closely related to SAGA. The update of the stored gradients is

$$\forall\, i \in [n]\colon\ \boldsymbol{\phi}_i^{k+1} = \boldsymbol{\phi}_i^k + \delta_{i,i_k}(\boldsymbol{x}^{k+1} - \boldsymbol{\phi}_i^k),$$

whereas in SAGA and its proximal version that we proposed in Algorithm 4.5, the update is

$$\forall\, i \in [n]\colon\ \boldsymbol{\phi}_i^{k+1} = \boldsymbol{\phi}_i^k + \delta_{i,i_k}(\boldsymbol{x}^{k} - \boldsymbol{\phi}_i^k).$$

In the smooth case, [29] provides linear convergence when f_i is $L-$smooth and $\mu-$strongly convex for every $i \in [n]$. Strongly convexity of each f_i implies F strong convexity. [29] also has an ergodic sublinear convergence for nonsmooth and strongly convex function. Even though the rate is ergodic and suboptimal for strongly convex functions, the stepsize provided is constant.

Still in the smooth setting, another work is [55], where the authors proved linear convergence for again strongly convex functions f_i and for the algorithm we called loopless SVRP (L-SVRP); see Algorithm 4.4. This algorithm is the proximal version of L-SVRG [60]. A difference is that [55] uses an approximation of the proximity operator at each iteration. It also uses a condition that is less strong than the $L-$smooth condition, i.e. "second-order similarity":

$$\frac{1}{n}\sum_{i=1}^{n} \|\nabla f_i(\mathbf{x}) - \nabla f(\mathbf{x}) - [\nabla f_i(\mathbf{y}) - \nabla f(\mathbf{y})]\|^2 \leq \delta^2 \|\mathbf{x} - \mathbf{y}\|^2.$$

Indeed, $L-$smoothness implies "second-order similarity".

In [73], a variant of SVRP 4.3, called SNSPP, is presented. The particularity of SNSPP is that it contains a sub-routine to compute the proximity operator. This can be useful in practice when we do not have a closed form solution of the proximity operator. Contingent on some additional condition on the conjugate of f_i and assuming semismoothness of the proximity mapping, they provide linear convergence rate for SNSPP in the L-smooth case and with F strongly convex. They also give an ergodic sublinear convergence rate for weakly convex functions.

	Algorithm	Smooth + convex	Smooth + SC	Smooth + PL	Non-Smooth + SC
Point-Saga [29]	Point-Saga	NA	$O(\varepsilon^k)$	NA	$O(1/k)$ (not optimal)
Khaled et al. [55]	L-SVRP	NA	$O(\varepsilon^k)$	NA	NA
Milzarek et al. [73]	SNSPP	NA	$O(\varepsilon^k)$	NA	NA
This chapter	Unified	$O(1/k)$ (not for SVRP)	$O(\varepsilon^k)$	$O(\varepsilon^k)$	NA

Table 4.1: This is a summary of the existing work in variance reduction for SPPA in the convex setting. In the table, SC stands for strongly convex and $0 < \varepsilon < 1$. We recall that Milzarek et al. have an ergodic sublinear rate for weakly convex functions.

Contributions. Our contribution can be summarized as follows:

- Assuming that the functions f_i are smooth, we propose the first unified variance reduction techniques for stochastic proximal point method (SPPA). Indeed, we devise

a unified analysis that extends several variance reduction techniques used for SGD to SPPA as listed in Section 4.4. In particular, we prove sub-linear convergence rate $\mathcal{O}(1/k)$ of the function values for convex functions. This convex case is new in the literature. Assuming additionally that the objective function F satisfies the Polyak-Łojasiewicz (PL) condition, we prove linear convergence rate both for the iterates and the function values. The PL condition on F is less strong than the strong convexity of F or even f_i that is used in the related previous work. Finally, we show that these results are achieved for constant stepsizes. The idea of unified study was inspired by what was done in [44] for the SGD case.

- As a byproduct, we derive and analyze some stochastic variance reduced proximal point algorithms, in analogy to SVRG [53], SAGA [30] and L-SVRG [60].
- The experiments show that, in most cases and especially for difficult problems, the proposed methods are more robust to the stepsizes and converge with bigger stepsizes, while retaining at least the same speed of convergence as their gradient counterparts. That generalizes the advantages of SPPA over SGD (see [5, 57]) to variance reduction settings.

Organization. The rest of the chapter is organized as follows: In section 4.2, we present our generic algorithm and the assumptions we will need in subsequent sections. In Section 4.3, we show the results pertaining to that algorithm. Then, in Section 4.4, we specialize the general results to particular variance reduction algorithms. Section 4.5 collects our numerical experiments. Proofs of auxiliry results can be found in Appendix 3.7.

4.2 Algorithm and assumptions

4.2.1 Algorithm

In this paragraph we describe a generic method for solving problem (2.2.1), based on the stochastic proximal point algorithm.

Algorithm 4.1: Let $(\boldsymbol{e}^k)_{k\in\mathbb{N}}$ be a sequence of random vectors in H and let $(i_k)_{k\in\mathbb{N}}$ be a sequence of i.i.d. random variables uniformly distributed on $\{1,\dots,n\}$, so that i_k is independent of $\boldsymbol{e}^0, \dots, \boldsymbol{e}^{k-1}$. Let $\gamma > 0$ and set the initial point $\boldsymbol{x}^0 \equiv \mathbf{x}^0 \in \mathsf{H}$. Then define

$$\begin{aligned}&\text{for } k = 0, 1, \dots\\ &\quad \boldsymbol{x}^{k+1} = \mathsf{prox}_{\gamma f_{i_k}}\left(\boldsymbol{x}^k + \gamma \boldsymbol{e}^k\right).\end{aligned}$$

Algorithm 4.1 is a stochastic proximal point method including a general variance reduction term $\boldsymbol{e}^k$. Note that $\boldsymbol{x}^{k+1}$ is defined as a random vector depending on $\boldsymbol{x}^k, i_k$, and $\boldsymbol{e}^k$. As we shall see in Section 4.4, depending on the specific algorithm (SPPA, SVRP, L-SVRP, SAPA), $\boldsymbol{e}^k$ may be defined in various ways. Note that by definition of the proximal operator, assuming f_{i_k} is differentiable and setting $\boldsymbol{w}^k = \nabla f_{i_k}(\boldsymbol{x}^{k+1}) - \boldsymbol{e}^k$, the update in Algorithm 4.1 can be also rewritten as:

$$\boldsymbol{x}^{k+1} = \boldsymbol{x}^k - \gamma\left[\nabla f_{i_k}(\boldsymbol{x}^{k+1}) - \boldsymbol{e}^k\right] = \boldsymbol{x}^k - \gamma \boldsymbol{w}^k. \tag{4.2.1}$$

Equation (4.2.1) shows that Algorithm 4.1 can be seen as an implicit stochastic gradient method, in contrast to the one proposed in [44], where $\boldsymbol{w}^k$ is replaced by the explicit

stochastic gradient direction

$$\boldsymbol{v}^k := \nabla f_{i_k}(\boldsymbol{x}^k) - \boldsymbol{e}^k. \tag{4.2.2}$$

We disclose here that, even thought $\boldsymbol{v}^k$ does not appear explicitly in Algorithm 4.1, it is still relevant in the associated analysis of the convergence bounds, and, for those derivations some assumptions will be required on $\boldsymbol{v}^k$.

4.2.2 Assumptions

The first assumptions are made on the functions $f_i : \mathsf{H} \to \mathbb{R}$, $i \in [n]$, as well as on the objective function $F : \mathsf{H} \to \mathbb{R}$.

Assumptions 4.1:

(A.i) $\operatorname{argmin} F \neq \varnothing$.

(A.ii) For all $i \in [n]$, f_i is convex and L-smooth, i.e., differentiable and such that

$$(\forall \mathbf{x}, \mathbf{y} \in \mathsf{H}) \quad \|\nabla f_i(\mathbf{x}) - \nabla f_i(\mathbf{y})\| \leq L\|\mathbf{x} - \mathbf{y}\|$$

for some $L > 0$. As a consequence, F is convex and L-smooth.

(A.iii) F satisfies the PL condition with constant $\mu > 0$, i.e.,

$$(\forall \mathbf{x} \in \mathsf{H}) \quad F(\mathbf{x}) - F_* \leq \frac{1}{2\mu}\|\nabla F(\mathbf{x})\|^2, \tag{4.2.3}$$

which is equivalent to the following quadratic growth condition when F is convex

$$(\forall \mathbf{x} \in \mathsf{H}) \quad \frac{\mu}{2} \operatorname{dist}(\mathbf{x}, \operatorname{argmin} F)^2 \leq F(\mathbf{x}) - F_*. \tag{4.2.4}$$

(A.i) and (A.ii) constitute the common assumptions that we use for all the convergence results presented in Sections 4.3 and 4.4. Assumption (A.iii) is often called *Polyak-Łojasiewicz condition* and was introduced in [68] (see also [83]) and is closely connected with the *quadratic growth condition* (4.2.4) (they are equivalent in the convex setting, see e.g. [16]). Conditions (4.2.3) and (4.2.4) are both relaxations of the strong convexity property and are powerful key tools in establishing linear convergence for many iterative schemes, both in the convex [16, 32, 41, 54] and the non-convex setting [4, 10, 15, 83, 84].

In the same fashion, in this work, Assumption (A.iii) will be used in order to deduce linear convergence rates in terms of objective function values for the sequence generated by Algorithm 4.1.

Assumptions 4.2: Let, for all $k \in \mathbb{N}$, $\boldsymbol{v}^k := \nabla f_{i_k}(\boldsymbol{x}^k) - \boldsymbol{e}^k$. Then there exists non-negative real numbers $A, B, C \in \mathbb{R}_+$ and $\rho \in [0, 1]$, and a non-positive real-valued random variable D such that, for every $k \in \mathbb{N}$,

(B.i) $\mathsf{E}[\boldsymbol{e}^k \,|\, \mathfrak{F}_k] = 0$ a.s.,

(B.ii) $\mathsf{E}\left[\|\boldsymbol{v}^k\|^2 \,|\, \mathfrak{F}_k\right] \leq 2A(F(\boldsymbol{x}^k) - F_*) + B\sigma_k^2 + D$ a.s.,

(B.iii) $\mathsf{E}\left[\sigma_{k+1}^2\right] \leq (1 - \rho)\mathsf{E}\left[\sigma_k^2\right] + 2C\mathsf{E}[F(\boldsymbol{x}^k) - F_*]$,

where σ_k is a real-valued random variable, $(\mathfrak{F}_k)_{k\in\mathbb{N}}$ is a sequence of σ-algebras such

that, $\forall k \in \mathbb{N}$, $\mathfrak{F}_k \subset \mathfrak{F}_{k+1} \subset \mathfrak{A}$, i_{k-1} and x^k are $\mathfrak{F}_k$-measurables, and i_k is independent of $\mathfrak{F}_k$.

Assumption (B.i) ensures that $\mathsf{E}[\boldsymbol{v}^k \,|\, \mathfrak{F}_k] = \mathsf{E}[\nabla f_{i_k}(\boldsymbol{x}^k) \,|\, \mathfrak{F}_k] = \nabla F(\boldsymbol{x}^k)$, so that the direction $\boldsymbol{v}^k$ is an unbiased estimator of the full gradient of F at $\boldsymbol{x}^k$, which is a standard assumption in the related literature. Assumption (B.ii) on $\mathsf{E}\left[\|\boldsymbol{v}^k\|^2 \,|\, \mathfrak{F}_k\right]$ is the equivalent of what is called, in the literature [44, 56], the ABC condition on $\mathsf{E}\left[\|\nabla f_{i_k}(\boldsymbol{x}^k)\|^2 \,|\, \mathfrak{F}_k\right]$ with $\sigma_k = \|\nabla F(\boldsymbol{x}^k)\|$ and D constant (see also [44]). Assumption (B.iii) is justified by the fact that it is needed for the theoretical study and it is satisfied by many examples of variance reduction techniques. For additional discussion on these assumptions, especially Assumption (B.iii), see [44].

4.3 Main results

In the rest of the chapter we will always suppose that Assumption (A.i) holds.

Before stating the main results of this work, we start with a technical proposition that constitutes the cornerstone of our analysis. The proof can be found in Appendix 4.6.1

Proposition 4.3

Suppose that Assumptions 4.2 and (A.ii) are verified and that the sequence $(\boldsymbol{x}^k)_{k\in\mathbb{N}}$ is generated by Algorithm 4.1. Let $M > 0$. Then, for all $k \in \mathbb{N}$,

$$\begin{aligned}\mathsf{E}[\operatorname{dist}(\boldsymbol{x}^{k+1}, \operatorname{argmin} F)^2] + \gamma^2 M \mathsf{E}[\sigma_{k+1}^2] \\ \leq \mathsf{E}[\operatorname{dist}(\boldsymbol{x}^k, \operatorname{argmin} F)^2] + \gamma^2 \left[M + B - \rho M\right] \mathsf{E}[\sigma_k^2] \\ - 2\gamma \left[1 - \gamma(A + MC)\right] \mathsf{E}[F(\boldsymbol{x}^k) - F_*] \\ + \gamma^2 \mathsf{E}[D].\end{aligned}$$

We now state two theorems that can be derived from the previous proposition. The first theorem deals with cases where the function F is only convex.

Theorem 4.4

Suppose that Assumptions (A.ii) and 4.2 hold with $\rho > 0$ and that the sequence $(\boldsymbol{x}^k)_{k\in\mathbb{N}}$ is generated by Algorithm 4.1. Let $M > 0$ and $\gamma > 0$ be such that $M \geq B/\rho$ and $\gamma < 1/(A + MC)$. Then, for all $k \in \mathbb{N}$,

$$\mathsf{E}[F(\bar{\boldsymbol{x}}^k) - F_*] \leq \frac{\operatorname{dist}(\boldsymbol{x}^0, \operatorname{argmin} F)^2 + \gamma^2 M \mathsf{E}[\sigma_0^2]}{2\gamma k \left[1 - \gamma(A + MC)\right]},$$

with $\bar{\boldsymbol{x}}^k = \frac{1}{k}\sum_{t=0}^{k-1} \boldsymbol{x}^t$.

Proof. Since $B - \rho M \leq 0$ and $\mathsf{E}[D] \leq 0$, it follows from Proposition 4.3 that

$$\begin{aligned}2\gamma \left[1 - \gamma(A + MC)\right] \mathsf{E}[F(\boldsymbol{x}^k) - F_*] \\ \leq \mathsf{E}[\operatorname{dist}(\boldsymbol{x}^k, \operatorname{argmin} F)^2] + \gamma^2 M \mathsf{E}[\sigma_k^2] \\ - \left(\mathsf{E}[\operatorname{dist}(\boldsymbol{x}^{k+1}, \operatorname{argmin} F)^2] + \gamma^2 M \mathsf{E}[\sigma_{k+1}]\right).\end{aligned}$$

Summing from 0 up to $k-1$ and dividing both sides by k, we obtain

$$\begin{aligned}2\gamma\left[1-\gamma(A+MC)\right]\sum_{t=0}^{k-1}\frac{1}{k}\mathsf{E}[F(\boldsymbol{x}^t)-F_*] \\ &\leq \frac{1}{k}\left(\operatorname{dist}(\boldsymbol{x}^0,\operatorname{argmin}F)^2+\gamma^2M\mathsf{E}[\sigma_0^2]\right) \\ &\quad -\frac{1}{k}\left(\mathsf{E}[\operatorname{dist}(\boldsymbol{x}^k,\operatorname{argmin}F)]^2+\gamma^2M\mathsf{E}[\sigma_k^2]\right) \\ &\leq \frac{1}{k}\left(\operatorname{dist}(\boldsymbol{x}^0,\operatorname{argmin}F)^2+\gamma^2M\mathsf{E}[\sigma_0^2]\right).\end{aligned}$$

Finally, by convexity of F, we get

$$\mathsf{E}[F(\bar{\boldsymbol{x}}^k)-F_*]\leq\frac{\operatorname{dist}(\boldsymbol{x}^0,\operatorname{argmin}F)^2+\gamma^2M\mathsf{E}[\sigma_0^2]}{2\gamma k\left[1-\gamma(A+MC)\right]}. \qquad \square$$

The next theorem shows that, when F satisfies additionally the PL property (4.2.3), the sequence generated by algorithm 4.1 exhibits a geometric convergence rate both in terms of the distance to a minimizer, as also of the objective function values.

Theorem 4.5

Suppose that Assumptions 4.1 and 4.2 are verified with $\rho>0$ and that the sequence $(\boldsymbol{x}^k)_{k\in\mathbb{N}}$ is generated by Algorithm 4.1. Let M be such that $M>B/\rho$ and $\gamma>0$ such that $\gamma<1/(A+MC)$. Set $q:=\max\left\{1-\gamma\mu\left(1-\gamma(A+MC)\right),1+\frac{B}{M}-\rho\right\}$. Then $q\in]0,1[$ and for all $k\in\mathbb{N}$,

$$V^{k+1}\leq qV^k,$$

with $V^k=\mathsf{E}[\operatorname{dist}(\boldsymbol{x}^k,\operatorname{argmin}F)^2]+\gamma^2M\mathsf{E}[\sigma_k^2]$ for all $k\in\mathbb{N}$. Moreover, for every $k\in\mathbb{N}$,

$$\mathsf{E}[\operatorname{dist}(\boldsymbol{x}^k,\operatorname{argmin}F)^2]\leq q^k\left(\operatorname{dist}(\boldsymbol{x}^0,\operatorname{argmin}F)^2+\gamma^2M\mathsf{E}[\sigma_0^2]\right),$$

$$\mathsf{E}[F(\boldsymbol{x}^k)-F_*]\leq\frac{q^kL}{2}\left(\operatorname{dist}(\boldsymbol{x}^0,\operatorname{argmin}F)^2+\gamma^2M\mathsf{E}[\sigma_0^2]\right).$$

Proof. Since $\gamma<\frac{1}{A+MC}$, we obtain thanks to Assumption (A.iii) and Proposition 4.3

$$\begin{aligned}\mathsf{E}[\operatorname{dist}(\boldsymbol{x}^{k+1},\operatorname{argmin}F)^2]+\gamma^2M\mathsf{E}[\sigma_{k+1}^2] \\ &\leq\left[1-\gamma\mu\left(1-\gamma(A+MC)\right)\right]\mathsf{E}[\operatorname{dist}(\boldsymbol{x}^k,\operatorname{argmin}F)^2] \\ &\quad+\gamma^2M\left[1+\frac{B}{M}-\rho\right]\mathsf{E}[\sigma_k^2]. \qquad (4.3.1)\end{aligned}$$

From $\gamma<\frac{1}{A+MC}$ and $M>\frac{B}{\rho}$, we obtain from (4.3.1) that

$$\begin{aligned}\mathsf{E}[\operatorname{dist}(\boldsymbol{x}^{k+1},\operatorname{argmin}F)^2]+\gamma^2M\mathsf{E}[\sigma_{k+1}^2] \\ &\leq q\left(\mathsf{E}[\operatorname{dist}(\boldsymbol{x}^k,\operatorname{argmin}F)^2]+\gamma^2M\mathsf{E}[\sigma_k^2]\right),\end{aligned}$$

with $q:=\max\left\{1-\gamma\mu\left(1-\gamma(A+MC)\right),1+\frac{B}{M}-\rho\right\}$.

Given $0<1-\rho\leq1+B/M-\rho<1$, it is clear that $q\in]0,1[$. Iterating down on k, we obtain

$$\mathsf{E}[\operatorname{dist}(\boldsymbol{x}^k,\operatorname{argmin}F)^2]\leq q^k\left(\operatorname{dist}(\boldsymbol{x}^0,\operatorname{argmin}F)^2+\gamma^2M\mathsf{E}[\sigma_0^2]\right). \qquad (4.3.2)$$

Let $\mathbf{x}_* \in \operatorname{argmin} F$. As F is L-Lipschitz smooth, from the Descent Lemma [77, Lemma 1.2.3], we have

$$(\forall \mathbf{x} \in \mathsf{H}) \quad F(\mathbf{x}) - F_* \leq \frac{L}{2}\|\mathbf{x} - \mathbf{x}_*\|^2.$$

In particular,

$$(\forall \mathbf{x} \in \mathsf{H}) \quad F(\mathbf{x}) - F_* \leq \frac{L}{2}\operatorname{dist}(\mathbf{x}, \operatorname{argmin} F)^2. \tag{4.3.3}$$

Using (4.3.3) with $\boldsymbol{x}^k$ in (4.3.2), we get

$$\mathsf{E}[F(\boldsymbol{x}^k) - F_*] \leq \frac{q^k L}{2}\left(\operatorname{dist}(\boldsymbol{x}^0, \operatorname{argmin} F)^2 + \gamma^2 M \mathsf{E}[\sigma_0^2]\right). \qquad \square$$

Remark 4.6:

(i) The convergence rate of order $O\left(\frac{1}{k}\right)$ for the general variance reduction scheme 4.1, with constant stepsize, as stated in Theorem 4.4, is an improved extension of the one found for the vanilla stochastic proximal gradient method (see e.g. [5, Proposition 3.8]). It is important to mention that the convergence with a constant step-size is no longer true when D in Assumption (B.ii) is positive. As we shall see in Section 4.4 several choices for the variance term $\boldsymbol{e}_k$ in Algorithm 4.1 can be beneficial regarding this issue, providing Assumption (B.ii) with $D \leq 0$.

(ii) The geometric rates as stated in Theorem 4.5 have some similarity with the ones found in [44, Theorem 4.1], where the authors present a unified study for variance reduced stochastic (explicit) gradient methods. However we note that the Polyak-Łojasiewicz condition (4.2.3) on F used here is slightly weaker than the quasi-strong convexity used in [44, Assumption 4.2].

4.4 Derivation of stochastic proximal point type algorithms

In this section, we provide and analyze several instances of the general scheme 4.1, corresponding to different choices of the variance reduction term $\boldsymbol{e}^k$. In particular in the next paragraphs we describe four different schemes, namely stochastic proximal point algorithm (SPPA), Stochastic Variance Reduced Proximal (SVRP) algorithm, Loopless SVRP (L-SVRP) and Stochastic Aggregate Proximal Algorithm (SAPA).

4.4.1 Stochastic Proximal Point Algorithm

We start by presenting the classic vanilla stochastic proximal method (SPPA), see e.g. [12, 14, 81]. We suppose that Assumptions (A.i) and (A.ii) hold and that for all $k \in \mathbb{N}$

$$\sigma_k^2 := \sup_{\mathbf{x}_* \in \operatorname{argmin} F} \frac{1}{n}\sum_{i=1}^n \|\nabla f_i(\mathbf{x}_*)\|^2 < +\infty. \tag{4.4.1}$$

We recall the stochastic proximal point algorithm

Algorithm 4.2 (SPPA):
Let $(i_k)_{k\in\mathbb{N}}$ be a sequence of i.i.d. random variables uniformly distributed on $\{1,\dots,n\}$. Let $\gamma_k > 0$ for all $k \in \mathbb{N}$ and set the initial point $\boldsymbol{x}^0 \equiv \mathbf{x}^0 \in \mathsf{H}$. Define

$$\begin{aligned} &\text{for } k = 0, 1, \dots \\ &\quad \boldsymbol{x}^{k+1} = \operatorname{prox}_{\gamma_k f_{i_k}}(\boldsymbol{x}^k). \end{aligned} \tag{4.4.2}$$

Algorithm 4.2 can be directly identified with the general scheme (4.1), by setting $\boldsymbol{e}^k = 0$ and $\boldsymbol{v}^k = \nabla f_{i_k}(\boldsymbol{x}^k)$. The following lemma provides a bound in expectation on the sequence $\|\boldsymbol{v}^k\|$ and can be found in the related literature, see e.g. [93, Lemma 1].

Lemma 4.7. *We suppose that Assumption (A.ii) holds and that $(\boldsymbol{x}^k)_{k\in\mathbb{N}}$ is a sequence generated by Algorithm 4.2 and $\boldsymbol{v}^k = \nabla f_{i_k}(\boldsymbol{x}^k)$. Then, for all $k \in \mathbb{N}$, it holds*

$$\begin{aligned} &\mathsf{E}[\|\boldsymbol{v}^k\|^2 \,|\, \mathfrak{F}_k] \leq 4L\big(F(\boldsymbol{x}^k) - F_*\big) + 2\sigma_k^2, \\ &\mathsf{E}\left[\sigma_{k+1}^2\right] = \sigma_{k+1}^2 = \sigma_k^2 = \mathsf{E}\left[\sigma_k^2\right], \end{aligned}$$

where $\mathfrak{F}_k = \sigma(i_0, \dots, i_{k-1})$ and σ_k^2 is defined as a constant random variable in (4.4.1).

From Lemma 4.7, we immediately notice that Assumptions 4.2 are verified with $A = 2L, B = 2, C = \rho = 0$ and $D \equiv 0$. In this setting, we are able to recover the following convergence result (see also [5, Lemma 3.10 and Proposition 3.8]).

Theorem 4.8

Suppose that Assumption (A.ii) holds and let $(\gamma_k)_{k\in\mathbb{N}}$ be a positive real valued sequence such that $\gamma_k \leq \frac{1}{4L}$, for all $k \in \mathbb{N}$. Suppose also that the sequence $(\boldsymbol{x}^k)_{k\in\mathbb{N}}$ is generated by Algorithm 4.2 with the sequence $(\gamma_k)_{k\in\mathbb{N}}$. Then, $\forall k \geq 1$,

$$\mathsf{E}[F(\bar{\boldsymbol{x}}^k) - F_*] \leq \frac{\operatorname{dist}(\boldsymbol{x}^0, \operatorname{argmin} F)^2}{\sum_{t=0}^{k-1}\gamma_t} + 2\sigma_1^2\frac{\sum_{t=0}^{k-1}\gamma_t^2}{\sum_{t=0}^{k-1}\gamma_t},$$

where $\bar{\boldsymbol{x}}^k = \displaystyle\sum_{t=0}^{k-1}\frac{\gamma_t}{\sum_{t=0}^{k-1}\gamma_t}\boldsymbol{x}^t$.

Proof. From Proposition 4.3 adapted to the update (4.4.2), it follows that

$$\begin{aligned} 2\gamma_k\left[1 - 2\gamma_k L\right]\mathsf{E}[F(\boldsymbol{x}^k) - F_*] \leq{}& \mathsf{E}[\operatorname{dist}(\boldsymbol{x}^k, \operatorname{argmin} F)]^2 \\ &- \mathsf{E}[\operatorname{dist}(\boldsymbol{x}^{k+1}, \operatorname{argmin} F)]^2 + 2\gamma_k^2\sigma_k^2. \end{aligned} \tag{4.4.3}$$

Since $\gamma_k \leq 1/4L$, for all $k \in \mathbb{N}$, we have from (4.4.3),

$$\begin{aligned} \gamma_k\mathsf{E}[F(\boldsymbol{x}^k) - F_*] &\leq \mathsf{E}[\operatorname{dist}(\boldsymbol{x}^k, \operatorname{argmin} F)]^2 - \mathsf{E}[\operatorname{dist}(\boldsymbol{x}^{k+1}, \operatorname{argmin} F)]^2 + 2\gamma_k^2\sigma_k^2 \\ &= \mathsf{E}[\operatorname{dist}(\boldsymbol{x}^k, \operatorname{argmin} F)]^2 - \mathsf{E}[\operatorname{dist}(\boldsymbol{x}^{k+1}, \operatorname{argmin} F)]^2 + 2\gamma_k^2\sigma_1^2. \end{aligned}$$

Let $k \geq 1$. Summing from 0 up to $k-1$ and dividing both side by $\sum_{t=0}^{k-1} \gamma_t$, we obtain

$$\begin{aligned}\sum_{t=0}^{k-1} \frac{\gamma_t}{\sum_{t=0}^{k-1}\gamma_t} \mathsf{E}[F(\boldsymbol{x}^t) - F_*] \\ &\leq \frac{1}{\sum_{t=0}^{k-1}\gamma_t}\left(\operatorname{dist}(\boldsymbol{x}^0, \operatorname{argmin} F)^2 - \mathsf{E}[\operatorname{dist}(\boldsymbol{x}^k, \operatorname{argmin} F)]^2\right) \\ &\qquad + 2\sigma_1^2 \frac{\sum_{t=0}^{k-1}\gamma_t^2}{\sum_{t=0}^{k-1}\gamma_t} \\ &\leq \frac{\operatorname{dist}(\boldsymbol{x}^0, \operatorname{argmin} F)^2}{\sum_{t=0}^{k-1}\gamma_t} + 2\sigma_1^2 \frac{\sum_{t=0}^{k-1}\gamma_t^2}{\sum_{t=0}^{k-1}\gamma_t}.\end{aligned}$$

Finally by convexity of F and using Jensen's inequality, we obtain

$$\mathsf{E}[F(\bar{\boldsymbol{x}}^k) - F_*] \leq \frac{\operatorname{dist}(\boldsymbol{x}^0, \operatorname{argmin} F)^2}{\sum_{t=0}^{k-1}\gamma_t} + 2\sigma_1^2 \frac{\sum_{t=0}^{k-1}\gamma_t^2}{\sum_{t=0}^{k-1}\gamma_t}. \qquad \square$$

4.4.2 Stochastic Variance Reduced Proximal point algorithm

In this paragraph we present a new Stochastic Proximal Point algorithm coupled with a variance reduction term, in the spirit of the Stochastic Variance Reduction Gradient (SVRG) method introduced in [53]. It is coined it *Stochastic Variance Reduced Proximal point algorithm* (SVRP).

The SVRP method involves two levels of iterative procedure: outer iterations and inner iterations. We shall stress out that the framework presented in the previous section covers only the inner iteration procedure and thus the convergence analysis for SVRP demands an additional care. In contrast to the subsequent schemes, Theorems 4.4 and 4.5 do not apply directly to SVRP. In particular, as it can be noted below, in the case of SVRP, the constant ρ appearing in (B.iii) in Assumptions 4.2, is null. Nevertheless, it is worth mentioning that the convergence analysis still uses Proposition 4.3.

Algorithm 4.3 (SVRP):
Let $m \in \mathbb{N}$, with $m \geq 1$, and $(\xi_s)_{s\in\mathbb{N}}$, $(i_t)_{t\in\mathbb{N}}$ be two independent sequences of i.i.d. random variables uniformly distributed on $\{0, 1, \ldots, m-1\}$ and $\{1, \ldots, n\}$ respectively. Let $\gamma > 0$ and set the initial point $\tilde{\boldsymbol{x}}^0 \equiv \tilde{\mathbf{x}}^0 \in \mathsf{H}$. Then

for $s = 0, 1, \ldots$
  $\boldsymbol{x}^0 = \tilde{\boldsymbol{x}}^s$
  for $k = 0, \ldots, m-1$
    $\boldsymbol{x}^{k+1} = \operatorname{prox}_{\gamma f_{i_{sm+k}}}\left(\boldsymbol{x}^k + \gamma\nabla f_{i_{sm+k}}(\tilde{\boldsymbol{x}}^s) - \gamma\nabla F(\tilde{\boldsymbol{x}}^s)\right)$
  $\tilde{\boldsymbol{x}}^{s+1} = \sum_{k=0}^{m-1} \delta_{k,\xi_s}\boldsymbol{x}^k,$
  or
  $\tilde{\boldsymbol{x}}^{s+1} = \sum_{k=0}^{m-1} \frac{1}{m}\boldsymbol{x}^k,$

where $\delta_{k,h}$ is the Kronecker symbol. In the case of the first option, one iterate is random selected among the inner iterates $\boldsymbol{x}_0, \boldsymbol{x}_1, \cdots, \boldsymbol{x}_{m-1}$, losing possibly a lot of information computed in the inner loop. For the second option, those inner iterates are averaged and most of the information are used.

Let $s \in \mathbb{N}$. In this case, for all $k \in \{0, 1, \cdots, m-1\}$, setting $j_k = i_{sm+k}$, the inner iteration procedure of Algorithm 4.3 can be identified with the general scheme 4.1, by

setting $\boldsymbol{e}^k \coloneqq \nabla f_{j_k}(\tilde{\boldsymbol{x}}^s) - \nabla F(\tilde{\boldsymbol{x}}^s)$. In addition let us define

$$\sigma_k^2 \coloneqq \frac{1}{n}\sum_{i=1}^{n} \|\nabla f_i(\tilde{\boldsymbol{x}}^s) - \nabla f_i(\tilde{y}^s)\|^2 \tag{4.4.4}$$

where $\tilde{y}^s \in \operatorname{argmin} F$ is such that $\|\tilde{\boldsymbol{x}}^s - \tilde{y}^s\| = \operatorname{dist}(\tilde{\boldsymbol{x}}^s, \operatorname{argmin} F)$. Moreover, setting $\mathfrak{F}_{s,k} = \sigma(\xi_0, \dots, \xi_{s-1}, i_0, \dots, i_{sm+k-1})$, we have that $\tilde{\boldsymbol{x}}^s, \tilde{y}^s$, and $\boldsymbol{x}^k$ are $\mathfrak{F}_{s,k}$-measurables and j_k is independent of $\mathfrak{F}_{s,k}$. The following result is proved in Appendix 4.6.2.

Lemma 4.9. *Suppose that Assumption (A.ii) holds true. Let $s \in \mathbb{N}$ and let $(\boldsymbol{x}^k)_{k\in[m]}$ be the (finite) sequence generated by the inner iteration in Algorithm 4.3. Set $\boldsymbol{v}^k = \nabla f_{j_k}(\boldsymbol{x}^k) - \nabla f_{j_k}(\tilde{\boldsymbol{x}}^s) + \nabla F(\tilde{\boldsymbol{x}}^s)$ and σ_k as defined in (4.4.4). Then, for every $k \in \{0, 1, \cdots, m-1\}$, it holds*

$$\mathsf{E}[\|\boldsymbol{v}^k\|^2 \,|\, \mathfrak{F}_{s,k}] \le 4L\big(F(\boldsymbol{x}^k) - F_*\big) + 2\sigma_k^2 - 2\|\nabla F(\tilde{\boldsymbol{x}}^s)\|^2, \tag{4.4.5}$$

and

$$\mathsf{E}\big[\sigma_{k+1}^2 \,|\, \mathfrak{F}_{s,0}\big] = \mathsf{E}\big[\sigma_k^2 \,|\, \mathfrak{F}_{s,0}\big].$$

As an immediate consequence, of Proposition 4.3, we have the following corollary regarding the inner iteration procedure of Algorithm 4.3.

Corollary 4.10. *Suppose that Assumption (A.ii) holds. Let $s \in \mathbb{N}$ and let $(\boldsymbol{x}^k)_{k\in[m]}$ be the sequence generated by the inner iteration in Algorithm 4.3. Then, for all $k \in \{0, 1, \cdots, m-1\}$,*

$$\begin{aligned}\mathsf{E}[\operatorname{dist}(\boldsymbol{x}^{k+1}, \operatorname{argmin} F)^2] \le \mathsf{E}[\operatorname{dist}(\boldsymbol{x}^k, \operatorname{argmin} F)^2] \\ - 2\gamma\,(1 - 2\gamma L)\,\mathsf{E}[F(\boldsymbol{x}^k) - F_*] \\ + 4L\gamma^2\mathsf{E}\,[F(\tilde{\boldsymbol{x}}^s) - F_*] - 2\gamma^2\mathsf{E}[\|\nabla F(\tilde{\boldsymbol{x}}^s)\|^2].\end{aligned} \tag{4.4.6}$$

Proof. Since Assumption (A.ii) is true, by Lemma 4.9 Assumptions 4.2 are satisfied with $\boldsymbol{v}^k = \nabla f_{j_k}(\boldsymbol{x}^k) - \nabla f_{j_k}(\tilde{\boldsymbol{x}}^s) + \nabla F(\tilde{\boldsymbol{x}}^s)$, $A = 2L$, $B = 2$, $\rho = C = 0$, and $D = -2\|\nabla F(\tilde{\boldsymbol{x}}^s)\|^2$. So Proposition 4.3 yields

$$\begin{aligned}\mathsf{E}[\operatorname{dist}(\boldsymbol{x}^{k+1}, \operatorname{argmin} F)^2 \,|\, \mathfrak{F}_{s,0}] \le \mathsf{E}[\operatorname{dist}(\boldsymbol{x}^k, \operatorname{argmin} F)^2 \,|\, \mathfrak{F}_{s,0}] \\ - 2\gamma\,(1 - 2\gamma L)\,\mathsf{E}[F(\boldsymbol{x}^k) - F_* \,|\, \mathfrak{F}_{s,0}] \\ + 4L\gamma^2\mathsf{E}\,[F(\tilde{\boldsymbol{x}}^s) - F_* \,|\, \mathfrak{F}_{s,0}] \\ - 2\gamma^2\mathsf{E}[\|\nabla F(\tilde{\boldsymbol{x}}^s)\|^2 \,|\, \mathfrak{F}_{s,0}].\end{aligned}$$

Thus, taking the total expectation, the statement follows. □

The next Theorem shows that under some additional assumptions on the choice of the step-size γ and the number of inner iterations $m \in \mathbb{N}$, Algorithm 4.3 yields a geometric convergence rate in terms of expectation of the objective function values of the outer iterates $(\tilde{\boldsymbol{x}}^s)_{s\in\mathbb{N}}$.

Theorem 4.11

Suppose that Assumptions 4.1 are satisfied and that the sequence $(\tilde{\boldsymbol{x}}^s)_{s\in\mathbb{N}}$ is generated by Algorithm 4.3 with

$$0 < \gamma < \frac{1}{2(2L - \mu)} \quad \text{and} \quad m > \frac{1}{\mu\gamma(1 - 2\gamma(2L - \mu))}. \tag{4.4.7}$$

Then, for all $s \in \mathbb{N}$, it holds

$$\mathsf{E}\left[F\left(\tilde{\boldsymbol{x}}^{s+1}\right) - F_*\right] \leq q^s \left(F(\boldsymbol{x}^0) - F_*\right), \tag{4.4.8}$$

with $q := \left(\frac{1}{\mu\gamma(1-2L\gamma)m} + \frac{2\gamma(L-\mu)}{1-2L\gamma}\right) < 1.$

Remark 4.12:

(i) Conditions (4.4.7) is used in this form if γ is set first and m is chosen after. They are needed to ensure that $q < 1$. Indeed, it is clear that $0 < \gamma < \frac{1}{2(2L-\mu)}$ is needed to have $\frac{2\gamma(L-\mu)}{1-2L\gamma} < 1$. If not, q can not be less than 1. Then we need $m > \frac{1}{\mu\gamma(1-2\gamma(2L-\mu))}$ once γ is fixed.

(ii) Conditions (4.4.7) can be equivalently stated as follows

$$m \geq \frac{8(2L-\mu)}{\mu} \quad \text{and} \quad \frac{1-\sqrt{1-8(2\kappa-1)/m}}{4(2L-\mu)} < \gamma \leq \frac{1+\sqrt{1-8(2\kappa-1)/m}}{4(2L-\mu)},$$

$\kappa = \frac{L}{\mu}$. The above formulas can be useful if one prefers to set the parameter m first and set the stepsize γ afterwards.

(iii) The convergence rate in Theorem 4.11 establishes the improvement from the outer step s to $s+1$. Of course, it depends on the number of inner iterations m. As expected, and as we can see from Equation 4.4.8, increasing m improve the bound on the rate, and since m is not bounded from above, the best choice would be to let m go to $+\infty$. In practice, there is not a best choice of m, but empirically a balance between the number of inner and outer iterations should be found. Consequently there is no optimal choice for γ either.

(iv) It is worth mentioning that the linear convergence factor q in (4.4.8) is better (smaller) than the one provided in [53, Theorem 1] for the SVRG method for strongly convex functions. There, it is $\frac{1}{\mu\gamma(1-2L\gamma)m} + \frac{2\gamma L}{1-2L\gamma}$. The linear rate of convergence in (4.4.8) is also better than the one in [112, Proposition 3.1], and also [43, Theorem 1], dealing with a proximal version of SVRG for functions satisfying the PL condition (4.2.3). In both papers, the factor is $\frac{1}{2\mu\gamma(1-4L\gamma)m} + \frac{4\gamma L(m+1)}{(1-4L\gamma)m}$. However, we note that this improvement can be also obtained for the aforementioned SVRG methods using a similar analysis.

(v) We shall stress out that by following the lines of the proof of Theorem 4.11, the same linear convergence rate found in (4.4.8), holds true also for the averaged iterate $\tilde{\boldsymbol{x}}^{s+1} = \frac{1}{m}\sum_{k=0}^{m-1} \boldsymbol{x}^k$. But the analysis does not work for the last iterate of the inner loop.

Proof of Theorem 4.11. We consider a fixed stage $s \in \mathbb{N}$ and $\tilde{\boldsymbol{x}}^{s+1}$ is defined as in Algorithm 4.3. By summing inequality (4.4.6) in Corollary 4.10 over $k = 0, \ldots, m-1$ and taking the

total expectation, we obtain

$$\begin{aligned}
\mathsf{E}[\operatorname{dist}(\boldsymbol{x}^m, \operatorname{argmin} F)]^2 &+ 2\gamma(1-2L\gamma)m\mathsf{E}[F\left(\tilde{\boldsymbol{x}}^{s+1}\right) - F_*] \\
&\leq \operatorname{dist}(\boldsymbol{x}^0, \operatorname{argmin} F)^2 + 4Lm\gamma^2\mathsf{E}\left[F(\tilde{\boldsymbol{x}}^s) - F_*\right] \\
&\qquad - 2\gamma^2 m\mathsf{E}[\|\nabla F(\tilde{\boldsymbol{x}}^s)\|^2] \\
&= \mathsf{E}[\operatorname{dist}(\tilde{\boldsymbol{x}}^s, \operatorname{argmin} F)]^2 + 4Lm\gamma^2\mathsf{E}\left[F(\tilde{\boldsymbol{x}}^s) - F_*\right] \\
&\qquad - 2\gamma^2 m\mathsf{E}[\|\nabla F(\tilde{\boldsymbol{x}}^s)\|^2] \\
&\leq \frac{2}{\mu}\mathsf{E}\left[F(\tilde{\boldsymbol{x}}^s) - F_*\right] + 4Lm\gamma^2\mathsf{E}\left[F(\tilde{\boldsymbol{x}}^s) - F_*\right] \\
&\qquad - 2\gamma^2 m\mathsf{E}[\|\nabla F(\tilde{\boldsymbol{x}}^s)\|^2] \\
&= 2\left(\mu^{-1} + 2Lm\gamma^2\right)\mathsf{E}\left[F(\tilde{\boldsymbol{x}}^s) - F_*\right] - 2\gamma^2 m\mathsf{E}[\|\nabla F(\tilde{\boldsymbol{x}}^s)\|^2] \\
&\leq 2\left(\mu^{-1} + 2Lm\gamma^2 - 2\mu m\gamma^2\right)\mathsf{E}\left[F(\tilde{\boldsymbol{x}}^s) - F_*\right].
\end{aligned} \tag{4.4.9}$$

In the first inequality, we used the fact that

$$\sum_{k=0}^{m-1} F(\boldsymbol{x}^k) = m\sum_{k=0}^{m-1}\frac{1}{m}F(\boldsymbol{x}^k) = m\sum_{\xi=0}^{m-1}\frac{1}{m}F\left(\textstyle\sum_{k=0}^{m-1}\delta_{k,\xi}\boldsymbol{x}^k\right) = m\mathsf{E}[F(\tilde{\boldsymbol{x}}^{s+1}) \,|\, \mathfrak{F}_{s,m-1}].$$

Notice that relation (4.4.9) is still valid by choosing $\tilde{\boldsymbol{x}}^{s+1} = \sum_{k=0}^{m-1}\frac{1}{m}\boldsymbol{x}^k$, in Algorithm 4.3, and using Jensen inequality to lower bound $\sum_{k=0}^{m-1} F(\boldsymbol{x}^k)$ by $mF(\tilde{\boldsymbol{x}}^{s+1})$.

The second and the last inequalities use respectively the quadratic growth (4.2.4) and the PL condition (4.2.3). We thus obtain

$$\mathsf{E}\left[F\left(\tilde{\boldsymbol{x}}^{s+1}\right) - F_*\right] \leq \left(\frac{1}{\mu\gamma(1-2L\gamma)m} + \frac{2\gamma(L-\mu)}{1-2L\gamma}\right)\mathsf{E}\left[F\left(\tilde{\boldsymbol{x}}^s\right) - F_*\right]. \qquad \square$$

4.4.3 Loopless SVRP

In this paragraph we propose a single-loop variant of the SVRP algorithm presented previously, by removing the burden of choosing the number of inner iterations. This idea is inspired by the loopless Stochastic Variance Reduced Gradient (L-SVRG) method, as proposed in [51, 60] (see also [44]) and here we present the stochastic proximal method variant that we call L-SVRP.

Algorithm 4.4 (L-SVRP):
Let $(i_k)_{k\in\mathbb{N}}$ be a sequence of i.i.d. random variables uniformly distributed on $\{1,\dots,n\}$ and let $(\varepsilon^k)_{k\in\mathbb{N}}$ be a sequence of i.i.d Bernoulli random variables such that $\mathsf{P}(\varepsilon^k = 1) = p \in]0,1]$. Let $\gamma > 0$ and set the initial points $\boldsymbol{x}^0 = \boldsymbol{u}^0 \equiv \mathsf{x}^0 \in \mathsf{H}$. Then

for $k = 0, 1, \dots$
$\quad \boldsymbol{x}^{k+1} = \operatorname{prox}_{\gamma f_{i_k}}\left(\boldsymbol{x}^k + \gamma\nabla f_{i_k}(\boldsymbol{u}^k) - \gamma\nabla F(\boldsymbol{u}^k)\right)$
$\quad \boldsymbol{u}^{k+1} = (1-\varepsilon^k)\boldsymbol{u}^k + \varepsilon^k\boldsymbol{x}^k.$

Here we note that Algorithm 4.4 can be identified with the general scheme 4.1, by setting $\boldsymbol{e}^k := \nabla f_{i_k}(\boldsymbol{u}^k) - \nabla F(\boldsymbol{u}^k)$. In addition we define

$$\sigma_k^2 := \frac{1}{n}\sum_{i=1}^n \left\|\nabla f_i(\boldsymbol{u}^k) - \nabla f_i(y^k)\right\|^2 \tag{4.4.10}$$

with $y^k \in \operatorname{argmin} F$ such that $\|\boldsymbol{u}^k - y^k\| = \operatorname{dist}(\boldsymbol{u}^k, \operatorname{argmin} F)$. Moreover, setting $\mathfrak{F}_k = \sigma(i_0,\dots,i_{k-1},\varepsilon^0,\dots,\varepsilon^{k-1})$, we have that $\boldsymbol{x}^k$, $\boldsymbol{u}^k$ and y^k are $\mathfrak{F}_k$-measurable, i_k and ε^k are independent of $\mathfrak{F}_k$.

Lemma 4.13. *Suppose that Assumption (A.ii) is satisfied. Let $(\boldsymbol{x}^k)_{k\in\mathbb{N}}$ be the sequence generated by Algorithm 4.4, with $\boldsymbol{v}^k = \nabla f_{i_k}(\boldsymbol{x}^k) - \nabla f_{i_k}(\boldsymbol{u}^k) + \nabla F(\boldsymbol{u}^k)$ and σ_k as defined in (4.4.10). Then for all $k \in \mathbb{N}$, it holds*

$$\mathsf{E}[\|\boldsymbol{v}^k\|^2 \,|\, \mathfrak{F}_k] \leq 4L\big(F(\boldsymbol{x}^k) - F_*\big) + 2\sigma_k^2, \tag{4.4.11}$$

and

$$\mathsf{E}\big[\sigma_{k+1}^2\big] \leq (1-p)\mathsf{E}\big[\sigma_k^2\big] + 2pL\mathsf{E}\big[F(\boldsymbol{x}^k) - F_*\big]. \tag{4.4.12}$$

Lemma 4.13 whose proof can be found in Appendix 4.6.2 ensures that Assumptions 4.2 hold true with constants $A = 2L, B = 2, C = pL$, $\rho = p$ and $D \equiv 0$. Then the following corollaries can be obtained by applying respectively Theorem 4.4 and 4.5 on Algorithm 4.4.

Corollary 4.14: Suppose that Assumption (A.ii) holds. Suppose also that the sequence $(\boldsymbol{x}^k)_{k\in\mathbb{N}}$ is generated by Algorithm 4.4. Let M such that $M \geq \frac{2}{p}$ and $\gamma > 0$ such that $\gamma < \frac{1}{L(2+pM)}$. Then, for all $k \in \mathbb{N}$,

$$\mathsf{E}[F(\bar{\boldsymbol{x}}^k) - F_*] \leq \frac{\operatorname{dist}(\boldsymbol{x}^0, \operatorname{argmin} F)^2 + \gamma^2 M \mathsf{E}[\sigma_0^2]}{2\gamma k\left[1 - \gamma L(2+pM)\right]},$$

with $\bar{\boldsymbol{x}}^k = \frac{1}{k}\sum_{t=0}^{k-1} \boldsymbol{x}^t$.

Corollary 4.15: Suppose that Assumptions 4.1 are verified. Suppose now that the sequence $(\boldsymbol{x}^k)_{k\in\mathbb{N}}$ is generated by Algorithm 4.4. Let M such that $M > 2/p$ and $\gamma > 0$ such that $\gamma < \frac{1}{L(2+pM)}$. Set $q := \max\left\{1 - \gamma\mu\left(1 - \gamma L(2+pM)\right), 1 + \frac{2}{M} - p\right\}$. Then $q \in]0,1[$ and for all $k \in \mathbb{N}$,

$$\mathsf{E}[\operatorname{dist}(\boldsymbol{x}^k, \operatorname{argmin} F)]^2 \leq q^k\left(\operatorname{dist}(\boldsymbol{x}^0, \operatorname{argmin} F)^2 + \gamma^2 M \mathsf{E}[\sigma_0^2]\right),$$

$$\mathsf{E}[F(\boldsymbol{x}^k) - F_*] \leq \frac{q^k L}{2}\left(\operatorname{dist}(\boldsymbol{x}^0, \operatorname{argmin} F)^2 + \gamma^2 M \mathsf{E}[\sigma_0^2]\right).$$

4.4.4 Stochastic Average Proximal Algorithm

In this paragraph, we propose a new stochastic proximal point method in analogy to SAGA [30], called *Stochastic Aggregated Proximal Algorithm* (SAPA).

Algorithm 4.5 (SAPA):
Let $(i_k)_{k\in\mathbb{N}}$ be a sequence of i.i.d. random variables uniformly distributed on $\{1,\ldots,n\}$. Let $\gamma > 0$. Set the initial point $\boldsymbol{x}^0 \equiv \mathbf{x}^0 \in \mathsf{H}$ and, for every $i \in [n]$, $\boldsymbol{\phi}_i^0 = \mathbf{x}^0$. Then

for $k = 0, 1, \ldots$
$\quad \boldsymbol{x}^{k+1} = \operatorname{prox}_{\gamma f_{i_k}}\left(\boldsymbol{x}^k + \gamma\nabla f_{i_k}(\boldsymbol{\phi}_{i_k}^k) - \frac{\gamma}{n}\sum_{i=1}^n \nabla f_i(\boldsymbol{\phi}_i^k)\right),$
$\quad \forall\, i \in [n]\colon\ \boldsymbol{\phi}_i^{k+1} = \boldsymbol{\phi}_i^k + \delta_{i,i_k}(\boldsymbol{x}^k - \boldsymbol{\phi}_i^k),$

where $\delta_{i,j}$ is the Kronecker symbol.

As for the previous cases, SAPA can be identified with Algorithm 4.1, by setting $\boldsymbol{e}^k :=$

$\nabla f_{i_k}(\boldsymbol{\phi}_{i_k}^k) - \frac{1}{n}\sum_{i=1}^n \nabla f_i(\boldsymbol{\phi}_i^k)$ for all $k \in \mathbb{N}$. In addition, let

$$\sigma_k^2 := \frac{1}{n}\sum_{i=1}^n \left\|\nabla f_i(\boldsymbol{\phi}_i^k) - \nabla f_i(\boldsymbol{x}_*)\right\|^2 \tag{4.4.13}$$

with $\boldsymbol{x}_* \in \operatorname{argmin} F$ such that $\|\boldsymbol{x}^0 - \boldsymbol{x}_*\| = \operatorname{dist}(\boldsymbol{x}^0, \operatorname{argmin} F)$. Setting $\mathfrak{F}_k = \sigma(i_0, \ldots, i_{k-1})$, we have that $\boldsymbol{x}^k$ and $\boldsymbol{\phi}_i^k$ are $\mathfrak{F}_k$-measurables and i_k is independent of $\mathfrak{F}_k$.

Lemma 4.16. *Suppose that Assumption (A.ii) holds. Let $(\boldsymbol{x}^k)_{k\in\mathbb{N}}$ be the sequence generated by Algorithm 4.5, with $\boldsymbol{v}^k = \nabla f_{i_k}\left(\boldsymbol{x}^k\right) - \nabla f_{i_k}(\boldsymbol{\phi}_{i_k}^k) + \frac{1}{n}\sum_{i=1}^n \nabla f_i(\boldsymbol{\phi}_i^k)$ and σ_k as defined in (4.4.13). Then, for all $k \in \mathbb{N}$, it holds*

$$\mathsf{E}\left[\|\boldsymbol{v}^k\|^2 \,|\, \mathfrak{F}_k\right] \leq 4L(F(\boldsymbol{x}^k) - F_*) + 2\sigma_k^2,$$

and

$$\mathsf{E}\left[\sigma_{k+1}^2\right] \leq \left(1 - \frac{1}{n}\right)\mathsf{E}\left[\sigma_k^2\right] + \frac{2L}{n}\mathsf{E}\left[F(\boldsymbol{x}^k) - F_*\right]. \tag{4.4.14}$$

From the above lemma we know that Assumptions 4.2 are verified with $A = 2L, B = 2, C = \frac{L}{n}$, $\rho = \frac{1}{n}$ and $D \equiv 0$. These allow us to state the next corollaries obtained by applying respectively Theorem 4.4 and 4.5 on Algorithm 4.5.

Corollary 4.17: Suppose that Assumptions (A.ii) are verified. Suppose also that the sequence $(\boldsymbol{x}^k)_{k\in\mathbb{N}}$ is generated by Algorithm 4.5. Let M such that $M \geq 2n$ and $\gamma > 0$ such that $\gamma < \frac{1}{L(2+M/n)}$. Then, for all $k \in \mathbb{N}$,

$$\mathsf{E}[F(\bar{\boldsymbol{x}}^k) - F_*] \leq \frac{\operatorname{dist}(\boldsymbol{x}^0, \operatorname{argmin} F)^2 + \gamma^2 M \mathsf{E}[\sigma_0^2]}{2\gamma k\left[1 - \gamma L(2 + M/n)\right]},$$

with $\bar{\boldsymbol{x}}^k = \frac{1}{k}\sum_{t=0}^{k-1} \boldsymbol{x}^t$.

Corollary 4.18: Suppose that Assumptions 4.1 hold. Suppose now that the sequence $(\boldsymbol{x}^k)_{k\in\mathbb{N}}$ is generated by Algorithm 4.5. Let M such that $M > 2n$ and $\gamma > 0$ such that $\gamma < \frac{1}{L(2+M/n)}$. Set $q := \max\left\{1 - \gamma\mu\left(1 - \gamma L(2 + M/n)\right), 1 + \frac{2}{M} - \frac{1}{n}\right\}$. Then $q \in]0, 1[$ and for all $k \in \mathbb{N}$,

$$\mathsf{E}[\operatorname{dist}(\boldsymbol{x}^k, \operatorname{argmin} F)]^2 \leq q^k\left(\operatorname{dist}(\boldsymbol{x}^0, \operatorname{argmin} F)^2 + \gamma^2 M \mathsf{E}[\sigma_0^2]\right),$$

$$\mathsf{E}[F(\boldsymbol{x}^k) - F_*] \leq \frac{q^k L}{2}\left(\operatorname{dist}(\boldsymbol{x}^0, \operatorname{argmin} F)^2 + \gamma^2 M \mathsf{E}[\sigma_0^2]\right).$$

4.5 Experiments

In this section we perform some experiments on synthetic data to compare the schemes presented and analyzed in Section 4.4. We compare the reduced variance algorithms SAPA (Algorithm 4.5) and SVRP (Algorithm 4.3) to their vanilla counterpart SPPA (Algorithm 4.2), as also to their explicit gradient counterparts: SAGA [30] and SVRG [53]. All the codes are available on GitHub[1].

[1] https://github.com/cheiktraore/Variance-reduction-for-SPPA

4.5.1 Comparing SAPA and SVRP to SPPA

The cost of each iteration of SVRP is different to that of SPPA. More precisely, SVRP consists of two nested iterations, where each outer iteration requires a full explicit gradient and m stochastic gradient computations. We will consider the minimization of the sum of n functions, therefore, we run SPPA for N iterations, with $N = S(m+n+1)$, where S is the maximum number of outer iterations and m is that of inner iterations as defined in Algorithm 4.3. Let s be the outer iterations counter. As for m, it is set at $2n$ like in [53]. Then the stepsize γ in SVRP is fixed at $1/(5L)$. For SAPA, we run it for $N-n$ iterations because there is a full gradient computation at the beginning of the algorithm. The SAPA stepsize is set to $1/(5L)$. Finally, the SPPA stepsize is is chosen to be $\gamma_k = 1/(k^{0.55})$.

For all three algorithms, we normalize the abscissa so to present the convergence with respect to the outer iterates $(\tilde{\boldsymbol{x}}^s)_{s\in\mathbb{N}}$ as in Theorem 4.11.

The algorithms are run for $n \in \{1000, 5000, 10000\}$, d fixed at 50 and $cond(A^\top A)$, the condition number of $A^\top A$, at 100.

Logistic regression

First, we consider experiments on the logistic loss:

$$F(\mathbf{x}) = \frac{1}{n}\sum_{i=1}^{n} \log(1+\exp\{-b_i\langle \mathsf{a}_i, \mathbf{x}\rangle\}),$$

where a_i is the i^{th} row of a matrix $A \in \mathbb{R}^{n\times d}$ and $b_i \in \{-1,1\}$ for all $i \in [n]$. If we set $f_i(\mathbf{x}) = \log(1+\exp\{-b_i\langle \mathsf{a}_i, \mathbf{x}\rangle\})$ for all $i \in [n]$, then $F(\mathbf{x}) = \sum_{i=1}^n \frac{1}{n} f_i(\mathbf{x})$. The matrix A is generated randomly. We first generate a matrix M according to the standard normal distribution. A singular value decomposition gives $M = UDV^\top$. We set the smallest singular value to zero and rescale the rest of the vector of singular values so that the biggest singular value is equal to a given condition number and the second smallest to one. We obtain a new diagonal matrix D'. Then A is given by $UD'V^\top$. In this problem, we have $L = 0.25 \times \max_i \|\mathsf{a}_i\|_2^2$. We compute the proximity operator of the logistic function f_i according to the formula and the subroutine code available in [23] and considering the rule of calculus of the proximity operator of a function composed with a linear map [8, Corollary 24.15].

Even though we don't have any theoretical result for SVRP in the convex case, we perform some experiments in this case as well. As it can readily be seen in Figure 4.1, both SAPA and SVRP are better than SPPA.

Ordinary least squares (OLS)

To analyze the practical behavior of the proposed methods when the PL condition holds, we test all the algorithms on an ordinary least squares problem:

$$\underset{\mathbf{x}\in\mathsf{H}}{\text{minimize}} \quad F(\mathbf{x}) = \frac{1}{2n}\|A\mathbf{x}-\boldsymbol{b}\|^2 = \frac{1}{n}\sum_{i=1}^{n}\frac{1}{2}\left(\langle \mathsf{a}_i, \mathbf{x}\rangle - b_i\right)^2, \tag{4.5.1}$$

where a_i is the i^{th} row of the matrix $A \in \mathbb{R}^{n\times d}$ and $b_i \in \mathbb{R}$ for all $i \in [n]$. In this setting, we have $F(\mathbf{x}) = \sum_{i=1}^n \frac{1}{n} f_i(\mathbf{x})$ with $f_i(\mathbf{x}) = \frac{1}{2}\left(\langle \mathsf{a}_i, \mathbf{x}\rangle - b_i\right)^2$ for all $i \in [n]$.

Here, the matrix A was generated as in Section 4.5.1. The proximity operator is computed with the following closed form solution:

$$\mathsf{prox}_{\gamma f_i}(\boldsymbol{x}) = \boldsymbol{x} + \gamma\frac{b_i - \langle \mathsf{a}_i, \boldsymbol{x}\rangle}{\gamma\|\mathsf{a}_i\|^2+1}\mathsf{a}_i.$$

Like in the logistic case, SVRP and SAPA exhibit faster convergence compared to SPPA. See Figure 4.2.

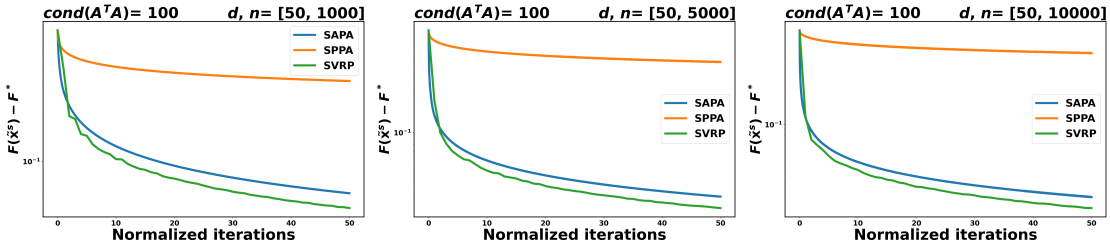


Figure 4.1: Evolution of $F(\tilde{\boldsymbol{x}}^s) - F_*$, with respect to the normalized iterations counter, for different values of number of functions n.We compare the performance of SAPA (blue), SVRP (green) and SPPA (orange) in the logistic regression case.

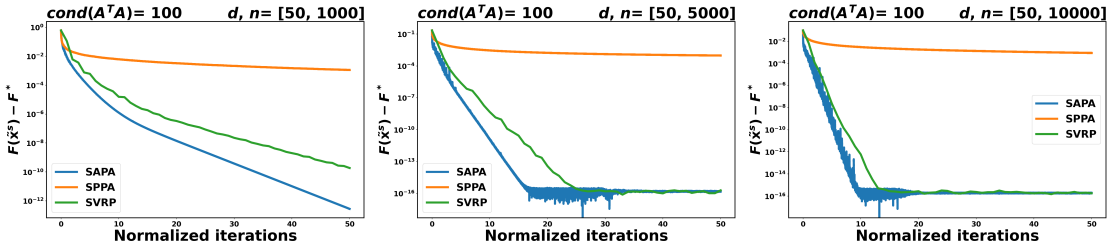


Figure 4.2: Evolution of $F(\tilde{\boldsymbol{x}}^s) - F_*$, with respect to the normalized iterations counter, for different values of number of functions n. We compare the performance of SAPA (blue), SVRP (green) and SPPA (orange) in the ordinary least squares case.

4.5.2 Comparing SAPA to SAGA

In the next experiments, we implement SAPA and SAGA to solve the least-squares problem (4.5.1). The data are exactly as in Section 4.5.1 and the matrix A is generated as explained in Section 4.5.1.

The aim of these experiments is twofold: on the one hand we aim to establish the practical performance of the proposed method in terms of convergence rate and to compare it with the corresponding variance reduction gradient algorithm, on the other hand we want to assess the stability of SAPA with respect to the stepsize selection. Indeed, SPPA has been shown to be more robust and stable with respect to the choice of the stepsize than the stochastic gradient algorithms, see [5, 57, 88]. In Figure 4.3 we plot the number of iterations that are needed for SAPA and SAGA to achieve an accuracy at least $F(x_t) - F_* \leq \varepsilon = 0.01$, along a fixed range of stepsizes. The algorithms are run for $n \in \{1000, 5000, 10000\}$, d fixed at 50 and $cond(A^\top A)$ at 100.

Our experimental results show that if the stepsize is small enough, SAPA and SAGA behave very similarly, see Figure 4.3. This behavior is in agreement with the theoretical results that establish convergence rates for SAPA that are similar to those of SAGA. By increasing the stepsizes, we observe that SAPA is more stable than SAGA: the range of stepsizes for which SAPA converges is wider that the one for SAGA.

4.5.3 Comparing SVRP to SVRG

In Figure 4.4 we compare the performance of Algorithm 4.3 (SVRP) with SVRG [53] for the least-squares problem (4.5.1), in terms of number of inner iterations (oracle calls) to achieve an accuracy at least $F(x_t) - F_* \leq \varepsilon = 0.01$, along a fixed range of stepsizes. In this experiment, the condition number is set cond$(A^\top A) = 100$, $n = 2000$ and the dimension of the objective variable d varies in $\{500, 1000, 1500, 2000\}$. The number of outer and inner iterations is set to $s = 40$ and $m = 1000$ respectively and the maximum number of iterations (oracle calls) is set to $N = s(m + n + 1)$. The results are reported in Figure 4.4.

Two main observations can be made. First we note that when the step-size is optimally tuned, SVRP performs always better than SVRG (overall, it requires less iterations to achieve an $\varepsilon = 0.01$ accuracy). Secondly, regarding the stability of the two methods with respect to the choice of the stepsize, while SVRG seems to be a bit more stable for easy problems ($d = 500$), the situation is reversed for harder problems (e.g. $d \in \{1500, 2000\}$), where SVRP is more robust.

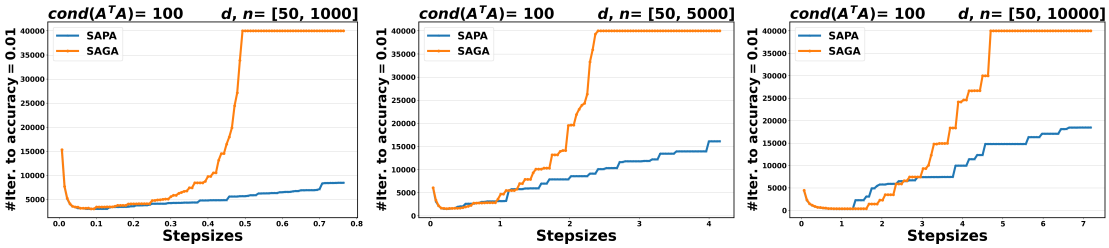


Figure 4.3: Number of iterations needed in order to achieve an accuracy of at least 0.01 for different stepsizes. A cap is put at 40000 iterations. Here we compare SAPA (in blue) with SAGA (in orange) for different value of the number of functions n.

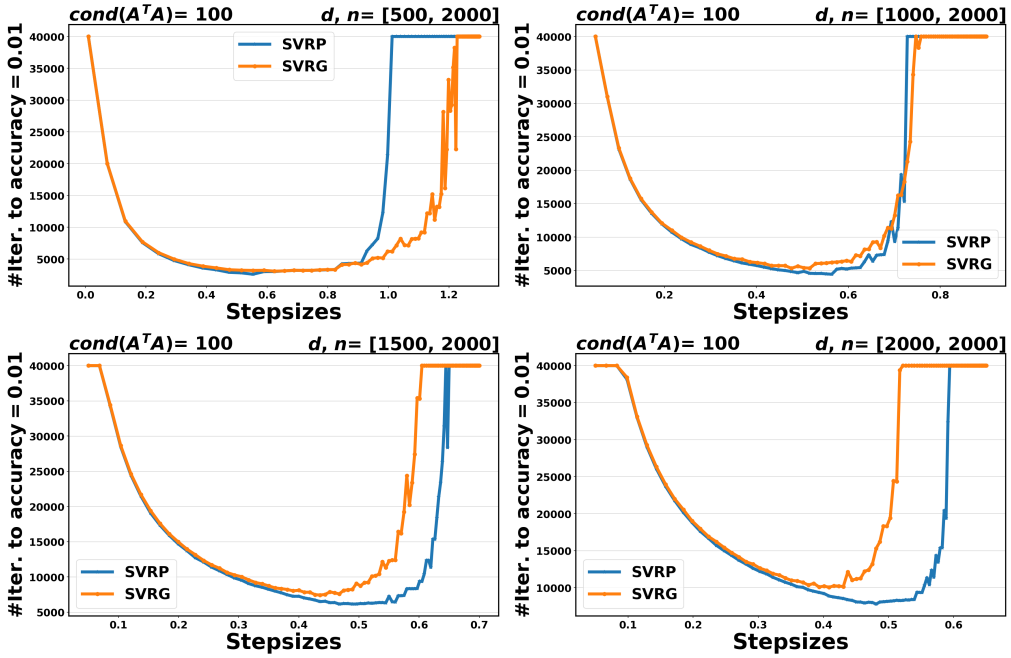


Figure 4.4: Number of iterations needed in order to achieve an accuracy of at least 0.01 for different stepsizes. Here we compare SVRP (in blue) with SVRG (in orange) for four different values of the dimension d in problem (4.5.1), starting from $d = 500$ (easy case) to $d = 2000$ (hard case).

Appendix for Chapter 4

4.6 Additional proofs

In this appendix we provide the proofs of some auxiliary lemmas used in Sections 4.3 and 4.4 in the main core of this chapter.

4.6.1 Proofs of Section 3

Proof of Proposition 4.3. Notice that from (4.2.1), $\boldsymbol{x}^{k+1}$ can be identified as

$$\boldsymbol{x}^{k+1} = \underset{\mathbf{x}\in\mathsf{H}}{\operatorname{argmin}}\{\underbrace{f_{i_k}(\boldsymbol{x}) - \langle \boldsymbol{e}^k, \mathbf{x} - \boldsymbol{x}^k\rangle}_{:=R_{i_k}(\mathbf{x})} + \frac{1}{2\gamma}\|\mathbf{x} - \boldsymbol{x}^k\|^2\}.$$

Let $\mathbf{x} \in \mathsf{H}$. The function $R_{i_k}(\mathbf{x}) = f_{i_k}(\mathbf{x}) - \langle \boldsymbol{e}^k, \mathbf{x} - \boldsymbol{x}^k\rangle$ is convex and continuously differentiable with $\nabla R_{i_k}(\mathbf{x}) = \nabla f_{i_k}(\mathbf{x}) - \boldsymbol{e}^k$

By convexity of R_{i_k}, we have:

$$\begin{aligned}\langle \nabla R_{i_k}(\boldsymbol{x}^{k+1}), \mathbf{x} - \boldsymbol{x}^{k+1}\rangle &\leq R_{i_k}(\mathbf{x}) - R_{i_k}(\boldsymbol{x}^{k+1}) \\ &= f_{i_k}(\mathbf{x}) - f_{i_k}(\boldsymbol{x}^{k+1}) - \langle \boldsymbol{e}^k, \mathbf{x} - \boldsymbol{x}^k\rangle + \langle \boldsymbol{e}^k, \boldsymbol{x}^{k+1} - \boldsymbol{x}^k\rangle.\end{aligned} \tag{4.6.1}$$

Since $\nabla R_{i_k}(\boldsymbol{x}^{k+1}) = \frac{\boldsymbol{x}^k - \boldsymbol{x}^{k+1}}{\gamma}$, from (4.6.1), it follows:

$$\frac{1}{\gamma}\langle \boldsymbol{x}^k - \boldsymbol{x}^{k+1}, \mathbf{x} - \boldsymbol{x}^{k+1}\rangle \leq f_{i_k}(\mathbf{x}) - f_{i_k}(\boldsymbol{x}^{k+1}) - \langle \boldsymbol{e}^k, \mathbf{x} - \boldsymbol{x}^k\rangle + \langle \boldsymbol{e}^k, \boldsymbol{x}^{k+1} - \boldsymbol{x}^k\rangle.$$

By using the identity $\langle \boldsymbol{x}^k - \boldsymbol{x}^{k+1}, \mathbf{x} - \boldsymbol{x}^{k+1}\rangle = \frac{1}{2}\|\boldsymbol{x}^{k+1} - \boldsymbol{x}^k\|^2 + \frac{1}{2}\|\boldsymbol{x}^{k+1} - \mathbf{x}\|^2 - \frac{1}{2}\|\boldsymbol{x}^k - \mathbf{x}\|^2$ in the previous inequality, we find

$$\begin{aligned}-\langle \boldsymbol{e}^k, \boldsymbol{x}^{k+1} - \boldsymbol{x}^k\rangle + f_{i_k}(\boldsymbol{x}^{k+1}) - f_{i_k}(\mathbf{x}) + \frac{1}{2\gamma}\|\boldsymbol{x}^{k+1} - \boldsymbol{x}^k\|^2 \\ \leq \frac{1}{2\gamma}\|\boldsymbol{x}^k - \mathbf{x}\|^2 - \frac{1}{2\gamma}\|\boldsymbol{x}^{k+1} - \mathbf{x}\|^2 - \langle \boldsymbol{e}^k, \mathbf{x} - \boldsymbol{x}^k\rangle.\end{aligned} \tag{4.6.2}$$

Recalling the definition of $\boldsymbol{v}^k$ in (4.2.2), we can lower bound the left hand term as follows:

$$\begin{aligned}
&-\langle \boldsymbol{e}^k, \boldsymbol{x}^{k+1} - \boldsymbol{x}^k\rangle + f_{i_k}(\boldsymbol{x}^{k+1}) - f_{i_k}(\mathbf{x}) + \frac{1}{2\gamma}\|\boldsymbol{x}^{k+1} - \boldsymbol{x}^k\|^2 \\
&\quad = -\langle \boldsymbol{e}^k, \boldsymbol{x}^{k+1} - \boldsymbol{x}^k\rangle + f_{i_k}(\boldsymbol{x}^{k+1}) - f_{i_k}(\boldsymbol{x}^k) + \frac{1}{2\gamma}\|\boldsymbol{x}^{k+1} - \boldsymbol{x}^k\|^2 \\
&\qquad + f_{i_k}(\boldsymbol{x}^k) - f_{i_k}(\mathbf{x}) \\
&\quad \geq -\langle \boldsymbol{e}^k, \boldsymbol{x}^{k+1} - \boldsymbol{x}^k\rangle + \langle \nabla f_{i_k}(\boldsymbol{x}^k), \boldsymbol{x}^{k+1} - \boldsymbol{x}^k\rangle + \frac{1}{2\gamma}\|\boldsymbol{x}^{k+1} - \boldsymbol{x}^k\|^2 \\
&\qquad + f_{i_k}(\boldsymbol{x}^k) - f_{i_k}(\mathbf{x}) \\
&\quad = \langle -\boldsymbol{e}^k + \nabla f_{i_k}(\boldsymbol{x}^k), \boldsymbol{x}^{k+1} - \boldsymbol{x}^k\rangle + \frac{1}{2\gamma}\|\boldsymbol{x}^{k+1} - \boldsymbol{x}^k\|^2 \\
&\qquad + f_{i_k}(\boldsymbol{x}^k) - f_{i_k}(\mathbf{x}) \\
&\quad = \langle \boldsymbol{v}^k, \boldsymbol{x}^{k+1} - \boldsymbol{x}^k\rangle + \frac{1}{2\gamma}\|\boldsymbol{x}^{k+1} - \boldsymbol{x}^k\|^2 \\
&\qquad + f_{i_k}(\boldsymbol{x}^k) - f_{i_k}(\mathbf{x}),
\end{aligned} \tag{4.6.3}$$

where in the first inequality, we used the convexity of f_i, for all $i \in [n]$. Since

$$\langle \boldsymbol{v}^k, \boldsymbol{x}^{k+1} - \boldsymbol{x}^k\rangle + \frac{1}{2\gamma}\|\boldsymbol{x}^{k+1} - \boldsymbol{x}^k\|^2 = \left\|\frac{\sqrt{\gamma}}{\sqrt{2}}\boldsymbol{v}^k + \frac{1}{\sqrt{2\gamma}}(\boldsymbol{x}^{k+1} - \boldsymbol{x}^k)\right\|^2 - \frac{\gamma}{2}\|\boldsymbol{v}^k\|^2.$$

From (4.6.3), it follows

$$\begin{aligned}
&-\langle \boldsymbol{e}^k, \boldsymbol{x}^{k+1} - \boldsymbol{x}^k\rangle + f_{i_k}(\boldsymbol{x}^{k+1}) - f_{i_k}(\mathbf{x}) + \frac{1}{2\gamma}\|\boldsymbol{x}^{k+1} - \boldsymbol{x}^k\|^2 \\
&\qquad \geq \left\|\frac{\sqrt{\gamma}}{\sqrt{2}}\boldsymbol{v}^k + \frac{1}{\sqrt{2\gamma}}(\boldsymbol{x}^{k+1} - \boldsymbol{x}^k)\right\|^2 - \frac{\gamma}{2}\|\boldsymbol{v}^k\|^2 + f_{i_k}(\boldsymbol{x}^k) - f_{i_k}(\mathbf{x}) \\
&\qquad = \frac{\gamma}{2}\left\|\nabla f_{i_k}(\boldsymbol{x}^k) - \nabla f_{i_k}(\boldsymbol{x}^{k+1})\right\|^2 - \frac{\gamma}{2}\|\boldsymbol{v}^k\|^2 + f_{i_k}(\boldsymbol{x}^k) - f_{i_k}(\mathbf{x}).
\end{aligned} \tag{4.6.4}$$

By using (4.6.4) in (4.6.2), we obtain

$$-\frac{\gamma}{2}\|\boldsymbol{v}^k\|^2 + f_{i_k}(\boldsymbol{x}^k) - f_{i_k}(\mathbf{x}) \leq \frac{1}{2\gamma}\|\boldsymbol{x}^k - \mathbf{x}\|^2 - \frac{1}{2\gamma}\|\boldsymbol{x}^{k+1} - \mathbf{x}\|^2 - \langle \boldsymbol{e}^k, \mathbf{x} - \boldsymbol{x}^k\rangle. \tag{4.6.5}$$

Now, define $\mathsf{E}_k[\cdot] = \mathsf{E}[\cdot \,|\, \mathfrak{F}_k]$, where $\mathfrak{F}_k$ is defined in Assumptions 4.2 and is such that $\boldsymbol{x}^k$ is $\mathfrak{F}_k$-measurable and i_k is independent of $\mathfrak{F}_k$. Thus, taking the conditional expectation of inequality 4.6.5 and rearranging the terms, we have

$$\begin{aligned}
\mathsf{E}_k[\|\boldsymbol{x}^{k+1} - \mathbf{x}\|^2] &\leq \|\boldsymbol{x}^k - \mathbf{x}\|^2 - 2\gamma\mathsf{E}_k\left[f_{i_k}(\boldsymbol{x}^k) - f_{i_k}(\mathbf{x})\right] + \gamma^2\mathsf{E}_k\|\boldsymbol{v}^k\|^2 \\
&= \|\boldsymbol{x}^k - \mathbf{x}\|^2 - 2\gamma(F(\boldsymbol{x}^k) - F(\mathbf{x})) + \gamma^2\mathsf{E}_k\|\boldsymbol{v}^k\|^2.
\end{aligned}$$

Replacing $\mathbf{x}$ by y^k, with $y^k \in \operatorname{argmin} F$ such that $\|\boldsymbol{x}^k - y^k\| = \operatorname{dist}(\boldsymbol{x}^k, \operatorname{argmin} F)$ and using Assumption (B.ii), we get

$$\begin{aligned}
\mathsf{E}_k\|\boldsymbol{x}^{k+1} - y^k\|^2 &\leq \|\boldsymbol{x}^k - y^k\|^2 \\
&\qquad - 2\gamma(F(\boldsymbol{x}^k) - F_*) + \gamma^2\left(2A(F(\boldsymbol{x}^k) - F_*) + B\sigma_k^2 + D\right) \\
&= \|\boldsymbol{x}^k - y^k\|^2 + \gamma^2 D \\
&\qquad - 2\gamma(1 - \gamma A)(F(\boldsymbol{x}^k) - F_*) + \gamma^2 B\sigma_k^2
\end{aligned} \tag{4.6.6}$$

By taking the total expectation in (4.6.6) and using (B.iii), for all $M > 0$ we have

$$\begin{aligned}
\mathsf{E}\|\boldsymbol{x}^{k+1} - y^k\|^2 + \gamma^2 M \mathsf{E}[\sigma_{k+1}^2] &\leq \mathsf{E}\|\boldsymbol{x}^k - y^k\|^2 + \gamma^2 \mathsf{E}[D] \\
&\quad - 2\gamma(1-\gamma A)\mathsf{E}[F(\boldsymbol{x}^k) - F_*] + \gamma^2 B \mathsf{E}[\sigma_k^2] \\
&\quad + \gamma^2 M(1-\rho)\mathsf{E}[\sigma_k^2] + 2\gamma^2 MC\mathsf{E}[F(\boldsymbol{x}^k) - F_*] \\
&= \mathsf{E}\|\boldsymbol{x}^k - y^k\|^2 + \gamma^2 \mathsf{E}[D] \\
&\quad - 2\gamma\left[1 - \gamma(A+MC)\right]\mathsf{E}[F(\boldsymbol{x}^k) - F_*] \\
&\quad + \gamma^2\left[M + B - \rho M\right]\mathsf{E}[\sigma_k^2] \\
&= \mathsf{E}[\operatorname{dist}(\boldsymbol{x}^k, \operatorname{argmin} F)^2] + \gamma^2 \mathsf{E}[D] \\
&\quad - 2\gamma\left[1 - \gamma(A+MC)\right]\mathsf{E}[F(\boldsymbol{x}^k) - F_*] \\
&\quad + \gamma^2\left[M + B - \rho M\right]\mathsf{E}[\sigma_k^2].
\end{aligned}$$

Since $\mathsf{E}[\operatorname{dist}(\boldsymbol{x}^{k+1}, \operatorname{argmin} F)^2] \leq \mathsf{E}\|\boldsymbol{x}^{k+1} - y^k\|^2$, the statement follows. □

4.6.2 Proofs of Section 4

Proof of Lemma 4.9. Given any $i \in [n]$, since f_i is convex and L-Lipschitz smooth, it is a standard fact (see [77, Equation 2.1.7]) that

$$(\forall\, \mathbf{x}, \mathbf{y} \in \mathsf{H}) \quad \left\|\nabla f_i(\mathbf{x}) - \nabla f_i(\mathbf{y})\right\|^2 \leq 2L\big[f_i(\mathbf{x}) - f_i(\mathbf{y}) - \langle \nabla f_i(\mathbf{y}), \mathbf{x} - \mathbf{y}\rangle\big].$$

By summing the above inequality over $i = 1, \ldots, n$, we have

$$\sum_{i=1}^n \frac{1}{n}\left\|\nabla f_i(\mathbf{x}) - \nabla f_i(\mathbf{y})\right\|^2 \leq 2L\big[F(\mathbf{x}) - F(\mathbf{y})\big] - 2L\langle \nabla F(\mathbf{y}), \mathbf{x} - \mathbf{y}\rangle$$

and hence if we suppose that $\nabla F(\mathbf{y}) = 0$, then we obtain

$$\sum_{i=1}^n \frac{1}{n}\left\|\nabla f_i(\mathbf{x}) - \nabla f_i(\mathbf{y})\right\|^2 \leq 2L\big[F(\mathbf{x}) - F(\mathbf{y})\big]. \tag{4.6.7}$$

Now, let $s \in \mathbb{N}$ and $k \in \{0, \ldots, m-1\}$ and set, for the sake of brevity, $j_k = i_{sm+k}$. Defining $\mathfrak{F}_k = \mathfrak{F}_{s,k} = \sigma(\xi_0, \ldots, \xi_{s-1}, i_0, \ldots, i_{sm+k-1})$ and $\mathsf{E}_k[\cdot] = \mathsf{E}[\cdot \,|\, \mathfrak{F}_k]$, and recalling that

$$\boldsymbol{v}^k = \nabla f_{j_k}(\boldsymbol{x}^k) - \nabla f_{j_k}(\tilde{\boldsymbol{x}}^s) + \nabla F(\tilde{\boldsymbol{x}}^s) \quad \text{and} \quad \tilde{y}^s = P_{\operatorname{argmin} F}(\tilde{\boldsymbol{x}}^s),$$

we obtain

$$\begin{aligned}
\mathsf{E}_k\big[\|\boldsymbol{v}^k\|^2\big] &\leq 2\mathsf{E}_k\left\|\nabla f_{j_k}(\boldsymbol{x}^k) - \nabla f_{j_k}(\tilde{y}^s)\right\|^2 + 2\mathsf{E}_k\left\|[\nabla f_{j_k}(\tilde{\boldsymbol{x}}^s) - \nabla f_{j_k}(\tilde{y}^s)] - \nabla F(\tilde{\boldsymbol{x}}^s)\right\|^2 \\
&= 2\mathsf{E}_k\left\|[\nabla f_{j_k}(\tilde{\boldsymbol{x}}^s) - \nabla f_{j_k}(\tilde{y}^s)] - \mathsf{E}_k[\nabla f_{j_k}(\tilde{\boldsymbol{x}}^s) - \nabla f_{j_k}(\tilde{y}^s)]\right\|^2 \\
&\quad + 2\mathsf{E}_k\left\|\nabla f_{j_k}(\boldsymbol{x}^k) - \nabla f_{j_k}(\tilde{y}^s)\right\|^2 \\
&= 2\mathsf{E}_k\left\|\nabla f_{j_k}(\tilde{\boldsymbol{x}}^s) - \nabla f_{j_k}(\tilde{y}^s)\right\|^2 - 2\left\|\mathsf{E}_k[\nabla f_{j_k}(\tilde{\boldsymbol{x}}^s) - \nabla f_{j_k}(\tilde{y}^s)]\right\|^2 \\
&\quad + 2\mathsf{E}_k\left\|\nabla f_{j_k}(\boldsymbol{x}^k) - \nabla f_{j_k}(\tilde{y}^s)\right\|^2 \\
&= 2\mathsf{E}_k\left\|\nabla f_{j_k}(\boldsymbol{x}^k) - \nabla f_{j_k}(\tilde{y}^s)\right\|^2 + 2\mathsf{E}_k\left\|\nabla f_{j_k}(\tilde{\boldsymbol{x}}^s) - \nabla f_{j_k}(\tilde{y}^s)\right\|^2 \\
&\quad - 2\|\nabla F(\tilde{\boldsymbol{x}}^s)\|^2 \\
&\leq 4L\big(F(\boldsymbol{x}^k) - F(\tilde{y}^s)\big) + 2\sigma_k^2 - 2\|\nabla F(\tilde{\boldsymbol{x}}^s)\|^2,
\end{aligned}$$

where in the last inequality, we used (4.6.7) with $\mathbf{y} = \tilde{y}^s$. □

Proof of Lemma 4.13 The proof of equation (4.4.11) is equal to that of (4.4.5) in Lemma 4.9 using $\boldsymbol{u}^k$ instead of $\tilde{\boldsymbol{x}}^s$ and $\mathsf{E}_k[\cdot] = \mathsf{E}[\cdot\,|\,\mathfrak{F}_k]$ with $\mathfrak{F}_k = \sigma(i_0, \ldots, i_{k-1}, \varepsilon^0, \ldots, \varepsilon^{k-1})$, which ensures that $\boldsymbol{x}^k$ and $\boldsymbol{u}^k$ are $\mathfrak{F}_k$-measurables, and i_k and ε^k are independent of $\mathfrak{F}_k$. Concerning (4.4.12), we note that

$$\sigma_{k+1}^2 = \frac{1}{n}\sum_{i=1}^n \left\|\nabla f_i((1-\varepsilon^k)\boldsymbol{u}^k + \varepsilon^k \boldsymbol{x}^k) - \nabla f_i(P_{\operatorname{argmin} F}((1-\varepsilon^k)\boldsymbol{u}^k + \varepsilon^k \boldsymbol{x}^k))\right\|^2.$$

Therefore, we have

$$\begin{aligned}\mathsf{E}_k\left[\sigma_{k+1}^2\right] &= (1-p)\sigma_k^2 + p\frac{1}{n}\sum_{i=1}^n \left\|\nabla f_i(\boldsymbol{x}^k) - \nabla f_i(P_{\operatorname{argmin} F}(\boldsymbol{x}^k))\right\|^2 \\ &\leq (1-p)\sigma_k^2 + 2pL(F(\boldsymbol{x}^k) - F^*),\end{aligned}$$

where in the last inequality we used (4.6.7) with $\mathbf{y} = P_{\operatorname{argmin} F}(\boldsymbol{x}^k)$. □

Proof of Lemma 4.16 We recall that, by definition,

$$(\forall\, i \in [n]) \quad \boldsymbol{\phi}_i^{k+1} = \boldsymbol{\phi}_i^k + \delta_{i,i_k}(\boldsymbol{x}^k - \boldsymbol{\phi}_i^k).$$

Now, set $\mathsf{E}_k[\cdot] = \mathsf{E}[\cdot\,|\,\mathfrak{F}_k]$ with $\mathfrak{F}_k = \sigma(i_0, \ldots, i_{k-1})$, so that $\boldsymbol{\phi}_i^k$ and $\boldsymbol{x}^k$ are $\mathfrak{F}_k$-measurable and i_k is independent of $\mathfrak{F}_k$. By definition of $\boldsymbol{v}^k$, and using the fact that $\nabla F(\boldsymbol{x}_*) = 0$ and inequality (4.6.7), we have

$$\begin{aligned}\mathsf{E}_k\left[\|\boldsymbol{v}^k\|^2\right] &= \mathsf{E}_k\left[\left\|\nabla f_{i_k}(\boldsymbol{x}^k) - \nabla f_{i_k}(\boldsymbol{\phi}_{i_k}^k) + \frac{1}{n}\sum_{i=1}^n \nabla f_i(\boldsymbol{\phi}_i^k) - \nabla F(\boldsymbol{x}_*)\right\|^2\right] \\ &= \mathsf{E}_k\Bigg[\Big\|\nabla f_{i_k}(\boldsymbol{x}^k) - \nabla f_{i_k}(\boldsymbol{x}_*) + \nabla f_{i_k}(\boldsymbol{x}_*) - \nabla f_{i_k}(\boldsymbol{\phi}_{i_k}^k) \\ &\qquad + \frac{1}{n}\sum_{i=1}^n \nabla f_i(\boldsymbol{\phi}_i^k) - \nabla F(\boldsymbol{x}_*)\Big\|^2\Bigg] \\ &\leq 2\mathsf{E}_k\left[\left\|\nabla f_{i_k}(\boldsymbol{x}^k) - \nabla f_{i_k}(\boldsymbol{x}_*)\right\|^2\right] \\ &\qquad + 2\mathsf{E}_k\left[\left\|\nabla f_{i_k}(\boldsymbol{x}_*) - \nabla f_{i_k}(\boldsymbol{\phi}_{i_k}^k) - \mathsf{E}_k[\nabla f_{i_k}(\boldsymbol{x}_*) - \nabla f_{i_k}(\boldsymbol{\phi}_{i_k}^k)]\right\|^2\right] \\ &= \frac{2}{n}\sum_{i=1}^n \left\|\nabla f_i(\boldsymbol{x}^k) - \nabla f_i(\boldsymbol{x}_*)\right\|^2 + 2\mathsf{E}_k\left[\left\|\nabla f_{i_k}(\boldsymbol{x}_*) - \nabla f_{i_k}(\boldsymbol{\phi}_{i_k}^k)\right\|^2\right] \\ &\qquad - 2\left\|\frac{1}{n}\sum_{i=1}^n \nabla f_i(\boldsymbol{\phi}_i^k)\right\|^2 \\ &\leq 4L\big(F(\boldsymbol{x}^k) - F_*\big) + 2\underbrace{\frac{1}{n}\sum_{i=1}^n \left\|\nabla f_i(\boldsymbol{\phi}_i^k) - \nabla f_i(\boldsymbol{x}_*)\right\|^2}_{\sigma_k^2} - 2\left\|\frac{1}{n}\sum_{i=1}^n \nabla f_i(\boldsymbol{\phi}_i^k)\right\|^2.\end{aligned}$$

For the second part (4.4.14), we proceed as follows

$$\begin{aligned}
\mathsf{E}_k\left[\sigma_{k+1}^2\right] &= \frac{1}{n}\sum_{i=1}^{n}\mathsf{E}_k\left[\left\|\nabla f_i(\boldsymbol{\phi}_i^{k+1}) - \nabla f_i(\boldsymbol{x}_*)\right\|^2\right] \\
&= \frac{1}{n}\sum_{i=1}^{n}\mathsf{E}_k\left[\left\|\nabla f_i(\boldsymbol{\phi}_i^{k} + \delta_{i,i_k}(\boldsymbol{x}^k - \boldsymbol{\phi}_i^k)) - \nabla f_i(\boldsymbol{x}_*)\right\|^2\right] \\
&= \frac{1}{n}\sum_{i=1}^{n}\left(\frac{1}{n}\sum_{j=1}^{n}\left\|\nabla f_i(\boldsymbol{\phi}_i^{k} + \delta_{i,j}(\boldsymbol{x}^k - \boldsymbol{\phi}_i^k)) - \nabla f_i(\boldsymbol{x}_*)\right\|^2\right) \\
&= \frac{1}{n}\sum_{i=1}^{n}\left(\frac{n-1}{n}\left\|\nabla f_i(\boldsymbol{\phi}_i^{k}) - \nabla f_i(\boldsymbol{x}_*)\right\|^2 + \frac{1}{n}\left\|\nabla f_i(\boldsymbol{x}^k) - \nabla f_i(\boldsymbol{x}_*)\right\|^2\right) \\
&= \left(1 - \frac{1}{n}\right)\frac{1}{n}\sum_{i=1}^{n}\left\|\nabla f_i(\boldsymbol{\phi}_i^{k}) - \nabla f_i(\boldsymbol{x}_*)\right\|^2 + \frac{2L}{n^2}\sum_{i=1}^{n}\left\|\nabla f_i(\boldsymbol{x}^k) - \nabla f_i(\boldsymbol{x}_*)\right\|^2 \\
&\leq \left(1 - \frac{1}{n}\right)\sigma_k^2 + \frac{2L}{n}\big(F(\boldsymbol{x}^k) - F_*\big),
\end{aligned}$$

where in the last inequality we used inequality (4.6.7) with $\mathbf{y} = \boldsymbol{x}_*$. □

CHAPTER 5

Conclusion and perspectives

The overall goal of our thesis was to tackle some problems of large-scale convex optimization for data science. Over the course of our PhD, we have been focusing on parallelization and stochastic algorithms. Our contribution to these two concepts has been developed in two parts of this document: Chapter 3 for parallelization and Chapter 4 for stochastic algorithms.

5.1 Asynchronous algorithms

In Chapter 3, we presented an asynchronous version of random block-coordinate descent. The algorithm allows for a coordinate-wise stepsize so that it can move along each coordinate according to the Lipschitz constant of the partial derivative of that coordinate. In contrast to the fixed stepsize, where the process moves with respect to the biggest Lipschitz constant, hence the slowest stepsize, a variable stepsize with respect to the blocks makes it possible to move faster along coordinates with a small Lipschitz constant. We also consider a general probability of selection of the coordinates. Thus, the algorithm covers any sort of sampling probability as long as each coordinate has a nonzero probability of being selected at each iteration, i.e., any sampling with replacement.

Remark 5.1: When trying to find the best probability distribution for our results, it turns out the best one is the uniform distribution because one should have $\mathsf{p}_{\min}$ as big as possible. This is different from the importance sampling $\mathsf{p}_i = \dfrac{L_i}{\sum_{j=1}^m L_j}$ for all $i \in [m]$, which was said in [50] to be optimal for coordinate descent without a proximal step. We believe that another analysis to get our results in Chapter 3 in terms of p_i instead of $\mathsf{p}_{\min}$ and $\mathsf{p}_{\max}$ would yield a different optimal probability distribution.

We proved convergence of the iterates and convergence rates of the function value in different settings. One of the key takeaways of our analysis is that it confirms that the dependence of the stepsize and the convergence on the delay is at most linear.

We piggyback on that last point to say that an interesting research direction is to see if this dependence can be improved and how exactly the delay affect the rate. To that end, we use Performance Estimation Problem (PEP) python toolkit, PEPit [45], to test the worst case convergence of the following delayed gradient algorithm with a delay $\tau \in \mathbb{N}$ at each iteration:

$$\mathsf{x}_{k+1} = \mathsf{x}_k - h_k \nabla F(\mathsf{x}_0) \text{ for } 0 \leq k \leq \tau,$$

and

$$\mathsf{x}_{k+1} = \mathsf{x}_k - h_k \nabla F(\mathsf{x}_{k-\tau}) \text{ for } k > \tau. \tag{DGD}$$

The initial experiments suggest that the delayed gradient descent above and the normal GD have the same worst case convergence rate for sufficiently small stepsize; see Figure 5.3. This is not the case for bigger stepsize, even for the standard stepsize of $1/L$; see Figures 5.1 and 5.2 So we make the following conjecture.

Conjecture 5.2 (C. Traoré, F. Glineur and S. Villa)**.** *Let $F\colon \mathbb{R}^d \to \mathbb{R}$ be convex and L-smooth. Suppose that $(\mathsf{x}_k)_{k\in\mathbb{N}}$ is generated by DGD and $\|\mathsf{x}_0 - x^*\| \leq R$. If $h_k = \frac{h}{L} \leq \frac{1}{L(2\tau+1)}$, we have, for $k \in \mathbb{N}$,*

$$F(\mathsf{x}_k) - F^* \leq \frac{LR^2}{2}\frac{1}{2hk+1}.$$

Remark 5.3:

(i) If the Conjecture 5.2 is proven true with the stepsize depending linearly on τ, it will validate the dependence on τ of our results in Chapter 3.

(ii) If we considered the one coordinate case of Algorithm 3.1, which corresponds to DGD, Theorem 3.11 gives, with $\gamma = h_k = \frac{h}{L} = \frac{1}{L(2\tau+1)}$, the following rate:

$$F(\mathsf{x}_k) - F^* \leq \frac{1}{k}\left(\frac{LR^2}{2h} + 2\tau(F(\mathsf{x}_0) - F^*)\right),$$

The constants are not as tight as those of Conjecture 5.2, which are known as the best that a gradient method can achieve.

If we extrapolate this conjecture to our Algorithm 3.1, it means, with better arguments, that we might be able to get, at least, the same worst case convergence rates as [90], the synchronous version.

Another research direction is to study accelerated versions of our algorithm. Finally, a more realistic and challenging extension of our work is to remove the assumption of independence between the delay vector and the coordinate. This is a common assumption that is made in the literature. But, it does not cover scenarios where the computation costs of the partial derivatives are different, for example when the data is sparse. So, in those scenarios the delay should dependent on the coordinate.

5.2 Variance reduction techniques

In Chapter 4, we proposed a general scheme of variance reduction, based on the stochastic proximal point algorithm (SPPA), along with its convergence results. Then, we derived, from that generic algorithm, variants of SPPA coupled with various variance reduction strategies, namely SVRP, SAPA and L-SVRP, for which we provide improved convergence results compared to the vanilla version. As the experiments suggest, the convergence behavior of these stochastic proximal variance reduced methods is most of the time more stable with respect to the step-size (empirically we can take bigger stepsizes, especially for higher dimensional problems), making them advantageous over their explicit-gradient counterparts.

Since in this work, differentiability of the summands is required, an interesting line of future research includes the extension of these results to the nonsmooth case. It could also

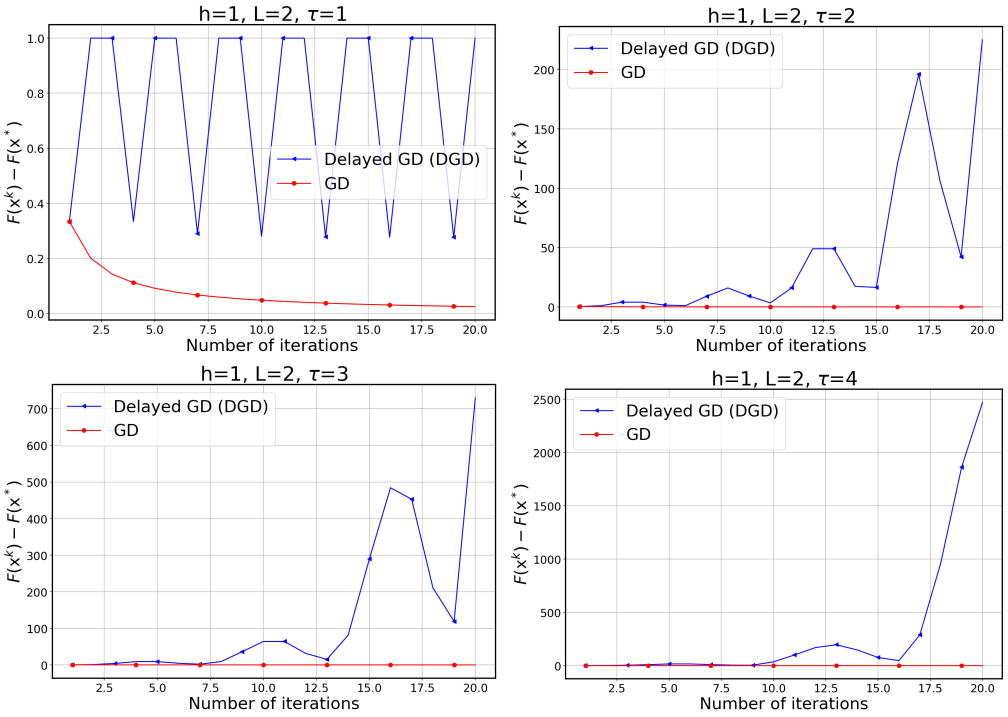


Figure 5.1: Comparison of the worst case convergence rate between GD and DGD for different value of delay $\tau \in \{1, 2, 3, 4\}$ using the standard GD stepsize of $1/L$, which is independent of the delay. We can see that DGD diverges.

be intersesting to consider the model-based framework, as well as inertial (accelerated) variants, hopefully leading to faster rates. Finally, variance reduction can also be useful in the stochastic zeroth order settings. We are currently working in its application to stochastic version of zeroth order descent with structured directions [85]. The problem we are trying to solve is again the finite sum of functions, but now in $\mathbb{R}^d$:

$$\underset{\mathbf{x}\in\mathbb{R}^d}{\text{minimize}} \;\; F(\mathbf{x}) = \frac{1}{n}\sum_{i=1}^{n} f_i(\mathbf{x}).$$

In this setting, we only have access to the functions value. So, we use finite difference approximations of the different gradients. We call by $O(d)$ the space of orthogonal matrix of dimension $d \times d$. Let $h > 0$ and $G \in O(d)$. $\forall\, i \in [n]$, we approximate ∇f_i by

$$(\forall \mathbf{x} \in \mathbb{R}^d) \qquad g_i(\mathbf{x}, h, G) := \frac{d}{\ell}\sum_{j=1}^{\ell} \frac{f_i(\mathbf{x} + hGe_j) - f_i(\mathbf{x})}{h} Ge_j, \tag{5.2.1}$$

where $\ell \leq d$ and $(e_1, e_2, \cdots, e_d)$ is the canonical basis of $\mathbb{R}^d$.

Remark 5.4: Estimating the gradient as in (5.2.1) is useful because it can be proved that in expectation this estimated gradient is the actual gradient of a smooth function f_i^h for every $h > 0$ and $i \in [n]$; see [85, Lemma 1]. This property is particularly handy in the theoretical analysis.

We also set

$$(\forall \mathbf{x} \in \mathbb{R}^d) \qquad g(\mathbf{x}, h, G) := \sum_{i=1}^{n} g_i(\mathbf{x}, h, G)$$

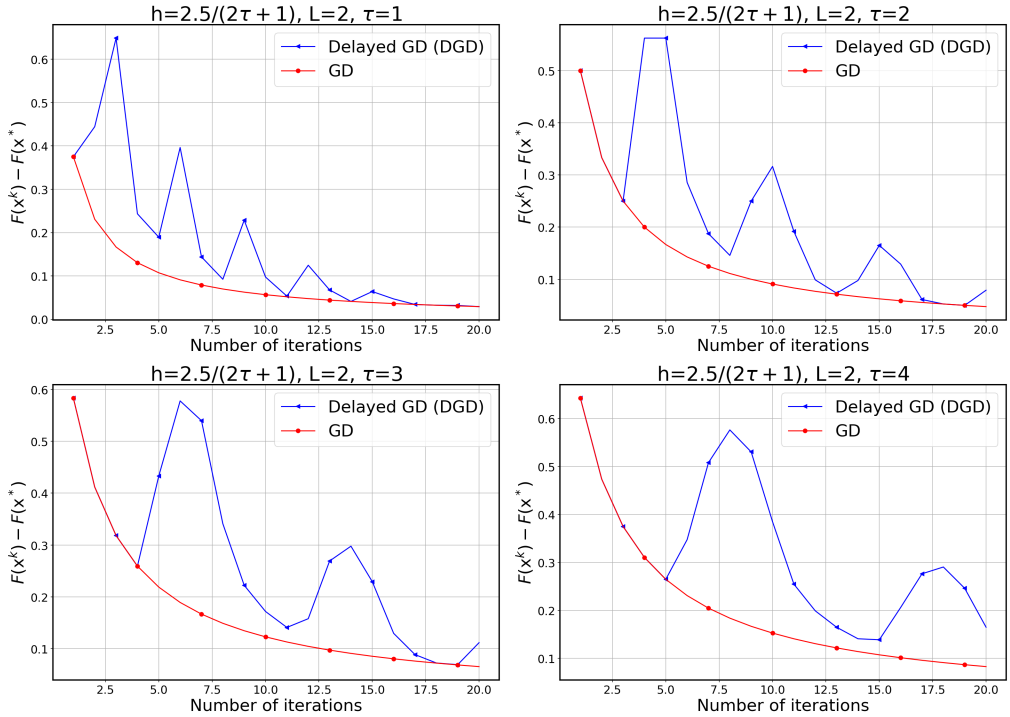


Figure 5.2: Comparison of the worst case convergence rate between GD and DGD for different value of delay $\tau \in \{1, 2, 3, 4\}$ using a stepsize of $\frac{h}{L} = \frac{5/2}{L(2\tau+1)}$. Even if DGD converges, it still does not have the same rate as GD across the board.

The zeroth order SVRG is the following.

Algorithm 5.1 (Zeroth order SVRG):
Let $m \in \mathbb{N}$, with $m \geq 1$, and $(\xi_s)_{s\in\mathbb{N}}$, $(i_t)_{t\in\mathbb{N}}$ be two independent sequences of i.i.d. random variables uniformly distributed on $\{0, 1, \ldots, m-1\}$ and $\{1, \ldots, n\}$ respectively. Let $\gamma > 0$ and set the initial point $\tilde{\boldsymbol{x}}^0 \equiv \tilde{\mathbf{x}}^0 \in \mathsf{H}$. Then

for $s = 0, 1, \ldots$
  sample G_s for $O(d)$ according to the normalized Haar measure
  $\boldsymbol{x}^0 = \tilde{\boldsymbol{x}}^s$
  for $k = 0, \ldots, m-1$
    sample G_k for $O(d)$ according to the normalized Haar measure
    $\boldsymbol{x}^{k+1} = \boldsymbol{x}^k - \gamma \left[g_{i_{sm+k}}(\boldsymbol{x}^k, h, G_k) - g_{i_{sm+k}}(\tilde{\boldsymbol{x}}^s, h, G_k) + g(\tilde{\boldsymbol{x}}^s, h, G_s) \right]$
  $\tilde{\boldsymbol{x}}^{s+1} = \sum_{k=0}^{m-1} \delta_{k,\xi_s} \boldsymbol{x}^k,$

where $\delta_{k,h}$ is the Kronecker symbol.

A simple experiment on linear regression shows that the zeroth order SVRG (S-SVRZD) 5.1 performs better than vanilla structured stochastic zeroth order algorithm (S-SZD); see Figure 5.4. So, this is a natural extension to this thesis.

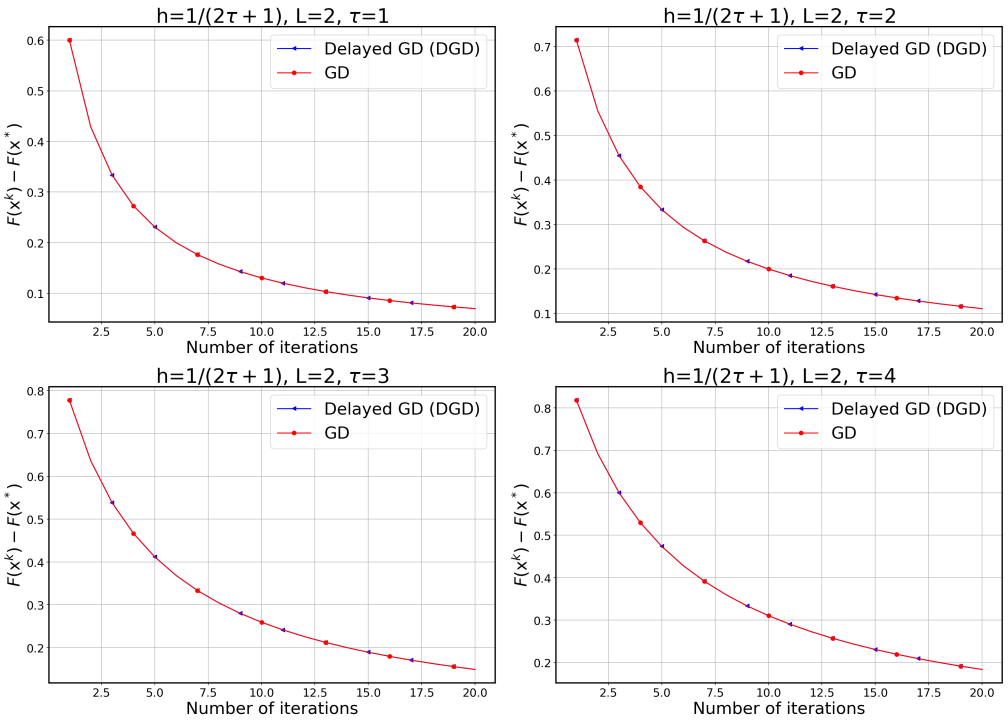


Figure 5.3: Comparison of the worst case convergence rate between GD and DGD for different value of delay $\tau \in \{1, 2, 3, 4\}$. It can be observed that the rate are the same for a small enough stepsize $\frac{h}{L} = \frac{1}{L(2\tau+1)}$.

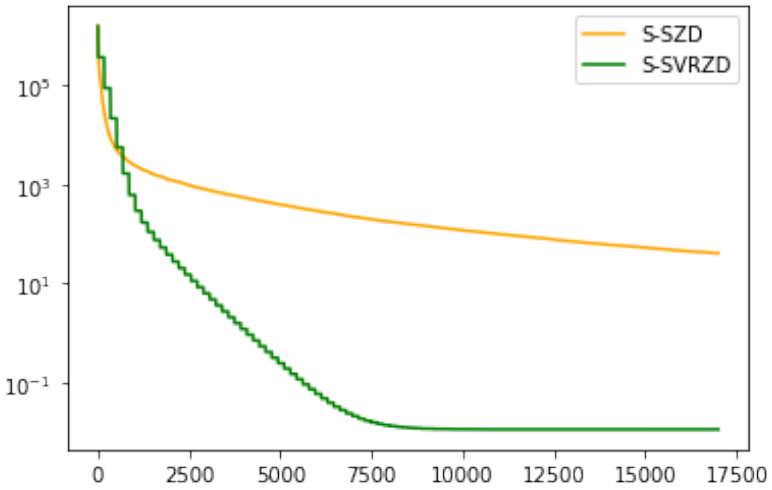


Figure 5.4: Comparison of S-SZD (vanilla) and S-SVRZD (variance reduced) in solving a linear regression problem.

Bibliography

[1] Alekh Agarwal and John C Duchi. “Distributed delayed stochastic optimization”. In: *Advances in neural information processing systems* 24 (2011) (Cited on pp. 28, 44).

[2] Alekh Agarwal and John C. Duchi. “Distributed Delayed Stochastic Optimization”. In: *Proceedings of the 24th International Conference on Neural Information Processing Systems*. NIPS’11. Granada, Spain: Curran Associates Inc., 2011, pp. 873–881. ISBN: 9781618395993 (Cited on p. 2).

[3] Zeyuan Allen-Zhu and Yang Yuan. “Improved SVRG for non-strongly-convex or sum-of-non-convex objectives”. In: *International conference on machine learning*. PMLR. 2016, pp. 1080–1089 (Cited on p. 22).

[4] Vassilis Apidopoulos, Nicolò Ginatta, and Silvia Villa. “Convergence rates for the heavy-ball continuous dynamics for non-convex optimization, under Polyak–Łojasiewicz condition”. In: *Journal of Global Optimization* 84.3 (2022), pp. 563–589 (Cited on p. 64).

[5] Hilal Asi and John C Duchi. “Stochastic (approximate) proximal point methods: convergence, optimality, and adaptivity”. In: *SIAM Journal on Optimization* 29.3 (2019), pp. 2257–2290 (Cited on pp. 25, 63, 67, 68, 76).

[6] Haim Avron, Alex Druinsky, and Anshul Gupta. “Revisiting asynchronous linear solvers: Provable convergence rate through randomization”. In: *Journal of the ACM* 62.6 (2015), pp. 1–27 (Cited on p. 28).

[7] Karl Bäckström, Marina Papatriantafilou, and Philippas Tsigas. “Mindthestep-asyncpsgd: Adaptive asynchronous parallel stochastic gradient descent”. In: *2019 IEEE International Conference on Big Data (Big Data)*. IEEE. 2019, pp. 16–25 (Cited on p. 29).

[8] Heinz H Bauschke and Patrick L Combettes. *Convex Analysis and Monotone Operator Theory in Hilbert Spaces*. Springer New York, NY, 2017 (Cited on pp. 4, 5, 75).

[9] Amir Beck and Marc Teboulle. “A fast iterative shrinkage-thresholding algorithm for linear inverse problems”. In: *SIAM journal on imaging sciences* 2.1 (2009), pp. 183–202 (Cited on p. 42).

[10] Pascal Bégout, Jérôme Bolte, and Mohamed Ali Jendoubi. “On damped second-order gradient systems”. In: *Journal of Differential Equations* 259.7 (2015), pp. 3115–3143 (Cited on p. 64).

[11] Alexandre Belloni, Victor Chernozhukov, and Lie Wang. “Pivotal estimation via square-root Lasso in nonparametric regression”. In: *The Annals of Statistics* 42.2 (2014), pp. 757–788. DOI: 10.1214/14-AOS1204. URL: https://doi.org/10.1214/14-AOS1204 (Cited on p. 42).

[12] Dimitri P Bertsekas. “Incremental proximal methods for large scale convex optimization”. In: *Mathematical programming* 129.2 (2011), pp. 163–195 (Cited on pp. 2, 25, 67).

[13] Dimitri P Bertsekas and John N Tsitsiklis. *Parallel and distributed computation: numerical methods*. Vol. 23. Prentice hall Englewood Cliffs, NJ, 1989 (Cited on pp. 17, 28).

[14] Pascal Bianchi. “Ergodic convergence of a stochastic proximal point algorithm”. In: *SIAM Journal on Optimization* 26.4 (2016), pp. 2235–2260 (Cited on p. 67).

[15] Jérôme Bolte, Aris Daniilidis, and Adrian Lewis. “The Łojasiewicz inequality for nonsmooth subanalytic functions with applications to subgradient dynamical systems”. In: *SIAM Journal on Optimization* 17.4 (2007), pp. 1205–1223 (Cited on p. 64).

[16] Jérôme Bolte et al. “From error bounds to the complexity of first-order descent methods for convex functions”. In: *Mathematical Programming* 165.2 (2017), pp. 471–507 (Cited on pp. 29, 37, 64).

[17] Léon Bottou. “Large-scale machine learning with stochastic gradient descent”. In: *Proceedings of COMPSTAT’2010: 19th International Conference on Computational StatisticsParis France, August 22-27, 2010 Keynote, Invited and Contributed Papers*. Springer. 2010, pp. 177–186 (Cited on p. 19).

[18] Léon Bottou. “On-Line Learning and Stochastic Approximations”. In: *On-Line Learning in Neural Networks*. USA: Cambridge University Press, 1999, pp. 9–42. ISBN: 0521652634 (Cited on p. 19).

[19] Loris Cannelli et al. “Asynchronous parallel algorithms for nonconvex optimization”. In: *Mathematical Programming* 184 (2020), pp. 121–154 (Cited on pp. 28, 33, 44, 45, 46, 47, 48).

[20] Daniel Chazan and Willard Miranker. “Chaotic relaxation”. In: *Linear algebra and its applications* 2.2 (1969), pp. 199–222 (Cited on p. 28).

[21] Scott Shaobing Chen, David L Donoho, and Michael A Saunders. “Atomic decomposition by basis pursuit”. In: *SIAM review* 43.1 (2001), pp. 129–159 (Cited on p. 42).

[22] Shixiang Chen, Alfredo Garcia, and Shahin Shahrampour. "On Distributed Non-convex Optimization: Projected Subgradient Method For Weakly Convex Problems in Networks". In: *IEEE Transactions on Automatic Control* (2021), pp. 1–1. ISSN: 2334-3303. DOI: 10.1109/tac.2021.3056535. URL: http://dx.doi.org/10.1109/TAC.2021.3056535 (Cited on p. 29).

[23] Giovanni Chierchia et al. "The proximity operator repository. User's guide". In: *http://proximity-operator. net/download/guide.pdf* 6 (2020) (Cited on p. 75).

[24] Patrick L Combettes and Jonathan Eckstein. "Asynchronous block-iterative primal-dual decomposition methods for monotone inclusions". In: *Mathematical Programming* 168.1 (2018), pp. 645–672 (Cited on p. 28).

[25] Patrick L Combettes and Jean-Christophe Pesquet. "Stochastic quasi-Fejér block-coordinate fixed point iterations with random sweeping". In: *SIAM Journal on Optimization* 25.2 (2015), pp. 1221–1248 (Cited on p. 30).

[26] Patrick L Combettes and Valérie Wajs. "Signal Recovery by Proximal Forward-Backward Splitting". In: *Multiscale Modeling & Simulation* 4.4 (2005), pp. 1168–1200 (Cited on p. 28).

[27] Patrick L Combettes and Valérie R Wajs. "Signal recovery by proximal forward-backward splitting". In: *Multiscale Modeling & Simulation* 4.4 (2005), pp. 1168–1200 (Cited on p. 42).

[28] Damek Davis. "The asynchronous palm algorithm for nonsmooth nonconvex problems". In: *arXiv preprint arXiv:1604.00526* (2016) (Cited on pp. 28, 29, 33).

[29] Aaron Defazio. "A simple practical accelerated method for finite sums". In: *Advances in neural information processing systems* 29 (2016) (Cited on pp. 4, 62).

[30] Aaron Defazio, Francis Bach, and Simon Lacoste-Julien. "SAGA: A fast incremental gradient method with support for non-strongly convex composite objectives". In: *Advances in neural information processing systems* 27 (2014), pp. 1646–1654 (Cited on pp. 22, 23, 24, 63, 73, 74).

[31] D.L. Donoho. "Compressed sensing". In: *IEEE Transactions on Information Theory* 52.4 (2006), pp. 1289–1306. DOI: 10.1109/TIT.2006.871582 (Cited on p. 42).

[32] Dmitriy Drusvyatskiy and Adrian S Lewis. "Error bounds, quadratic growth, and linear convergence of proximal methods". In: *Mathematics of Operations Research* 43.3 (2018), pp. 919–948 (Cited on pp. 29, 37, 64).

[33] John Duchi, Elad Hazan, and Yoram Singer. "Adaptive Subgradient Methods for Online Learning and Stochastic Optimization". In: *Journal of Machine Learning Research* 12.61 (2011), pp. 2121–2159. URL: http://jmlr.org/papers/v12/duchi11a.html (Cited on p. 19).

[34] John C Duchi, Alekh Agarwal, and Martin J Wainwright. "Dual averaging for distributed optimization: Convergence analysis and network scaling". In: *IEEE Transactions on Automatic control* 57.3 (2011), pp. 592–606 (Cited on p. 2).

[35] Rick Durrett. *Probability: theory and examples*. Vol. 49. Cambridge university press, 2019 (Cited on p. 31).

[36] Cong Fang et al. "Spider: Near-optimal non-convex optimization via stochastic path-integrated differential estimator". In: *Advances in Neural Information Processing Systems* 31 (2018), pp. 689–699 (Cited on p. 22).

[37] Michael C Ferris and Olvi L Mangasarian. "Parallel variable distribution". In: *SIAM Journal on Optimization* 4.4 (1994), pp. 815–832 (Cited on p. 2).

[38] Hamid Reza Feyzmahdavian, Arda Aytekin, and Mikael Johansson. "An asynchronous mini-batch algorithm for regularized stochastic optimization". In: *IEEE Transactions on Automatic Control* 61.12 (2016), pp. 3740–3754 (Cited on pp. 28, 29).

[39] Jerome Friedman, Trevor Hastie, and Rob Tibshirani. "Regularization paths for generalized linear models via coordinate descent". In: *Journal of statistical software* 33.1 (2010), pp. 1–1 (Cited on p. 42).

[40] Wenjiang J Fu. "Penalized regressions: the bridge versus the lasso". In: *Journal of computational and graphical statistics* 7.3 (1998), pp. 397–416 (Cited on p. 42).

[41] Guillaume Garrigos, Lorenzo Rosasco, and Silvia Villa. "Convergence of the forward-backward algorithm: beyond the worst-case with the help of geometry". In: *Mathematical Programming* 198 (2023), pp. 937–996 (Cited on p. 64).

[42] Donald Goldfarb and Shiqian Ma. "Fast multiple-splitting algorithms for convex optimization". In: *SIAM Journal on Optimization* 22.2 (2012), pp. 533–556 (Cited on p. 2).

[43] Pinghua Gong and Jieping Ye. "Linear convergence of variance-reduced stochastic gradient without strong convexity". In: *arXiv preprint arXiv:1406.1102* (2014) (Cited on p. 71).

[44] Eduard Gorbunov, Filip Hanzely, and Peter Richtárik. "A unified theory of SGD: Variance reduction, sampling, quantization and coordinate descent". In: *International Conference on Artificial Intelligence and Statistics*. PMLR. 2020, pp. 680–690 (Cited on pp. 3, 22, 63, 65, 67, 72).

[45] Baptiste Goujaud et al. "PEPit: computer-assisted worst-case analyses of first-order optimization methods in Python". In: *arXiv preprint arXiv:2201.04040* (2022) (Cited on p. 83).

[46] Robert M Gower. "Convergence theorems for gradient descent". In: *Lecture notes for Statistical Optimization* (2018) (Cited on pp. 19, 20).

[47] Robert M Gower et al. “Variance-reduced methods for machine learning”. In: *Proceedings of the IEEE* 108.11 (2020), pp. 1968–1983 (Cited on p. 24).

[48] Osman Güler. “On the convergence of the proximal point algorithm for convex minimization”. In: *SIAM journal on control and optimization* 29.2 (1991), pp. 403–419 (Cited on p. 14).

[49] Robert Hannah and Wotao Yin. “On unbounded delays in asynchronous parallel fixed-point algorithms”. In: *Journal of Scientific Computing* 76.1 (2018), pp. 299–326 (Cited on p. 28).

[50] Filip Hanzely and Peter Richtarik. “Accelerated Coordinate Descent with Arbitrary Sampling and Best Rates for Minibatches”. In: *Proceedings of the Twenty-Second International Conference on Artificial Intelligence and Statistics*. Ed. by Kamalika Chaudhuri and Masashi Sugiyama. Vol. 89. Proceedings of Machine Learning Research. PMLR, 16–18 Apr 2019, pp. 304–312. URL: https://proceedings.mlr.press/v89/hanzely19a.html (Cited on p. 83).

[51] Thomas Hofmann et al. “Variance reduced stochastic gradient descent with neighbors”. In: *Advances in Neural Information Processing Systems* 28 (2015), pp. 2305–2313 (Cited on p. 72).

[52] D.R. Hunter and K. Lange. “A tutorial on MM algorithms”. In: *Amer. Stat.* 58.5 (2004), pp. 30–37 (Cited on p. 33).

[53] Rie Johnson and Tong Zhang. “Accelerating stochastic gradient descent using predictive variance reduction”. In: *Advances in neural information processing systems* 26 (2013), pp. 315–323 (Cited on pp. 22, 23, 63, 69, 71, 74, 75, 76).

[54] Hamed Karimi, Julie Nutini, and Mark Schmidt. “Linear convergence of gradient and proximal-gradient methods under the Polyak-Łojasiewicz condition”. In: *Machine Learning and Knowledge Discovery in Databases: European Conference, ECML PKDD 2016, Riva del Garda, Italy, September 19-23, 2016, Proceedings, Part I 16*. Springer. 2016, pp. 795–811 (Cited on p. 64).

[55] Ahmed Khaled and Chi Jin. “Faster federated optimization under second-order similarity”. In: *arXiv preprint arXiv:2209.02257* (2022) (Cited on pp. 4, 62).

[56] Ahmed Khaled and Peter Richtárik. “Better theory for SGD in the nonconvex world”. In: *arXiv preprint arXiv:2002.03329* (2020) (Cited on pp. 19, 65).

[57] Junhyung Lyle Kim, Panos Toulis, and Anastasios Kyrillidis. “Convergence and Stability of the Stochastic Proximal Point Algorithm with Momentum”. In: *Learning for Dynamics and Control Conference*. PMLR. 2022, pp. 1034–1047 (Cited on pp. 25, 63, 76).

[58] Seung-Jean Kim et al. “An interior-point method for large-scale ℓ_1-regularized least squares”. In: *IEEE journal of selected topics in signal processing* 1.4 (2007), pp. 606–617 (Cited on p. 42).

[59] Diederik P Kingma and Jimmy Ba. “Adam: A method for stochastic optimization”. In: *arXiv preprint arXiv:1412.6980* (2014) (Cited on p. 19).

[60] Dmitry Kovalev, Samuel Horváth, and Peter Richtárik. “Don’t jump through hoops and remove those loops: SVRG and Katyusha are better without the outer loop”. In: *Algorithmic Learning Theory*. PMLR. 2020, pp. 451–467 (Cited on pp. 4, 22, 23, 62, 63, 72).

[61] Rainer Kress. “Ill-conditioned linear systems”. In: *Numerical Analysis*. Springer, 1998, pp. 77–92 (Cited on p. 43).

[62] Ching-Pei Lee and Stephen Wright. “First-Order Algorithms Converge Faster than $O(1/k)$ on Convex Problems”. In: *Proceedings of the 36th International Conference on Machine Learning*. Ed. by Kamalika Chaudhuri and Ruslan Salakhutdinov. Vol. 97. Proceedings of Machine Learning Research. PMLR, June 2019, pp. 3754–3762 (Cited on p. 33).

[63] Xiangru Lian et al. “A comprehensive linear speedup analysis for asynchronous stochastic parallel optimization from zeroth-order to first-order”. In: *Advances in Neural Information Processing Systems* 29 (2016) (Cited on p. 28).

[64] Xiangru Lian et al. “Asynchronous parallel stochastic gradient for nonconvex optimization”. In: *Advances in Neural Information Processing Systems* 28 (2015) (Cited on pp. 29, 45).

[65] Ji Liu and Stephen J Wright. “Asynchronous stochastic coordinate descent: Parallelism and convergence properties”. In: *SIAM Journal on Optimization* 25.1 (2015), pp. 351–376 (Cited on pp. 28, 29, 31, 32, 33, 34, 40, 44, 45, 46, 47, 48, 50).

[66] Ji Liu, Stephen J Wright, and Srikrishna Sridhar. “An asynchronous parallel randomized kaczmarz algorithm”. In: *arXiv preprint arXiv:1401.4780* (2014) (Cited on p. 28).

[67] Ji Liu et al. “An asynchronous parallel stochastic coordinate descent algorithm”. In: *International Conference on Machine Learning*. PMLR. 2014, pp. 469–477 (Cited on pp. 17, 28).

[68] Stanislaw Łojasiewicz. “Une propriété topologique des sous-ensembles analytiques réels”. In: *Les équations aux dérivées partielles* 117 (1963), pp. 87–89 (Cited on p. 64).

[69] Zhaosong Lu and Lin Xiao. “On the complexity analysis of randomized block-coordinate descent methods”. In: *Mathematical Programming* 152 (2015), pp. 615–642 (Cited on p. 16).

[70] Zhi-Quan Luo and Paul Tseng. “Error bounds and convergence analysis of feasible descent methods: a general approach”. In: *Annals of Operations Research* 46.1 (1993), pp. 157–178 (Cited on p. 36).

[71] Vien Mai and Mikael Johansson. “Convergence of a stochastic gradient method with momentum for non-smooth non-convex optimization”. In: *International Conference on Machine Learning*. PMLR. 2020, pp. 6630–6639 (Cited on p. 29).

[72] Bernard Martinet. “Brève communication. Régularisation d’inéquations variationnelles par approximations successives”. In: *Revue française d’informatique et de recherche opérationnelle. Série rouge* 4.R3 (1970), pp. 154–158 (Cited on p. 1).

[73] Andre Milzarek, Fabian Schaipp, and Michael Ulbrich. “A semismooth Newton stochastic proximal point algorithm with variance reduction”. In: *arXiv preprint arXiv:2204.00406* (2022) (Cited on pp. 4, 62).

[74] Eric Moulines and Francis Bach. “Non-asymptotic analysis of stochastic approximation algorithms for machine learning”. In: *Advances in neural information processing systems* 24 (2011), pp. 451–459 (Cited on pp. 19, 21).

[75] Angelia Nedić, Dimitri P Bertsekas, and Vivek S Borkar. “Distributed asynchronous incremental subgradient methods”. In: *Studies in Computational Mathematics* 8.C (2001), pp. 381–407 (Cited on p. 29).

[76] Yu Nesterov. “Gradient methods for minimizing composite functions”. In: *Mathematical Programming* 140.1 (2013), pp. 125–161 (Cited on p. 42).

[77] Yurii Nesterov. *Introductory lectures on convex optimization: A basic course*. Vol. 87. Springer Science & Business Media, 2003 (Cited on pp. 1, 30, 67, 80).

[78] Yurii Nesterov and Vladimir Spokoiny. “Random gradient-free minimization of convex functions”. In: *Foundations of Computational Mathematics* 17 (2017), pp. 527–566 (Cited on p. 2).

[79] Lam M Nguyen et al. “SARAH: A novel method for machine learning problems using stochastic recursive gradient”. In: *International Conference on Machine Learning*. PMLR. 2017, pp. 2613–2621 (Cited on p. 22).

[80] Thomas Paine et al. “Gpu asynchronous stochastic gradient descent to speed up neural network training”. In: *arXiv preprint arXiv:1312.6186* (2013) (Cited on p. 28).

[81] Andrei Patrascu and Ion Necoara. “Nonasymptotic convergence of stochastic proximal point methods for constrained convex optimization”. In: *The Journal of Machine Learning Research* 18.1 (2017), pp. 7204–7245 (Cited on pp. 25, 67).

[82] Zhimin Peng et al. “Arock: an algorithmic framework for asynchronous parallel coordinate updates”. In: *SIAM Journal on Scientific Computing* 38.5 (2016), A2851–A2879 (Cited on p. 28).

[83] Boris T Polyak. “Gradient methods for the minimisation of functionals”. In: *USSR Computational Mathematics and Mathematical Physics* 3.4 (1963), pp. 864–878 (Cited on p. 64).

[84] Boris T Polyak. “Some methods of speeding up the convergence of iteration methods”. In: *Ussr computational mathematics and mathematical physics* 4.5 (1964), pp. 1–17 (Cited on p. 64).

[85] Marco Rando et al. “An optimal structured zeroth-order algorithm for non-smooth optimization”. In: *Advances in Neural Information Processing Systems* 36 (2024) (Cited on p. 85).

[86] Benjamin Recht and Stephen J. Wright. *Optimization for Modern Data Analysis*. Preprint available at http://eecs.berkeley.edu/~brecht/opt4mlbook. 2019 (Cited on p. 14).

[87] Benjamin Recht et al. “Hogwild!: A lock-free approach to parallelizing stochastic gradient descent”. In: *Advances in neural information processing systems* 24 (2011) (Cited on pp. 17, 28, 44).

[88] Herbert Robbins and Sutton Monro. “A stochastic approximation method”. In: *The annals of mathematical statistics* 22.3 (1951), pp. 400–407 (Cited on pp. 2, 19, 76).

[89] Ernest K Ryu and Stephen Boyd. “Stochastic proximal iteration: a non-asymptotic improvement upon stochastic gradient descent”. In: *Author website, early draft* (2014) (Cited on p. 25).

[90] Saverio Salzo and Silvia Villa. “Parallel random block-coordinate forward–backward algorithm: a unified convergence analysis”. In: *Mathematical Programming* 193.1 (2022), pp. 225–269 (Cited on pp. 16, 28, 29, 32, 33, 34, 35, 37, 40, 46, 50, 84).

[91] Saverio Salzo and Silvia Villa. “Proximal Gradient Methods for Machine Learning and Imaging”. In: *Harmonic and Applied Analysis: from Radon transforms to machine learning*. Ed. by Filippo De Mari and Ernesto De Vito. Cham: Springer International Publishing, 2022 (Cited on pp. 5, 14, 15, 28, 33, 42).

[92] Mark Schmidt, Nicolas Le Roux, and Francis Bach. “Minimizing finite sums with the stochastic average gradient”. In: *Mathematical Programming* 162.1 (2017), pp. 83–112 (Cited on p. 22).

[93] Othmane Sebbouh, Robert M Gower, and Aaron Defazio. “Almost sure convergence rates for stochastic gradient descent and stochastic heavy ball”. In: *Conference on Learning Theory*. PMLR. 2021, pp. 3935–3971 (Cited on p. 68).

[94] Shai Shalev-Shwartz and Shai Ben-David. *Understanding Machine Learning: From Theory to Algorithms*. USA: Cambridge University Press, 2014. ISBN: 1107057132 (Cited on p. 1).

[95] Shai Shalev-Shwartz and Shai Ben-David. *Understanding Machine Learning: From Theory to Algorithms*. Cambridge University Press, 2014. DOI: 10.1017/CBO9781107298019 (Cited on p. 43).

[96] Shai Shalev-Shwartz and Shai Ben-David. *Understanding machine learning: From theory to algorithms*. Cambridge university press, 2014 (Cited on p. 18).

[97] Tao Sun, Robert Hannah, and Wotao Yin. "Asynchronous coordinate descent under more realistic assumptions". In: *Advances in Neural Information Processing Systems* 30 (2017) (Cited on p. 28).

[98] Tingni Sun and Cun-Hui Zhang. "Sparse matrix inversion with scaled lasso". In: *The Journal of Machine Learning Research* 14.1 (2013), pp. 3385–3418 (Cited on p. 42).

[99] Adrien B Taylor, Julien M Hendrickx, and François Glineur. "Exact worst-case convergence rates of the proximal gradient method for composite convex minimization". In: *Journal of Optimization Theory and Applications* 178 (2018), pp. 455–476 (Cited on p. 13).

[100] Adrien B Taylor, Julien M Hendrickx, and François Glineur. "Exact worst-case performance of first-order methods for composite convex optimization". In: *SIAM Journal on Optimization* 27.3 (2017), pp. 1283–1313 (Cited on p. 13).

[101] Adrien B Taylor, Julien M Hendrickx, and François Glineur. "Smooth strongly convex interpolation and exact worst-case performance of first-order methods". In: *Mathematical Programming* 161 (2017), pp. 307–345 (Cited on p. 13).

[102] Marc Teboulle and Yakov Vaisbourd. "An elementary approach to tight worst case complexity analysis of gradient based methods". In: *Mathematical Programming* 201.1 (2023), pp. 63–96 (Cited on p. 13).

[103] Robert Tibshirani. "Regression shrinkage and selection via the lasso". In: *Journal of the Royal Statistical Society: Series B (Methodological)* 58.1 (1996), pp. 267–288 (Cited on p. 42).

[104] Panos Toulis and Edoardo M Airoldi. "Asymptotic and finite-sample properties of estimators based on stochastic gradients". In: *The Annals of Statistics* 45.4 (2017), pp. 1694–1727 (Cited on p. 25).

[105] Panos Toulis, Thibaut Horel, and Edoardo M Airoldi. "The proximal robbins–monro method". In: *Journal of the Royal Statistical Society Series B: Statistical Methodology* 83.1 (2021), pp. 188–212 (Cited on p. 25).

[106] Panos Toulis, Dustin Tran, and Edo Airoldi. "Towards stability and optimality in stochastic gradient descent". In: *Artificial Intelligence and Statistics*. PMLR. 2016, pp. 1290–1298 (Cited on p. 25).

[107] Joel A Tropp. "Just relax: Convex programming methods for identifying sparse signals in noise". In: *IEEE transactions on information theory* 52.3 (2006), pp. 1030–1051 (Cited on p. 42).

[108] Paul Tseng. “Approximation accuracy, gradient methods, and error bound for structured convex optimization”. In: *Mathematical Programming* 125.2 (2010), pp. 263–295 (Cited on p. 42).

[109] Paul Tseng. “Convergence of a block coordinate descent method for nondifferentiable minimization”. In: *Journal of optimization theory and applications* 109.3 (2001), pp. 475–494 (Cited on p. 42).

[110] Kiwon Um et al. “Solver-in-the-loop: Learning from differentiable physics to interact with iterative pde-solvers”. In: *Advances in Neural Information Processing Systems* 33 (2020), pp. 6111–6122 (Cited on p. 28).

[111] Zhe Wang et al. “Spiderboost and momentum: Faster variance reduction algorithms”. In: *Advances in Neural Information Processing Systems* 32 (2019), pp. 2406–2416 (Cited on p. 22).

[112] Jin Zhang and Xide Zhu. “Linear Convergence of Prox-SVRG Method for Separable Non-smooth Convex Optimization Problems under Bounded Metric Subregularity”. In: *Journal of Optimization Theory and Applications* 192 (2022), pp. 564–597 (Cited on p. 71).