NEW STOCHASTIC AND RANDOMIZED ALGORITHMS FOR NONCONVEX OPTIMIZATION IN MACHINE LEARNING

Nhan H. Pham

A dissertation submitted to the faculty of the University of North Carolina at Chapel Hill in partial fulfillment of the requirements for the degree of Doctor of Philosophy in the Department of Statistics and Operations Research.

Chapel Hill
2021

Approved by

Yufeng Liu

Ion Necoara

Dzung T. Phan

Quoc Tran-Dinh

Serhan Ziya

ABSTRACT

Nhan H. Pham: New Stochastic and Randomized Algorithms for Nonconvex Optimization in Machine Learning
(Under the direction of Quoc Tran-Dinh)

The goal of this dissertation is to develop efficient stochastic and randomized first-order methods to solve composite nonconvex problems arising from modern machine learning applications. The content of this dissertation is divided into four main chapters. Firstly, we motivate our research topics by briefly introducing our interested problems and their challenges. We also review necessary mathematical concepts and tools used throughout this dissertation.

Our first contribution is in Chapter 2, where we propose **ProxSARAH**, a new framework that uses a variance reduced stochastic gradient estimator called SARAH, to develop new algorithms for solving the stochastic composite nonconvex problems. Our analysis shows that our methods can achieve the best-known convergence results and even match the lower bound complexity. We also provide extensive numerical experiments to illustrate the advantages of our methods compared to existing ones.

Next, we study a policy gradient strategy in reinforcement learning in Chapter 3. We propose a new proximal hybrid stochastic policy gradient algorithm, called **ProxHSPGA**, using a new policy gradient estimator built from two different estimators. **ProxHSPGA** makes uses of a newly hybrid stochastic estimator introduced in Tran-Dinh et al. (2019b), and apply it to reinforcement learning. This new algorithm is able to solve the general composite policy optimization problem which includes regularization or constraint on the policy parameters. It also achieves the best-known sample complexity compared to existing methods. Our experiments on both discrete and continuous control tasks show that our proposed methods indeed are advantageous over existing ones.

Then, in Chapter 4, we focus on a new machine learning paradigm, called federated learning (FL), where multiple agents collaboratively train a machine learning model in a distributed

fashion. We propose two new algorithms, **FedDR** and **asyncFedDR**, for solving the nonconvex composite optimization problem which can handle convex regularizers in FL. Our algorithms rely on a novel combination between a nonconvex Douglas-Rachford splitting method, randomized block-coordinate strategies, and asynchronous implementation. Unlike previous primal-dual based method for FL, our algorithms allow not only partial participation at each communication round but also asynchronous updates between agents which greatly improves their practicality. Our convergence analyses show that the new algorithms match the communication complexity lower bound up to a constant factor under standard assumptions. Our numerical experiments illustrate the advantages of our methods compared to existing ones on various datasets.

Finally, we summarize our contribution, further discuss some notable points of our results, and outline some ongoing and possible future directions. One of our ongoing works is to develop a class of accelerated Douglas-Rachford splitting algorithms for federated learning.

ACKNOWLEDGEMENTS

First of all, I would like to thank my advisor, Dr. Quoc Tran-Dinh, who has guided me throughout all the topics of my research, and provided invaluable support during my Ph.D period at UNC. His vision, discipline, and motivation have deeply inspired me. It was a great privilege and honor to work and study under his guidance.

I would like to express my gratitude to the faculty and staff at the Department of Statistics and Operations Research, The University of North Carolina at Chapel Hill, for their support. Within a great and friendly environment, the last five years have allowed me to grow as a researcher, an educator, as well as an individual.

I am also grateful to my committee members, Dr. Yufeng Liu, Dr. Ion Necoara, Dr. Dzung T. Phan, and Dr. Serhan Ziya for their help, comments, and suggestions to further improve my dissertation. My gratitude also goes to Dr. Lam Nguyen and Dr. Marten van Dijk for their assistant and support in our joint projects. My research has also been partly supported by the National Science Foundation (Grant No. 1619884) and the Office of Naval Research (Grant No. N00014-20-1-2088).

I would like to express my deepest gratitude to my parents. Without you, I would not be able to pursue my dream. A special thank to my beloved wife, Quynh, for her unconditional love, endless faith, and patient support with my work, my dreams, my goals, and the life that we share together. You remain my strength and inspiration through it all.

Last but not least, I would like to thank all my friends and colleagues, who are numerous to list without risking an acknowledgment longer than the thesis paper itself. Thank you everyone.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

CHAPTER 1
Introduction

1.1 Overview

In the last two decades, large-scale optimization has played an important role in computational sciences, including compressive sensing, signal and image processing, modern statistics, machine learning, and data science. Some applications may have the input sizes ranging up to millions which make these large-scale problems more challenging to solve, even approximately, as the cost of evaluating full gradient and function values for first-order methods becomes significantly expensive or even impossible in the contemporary computational devices.

On the one hand, second order methods often have faster convergence rates, and therefore, better iteration-complexity, it may be more costly to compute the second order oracle (i.e., Hessian or its approximations) when the sample size is large so we do not consider it here. On the other hand, although first-order methods can only guarantee convergence to a stationary point (Nesterov, 2013), it has been empirically shown that for many large-size optimization problems such as neural network training and matrix factorization, local optima is almost as good as a global one (Choromanska et al., 2015). Therefore, first-order methods have been widely used in practice in the past few years, especially for solving large-scale problems.

This dissertation will focus on developing different stochastic optimization algorithms to handle complex and large-scale optimization models which cover many applications in different fields. In particular, we focus on designing new stochastic first-order methods which can achieve better first-order oracle complexity than existing methods and achieve state-of-the-art performance to solve common classes of nonconvex optimization problems in machine learning. We also develop randomized algorithms for solving optimization problems in federated learning.

Let us first discuss the motivation for research presented in this dissertation where we briefly describe the problems of interest. Next, we identify fundamental challenges related to those

problems, and then clearly state the goal of our research. Finally, we review several important mathematical concepts and tools which will be used in the following chapters of this dissertation.

1.1.1 Problems of interest

There has been intensive research focusing on designing gradient methods to solve convex problems. Instead, we want to target the nonconvex setting which is often more challenging to solve. This setting also covers many problems related to classification using nonconvex losses or neural network training where the objective function is highly nonconvex. As the theoretical results of variance reduction method to solve nonconvex problems in some areas are still limited, our first goal of this dissertation is to develop new stochastic first-order methods that achieve state-of-the-art complexity over existing methods with practical performance.

In this dissertation, we also consider applications of stochastic estimators in reinforcement learning which is also a highly active research area. In reinforcement learning, there is an agent in a normally unknown environment and the agent can obtain some rewards by interacting with the environment. The agent should take actions to maximize the cumulative rewards. The overall goal of reinforcement learning is to learn good strategy for the agent via interacting with the environment and rewards returned from the environment. An optimal strategy can help the agent adapt to the environment and maximize future rewards.

There are many methods in the literature to solve reinforcement learning problems. Classical algorithms include policy iteration, temporal-difference learning, and Q-learning (Sutton and Barto, 2018). Another well-known approach is the policy gradient method where we can model the problem as a maximization over a reward function then apply the gradient ascent step to update the policy parameter given that we can compute the gradient of the objective function. We will first propose a new stochastic policy gradient estimator based on Tran-Dinh et al. (2019b) in reinforcement learning which is a combination of two other estimators and propose a new policy gradient algorithm that is advantageous over existing ones.

We also consider problems in an emerging computing paradigm called federated learning where multiple local devices collaboratively learn a machine learning model. This setting was first introduced in 2016 and has been extensively studied. A notable difference of federated learning from distributed optimization is that the local agents only exchange the local model instead

of their gradients to promote data privacy. In federated learning, communication becomes the bottleneck so our goal is to design algorithms that can achieve state-of-the-art communication complexity to solve federated learning problems.

1.1.2 Challenges

The large-scale optimization models in modern data analysis and machine learning applications create several theoretical and computational challenges. Among these challenges, the following three are common in most applications. The first challenge is large-scale instances. Problems are getting bigger and bigger in terms of both the number of variables and the size of input data. The second one is nonconvexity, which is present in various applications, where nonlinear models are used. The third challenge is nonsmoothness, which is often due to the use of regularizers, penalty terms, and the presence of constraints. These three fundamental challenges make traditional optimization techniques inefficient or even infeasible to solve.

In addition, the data and computational devices are distributed across multiple nodes in a common network which poses another challenge. As machine learning and data science applications are often equipped with uncertainty and can accept solutions with moderate or low accuracy, gradient-based methods become more desirable especially stochastic gradient-based methods. The classical stochastic gradient descent (SGD) method suffers from slow convergence rate due to the effect of non-diminishing variance of the stochastic gradient estimator (Ghadimi and Lan, 2012, 2013; Ghadimi et al., 2016), another type of gradient-based method has been proposed which is the so-called variance reduction method, e.g. Johnson and Zhang (2013); Defazio et al. (2014); Nguyen et al. (2017a), where the variance of the stochastic gradient estimator can decrease over the iterations.

Concurrently, improving oracle complexity, i.e. number of stochastic gradient evaluations, becomes an attractive research direction on computation complexity of optimization algorithms when the training sample sizes are huge. Therefore, we also take into account this trend when developing new stochastic optimization methods. In general, most of practical problems nowadays are nonconvex and possibly nonsmooth which is hard to solve. Note that first-order or gradient-based method can only guarantee to find a stationary point or a local optimum which is also our aim in this dissertation (Nesterov, 2013).

1.1.3 The goals of this research

The first goal of this research is to develop a new variance reduction framework to solve a general composite stochastic optimization problem that cover many applications in statistics, machine learning, and data science. Our algorithms rely on the SARAH stochastic gradient estimator (Nguyen et al., 2017a) which possesses the variance reduced property. The new algorithms achieve the best-known oracle complexity along with cutting edge performance over existing methods.

Another goal of this research is to design variance reduction methods to improve the sample efficiency in reinforcement learning. We first consider existing policy gradient estimators and proposed a new "hybrid" estimator which is the key element in our new algorithms. Our algorithms should also have an edge over existing methods in terms of both sample complexity and practical performance.

The third goal of this dissertation is to introduce new methods to solve problems in federated learning, a new machine learning paradigm, which has recently gained a lot of attention. By looking at a reformulation of the original problem and existence of convex regularizers, we propose two new algorithms, FedDR and asyncFedDR, inspired by the classical Douglas-Rachford splitting scheme. The proposed methods are shown to achieve state-of-the-art results in terms of communication complexity while being able to handle data and system heterogeneity.

Finally, we summarize our research contributions presented in this dissertation and provide details on possible research direction based on our current results.

1.2 Background and mathematical tools

This section only recalls necessary concepts and mathematical tools which will be used in the sequel. Further definitions, properties, and examples can be found in, e.g., Bauschke and Combettes (2011); Nesterov (2013).

1.2.1 Basic concepts

We are working with Euclidean spaces, $\mathbb{R}^p$ or $\mathbb{R}^n$, equipped with the standard inner product $\langle \cdot, \cdot \rangle$, and the Euclidean norm $\|\cdot\|$. Given a function $f : \mathbb{R}^p \to \mathbb{R} \cup \{+\infty\}$, we use $\mathrm{dom}(f) :=$

$\left\{w \in \mathbb{R}^d \mid f(w) < +\infty\right\}$ to denote its (effective) domain. We say that f is proper if it does not take $-\infty$ as its value and $\mathrm{dom}(f)$ is nonempty. A set is closed if it contains all its limit points. A function $f : \mathbb{R}^p \to \mathbb{R}$ is said to be closed if its epi-graph $\{(x,t) \in \mathrm{dom}(f) \times \mathbb{R} : f(x) \leq t\}$ is a closed set. We denote $[n] := \{i \in \mathbb{N} : 1 \leq i \leq n\}$, the set of integers from 1 to n for a given integer n.

1.2.2 Tools from convex analysis

We also work with the class of proper, closed, and convex functions. A function f is called convex if for any $x, y \in \mathrm{dom}(f)$ and $\alpha \in [0,1]$, we have $f((1-\alpha)x + \alpha y) \leq (1-\alpha)f(x) + \alpha f(y)$ where $\mathrm{dom}(f)$ is a convex set. If f is proper, closed, and convex, its subdifferential at $x \in \mathrm{dom}(f)$ is defined as

$$\partial f(x) := \left\{v \in \mathbb{R}^d \mid f(z) \geq f(x) + \langle v, z - x\rangle, \quad \forall z \in \mathrm{dom}(f)\right\}.$$

If $x \notin \mathrm{dom}(f)$, $\partial f(x) \equiv \emptyset$. Any element $\nabla f(x)$ of $\partial f(x)$ is called a subgradient of f at x. If f is differentiable at x, then $\partial f(x) = \{\nabla f(x)\}$, the gradient of f at x. In this case, the convexity of f is equivalent to $f(x) + \nabla f(x)^\top (y - x) \leq f(y)$ for all $x, y \in \mathrm{dom}(f)$.

Smoothness. A continuously differentiable function $f : \mathbb{R}^d \to \mathbb{R}$ is said to be L_f-smooth with a Lipschitz constant $L_f \in [0, +\infty)$ if ∇f is Lipschitz continuous on its domain, i.e.:

$$\|\nabla f(x) - \nabla f(y)\| \leq L_f \|x - y\|, \quad \forall x, y \in \mathrm{dom}(f). \tag{1.1}$$

As proved in (Nesterov, 2013), (1.1) is equivalent to

$$-\frac{L_f}{2}\|y - x\|^2 + f(x) + \langle \nabla f(x), y - x\rangle \leq f(y) \leq f(x) + \langle \nabla f(x), y - x\rangle + \frac{L_f}{2}\|y - x\|^2, \tag{1.2}$$

for all $x, y \in \mathrm{dom}(f)$.

Proximal operator. The proximal operator for a proper, closed, and convex function is also defined as

$$\mathrm{prox}_f(x) := \arg\min_z \left\{f(z) + \frac{1}{2}\|z - x\|^2\right\}. \tag{1.3}$$

Note that if f is the indicator of a nonempty, closed, and convex set $\mathcal{X}$, i.e., $f(x) = \delta_{\mathcal{X}}(x)$, then $\text{prox}_f(\cdot) = \text{proj}_{\mathcal{X}}(\cdot)$, the projection of x onto $\mathcal{X}$. When the function f is separable such as $f(x) = \frac{1}{n}\sum_{i=1}^n f_i(x)$, evaluating $\text{prox}_{\eta f}$ is equivalent to solving

$$\text{prox}_{\eta f_i}(x) := \underset{y}{\text{argmin}} \left\{ f_i(y) + \tfrac{1}{2\eta}\|y - x\|^2 \right\}, \tag{1.4}$$

for $i = 1, \cdots, n$ seperately. Even when f_i is nonconvex, if f_i is L-smooth as in (1.1), if we choose $0 < \eta < \frac{1}{L}$, then $\text{prox}_{\eta f_i}$ is well-defined and single-valued. Evaluating $\text{prox}_{\eta f_i}$ requires to solve a strongly convex program. Evaluating $\text{prox}_{\eta f_i}$ can be done by various existing methods, including local SGD and accelerated GD-type algorithms. More details about proximal operators can be found in Parikh and Boyd (2014).

Gradient mapping. given a composite function $F(x) := f(x) + g(x)$, suppose $f(x)$ is smooth and $g(x)$ is convex and possibly nonsmooth. For any fixed $\eta > 0$, the quantity

$$G_\eta(x) := \frac{1}{\eta}\Big(x - \text{prox}_{\eta g}(x - \eta \nabla f(x))\Big) \tag{1.5}$$

is called the *gradient mapping* of F (Nesterov, 2013). When $g(\cdot) \equiv 0$, $G_\eta(x) = \nabla f(x)$ which is the gradient of f at x.

We have the following property: $\|G_\eta(x)\| = 0$ iff $x^\star$ is a stationary point of $F(x)$, i.e. $0 \in \nabla f(x^\star) + \partial g(x^\star)$. Let us give more details why $x^\star$ is a stationary point of F when $\|G_\eta(x^\star)\| = 0$. If we define $x^+ := \text{prox}_{\eta g}(x^\star - \eta \nabla f(x^\star))$, then we can write x^+ as

$$x^+ = \underset{z}{\text{argmin}} \left\{ g(z) + \frac{1}{2\eta}\left\|z - (x^\star - \eta \nabla f(x^\star))\right\|^2 \right\}.$$

Therefore, $x^\star$ satisfies the optimality condition of this problem. Equivalently, we have

$$0 \in \partial g(x^+) + \frac{1}{\eta}(x^+ - (x^\star - \eta \nabla f(x^\star))),$$

or can write it as

$$\frac{1}{\eta}(x^\star - x^+) \in \nabla f(x^\star) + \partial g(x^+). \tag{1.6}$$

The condition $\|G_\eta(x^\star)\| = 0$ leads to $\left\|\frac{1}{\eta}(x^\star - \mathrm{prox}_{\eta g}(x^\star - \eta\nabla f(x^\star)))\right\| = 0$, which further results in $\left\|\frac{1}{\eta}(x^\star - x^+)\right\| = 0$ or $x^\star = x^+$. As a result, we have $0 \in \nabla f(x^\star) + \partial g(x^\star)$ by replacing x^+ by $x^\star$ in (1.6).

1.2.3 Other mathematical tools

In this subsection, we recall some other concepts and notations used in this dissertation.

Basic concepts and notations in reinforcement learning. We first provide a brief overview on basic concepts in reinforcement learning which will be used in Chapter 3. In general, most problems in reinforcement learning can be formulated as a Markov Decision Process (MDP). In a MDP, all states have Markov property, i.e., the transition to next state only depends on the current state and the action taken at that state.

We consider a MDP (Sutton and Barto, 2018) equipped with 6 components $\{\mathcal{S}, \mathcal{A}, \mathcal{P}, \mathcal{R}, \gamma, \mathcal{P}_0\}$ where $\mathcal{S}$, $\mathcal{A}$ are the state and action spaces, $\mathcal{P}$ denotes the set of transition probabilities when taking certain actions, $\mathcal{R}$ is the reward function which characterizes the immediate reward earned by taking certain action, γ is a discount factor, and $\mathcal{P}_0$ is the initial state distribution.

Let $\pi(\cdot|s)$ be a density function over $\mathcal{A}$ when current state is s and $\pi_\theta(\cdot|s)$ is a policy parameterized by parameter θ. We define a trajectory $\tau = \{s_0, a_0, s_1, a_1, \cdots, s_{H-1}, a_{H-1}\}$ with effective length H as a collection of states and actions sampled from a stationary policy. Denote $p_\theta(\cdot)$ as the density induced by policy π_θ over all possible trajectories and $p_\theta(\tau)$ is the probability of observing a trajectory τ. Let $\mathcal{R}(\tau) = \sum_{t=0}^{H-1} \gamma^t \mathcal{R}(s_t, a_t)$ be the total discounted reward for a trajectory τ.

Complexity notions. Thought out the dissertation, we often come across the Big-O notation $\mathcal{O}$. Formally, given two function $f(x)$ and $g(x)$ defined on some unbounded subset of $\mathbb{R}_+$ (positive real numbers) and $g(x)$ is strictly positive for all large enough x, we say

$$f(x) = \mathcal{O}\left(g(x)\right) \quad \text{as} \quad x \to \infty$$

if $|f(x)| \leq Mg(x)$ for all $x \geq x_0$, i.e. the magnitude of $f(x)$ is at most a positive constant multiple of $g(x)$ for sufficiently large x. In other words, we also say that the growth rate of

$f(x)$ is at most the growth rate of $g(x)$. In addition, we may encounter the Big-Theta (Θ) or Big-Omega (Ω) notations where these notation means f(x) grows in the same order or at least as fast as g(x), respectively.

1.3 The outline of dissertation

The main content of this dissertation is divided into four chapters, and is organized as follows. Note that the results of Chapter 2, 3, and 4 are presented in Pham et al. (2020b), Pham et al. (2020a), and Tran-Dinh et al. (2021), respectively.

- **Chapter 2:** This chapter is based on the following paper:

  [1] Nhan H. Pham, Lam M. Nguyen, Dzung T. Phan, and Quoc Tran-Dinh. ProxSARAH: An efficient algorithmic framework for stochastic composite nonconvex optimization. *Journal of Machine Learning Research*, 21(110):1–48, 2020. (*Reference Pham et al. (2020b)*).

  Summary: In this chapter, we consider two common stochastic composite nonconvex formulation that covers many applications in statistics and machine learning. As theoretical results for variance reduction methods to solve these composite models are still limited, we propose a proximal algorithm using SARAH estimator, a variance reduced stochastic gradient estimator, and show that this method can achieve the best-known complexity to solve both models and also matches the lower bound in certain settings. We provide multiple numerical examples to illustrate the advantages of the proposed algorithms compared to state-of-the-art methods.

  Our main contribution: We propose ProxSARAH, a new and general stochastic variance reduction framework relying on the SARAH estimator to solve both expectation and finite-sum stochastic composite optimization models. We propose different variants of ProxSARAH with constant and dynamic step-size. Our convergence analysis shows that all ProxSARAH variants achieve the best-known complexity under standard assumptions to solve both problems. The results in the expectation case also match the lower bound in Arjevani et al. (2019).

- **Chapter 3:** This chapter is based on the following paper:

  [1] Nhan H. Pham, Lam M. Nguyen, Dzung T. Phan, Phuong Ha Nguyen, Marten Van Dijk, and Quoc Tran-Dinh. A hybrid stochastic policy gradient algorithm for reinforcement learning. *International Conference on Artificial Intelligence and Statistics*, PMLR 108:374-385, 2020. (*Reference Pham et al. (2020a)*).

  Summary: In this chapter, we focus on dealing with one of the common problem in reinforcement learning which is the policy optimization problem. We consider the composite policy optimization which is a generalized model of the original problem. We first introduce a new "hybrid" stochastic policy gradient estimator and then propose a new algorithm using the new hybrid estimator. We also evaluate the performance of our algorithms in multiple discrete and continuous control tasks using well-known simulators in reinforcement learning and show that our algorithms indeed are advantageous over existing methods and solving the composite model may be beneficial than the original policy optimization problem.

  Our main contribution: Our first contribution is the novel hybrid stochastic policy gradient estimator by combining existing REINFORCE estimator with the adapted SARAH estimator for policy gradient. We then propose a new algorithm to solve a composite maximization problem for policy optimization in reinforcement learning. Our model not only covers existing settings but also handles constraints and convex regularizers on policy parameters. We show that under standard assumptions, our algorithms not only achieve the best-known sample complexity but also consider a more general setting compared to existing policy gradient methods.

- **Chapter 4** This chapter is based on the following paper:

  [1] Quoc Tran-Dinh, Nhan H. Pham, Dzung T. Phan, and Lam M. Nguyen. FedDR - Randomized Douglas-Rachford splitting algorithms for nonconvex federated composite optimization. *Thirty-fifth Conference on Neural Information Processing Systems*, 2021. (*Reference Tran-Dinh et al. (2021)*).

Summary: In this chapter, we develop two new algorithms, **FedDR** and **asyncFedDR**, for solving a nonconvex composite optimization problem which can handle convex regularizers in federated learning. Our algorithms rely on a novel combination between nonconvex Douglas-Rachford splitting method, randomized block-coordinate strategies, and asynchronization. The new algorithms are able to address multiple key challenges in federated learning including communication efficiency, data and system heterogeneity. Our convergence analysis shows that the proposed algorithms match the communication complexity lower bound up to a constant factor under standard assumptions. Our numerical experiments illustrate the advantages of our methods compared to existing ones on various synthetic and real datasets.

Our main contribution: Our first contribution is to develop a new FL algorithm, called **FedDR** (**Fed**erated **D**ouglas-**R**achford), by combining the classical DR splitting technique and randomized block-coordinate strategy to solve the nonconvex composite optimization problem in FL. Our algorithm can handle nonsmooth convex regularizers and allows inexact evaluation of the underlying proximal operators as in FedProx or FedPD. It also achieves the best known $\mathcal{O}\left(\varepsilon^{-2}\right)$ communication complexity for finding a stationary point under standard assumptions, where ε is a given accuracy. Different from FedSplit (Pathak and Wainwright, 2020) and FedPD (Zhang et al., 2020), our algorithm allows partial participation requiring only a subset of workers to perform local update at each communication round. The second contribution lies within the asynchronous algorithm, **asyncFedDR**, where each worker independently performs local update and send the update to the server for proximal aggregation in an asynchronous fashion. **asyncFedDR** achieves the same communication complexity $\mathcal{O}\left(\varepsilon^{-2}\right)$ as **FedDR** (up to a constant factor) under the same standard assumptions.

- **Chapter 5** concludes our research presented in this dissertation and propose potential research directions in the future where we incorporate acceleration techniques into FedDR to further improve its communication complexity.

CHAPTER 2

An Efficient Framework for Stochastic Composite Nonconvex Optimization

2.1 Introduction

With the recent advances in computing power making the field of large-scale optimization more active than ever. As a result, there is also need to design efficient numerical methods to solve those challenging problems. In this chapter, we want to focus on an optimization problem that is general enough to cover many applications in different fields and propose a new algorithmic framework that can solve the problem efficiently.

2.1.1 Problem of interest

We first consider the following finite-sum composite optimization problem

$$\min_{x\in\mathbb{R}^d} \Big\{ F(x) := f(x) + \psi(x) \equiv \frac{1}{n}\sum_{i=1}^{n} f_i(x) + \psi(x) \Big\}. \tag{FS-OPT}$$

where $f(\cdot) : \mathbb{R}^d \to \mathbb{R}$ is a smooth (possibly nonconvex) function and $\psi : \mathbb{R}^d \to \mathbb{R} \cup \{+\infty\}$ is a (possibly nonsmooth) proper, closed, and convex function.

This problem is general enough to cover many applications in statistical learning, machine learning and deep learning. Here are some examples:

- Regularized least-squares: given an input data $A \in \mathbb{R}^{n\times p}$ and response vector $b \in \mathbb{R}^n$ a penalty parameter λ, we need to solve

$$\min_{x\in\mathbb{R}^p} \ \frac{1}{2}\left\|Ax - b\right\|^2 + \frac{\lambda}{2}\left\|x\right\|^2.$$

  We can cast this problem into (FS-OPT) by defining $f_i(x) = \frac{1}{2}(a_i^\top x - b_i)^2$ and $\psi(x) = \frac{\lambda}{2}\left\|x\right\|^2$ where a_i is the i-th row of A and b_i is the i-th component of b.

- Regularized binary classification: given an input data $A \in \mathbb{R}^{n\times p}$, a response vector y and a bias vector b, we need to solve

$$\min_{x\in\mathbb{R}^p} \frac{1}{n}\sum_{i=1}^{n} \ell(a_i^\top x + b_i, y_i) + \lambda \|x\|_1 ,$$

where $\ell(\cdot)$ represents certain loss function which can be convex (logistic regression) or nonconvex (Zhao et al., 2010) and λ is a penalty parameter. This problem can be cast into (FS-OPT) using $f_i(x) = \ell(a_i^\top x + b_i, y_i)$ and $\psi(x) = \lambda \|x\|_1$.

- Neural network training for image classification: given a set of n observations (images) with size $d \times p$ represented as a tensor $A \in \mathbb{R}^{n\times d\times p}$ and an input label matrix $y \in 0, 1^{n\times m}$ where m is the number of class, i.e. y_i is the vector of binary values indicating the correct class of observation i. The problem we want to solve is

$$\min_{w} \frac{1}{n}\sum_{i=1}^{n} \ell(f(w, a_i), y_i),$$

where $f(w, a_i)$ is the output of a neural network using input data a_i and weight w and ℓ is the cross-entropy loss function. We can define $f_i(w) = \ell(f(w, a_i), y_i) = -\sum_{j=1}^{m} y_{i,j} \log(f(w, a_i))$ and $\psi(w) = 0$.

If the input data is constantly collected (streaming data setting), the size of input data becomes too large in which the problem (FS-OPT) becomes inapplicable. Therefore, each input data point can be viewed as an i.i.d. sample of a random variable with a certain distribution (usually unknown). In particular, we come up with a more general model which is referred to as the stochastic composite optimization as

$$\min_{x\in\mathbb{R}^d} \Big\{ F(x) := f(x) + \psi(x) \equiv \mathbb{E}\left[\mathbf{f}(x;\xi)\right] + \psi(x) \Big\}, \tag{St-OPT}$$

where $f(x) := \mathbb{E}\left[\mathbf{f}(w;\xi)\right]$ is the expectation of a stochastic function $\mathbf{f}(x;\xi)$ depending on a random vector ξ in a given probability space $(\Omega, \mathbb{P})$, and $\psi : \mathbb{R}^d \to \mathbb{R} \cup \{+\infty\}$ is a proper, closed, and convex function. (FS-OPT) can be seen as a special case of (St-OPT) where we use sample average approximation for the expectation.

2.1.2 Related work

Most of available methods to solve (St-OPT) and (FS-OPT) rely on stochastic approaches (Johnson and Zhang, 2013; Schmidt et al., 2017; Shapiro et al., 2014; Defazio et al., 2014; Frostig et al., 2015; Lei and Jordan, 2017; Lin et al., 2015). In the convex case (when $f(\cdot)$ is convex), both non-composite and composite settings of (St-OPT) and (FS-OPT) have been intensively studied with different schemes such as standard stochastic gradient (Robbins and Monro, 1951), proximal stochastic gradient (Nemirovski et al., 2009; Ghadimi and Lan, 2013), stochastic dual coordinate descent (Shalev-Shwartz and Zhang, 2013), stochastic conditional gradient (Frank-Wolfe) methods (Reddi et al., 2016c), stochastic primal-dual methods (Chambolle et al., 2018), and variance reduction methods (Defazio et al., 2014; Allen-Zhu, 2017a; Johnson and Zhang, 2013; Schmidt et al., 2017; Nitanda, 2014; Shalev-Shwartz and Zhang, 2014; Xiao and Zhang, 2014). The two most popular variance reduction methods are SAGA and SVRG. While SAGA (fast incremental gradient algorithm), a successor of SAG (**S**tochastic **A**verage **G**radient) (Schmidt et al., 2017), can only solve the finite-sum problem, SVRG (**S**tochastic **V**ariance **R**educed **G**radient) (Johnson and Zhang, 2013) can solve both finite-sum and expectation problems. Thanks to variance reduction techniques, several efficient methods with constant step-sizes have been developed for convex settings that match the lower-bound worst-case complexity (Agarwal et al., 2012).

In the nonconvex case, both problems (St-OPT) and (FS-OPT) have been intensively studied in recent years with a vast number of research papers. While numerical algorithms for solving the non-composite setting, i.e., $\psi = 0$, are well-developed and have received considerable attention (see Allen-Zhu, 2018; Allen-Zhu and Li, 2018; Allen-Zhu and Yuan, 2016; Fang et al., 2018; Lihua et al., 2017; Nguyen et al., 2017b, 2020, 2019; Reddi et al., 2016b; Zhou et al., 2018a), methods for composite setting remain limited (Reddi et al., 2016b; Wang et al., 2019). In terms of algorithms, Reddi et al. (2016b) study a non-composite finite-sum problem as a special case of (FS-OPT) using SVRG estimator from Johnson and Zhang (2013). Additionally, they extend their method to the composite setting by simply applying the proximal operator of ψ as in the well-known forward-backward scheme. Another related work using SVRG estimator can be found in Li and Li (2018). These algorithms have some limitation as will be discussed later.

The same technique is applied in Wang et al. (2019) to develop other variants for both (St-OPT) and (FS-OPT), but using the SARAH estimator from Nguyen et al. (2017a). The authors derive a large constant step-size, but at the same time control mini-batch size to achieve desired complexity bounds. Consequently, it has an essential limitation as will also be discussed in Section 2.3.4. Both algorithms achieve the best-known complexity bounds for solving (St-OPT) and (FS-OPT). In addition, Reddi et al. (2016c) propose a stochastic Frank-Wolfe method that can handle constraints as special cases of (FS-OPT). Recently, a stochastic variance reduction method with momentum was studied in Zhou et al. (2019) for solving (FS-OPT) which can be viewed as a modification of SpiderBoost in Wang et al. (2019).

In terms of theory, many researchers have been working on theoretical aspects of existing algorithms. For example, Ghadimi and Lan (2013) appear to be one of the first pioneering works studying convergence rates of stochastic gradient descent-type methods for nonconvex and non-composite finite-sum problems. They later extend it to the composite setting in Ghadimi et al. (2016). Wang et al. (2019) also investigate the gradient dominance case, and Karimi et al. (2016) consider both finite-sum and composite finite-sum under different assumptions, including Polyak-Łojasiewicz condition.

Whereas many researchers have been trying to improve complexity upper bounds of stochastic first-order methods using different techniques (Allen-Zhu, 2018; Allen-Zhu and Li, 2018; Allen-Zhu and Yuan, 2016; Fang et al., 2018), other researchers attempt to construct examples for lower-bound complexity estimates. Existing works for lower-bound in the convex case include Agarwal et al. (2012); Nemirovskii and Yudin (1983); Nesterov (2013). In Fang et al. (2018); Zhou and Gu (2019), the authors have constructed a lower-bound complexity for nonconvex finite-sum problem covered by (FS-OPT). They showed that the lower-bound complexity for any stochastic gradient method using only smoothness assumption to achieve an ε-stationary point in expectation is $\Omega\left(n^{1/2}\varepsilon^{-2}\right)$ given that the number of objective components n does not exceed $\mathcal{O}\left(\varepsilon^{-4}\right)$, where ε is a desired accuracy for the approximate solution.

For the expectation problem (St-OPT), the best-known complexity bound to achieve an ε-stationary point in expectation is $\mathcal{O}\left(\sigma\varepsilon^{-3} + \sigma^2\varepsilon^{-2}\right)$ as shown in Fang et al. (2018); Wang et al. (2019), where $\sigma > 0$ is an upper bound of the variance of the stochastic gradient. This

complexity matches the lower bound recently developed in Arjevani et al. (2019) up to a given constant under the same assumptions for the non-composite setting of (St-OPT).

Theory for stochastic methods to solve both composite nonconvex problems (St-OPT) and (FS-OPT) are still in progress and require substantial effort to obtain efficient algorithms with rigorous convergence guarantees. It is shown in Fang et al. (2018); Zhou and Gu (2019) that there is still a gap between the upper-bound complexity in state-of-the-art methods and the lower-bound worst-case complexity for the nonconvex problem (FS-OPT) under standard smoothness assumption. Motivated by this fact, we attempt to develop a new algorithmic framework that can reduce and at least nearly close this gap in the composite finite-sum setting (FS-OPT). In addition to the best-known complexity bounds, we expect to design practical algorithms advancing beyond existing methods by providing a dynamic rule to update step-sizes with rigorous complexity analysis.

2.1.3 Our approach

We take advantage of the SARAH estimator, a biased stochastic recursive gradient estimator, to design new proximal variance reduction stochastic gradient algorithms to solve both (St-OPT) and (FS-OPT). The SARAH algorithm (Nguyen et al., 2017a) is simply a double-loop stochastic gradient method with a flavor of SVRG (Johnson and Zhang, 2013), but using a novel biased estimator different from SVRG. Although SARAH algorithm is a recursive method as SAGA (Defazio et al., 2014), it is more advantageous than SAGA as it avoids the major issue of storing gradients which is memory-inefficient in big-data regime.

Our algorithm remains a variance reduction stochastic method, but it is different from these works at two major points: an additional averaging step and two different step-sizes. Having two step-sizes allows us to flexibly trade-off them and develop a dynamic update rule. Note that our averaging step looks similar to the robust stochastic gradient method in Nemirovski et al. (2009), but is fundamentally different since it evaluates the proximal step at the averaging point. In fact, it is closely related to averaged fixed-point schemes in the literature (Bauschke and Combettes, 2011).

2.1.4 Our contribution

Our main contributions can be summarized as follows:

(a) ***Novel algorithms:*** We propose **ProxSARAH**, a new and general stochastic variance reduction framework relying on the SARAH estimator to solve both expectation and finite-sum problems (St-OPT) and (FS-OPT) in composite settings. As usual, the algorithm has double loops, where the outer loop can either take full gradient or mini-batch to reduce computational burden in large-scale and expectation settings. The inner loop can work with single sample or a broad range of mini-batch sizes. This framework has two different step-sizes as opposed to existing methods. We also derive different variants of ProxSARAH for using constant or dynamic step-sizes and for non-composite settings of (St-OPT) and (FS-OPT) (i.e., $\psi = 0$)

(b) ***Best-known complexity guarantees under constant step-sizes:*** We analyze our framework and its variants to design appropriate constant step-sizes instead of diminishing step-sizes as in standard **S**tochastic **G**radient **D**escent (SGD) methods. In the finite-sum setting (FS-OPT), our methods achieve $\mathcal{O}\left(n + n^{1/2}\varepsilon^{-2}\right)$ complexity bound to attain an ε-stationary point in expectation under only the smoothness of f_i. This complexity matches the lower-bound worst-case complexity in (Fang et al., 2018; Zhou and Gu, 2019) up to a constant factor when $n \leq \mathcal{O}\left(\varepsilon^{-4}\right)$. In the expectation setting (St-OPT), our algorithms require $\mathcal{O}\left(\sigma^2\varepsilon^{-2} + \sigma\varepsilon^{-3}\right)$ stochastic first-order oracle calls of $\mathbf{f}$ to achieve an ε-stationary point in expectation under only the smoothness of $\mathbf{f}$ and bounded variance $\sigma^2 > 0$. To the best of our knowledge, this is the best-known complexity so far for (St-OPT) under standard assumptions in both the single sample and mini-batch cases. This complexity also matches the lower bound recently studied in Arjevani et al. (2019) up to a constant.

(c) ***Best-known complexity guarantees under dynamic step-sizes:*** Apart from constant step-size algorithms, we also analyze variants of Algorithm 1 using dynamic step-sizes for both composite and non-composite settings in both single sample and mini-batch cases. Our dynamic step-sizes are increasing along the inner iterations rather than diminishing as usually used in standard SGDs.

Our result covers the non-composite setting in the finite-sum case (Nguyen et al., 2019), and matches the best-known complexity in Fang et al. (2018); Wang et al. (2019) for both problems (St-OPT) and (FS-OPT). Since the composite setting covers a broader class of nonconvex problems including convex constraints, we believe that our method has better chance to handle new applications than non-composite methods. It also allows one to deal with composite problems under different type of regularizers such as sparsity or constraints on weights as in neural network training applications.

2.1.5 Overview and chapter outline

The remaining of this chapter is organized as follows. Section 2.2 provides essential mathematical tools along with fundamental assumptions on the problem. It also contains useful properties of the stochastic gradient estimators used in our proposed methods. Section 2.3 introduces the ProxSARAH framework and main convergence results of the proposed algorithms. Next, we illustrate the advantage of our algorithm via three numerical examples in Section 2.5. Lastly, Section 2.6 presents the missing proofs for the results presented in Sections 2.2 and 2.3.

2.2 Mathematical tools and preliminary results

In this section, we recall basic notations and concepts in optimization which can be found in Bauschke and Combettes (2011); Nesterov (2013). We then state the fundamental assumptions and show how to characterize the optimality condition of (St-OPT) and (FS-OPT). Eventually, we present several preliminary results needed for the convergence analysis of our algorithms.

2.2.1 Notation

Apart from the basic concepts presented in 1.2 we present additional notations used in this chapter. We use $\mathbf{U}_p(S)$ to denote a finite set $S := \{s_1, s_2, \cdots, s_n\}$ equipped with a probability distribution p over S. If p is uniform, then we simply use $\mathbf{U}(S)$. For any real number a, $\lfloor a \rfloor$ denotes the largest integer less than or equal to a. We use $[n]$ to denote the set $\{1, 2, \cdots, n\}$. In addition, Table 2.1 provides common notations used in this chapter.

Notation	Meaning	Type and range
ε	The target accuracy for stochastic gradient mapping	positive real
m	The epoch length (i.e., the number of iterations of the inner loop t)	positive integer
$\mathcal{B}_s$	The mini-batch of the snapshot point $\widetilde{w}_{s-1}$	finite set of realizations
b_s	The size of the mini-batch $\mathcal{B}_s$ of the snapshot point $\widetilde{w}_{s-1}$	positive integer
$\hat{\mathcal{B}}_t^{(s)}$	The mini-batch for evaluating SARAH estimator in the inner loop t	finite set of realizations
$\hat{b}_t^{(s)}$	The size of the mini-batch $\hat{\mathcal{B}}_t^{(s)}$	positive integer

Table 2.1: Common quantities used in this chapter.

2.2.2 Fundamental assumptions

We aim to develop numerical methods for solving (St-OPT) and (FS-OPT) that rely on basic assumptions usually used in stochastic optimization methods.

Assumption 2.1 (Bounded from below)**.** Both problems (St-OPT) and (FS-OPT) are bounded from below. That is $F^\star := \inf_{w\in\mathbb{R}^d} F(w) > -\infty$. Moreover, $\mathrm{dom}(F) := \mathrm{dom}(f) \cap \mathrm{dom}(\psi) \neq \emptyset$.

This assumption usually holds in practice since f often represents a loss function which is nonnegative or bounded from below. In addition, the regularizer ψ is also nonnegative or bounded from below, and its domain intersects $\mathrm{dom}(f)$.

Our next assumption is the smoothness of f with respect to the argument w.

Assumption 2.2 (L-average smoothness)**.** In the expectation setting (St-OPT), for any realization of $\xi \in \Omega$, $\mathbf{f}(\cdot;\xi)$ is L-smooth (on average), i.e., $\mathbf{f}(\cdot;\xi)$ is continuously differentiable and its gradient $\nabla_w \mathbf{f}(\cdot;\xi)$ is Lipschitz continuous with the same Lipschitz constant $L \in (0,+\infty)$, i.e.:

$$\mathbb{E}_\xi\left[\|\nabla_w \mathbf{f}(w;\xi) - \nabla_w \mathbf{f}(\hat{w};\xi)\|^2\right] \leq L^2\|w-\hat{w}\|^2, \quad w,\hat{w}\in\mathrm{dom}(f). \tag{2.1}$$

In the finite-sum setting (FS-OPT), the condition (2.1) reduces to

$$\frac{1}{n}\sum_{i=1}^n \|\nabla f_i(w) - \nabla f_i(\hat{w})\|^2 \leq L^2\|w-\hat{w}\|^2, \quad w,\hat{w}\in\mathrm{dom}(f). \tag{2.2}$$

We can write (2.2) as $\mathbb{E}_i\left[\|\nabla f_i(w) - \nabla f_i(\hat{w})\|^2\right] \leq L^2\|w-\hat{w}\|^2$. Note that (2.2) is weaker than assuming that each component f_i is L_i-smooth, i.e., $\|\nabla f_i(w) - \nabla f_i(\hat{w})\| \leq L_i\|w-\hat{w}\|$ for all $w,\hat{w}\in\mathrm{dom}(f)$ and $i\in[n]$. Indeed, the individual L_i-smoothness implies (2.2) with

$L^2 := \frac{1}{n}\sum_{i=1}^n L_i^2$. Conversely, if (2.2) holds, then $\|\nabla f_i(w) - \nabla f_i(\hat{w})\|^2 \leq \sum_{i=1}\|\nabla f_i(w) - \nabla f_i(\hat{w})\|^2 \leq nL^2\|w-\hat{w}\|^2$ for $i \in [n]$. Therefore, each component f_i is $\sqrt{n}L$-smooth, which is larger than (2.2) within a factor of $\sqrt{n}$ in the worst-case. We emphasize that ProxSVRG, ProxSVRG+, and ProxSpiderBoost all require the L-smoothness of each component f_i in (FS-OPT). However, the condition (2.1) is stronger than the L-smoothness of the expected function f (i.e., $\|\nabla f(w) - \nabla f(\hat{w})\| \leq L_f\|w-\hat{w}\|$ for $w, \hat{w} \in \mathrm{dom}(f)$) as used in standard SGD algorithms (Ghadimi and Lan, 2013).

It is well-known that the L-smooth condition leads to the following bound

$$\mathbb{E}_\xi\left[\mathbf{f}(\hat{w};\xi)\right] \leq \mathbb{E}_\xi\left[\mathbf{f}(w;\xi)\right] + \mathbb{E}_\xi\left[\langle\nabla_w\mathbf{f}(w;\xi), \hat{w}-w\rangle\right] + \frac{L}{2}\|\hat{w}-w\|^2, \quad w, \hat{w} \in \mathrm{dom}(f). \tag{2.3}$$

Indeed, from (2.1), we have

$$\begin{aligned}\|\nabla f(w) - \nabla f(\hat{w})\|^2 &= \|\mathbb{E}_\xi\left[\nabla_w\mathbf{f}(w;\xi) - \nabla_w\mathbf{f}(\hat{w};\xi)\right]\|^2\\ &\leq \mathbb{E}_\xi\left[\|\nabla_w\mathbf{f}(w;\xi) - \nabla_w\mathbf{f}(\hat{w};\xi)\|^2\right]\\ &\leq L^2\|w-\hat{w}\|^2,\end{aligned}$$

which shows that $\|\nabla f(w) - \nabla f(\hat{w})\| \leq L\|w-\hat{w}\|$. Hence, using either (2.1) or (2.2), we get

$$f(\hat{w}) \leq f(w) + \langle\nabla f(w), \hat{w}-w\rangle + \frac{L}{2}\|\hat{w}-w\|^2, \quad w, \hat{w} \in \mathrm{dom}(f). \tag{2.4}$$

The L-smooth condition also leads to the *L-almost convexity* of f (see Zhou and Gu, 2019) since $f(\cdot) + \frac{L}{2}\|\cdot\|^2$ is convex.

In the expectation setting (St-OPT), we need the following bounded variance condition:

Assumption 2.3 (Bounded variance)**.** For the expectation problem (St-OPT), there exists a uniform constant $\sigma \in (0, +\infty)$ such that

$$\mathbb{E}_\xi\left[\|\nabla_w\mathbf{f}(w;\xi) - \nabla f(w)\|^2\right] \leq \sigma^2, \quad \forall w \in \mathrm{dom}(f). \tag{2.5}$$

For the finite-sum problem (FS-OPT), there exists a uniform constant $\sigma \in (0, +\infty)$ s.t.

$$\frac{1}{n}\sum_{i=1}^{n} \|\nabla f_i(w) - \nabla f(w)\|^2 \leq \sigma^2, \quad \forall w \in \mathrm{dom}(f). \tag{2.6}$$

This assumption is standard in stochastic optimization and often required in almost any solution method for solving (St-OPT) (see Ghadimi and Lan, 2013). For problem (FS-OPT), if n is extremely large, passing over n data points is exhaustive or impossible. We refer to this case as the online case mentioned in Fang et al. (2018), and can be cast into Assumption 2.3. Therefore, we do not consider this case separately. However, our theory and algorithms developed in this chapter do apply to such a setting. In addition, for the finite-sum problem (FS-OPT), if we define $\sigma_n^2(w) := \frac{1}{n}\sum_{i=1}^{n}\left[\|\nabla f_i(w)\|^2 - \|\nabla f(w)\|^2\right]$, then (2.6) becomes $\sigma_n^2(w) \leq \sigma^2$ for all $w \in \mathrm{dom}(f)$, which is consistent to (2.5). We only use the condition (2.6) in Remark 2.

2.2.3 Optimality conditions

Under Assumption 2.1, we have $\mathrm{dom}(f) \cap \mathrm{dom}(\psi) \neq \emptyset$. When $\mathbf{f}(\cdot;\xi)$ is nonconvex in w, the first order optimality condition of (St-OPT) can be stated as

$$0 \in \partial F(w^\star) \equiv \nabla f(w^\star) + \partial\psi(w^\star) \equiv \mathbb{E}_\xi\left[\nabla_w \mathbf{f}(w^\star;\xi)\right] + \partial\psi(w^\star). \tag{2.7}$$

Here, $w^\star$ is called a stationary point of F. We denote $\mathcal{S}^\star$ the set of all stationary points. The condition (2.7) is called the first-order optimality condition, and also holds for (FS-OPT).

Since ψ is proper, closed, and convex, its proximal operator $\mathrm{prox}_{\eta\psi}$ satisfies the nonexpansiveness, i.e., $\|\mathrm{prox}_{\eta\psi}(w) - \mathrm{prox}_{\eta\psi}(z)\| \leq \|w - z\|$ for all $w, z \in \mathbb{R}^d$.

Recall the definition of gradient mapping in (1.5), by using $G_\eta(\cdot)$, the optimality condition (2.7) can be equivalently written as

$$\|G_\eta(w^\star)\|^2 = 0. \tag{2.8}$$

If we apply gradient-type methods to solve (St-OPT) or (FS-OPT), then we can only aim at finding an ε-approximate stationary point $\widetilde{w}_T$ to $w^\star$ in (2.8) after at most T iterations within a

given accuracy $\varepsilon > 0$, i.e.:

$$\mathbb{E}\left[\|G_\eta(\widetilde{w}_T)\|^2\right] \leq \varepsilon^2. \tag{2.9}$$

The condition (2.9) is standard in stochastic nonconvex optimization methods. To obtain approximate second-order optimality, one can use the idea of perturbed gradient as in Lu et al. (2019); Tziotis et al. (2020); Chen et al. (2021) for structured problems or require additional assumptions and more sophisticated optimization methods such as cubic regularized Newton-type schemes (see Nesterov and Polyak, 2006).

2.2.4 Stochastic gradient estimators

One key step to design a stochastic gradient method for (St-OPT) or (FS-OPT) is to query an estimator for the gradient $\nabla f(w)$ at any w. Let us recall some existing stochastic estimators.

2.2.4.1 Single sample estimators

A simple estimator of $\nabla f(w)$ can be computed as follows:

$$\widetilde{\nabla} f(w_t) := \nabla_w \mathbf{f}(w_t; \xi_t), \tag{2.10}$$

where ξ_t is a realization of ξ. This estimator is unbiased, i.e., $\mathbb{E}\left[\widetilde{\nabla} f(w_t) \mid \mathcal{F}_t\right] = \nabla f(w_t)$, but its variance is fixed for any w_t, where $\mathcal{F}_t$ is the history of randomness collected up to the t-th iteration, i.e.:

$$\mathcal{F}_t := \sigma\big(w_0, w_1, \cdots, w_t\big). \tag{2.11}$$

This is a σ-field generated by random variables $\{w_0, w_1, \cdots, w_t\}$. In the finite-sum setting (FS-OPT), we have $\widetilde{\nabla} f(w_t) := \nabla f_{i_t}(w_t)$, where $i_t \sim \mathbf{U}([n])$ with $[n] := \{1, 2, \cdots, n\}$.

In recent years, there has been huge interest in designing stochastic estimators with variance reduction properties. The first variance reduction method was perhaps proposed in Schmidt et al. (2017) since 2013, and then in Defazio et al. (2014) for convex optimization. However, the most well-known method is SVRG introduced by Johnson and Zhang (2013) that works for

both convex and nonconvex problems. The SVRG estimator for ∇f in (FS-OPT) is given as

$$\widetilde{\nabla} f(w_t) := \nabla f(\widetilde{w}) + \nabla f_{i_t}(w_t) - \nabla f_{i_t}(\widetilde{w}), \tag{2.12}$$

where $\nabla f(\widetilde{w})$ is the full gradient of f at a snapshot point $\widetilde{w}$, and i_t is a uniformly random index in $[n]$. It is clear that $\mathbb{E}\left[\widetilde{\nabla} f(w_t) \mid \mathcal{F}_t\right] = \nabla f(w_t)$, which shows that $\widetilde{\nabla} f(w_t)$ is an unbiased estimator of $\nabla f(w_t)$. Moreover, its variance is reduced along the snapshots.

Our methods rely on the SARAH estimator introduced in Nguyen et al. (2017a) for the non-composite convex problem instances of (FS-OPT). We instead consider it in a more general setting to cover both (FS-OPT) and (St-OPT), which is defined as follows:

$$v_t := v_{t-1} + \nabla_w \mathbf{f}(w_t; \xi_t) - \nabla_w \mathbf{f}(w_{t-1}; \xi_t), \tag{2.13}$$

for a given realization ξ_t of ξ where v_0 is a snapshot gradient estimator whose definition is presented in Sections 2.3.2 and 2.3.5. Each evaluation of v_t requires two gradient evaluations. Clearly, the SARAH estimator is biased, since $\mathbb{E}\left[v_t \mid \mathcal{F}_t\right] = v_{t-1} + \nabla f(w_t) - \nabla f(w_{t-1}) \neq \nabla f(w_t)$. However, it has a variance reduced property.

2.2.4.2 Mini-batch estimators

We consider a mini-batch estimator of the gradient ∇f in (2.10) and of the SARAH estimator (2.13) respectively as follows:

$$\begin{aligned} \widetilde{\nabla} f_{\mathcal{B}_t}(w_t) &:= \frac{1}{b_t} \sum_{\xi_i \in \mathcal{B}_t} \nabla_w \mathbf{f}(w_t; \xi_i), \\ \text{and} \quad v_t &:= v_{t-1} + \frac{1}{b_t} \sum_{\xi_i \in \mathcal{B}_t} \left(\nabla_w \mathbf{f}(w_t; \xi_i) - \nabla_w \mathbf{f}(w_{t-1}; \xi_i)\right), \end{aligned} \tag{2.14}$$

where $\mathcal{B}_t$ is a mini-batch of the size $b_t := |\mathcal{B}_t| \geq 1$. For the finite-sum problem (FS-OPT), we replace $\mathbf{f}(\cdot; \xi_i)$ by $f_i(\cdot)$. In this case, $\mathcal{B}_t$ is a uniformly random subset of $[n]$. Clearly, if $b_t = n$, then we take the full gradient ∇f as the exact estimator.

2.2.5 Basic properties of stochastic and SARAH estimators

We recall some basic properties of the standard stochastic and SARAH estimators for (St-OPT) and (FS-OPT). The following result was proved in Nguyen et al. (2017a).

Lemma 2.1. *Let $\{v_t\}_{t\geq 0}$ be defined by* (2.13) *and $\mathcal{F}_t$ be defined by* (2.11)*. Then*

$$\begin{aligned}\mathbb{E}\left[v_t \mid \mathcal{F}_t\right] &= \nabla f(w_t) + \epsilon_t \quad \neq \quad \nabla f(w_t), \quad \text{where} \quad \epsilon_t := v_{t-1} - \nabla f(w_{t-1}).\\ \mathbb{E}\left[\|v_t - \nabla f(w_t)\|^2 \mid \mathcal{F}_t\right] &= \|v_{t-1} - \nabla f(w_{t-1})\|^2 + \mathbb{E}\left[\|v_t - v_{t-1}\|^2 \mid \mathcal{F}_t\right]\\ &\quad - \|\nabla f(w_t) - \nabla f(w_{t-1})\|^2.\end{aligned} \tag{2.15}$$

Consequently, for any $t \geq 0$, we have

$$\begin{aligned}\mathbb{E}\left[\|v_t - \nabla f(w_t)\|^2\right] &= \mathbb{E}\left[\|v_0 - \nabla f(w_0)\|^2\right] + \textstyle\sum_{j=1}^t \mathbb{E}\left[\|v_j - v_{j-1}\|^2\right]\\ &\quad - \textstyle\sum_{j=1}^t \mathbb{E}\left[\|\nabla f(w_j) - \nabla f(w_{j-1})\|^2\right].\end{aligned} \tag{2.16}$$

Our next result is some properties of the mini-batch estimators in (2.14). Most of the proof have been presented in Harikandeh et al. (2015); Lohr (2009); Nguyen et al. (2017b, 2020), and we only provide the missing proof of (2.20) and (2.21) in Section 2.6.1.

Lemma 2.2. *If $\widetilde{\nabla} f_{\mathcal{B}_t}(w_t)$ is generated by* (2.14)*, then, under Assumption 2.3, we have*

$$\begin{aligned}&\mathbb{E}\left[\widetilde{\nabla} f_{\mathcal{B}_t}(w_t) \mid \mathcal{F}_t\right] = \nabla f(w_t) \quad \text{and}\\ &\mathbb{E}\left[\|\widetilde{\nabla} f_{\mathcal{B}_t}(w_t) - \nabla f(w_t)\|^2 \mid \mathcal{F}_t\right] = \frac{1}{b_t}\mathbb{E}\left[\|\nabla_w \mathbf{f}(w_t;\xi) - \nabla f(w_t)\|^2 \mid \mathcal{F}_t\right] \leq \frac{\sigma^2}{b_t}.\end{aligned} \tag{2.17}$$

If $\widetilde{\nabla} f_{\mathcal{B}_t}(w_t)$ is generated by (2.14) *for the finite-sum problem* (FS-OPT)*, then*

$$\begin{aligned}&\mathbb{E}\left[\widetilde{\nabla} f_{\mathcal{B}_t}(w_t) \mid \mathcal{F}_t\right] = \nabla f(w_t) \quad \text{and}\\ &\mathbb{E}\left[\|\widetilde{\nabla} f_{\mathcal{B}_t}(w_t) - \nabla f(w_t)\|^2 \mid \mathcal{F}_t\right] \leq \tfrac{1}{b_t}\left(\tfrac{n-b_t}{n-1}\right)\sigma_n^2(w_t),\end{aligned} \tag{2.18}$$

where $\sigma_n^2(w)$ is defined as

$$\sigma_n^2(w) := \frac{1}{n}\sum_{i=1}^n \left[\|\nabla f_i(w)\|^2 - \|\nabla f(w)\|^2\right]. \tag{2.19}$$

If v_t is generated by (2.14) *for the finite-sum problem* (FS-OPT)*, then*

$$\begin{aligned}\mathbb{E}\left[\|v_t - v_{t-1}\|^2 \mid \mathcal{F}_t\right] &= \frac{n(b_t-1)}{b_t(n-1)}\|\nabla f(w_t) - \nabla f(w_{t-1})\|^2 \\ &\quad + \frac{(n-b_t)}{b_t(n-1)}\cdot\frac{1}{n}\sum_{i=1}^n \|\nabla f_i(w_t) - \nabla f_i(w_{t-1})\|^2.\end{aligned} \tag{2.20}$$

If v_t is generated by (2.14) *for the expectation problem* (St-OPT)*, then*

$$\begin{aligned}\mathbb{E}\left[\|v_t - v_{t-1}\|^2 \mid \mathcal{F}_t\right] &= \left(1 - \frac{1}{b_t}\right)\|\nabla f(w_t) - \nabla f(w_{t-1})\|^2 \\ &\quad + \frac{1}{b_t}\mathbb{E}\left[\|\nabla_w \mathbf{f}(w_t;\xi) - \nabla_w \mathbf{f}(w_{t-1};\xi)\|^2 \mid \mathcal{F}_t\right].\end{aligned} \tag{2.21}$$

Note that if $b_t = n$, i.e., we take a full gradient estimate, then the second estimate of (2.18) is vanished and independent of $\sigma_n(\cdot)$. The second term of (2.20) is also vanished.

2.3 ProxSARAH framework and convergence analysis

We describe our unified algorithmic framework and then specify it to solve different instances of (St-OPT) and (FS-OPT) under appropriate structures. The general algorithm is described in Algorithm 1, which is abbreviated by ProxSARAH.

Algorithm 1 (Proximal SARAH with stochastic recursive gradient estimators)

1: **Initialization:** An initial point $\widetilde{w}_0$ and necessary parameters $\eta_t > 0$ and $\gamma_t \in (0, 1]$ (will be specified in the sequel).
2: **Outer Loop: For** $s := 1, 2, \cdots, S$ **do**
3: Generate a snapshot $v_0^{(s)}$ at $w_0^{(s)} := \widetilde{w}_{s-1}$ using (2.34) for (St-OPT) and (2.26) for (FS-OPT).
4: Update $\widehat{w}_1^{(s)} := \mathrm{prox}_{\eta_0\psi}(w_0^{(s)} - \eta_0 v_0^{(s)})$ and $w_1^{(s)} := (1-\gamma_0)w_0^{(s)} + \gamma_0\widehat{w}_1^{(0)}$.
5: **Inner Loop: For** $t := 1, \cdots, m$ **do**
6: Generate a proper single random sample or mini-batch $\hat{\mathcal{B}}_t^{(s)}$.
7: Evaluate $v_t^{(s)} := v_{t-1}^{(s)} + \frac{1}{|\hat{\mathcal{B}}_t^{(s)}|}\sum_{\xi_t^{(s)}\in\hat{\mathcal{B}}_t^{(s)}}\left[\nabla_w \mathbf{f}(w_t^{(s)};\xi_t^{(s)}) - \nabla_w \mathbf{f}(w_{t-1}^{(s)};\xi_t^{(s)})\right]$.
8: Update $\widehat{w}_{t+1}^{(s)} := \mathrm{prox}_{\eta_t\psi}(w_t^{(s)} - \eta_t v_t^{(s)})$ and $w_{t+1}^{(s)} := (1-\gamma_t)w_t^{(s)} + \gamma_t\widehat{w}_{t+1}^{(s)}$.
9: **End For**
10: Set $\widetilde{w}_s := w_{m+1}^{(s)}$
11: **End For**

In terms of algorithm, ProxSARAH is different from SARAH where it has one proximal step followed by an additional averaging step, Step 8. However, using an approximation $\widetilde{G}_\eta$ of

the gradient mapping G_η defined by (1.5), we can view Step 8 as:

$$w_{t+1}^{(s)} := w_t^{(s)} - \eta_t\gamma_t\widetilde{G}_{\eta_t}(w_t^{(s)}), \tag{2.22}$$

where $\widetilde{G}_{\eta_t}(w_t^{(s)}) := \frac{1}{\eta_t}\big(w_t^{(s)} - \mathrm{prox}_{\eta_t\psi}(w_t^{(s)} - \eta_t v_t^{(s)})\big)$ can be considered as an approximation of $G_{\eta_t}(w_t^{(s)})$ and $\hat{\eta}_t := \eta_t\gamma_t$ can be viewed as a combined step-size. Hence, the update (2.22) is similar to the gradient step applying to the approximate gradient mapping $\widetilde{G}_{\eta_t}(w_t^{(s)})$ of F. In particular, if we set $\gamma_t = 1$, then we obtain a vanilla proximal SARAH variant which is similar to ProxSVRG, ProxSVRG+, and ProxSpiderBoost discussed above. ProxSVRG, ProxSVRG+, and ProxSpiderBoost are simply vanilla proximal gradient-type methods in stochastic setttings. If $\psi = 0$, then $\widetilde{G}_{\eta_t}(w_t^{(s)}) \equiv v_t^{(s)}$ and ProxSARAH is reduced to SARAH in Nguyen et al. (2017a,b, 2020) with a step-size $\hat{\eta}_t := \gamma_t\eta_t$. Note that Step 8 can be represented as a weighted averaging step with given weights $\{\tau_j^{(s)}\}_{j=0}^m$:

$$w_{t+1}^{(s)} := \frac{1}{\Sigma_t^{(s)}}\sum_{j=0}^{t}\tau_j^{(s)}\widehat{w}_{j+1}^{(s)}, \quad \text{where } \Sigma_t^{(s)} := \sum_{j=0}^{t}\tau_j^{(s)} \text{ and } \gamma_j^{(s)} := \frac{\tau_j^{(s)}}{\Sigma_t^{(s)}}.$$

Compared to Ghadimi and Lan (2012); Nemirovski et al. (2009), ProxSARAH evaluates v_t at the averaged point $w_t^{(s)}$ instead of $\widehat{w}_t^{(s)}$. Therefore, it can be written as

$$w_{t+1}^{(s)} := (1-\gamma_t)w_t^{(s)} + \gamma_t\mathrm{prox}_{\eta_t\psi}(w_t^{(s)} - \eta_t v_t^{(s)}),$$

which is similar to averaged fixed-point schemes (e.g., the Krasnosel'skii—Mann scheme) in the literature (see Bauschke and Combettes, 2011).

In addition, we will show in our analysis a key difference in terms of step-sizes η_t and γ_t, mini-batch, and epoch length between ProxSARAH and existing methods, including SPIDER (Fang et al., 2018) and SpiderBoost (Wang et al., 2019).

2.3.1 Analysis of the inner-loop: Key estimates

This subsection proves two key estimates of the inner loop for $t = 1$ to m. We break our analysis into two different lemmas, which provide key estimates for our convergence analysis. We assume that the mini-batch size $\hat{b} := |\hat{\mathcal{B}}_t^{(s)}|$ in the inner loop is fixed.

Lemma 2.3. *Let $\{(w_t, \widehat{w}_t)\}$ be generated by the inner-loop of Algorithm 1 with $|\hat{\mathcal{B}}_t^{(s)}| = \hat{b} \in [n-1]$ fixed. Then, under Assumption 2.2, we have*

$$\begin{aligned}\mathbb{E}\left[F(w_{m+1}^{(s)})\right] \;\leq\; & \mathbb{E}\left[F(w_0^{(s)})\right] + \frac{\rho L^2}{2}\sum_{t=0}^{m}\gamma_t\left(1+2\eta_t^2\right)\sum_{j=1}^{t}\gamma_{j-1}^2\mathbb{E}\left[\|\widehat{w}_j^{(s)} - w_{j-1}^{(s)}\|^2\right] \\ & - \frac{1}{2}\sum_{t=0}^{m}\gamma_t\left(\frac{2}{\eta_t} - L\gamma_t - 3\right)\mathbb{E}\left[\|\widehat{w}_{t+1}^{(s)} - w_t^{(s)}\|^2\right] \\ & + \frac{1}{2}\bar{\sigma}^{(s)}\Big(\sum_{t=0}^{m}\beta_t\Big) - \sum_{t=0}^{m}\frac{\gamma_t\eta_t^2}{2}\mathbb{E}\left[\|G_{\eta_t}(w_t^{(s)})\|^2\right],\end{aligned} \tag{2.23}$$

where $\bar{\sigma}^{(s)} := \mathbb{E}\left[\|v_0^{(s)} - \nabla f(w_0^{(s)})\|^2\right] \geq 0$, $\rho := \frac{1}{\hat{b}}$ if Algorithm 1 solves (St-OPT)*, and $\rho := \frac{(n-\hat{b})}{\hat{b}(n-1)}$ if Algorithm 1 solves* (FS-OPT)*.*

The proof of Lemma 2.3 is deferred to Appendix 2.6.2.1. The next lemma shows how to choose constant step-sizes γ and η by fixing other parameters in Lemma 2.3 to obtain a descent property. The proof of this lemma is given in Appendix 2.6.2.2.

Lemma 2.4. *Under Assumption 2.2 and $\hat{b} := |\hat{\mathcal{B}}_t^{(s)}| \in [n-1]$, let us choose $\eta_t = \eta > 0$ and $\gamma_t = \gamma > 0$ in Algorithm 1 such that*

$$\gamma_t = \gamma := \frac{1}{L\sqrt{\omega m}} \quad \text{and} \quad \eta_t = \eta := \frac{2\sqrt{\omega m}}{4\sqrt{\omega m}+1}, \tag{2.24}$$

where $\omega := \frac{3}{2\hat{b}}$ if Algorithm 1 solves (St-OPT) *and $\omega := \frac{3(n-\hat{b})}{2\hat{b}(n-1)}$ if Algorithm 1 solves* (FS-OPT)*. Then*

$$\mathbb{E}\left[F(w_{m+1}^{(s)})\right] \leq \mathbb{E}\left[F(w_0^{(s)})\right] - \frac{\gamma\eta^2}{2}\sum_{t=0}^{m}\mathbb{E}\left[\|G_\eta(w_t^{(s)})\|^2\right] + \frac{\gamma\theta}{2}(m+1)\bar{\sigma}^{(s)}, \tag{2.25}$$

where $\theta := 1 + 2\eta^2 \leq \frac{3}{2}$.

Remark 1. As mentioned in (2.22), the main update at Step 8 of Algorithm 1 can be written as $w_{t+1}^{(s)} := w_t^{(s)} - \eta_t\gamma_t\widetilde{G}_{\eta_t}(w_t^{(s)})$, where $\hat{\eta}_t := \eta_t\gamma_t$ can be viewed as a combined step-size. Using (2.24), we have $\hat{\eta}_t = \frac{2}{L(4\sqrt{\omega m}+1)} = \mathcal{O}\left(\frac{1}{L}\right)$. This step-size is proportional to $\frac{1}{L}$ as commonly seen in gradient-based methods (Nesterov, 2013).

2.3.2 Convergence analysis for the Problem (FS-OPT)

In this section, we specify Algorithm 1 to solve the composite finite-sum problem (FS-OPT). We replace $v_0^{(s)}$ at **Step 3** and $v_t^{(s)}$ at **Step 7** of Algorithm 1 by the following ones:

$$v_0^{(s)} := \frac{1}{b_s}\sum_{j\in\mathcal{B}_s}\nabla f_j(w_0^{(s)}), \quad \text{and} \quad v_t^{(s)} := v_{t-1}^{(s)} + \frac{1}{\hat{b}_t^{(s)}}\sum_{i\in\hat{\mathcal{B}}_t^{(s)}}\Big(\nabla f_i(w_t^{(s)}) - \nabla f_i(w_{t-1}^{(s)})\Big), \tag{2.26}$$

where $\mathcal{B}_s$ is an outer mini-batch of a fixed size $b_s := |\mathcal{B}_s| = b$, and $\hat{\mathcal{B}}_t^{(s)}$ is an inner mini-batch of a fixed size $\hat{b}_t^{(s)} := |\hat{\mathcal{B}}_t^{(s)}| = \hat{b}$. Moreover, $\mathcal{B}_s$ is independent of $\mathcal{B}_t^{(s)}$.

We consider two separate cases of this algorithmic variant: dynamic[1] step-sizes and constant step-sizes, but with fixed inner mini-batch size $\hat{b} \in [n-1]$. The following theorem proves the convergence of the dynamic step-size variant, whose proof is in Appendix 2.6.2.3.

Theorem 2.1. *Assume that we apply Algorithm 1 to solve* (FS-OPT)*, where the estimators* $v_0^{(s)}$ *and* $v_t^{(s)}$ *are defined by* (2.26) *such that* $b_s = b \in [n]$ *and* $\hat{b}_t^{(s)} = \hat{b} \in [n-1]$*, respectively. Let* $\eta_t := \eta \in (0, \frac{2}{3})$ *be fixed,* $\omega_\eta := \frac{(1+2\eta^2)(n-\hat{b})}{\hat{b}(n-1)}$*, and* $\delta := \frac{2}{\eta} - 3 > 0$*. Let* $\{\gamma_t\}_{t=0}^m$ *be the sequence of step-sizes updated in a backward mode as*

$$\gamma_m := \frac{\delta}{L}, \quad \textit{and} \quad \gamma_t := \frac{\delta}{L\big[\eta + \omega_\eta L\sum_{j=t+1}^m \gamma_j\big]}, \quad t = 0, \cdots, m-1, \tag{2.27}$$

Then, the following statements hold:

(a) *The sequence of step-sizes* $\{\gamma_t\}_{t=0}^m$ *satisfies*

$$\frac{\delta}{L(1+\delta\omega_\eta m)} \le \gamma_0 < \gamma_1 < \cdots < \gamma_m, \textit{ and } \Sigma_m := \sum_{t=0}^m \gamma_t \ge \frac{2\delta(m+1)}{L(\sqrt{2\delta\omega_\eta m + 1} + 1)}. \tag{2.28}$$

[1]We call γ_t defined by (2.27) a dynamic step-size since γ_t is computed based on its previously computed candidates $\gamma_{t+1}, \gamma_{t+2}, \cdots, \gamma_m$.

(b) *Under Assumptions 2.1 and 2.2, and* $\sigma_n^2(w)$ *defined by* (2.19) ($\sigma_n^2(w)$ *can be unbounded*), *the following bound holds:*

$$\begin{aligned}\frac{1}{S\Sigma_m}\sum_{s=1}^{S}\sum_{t=0}^{m}\gamma_t\mathbb{E}\left[\|G_\eta(w_t^{(s)})\|^2\right] &\leq \frac{2}{\eta^2 S\Sigma_m}\left[F(\widetilde{w}_0)-F^\star\right]\\ &\quad+\frac{3}{2\eta^2 S}\sum_{s=1}^{S}\frac{(n-b_s)\sigma_n^2(\widetilde{w}_{s-1})}{nb_s}.\end{aligned} \tag{2.29}$$

(c) *Under Assumptions 2.1 and 2.2, if we choose* $\eta := \frac{1}{2}$, $m := \left\lfloor \frac{n}{\hat{b}} \right\rfloor$, $b_s := n$, *and* $\hat{b} \in [1, \lfloor\sqrt{n}\rfloor]$, *then for* $\widetilde{w}_T \sim \mathbf{U}_p\big(\{w_t^{(s)}\}_{t=0\to m}^{s=1\to S}\big)$ *such that*

$$\mathbf{Prob}\left(\widetilde{w}_T = w_t^{(s)}\right) = p_{(s-1)m+t} := \frac{\gamma_t}{S\Sigma_m},$$

we have

$$\mathbb{E}\left[\|G_\eta(\widetilde{w}_T)\|^2\right] \leq \frac{4\sqrt{6}L\left[F(\widetilde{w}_0)-F^\star\right]}{S\sqrt{n}}. \tag{2.30}$$

Consequently, the number of outer iterations S *needed to obtain an output* $\widetilde{w}_T$ *of Algorithm 1 such that* $\mathbb{E}\left[\|G_\eta(\widetilde{w}_T)\|^2\right] \leq \varepsilon^2$ *is at most* $S := \frac{4\sqrt{6}L[F(\widetilde{w}_0)-F^\star]}{\sqrt{n}\varepsilon^2}$. *Moreover, if* $1 \leq n \leq \frac{96L^2[F(\widetilde{w}_0)-F^\star]^2}{\varepsilon^4}$, *then* $S \geq 1$.

The number of individual stochastic gradient evaluations ∇f_i *does not exceed*

$$\mathcal{T}_{\rm grad} := \frac{20\sqrt{6}L\sqrt{n}\left[F(\widetilde{w}_0)-F^\star\right]}{\varepsilon^2} = \mathcal{O}\left(\frac{L\sqrt{n}}{\varepsilon^2}\left[F(\widetilde{w}_0)-F^\star\right]\right).$$

The number of $\mathrm{prox}_{\eta\psi}$ *operations does not exceed* $\mathcal{T}_{\rm prox} := \frac{4\sqrt{6}(\sqrt{n}+1)L[F(\widetilde{w}_0)-F^\star]}{\hat{b}\varepsilon^2}$.

Remark 2. When n is sufficiently large, if we choose $b_s < n$, then to guarantee convergence of Algorithm 1 for solving (FS-OPT), we need to impose Assumption 2.3 and choose $b_s := \mathcal{O}\left(n \wedge \varepsilon^{-2}\right)$. Then we can derive similar conclusions as in Theorem 2.1(c).

Alternatively, Theorem 2.2 below shows the convergence of Algorithm 1 for the constant step-size case, whose proof is given in Appendix 2.6.2.4.

Theorem 2.2. *Assume that we apply Algorithm 1 to solve* (FS-OPT), *where the estimators* $v_0^{(s)}$ *and* $v_t^{(s)}$ *are defined by* (2.26) *such that* $b_s = b \in [n]$.

Let us choose constant step-sizes $\gamma_t = \gamma$ *and* $\eta_t = \eta$ *as*

$$\gamma := \frac{1}{L\sqrt{\omega m}} \quad \text{and} \quad \eta := \frac{2\sqrt{\omega m}}{4\sqrt{\omega m}+1}, \quad \text{where} \quad \omega := \frac{3(n-\hat{b})}{2\hat{b}(n-1)} \quad \text{and} \quad \hat{b} \in [1, \lfloor\sqrt{n}\rfloor]. \tag{2.31}$$

Then, under Assumptions 2.1 and 2.2, if we choose $\hat{b}$ *such that* $\hat{b} \in [1, \lfloor\sqrt{n}\rfloor]$, $m := \lfloor \frac{n}{\hat{b}} \rfloor$, $b_s := n$, *and* $\widetilde{w}_T \sim \mathbf{U}\big(\{w_t^{(s)}\}_{t=0\to m}^{s=1\to S}\big)$, *then the number of outer iterations* S *to achieve* $\mathbb{E}\left[\|G_\eta(\widetilde{w}_T)\|^2\right] \le \varepsilon^2$ *does not exceed*

$$S := \frac{16\sqrt{3}L}{\sqrt{2n}\varepsilon^2}\big[F(\widetilde{w}_0) - F^\star\big].$$

Moreover, if $n \le \frac{384L^2}{\varepsilon^4}\big[F(\widetilde{w}_0) - F^\star\big]^2$, *then* $S \ge 1$.

Consequently, the number of stochastic gradient evaluations $\mathcal{T}_{\text{grad}}$ *does not exceed*

$$\mathcal{T}_{\text{grad}} := \frac{16\sqrt{3}L\sqrt{n}}{\sqrt{2}\varepsilon^2}\big[F(\widetilde{w}_0) - F^\star\big] = \mathcal{O}\left(\frac{L\sqrt{n}}{\varepsilon^2}\big[F(\widetilde{w}_0) - F^\star\big]\right).$$

The number of $\text{prox}_{\eta\psi}$ *operations does not exceed* $\mathcal{T}_{\text{prox}} := \frac{16\sqrt{3}L(\sqrt{n}+1)}{\hat{b}\sqrt{2}\varepsilon^2}\big[F(\widetilde{w}_0) - F^\star\big]$.

Note that the condition $n \le \mathcal{O}\left(\varepsilon^{-4}\right)$ is to guarantee that $S \ge 1$ in Theorems 2.1 and 2.2. In this case, our complexity bound is $\mathcal{O}\left(n^{1/2}\varepsilon^{-2}\right)$. Otherwise, when $n > \mathcal{O}\left(\varepsilon^{-4}\right)$, then our complexity becomes $\mathcal{O}\left(n + n^{1/2}\varepsilon^{-2}\right)$ due to the full gradient snapshots. In the non-composite setting, this complexity is the same as SPIDER (Fang et al., 2018), and the range of our mini-batch size $\hat{b} \in [1, \sqrt{n}]$, which is the same as in SPIDER, instead of fixed $\hat{b} = \lfloor\sqrt{n}\rfloor$ as in SpiderBoost (Wang et al., 2019). We can extend our mini-batch size $\hat{b}$ such that $\sqrt{n} < \hat{b} \le n-1$, but our complexity bound is no longer the best-known one.

The step-size η in (2.31) can be bounded by $\eta \in [\frac{2}{5}, \frac{1}{2}]$ for any batch-size $\hat{b}$ and m instead of fixing at $\eta = \frac{1}{2}$. Nevertheless, this interval can be enlarged by slightly modifying the proof of Lemma 2.3. For example, we can show that η can go up to $\frac{2}{3}$ by appropriately manipulating the parameters in the proof of Lemma 2.3. The step-size $\gamma \in (0, 1]$ can change from a small to a large value close to 1 as the batch-size $\hat{b}$ and the epoch length m change as we will discuss in Section 2.3.4.

2.3.3 Lower-bound complexity for the Problem (FS-OPT)

Let us analyze a special case of (FS-OPT) with $\psi = 0$. We consider any stochastic first-order methods to generate an iterate sequence $\{w_t\}$ as follows:

$$[w_t, i_t] := \mathcal{A}^{t-1}\left(\omega, \nabla f_{i_0}(w^0), \nabla f_{i_1}(w^1), \cdots, \nabla f_{i_{t-1}}(w^{t-1})\right), \quad t \geq 1, \tag{2.32}$$

where $\mathcal{A}^{t-1}$ are measure mapping into $\mathbb{R}^{d+1}$, f_{i_t} is an individual function chosen by $\mathcal{A}^{t-1}$ at iteration t, $\omega \sim \mathbf{U}([0,1])$ is a random vector, and $[w^0, i_0] := \mathcal{A}^0(\omega)$. Clearly, Algorithm 1 can be cast as a special case of (2.32). As shown in Fang et al. (2018, Theorem 3) and later in Zhou and Gu (2019, Theorem 4.5.), under Assumptions 2.1 and 2.2, for any $L > 0$ and $2 \leq n \leq \mathcal{O}\left(L^2\left[F(w^0) - F^\star\right]^2 \varepsilon^{-4}\right)$, there exists a dimension $d = \widetilde{\mathcal{O}}(L^2\left[F(w^0) - F^\star\right]^2 n^2 \varepsilon^{-4})$ such that the lower-bound complexity of Algorithm 1 to produce an output $\widetilde{w}_T$ such that $\mathbb{E}\left[\|\nabla f(\widetilde{w}_T)\|^2\right] \leq \varepsilon^2$ is $\Omega\left(\frac{L\left[F(w^0)-F^\star\right]\sqrt{n}}{\varepsilon^2}\right)$. This lower-bound clearly matches the upper bound $\mathcal{T}_{\text{grad}}$ in Theorems 2.1 and 2.2 up to a given constant factor.

2.3.4 Mini-batch size and learning rate trade-offs

Although our step-size defined by (2.31) in the single sample case is much larger than that of ProxSVRG (Reddi et al., 2016b, Theorem 1), it still depends on $\sqrt{m}$, where m is the epoch length. To obtain larger step-sizes, we can choose m and the mini-batch size $\hat{b}$ using the same trick as in Reddi et al. (2016b, Theorem 2). Let us first fix $\gamma := \bar{\gamma} \in (0, 1]$. From (2.31), we have $\omega m = \frac{1}{L^2\bar{\gamma}^2}$. It makes sense to choose $\bar{\gamma}$ close to 1 in order to use new information from $\widehat{w}_{t+1}^{(s)}$ instead of the old one in $w_t^{(s)}$.

Our goal is to choose m and $\hat{b}$ such that $\omega m = \frac{3(n-\hat{b})m}{2\hat{b}(n-1)} = \frac{1}{L^2\bar{\gamma}^2}$. If we define $C := \frac{2}{3L^2\bar{\gamma}^2}$, then the last condition implies that $\hat{b} := \frac{mn}{Cn+m-C} \leq \frac{m}{C}$ provided that $m \geq C$. Our suggestion is to choose

$$\gamma := \bar{\gamma} \in (0, 1], \quad \hat{b} := \left\lfloor \frac{mn}{Cn + m - C} \right\rfloor, \quad \text{and} \quad \eta := \frac{2}{4 + L\bar{\gamma}}. \tag{2.33}$$

If we choose $m = \lfloor n^{1/3} \rfloor$, then $\hat{b} = \mathcal{O}\left(n^{1/3}\right) \leq \frac{n^{1/3}}{C}$. This mini-batch size is much smaller than $\lfloor n^{2/3} \rfloor$ in ProxSVRG. Note that, in ProxSVRG, they set $\gamma := 1$ and $\eta := \frac{1}{3L}$.

In ProxSpiderBoost (Wang et al., 2019), m and the mini-batch size $\hat{b}$ were chosen as $m = \hat{b} = \lfloor n^{1/2} \rfloor$ so that they can use constant step-sizes $\gamma = 1$ and $\eta = \frac{1}{2L}$. In our case, if $\gamma = 1$, then $\eta = \frac{2}{4+L}$. Hence, if $L = 1$, then $\eta_{\text{ProxSpiderBoost}} = \frac{1}{2} > \eta_{\text{ProxSARAH}} = \frac{2}{5} > \eta_{\text{ProxSVRG}} = \frac{1}{3}$. But if $L > 4$, then our step-size $\eta_{\text{ProxSARAH}}$ dominates $\eta_{\text{ProxSpiderBoost}}$. However, by manipulating some parameters in the proof of Lemma 2.3, we can obtain $\eta_{\text{ProxSARAH}} = \frac{2}{3}$, which shows that $\eta_{\text{ProxSARAH}} > \eta_{\text{ProxSpiderBoost}} = \frac{1}{2}$ when $L = 1$.

If we choose $m = \mathcal{O}\left(n^{1/2}\right)$ and $\hat{b} = \mathcal{O}\left(n^{1/2}\right)$, then we maintain the same complexity bound $\mathcal{O}\left(n^{1/2}\varepsilon^{-2}\right)$ as in Theorems 2.1 and 2.2. Nevertheless, if we choose $m = \mathcal{O}\left(n^{1/3}\right)$ and $\hat{b} = \mathcal{O}\left(n^{1/3}\right)$, then the complexity bound becomes $\mathcal{O}\left((n^{2/3} + n^{1/3})\varepsilon^{-2}\right)$, which is similar to ProxSVRG. The choice of m in Theorem 2.1 affects the values of $\{\gamma_t\}_{t=0}^m$. Hence, a reasonably small value of m is recommended in the dynamic step-size case.

2.3.5 Convergence analysis for the Problem (St-OPT)

In this section, we apply Algorithm 1 to solve the composite expectation setting (St-OPT). In this case, we generate the snapshot at **Step 3** of Algorithm 1 as follows:

$$v_0^{(s)} := \frac{1}{b_s} \sum_{\zeta_i^{(s)} \in \mathcal{B}_s} \nabla_w \mathbf{f}(w_0^{(s)}; \zeta_i^{(s)}), \tag{2.34}$$

where $\mathcal{B}_s := \left\{\zeta_1^{(s)}, \cdots, \zeta_{b_s}^{(s)}\right\}$ is a mini-batch of i.i.d. realizations of ξ at the s-th outer iteration and independent of ξ_t from the inner loop, and $b_s := |\mathcal{B}_s| = b \geq 1$ is fixed.

Now, we analyze the convergence of Algorithm 1 for solving (St-OPT) using (2.34) above. For simplicity of discussion, we only consider the constant step-size case. The dynamic step-size variant can be derived similarly as in Theorem 2.1 and we omit the details. The proof of the following theorem can be found in Appendix 2.6.2.5.

Theorem 2.3. *Let us apply Algorithm 1 to solve* (St-OPT) *using* (2.34) *for* $v_0^{(s)}$ *at* ***Step 3*** *of Algorithm 1 with fixed outer loop batch-size* $b_s = b \geq 1$ *and inner loop batch-size* $\hat{b} := |\mathcal{B}_t^{(s)}| \geq 1$.

If we choose fixed step-sizes γ *and* η *as*

$$\gamma := \frac{1}{L\sqrt{\bar{\omega}m}} \quad \text{and} \quad \eta := \frac{2\sqrt{\bar{\omega}m}}{4\sqrt{\bar{\omega}m} + 1}, \quad \text{with} \;\; \bar{\omega} := \frac{3}{2\hat{b}}, \tag{2.35}$$

then, under Assumptions 2.1, 2.2, and 2.3, we have the following estimate:

$$\frac{1}{(m+1)S}\sum_{s=1}^{S}\sum_{t=0}^{m}\mathbb{E}\left[\|G_\eta(w_t^{(s)})\|^2\right] \le \frac{2}{\gamma\eta^2(m+1)S}\big[F(\widetilde{w}_0)-F^\star\big] + \frac{3\sigma^2}{2\eta^2 b}. \tag{2.36}$$

In particular, if we choose $b := \left\lfloor \frac{75\sigma^2}{\varepsilon^2}\right\rfloor$ *and* $m := \left\lfloor \frac{\sigma^2}{\hat{b}\varepsilon^2}\right\rfloor$ *for* $\hat{b} \le \frac{\sigma^2}{\varepsilon^2}$*, then after at most*

$$S := \frac{32L[F(\widetilde{w}_0)-F^\star]}{\sigma\varepsilon}$$

outer iterations, we obtain $\mathbb{E}\left[\|G_\eta(\widetilde{w}_T)\|^2\right] \le \varepsilon^2$*, where* $\widetilde{w}_T \sim \mathbf{U}\big(\{w_t^{(s)}\}_{t=0\to m}^{s=1\to S}\big)$.

Consequently, the number of individual stochastic gradient evaluations $\nabla_w f(w_t^{(s)};\xi_t)$ *and the number of proximal operations* $\mathrm{prox}_{\eta\psi}$*, respectively do not exceed:*

$$\mathcal{T}_{\mathrm{grad}} := \frac{2464\sigma L[F(\widetilde{w}_0)-F^\star]}{\varepsilon^3}, \quad \textit{and} \quad \mathcal{T}_{\mathrm{prox}} := \frac{32\sigma L[F(\widetilde{w}_0)-F^\star]}{\hat{b}\varepsilon^2}.$$

If $\sigma = 0$, i.e., no stochasticity involved in our problem (St-OPT), then (2.36) reduces to

$$\frac{1}{(m+1)S}\sum_{s=1}^{S}\sum_{t=0}^{m}\mathbb{E}\left[\|G_\eta(w_t^{(s)})\|^2\right] \le \frac{2}{\gamma\eta^2(m+1)S}\big[F(\widetilde{w}_0)-F^\star\big],$$

where the expectation is taken over all the randomness generated by the algorithm. From this bound, we can derive the well-known $\mathcal{O}\left(\varepsilon^{-2}\right)$ oracle complexity bound for gradient-based methods in the deterministic case as often seen in the literature.

If $\sigma > 0$, then Theorem 2.3 achieves the best-known complexity $\mathcal{O}\left(\sigma L\varepsilon^{-3}\right)$ for the composite expectation problem (St-OPT) as long as $\sigma \le \frac{32L[F(\widetilde{w}_0)-F^\star]}{\varepsilon^2}$. Otherwise, our complexity is $\mathcal{O}\left(\sigma\varepsilon^{-3}+\sigma^2\varepsilon^{-2}\right)$ due to the snapshot gradient for evaluating $v_0^{(s)}$. This complexity is the same as SPIDER (Fang et al., 2018) in the non-composite setting and ProxSpiderBoost (Wang et al., 2019) in the mini-batch setting. It also matches the lower bound complexity recently studied in Arjevani et al. (2019) up to a constant under the same set of assumptions, but only for the non-composite setting of (St-OPT). Hence, our complexity is nearly optimal. Note that our method does not require to perform mini-batch in the inner loop, i.e., it is independent of $\hat{\mathcal{B}}_t^{(s)}$, and the mini-batch is independent of the number of iterations m of the inner loop, while in Wang et al. (2019), the mini-batch size $|\hat{\mathcal{B}}_t^{(s)}|$ must be proportional to $\sqrt{|\mathcal{B}_s|} = \mathcal{O}\left(\varepsilon^{-1}\right)$, where

$\mathcal{B}_s$ is the mini-batch of the outer loop. This is perhaps the reason why ProxSpiderBoost can take a large constant step-size $\eta = \frac{1}{2L}$ as discussed in Section 2.3.4.

Remark 3. The constants have not been optimized in the complexity bounds of all theorems above including Theorems 2.1, 2.2, and 2.3. The analysis can be refined to possibly obtain smaller constants in these complexity bounds by manipulating different parameters.

2.4 Dynamic step-sizes for non-composite problems

In this section, we consider the non-composite settings of (St-OPT) and (FS-OPT) as special cases of Algorithm 1. Note that if we solely apply Algorithm 1 with constant step-sizes to solve the non-composite case of (St-OPT) and (FS-OPT) when $\psi \equiv 0$, then by using the same step-size as in Theorems 2.1, 2.2, and 2.3, we can obtain the same complexity as stated in Theorems 2.1, 2.2, and 2.3, respectively. However, we will modify our proof of Theorem 2.1 to take advantage of the extra term $\sum_{t=0}^{m} \frac{\gamma_t}{2} \mathbb{E}\left[\|\nabla f(w_t^{(s)}) - v_t^{(s)} - (\widehat{w}_{t+1}^{(s)} - w_t^{(s)})\|^2\right]$ in the proof of Lemma 2.3. The proof of this theorem is given in Appendix 2.6.2.6.

Theorem 2.4. *Let* $\{w_t^{(s)}\}$ *be generated by a variant of Algorithm 1 to solve the non-composite instance of* (St-OPT) *or* (FS-OPT) *using the following update for both Step 4 and Step 8:*

$$w_{t+1}^{(s)} := w_t^{(s)} - \hat{\eta}_t v_t^{(s)}. \tag{2.37}$$

Let $\rho := \frac{1}{\hat{b}}$ *for* (St-OPT) *and* $\rho := \frac{n-\hat{b}}{\hat{b}(n-1)}$ *for* (FS-OPT)*, and* $\hat{\eta}_t$ *is computed recursively as:*

$$\hat{\eta}_m = \frac{1}{L} \quad \text{and} \quad \hat{\eta}_t := \frac{1}{L\big(1 + \rho L \sum_{j=t+1}^{m} \hat{\eta}_j\big)}, \quad \forall t = 0, \cdots, m-1. \tag{2.38}$$

Then, we have $\Sigma_m := \sum_{t=0}^{m} \hat{\eta}_t \geq \frac{2(m+1)}{(\sqrt{2\rho m+1}+1)L}$.

Suppose that Assumptions 2.1 and 2.2 hold. Then, we have

$$\frac{1}{S\Sigma_m} \sum_{s=1}^{S} \sum_{t=0}^{m} \hat{\eta}_t \mathbb{E}\left[\|\nabla f(w_t^{(s)})\|^2\right] \leq \frac{(\sqrt{2\nu m+1}+1)L}{S(m+1)} \big[f(\widetilde{w}_0) - f^\star\big] + \frac{1}{S} \sum_{s=1}^{S} \hat{\sigma}_s, \tag{2.39}$$

where $\hat{\sigma}_s := \mathbb{E}\left[\|\nabla f(w_0^{(s)}) - v_0^{(s)}\|^2\right]$.

Let $\widetilde{w}_T \sim \mathbf{U}_p\big(\{w_t^{(s)}\}_{t=0\to m}^{s=1\to S}\big)$ *such that* $\mathbf{Prob}\left(\widetilde{w}_T = w_t^{(s)}\right) = p_{(s-1)m+t} := \frac{\hat{\eta}_t}{S\Sigma_m}$ *for all* $s = 1, \cdots, S$ *and* $t = 0, \cdots, m$*, be the output of Algorithm 1. Then:*

(a) ***The finite-sum case:*** *If we apply this variant of Algorithm 1 to solve* (FS-OPT) *with* $\psi = 0$ *using* $b_s := n$*,* $m := \lfloor \frac{n}{\hat{b}} \rfloor$*, and* $\hat{b} \in [1, \sqrt{n}]$*, then under Assumptions 2.1 and 2.2 the following holds:*

$$\mathbb{E}\left[\|\nabla f(\widetilde{w}_T)\|^2\right] \leq \frac{2L}{S\sqrt{n}}[f(\widetilde{w}_0) - f^\star]. \tag{2.40}$$

Consequently, the total of outer iterations S *to achieve* $\mathbb{E}\left[\|\nabla f(\widetilde{w}_T)\|^2\right] \leq \varepsilon^2$ *does not exceed* $S := \frac{2L[f(\widetilde{w}_0)-f^\star]}{\sqrt{n}\varepsilon^2}$*. The number of individual stochastic gradient evaluations* ∇f_i *does not exceed* $\mathcal{T}_{\rm grad} := \frac{10\sqrt{n}L[f(\widetilde{w}_0)-f^\star]}{\varepsilon^2}$*.*

(b) ***The expectation case:*** *If we apply this variant of Algorithm 1 to solve* (St-OPT) *with* $\psi = 0$ *using* $b_s = b := \frac{2\sigma^2}{\varepsilon^2}$ *for the outer-loop,* $m := \frac{\sigma^2}{\hat{b}\varepsilon^2}$*, and* $\hat{b} \leq \frac{\sigma^2}{\varepsilon^2}$*, then under Assumptions 2.1, 2.2, and 2.3:*

$$\mathbb{E}\left[\|\nabla f(\widetilde{w}_T)\|^2\right] \leq \frac{2L}{S\sqrt{\hat{b}m}}\big[f(\widetilde{w}_0) - f^\star\big] + \frac{\sigma^2}{b}. \tag{2.41}$$

Consequently, the total of outer iterations S *to achieve* $\mathbb{E}\left[\|\nabla f(\widetilde{w}_T)\|^2\right] \leq \varepsilon^2$ *does not exceed* $S := \frac{4L[f(\widetilde{w}_0)-f^\star]}{\sigma\varepsilon}$*. The number of individual stochastic gradient evaluations does not exceed* $\mathcal{T}_{\rm grad} := \frac{16\sigma L[f(\widetilde{w}_0)-f^\star]}{\varepsilon^3}$*, provided that* $\sigma \leq \frac{8L[f(\widetilde{w}_0)-f^\star]}{\varepsilon}$*.*

Note that the first statement (a) of Theorem 2.4 covers the nonconvex case of Nguyen et al. (2019) by fixing step-size $\hat{\eta}_t = \hat{\eta} = \frac{2}{L(1+\sqrt{4m+1})}$. However, this constant step-size is rather small if $m \leq \mathcal{O}(n)$ is large. Hence, it is better to update $\hat{\eta}_t$ dynamically increasing as in (2.38), where $\hat{\eta}_m = \frac{1}{L}$ is a large step-size. In addition, Nguyen et al. (2019) only study the finite-sum problem, while we also consider the expectation setting (St-OPT).

Again, combining the first statement (a) of Theorem 2.4 and the complexity lower bound in Fang et al. (2018), we conclude that this variant still achieves a nearly-optimal complexity $\mathcal{O}\left(n^{1/2}\varepsilon^{-2}\right)$ for the non-composite finite-sum problem in (FS-OPT) to find an ε-stationary point in expectation if $n \leq \mathcal{O}\left(\varepsilon^{-4}\right)$. In Statement (b), if $\sigma > \frac{8L[f(\widetilde{w}_0)-f^\star]}{\varepsilon}$, then the complexity of our method is $\mathcal{O}\left(\sigma^2\varepsilon^{-2} + \sigma\varepsilon^{-3}\right)$ due to the gradient snapshot of the size $b = \mathcal{O}\left(\sigma^2\varepsilon^{-2}\right)$ to evaluate $v_0^{(s)}$. Note that this complexity matches the lower bound in Arjevani et al. (2019).

2.5 Numerical experiments

We present three numerical examples to illustrate our theory and compare our methods with state-of-the-art algorithms in the literature. We implement 8 different variants of our ProxSARAH algorithm:

- ProxSARAH-v1: Single sample and fixed step-sizes $\gamma := \frac{\sqrt{2}}{L\sqrt{3m}}$ and $\eta := \frac{2\sqrt{3m}}{4\sqrt{3m}+\sqrt{2}}$.
- ProxSARAH-v2: $\gamma := 0.95$ and mini-batch size $\hat{b} := \lfloor \frac{\sqrt{n}}{C} \rfloor$ and $m := \lfloor \sqrt{n} \rfloor$.
- ProxSARAH-v3: $\gamma := 0.99$ and mini-batch size $\hat{b} := \lfloor \frac{\sqrt{n}}{C} \rfloor$ and $m := \lfloor \sqrt{n} \rfloor$.
- ProxSARAH-v4: $\gamma := 0.95$ and mini-batch size $\hat{b} := \lfloor \frac{n^{\frac{1}{3}}}{C} \rfloor$ and $m := \lfloor n^{\frac{1}{3}} \rfloor$.
- ProxSARAH-v5: $\gamma := 0.99$ and mini-batch size $\hat{b} := \lfloor \frac{n^{\frac{1}{3}}}{C} \rfloor$ and $m := \lfloor n^{\frac{1}{3}} \rfloor$.
- ProxSARAH-A-v1: Single sample (i.e., $\hat{b} = 1$), and dynamic step-sizes.
- ProxSARAH-A-v2: $\gamma_m := 0.99$ and mini-batch size $\hat{b} := \lfloor \sqrt{n} \rfloor$ and $m := \lfloor \sqrt{n} \rfloor$.
- ProxSARAH-A-v3: $\gamma_m := 0.99$ and mini-batch size $\hat{b} := \lfloor n^{\frac{1}{3}} \rfloor$ and $m := \lfloor n^{\frac{1}{3}} \rfloor$.

Here, C is given in Section 2.3.4. We also implement 4 other algorithms:

- ProxSVRG: The proximal SVRG algorithm in Reddi et al. (2016b) for single sample with theoretical step-size $\eta = \frac{1}{3nL}$, and for the mini-batch case with $\hat{b} := \lfloor n^{2/3} \rfloor$, the epoch length $m := \lfloor n^{1/3} \rfloor$, and the step-size $\eta := \frac{1}{3L}$.
- ProxSpiderBoost: The proximal SpiderBoost method in Wang et al. (2019) with $\hat{b} := \lfloor \sqrt{n} \rfloor$, $m := \lfloor \sqrt{n} \rfloor$, and step-size $\eta := \frac{1}{2L}$.
- ProxSGD: Proximal stochastic gradient descent scheme (Ghadimi and Lan, 2013) with step-size $\eta_t := \frac{\eta_0}{1+\tilde{\eta}\lfloor t/n \rfloor}$, where $\eta_0 > 0$ and $\tilde{\eta} \geq 0$ will be given in each example.
- ProxGD: Standard proximal gradient descent algorithm with step-size $\eta := \frac{1}{L}$.

All algorithms are implemented in Python running on a single node of a Linux server (called Longleaf) with configuration: 3.40GHz Intel processors, 30M cache, and 256GB RAM. For the neural network example, we implement these algorithms in **TensorFlow** (Abadi et al., 2016) running on a GPU system. The code is available online at

https://github.com/unc-optimization/StochasticProximalMethods.

To be fair for comparison, we compute the norm of gradient mapping $\|G_\eta(w_t^{(s)})\|$ for visualization at the same value $\eta := 0.5$ in all methods. To compute the relative loss residuals $\frac{F(\widetilde{w}_T)-F^*}{|F^*|}$, we

use $F^* := \min\left\{\widetilde{F}_j^* \mid j\right\}$ as the minimum loss values $\widetilde{F}_j^*$ generated by all algorithms. To increase the readability of figures, we only plot the performance of some representative variants among the 8 instead of reporting them all. We run the first and second examples for 20 and 30 epochs, respectively whereas we increase it up to 150 and 300 epochs in the last example. Several data sets used in this section are from Chang and Lin (2011)[2]. Two other well-known data sets are `mnist`[3] and `fashion_mnist`[4].

2.5.1 Nonnegative principal component analysis

We reconsider the problem of non-negative principal component analysis (NN-PCA) studied in Reddi et al. (2016b). More precisely, for a given set of samples $\{z_i\}_{i=1}^n$ in $\mathbb{R}^d$, we solve the following constrained nonconvex problem:

$$f^\star := \min_{w\in\mathbb{R}^d}\Big\{f(w) := -\frac{1}{2n}\sum_{i=1}^n w^\top(z_i z_i^\top)w \mid \|w\| \le 1,\ w \ge 0\Big\}. \tag{2.42}$$

By defining $f_i(w) := -\frac{1}{2}w^\top(z_i z_i^\top)w$ for $i = 1,\cdots,n$, and $\psi(w) := \delta_{\mathcal{X}}(w)$, the indicator of $\mathcal{X} := \big\{w \in \mathbb{R}^d \mid \|w\| \le 1, w \ge 0\big\}$, we can formulate (2.42) into (FS-OPT). Moreover, since z_i is normalized, the Lipschitz constant of ∇f_i is $L = 1$ for $i = 1,\cdots,n$. Since (2.42) is nonconvex, it may have different stationary points. For a given algorithm to approximate a good stationary point of (2.42), it crucially depends on initial point. Following Reddi et al. (2016b), we use ProxSGD to generate an initial point and use it for all algorithms.

(a) ***Small and medium data sets:*** We test all the algorithms on three different well-known data sets: `mnist` ($n = 60000$, $d = 784$), `rcv1-binary` ($n = 20242$, $d = 47236$), and `real-sim` ($n = 72309$, $d = 20958$). In ProxSGD, after manipulating different values, we set $\eta_0 := 0.1$ and $\tilde{\eta} := 1.0$ that allow us to obtain good performance.

Experiment 1 (*Single sample comparison*)*:* We first verify our theory by running 5 algorithms with single sample (i.e., $\hat{b} = 1$). The relative objective residuals and the absolute norm of gradient mappings of these algorithms after 20 epochs are plotted in Figure 2.1.

[2]Available online at https://www.csie.ntu.edu.tw/∼cjlin/libsvm/

[3]Available online at http://yann.lecun.com/exdb/mnist/

[4]Available online at https://github.com/zalandoresearch/fashion-mnist

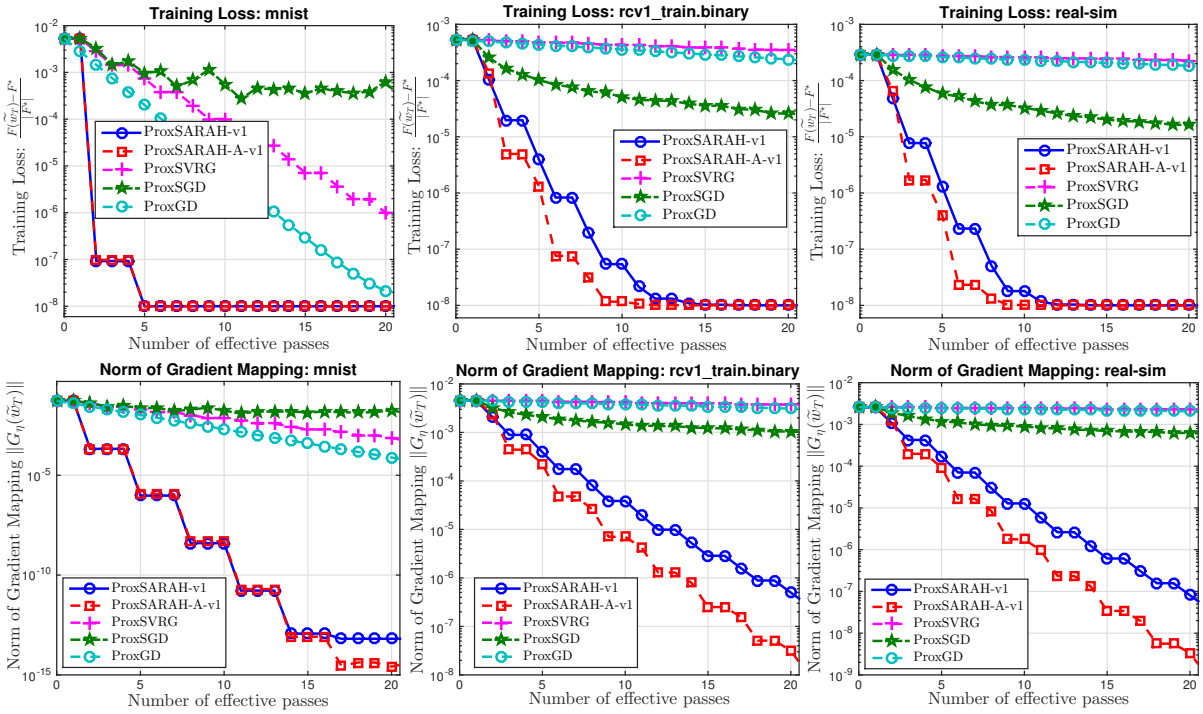


Figure 2.1: The objective value residuals and gradient mapping norms of (2.42) on three data sets: `mnist`, `rcv1-binary`, and `real-sim`.

Figure 2.1 shows that both ProxSARAH-v1 and its dynamic variant work really well and dominate all other methods. ProxSARAH-A-v1 is still better than ProxSARAH-v1. ProxSVRG is slow since its theoretical step-size $\frac{1}{3nL}$ is too small.

Experiment 2 (The effect of mini-batch sizes on ProxSARAH): In this experiment, we evaluate the effect of mini-batch sizes on the performance of ProxSARAH by running ProxSARAH on these data sets with different mini-batch sizes. We choose $\hat{b}$ among 6 values $\{n^{1/2}, 0.75n^{1/2}, 0.5n^{1/2}, 0.25n^{1/2}, 0.1n^{1/2}, 0.05n^{1/2}\}$. The results are shown in Figure 2.2.

As we can see from Figure 2.2 that the performance of each particular batch-size varies between data sets. Variants with larger mini-batch sizes work well in the `mnist` data set while variants with smaller mini-batch sizes are better in `rcv1_train.binary` and `real-sim`. It is unclear for our methods to show that a larger mini-batch size leads to a better performance or vice versa. Therefore, to achieve the best performance, a search over mini-batch size is recommended for each particular data set.

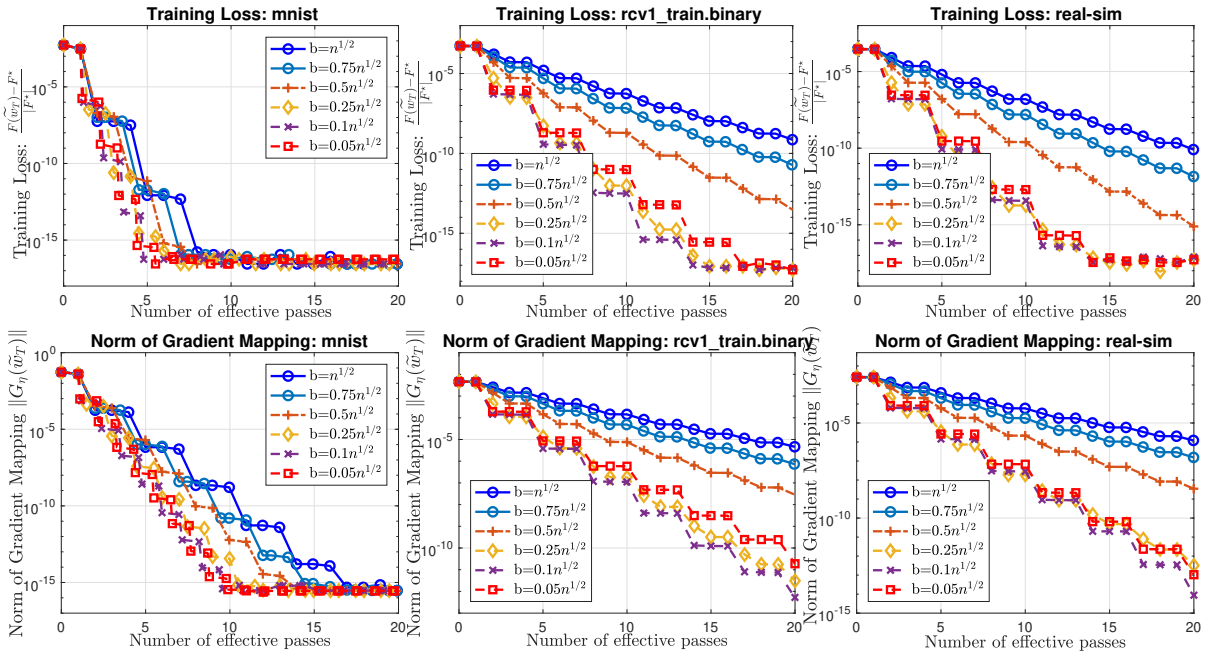


Figure 2.2: The relative objective residuals and the norms of gradient mappings of Prox-SARAH algorithms with different mini-batch sizes for solving (2.42) on three data sets: `mnist`, `rcv1-binary`, and `real-sim`.

Experiment 3 (*Mini-batch comparison*)*:* Next, we run all the mini-batch variants of the methods described above to solve (2.42). The relative objective residuals and the norms of gradient mapping are plotted in Figure 2.3.

From Figure 2.3, we observe that ProxSpiderBoost works well since it has a large step-size $\eta = \frac{1}{2L}$, and it is comparable with ProxSARAH-A-v2. The variants with $\hat{b} = \mathcal{O}\left(n^{\frac{1}{3}}\right)$ of ProxSARAH and ProxSARAH-A perform well for `mnist` data set while the variants with $\hat{b} = \mathcal{O}\left(n^{\frac{1}{2}}\right)$ are better for the other two data sets. Although ProxSVRG takes $\eta = \frac{1}{3L}$, its choice of batch-size and epoch length also affects the performance resulting in a slower convergence. ProxSGD has good progress at early stage but then its relative objective residual is saturated around 10^{-5} accuracy. Also, its gradient mapping norms do not significantly decrease as in ProxSARAH variants or ProxSpiderBoost. Note that ProxSARAH variants with large step-size γ (e.g., $\gamma = 0.99$) are very similar to ProxSpiderBoost which results in resemblance in their performance.

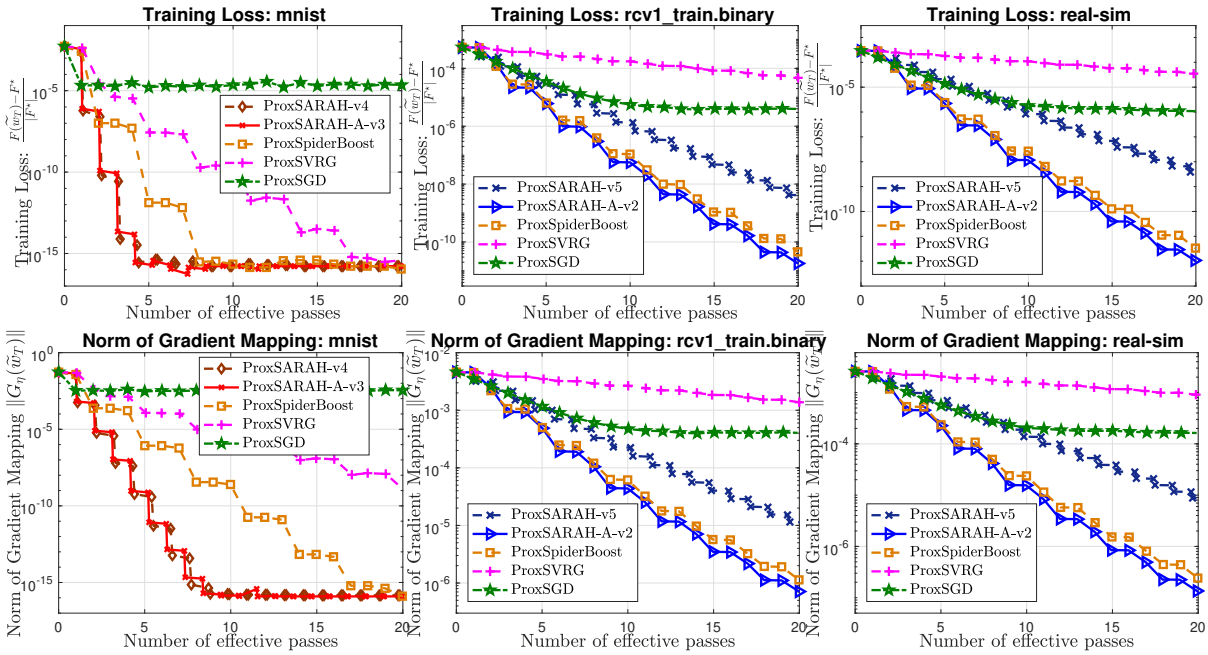


Figure 2.3: The relative objective residuals and the norms of gradient mappings of 5 algorithms for solving (2.42) on three data sets: `mnist`, `rcv1-binary`, and `real-sim`.

(b) ***Large data sets:*** Now, we test these algorithms on larger data sets: `url_combined` ($n = 2,396,130; d = 3,231,961$), `news20.binary` ($n = 19,996; d = 1,355,191$), and `avazu-app` ($n = 14,596,137; d = 999,990$). The relative objective residuals and the absolute norms of gradient mapping of this experiment are depicted in Figure 2.4.

Experiment 4 (*Mini-batch comparison on large data sets*)*:* Figure 2.4 shows that ProxSARAH variants still work well and depend on the data set in which ProxSARAH-A-v2 or the variants with $\hat{b} = \mathcal{O}(n^{\frac{1}{3}})$ dominates other algorithms. In this experiment, ProxSpiderBoost gives smaller gradient mapping norms for `url_combined` and `avazu-app` in the last epochs than the others. However, these algorithms have achieved up to 10^{-13} accuracy in absolute values, the improvement of ProxSpiderBoost may not be necessary. With the same step-size as in the previous test, ProxSGD performs quite poorly on these three data sets, and we did not report its performance here.

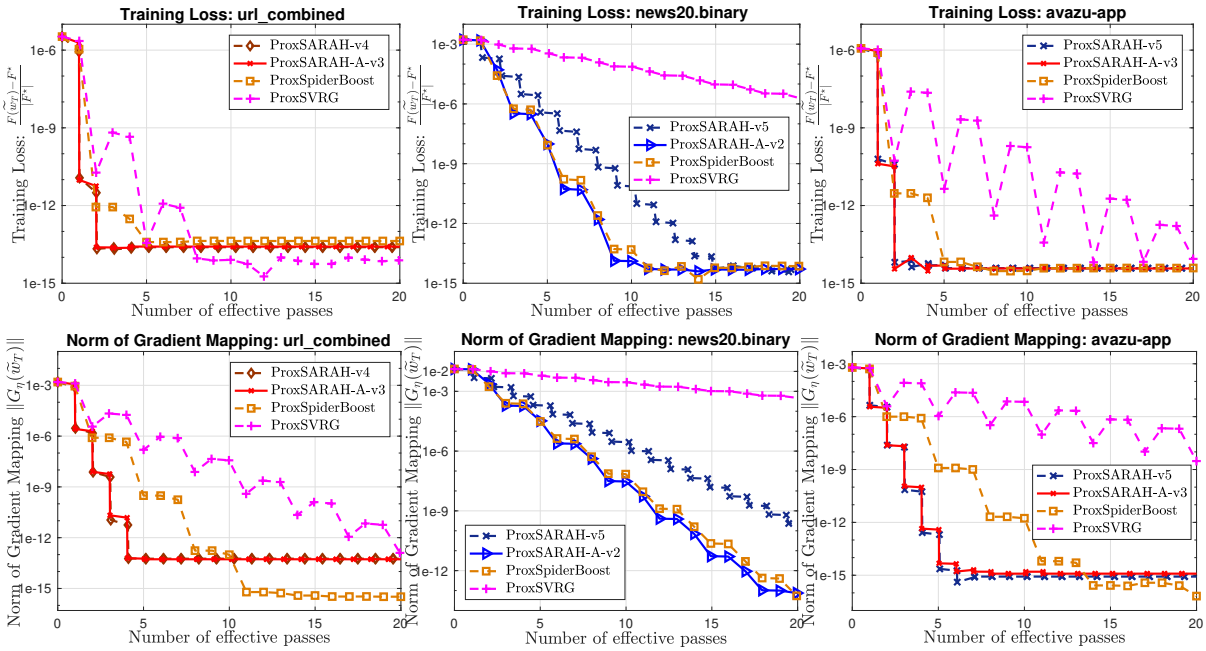


Figure 2.4: The relative objective residuals and the absolute gradient mapping norms of 4 algorithms for solving (2.42) on three data sets: `url_combined`, `news20.binary`, and `avazu-app`.

2.5.2 Sparse binary classification with nonconvex losses

We consider the following sparse binary classification involving nonconvex loss function:

$$\min_{w\in\mathbb{R}^d}\left\{F(w) := \frac{1}{n}\sum_{i=1}^{n}\ell(a_i^\top w, b_i) + \lambda\|w\|_1\right\}, \tag{2.43}$$

where $\{(a_i, b_i)\}_{i=1}^n \subset \mathbb{R}^d \times \{-1, 1\}^n$ is a given training data set, $\lambda > 0$ is a regularization parameter, and $\ell(\cdot,\cdot)$ is a given smooth and nonconvex loss function as studied in Zhao et al. (2010). By setting $f_i(w) := \ell(a_i^\top w, b_i)$ and $\psi(w) := \lambda\|w\|_1$ for $i \in [n]$, we obtain (FS-OPT).

The loss function ℓ is chosen from one of the following three cases (Zhao et al., 2010):

- ***Normalized sigmoid loss:*** $\ell_1(s,\tau) := 1-\tanh(\omega\tau s)$ for a given $\omega > 0$. Since $\left|\frac{d^2\ell_1(s,\tau)}{ds^2}\right| \le \frac{8(2+\sqrt{3})(1+\sqrt{3})\omega^2\tau^2}{(3+\sqrt{3})^2}$ and $|\tau| = 1$, we can show that $\ell_1(\cdot,\tau)$ is L-smooth with respect to s, where $L := \frac{8(2+\sqrt{3})(1+\sqrt{3})\omega^2}{(3+\sqrt{3})^2} \approx 0.7698\omega^2$.
- ***Nonconvex loss in 2-layer neural networks:*** $\ell_2(s,\tau) := \left(1 - \frac{1}{1+\exp(-\tau s)}\right)^2$. For this function, we have $\left|\frac{d^2\ell_2(s,\tau)}{ds^2}\right| \le 0.15405\tau^2$. If $|\tau| = 1$, then this function is also L-smooth with $L = 0.15405$.

- ***Logistic difference loss:*** $\ell_3(s,\tau) := \ln(1+\exp(-\tau s)) - \ln(1+\exp(-\tau s - \omega))$ for some $\omega > 0$. With $\omega = 1$, we have $|\frac{d^2\ell_3(s,\tau)}{ds^2}| \le 0.092372\tau^2$. Therefore, if $|\tau| = 1$, then this function is also L-smooth with $L = 0.092372$.

We set the regularization parameter $\lambda := \frac{1}{n}$ in all the tests, which gives us relatively sparse solutions. We test the above algorithms on different scenarios ranging from small to large data sets, where we use 6 different data sets from LIBSVM.

(a) ***Small and medium data sets:*** We consider three small to medium data sets: `rcv1.binary` ($n = 20,242$, $d = 47,236$), `real-sim` ($n = 72,309$, $d = 20,958$), and `epsilon` ($n = 400,000$, $d = 2,000$).

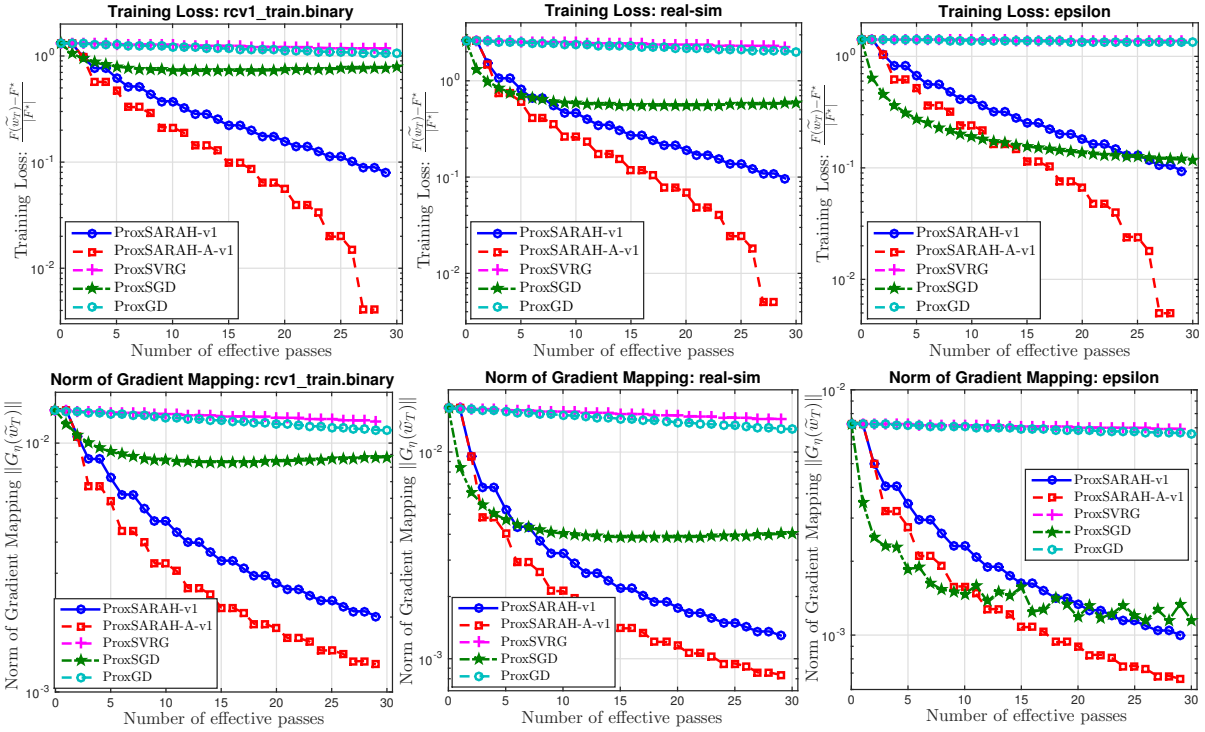


Figure 2.5: The relative objective residuals and gradient mapping norms of (2.43) on three data sets using the loss $\ell_2(s,\tau)$ - The single sample case.

Experiment 5 (*Singe sample comparison on* (2.43))*:* Figure 2.5 shows the relative objective residuals and the gradient mapping norms on these three data sets for the loss function $\ell_2(\cdot)$ in the single sample case. Similar to the first example, ProxSARAH-v1 and its dynamic variant work well, whereas ProxSARAH-A-v1 is better. ProxSVRG is still slow due to small step-size. ProxSGD appears to be better than ProxSVRG and ProxGD within 30 epochs.

Now, we test the loss function $\ell_2(\cdot)$ with the mini-batch variants using the same three data sets. Figure 2.6 shows the results on these data sets.

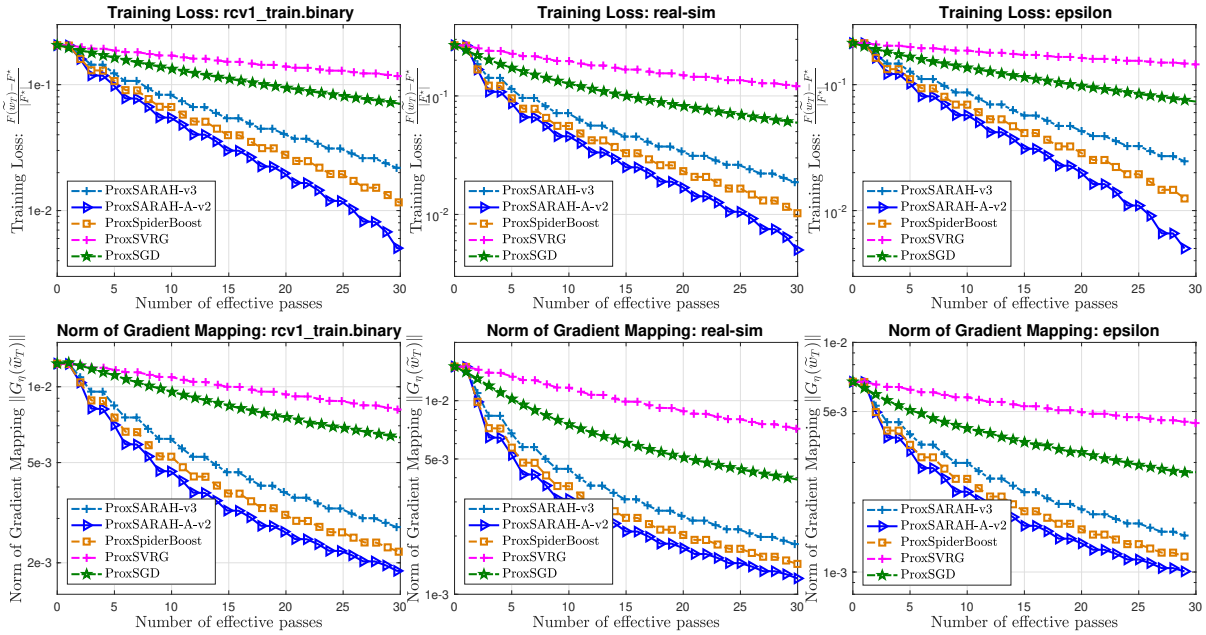


Figure 2.6: The relative objective residuals and gradient mapping norms of (2.43) on three data sets using the loss $\ell_2(s,\tau)$ - The mini-batch case.

We can see that ProxSARAH-A-v2 is the most effective algorithm whereas ProxSpiderBoost also performs well due to large step-size as discussed. ProxSVRG remains slow in this test. Notice that ProxSARAH dynamic variants normally perform better than their corresponding fixed step-size variants in this experiment. Additionally, ProxSARAH-A-v2 still preserves the best-known complexity $\mathcal{O}\left(n + n^{1/2}\varepsilon^{-2}\right)$.

Experiment 6 (*Mini-batch comparison on* (2.43))*:* To further illustrate the advantage of the increasing step-size, we run ProxSARAH and ProxSARAH-A with different mini-batch sizes and select the top two variants of each for comparison when applying to solve (2.43) using the loss function ℓ_2. Their results along with the chosen mini-batch sizes are depicted in Figure 2.7. We can see that ProxSARAH-A performs better than ProxSARAH in all three data sets which confirms the advantage of the dynamic step-size scheme.

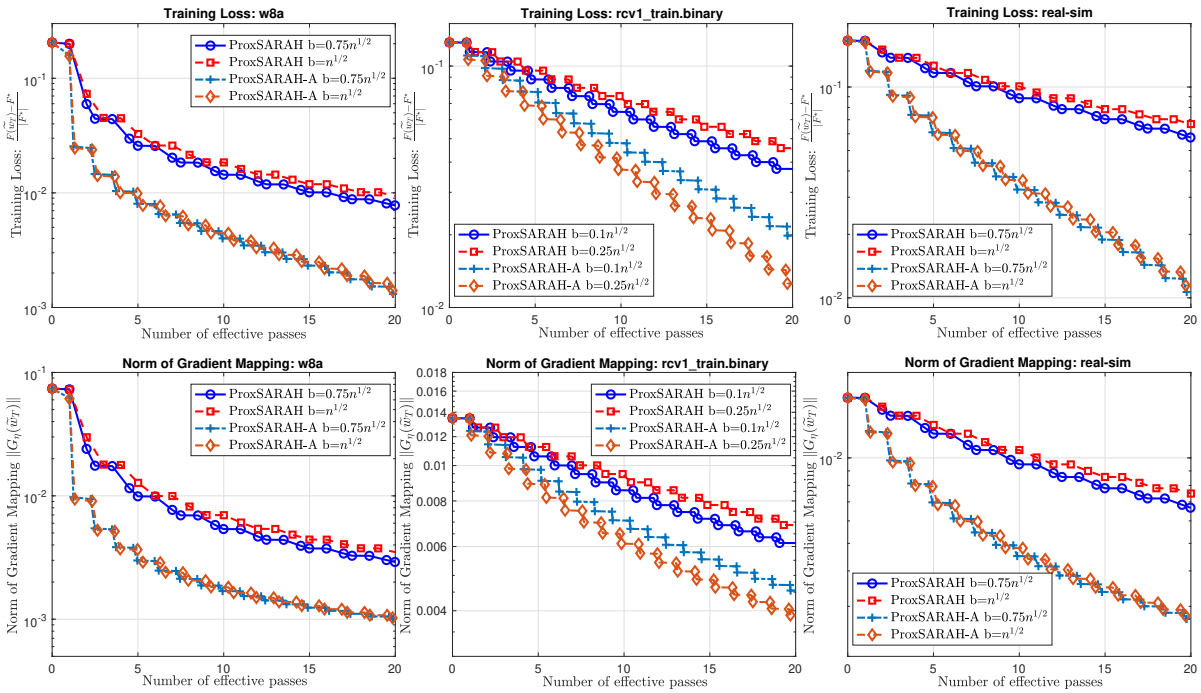


Figure 2.7: The relative objective residuals and gradient mapping norms of (2.43) on three data sets using the loss $\ell_2(s,\tau)$.

(b) ***Large data sets:*** Next, we test these algorithms on three large data sets: `url_combined` ($n = 2,396,130$, $d = 3,231,961$), `avazu-app` ($n = 14,596,137$, $d = 999,990$), and `kddb-raw` ($n = 19,264,097$, $d = 3,231,961$).

Experiment 7 (Mini-batch comparison on large data sets): Since we use large data sets, we only test the mini-batch variants. Figure 2.8 presents the results on these data sets.

Again, we can observe from Figure 2.8 that, ProxSARAH-A-v2 achieves the best performance. ProxSpiderBoost also works well in this experiment while ProxSVRG are comparable with ProxSARAH-v1 and ProxSARAH-v2. ProxSGD also has good performance but is not as good as ProxSpiderBoost and ProxSARAH variants.

The complete results on these three data sets with three loss functions are presented in Table 2.2. Apart from the relative objective residuals and gradient mapping norms, the table consists of both training and test accuracies where we use 10% of the data set to evaluate the testing accuracy.

Among three loss functions, the loss ℓ_2 gives the best training and testing accuracy. The accuracy is consistent with the result reported in Zhao et al. (2010). ProxSGD seems to give

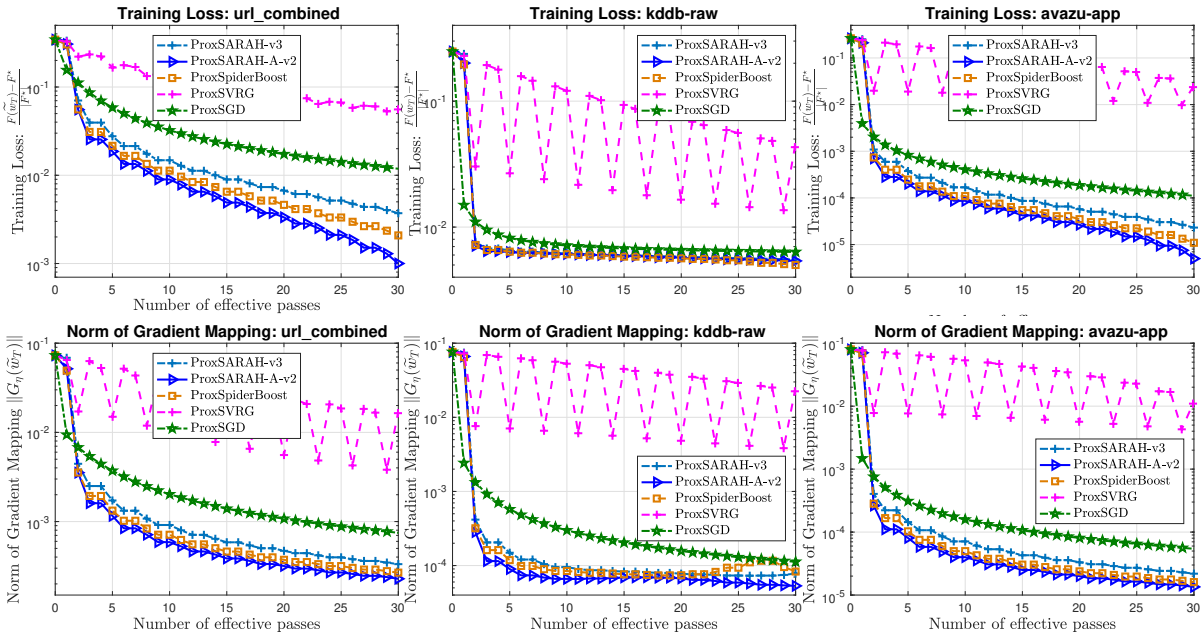


Figure 2.8: The relative objective residuals and gradient mapping norms of (2.43) on three large data sets using the loss $\ell_2(s,\tau)$ - The mini-batch case.

Algorithms	$\|G_\eta(\widetilde{w}_T)\|^2$			$(F(w_T)-F^*)/\|F^*\|$			Training Accuracy			Test Accuracy		
	ℓ_1-Loss	ℓ_2-Loss	ℓ_3-Loss	ℓ_1-Loss	ℓ_2-Loss	ℓ_3-Loss	ℓ_1-Loss	ℓ_2-Loss	ℓ_3-Loss	ℓ_1-Loss	ℓ_2-Loss	ℓ_3-Loss
`url_combined` ($n = 2,396,130$, $d = 3,231,961$)												
ProxSARAH-v2	2.534e-06	5.827e-08	1.181e-07	1.941e-01	1.397e-02	8.092e-02	0.965	0.9684	0.9657	0.9636	0.9672	0.9646
ProxSARAH-v3	2.772e-06	5.515e-08	1.110e-07	2.065e-01	9.149e-03	7.399e-02	0.965	0.9685	0.9658	0.9635	0.9673	0.9647
ProxSARAH-v4	1.252e-05	6.003e-06	1.433e-05	4.749e-01	8.210e-01	1.597e+00	0.962	0.9617	0.9558	0.9614	0.9607	0.9528
ProxSARAH-v5	1.182e-05	5.595e-06	1.346e-05	4.617e-01	7.931e-01	1.546e+00	0.962	0.9617	0.9568	0.9615	0.9609	0.9537
ProxSARAH-A-v2	1.115e-06	**4.969e-08**	**5.215e-08**	9.225e-02	**1.076e-05**	**1.268e-05**	**0.966**	**0.9687**	**0.9672**	0.9645	**0.9676**	**0.9662**
ProxSARAH-A-v3	1.034e-05	3.639e-07	4.555e-07	4.325e-01	1.946e-01	2.619e-01	0.962	0.9644	0.9634	0.9616	0.9631	0.9625
ProxSpiderBoost	1.375e-06	6.454e-08	7.158e-08	1.178e-01	2.274e-02	2.947e-02	0.965	0.9681	0.9664	0.9641	0.9669	0.9653
ProxSVRG	7.391e-03	2.043e-04	2.697e-04	2.196e+00	1.091e+00	1.490e+00	0.958	0.9601	0.9595	0.9570	0.9585	0.9579
ProxSGD	**5.005e-07**	2.340e-07	5.963e-07	**4.446e-03**	1.406e-01	3.062e-01	0.968	0.9651	0.9633	**0.9667**	0.9637	0.9624
`avazu-app` ($n = 14,596,137$, $d = 999,990$)												
ProxSARAH-v2	8.647e-09	1.053e-08	5.074e-10	4.354e-04	1.958e-03	1.687e-04	0.883	0.8843	0.8834	0.8615	0.8617	0.8615
ProxSARAH-v3	9.757e-09	9.792e-09	4.776e-10	4.615e-04	1.397e-03	1.554e-04	0.883	**0.8844**	0.8834	0.8615	0.8617	0.8615
ProxSARAH-v4	9.087e-08	3.179e-07	1.841e-07	1.738e-03	5.102e-02	9.816e-03	0.883	0.8834	0.8834	0.8615	0.8615	0.8615
ProxSARAH-v5	8.568e-08	3.029e-07	1.702e-07	1.675e-03	5.036e-02	9.433e-03	0.883	0.8834	0.8834	0.8615	0.8615	0.8615
ProxSARAH-A-v2	3.062e-09	**8.724e-09**	**1.814e-10**	2.046e-04	**5.467e-07**	**1.388e-08**	0.883	**0.8844**	0.8834	0.8615	0.8617	0.8615
ProxSARAH-A-v3	7.784e-08	5.124e-08	4.405e-09	1.604e-03	2.499e-02	1.223e-03	0.883	0.8834	0.8834	0.8615	0.8615	0.8615
ProxSpiderBoost	4.050e-09	1.152e-08	2.579e-10	2.626e-04	3.090e-03	5.073e-05	0.883	0.8842	0.8834	0.8615	0.8617	0.8615
ProxSVRG	4.218e-03	1.309e-03	1.202e-04	3.137e-01	4.287e-01	2.031e-01	0.883	0.8648	0.8834	0.8615	0.8146	0.8615
ProxSGD	**9.063e-10**	2.839e-08	3.150e-09	**6.449e-06**	1.595e-02	9.536e-04	0.883	0.8835	0.8834	0.8615	0.8616	0.8615
`kddb-raw` ($n = 19,264,097$, $d = 3,231,961$)												
ProxSARAH-v2	2.013e-08	1.770e-08	5.688e-09	7.235e-04	3.455e-03	4.295e-03	0.862	0.8654	0.8619	0.8531	0.8560	0.8534
ProxSARAH-v3	2.168e-08	1.669e-08	6.105e-09	7.903e-04	2.275e-03	3.741e-03	0.862	0.8655	0.8619	0.8530	0.8561	0.8534
ProxSARAH-v4	2.265e-07	4.066e-07	2.796e-07	3.862e-03	9.196e-02	2.203e-02	0.862	0.8617	0.8615	0.8530	0.8533	0.8531
ProxSARAH-v5	2.127e-07	3.943e-07	2.600e-07	3.725e-03	9.098e-02	2.152e-02	0.862	0.8617	0.8615	0.8530	0.8533	0.8531
ProxSARAH-A-v2	7.955e-09	**1.490e-08**	**2.830e-09**	2.106e-04	**8.502e-07**	2.829e-03	0.862	**0.8656**	**0.8621**	0.8531	**0.8562**	**0.8536**
ProxSARAH-A-v3	1.951e-07	1.036e-07	9.293e-09	3.539e-03	4.887e-02	9.223e-03	0.862	0.8627	0.8616	0.8530	0.8544	0.8531
ProxSpiderBoost	9.867e-09	1.906e-08	6.889e-09	3.082e-04	5.249e-03	**5.026e-07**	0.862	0.8652	0.8619	0.8531	0.8559	0.8534
ProxSVRG	1.225e-02	1.105e-03	5.040e-04	3.541e-01	3.471e-01	2.780e-01	0.860	0.8611	0.8599	0.8518	0.8529	0.8519
ProxSGD	**6.027e-09**	8.899e-08	1.331e-08	**2.593e-05**	4.320e-02	9.937e-03	0.862	0.8629	0.8616	0.8530	0.8546	0.8531

Table 2.2: The results of 9 algorithms on three data sets: url_combined, avazu-app, and kddb-raw.

good results on the ℓ_1-loss, but ProxSARAH-A-v2 is the best for the ℓ_2 and ℓ_3-losses in the majority of the test.

2.5.3 Feedforward neural network training

We consider the following composite nonconvex optimization model arising from a feedforward neural network configuration:

$$\min_{w\in\mathbb{R}^d}\left\{F(w) := \frac{1}{n}\sum_{i=1}^{n}\ell\big(h(w,a_i),b_i\big) + \psi(w)\right\}, \tag{2.44}$$

where we concatenate all the weight matrices and bias vectors of the neural network in one vector of variable w, $\{(a_i,b_i)\}_{i=1}^n$ is a training data set, $h(\cdot)$ is a composition between all linear transforms and activation functions as $h(w,a) := \boldsymbol{\sigma}_l(W_l\boldsymbol{\sigma}_{l-1}(W_{l-1}\boldsymbol{\sigma}_{l-2}(\cdots\boldsymbol{\sigma}_0(W_0a+\mu_0)\cdots)+\mu_{l-1})+\mu_l)$, where W_i is a weight matrix, μ_i is a bias vector, $\boldsymbol{\sigma}_i$ is an activation function, l is the number of layers, $\ell(\cdot)$ is the soft-max cross-entropy loss, and ψ is a convex regularizer (e.g., $\psi(w) := \lambda\|w\|_1$ for some $\lambda > 0$ to obtain sparse weights). Again, by defining $f_i(w) := \ell(h(w,a_i),b_i)$ for $i\in[n]$, we can bring (2.44) into the same composite finite-sum setting (FS-OPT).

We implement our algorithms and other methods in TensorFlow (Abadi et al., 2016) and use two data sets `mnist` and `fashion_mnist` to evaluate their performance. In the first experiment, we use a one-hidden-layer fully connected neural network: $784\times100\times10$ for both `mnist` and `fashion_mnist`. The activation function $\boldsymbol{\sigma}_i$ of the hidden layer is ReLU and the loss function is soft-max cross-entropy. To estimate the Lipschitz constant L, we normalize the input data. The regularization parameter λ is set at $\lambda := \frac{1}{n}$ and $\psi(\cdot) := \lambda\|\cdot\|_1$.

Experiment 8 ($784\times100\times10$ *network*)*:* We first test ProxSARAH, ProxSVRG, ProxSpiderBoost, and ProxSGD using mini-batch. For ProxSGD, we use the mini-batch $\hat{b} = 245$, $\eta_0 = 0.1$, and $\tilde{\eta} = 0.5$ for both data sets. For the `mnist` data set, we tune $L = 1$ then follow the configuration in Section 2.3.4 to choose η, γ, m, and $\hat{b}$ for ProxSARAH variants. We also tune the learning rate for ProxSVRG at $\eta = 0.2$, and for ProxSpiderBoost at $\eta = 0.12$. However, for the `fashion_mnist` data set, it requires a smaller learning rate. Therefore, we choose $L = 4$ for ProxSARAH and follow the theory in Section 2.3.4 to set η, γ, m, and $\hat{b}$. We also tune the

learning rate for ProxSVRG and ProxSpiderBoost until they are stabilized to obtain the best possible step-size in this example as $\eta_{\text{ProxSVRG}} = 0.11$ and $\eta_{\text{ProxSpiderBoost}} = 0.15$, respectively.

Figure 2.9 shows the convergence of different variants of ProxSARAH, ProxSpiderBoost, ProxSVRG, and ProxSGD on three criteria for `mnist` and `fashion_mnist`: training loss values, the absolute norm of gradient mapping, and the test accuracy. For ProxSARAH, we find that two variants with $\hat{b} = \mathcal{O}\left(n^{\frac{1}{2}}\right)$ and $\hat{b} = \mathcal{O}\left(n^{\frac{1}{3}}\right)$ perform well among other choices.

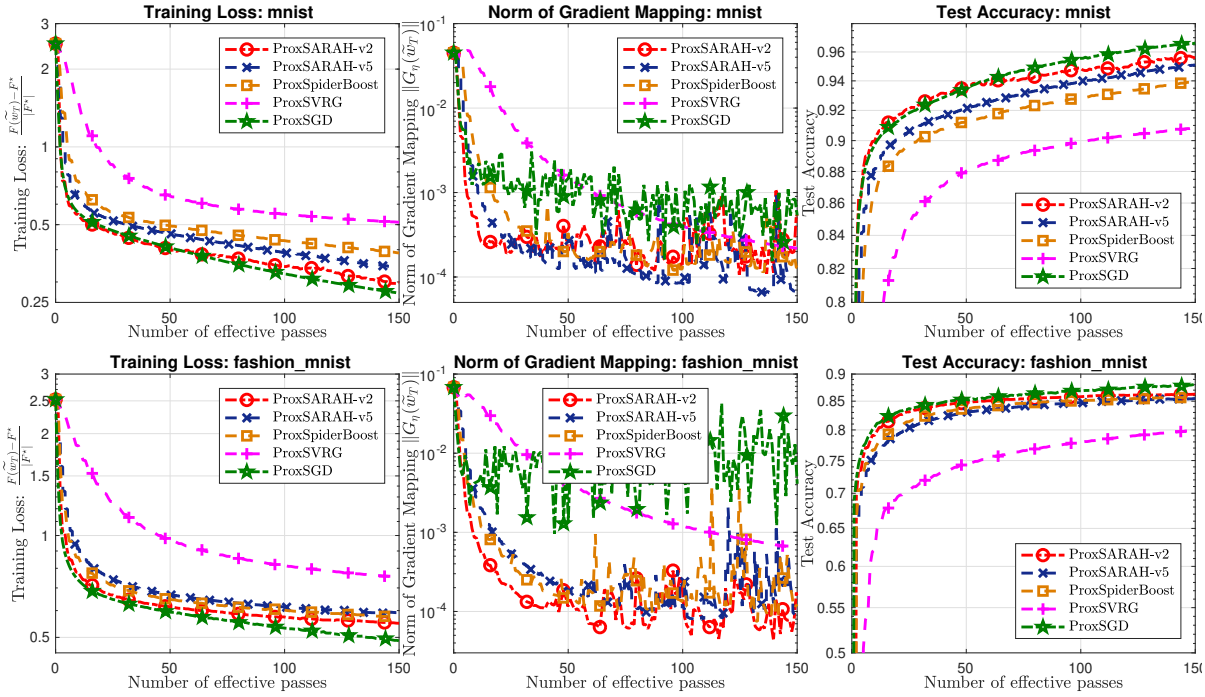


Figure 2.9: The training loss, gradient mapping, and test accuracy on `mnist` (top line) and `fashion_mnist` (bottom line) of 5 algorithms.

In this example, ProxSGD appears to be the best in terms of training loss and test accuracy. However, the norm of gradient mapping is rather different from others, relatively large, and oscillated. ProxSVRG is clearly slower than ProxSpiderBoost due to smaller learning rate. The two variants of ProxSARAH perform relatively well, but the variants with $\hat{b} = \mathcal{O}(\sqrt{n})$ seem to be slightly better. Note that the norm of gradient mapping tends to be decreasing but still oscillated since perhaps we are taking the last iterate instead of a random choice of intermediate iterates as stated in the theory.

Experiment 9 ($784 \times 800 \times 10$ *network*)*:* Finally, we test the above algorithm on `mnist` using a $784 \times 800 \times 10$ network as known to give a better test accuracy. We run all algorithms for 300 epochs and the results are given in Figure 2.10.

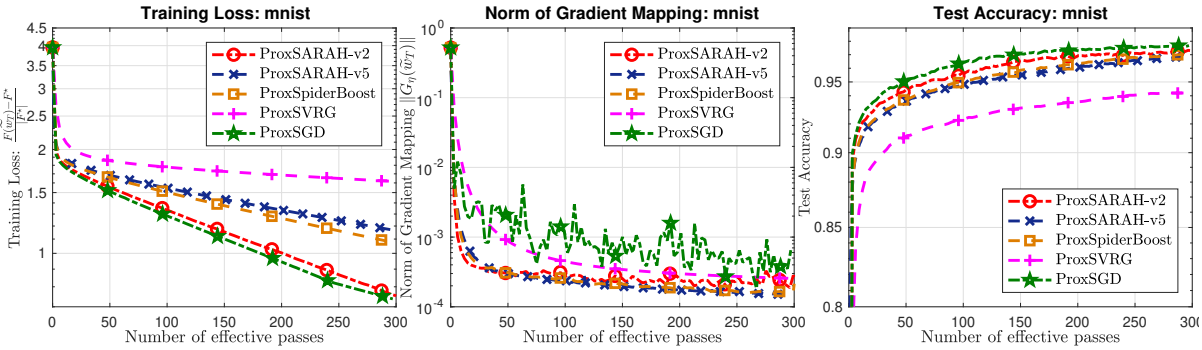


Figure 2.10: The training loss, gradient mapping, and test accuracy on `mnist` of 5 algorithms on a $784 \times 800 \times 10$ neural network (See http://yann.lecun.com/exdb/mnist/).

As we can see from Figure 2.10 that ProxSARAH-v2, ProxSARAH-v3, and ProxSGD performs really well in terms of training loss and test accuracy. However, our method can achieve lower as well as less oscillated gradient mapping norm than ProxSGD. Also, ProxSpiderBoost has similar performance to ProxSARAH-v4 and ProxSARAH-v5. ProxSVRG again does not have a good performance in this example in terms of loss and test accuracy but is slightly better than ProxSGD regarding gradient mapping norm.

2.6 Proofs of technical results

Before moving to the proofs of main results, we first introduce a technical lemma that is useful to prove the convergence of ProxSARAH with dynamic step-size. We also provide the proof for the properties of stochastic estimators presented in Lemma 2.2.

2.6.1 Technical lemma

This section provides the missing proofs of Lemma 2.2 and one elementary result, Lemma 2.5, used in our analysis in the sequel.

Lemma 2.5. *Given three positive constants ν, δ, and L, let $\{\gamma_t\}_{t=0}^m$ be a positive sequence satisfying the following conditions:*

$$\begin{cases} L\gamma_m - \delta & \le 0, \\ \nu L^2 \gamma_t \sum_{j=t+1}^m \gamma_j - \delta + L\gamma_t & \le 0, \quad t = 0, \cdots, m-1. \end{cases} \tag{2.45}$$

Then, the following statements hold:

(a) *The sequence $\{\gamma_t\}_{t=0}^m$ computed recursively in a backward mode as*

$$\gamma_m := \frac{\delta}{L}, \quad \text{and} \quad \gamma_t := \frac{\delta}{L\big[1 + \nu L \sum_{j=t+1}^m \gamma_j\big]}, \quad t = 0, \cdots, m-1, \tag{2.46}$$

tightly satisfies (2.45). *Moreover, we have $\frac{\delta}{L(1+\delta\nu m)} < \gamma_0 < \gamma_1 < \cdots < \gamma_m$ and*

$$\Sigma_m := \sum_{t=0}^m \gamma_t \ge \frac{2\delta(m+1)}{L\big[\sqrt{1 + 2\delta\nu m} + 1\big]}. \tag{2.47}$$

(b) *The constant sequence $\{\gamma_t\}_{t=0}^m$ with $\gamma_t := \frac{2\delta}{L(\sqrt{1+4\delta\nu m}+1)}$ satisfies* (2.45).

Proof. (a) The sequence $\{\gamma_t\}_{t=0}^m$ given by (2.46) is in fact computed from (2.45) by setting all the inequalities "$\le$" to equalities "$=$". Hence, it automatically satisfies (2.45). Moreover, it is obvious that $\gamma_0 < \gamma_1 < \cdots < \gamma_m$. Since $\sum_{t=1}^m \gamma_t < m\gamma_m = \frac{m\delta}{L}$, we have $\gamma_0 > \frac{\delta}{L(1+\delta\nu m)}$.

Let $\Sigma_m := \sum_{t=0}^m \gamma_t$. Using Σ_m into (2.45) with all equalities, we can rewrite it as

$$\begin{cases} \nu L^2 \gamma_m \Sigma_m & = \delta - L\gamma_m & + \nu L^2(\gamma_m^2 + \gamma_m\gamma_{m-1} + \gamma_m\gamma_{m-2} + \cdots + \gamma_m\gamma_0) \\ \nu L^2 \gamma_{m-1} \Sigma_m & = \delta - L\gamma_{m-1} & + \nu L^2(\gamma_{m-1}^2 + \gamma_{m-1}\gamma_{m-2} + \gamma_{m-1}\gamma_{m-3} + \cdots + \gamma_{m-1}\gamma_0) \\ \cdots & \cdots & \cdots \\ \nu L^2 \gamma_1 \Sigma_m & = \delta - L\gamma_1 & + \nu L^2(\gamma_1^2 + \gamma_1\gamma_0) \\ \nu L^2 \gamma_0 \Sigma_m & = \delta - L\gamma_0 & + \nu L^2 \gamma_0^2. \end{cases}$$

Summing up both sides of these equations, and using the definition of Σ_m and $S_m^2 := \sum_{t=0}^m \hat{\eta}_t^2$, we obtain

$$\nu L^2 \Sigma_m^2 = (m+1)\delta - L\Sigma_m + \frac{\nu L^2}{2}(\Sigma_m^2 + S_m^2).$$

Since $(m+1)S_m^2 \geq \Sigma_m^2$ by the Cauchy-Schwarz inequality, the last expression leads to

$$\nu L^2 \Sigma_m^2 + 2L\Sigma_m - 2\delta(m+1) = \nu L^2 S_m^2 \geq \frac{\nu L^2 \Sigma_m^2}{m+1}.$$

Therefore, by solving the quadratic inequation $\nu m L^2 \Sigma_m^2 + 2(m+1)L\Sigma_m - 2\delta(m+1)^2 \geq 0$ in Σ_m with $\Sigma_m > 0$, we obtain

$$\Sigma_m \geq \frac{2\delta(m+1)}{L\big[1+\sqrt{1+2\delta\nu m}\big]},$$

which is exactly (2.47).

(b) Let $\gamma_t := \gamma > 0$ for $t = 0, \cdots, m$. Then (2.45) holds if $\nu L^2 \gamma^2 m - \delta + L\gamma = 0$. Solving this quadratic equation in γ and noting that $\gamma > 0$, we obtain $\gamma = \frac{2\delta}{L(\sqrt{1+4\delta\nu m}+1)}$. □

The proof of Lemma 2.2: Properties of stochastic estimators

Proof. We only prove (2.20), since other statements were proved in Harikandeh et al. (2015); Lohr (2009); Nguyen et al. (2017b, 2020). The proof of (2.20) for the finite-sum case (FS-OPT) was also given in Nguyen et al. (2020) but under the L-smoothness of each f_i, we conduct this proof here by following the same path as in Nguyen et al. (2020) for completeness.

Our goal is to prove (2.21) by upper bounding the following quantity:

$$\mathcal{A}_t := \mathbb{E}\left[\|v_t - v_{t-1}\|^2 \mid \mathcal{F}_t\right] - \|\nabla f(w_t) - \nabla f(w_{t-1})\|^2. \tag{2.48}$$

Let $\mathcal{F}_t := \sigma(w_0^{(s)}, \mathcal{B}_1, \cdots, \mathcal{B}_{t-1})$ be the σ-field generated by $w_0^{(s)}$ and mini-batches $\mathcal{B}_1, \cdots, \mathcal{B}_{t-1}$, and $\mathcal{F}_0 = \mathcal{F}_1 = \sigma(w_0^{(s)})$. If we define $\Xi_i := \nabla f_i(w_t) - \nabla f_i(w_{t-1})$, then using the update rule

(2.14), we can upper bound $\mathcal{A}_t$ in (2.48) as

$$\begin{aligned}
\mathcal{A}_t &= \mathbb{E}\left[\|\tfrac{1}{b_t}\sum_{i\in\mathcal{B}_t}\Xi_i\|^2 \mid \mathcal{F}_t\right] - \|\tfrac{1}{n}\sum_{i=1}^n \Xi_i\|^2 \\
&= \tfrac{1}{b_t^2}\mathbb{E}\left[\sum_{i\in\mathcal{B}_t}\sum_{j\in\mathcal{B}_t}\langle\Xi_i,\Xi_j\rangle \mid \mathcal{F}_t\right] - \tfrac{1}{n^2}\sum_{i=1}^n\sum_{j=1}^n\langle\Xi_i,\Xi_j\rangle \\
&= \tfrac{1}{b_t^2}\mathbb{E}\left[\sum_{i,j\in\mathcal{B}_t,i\neq j}\langle\Xi_i,\Xi_j\rangle + \sum_{i\in\mathcal{B}_t}\|\Xi_i\|^2 \mid \mathcal{F}_t\right] - \tfrac{1}{n^2}\sum_{i=1}^n\sum_{j=1}^n\langle\Xi_i,\Xi_j\rangle \\
&= \tfrac{1}{b_t^2}\left[\tfrac{b_t(b_t-1)}{n(n-1)}\sum_{i,j=1,i\neq j}^n\langle\Xi_i,\Xi_j\rangle + \tfrac{b_t}{n}\sum_{i=1}^n\|\Xi_i\|^2\right] - \tfrac{1}{n^2}\sum_{i=1}^n\sum_{j=1}^n\langle\Xi_i,\Xi_j\rangle \\
&= \tfrac{(b_t-1)}{b_tn(n-1)}\sum_{i,j=1}^n\langle\Xi_i,\Xi_j\rangle + \tfrac{(n-b_t)}{b_tn(n-1)}\sum_{i=1}^n\|\Xi_i\|^2 - \tfrac{1}{n^2}\sum_{i=1}^n\sum_{j=1}^n\langle\Xi_i,\Xi_j\rangle \\
&= \tfrac{(n-b_t)}{b_tn(n-1)}\sum_{i=1}^n\|\Xi_i\|^2 - \tfrac{(n-b_t)}{(n-1)b_t}\|\tfrac{1}{n}\sum_{i=1}^n\Xi_i\|^2 \\
&= \tfrac{(n-b_t)}{b_t(n-1)}\tfrac{1}{n}\sum_{i=1}^n\|\nabla f_i(w_t)-\nabla f_i(w_{t-1})\|^2 - \tfrac{(n-b_t)}{(n-1)b_t}\|\nabla f(w_t)-\nabla f(w_{t-1})\|^2,
\end{aligned}$$

where we use the facts that

$$\begin{aligned}
&\mathbb{E}\left[\sum_{i,j\in\mathcal{B}_t,i\neq j}\langle\Xi_i,\Xi_j\rangle \mid \mathcal{F}_t\right] = \tfrac{b_t(b_t-1)}{n(n-1)}\sum_{i,j=1,i\neq j}^n\langle\Xi_i,\Xi_j\rangle \\
\text{and}\quad &\mathbb{E}\left[\sum_{i\in\mathcal{B}_t}\|\Xi_i\|^2 \mid \mathcal{F}_t\right] = \tfrac{b_t}{n}\sum_{i=1}^n\|\Xi_i\|^2
\end{aligned}$$

in the third line of the above derivation. Rearranging the estimate $\mathcal{A}_t$, we obtain (2.20).

To prove (2.21), we define $\Xi_i := \nabla_w\mathbf{f}(w_t;\xi_i) - \nabla_w\mathbf{f}(w_{t-1};\xi_i)$. Clearly, $\mathbb{E}\left[\Xi_i \mid \mathcal{F}_t\right] = \nabla f(w_t) - \nabla f(w_{t-1})$ and $v_t - v_{t-1} = \frac{1}{b_t}\sum_{i\in\mathcal{B}_t}\Xi_i$. Similar to (2.17), we have

$$\mathbb{E}\left[\|(v_t-v_{t-1}) - \mathbb{E}\left[\Xi_i \mid \mathcal{F}_t\right]\|^2 \mid \mathcal{F}_t\right] = \tfrac{1}{b_t}\mathbb{E}\left[\|\Xi_i - \mathbb{E}\left[\Xi_i \mid \mathcal{F}_t\right]\|^2 \mid \mathcal{F}_t\right].$$

Using the fact that $\mathbb{E}\left[\|X - \mathbb{E}\left[X\right]\|^2\right] = \mathbb{E}\left[\|X\|^2\right] - \|\mathbb{E}\left[X\right]\|^2$, after rearranging, we obtain from the last expression that

$$\begin{aligned}
\mathbb{E}\left[\|v_t - v_{t-1}\|^2 \mid \mathcal{F}_t\right] &= \left(1-\tfrac{1}{b_t}\right)\|\nabla f(w_t) - \nabla f(w_{t-1})\|^2 \\
&\quad + \tfrac{1}{b_t}\mathbb{E}\left[\|\nabla_w\mathbf{f}(w_t;\xi) - \nabla_w\mathbf{f}(w_{t-1};\xi)\|^2 \mid \mathcal{F}_t\right],
\end{aligned}$$

which is indeed (2.21). □

2.6.2 The proof of technical results in Section 2.3

We provide the full proof of the results in Section 2.3.

2.6.2.1 The proof of Lemma 2.3: The analysis of the inner loop

Proof. From the update $w_{t+1}^{(s)} := (1-\gamma_t)w_t^{(s)} + \gamma_t \widehat{w}_{t+1}^{(s)}$, we have $w_{t+1}^{(s)} - w_t^{(s)} = \gamma_t(\widehat{w}_{t+1}^{(s)} - w_t^{(s)})$. Firstly, using the L-smoothness of f from (2.4) of Assumption 2.2, we can derive

$$\begin{aligned} f(w_{t+1}^{(s)}) &\leq f(w_t^{(s)}) + \langle \nabla f(w_t^{(s)}), w_{t+1}^{(s)} - w_t^{(s)} \rangle + \tfrac{L}{2}\|w_{t+1}^{(s)} - w_t^{(s)}\|^2 \\ &= f(w_t^{(s)}) + \gamma_t \langle \nabla f(w_t^{(s)}), \widehat{w}_{t+1}^{(s)} - w_t^{(s)} \rangle + \tfrac{L\gamma_t^2}{2}\|\widehat{w}_{t+1}^{(s)} - w_t^{(s)}\|^2. \end{aligned} \tag{2.49}$$

Next, using the convexity of ψ, one can show that

$$\psi(w_{t+1}^{(s)}) \leq (1-\gamma_t)\psi(w_t^{(s)}) + \gamma_t \psi(\widehat{w}_{t+1}^{(s)}) \leq \psi(w_t^{(s)}) + \gamma_t \langle \nabla \psi(\widehat{w}_{t+1}^{(s)}), \widehat{w}_{t+1}^{(s)} - w_t^{(s)} \rangle, \tag{2.50}$$

where $\nabla \psi(\widehat{w}_{t+1}^{(s)}) \in \partial \psi(\widehat{w}_{t+1}^{(s)})$.

By the optimality condition of $\widehat{w}_{t+1}^{(s)} := \mathrm{prox}_{\eta_t \psi}(w_t^{(s)} - \eta_t v_t^{(s)})$, we have $\nabla \psi(\widehat{w}_{t+1}^{(s)}) = -v_t^{(s)} - \frac{1}{\eta_t}(\widehat{w}_{t+1}^{(s)} - w_t^{(s)})$ for some $\nabla \psi(\widehat{w}_{t+1}^{(s)}) \in \partial \psi(\widehat{w}_{t+1}^{(s)})$. Substituting this expression into (2.50) yields

$$\psi(w_{t+1}^{(s)}) \leq \psi(w_t^{(s)}) + \gamma_t \langle v_t^{(s)}, w_t^{(s)} - \widehat{w}_{t+1}^{(s)} \rangle - \frac{\gamma_t}{\eta_t}\|\widehat{w}_{t+1}^{(s)} - w_t^{(s)}\|^2. \tag{2.51}$$

Combining (2.49) and (2.51), and then using $F(w) := f(w) + \psi(w)$ yields

$$F(w_{t+1}^{(s)}) \leq F(w_t^{(s)}) + \gamma_t \langle \nabla f(w_t^{(s)}) - v_t^{(s)}, \widehat{w}_{t+1}^{(s)} - w_t^{(s)} \rangle - \Big(\frac{\gamma_t}{\eta_t} - \frac{L\gamma_t^2}{2}\Big)\|\widehat{w}_{t+1}^{(s)} - w_t^{(s)}\|^2. \tag{2.52}$$

Also, the following expression holds

$$\begin{aligned} \langle \nabla f(w_t^{(s)}) - v_t^{(s)}, \widehat{w}_{t+1}^{(s)} - w_t^{(s)} \rangle &= \tfrac{1}{2}\|\nabla f(w_t^{(s)}) - v_t^{(s)}\|^2 + \tfrac{1}{2}\|\widehat{w}_{t+1}^{(s)} - w_t^{(s)}\|^2 \\ &\quad - \tfrac{1}{2}\|\nabla f(w_t^{(s)}) - v_t^{(s)} - (\widehat{w}_{t+1}^{(s)} - w_t^{(s)})\|^2. \end{aligned}$$

From this expression, we can rewrite (2.52) as

$$F(w_{t+1}^{(s)}) \leq F(w_t^{(s)}) + \frac{\gamma_t}{2}\|\nabla f(w_t^{(s)}) - v_t^{(s)}\|^2 - \Big(\frac{\gamma_t}{\eta_t} - \frac{L\gamma_t^2}{2} - \frac{\gamma_t}{2}\Big)\|\widehat{w}_{t+1}^{(s)} - w_t^{(s)}\|^2 - \sigma_t^{(s)},$$

where $\sigma_t^{(s)} := \frac{\gamma_t}{2}\|\nabla f(w_t^{(s)}) - v_t^{(s)} - (\widehat{w}_{t+1}^{(s)} - w_t^{(s)})\|^2 \geq 0$.

Taking expectation both sides of this inequality over the entire history, we obtain

$$\begin{aligned}\mathbb{E}\left[F(w_{t+1}^{(s)})\right] &\leq \mathbb{E}\left[F(w_t^{(s)})\right] + \tfrac{\gamma_t}{2}\mathbb{E}\left[\|\nabla f(w_t^{(s)}) - v_t^{(s)}\|^2\right] \\ &\quad - \left(\tfrac{\gamma_t}{\eta_t} - \tfrac{L\gamma_t^2}{2} - \tfrac{\gamma_t}{2}\right)\mathbb{E}\left[\|\widehat{w}_{t+1}^{(s)} - w_t^{(s)}\|^2\right] - \mathbb{E}\left[\sigma_t^{(s)}\right].\end{aligned} \tag{2.53}$$

Next, recall from (1.5) that $G_\eta(w) := \frac{1}{\eta}\big(w - \mathrm{prox}_{\eta\psi}(w - \eta\nabla f(w))\big)$ is the gradient mapping of F. In this case, it is obvious that

$$\eta_t\|G_{\eta_t}(w_t^{(s)})\| = \|w_t^{(s)} - \mathrm{prox}_{\eta_t\psi}(w_t^{(s)} - \eta_t\nabla f(w_t^{(s)}))\|.$$

Using this definition, the triangle inequality, and the nonexpansive property $\|\mathrm{prox}_{\eta\psi}(z) - \mathrm{prox}_{\eta\psi}(w)\| \leq \|z - w\|$ of $\mathrm{prox}_{\eta\psi}$, we can derive that

$$\begin{aligned}\eta_t\|G_{\eta_t}(w_t^{(s)})\| &\leq \|\widehat{w}_{t+1}^{(s)} - w_t^{(s)}\| + \|\mathrm{prox}_{\eta_t\psi}(w_t^{(s)} - \eta_t\nabla f(w_t^{(s)})) - \widehat{w}_{t+1}^{(s)}\| \\ &= \|\widehat{w}_{t+1}^{(s)} - w_t^{(s)}\| + \|\mathrm{prox}_{\eta_t\psi}(w_t^{(s)} - \eta_t\nabla f(w_t^{(s)})) - \mathrm{prox}_{\eta_t\psi}(w_t^{(s)} - \eta_t v_t^{(s)})\| \\ &\leq \|\widehat{w}_{t+1}^{(s)} - w_t^{(s)}\| + \eta_t\|\nabla f(w_t^{(s)}) - v_t^{(s)}\|.\end{aligned}$$

Now, the last estimate leads to

$$\eta_t^2\mathbb{E}\left[\|G_{\eta_t}(w_t^{(s)})\|^2\right] \leq 2\mathbb{E}\left[\|\widehat{w}_{t+1}^{(s)} - w_t^{(s)}\|^2\right] + 2\eta_t^2\mathbb{E}\left[\|\nabla f(w_t^{(s)}) - v_t^{(s)}\|^2\right].$$

Multiplying this inequality by $\frac{\gamma_t}{2} > 0$ and adding the result to (2.53), we finally get

$$\begin{aligned}\mathbb{E}\left[F(w_{t+1}^{(s)})\right] &\leq \mathbb{E}\left[F(w_t^{(s)})\right] - \tfrac{\gamma_t\eta_t^2}{2}\mathbb{E}\left[\|G_{\eta_t}(w_t^{(s)})\|^2\right] \\ &\quad + \tfrac{\gamma_t}{2}\left(1 + 2\eta_t^2\right)\mathbb{E}\left[\|\nabla f(w_t^{(s)}) - v_t^{(s)}\|^2\right] \\ &\quad - \tfrac{\gamma_t}{2}\left(\tfrac{2}{\eta_t} - L\gamma_t - 3\right)\mathbb{E}\left[\|\widehat{w}_{t+1}^{(s)} - w_t^{(s)}\|^2\right] \\ &\quad - \mathbb{E}\left[\sigma_t^{(s)}\right].\end{aligned}$$

Summing up this inequality from $t = 0$ to $t = m$, we obtain

$$\begin{aligned}\mathbb{E}\left[F(w_{m+1}^{(s)})\right] \;\leq\; & \mathbb{E}\left[F(w_0^{(s)})\right] + \frac{1}{2}\sum_{t=0}^{m}\gamma_t\left(1+2\eta_t^2\right)\mathbb{E}\left[\|\nabla f(w_t^{(s)}) - v_t^{(s)}\|^2\right] \\ & - \frac{1}{2}\sum_{t=0}^{m}\gamma_t\Big(\frac{2}{\eta_t} - L\gamma_t - 3\Big)\mathbb{E}\left[\|\widehat{w}_{t+1}^{(s)} - w_t^{(s)}\|^2\right] \\ & - \sum_{t=0}^{m}\frac{\gamma_t\eta_t^2}{2}\mathbb{E}\left[\|G_{\eta_t}(w_t^{(s)})\|^2\right] - \sum_{t=0}^{m}\mathbb{E}\left[\sigma_t^{(s)}\right].\end{aligned} \tag{2.54}$$

We consider two cases:

Case 1: In the finite-sum setting (FS-OPT), i.e., Algorithm 1 solves (FS-OPT), then from (2.20) of Lemma 2.2, the L-smoothness condition (2.2) in Assumption 2.2, the choice $\hat{b}_t^{(s)} = \hat{b} \geq 1$, and $w_j^{(s)} - w_{j-1}^{(s)} = \gamma_{j-1}(\widehat{w}_j^{(s)} - w_{j-1}^{(s)})$, we can estimate

$$\begin{aligned}\mathbb{E}\left[\|v_j^{(s)} - v_{j-1}^{(s)}\|^2 \mid \mathcal{F}_j\right] \overset{(2.20)}{=} & \;\tfrac{n(\hat{b}-1)}{\hat{b}(n-1)}\|\nabla f(w_j) - \nabla f(w_{j-1})\|^2 \\ & + \tfrac{n-\hat{b}}{\hat{b}(n-1)}\tfrac{1}{n}\textstyle\sum_{i=1}^{n}\|\nabla f_i(w_j^{(s)}) - \nabla f_i(w_{j-1}^{(s)})\|^2.\end{aligned}$$

Using (2.2), we can further bound

$$\begin{aligned}\mathbb{E}\left[\|v_j^{(s)} - v_{j-1}^{(s)}\|^2 \mid \mathcal{F}_j\right] \overset{(2.2)}{\leq} & \;\|\nabla f(w_j) - \nabla f(w_{j-1})\|^2 + \tfrac{(n-\hat{b})L^2}{\hat{b}(n-1)}\|w_j^{(s)} - w_{j-1}^{(s)}\|^2 \\ = & \;\|\nabla f(w_j) - \nabla f(w_{j-1})\|^2 + \tfrac{(n-\hat{b})L^2\gamma_{j-1}^2}{\hat{b}(n-1)}\|\widehat{w}_j^{(s)} - w_{j-1}^{(s)}\|^2.\end{aligned}$$

where the equality comes from the fact that $w_j^{(s)} - w_{j-1}^{(s)} = \gamma_{j-1}(\widehat{w}_j^{(s)} - w_{j-1}^{(s)})$.

Case 2: In the expectation setting (St-OPT), i.e., Algorithm 1 solves (St-OPT), then from (2.21) of Lemma 2.2, we have

$$\begin{aligned}\mathbb{E}\left[\|v_j^{(s)} - v_{j-1}^{(s)}\|^2 \mid \mathcal{F}_j\right] \overset{(2.21)}{=} & \;\left(1 - \tfrac{1}{\hat{b}}\right)\|\nabla f(w_j) - \nabla f(w_{j-1})\|^2 \\ & + \tfrac{1}{\hat{b}}\mathbb{E}\left[\|\nabla_w \mathbf{f}(w_j;\xi) - \nabla_w \mathbf{f}(w_{j-1};\xi)\|^2 \mid \mathcal{F}_j\right] \\ \overset{(2.1)}{\leq} & \;\|\nabla f(w_j) - \nabla f(w_{j-1})\|^2 + \tfrac{L^2}{\hat{b}}\|w_j^{(s)} - w_{j-1}^{(s)}\|^2 \\ = & \;\|\nabla f(w_j) - \nabla f(w_{j-1})\|^2 + \tfrac{L^2\gamma_{j-1}^2}{\hat{b}}\|\widehat{w}_j^{(s)} - w_{j-1}^{(s)}\|^2.\end{aligned}$$

Using either one of the two last inequalities and (2.16), then taking the full expectation yields

$$\begin{aligned}\mathbb{E}\left[\|\nabla f(w_t^{(s)}) - v_t^{(s)}\|^2\right] &= \mathbb{E}\left[\|\nabla f(w_0^{(s)}) - v_0^{(s)}\|^2\right] \textstyle\sum_{j=1}^t \mathbb{E}\left[\|v_j^{(s)} - v_{j-1}^{(s)}\|^2\right] \\ &\quad - \textstyle\sum_{j=1}^t \mathbb{E}\left[\|\nabla f(w_j) - \nabla f(w_{j-1})\|^2\right] \\ &\leq \mathbb{E}\left[\|\nabla f(w_0^{(s)}) - v_0^{(s)}\|^2\right] + \rho L^2 \textstyle\sum_{j=1}^t \gamma_{j-1}^2 \mathbb{E}\left[\|\widehat{w}_j^{(s)} - w_{j-1}^{(s)}\|^2\right] \\ &= \bar{\sigma}^{(s)} + \rho L^2 \textstyle\sum_{j=1}^t \gamma_{j-1}^2 \mathbb{E}\left[\|\widehat{w}_j^{(s)} - w_{j-1}^{(s)}\|^2\right],\end{aligned} \tag{2.55}$$

where $\bar{\sigma}^{(s)} := \mathbb{E}\left[\|\nabla f(w_0^{(s)}) - v_0^{(s)}\|^2\right] \geq 0$, and $\rho := \frac{1}{\hat{b}}$ if Algorithm 1 solves (St-OPT), and $\rho := \frac{n-\hat{b}}{\hat{b}(n-1)}$ if Algorithm 1 solves (FS-OPT).

Substituting (2.55) into (2.54) and dropping the term $-\sum_{t=0}^m \mathbb{E}\left[\sigma_t^{(s)}\right]$ (≤ 0), we finally arrive at

$$\begin{aligned}\mathbb{E}\left[F(w_{m+1}^{(s)})\right] \;\leq\; & \mathbb{E}\left[F(w_0^{(s)})\right] + \tfrac{\rho L^2}{2} \sum_{t=0}^m \gamma_t \left(1 + 2\eta_t^2\right) \sum_{j=1}^t \gamma_{j-1}^2 \mathbb{E}\left[\|\widehat{w}_j^{(s)} - w_{j-1}^{(s)}\|^2\right] \\ & - \frac{1}{2} \sum_{t=0}^m \gamma_t \left(\frac{2}{\eta_t} - L\gamma_t - 3\right) \mathbb{E}\left[\|\widehat{w}_{t+1}^{(s)} - w_t^{(s)}\|^2\right] \\ & - \sum_{t=0}^m \frac{\gamma_t \eta_t^2}{2} \mathbb{E}\left[\|G_{\eta_t}(w_t^{(s)})\|^2\right] + \frac{1}{2} \sum_{t=0}^m \gamma_t \left(1 + 2\eta_t^2\right) \bar{\sigma}^{(s)},\end{aligned}$$

which is exactly (2.23). □

2.6.2.2 The Proof of lemma 2.4: The selection of constant step-sizes

Proof. Let us first fix all the step-sizes of Algorithm 1 as constants as follows:

$$\gamma_t := \gamma \in (0, 1] \quad \text{and} \quad \eta_t := \eta > 0.$$

We also denote $a_t^{(s)} := \mathbb{E}\left[\|\widehat{w}_{t+1}^{(s)} - w_t^{(s)}\|^2\right] \geq 0$.

Let $\rho := \frac{1}{\hat{b}}$ if Algorithm 1 solves (St-OPT) and $\rho := \frac{n-\hat{b}}{\hat{b}(n-1)}$ if Algorithm 1 solves (FS-OPT). Using these expressions into (2.23), we can easily show that

$$\begin{aligned}
\mathbb{E}\left[F(w_{m+1}^{(s)})\right] &\leq \mathbb{E}\left[F(w_0^{(s)})\right] + \tfrac{\rho L^2\gamma^3}{2}\left(1+2\eta^2\right)\sum_{t=0}^{m}\sum_{j=1}^{t} a_{j-1}^{(s)} \\
&\quad - \tfrac{\gamma}{2}\left(\tfrac{2}{\eta} - L\gamma - 3\right)\sum_{t=0}^{m} a_t^{(s)} - \tfrac{\gamma\eta^2}{2}\sum_{t=0}^{m}\mathbb{E}\left[\|G_{\eta_t}(w_t^{(s)})\|^2\right] \\
&\quad + \tfrac{\gamma}{2}\left(1+2\eta^2\right)(m+1)\bar{\sigma}^{(s)} \\
&= \mathbb{E}\left[F(w_0^{(s)})\right] - \tfrac{\gamma\eta^2}{2}\sum_{t=0}^{m}\mathbb{E}\left[\|G_{\eta_t}(w_t^{(s)})\|^2\right] \\
&\quad + \tfrac{\gamma}{2}\left(1+2\eta^2\right)(m+1)\bar{\sigma}^{(s)} + \mathcal{T}_m,
\end{aligned} \tag{2.56}$$

where $\mathcal{T}_m$ is defined as

$$\mathcal{T}_m := \frac{\rho L^2\gamma^3\left(1+2\eta^2\right)}{2}\sum_{t=0}^{m}\sum_{j=1}^{t} a_{j-1}^{(s)} - \frac{\gamma}{2}\left(\frac{2}{\eta} - L\gamma - 3\right)\sum_{t=0}^{m} a_t^{(s)}.$$

Our goal is to choose $\eta > 0$, and $\gamma \in (0, 1]$ such that $\mathcal{T}_m \leq 0$. We first rewrite $\mathcal{T}_m$ as follows:

$$\begin{aligned}
\mathcal{T}_m &= \tfrac{\rho L^2\gamma^3\left(1+2\eta^2\right)}{2}\left[m a_0^{(s)} + (m-1)a_1^{(s)} + \cdots + 2a_{m-2}^{(s)} + a_{m-1}^{(s)}\right] \\
&\quad - \tfrac{\gamma}{2}\left(\tfrac{2}{\eta} - L\gamma - 3\right)\left[a_0^{(s)} + a_1^{(s)} + \cdots + a_m^{(s)}\right].
\end{aligned}$$

By synchronizing the coefficients of the terms $a_0^{(s)}, a_1^{(s)}, \cdots, a_m^{(s)}$, to guarantee $\mathcal{T}_m \leq 0$, we need to satisfy

$$\begin{cases}
\rho\left(1+2\eta^2\right)L^2\gamma^2 m - \left(\frac{2}{\eta} - L\gamma - 3\right) & \leq 0, \\
\frac{2}{\eta} - L\gamma - 3 & \geq 0.
\end{cases} \tag{2.57}$$

Assume that $\frac{2}{\eta} - L\gamma - 3 = 1 > 0$. This implies that $\eta = \frac{2}{L\gamma+4}$. Next, since $L\gamma > 0$, we have $\eta \leq \frac{1}{2}$. Therefore, we can upper bound

$$\rho L^2\gamma^2 m(1+2\eta^2) - \left(\frac{2}{\eta} - L\gamma - 3\right) \leq \frac{3\rho L^2\gamma^2 m}{2} - 1 = 0.$$

The last equation and $\eta = \frac{2}{L\gamma+4}$ lead to

$$\gamma := \frac{1}{L\sqrt{\omega m}} \quad \text{and} \quad \eta := \frac{2\sqrt{\omega m}}{4\sqrt{\omega m}+1},$$

which is exactly (2.24), where $\omega := \frac{3(n-\hat{b})}{2\hat{b}(n-1)}$ for (FS-OPT) and $\omega := \frac{3}{2\hat{b}}$ for (St-OPT).

Finally, using this choice (2.24) of the step-sizes, we can derive that

$$\mathbb{E}\left[F(w_{m+1}^{(s)}\right] \leq \mathbb{E}\left[F(w_0^{(s)})\right] - \frac{\gamma\eta^2}{2}\sum_{t=0}^{m}\mathbb{E}\left[\|G_\eta(w_t^{(s)})\|^2\right] + \frac{\gamma\theta}{2}(m+1)\bar{\sigma}^{(s)}, \tag{2.58}$$

which is exactly (2.25), where $\theta := 1 + 2\eta^2 \leq \frac{3}{2}$. □

2.6.2.3 The proof of Theorem 2.1: The dynamic step-size case

Proof. Let $\beta_t := \gamma_t\left(1+2\eta_t^2\right)$ and $\kappa_t := \gamma_t\left(\frac{2}{\eta_t} - L\gamma_t - 3\right)$. From (2.23) of Lemma 2.3 we have

$$\mathbb{E}\left[F(w_{m+1}^{(s)})\right] \leq \mathbb{E}\left[F(w_0^{(s)})\right] - \sum_{t=0}^{m}\frac{\gamma_t\eta_t^2}{2}\mathbb{E}\left[\|G_{\eta_t}(w_t^{(s)})\|^2\right] + \frac{1}{2}\bar{\sigma}^{(s)}\Big(\sum_{t=0}^{m}\beta_t\Big) + \mathcal{T}_m, \tag{2.59}$$

where we define

$$\mathcal{T}_m := \frac{L^2(n-\hat{b})}{2\hat{b}(n-1)}\sum_{t=0}^{m}\beta_t\sum_{j=1}^{t}\gamma_{j-1}^2\mathbb{E}\left[\|\widehat{w}_j^{(s)} - w_{j-1}^{(s)}\|^2\right] - \frac{1}{2}\sum_{t=0}^{m}\kappa_t\mathbb{E}\left[\|\widehat{w}_{t+1}^{(s)} - w_t^{(s)}\|^2\right].$$

Now, to guarantee $\mathcal{T}_m \leq 0$, let us choose all the parameters such that

$$\begin{cases} \kappa_m & = 0, \\ \frac{(n-\hat{b})}{\hat{b}(n-1)}L^2\gamma_t^2\sum_{j=t+1}^{m}\beta_j - \kappa_t & = 0, \quad t = 0, \cdots, m-1. \end{cases} \tag{2.60}$$

Then, (2.59) becomes

$$\mathbb{E}\left[F(w_{m+1}^{(s)})\right] \leq \mathbb{E}\left[F(w_0^{(s)})\right] - \sum_{t=0}^{m}\frac{s_t\eta_t^2}{2}\mathbb{E}\left[\|G_{\eta_t}(w_t^{(s)})\|^2\right] + \frac{1}{2}\sum_{t=0}^{m}\beta_t\bar{\sigma}^{(s)}. \tag{2.61}$$

If we fix $\eta_t = \eta \in (0, \frac{2}{3})$, and define $\delta := \frac{2}{\eta} - 3 > 0$, then (2.60) reduces to

$$\begin{cases} \delta - L\gamma_m & = 0, \\ \frac{L^2(n-\hat{b})(1+2\eta^2)}{\hat{b}(n-1)}\gamma_t \sum_{j=t+1}^{m} \gamma_j - \delta + L\gamma_t & = 0, \quad t = 0, \cdots, m-1. \end{cases} \tag{2.62}$$

Applying Lemma 2.5(a) with $\nu = \omega_\eta := \frac{(n-\hat{b})(1+2\eta^2)}{\hat{b}(n-1)}$, we obtain from (2.62) that

$$\gamma_m := \frac{\delta}{L}, \quad \text{and} \quad \gamma_t := \frac{\delta}{L\big[1 + \omega_\eta L \sum_{j=t+1}^{m} \gamma_j\big]}, \quad t = 0, \cdots, m-1. \tag{2.63}$$

Moreover, we have

$$\frac{\delta}{L(1+\omega_\eta \delta m)} < \gamma_0 < \gamma_1 < \cdots < \gamma_m, \quad \text{and} \quad \Sigma_m := \sum_{t=0}^{m} \gamma_t \geq \frac{2\delta(m+1)}{L(\sqrt{2\omega_\eta \delta m + 1} + 1)},$$

which proves (2.28).

On the other hand, since $\bar{\sigma}^{(s)} := \mathbb{E}\left[\|\nabla f(w_0^{(s)}) - v_0^{(s)}\|^2\right] = \mathbb{E}\left[\|\widetilde{\nabla} f_{\mathcal{B}_s}(\widetilde{w}_{s-1}) - \nabla f(\widetilde{w}_{s-1})\|^2\right]$, by using (2.18), we have $\bar{\sigma}^{(s)} \leq \left(\frac{n-b_s}{nb_s}\right)\sigma_n^2(\widetilde{w}_{s-1})$. Using this upper bound and $\beta_t := \gamma_t(1+2\eta^2) \leq \frac{3\gamma_t}{2} \leq \frac{3}{2}$ (since $\gamma_t \in [0,1]$), into the estimate (2.61), we can arrive at

$$\frac{1}{S\Sigma_m}\sum_{s=1}^{S}\sum_{t=0}^{m}\gamma_t \mathbb{E}\left[\|G_\eta(w_t^{(s)})\|^2\right] \leq \frac{2}{\eta^2 S\Sigma_m}\big[F(\widetilde{w}_0) - F^\star\big] + \frac{3}{2\eta^2 S}\sum_{s=1}^{S}\frac{(n-b_s)\sigma_n^2(\widetilde{w}_{s-1})}{nb_s},$$

which is exactly (2.29). Now, let us choose $\eta := \frac{1}{2} \in (0, \frac{2}{3})$. Then, we have $\delta = 1$, $\omega_\eta = \frac{3(n-\hat{b})}{2\hat{b}(n-1)}$, and $\Sigma_m \geq \frac{2\delta(m+1)}{L(\sqrt{2\omega_\eta m+1}+1)}$.

Using these facts, $\widetilde{w}_T \sim \mathbf{U}_p\big(\{w_t^{(s)}\}_{t=0\to m}^{s=1\to S}\big)$ with $\mathbf{Prob}\left(\widetilde{w}_T = w_t^{(s)}\right) = p_{(s-1)m+t} := \frac{\gamma_t}{S\Sigma_m}$, and $b_s = n$, we obtain from (2.29) that

$$\mathbb{E}\left[\|G_\eta(\widetilde{w}_T)\|^2\right] = \frac{1}{S\Sigma_m}\sum_{s=1}^{S}\sum_{t=0}^{m}\gamma_t\mathbb{E}\left[\|G_\eta(w_t^{(s)})\|^2\right] \leq \frac{4L(\sqrt{2\omega m+1}+1)}{S(m+1)}\big[F(\widetilde{w}_0) - F^\star\big].$$

Next, using $m = \lfloor \frac{n}{\hat{b}} \rfloor$ and $\omega := \omega_\eta = \frac{3(n-\hat{b})}{2\hat{b}(n-1)}$, if $\hat{b} \leq \sqrt{n}$, then we can bound

$$\frac{\sqrt{2\omega m+1}+1}{m+1} \leq \frac{2\sqrt{\omega}}{\sqrt{m+1}} \leq \frac{\sqrt{6}}{\sqrt{n}}.$$

Using this bound, we can further bound the above estimate obtained from (2.29) as

$$\mathbb{E}\left[\|G_\eta(\widetilde{w}_T)\|^2\right] \leq \frac{4\sqrt{6}L\left[F(\widetilde{w}_0) - F^\star\right]}{S\sqrt{n}},$$

which is (2.30).

To achieve $\mathbb{E}\left[\|G_\eta(\widetilde{w}_T)\|^2\right] \leq \varepsilon^2$, we impose $\frac{4\sqrt{6}L[F(\widetilde{w}_0)-F^\star]}{S\sqrt{n}} = \varepsilon^2$, which shows that the number of outer iterations $S := \frac{4\sqrt{6}L[F(\widetilde{w}_0)-F^\star]}{\sqrt{n}\varepsilon^2}$. To guarantee $S \geq 1$, we need $n \leq \frac{96L^2[F(\widetilde{w}_0)-F^\star]^2}{\varepsilon^4}$.

Hence, we can estimate the number of gradient evaluations $\mathcal{T}_{\rm grad}$ by

$$\mathcal{T}_{\rm grad} = Sn + 2S(m+1)\hat{b} \leq 5Sn = \frac{20\sqrt{6}L\sqrt{n}\left[F(\widetilde{w}_0) - F^\star\right]}{\varepsilon^2}.$$

We can conclude that the number of stochastic gradient evaluations does not exceed $\mathcal{T}_{\rm grad} = \mathcal{O}\left(\frac{L\sqrt{n}[F(\widetilde{w}_0)-F^\star]}{\varepsilon^2}\right)$. The number of proximal operations $\mathrm{prox}_{\eta\psi}$ does not exceed $\mathcal{T}_{\rm prox} := S(m+1) \leq \frac{4\sqrt{6}(\sqrt{n}+1)L[F(\widetilde{w}_0)-F^\star]}{\hat{b}\varepsilon^2}$. □

2.6.2.4 The proof of Theorem 2.2: The constant step-size case

Proof. If we choose $(\gamma_t, \eta_t) = (\gamma, \eta) > 0$ for all $t = 0, \cdots, m$, then, by applying Lemma 2.4, we can update

$$\gamma := \frac{1}{L\sqrt{\omega m}} \quad \text{and} \quad \eta := \frac{2\sqrt{\omega m}}{4\sqrt{\omega m}+1},$$

which is exactly (2.31), where $\omega := \frac{3(n-\hat{b})}{2(n-1)\hat{b}}$. With this update, we can simplify (2.25) as

$$\mathbb{E}\left[F(w_{m+1}^{(s)})\right] \leq \mathbb{E}\left[F(w_0^{(s)})\right] - \frac{\gamma\eta^2}{2}\sum_{t=0}^{m}\mathbb{E}\left[\|G_\eta(w_t^{(s)})\|^2\right] + \frac{3\gamma}{4}(m+1)\bar{\sigma}^{(s)}.$$

With the same argument as above, we obtain

$$\frac{1}{(m+1)S}\sum_{s=1}^{S}\sum_{t=0}^{m}\mathbb{E}\left[\|G_\eta(w_t^{(s)})\|^2\right] \leq \frac{2}{\gamma\eta^2(m+1)S}\big[F(\widetilde{w}_0) - F^\star\big] + \frac{3}{2\eta^2 S}\sum_{s=1}^{S}\frac{(n-b_s)\sigma_n^2(\widetilde{w}_{s-1})}{nb_s}.$$

For $\widetilde{w}_T \sim \mathbf{U}\big(\{w_t^{(s)}\}_{t=0\to m}^{s=1\to S}\big)$ with $T := (m+1)S$ and $b_s = n$, the last estimate implies

$$\mathbb{E}\left[\|G_\eta(\widetilde{w}_T)\|^2\right] = \frac{1}{(m+1)S}\sum_{s=1}^{S}\sum_{t=0}^{m}\mathbb{E}\left[\|G_\eta(w_t^{(s)})\|^2\right] \leq \frac{2}{\gamma\eta^2(m+1)S}\big[F(\widetilde{w}_0) - F^\star\big].$$

By the update rule of η and γ, we can easily show that $\gamma\eta^2 \geq \frac{4\sqrt{\omega m}}{L(4\sqrt{\omega m}+1)^2}$. Therefore, using $m := \lfloor \frac{n}{\hat{b}} \rfloor$, we can overestimate

$$\frac{1}{\gamma\eta^2(m+1)} \leq \frac{L(4\sqrt{\omega m}+1)^2}{4\sqrt{\omega m}(m+1)} \leq \frac{8L\sqrt{\omega}}{\sqrt{m}} \leq \frac{8\sqrt{3}L}{\sqrt{2n}}.$$

Using this upper bound, to guarantee $\mathbb{E}\left[\|G_\eta(\widetilde{w}_T)\|^2\right] \leq \varepsilon^2$, we choose S and m such that $\frac{16\sqrt{3}L}{S\sqrt{2n}}\left[F(\widetilde{w}_0) - F^\star\right] = \varepsilon^2$, which leads to $S := \frac{16\sqrt{3}L}{\sqrt{2n}\varepsilon^2}\left[F(\widetilde{w}_0) - F^\star\right]$ as the number of outer iterations. To guarantee $S \geq 1$, we need to choose $n \leq \frac{384L^2}{\varepsilon^4}\left[F(\widetilde{w}_0) - F^\star\right]^2$.

Finally, we can estimate the number of stochastic gradient evaluations $\mathcal{T}_{\rm grad}$ as

$$\mathcal{T}_{\rm grad} = Sn + 2S(m+1) \leq 5Sn = \frac{16\sqrt{3}L\sqrt{n}}{\sqrt{2}\varepsilon^2}\left[F(\widetilde{w}_0) - F^\star\right] = \mathcal{O}\left(\frac{L\sqrt{n}}{\varepsilon^2}\left[F(\widetilde{w}_0) - F^\star\right]\right).$$

The number of $\mathrm{prox}_{\eta\psi}$ is $\mathcal{T}_{\rm prox} = S(m+1) \leq \frac{16\sqrt{3}L(\sqrt{n}+1)}{\hat{b}\sqrt{2}\varepsilon^2}\left[F(\widetilde{w}_0) - F^\star\right]$. □

2.6.2.5 The proof of Theorem 2.3: The expectation problem

Proof. Summing up (2.25) from $s = 1$ to $s = S$, and then using $w_0^{(0)} = \widetilde{w}_0$, we obtain

$$\frac{\gamma\eta^2}{2}\sum_{s=1}^{S}\sum_{t=0}^{m}\mathbb{E}\left[\|G_\eta(w_t^{(s)})\|^2\right] \leq F(\widetilde{w}_0) - \mathbb{E}\left[F(w_{m+1}^{(S)})\right] + \frac{\gamma\theta(m+1)}{2}\sum_{s=1}^{S}\bar{\sigma}^{(s)}. \tag{2.64}$$

Note that $\mathbb{E}\left[F(w_{m+1}^{(S)})\right] \geq F^\star$ by Assumption 2.1. Moreover, by (2.17), we have

$$\bar{\sigma}^{(s)} := \mathbb{E}\left[\|v_0^{(s)} - \nabla f(w_0^{(s)})\|^2\right] = \mathbb{E}\left[\|\widetilde{\nabla} f_{\mathcal{B}_s}(w_0^{(s)}) - \nabla f(w_0^{(s)})\|^2\right] \leq \frac{\sigma^2}{b_s} = \frac{\sigma^2}{b}.$$

Recall that $\rho := \frac{1}{\hat{b}}$ for (St-OPT). Therefore, we have $\theta = 1 + \frac{8\bar{\omega}m}{(1+4\sqrt{\bar{\omega}m})^2} < \frac{3}{2}$, where $\bar{\omega} := \frac{3}{2\hat{b}}$. Using these estimates into (2.64), we obtain (2.36).

Now, since $\widetilde{w}_T \sim \mathbf{U}\big(\{w_t^{(s)}\}_{t=0\to m}^{s=1\to S}\big)$ for $T := S(m+1)$, we have

$$\begin{aligned}\mathbb{E}\left[\|G_\eta(\widetilde{w}_T)\|^2\right] &= \frac{1}{(m+1)S}\sum_{s=1}^{S}\sum_{t=0}^{m}\mathbb{E}\left[\|G_\eta(w_t^{(s)})\|^2\right]\\ &\leq \frac{2}{\gamma\eta^2(m+1)S}[F(\widetilde{w}_0) - F^\star] + \frac{3\sigma^2}{2\eta^2 b}.\end{aligned}$$

Since $\eta = \frac{2\sqrt{\bar{\omega}m}}{4\sqrt{\bar{\omega}m}+1} \geq \frac{2}{5}$ and $\frac{1}{\gamma\eta^2(m+1)} \leq \frac{25L\sqrt{\bar{\omega}m}}{4(m+1)} \leq \frac{8L}{\sqrt{\hat{b}m}}$ as proved above, to guarantee $\mathbb{E}\left[\|G_\eta(\widetilde{w}_T)\|^2\right] \leq \varepsilon^2$, we need to set

$$\frac{16L}{S\sqrt{\hat{b}m}}[F(\widetilde{w}_0) - F^\star] + \frac{75\sigma^2}{8b} = \varepsilon^2.$$

Let us choose b such that $\frac{75\sigma^2}{8b} = \frac{\varepsilon^2}{2}$, which leads to $b := \frac{75\sigma^2}{8\varepsilon^2}$. We also choose $m := \frac{\sigma^2}{\hat{b}\varepsilon^2}$. To guarantee $m \geq 1$, we have $\hat{b} \leq \frac{\sigma^2}{\varepsilon^2}$. Then, since $\frac{1}{\sqrt{\hat{b}m}} = \frac{\varepsilon}{\sigma}$, the above condition is equivalent to $\frac{16L\varepsilon}{S\sigma}[F(\widetilde{w}_0) - F^\star] = \frac{\varepsilon^2}{2}$, which leads to

$$S := \frac{32L}{\sigma\varepsilon}[F(\widetilde{w}_0) - F^\star].$$

To guarantee $S \geq 1$, we need to choose $\varepsilon \leq \frac{32L}{\sigma}[F(\widetilde{w}_0) - F^\star]$ if σ is sufficiently large.

Now, we estimate the total number of stochastic gradient evaluations as

$$\begin{aligned}\mathcal{T}_{\text{grad}} &= \textstyle\sum_{s=1}^S b_s + 2m\hat{b}S = (b + 2m\hat{b})S = \frac{32L}{\sigma\varepsilon}[F(\widetilde{w}_0) - F^\star]\left(\frac{75\sigma^2}{\varepsilon^2} + \frac{2\sigma^2}{\hat{b}\varepsilon^2}\hat{b}\right)\\ &= \frac{2464L\sigma}{\varepsilon^3}[F(\widetilde{w}_0) - F^\star].\end{aligned}$$

Hence, the number of gradient evaluations is $\mathcal{O}\left(\frac{L\sigma[F(\widetilde{w}_0)-F^\star]}{\varepsilon^3}\right)$, and the number of proximal operator calls is also $\mathcal{T}_{\text{prox}} := S(m+1) = \frac{32\sigma L}{\hat{b}\varepsilon^2}[F(\widetilde{w}_0) - F^\star]$. □

2.6.2.6 The proof of Theorem 2.4: The non-composite cases

Proof. Since $\psi = 0$, we have $\widehat{w}_{t+1}^{(s)} = w_t^{(s)} - \eta_t v_t^{(s)}$. Therefore, $\widehat{w}_{t+1}^{(s)} - w_t^{(s)} = -\eta_t v_t^{(s)}$ and $w_{t+1}^{(s)} = (1-\gamma_t)w_t^{(s)} + \gamma_t\widehat{w}_{t+1}^{(s)} = w_t^{(s)} - \gamma_t\eta_t v_t^{(s)} = w_t^{(s)} - \hat{\eta}_t v_t^{(s)}$, where $\hat{\eta}_t := \gamma_t\eta_t$. Using these relations and choose $c_t = \frac{1}{\eta_t}$, we can easily show that

$$\left\{\begin{array}{l} \mathbb{E}\left[\|\widehat{w}_{t+1}^{(s)} - w_t^{(s)}\|^2\right] = \eta_t^2\mathbb{E}\left[\|v_t^{(s)}\|^2\right],\\ \sigma_t^{(s)} := \frac{\gamma_t}{2c_t}\|\nabla f(w_t^{(s)}) - v_t^{(s)} - c_t(\widehat{w}_{t+1}^{(s)} - w_t^{(s)})\|^2 = \frac{\hat{\eta}_t}{2}\|\nabla f(w_t^{(s)})\|^2.\end{array}\right.$$

Substituting these estimates into (2.53) and noting that $f = F$ and $\hat{\eta}_t := \gamma_t\eta_t$, we obtain

$$\begin{aligned}\mathbb{E}\left[f(w_{t+1}^{(s)})\right] &\leq \mathbb{E}\left[f(w_t^{(s)})\right] + \tfrac{\hat{\eta}_t}{2}\mathbb{E}\left[\|\nabla f(w_t^{(s)}) - v_t^{(s)}\|^2\right] \\ &\quad - \tfrac{\hat{\eta}_t}{2}\big(1 - L\hat{\eta}_t\big)\mathbb{E}\left[\|v_t^{(s)}\|^2\right] - \tfrac{\hat{\eta}_t}{2}\mathbb{E}\left[\|\nabla f(w_t^{(s)})\|^2\right].\end{aligned} \tag{2.65}$$

On the other hand, from (2.16), by Assumption 2.2, (2.13), and $w_{t+1}^{(s)} := w_t^{(s)} - \hat{\eta}_t v_t^{(s)}$, we can derive

$$\begin{aligned}\mathbb{E}\left[\|\nabla f(w_t^{(s)}) - v_t^{(s)}\|^2\right] &\leq \mathbb{E}\left[\|\nabla f(w_0^{(s)}) - v_0^{(s)}\|^2\right] + \textstyle\sum_{j=1}^t \mathbb{E}\left[\|v_j^{(s)} - v_{j-1}^{(s)}\|^2\right] \\ &\leq \mathbb{E}\left[\|\nabla f(w_0^{(s)}) - v_0^{(s)}\|^2\right] \\ &\quad + \rho\textstyle\sum_{j=1}^t \mathbb{E}\left[\|\nabla_w \mathbf{f}(w_j^{(s)};\xi_j^{(s)}) - \nabla_w \mathbf{f}(w_{j-1}^{(s)};\xi_j^{(s)})\|^2\right] \\ &\leq \mathbb{E}\left[\|\nabla f(w_0^{(s)}) - v_0^{(s)}\|^2\right] + \rho L^2\textstyle\sum_{j=1}^t \mathbb{E}\left[\|w_j^{(s)} - w_{j-1}^{(s)}\|^2\right] \\ &\leq \mathbb{E}\left[\|\nabla f(w_0^{(s)}) - v_0^{(s)}\|^2\right] + \rho L^2\textstyle\sum_{j=1}^t \hat{\eta}_{j-1}^2\mathbb{E}\left[\|v_{j-1}^{(s)}\|^2\right],\end{aligned}$$

where $\rho := \frac{1}{\hat{b}}$ if Algorithm 1 solves (St-OPT) and $\rho := \frac{n-\hat{b}}{\hat{b}(n-1)}$ if Algorithm 1 solves (FS-OPT).

Substituting this estimate into (2.65), and summing up the result from $t = 0$ to $t = m$, we eventually get

$$\begin{aligned}\mathbb{E}\left[f(w_{m+1}^{(s)})\right] \leq \mathbb{E}\left[f(w_0^{(s)})\right] &- \sum_{t=0}^m \frac{\hat{\eta}_t}{2}\mathbb{E}\left[\|\nabla f(w_t^{(s)})\|^2\right] + \frac{1}{2}\Big(\sum_{t=0}^m \hat{\eta}_t\Big)\mathbb{E}\left[\|\nabla f(w_0^{(s)}) - v_0^{(s)}\|^2\right] \\ &+ \frac{\rho L^2}{2}\sum_{t=0}^m \hat{\eta}_t \sum_{j=1}^t \hat{\eta}_{j-1}^2\mathbb{E}\left[\|v_{j-1}^{(s)}\|^2\right] - \sum_{t=0}^m \frac{\hat{\eta}_t(1 - L\hat{\eta}_t)}{2}\mathbb{E}\left[\|v_t^{(s)}\|^2\right].\end{aligned} \tag{2.66}$$

Our next step is to choose $\hat{\eta}_t$ such that

$$\rho L^2\sum_{t=0}^m \hat{\eta}_t \sum_{j=1}^t \hat{\eta}_{j-1}^2\mathbb{E}\left[\|v_{j-1}^{(s)}\|^2\right] - \sum_{t=0}^m \hat{\eta}_t(1 - L\hat{\eta}_t)\mathbb{E}\left[\|v_t^{(s)}\|^2\right] \leq 0.$$

This condition can be rewritten explicitly as

$$\begin{aligned}
&\big[\rho L^2\hat{\eta}_0^2(\hat{\eta}_1+\cdots+\hat{\eta}_m)-\hat{\eta}_0(1-L\hat{\eta}_0)\big]\mathbb{E}\left[\|v_0^{(s)}\|^2\right]\\
&+\big[\rho L^2\hat{\eta}_1^2(\hat{\eta}_2+\cdots+\hat{\eta}_m)-\hat{\eta}_1(1-L\hat{\eta}_1)\big]\mathbb{E}\left[\|v_1^{(s)}\|^2\right]+\cdots\\
&+\big[\rho L^2\hat{\eta}_{m-1}^2\hat{\eta}_m-\hat{\eta}_{m-1}(1-L\hat{\eta}_{m-1})\big]\mathbb{E}\left[\|v_{m-1}^{(s)}\|^2\right]-\hat{\eta}_m(1-L\hat{\eta}_m)\mathbb{E}\left[\|v_m^{(s)}\|^2\right]\leq 0.
\end{aligned}$$

Similar to (2.45), to guarantee the last inequality, we impose the following conditions

$$\begin{cases}
-\hat{\eta}_m(1-L\hat{\eta}_m) & \leq 0,\\
\rho L^2\hat{\eta}_t^2\sum_{j=t+1}^m\hat{\eta}_j-\hat{\eta}_0(1-L\hat{\eta}_0) & \leq 0.
\end{cases}\tag{2.67}$$

Applying Lemma 2.45 (a) with $\nu=\rho$ and $\delta=1$, we obtain

$$\hat{\eta}_m=\frac{1}{L},\quad\text{and}\quad\hat{\eta}_{m-t}:=\frac{1}{L\big(1+\rho L\sum_{j=1}^t\hat{\eta}_{m-j+1}\big)},\quad\forall t=1,\cdots,m,$$

which is exactly (2.38). With this update rule, we have $\frac{1}{L(1+\rho m)}<\hat{\eta}_0<\hat{\eta}_1<\cdots<\hat{\eta}_m$ and $\Sigma_m\geq\frac{2(m+1)}{L(\sqrt{2\rho m+1}+1)}$.

Using the update (2.38), we can simplify (2.66) as follows:

$$\mathbb{E}\left[f(w_{m+1}^{(s)})\right]\leq\mathbb{E}\left[f(w_0^{(s)})\right]-\sum_{t=0}^m\frac{\hat{\eta}_t}{2}\mathbb{E}\left[\|\nabla f(w_t^{(s)})\|^2\right]+\frac{\sum_{t=0}^m\hat{\eta}_t}{2}\mathbb{E}\left[\|\nabla f(w_0^{(s)})-v_0^{(s)}\|^2\right].$$

Let us define $\hat{\sigma}_s:=\mathbb{E}\left[\|\nabla f(w_0^{(s)})-v_0^{(s)}\|^2\right]$ and noting that $f^\star:=F^\star\leq\mathbb{E}\left[f(w_{m+1}^{(S)})\right]$ and $\widetilde{w}_0:=w_0^{(0)}$. Summing up the last inequality from $s=1$ to S and using these relations, we can further derive

$$\sum_{s=1}^S\sum_{t=0}^m\hat{\eta}_t\mathbb{E}\left[\|\nabla f(w_t^{(s)})\|^2\right]\leq 2\big[f(\widetilde{w}_0)-f^\star\big]+\Big(\sum_{t=0}^m\hat{\eta}_t\Big)\sum_{s=1}^S\hat{\sigma}_s.$$

Using the lower bound of Σ_m as $\Sigma_m\geq\frac{2(m+1)}{L(\sqrt{2\rho m+1}+1)}$, the above inequality leads to

$$\frac{1}{S\Sigma_m}\sum_{s=1}^S\sum_{t=0}^m\hat{\eta}_t\mathbb{E}\left[\|\nabla f(w_t^{(s)})\|^2\right]\leq\frac{(\sqrt{2\rho m+1}+1)L}{S(m+1)}\big[f(\widetilde{w}_0)-f^\star\big]+\frac{1}{S}\sum_{s=1}^S\hat{\sigma}_s.\tag{2.68}$$

Since $\mathbf{Prob}\left(\widetilde{w}_T = w_t^{(s)}\right) = p_{(s-1)m+t}$ with $p_{(s-1)m+t} = \frac{\hat{\eta}_t}{S\Sigma_m}$ for $s = 1, \cdots, S$ and $t = 0, \cdots, m$, we have

$$\mathbb{E}\left[\|\nabla f(\widetilde{w}_T)\|^2\right] = \frac{1}{S\Sigma_m}\sum_{s=1}^{S}\sum_{t=0}^{m}\hat{\eta}_t\mathbb{E}\left[\|\nabla f(w_t^{(s)})\|^2\right].$$

Substituting this estimate into (2.68), we obtain (2.39).

Now, we consider two cases:

Case (a): If we apply this algorithm variant to solve the non-composite finite-sum problem of (FS-OPT) (i.e., $\psi = 0$) using the full-gradient snapshot for the outer-loop with $b_s = n$, then $v_0^{(s)} = \nabla f(w_0^{(s)})$, which leads to $\hat{\sigma}_s = 0$. By the choice of epoch length $m = \lfloor \frac{n}{\hat{b}} \rfloor$ and $\hat{b} \leq \sqrt{n}$, we have $\frac{\sqrt{2\rho m+1}+1}{m+1} \leq \frac{2}{\sqrt{n}}$. Using these facts into (2.39), we obtain

$$\mathbb{E}\left[\|\nabla f(\widetilde{w}_T)\|^2\right] \leq \frac{2L}{S\sqrt{n}}\big[f(\widetilde{w}_0) - f^\star\big],$$

which is exactly (2.40).

To achieve $\mathbb{E}\left[\|\nabla f(\widetilde{w}_T)\|^2\right] \leq \varepsilon^2$, we impose $\frac{2L}{S\sqrt{n}}\big[f(\widetilde{w}_0) - f^\star\big] = \varepsilon^2$. Hence, the maximum number of outer iterations is at most $S = \frac{2L}{\sqrt{n}\varepsilon^2}[f(\widetilde{w}_0) - f^\star]$. The number of gradient evaluations ∇f_i is at most $\mathcal{T}_{\text{grad}} := nS + 2(m+1)\hat{b}S \leq 5nS = \frac{10L\sqrt{n}}{\varepsilon^2}[f(\widetilde{w}_0) - f^\star]$.

Case (b): Let us apply this algorithm variant to solve the non-composite expectation problem of (St-OPT) (i.e., $\psi = 0$). Then, by using $\rho := \frac{1}{\hat{b}}$ and $\hat{\sigma}_s := \mathbb{E}\left[\|\nabla f(w_0^{(s)}) - v_0^{(s)}\|^2\right] \leq \frac{\sigma^2}{b_s} = \frac{\sigma^2}{b}$, we have from (2.39) that

$$\mathbb{E}\left[\|\nabla f(\widetilde{w}_T)\|^2\right] \leq \frac{2L}{S\sqrt{\hat{b}m}}\big[f(\widetilde{w}_0) - f^\star\big] + \frac{\sigma^2}{b}.$$

This is exactly (2.41). Using the mini-batch $b := \frac{2\sigma^2}{\varepsilon^2}$ for the outer-loop and $m := \frac{\sigma^2}{\hat{b}\varepsilon^2}$, we can show that the number of outer iterations $S := \frac{4L}{\sigma\varepsilon}\big[f(\widetilde{w}_0) - f^\star\big]$. The number of stochastic gradient evaluations is at most $\mathcal{T}_{\text{grad}} := Sb + 2S(m+1)\hat{b} = \frac{4S\sigma^2}{\varepsilon^2} = \frac{16L\sigma}{\varepsilon^3}\big[f(\widetilde{w}_0) - f^\star\big]$. This holds if $\frac{2\sigma^2}{\varepsilon^2} \leq \frac{4S\sigma^2}{\varepsilon^2} = \frac{16L\sigma}{\varepsilon^3}\big[f(\widetilde{w}_0) - f^\star\big]$ leading to $\sigma \leq \frac{8L}{\varepsilon}\big[f(\widetilde{w}_0) - f^\star\big]$. □

CHAPTER 3

A Hybrid Stochastic Policy Gradient Algorithm for Reinforcement Learning

3.1 Introduction

Recently, research on reinforcement learning (RL) (Sutton and Barto, 2018), an area of machine learning to learn how to make a series of decisions while interacting with the underlying environment, has been immensely active. Unlike supervised learning, reinforcement learning agents often have limited or no knowledge about the environment and the rewards of taking certain actions might not be immediately observed, making these problems more challenging to solve. Over the past decade, there has been a large number of research works developing and using reinforcement learning to solve emerging problems. Notable reinforcement learning agents include, but not limited to, AlphaGo and AlphaZero (Silver et al., 2016, 2018), OpenAIFive (OpenAI, 2018), and AlphaStar (DeepMind, 2019).

In modern RL tasks, the environment is often not known beforehand so the agent has to simultaneously learn the environment while making appropriate decisions. One approach is to estimate the value function or the state-value function such as Q-learning (Watkins and Dayan, 1992) and its variants including Deep Q-learning (DQN) (Mnih et al., 2013, 2015), Dueling DQN (Wang et al., 2016), and double Q-learning (Hasselt et al., 2016).

In this chapter, we take a closer look into a popular method in reinforcement learning, i.e. the policy gradient method, and propose a new algorithm that is able to solve the stochastic composite policy optimization problem.

3.1.1 Problem of interest

Classical policy gradient methods: Policy gradient methods seek a differentiable parameterized policy π_θ that maximizes the expected cumulative discounted rewards as

$$\max_{\theta \in \mathbb{R}^q} \Big\{ J(\theta) := \mathbb{E}_{\tau \sim p_\theta} \left[\mathcal{R}(\tau) \right] \Big\}. \tag{3.1}$$

where q is the parameter dimension and p_θ is the probability density induced by the policy π_θ (see Section 1.2.3). The policy gradient theorem (Sutton et al., 1999) shows that

$$\nabla J(\theta) = \mathbb{E}_{\tau \sim p_\theta} \left[\nabla \log p_\theta(\tau) \mathcal{R}(\tau) \right], \tag{3.2}$$

where the policy gradient does not depend on the gradient of the state distribution despite the fact that the state distribution depends on the policy parameters (Silver et al., 2014).

This policy gradient can be used in gradient ascent algorithms to update the parameter θ. However, we cannot calculate the full gradient at each update as we only get a finite number of samples at each iteration. Consequently, the policy gradient is often estimated by its sample average. At each iteration, a batch of trajectories $\mathcal{B} = \{\tau_i\}_{i=1,\cdots,N}$ will be sampled from the environment to estimate the policy gradient as

$$\widetilde{\nabla} J(\theta) := \frac{1}{N} \sum_{i=1}^{N} g(\tau_i | \theta),$$

where $g(\tau_i|\theta)$ is a sample estimator of $\mathbb{E}_{\tau_i \sim p_\theta} \left[\nabla \log p_\theta(\tau_i) \mathcal{R}(\tau_i) \right]$. We call $\widetilde{\nabla} J(\theta)$ a stochastic policy gradient (SPG) estimator. This estimator has been exploited in the two well-known REINFORCE (Williams, 1992) and GPOMDP (Baxter and Bartlett, 2001) methods. The main step of policy gradient ascent methods is to update the parameters as

$$\theta_{t+1} := \theta_t + \eta \nabla J(\theta_t), \ t = 0, 1, \cdots,$$

where $\eta > 0$ is some appropriate learning rate, which can be fixed or varied over t. Since the policy changes after each update, the density $p_\theta(\cdot)$ also changes and creates non-stationarity in the problem which will be handled by importance weight in Section 3.2.

While the objective function in (3.1) is standard in most policy gradient methods, it is natural to have some constraints or regularizers on the policy parameters. In addition, adding constraints can prevent the explosion of parameters in highly nonlinear models as often seen in deep learning (Srivastava et al., 2014). Adopting the idea of composite nonconvex optimization (Pham et al., 2020b), we are interested in a more general optimization problem in reinforcement learning as follow:

$$\max_{\theta \in \mathbb{R}^q} \Big\{ J(\theta) - Q(\theta) = \mathbb{E}_{\tau \sim p_\theta} \left[\mathcal{R}(\tau) \right] - Q(\theta) \Big\}, \tag{CP-OPT}$$

where $Q(\theta)$ is a proper, closed, and convex function acting as a regularizer which can be the indicator function of a convex set representing the constraints on the parameters or some standard regularizers such as ℓ_1-norm or ℓ_2-norm. If there is no regularizer $Q(\theta)$, the problem (CP-OPT) reduces to the standard one in (3.1).

3.1.2 Related work

It has been observed that learning the state-value function is not efficient when the action space is large or even infinite. In that case, policy gradient methods learn the policy directly with a parameterized function. Silver et al. (2014) present a framework for deterministic policy gradient algorithms which can be estimated more efficiently than their stochastic counterparts whereas Lillicrap et al. (2016) adapt the idea of deep Q-learning into continuous action tasks in RL. TRPO (Schulman et al., 2015) uses a constraint on the KL divergence between the new and old policies to improve the robustness of each update. PPO (Schulman et al., 2017) is an extension of TRPO which uses a clipped surrogate objective resulting a simpler implementation. Other policy gradient methods utilize the actor-critic paradigm including ACER (Wang et al., 2017), A3C (Mnih et al., 2016) and its synchronous variant A2C, ACKTR (Wu et al., 2017), and SAC (Haarnoja et al., 2018).

REINFORCE (Williams, 1992) is perhaps one classical method closely related to our work here. It first computes an estimator of the policy gradient in (3.2) and applies a gradient ascent

step to update the policy. Nevertheless, the REINFORCE estimator is known to have high variance leading to several weaknesses. Other improvements to reduce the variance such as adding baselines (Sutton and Barto, 2018; Zhao et al., 2011), discarding some rewards in the so-called GPOMDP estimator (Baxter and Bartlett, 2001) were proposed. While REINFORCE estimator is an unbiased policy gradient estimator, GPOMDP is shown to be biased (Baxter and Bartlett, 2001) making theoretical analysis harder.

The nature of REINFORCE algorithm appears to be closely related to stochastic gradient descent (SGD) (Robbins and Monro, 1951) in stochastic nonconvex optimization. In particular, the standard SGD estimator is also known to often have fixed variance, which is often high. On the one hand, there are algorithms trying to reduce the oscillation (Tieleman and Hinton, 2012) or introduce momentums or adaptive updates (Allen-Zhu, 2017a, 2018; Kingma and Ba, 2014) for SGD methods to accelerate performance. On the other hand, other researchers are searching for new gradient estimators. One approach is the SAGA estimator proposed by Defazio et al. (2014). Another well-known estimator is the SVRG estimator (Johnson and Zhang, 2013) which has been intensively studied in recent works, e.g., in Allen-Zhu and Yuan (2016); Li and Li (2018); Reddi et al. (2016a); Zhou et al. (2018b). This estimator not only overcomes the storage issue of SAGA but also possesses variance reduced property, i.e., the variance of the estimator decreases over epochs. Methods based on SVRG estimators have recently been developed for reinforcement learning, e.g., SVRPG (Papini et al., 2018). Xu et al. (2019a) refine the analysis of SVRPG to achieve an improved trajectory complexity of $\mathcal{O}\left(\varepsilon^{-10/3}\right)$. Shen et al. (2019) also adopt the SVRG estimator into policy gradient and achieve the trajectory oracle complexity of $\mathcal{O}\left(\varepsilon^{-3}\right)$ with the use of a second-order estimator.

While SGD, SAGA, and SVRG estimators are unbiased, there have been algorithms developed based on a biased gradient estimator named SARAH (Nguyen et al., 2017a). Such algorithms include SARAH (Nguyen et al., 2017b, 2019), SPIDER (Fang et al., 2018), SpiderBoost (Wang et al., 2019), and ProxSARAH (Pham et al., 2020b). Similar to SVRG, all these methods can potentially be extended to reinforcement learning. A recent attempt is SARAPO (Yuan et al., 2019) which combines SARAH (Nguyen et al., 2019) with TRPO (Schulman et al., 2015) algorithm but no theoretical guarantee is provided. Yang and Zhang (2019) propose Mirror Policy Optimization (MPO) algorithm which covers the classical policy gradient and the

natural policy gradient as special cases. They also introduce a variance reduction variant, called VRMPO, which achieves $\mathcal{O}\left(\varepsilon^{-3}\right)$ trajectory complexity. Another notable work is SRVR-PG (Xu et al., 2019b) where the policy gradient estimator is the adapted version of SARAH estimator for reinforcement learning. Note that Yang and Zhang (2019) and Xu et al. (2019b) achieve the same trajectory complexity of $\mathcal{O}\left(\varepsilon^{-3}\right)$ as ours. However, our algorithm is essentially different. Xu et al. (2019b) and Yang and Zhang (2019) use two different adaptation of the SARAH estimator for policy gradient. Xu et al. (2019b) use the importance weight in their estimator to handle distribution shift while Yang and Zhang (2019) remove it as seen in Shen et al. (2019). Meanwhile, we introduce a new policy gradient estimator which can also be calculated recursively. The new estimator is fundamentally different from the other two since it combines the adapted SARAH estimator as in Xu et al. (2019b) with the classical REINFORCE estimator. In addition, our analysis shows that the best-known convergence rate and complexity can be achieved by our single-loop algorithm (Algorithm 2) while SRVR-PG and VRMPO require double loops to achieve the same oracle complexity. Moreover, Xu et al. (2019b); Yang and Zhang (2019) do not consider the composite setting that considers both constraints and regularizers on the policy parameters as we do.

3.1.3 Our approach and contribution

Our approach lies in the stochastic variance reduction avenue, but using a completely new **hybrid** approach, leading to a novel estimator compared to existing methods in reinforcement learning. We construct our estimator by taking a convex combination of two estimators: the adapted SARAH (Nguyen et al., 2017a) and REINFORCE (Williams, 1992), a classical unbiased policy gradient estimator. This hybrid estimator not only allows us to trade-off the bias and variance between these two estimators but also possesses useful properties for developing new algorithms. Note that the idea of combining stochastic estimators was first proposed for stochastic optimization in our recent works (Tran-Dinh et al., 2019a,b). Unlike existing policy gradient methods, our algorithm first samples a large batch of trajectories to establish a good search direction. After that, it iteratively updates the policy parameters using our hybrid estimator leading to a single-loop method without any snapshot loop as in SVRG or SARAH variants. In addition, as regularization techniques have shown their effectiveness in deep learning

(Neyshabur et al., 2017; Zhang et al., 2017), they possibly have great potential in reinforcement learning algorithms too. A recent study (Liu et al., 2019) shows that regularizations on the policy parameters can greatly improve the performance of policy gradient algorithms. Motivated by these facts, we directly consider a new composite setting (CP-OPT) as presented in Section 3.2. For this new composite model, it is not clear if existing algorithms remain convergent by simply adding a projection step on the constraint set, while our method does guarantee convergence.

To this end, our contribution can be summarized as follows:

(a) We introduce a novel hybrid stochastic policy gradient estimator by combining existing REINFORCE estimator with the adapted SARAH estimator for policy gradient. We investigate some key properties of our estimator that can be used for algorithmic development.

(b) We propose a new algorithm to solve a composite maximization problem for policy optimization in reinforcement learning. Our model not only covers existing settings but also handles constraints and convex regularizers on policy parameters.

(c) We provide convergence analysis as the first theoretical result for composite optimization in reinforcement learning and estimate the trajectory complexity of our algorithm and show that our algorithm can achieve the best-known complexity over existing first-order methods (see Table 3.1).

Our algorithm only has one loop as REINFORCE or GPOMDP, which is fundamentally different from SVRPG, SVRG-adapted, and other SARAH-based algorithms for RL. It can work with single sample or mini-batch and has two steps: proximal gradient step and averaging step with different step-sizes. This makes the algorithm more flexible to use different step-sizes without sacrificing the overall complexity.

3.1.4 Chapter outline

The rest of this chapter is organized as follows. Section 3.2 introduces our new hybrid estimator for policy gradient and develops the main algorithm. Section 3.3 presents a key property of the new hybrid policy gradient estimator then provide the complexity analysis of our proposed algorithms. Multiple numerical experiments in both discrete and continuous control tasks are illustrated in Section 3.4. The proofs of some technical results are given in Section 3.5.

Algorithms	Complexity	Composite	Single-loop
REINFORCE (Williams, 1992)	$\mathcal{O}\left(\varepsilon^{-4}\right)$	☒	☑
GPOMDP (Baxter and Bartlett, 2001)	$\mathcal{O}\left(\varepsilon^{-4}\right)$	☒	☑
SVRPG (Papini et al., 2018)	$\mathcal{O}\left(\varepsilon^{-4}\right)$	☒	☒
SVRPG (Xu et al., 2019a)	$\mathcal{O}\left(\varepsilon^{-10/3}\right)$	☒	☒
HAPG (Shen et al., 2019)	$\mathcal{O}\left(\varepsilon^{-3}\right)$	☒	☒
VRMPO (Yang and Zhang, 2019)	$\mathcal{O}\left(\varepsilon^{-3}\right)$	☒	☒
SRVR-PG (Xu et al., 2019b)	$\mathcal{O}\left(\varepsilon^{-3}\right)$	☒	☒
ProxHSPGA (this work)	$\mathcal{O}\left(\varepsilon^{-3}\right)$	☑	☑

Table 3.1: A summary of various methods for the non-composite setting (3.1) of (CP-OPT).

3.2 A new hybrid stochastic policy gradient algorithm

In this section, we first present the standard assumptions used to analyze the convergence of our algorithms and discuss the optimality condition of (CP-OPT). After that, we show how to extend the hybrid gradient idea from Tran-Dinh et al. (2019b) to form a new policy gradient estimator. We then develop a new proximal policy gradient algorithm and its restart variant to solve the composite policy optimization problem and analyze their trajectory complexity.

3.2.1 Assumptions

Let $F(\theta) := J(\theta) - Q(\theta)$ be the total objective function. We impose the following assumptions for our convergence analysis, which are often used in practice.

Assumption 3.1. The regularizer $Q : \mathbb{R}^q \to \mathbb{R} \cup \{+\infty\}$ is a proper, closed, and convex function. We also assume that the domain of F is nonempty and there exists a finite upper bound

$$F^* := \sup_{\theta \in \mathbb{R}^q} \{F(\theta) := J(\theta) - Q(\theta)\} < +\infty.$$

Assumption 3.2. The immediate reward function is bounded, i.e., there exists $R > 0$ such that for all $a \in \mathcal{A}$, $s \in \mathcal{S}$, $|\mathcal{R}(s, a)| \leq R$.

Assumption 3.3. Let $\pi_\theta(s,a)$ be the policy for a given state-action pair (s,a). Then, there exist two positive constants G and M such that

$$\|\nabla \log \pi_\theta(s,a)\| \leq G \text{ and } \|\nabla^2 \log \pi_\theta(s,a)\| \leq M,$$

for any $a \in \mathcal{A}$, $s \in \mathcal{S}$ where $\|\cdot\|$ is the ℓ_2-norm.

This assumption leads to useful results about the smoothness of $J(\theta)$ and $g(\tau|\theta)$ and the upper bound on the variance of the policy gradient estimator.

Remark 4. Assumption 3.3 holds for Gaussian policy, a common policy used for continuous action tasks, where the probability to take action a given current state s is define as.

$$\pi_\theta(s,a) := \frac{1}{\sqrt{2\pi}\sigma(s)} e^{-\frac{(a-\mu(s))^2}{2\sigma^2(s)}},$$

where $\mu(s) = \Phi_\theta(s)$ is the mean approximated by a parametrized function Φ_θ and $\sigma(s)$ is the standard deviation which can be fixed or parametrized.

Lemma 3.1 (Papini et al. (2018); Shen et al. (2019); Xu et al. (2019a))**.** *Under Assumption 3.2 and 3.3, for all $\theta, \theta_1, \theta_2 \in \mathbb{R}^q$, we have*

- $\|\nabla J(\theta_1) - \nabla J(\theta_2)\| \leq L\|\theta_1 - \theta_2\|$;
- $\|g(\tau|\theta_1) - g(\tau|\theta_2)\| \leq L_g \|\theta_1 - \theta_2\|$;
- $\|g(\tau,\theta)\| \leq C_g$; *and*
- $\|g(\tau|\theta) - \nabla J(\theta)\|^2 \leq \sigma^2$,

where $g(\cdot)$ is the REINFORCE estimator and L, L_g, C_g, and σ^2 are constants depending only on R, G, M, H, γ, and the baseline b.

For more details about the constants and the proofs of Lemma 3.1 we refer to, e.g. Papini et al. (2018); Shen et al. (2019); Xu et al. (2019a).

Assumption 3.4. There exists a constant $W > 0$ such that, for each pair of policies encountered in our algorithms the following holds

$$\text{Var}\left[\omega(\tau|\theta_1,\theta_2)\right] \leq W, \quad \theta_1,\theta_2 \in \mathbb{R}^q,\ \tau \sim p_{\theta_1},$$

where $\omega(\tau|\theta_1, \theta_2) = \frac{p_{\theta_2}(\tau)}{p_{\theta_1}(\tau)}$ is the importance weight between $p_{\theta_2}(\cdot)$ and $p_{\theta_1}(\cdot)$.

Since the importance weight ω introduces another source of variance, we require this assumption for our convergence analysis as used in previous works, e.g., in Papini et al. (2018); Xu et al. (2019a).

Remark 5. Cortes et al. (2010) show that if σ_Q, σ_P are variances of two Gaussian distributions P and Q, and $\sigma_Q > \frac{1}{\sqrt{2}}\sigma_P$ then the variance of the importance weights is bounded, i.e. Assumption 3.4 holds for Gaussian policies which are commonly used to represent the policy in continuous control tasks.

3.2.2 Optimality condition

Associated with problem (CP-OPT), we define

$$\mathcal{G}_\eta(\theta) := \frac{1}{\eta}\left[\text{prox}_{\eta Q}\left(\theta + \eta\nabla J(\theta)\right) - \theta\right], \tag{3.3}$$

for some $\eta > 0$ as the gradient mapping of $F(\theta)$ (Nesterov, 2013), where $\text{prox}_Q(\theta)$ denotes the proximal operator of Q as in (1.3).

A point θ^* is called a stationary point of (CP-OPT) if

$$\mathbb{E}\left[\|\mathcal{G}_\eta(\theta^*)\|^2\right] = 0.$$

Our goal is to design an iterative method to produce an ε-approximate stationary point $\tilde{\theta}_T$ of (CP-OPT) after at most T iterations defined as

$$\mathbb{E}\left[\|\mathcal{G}_\eta(\tilde{\theta}_T)\|^2\right] \leq \varepsilon^2,$$

where $\varepsilon > 0$ is a desired tolerance, and the expectation is taken over all the randomness up to the T-th iteration.

3.2.3 Novel hybrid stochastic policy gradient estimator

We first provide a brief summary of the REINFORCE estimator then introduce a new stochastic policy gradient (SPG) estimator.

3.2.3.1 REINFORCE - an unbiased estimator:

Recall that given a trajectory $\tau := \{s_0, a_0, \cdots, s_{H-1}, a_{H-1}\}$, the REINFORCE SPG estimator is defined as

$$g(\tau|\theta) \quad := \left[\sum_{t=0}^{H-1} \nabla \log \pi_\theta(a_t|s_t)\right] \mathcal{R}(\tau),$$

where $\mathcal{R}(\tau) := \sum_{t=0}^{H-1} \gamma^t \mathcal{R}(s_t, a_t)$.

Note that the REINFORCE estimator is unbiased, i.e. $\mathbb{E}_{\tau\sim p_\theta}\left[g(\tau|\theta)\right] = \nabla J(\theta)$. In order to reduce the variance of these estimators, a baseline is normally added while maintaining the unbiasedness of the estimators (Sutton and Barto, 2018; Zhao et al., 2011). From now on, we will refer to $g(\tau|\theta)$ as the baseline-added version defined as

$$g(\tau|\theta) \quad := \sum_{t=0}^{T-1} \nabla \log \pi_\theta(a_t|s_t) A_t,$$

where $A_t := \mathcal{R}(\tau) - b_t$ with b_t being a baseline and possibly depending only on s_t.

3.2.3.2 New stochastic policy gradient estimator:

In order to reduce the number of trajectories sampled, we extend the idea in Tran-Dinh et al. (2019b) for stochastic optimization to develop a new **hybrid** stochastic policy gradient (HSPG) estimator that helps balance the bias-variance trade-off. The estimator is formed by taking a convex combination of two other estimators: one is an unbiased estimator which can be REINFORCE estimator, and another is the adapted SARAH estimator (Nguyen et al., 2019) for policy gradient which is biased.

More precisely, if $\mathcal{B}_t$ and $\widehat{\mathcal{B}}_t$ are two random batches of trajectories with sizes B and $\widehat{B}$, respectively, sampled from $p_{\theta_t}(\cdot)$, the hybrid stochastic policy gradient estimator at t-th iteration

can be expressed as

$$v_t \quad := \beta v_{t-1} + \tfrac{\beta}{B} \sum_{\tau \in \mathcal{B}_t} [g(\tau|\theta_t) - \omega(\tau|\theta_t, \theta_{t-1}) g(\tau|\theta_{t-1})] + \frac{(1-\beta)}{\widehat{B}} \sum_{\hat{\tau} \in \widehat{\mathcal{B}}_t} g(\hat{\tau}|\theta_t), \tag{3.4}$$

and

$$v_0 := \frac{1}{N} \sum_{\tau \in \tilde{\mathcal{B}}} g(\tau|\theta_0),$$

where $\tilde{\mathcal{B}}$ is a batch of trajectories collected at the beginning. Note that $\omega(\tau|\theta_t, \theta_{t-1})$ is an importance weight added to account for the distribution shift since the trajectories $\tau \in \mathcal{B}_t$ are sampled from $p_{\theta_t}(\cdot)$ but not from $p_{\theta_{t-1}}(\cdot)$. Note also that v_t in (3.4) is also different from the momentum SARAH estimator recently proposed in Cutkosky and Orabona (2019).

3.2.4 The complete algorithm

The novel Proximal Hybrid Stochastic Policy Gradient Algorithm (abbreviated by ProxHSPGA) to solve (CP-OPT) is presented in Algorithm 2.

Algorithm 2 (ProxHSPGA)

1: **Initialization:** An initial point $\theta_0 \in \mathbb{R}^q$, and positive parameters m, N, B, $\widehat{B}$, β, α, and η (specified later).
2: Sample a batch of trajectories $\tilde{\mathcal{B}}$ of size N from $p_{\theta_0}(\cdot)$.
3: Calculate $v_0 := \frac{1}{N} \sum_{\tau \in \tilde{\mathcal{B}}} g(\tau|\theta_0)$.
4: Update

$$\begin{cases} \widehat{\theta}_1 & := \text{prox}_{\eta Q}(\theta_0 + \eta v_0) \\ \theta_1 & := (1-\alpha)\theta_0 + \alpha \widehat{\theta}_1. \end{cases}$$

5: **For** $t := 1, \cdots, m$ **do**
6: Generate 2 independent batches of trajectories $\mathcal{B}_t$ and $\widehat{\mathcal{B}}_t$ with size B and $\hat{B}$ from $p_{\theta_t}(\cdot)$.
7: Evaluate the hybrid estimator v_t as in (3.4).
8: Update

$$\begin{cases} \widehat{\theta}_{t+1} & := \text{prox}_{\eta Q}(\theta_t + \eta v_t) \\ \theta_{t+1} & := (1-\alpha)\theta_t + \alpha \widehat{\theta}_{t+1}. \end{cases}$$

9: **EndFor**
10: Choose $\widetilde{\theta}_T$ from $\{\theta_t\}_{t=1}^m$ uniformly randomly.

Unlike SVRPG (Papini et al., 2018; Xu et al., 2019a) and HAPG (Shen et al., 2019), Algorithm 2 only has **one loop** as REINFORCE or GPOMDP. Moreover, Algorithm 2 does not use the estimator for the policy Hessian as in HAPG. At the initial stage, a batch of trajectories

is sampled using p_{θ_0} to estimate an initial policy gradient estimator which provides a good initial search direction. At the t-th iteration, two independent batches of trajectories are sampled from p_{θ_t} to evaluate the hybrid stochastic policy gradient estimator. After that, a proximal step followed by an averaging step are performed which are inspired by Pham et al. (2020b). Note that the batches of trajectories at each iteration are sampled from the current distribution which will change after each update. Therefore, the importance weight $\omega(\tau|\theta_t, \theta_{t-1})$ is introduced to account for the non-stationarity of the sampling distribution. As a result, we still have $\mathbb{E}_{\tau\sim p_{\theta_t}}[\omega(\tau|\theta_t, \theta_{t-1})g(\tau|\theta_{t-1})] = \nabla J(\theta_{t-1})$.

3.2.5 Restarting variant

While Algorithm 2 has the best-known theoretical complexity as shown in Section 3.3, its practical performance may be affected by the constant step-size α depending on m. As will be shown later, the step-size $\alpha \in [0, 1]$ is inversely proportional to the number of iterations m and it is natural to have α close to 1 to take advantage of the newly computed information. To increase the practical performance of our algorithm without sacrificing its complexity, we propose to inject a simple restarting strategy by repeatedly running Algorithm 2 for multiple stages as in Algorithm 3.

Algorithm 3 (Restarting ProxHSPGA)

1: **Initialization:** Input an initial point $\theta_0^{(0)}$.
2: **For** $s := 0, \cdots, S-1$ **do**
3: Run Algorithm 2 with $\theta_0 := \theta_0^{(s)}$.
4: Output $\theta_0^{(s+1)} := \theta_{m+1}$.
5: **EndFor**
6: Choose $\widetilde{\theta}_T$ uniformly randomly from $\{\theta_t^{(s)}\}_{t=0\to m}^{s=0\to S-1}$.

We emphasize that without this restarting strategy, Algorithm 2 still converges and the restarting loop in Algorithm 3 does not sacrifice the best-known complexity as stated in the next section.

3.3 Convergence analysis

This section presents key properties of the hybrid stochastic policy gradient estimators as well as the theoretical convergence analysis and complexity estimate.

3.3.1 Properties of the hybrid SPG estimator

Let $\mathcal{F}_t := \sigma\left(\tilde{\mathcal{B}}, \mathcal{B}_1, \widehat{\mathcal{B}}_1, \cdots, \mathcal{B}_{t-1}, \widehat{\mathcal{B}}_{t-1}\right)$ be the σ-field generated by all trajectories sampled up to the t-th iteration. For the sake of simplicity, we assume that $B = \widehat{B}$ but our analysis can be easily extended for the case $B \neq \widehat{B}$. Consequently, the hybrid SPG estimator v_t has the following properties

Lemma 3.2 (Key properties)**.** *Let v_t be defined as in* (3.4) *and $\Delta v_t := v_t - \nabla J(\theta_t)$. Then*

$$\mathbb{E}_{\tau,\hat{\tau}\sim p_{\theta_t}}\left[v_t\right] = \nabla J(\theta_t) + \beta \Delta v_{t-1}. \tag{3.5}$$

If $\beta \neq 0$, then v_t is an biased estimator. In addition, we have

$$\mathbb{E}_{\tau,\hat{\tau}\sim p_{\theta_t}}\left[\|\Delta v_t\|^2\right] \leq \beta^2 \|\Delta v_{t-1}\|^2 + \frac{\beta^2 \overline{C}}{B}\|\theta_t - \theta_{t-1}\|^2 + \frac{(1-\beta)^2\sigma^2}{B}, \tag{3.6}$$

where $\overline{C} > 0$ is a given constant.

The proof of Lemma 3.2 and definition of the constants are given in Section 3.5.1.

3.3.2 Complexity estimates

The following lemma presents a key estimate for our convergence results whose proof is given in Section 3.5.2.

Lemma 3.3 (One-iteration analysis)**.** *Under Assumptions 3.2, 3.3, and 3.4, let $\{\widehat{\theta}_t, \theta_t\}_{t=0}^m$ be the sequence generated by Algorithm 2 and $\mathcal{G}_\eta$ be the gradient mapping defined in* (3.3)*. Then*

$$\mathbb{E}\left[F(\theta_{t+1})\right] \geq \mathbb{E}\left[F(\theta_t)\right] + \frac{\eta^2\alpha}{2}\mathbb{E}\left[\|\mathcal{G}_\eta(\theta_t)\|^2\right] - \frac{\xi}{2}\mathbb{E}\left[\|v_t - \nabla J(\theta_t)\|^2\right] + \frac{\zeta}{2}\mathbb{E}\left[\|\widehat{\theta}_{t+1} - \theta_t\|^2\right], \tag{3.7}$$

where $\xi := \alpha(1+2\eta^2)$ and $\zeta := \alpha\left(\frac{2}{\eta} - L\alpha - 3\right) > 0$ provided that $\alpha \in (0, 1]$ and $\frac{2}{\eta} - L\alpha - 3 > 0$.

Using Lemma 3.2 and 3.3, we can show the convergence analysis of Algorithm 2 as follows.

Theorem 3.1. *Under Assumptions 3.1, 3.2, 3.3, and 3.4, let* $\{\theta_t\}_{t=0}^m$ *be the sequence generated by Algorithm 2 with*

$$\begin{cases} \beta := 1 - \frac{\sqrt{B}}{\sqrt{N(m+1)}} \\ \alpha := \frac{\hat{c}\sqrt{2}B^{3/4}}{\sqrt{3\overline{C}}N^{1/4}(m+1)^{1/4}} \\ \eta := \frac{2}{4+L\alpha}, \end{cases}, \tag{3.8}$$

where $B, \hat{c}, L,$ *and* $\overline{C}$ *are given constants. If* $\tilde{\theta}_T$ *is chosen uniformly at random from* $\{\theta_t\}_{t=0}^m$*, then the following estimate holds*

$$\mathbb{E}\left[\|\mathcal{G}_\eta(\tilde{\theta}_T)\|^2\right] \leq \frac{3(4+L)^2\sigma^2}{4[BN(m+1)]^{1/2}} + \frac{(4+L)^2\sqrt{3\overline{C}}N^{1/4}}{4\hat{c}\sqrt{2}[B(m+1)]^{3/4}}\left[F^* - F(\theta_0)\right]. \tag{3.9}$$

The proof of Theorem 3.1 is given in Section 3.5.3. Consequently, the trajectory complexity is presented in the following corollary whose proof can be found in Section 3.5.4.

Corollary 3.1. *For both Algorithm 2 and Algorithm 3, let us fix* $B \in \mathbb{N}_+$ *and set* $N := \tilde{c}\sigma^{8/3}[B(m+1)]^{1/3}$ *for some* $\tilde{c} > 0$ *in Theorem 3.1. If we also choose* m *in Algorithm 2 such that* $m+1 = \frac{\Psi_0^{3/2}\sigma}{B\varepsilon^3}$ *and choose* m, S *in Algorithm 3 such that* $S(m+1) = \frac{\Psi_0^{3/2}\sigma}{B\varepsilon^3}$ *for some constant* Ψ_0*, then the number of trajectories* $\mathcal{T}_{traj}$ *to achieve* $\tilde{\theta}_T$ *such that* $\mathbb{E}\left[\|\mathcal{G}_\eta(\tilde{\theta}_T)\|^2\right] \leq \varepsilon^2$ *for any* $\varepsilon > 0$ *is at most*

$$\mathcal{T}_{traj} = \mathcal{O}\left(\varepsilon^{-3}\right).$$

where $\tilde{\theta}_T$ *is chosen uniformly at random from* $\{\theta_t^{(s)}\}_{t=0,\cdots,m}^{s=0,\cdots,S-1}$ *if using Algorithm 3.*

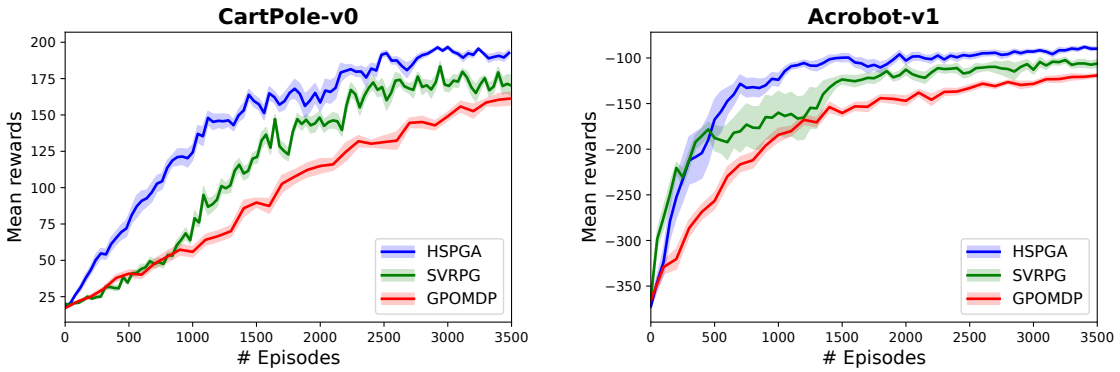
Comparing our complexity bound with other existing methods in Table 3.1, we can see that we improve a factor of $\boldsymbol{\varepsilon^{-1/3}}$ over SVRPG in Xu et al. (2019a) while matching the best-known complexity without the need of using the policy Hessian estimator as HAPG from Shen et al. (2019).

3.4 Numerical experiments

In this section, we present three examples to provide comparison between the performance of HSPGA and other related policy gradient methods. We also provide an example to illustrate

the effect of the regularizer $Q(\cdot)$ in the composite problem (CP-OPT). All experiments are run on a Macbook Pro with 2.3 GHz Quad-Core, 8GB RAM. The source code is available at

https://github.com/unc-optimization/ProxHSPGA.



(a) (b)

Figure 3.1: Performance of three algorithms on `Carpole-v0` and `Acrobot-v1` environments.

We implement our restarting algorithm, Algorithm 3, on top of the `rllab`[1] library (Duan et al., 2016). We compare our algorithm with two other methods: SVPRG (Papini et al., 2018; Xu et al., 2019a) and GPOMDP (Baxter and Bartlett, 2001). Although REINFORCE and GPOMDP have the same trajectory complexity, as observed in Papini et al. (2018), GPOMDP often performs better than REINFORCE, so we only choose to implement GPOMDP in our experiments. Since SVRPG and GPOMDP solves the non-composite problems (3.1), we set $Q(\theta) = 0$ in the first three examples and adjust our algorithm, denoted as HSPGA, accordingly. We compare our algorithm with the fixed epoch length variant of SVRPG as reported in Papini et al. (2018); Xu et al. (2019a). For the implementation of SVRPG and GPOMDP, we reuse the implementation of Papini et al.[2]. We test these algorithms on three well-studied reinforcement learning tasks: `Cart Pole`, `Acrobot`, and `Moutain Car` which are available in OpenAI `gym` (Brockman et al., 2016), a well-known toolkit for developing and comparing reinforcement learning algorithms. We also test these algorithms on continuous control tasks using other simulators such as `Roboschool` (Klimov and Schulman, 2017) and `Mujoco` (Todorov et al., 2012).

[1]Available at https://github.com/rll/rllab

[2]Available at https://github.com/Dam930/rllab

Environment	Algorithm	Policy Network	Discount Factor γ	Trajectory Length H	Minibatch Size	Snapshot Batchsize	Learning Rate	Epoch Length m
CartPole-v0	GPOMDP	$4 \times 8 \times 2$	0.99	200	10		10^{-3}	
	SVRPG				10	25	5×10^{-3}	3
	HSPGA				5	25	5×10^{-3}	3
Acrobot-v1	GPOMDP	$6 \times 16 \times 3$	0.999	500	10		2.5×10^{-3}	
	SVRPG				5	10	5×10^{-3}	3
	HSPGA				3	10	5×10^{-3}	3
MoutainCar-v0	GPOMDP	$2 \times 8 \times 1$	0.999	1000	25		5×10^{-3}	
	SVRPG				10	50	7.5×10^{-3}	3
	HSPGA				5	50	7.5×10^{-3}	3
RoboschoolInvertedPendulum-v1	GPOMDP	$5 \times 16 \times 1$	0.999	1000	20		7.5×10^{-4}	
	SVRPG				10	50	10^{-3}	3
	HSPGA				5	50	10^{-3}	3
	ProxHSPGA				5	50	10^{-3}	3
Swimmer-v2	GPOMDP	$8 \times 32 \times 32 \times 2$	0.99	500	50		5×10^{-4}	
	SVRPG				5	50	5×10^{-4}	3
	HSPGA				5	50	5×10^{-4}	3
	ProxHSPGA				5	50	5×10^{-4}	3
Hopper-v2	GPOMDP	$11 \times 32 \times 32 \times 3$	0.99	500	50		5×10^{-4}	
	SVRPG				5	50	5×10^{-4}	3
	HSPGA				5	50	5×10^{-4}	3
	ProxHSPGA				5	50	5×10^{-4}	3
Walker2d-v2	GPOMDP	$17 \times 32 \times 32 \times 6$	0.99	500	50		5×10^{-4}	
	SVRPG				5	50	5×10^{-4}	3
	HSPGA				5	50	5×10^{-4}	3
	ProxHSPGA				5	50	5×10^{-4}	3

Table 3.2: All algorithms' configurations on discrete and continuous control environments.

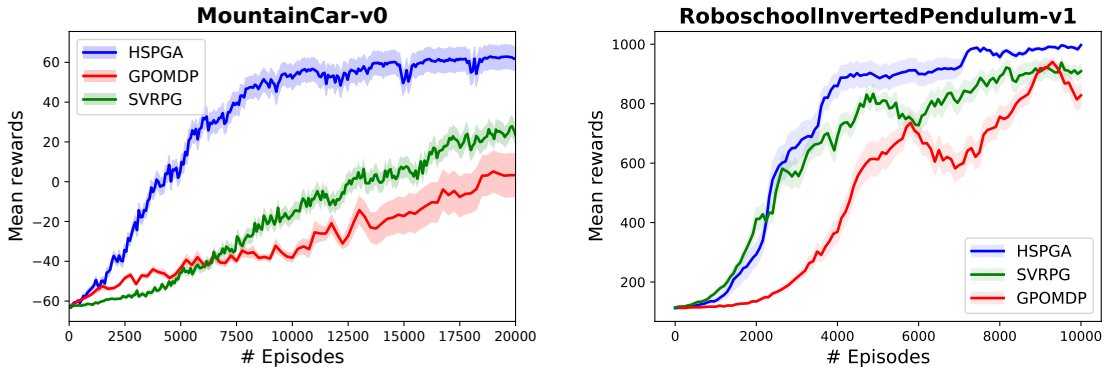
For each environment, we initialize the policy randomly and use it as initial policies for all 10 runs of all algorithms. The performance measure, i.e., mean rewards, is computed by averaging the final rewards of 50 trajectories sampled by the current policy. We then compute the mean and 90% confidence interval across 10 runs of these performance measures at different time point. In all plots, the solid lines represent the mean and the shaded areas are the confidence band of the mean rewards.

The configurations of all algorithms considered in this section are chosen as follows. We set $\beta := 0.99$ for HSPGA and $\alpha := 0.99$ for ProxHSPGA in all experiments. To choose the learning rate, we conduct a grid search over different choices. For `Acrobot-v1`, `Cart pole-v0`, and `Mountain Car-v0` environments, we consider different values ranging from 0.0005 to 0.01. Meanwhile, we use values from 0.0005 to 0.005 for the remaining environments. The snapshot batch-sizes are also chosen from $\{10, 25, 50, 100\}$ while the mini-batch sizes are selected from $\{3, 5, 10, 15, 20, 25\}$. More details about the selected parameters for each experiment are shown in Table 3.2. We note that the architecture of the neural network is denoted as [observation space] $\times$ [hidden layers] $\times$ [action space].

Cart Pole-v0 environment: For the `Cart pole` environment, we use a deep soft-max policy network (Bridle, 1990; Levine, 2017; Sutton and Barto, 2018) with one hidden layer of 8

neurons. Figure 3.1a depicts the results where we run each algorithm for 10 trials where the solid lines represent the mean rewards and shaded regions represent the 90% confidence intervals.

From Figure 3.1a, we can see that HSPGA outperforms the other 2 algorithms while SVRPG works better than GPOMDP as expected. HSPGA is able to reach the maximum reward of 200 in less than 4000 episodes.



(a) (b)

Figure 3.2: Performance of three non-composite algorithms on the `MountainCar-v0` environment and the `Roboschool Inverted Pendulum-v1` environments.

Acrobot environment: Next, we evaluate three algorithms on the `Acrobot-v1` environment. Here, we use a deep soft-max policy with one hidden layer of 16 neurons. The performance of these 3 algorithms are illustrated in Figure 3.1b.

We observe similar results as in the previous example where HSPGA has the best performance over three candidates. SVRPG is still better than GPOMDP in this example.

Mountain Car environment: For the `MountainCar-v0` environment, we use a deep Gaussian policy (Sutton and Barto, 2018) where the mean is the output of a neural network containing one hidden layer of 8 neurons and the standard deviation is fixed at 1. The results of three algorithms are presented in Figure 3.2a.

Figure 3.2a shows that HSPGA highly outperforms the other two algorithms. Again, SVRPG remains better than GPOMDP as expected.

Inverted Pendulum environment: We also test these three algorithms on the `Inverted Pendulum-v0` environment. The results of three algorithms are presented in Figure 3.2a.

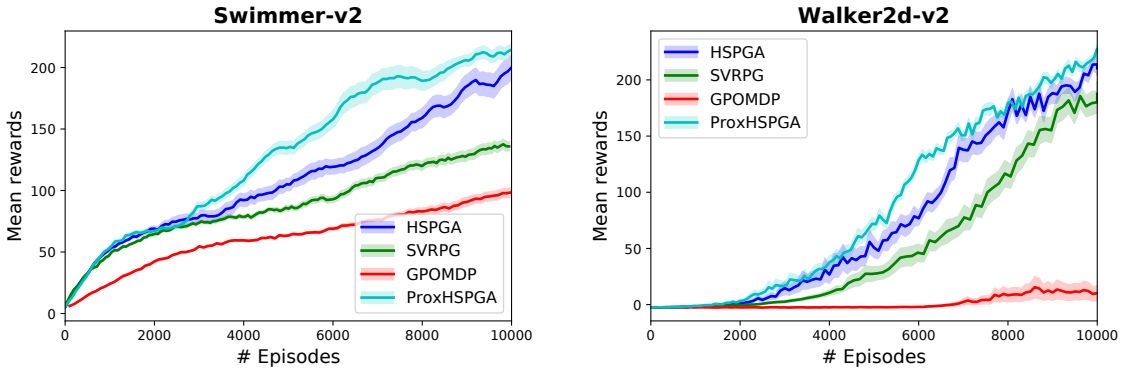
From Figure 3.2b, HSPGA also shows better performance than SVRPG and while GPOMDP appears to be not as effective as SVRPG.

The effect of regularizers: We test the effect of the regularizer $Q(\cdot)$ by adding a Tikhonov one as

$$\max_{\theta \in \mathbb{R}^q} \left\{ J(\theta) - \lambda \|\theta\|_2^2 \right\}.$$

This model was intensively studied in Liu et al. (2019).

We also compare all non-composite algorithms with ProxHSPGA in two continuous control tasks in Mujoco: `Swimmer-v2` and `Walker2d-v2`. In this experiment, we set the penalty parameter $\lambda = 0.001$ for ProxHSPGA. The performance of four algorithms running on these environments are illustrated in Figure 3.3.



(a) (b)

Figure 3.3: Performance of composite vs. non-composite algorithms on the `Swimmer-v2` and `Walker2d-v2` environments.

Again, Figure 3.3 shows similar pattern for non-composite algorithms while HSPGA works better than SVRPG and GPOMDP is the slowest. It also reveals the benefit of adding a regularizer, which potentially gains more reward than without using regularizer as ProxHSPGA seems to achieve higher mean rewards than HSPGA. We believe that the choice of regularizer is also critical and may lead to different performance. We refer to Liu et al. (2019) for more evidence of using regularizers in reinforcement learning.

3.5 Proofs of technical results

This section presents the missing proofs of technical results presented in Section 3.3.

3.5.1 Proof of Lemma 3.2

Proof. Part of this proof comes from the proof of Lemma 1 in Tran-Dinh et al. (2019b). Let $\mathbb{E}_{\mathcal{B},\widehat{\mathcal{B}}}[\cdot] := \mathbb{E}_{\tau,\hat{\tau}\sim p_{\theta_t}}[\cdot]$ be the total expectation. Using the independence of τ and $\hat{\tau}$, taking the total expectation on (3.4), we obtain

$$\mathbb{E}_{\mathcal{B},\widehat{\mathcal{B}}}[v_t] \;= \beta v_{t-1} + \beta\left[\nabla J(\theta_t) - \nabla J(\theta_{t-1})\right] + (1-\beta)\nabla J(\theta_t) = \nabla J(\theta_t) + \beta\left[v_{t-1} - \nabla J(\theta_{t-1}\right],$$

which is the same as (3.5).

To prove (3.6), we first define $u_t := \frac{1}{B}\sum\limits_{\hat{\tau}\in\widehat{\mathcal{B}}_t} g(\hat{\tau}|\theta_t)$ and $\Delta u_t := u_t - \nabla J(\theta_t)$. We have

$$\begin{aligned}
\|\Delta v_t\|^2 \;=\;& \beta^2\|\Delta v_{t-1}\|^2 + \tfrac{\beta^2}{B^2}\Big\|\sum_{\tau\in\mathcal{B}_t}\Delta g(\tau|\theta_t)\Big\|^2 + (1-\beta)^2\|\Delta u_t\|^2 + \beta^2\|\nabla J(\theta_{t-1}) - \nabla J(\theta_t)\|^2\\
&+ \tfrac{2\beta^2}{B}\sum_{\tau\in\mathcal{B}_t}(\Delta v_{t-1})^\top[\Delta g(\tau|\theta_t)] + 2\beta^2(\Delta v_{t-1})^\top[\nabla J(\theta_{t-1}) - \nabla J(\theta_t)]\\
&+ 2\beta(1-\beta)(\Delta v_{t-1})^\top[u_t - \nabla J(\theta_t)] + \tfrac{2\beta(1-\beta)}{B}\sum_{\tau\in\mathcal{B}_t}[\Delta g(\tau|\theta_t)]^\top(\Delta u_t)\\
&+ \tfrac{2\beta^2}{B}\sum_{\tau\in\mathcal{B}_t}(\Delta g(\tau|\theta_t))^\top[\nabla J(\theta_{t-1}) - \nabla J(\theta_t)]\\
&+ 2\beta(1-\beta)(\Delta u_t)^\top[\nabla J(\theta_{t-1}) - \nabla J(\theta_t)].
\end{aligned}$$

Taking the total expectation and note that

$$\begin{aligned}
&\mathbb{E}_{\widehat{\mathcal{B}}}[u_t] := \mathbb{E}_{\hat{\tau}\sim p_{\theta_t}}[u_t] = \nabla J(\theta_t) \quad \text{and}\\
&\mathbb{E}_{\widehat{\mathcal{B}}}\left[\|u_t - \nabla J(\theta_t)\|^2\right] \le \tfrac{1}{B^2}\sum_{\hat{\tau}\in\widehat{\mathcal{B}}}\mathbb{E}\left[\|g(\hat{\tau}|\theta_t) - \nabla J(\theta_t)\|^2\right] \le \frac{\sigma^2}{B},
\end{aligned}$$

we get

$$\begin{aligned}
\mathbb{E}_{\mathcal{B},\widehat{\mathcal{B}}}\left[\|\Delta v_t\|^2\right] \;=\;& \beta^2\|\Delta v_{t-1}\|^2 + \tfrac{\beta^2}{B^2}\mathbb{E}_{\mathcal{B}}\left[\Big\|\sum_{\tau\in\mathcal{B}_t}\Delta g(\tau|\theta_t)\Big\|^2\right] + (1-\beta)^2\mathbb{E}_{\widehat{\mathcal{B}}}\left[\|\Delta u_t\|^2\right]\\
&- \beta^2\|\nabla J(\theta_{t-1}) - \nabla J(\theta_t)\|^2.
\end{aligned}$$

Using triangle inequality, we obtain

$$\begin{aligned}\mathbb{E}_{\mathcal{B},\widehat{\mathcal{B}}}\left[\|\Delta v_t\|^2\right] &\leq \beta^2\|\Delta v_{t-1}\|^2 + \tfrac{\beta^2}{B^2}\sum_{\tau\in\mathcal{B}_t}\mathbb{E}_{\mathcal{B}}\left[\|\Delta g(\tau|\theta_t)\|^2\right] \\ &\quad - \beta^2\|\nabla J(\theta_{t-1}) - \nabla J(\theta_t)\|^2 + \tfrac{(1-\beta)^2\sigma^2}{B} \\ &\leq \beta^2\|\Delta v_{t-1}\|^2 + \tfrac{\beta^2}{B^2}\sum_{\tau\in\mathcal{B}_t}\mathbb{E}_{\mathcal{B}}\left[\|\Delta g(\tau|\theta_t)\|^2\right] + \frac{(1-\beta)^2}{B}\sigma^2,\end{aligned} \tag{3.10}$$

where we ignore the non-negative terms to arrive at the second inequality.

Additionally, Lemma 6.1 in Xu et al. (2019a) shows that

$$\mathrm{Var}\left[\omega(\tau|\theta_t,\theta_{t-1})\right] \leq C_\omega\left\|\theta_t - \theta_{t-1}\right\|^2, \tag{3.11}$$

where $C_\omega := H(2HG^2 + M)(W+1)$.

Using (3.11) we can bound $\mathbb{E}_{\mathcal{B}}\left[\|\Delta g(\tau|\theta_t)\|^2\right]$ as

$$\begin{aligned}\mathbb{E}_{\mathcal{B}}\left[\|\Delta g(\tau|\theta_t)\|^2\right] &= \mathbb{E}_{\mathcal{B}}\left[\|g(\tau|\theta_t) - \omega(\tau|\theta_t,\theta_{t-1})g(\tau|\theta_{t-1})\|^2\right] \\ &= \mathbb{E}_{\mathcal{B}}\left[\|[1-\omega(\tau|\theta_t,\theta_{t-1})]g(\tau|\theta_{t-1}) + (g(\tau|\theta_t) - g(\tau|\theta_{t-1})\|^2\right] \\ &\leq \mathbb{E}_{\mathcal{B}}\left[\|[1-\omega(\tau|\theta_t,\theta_{t-1})]g(\tau|\theta_{t-1})\|^2\right] + \mathbb{E}_{\mathcal{B}}\left[\|g(\tau|\theta_t) - g(\tau|\theta_{t-1})\|^2\right] \\ &\overset{(\star)}{\leq} C_g^2\mathbb{E}_{\mathcal{B}}\left[\|1-\omega(\tau|\theta_t,\theta_{t-1})\|^2\right] + L_g^2\left\|\theta_t - \theta_{t-1}\right\|^2 \\ &\overset{(\star\star)}{=} C_g^2\mathrm{Var}\left[\omega(\tau|\theta_t,\theta_{t-1})\right] + L_g^2\left\|\theta_t - \theta_{t-1}\right\|^2 \\ &\overset{(3.11)}{\leq} \left(C_g^2 C_\omega + L_g^2\right)\left\|\theta_t - \theta_{t-1}\right\|^2,\end{aligned}$$

where $L_g := \frac{HM(R+|b|)}{(1-\gamma)}$, $C_g := \frac{HG(R+|b|)}{(1-\gamma)}$, and b is a baseline reward. Here, $(\star)$ comes from Lemma 3.1 and $(\star\star)$ is from Cortes et al. (2010, Lemma 1).

Plugging the last estimate into (3.10) yields

$$\mathbb{E}_{\mathcal{B},\widehat{\mathcal{B}}}\left[\|\Delta v_t\|^2\right] \leq \beta^2\|\Delta v_{t-1}\|^2 + \frac{\beta^2(C_g^2 C_\omega + L_g^2)}{B}\left\|\theta_t - \theta_{t-1}\right\|^2 + \frac{(1-\beta)^2}{B}\sigma^2,$$

which is (3.6), where $\overline{C} := C_g^2 C_\omega + L_g^2$. □

3.5.2 Proof of Lemma 3.3

Proof. Similar to the proof of Lemma 5 in Tran-Dinh et al. (2019b) , from the update in Algorithm 2, we have $\theta_{t+1} = (1-\gamma)\theta_t + \gamma\widehat{\theta}_{t+1}$, which leads to $\theta_{t+1} - \theta_t = \gamma(\widehat{\theta}_{t+1} - \theta_t)$. Combining this expression and the L-smoothness of $J(\theta)$ in Lemma 3.1, we have

$$\begin{aligned} J(\theta_{t+1}) &\geq J(\theta_t) + [\nabla J(\theta_t)]^\top (\theta_{t+1} - \theta_t) - \tfrac{L}{2}\|\theta_{t+1} - \theta_t\|^2 \\ &= J(\theta_t) + \alpha\,[\nabla J(\theta_t)]^\top (\widehat{\theta}_{t+1} - \theta_t) - \tfrac{L\alpha^2}{2}\|\widehat{\theta}_{t+1} - \theta_t\|^2. \end{aligned} \tag{3.12}$$

From the convexity of Q, we have

$$Q(\theta_{t+1}) \leq (1-\alpha)Q(\theta_t) + \alpha Q(\widehat{\theta}_{t+1}) \leq Q(\theta_t) + \alpha \nabla Q(\widehat{\theta}_{t+1})^\top (\widehat{\theta}_{t+1} - \theta_t), \tag{3.13}$$

where $\nabla Q(\widehat{\theta}_{t+1})$ is a subgradient of Q at $\widehat{\theta}_{t+1}$.

By the optimality condition of $\widehat{\theta}_{t+1} = \text{prox}_{\eta Q}(\theta_t + \eta v_t)$, we can show that $\nabla Q(\widehat{\theta}_{t+1}) = v_t - \frac{1}{\eta}(\widehat{\theta}_{t+1} - \theta_t)$ for some $\nabla Q(\widehat{\theta}_{t+1}) \in \partial Q(\widehat{\theta}_{t+1})$ where ∂Q is the subdifferential of Q at $\widehat{\theta}_{t+1}$. Plugging this into (3.13), we get

$$Q(\theta_{t+1}) \leq Q(\theta_t) + \alpha v_t^\top (\widehat{\theta}_{t+1} - \theta_t) - \frac{\alpha}{\eta}\|\widehat{\theta}_{t+1} - \theta_t\|^2. \tag{3.14}$$

Subtracting (3.14) from (3.12), we obtain

$$\begin{aligned} F(\theta_{t+1}) &\geq F(\theta_t) + \alpha\,[\nabla J(\theta_t) - v_t]^\top (\widehat{\theta}_{t+1} - \theta_t) + \left(\tfrac{\alpha}{\eta} - \tfrac{L\alpha^2}{2}\right)\|\widehat{\theta}_{t+1} - \theta_t\|^2 \\ &= F(\theta_t) - \alpha\,[v_t - \nabla J(\theta_t)]^\top (\widehat{\theta}_{t+1} - \theta_t) + \left(\tfrac{\alpha}{\eta} - \tfrac{L\alpha^2}{2}\right)\|\widehat{\theta}_{t+1} - \theta_t\|. \end{aligned} \tag{3.15}$$

Using the fact that

$$\begin{aligned} [v_t - \nabla J(\theta_t)]^\top (\widehat{\theta}_{t+1} - \theta_t) &= \tfrac{1}{2}\|v_t - \nabla J(\theta_t)\|^2 + \tfrac{1}{2}\|\widehat{\theta}_{t+1} - \theta_t\|^2 \\ &\quad - \tfrac{1}{2}\|v_t - \nabla J(\theta_t) - (\widehat{\theta}_{t+1} - \theta_t)\|^2, \end{aligned}$$

and ignoring the non-negative term $\frac{1}{2}\|v_t - \nabla J(\theta_t) - (\widehat{\theta}_{t+1} - \theta_t)\|^2$, we can rewrite (3.15) as

$$F(\theta_{t+1}) \geq F(\theta_t) - \frac{\alpha}{2}\|\nabla J(\theta_t) - v_t\|^2 + \left(\frac{\alpha}{\eta} - \frac{L\alpha^2}{2} - \frac{\alpha}{2}\right)\|\widehat{\theta}_{t+1} - \theta_t\|^2.$$

Taking the total expectation over the entire history $\mathcal{F}_{t+1}$, we obtain

$$\mathbb{E}\left[F(\theta_{t+1})\right] \geq \mathbb{E}\left[F(\theta_t)\right] - \frac{\alpha}{2}\mathbb{E}\left[\|\nabla J(\theta_t) - v_t\|^2\right] + \left(\frac{\alpha}{\eta} - \frac{L\alpha^2}{2} - \frac{\alpha}{2}\right)\mathbb{E}\left[\|\widehat{\theta}_{t+1} - \theta_t\|^2\right]. \quad (3.16)$$

From the definition of the gradient mapping (3.3), we have

$$\eta\|\mathcal{G}_\eta(\theta_t)\| = \|\mathrm{prox}_{\eta Q}(\theta_t + \eta\nabla J(\theta_t)) - \theta_t\|.$$

Applying the triangle inequality, we can derive

$$\begin{aligned}
\eta\left\|\mathcal{G}_\eta(\theta_t)\right\| &\leq \|\widehat{\theta}_{t+1} - \theta_t\| + \|\mathrm{prox}_{\eta Q}(\theta_t + \eta\nabla J(\theta_t)) - \widehat{\theta}_{t+1}\| \\
&= \|\widehat{\theta}_{t+1} - \theta_t\| + \|\mathrm{prox}_{\eta Q}(\theta_t + \eta\nabla J(\theta_t)) - \mathrm{prox}_{\eta Q}(\theta_t + \eta v_t)\| \\
&\leq \|\widehat{\theta}_{t+1} - \theta_t\| + \eta\|v_t - \nabla J(\theta_t)\|.
\end{aligned}$$

Taking the full expectation over the entire history $\mathcal{F}_{t+1}$ yields

$$\eta^2\mathbb{E}\left[\mathcal{G}_\eta(\theta_t)\right]^2 \leq 2\mathbb{E}\left[\|\widehat{\theta}_{t+1} - \theta_t\|^2\right] + 2\eta^2\mathbb{E}\left[\|v_t - \nabla J(\theta_t)\|^2\right].$$

Multiply this inequality by $-\frac{\alpha}{2}$ and add to (3.16), we arrive at

$$\begin{aligned}
\mathbb{E}\left[F(\theta_{t+1})\right] \;\geq\; & \mathbb{E}\left[F(\theta_t)\right] + \tfrac{\eta^2\alpha}{2}\mathbb{E}\left[\|\mathcal{G}_\eta(\theta_t)\|^2\right] - \tfrac{\alpha}{2}\left(1 + 2\eta^2\right)\mathbb{E}\left[\|v_t - \nabla J(\theta_t)\|^2\right] \\
& + \tfrac{\alpha}{2}\left(\tfrac{2}{\eta} - L\alpha - 3\right)\mathbb{E}\left[\|\widehat{\theta}_{t+1} - \theta_t\|^2\right],
\end{aligned}$$

which can be rewritten as

$$\mathbb{E}\left[F(\theta_{t+1})\right] \geq \mathbb{E}\left[F(\theta_t)\right] + \tfrac{\eta^2\alpha}{2}\mathbb{E}\left[\|\mathcal{G}_\eta(\theta_t)\|^2\right] - \tfrac{\xi}{2}\mathbb{E}\left[\|v_t - \nabla J(\theta_t)\|^2\right] + \tfrac{\zeta}{2}\mathbb{E}\left[\|\widehat{\theta}_{t+1} - \theta_t\|^2\right],$$

where $\xi := \alpha(1 + 2\eta^2)$ and $\zeta := \alpha\left(\frac{2}{\eta} - L\alpha - 3\right)$ which is exactly (3.7). □

3.5.3 Proof of Theorem 3.1: Key bound on the gradient mapping

Proof. Firstly, using the identity $\theta_{t+1} - \theta_t = \gamma(\widehat{\theta}_{t+1} - \theta_t)$, taking the total expectation over the entire history $\mathcal{F}_{t+1}$, we can rewrite (3.6) as

$$\begin{aligned}\mathbb{E}\left[\|v_{t+1} - \nabla J(\theta_{t+1})\|^2\right] &\leq \beta^2 \mathbb{E}\left[\|v_t - \nabla J(\theta_t)\|^2\right] + \tfrac{\beta^2 \overline{C}}{B}\mathbb{E}\left[\|\theta_{t+1} - \theta_t\|^2\right] + \tfrac{(1-\beta)^2}{B}\sigma^2 \\ &= \beta^2 \mathbb{E}\left[\|v_t - \nabla J(\theta_t)\|^2\right] + \tfrac{\beta^2 \overline{C}\alpha^2}{B}\mathbb{E}\left[\|\widehat{\theta}_{t+1} - \theta_t\|^2\right] + \tfrac{(1-\beta)^2}{B}\sigma^2.\end{aligned} \tag{3.17}$$

Multiply (3.17) by $-\dfrac{\kappa}{2}$ for some $\kappa > 0$, then add to (3.7), we have

$$\begin{aligned}& \mathbb{E}\left[F(\theta_{t+1})\right] - \tfrac{\kappa}{2}\mathbb{E}\left[\|v_{t+1} - \nabla J(\theta_{t+1})\|^2\right] \\ \geq\ & \mathbb{E}\left[F(\theta_t)\right] - \tfrac{(\kappa\beta^2 + \xi)}{2}\mathbb{E}\left[\|v_t - \nabla J(\theta_t)\|^2\right] + \tfrac{\eta^2\alpha}{2}\mathbb{E}\left[\|\mathcal{G}_\eta(\theta_t)\|^2\right] + \tfrac{1}{2}\left(\zeta - \tfrac{\kappa\beta^2\overline{C}\alpha^2}{B}\right)\mathbb{E}\left[\|\widehat{\theta}_{t+1} - \theta_t\|^2\right] \\ & - \tfrac{\kappa(1-\beta^2)\sigma^2}{2B} \\ =\ & \mathbb{E}\left[F(\theta_t)\right] - \tfrac{\kappa}{2}\mathbb{E}\left[\|v_t - \nabla J(\theta_t)\|^2\right] + \tfrac{\eta^2\alpha}{2}\mathbb{E}\left[\|\mathcal{G}_\eta(\theta_t)\|^2\right] - \tfrac{[\xi - \kappa(1-\beta^2)]}{2}\mathbb{E}\left[\|v_t - \nabla J(\theta_t)\|^2\right] \\ & + \tfrac{1}{2}\left(\zeta - \tfrac{\kappa\beta^2\overline{C}\alpha^2}{B}\right)\mathbb{E}\left[\|\widehat{\theta}_{t+1} - \theta_t\|^2\right] - \tfrac{\kappa(1-\beta^2)\sigma^2}{2B}.\end{aligned}$$

Let us define $\overline{F}(\theta_t) := \mathbb{E}\left[F(\theta_t)\right] - \frac{\kappa}{2}\mathbb{E}\left[\|v_t - \nabla J(\theta_{t+1})\|^2\right]$. Then, the last inequality can be written as

$$\begin{aligned}\overline{F}(\theta_{t+1}) \ \geq\ & \overline{F}(\theta_t) + \tfrac{\eta^2\alpha}{2}\mathbb{E}\left[\|\mathcal{G}_\eta(\theta_t)\|^2\right] - \tfrac{[\xi - \kappa(1-\beta^2)]}{2}\mathbb{E}\left[\|v_t - \nabla J(\theta_t)\|^2\right] \\ & - \tfrac{\kappa(1-\beta^2)\sigma^2}{2B} + \tfrac{1}{2}\left(\zeta - \tfrac{\kappa\beta^2\overline{C}\alpha^2}{B}\right)\mathbb{E}\left[\|\widehat{\theta}_{t+1} - \theta_t\|^2\right].\end{aligned} \tag{3.18}$$

Suppose that η, α, β are chosen such that

$$\frac{2}{\eta} - L\alpha - 3 \geq \frac{\kappa\beta^2\overline{C}\alpha}{B} > 0 \quad \text{and} \quad \alpha(1 + 2\eta^2) \leq \kappa(1 - \beta^2). \tag{3.19}$$

Then, we have $\zeta \geq \dfrac{\kappa\beta^2\overline{C}\alpha^2}{B}$ and $\xi \leq \kappa(1-\beta^2)$. By ignoring the non-negative terms in (3.18), we can rewrite it as

$$\overline{F}(\theta_{t+1}) \geq \overline{F}(\theta_t) + \frac{\eta^2\alpha}{2}\mathbb{E}\left[\|\mathcal{G}_\eta(\theta_t)\|^2\right] - \frac{\kappa(1-\beta^2)\sigma^2}{2B}.$$

Summing the above inequality for $t = 0, \cdots, m$, we obtain

$$\overline{F}(\theta_{m+1}) \geq \overline{F}(\theta_0) + \frac{\eta^2\alpha}{2}\sum_{t=0}^{m}\mathbb{E}\left[\|\mathcal{G}_\eta(\theta_t)\|^2\right] - \frac{\kappa(m+1)(1-\beta^2)\sigma^2}{2B}. \tag{3.20}$$

Rearranging terms and multiply both sides by $\dfrac{2}{\eta^2\alpha}$, (3.20) becomes

$$\sum_{t=0}^{m}\mathbb{E}\left[\|\mathcal{G}_\eta(\theta_t)\|^2\right] \leq \frac{2}{\eta^2\alpha}\left[\overline{F}(\theta_{m+1}) - \overline{F}(\theta_0)\right] + \frac{\kappa(m+1)(1-\beta^2)\sigma^2}{\eta^2\alpha B}. \tag{3.21}$$

Note that

$$\overline{F}(\theta_0) = F(\theta_0) - \frac{\kappa}{2}\mathbb{E}\left[\|v_0 - \nabla J(\theta_0)\|^2\right] \geq F(\theta_0) - \frac{\kappa\sigma^2}{2N},$$

and $\overline{F}(\theta_{m+1}) = F(\theta_{m+1}) - \frac{\kappa}{2}\mathbb{E}\left[\|v_{m+1} - \nabla J(\theta_{m+1})\|^2\right] \leq F(\theta_{m+1})$. Using these estimate in (3.21), we obtain

$$\begin{aligned}\sum_{t=0}^{m}\mathbb{E}\left[\|\mathcal{G}_\eta(\theta_t)\|^2\right] &\leq \frac{2}{\eta^2\alpha}\left[F(\theta_{m+1}) - F(\theta_0)\right] + \frac{\kappa\sigma^2}{\eta^2\alpha N} + \frac{\kappa(m+1)(1-\beta^2)\sigma^2}{\eta^2\alpha B} \\ &= \frac{2}{\eta^2\alpha}\left[F(\theta_{m+1}) - F(\theta_0)\right] + \frac{(m+1)\kappa\sigma^2}{\eta^2\alpha}\left[\frac{1}{N(m+1)} + \frac{(1-\beta^2)}{B}\right].\end{aligned}$$

Multiplying both sides by $\frac{1}{m+1}$, we have

$$\begin{aligned}\tfrac{1}{m+1}\sum_{t=0}^{m}\mathbb{E}\left[\|\mathcal{G}_\eta(\theta_t)\|^2\right] \;\leq\; & \tfrac{2}{\eta^2\alpha(m+1)}\left[F(\theta_{m+1}) - F(\theta_0)\right] \\ & + \tfrac{\kappa\sigma^2}{\eta^2\alpha}\left[\tfrac{1}{N(m+1)} + \tfrac{(1-\beta^2)}{B}\right].\end{aligned} \tag{3.22}$$

Now we choose $\beta := 1 - \frac{\sqrt{B}}{\sqrt{N(m+1)}}$ so that the right-hand side of (3.22) is minimized. Note that if $1 \leq B \leq N(m+1)$, then $\beta \in [0, 1)$.

Let us choose $\eta := \frac{2}{4+L\alpha} \leq \frac{1}{2}$ which means $\zeta := \frac{2}{\eta} - L\alpha - 3 = 1$. We can satisfy the first condition of (3.19) by choosing $0 < \alpha \leq \frac{B}{\kappa C}$.

Besides, the second condition in (3.19) holds if $0 < \alpha \leq \frac{\kappa(1-\beta^2)}{1+2\eta^2}$. Since we have $\eta \leq \frac{1}{2}$ which leads to $1 + 2\eta^2 \leq \frac{3}{2}$ and using $1 - \beta^2 \geq 1 - \beta = \frac{B^{1/2}}{N^{1/2}(m+1)^{1/2}}$ we derive the condition for α as

$$0 < \alpha \leq \frac{2\kappa\sqrt{B}}{3\sqrt{N(m+1)}}.$$

Therefore, the overall condition for α is given as

$$0 < \alpha \leq \min\left\{1, \frac{B}{\kappa\overline{C}}, \frac{2\kappa\sqrt{B}}{3\sqrt{N(m+1)}}\right\}.$$

If we choose $\kappa := \frac{\sqrt{3}[NB(m+1)]^{1/4}}{\sqrt{2\overline{C}}}$, then we can update α as

$$\alpha := \frac{\hat{c}\sqrt{2}B^{3/4}}{\sqrt{3\overline{C}}[N(m+1)]^{1/4}}. \tag{3.23}$$

Using $1 \leq B \leq N(m+1)$, we can bound $\alpha \leq \hat{c}\sqrt{\frac{2B}{3\overline{C}}}$ then we can choose $\hat{c} \in \left(0, \sqrt{\frac{3\overline{C}}{2B}}\right]$ so that $\gamma \in (0, 1]$.

With all the choices of β, η, α, and κ above, if we let the output $\tilde{\theta}_T$ be selected uniformly at random from $\{\theta_t\}_{t=0}^m$, then we have

$$\begin{aligned}\mathbb{E}\left[\|\mathcal{G}_\eta(\tilde{\theta}_T)\|^2\right] &= \frac{1}{m+1}\sum_{t=0}^{m}\mathbb{E}\left[\|\mathcal{G}_\eta(\theta_t)\|^2\right] \\ &\leq \frac{\sqrt{3\overline{C}}N^{1/4}}{\eta^2\hat{c}\sqrt{2}[B(m+1)]^{3/4}}\left[F(\theta_{m+1}) - F(\theta_0)\right] + \frac{3\sigma^2}{\eta^2[BN(m+1)]^{1/2}}.\end{aligned} \tag{3.24}$$

Note that $\eta = \frac{2}{4+L\alpha}$ and since $\alpha \leq 1$ we have $\frac{1}{\eta^2} \leq \frac{(4+L)^2}{4}$. Plugging these into (3.24) yields

$$\begin{aligned}\mathbb{E}\left[\|\mathcal{G}_\eta(\tilde{\theta}_T)\|^2\right] &= \frac{1}{m+1}\sum_{t=0}^{m}\mathbb{E}\left[\|\mathcal{G}_\eta(\theta_t)\|^2\right] \\ &\leq \frac{(4+L)^2\sqrt{3\overline{C}}N^{1/4}}{4\hat{c}\sqrt{2}[B(m+1)]^{3/4}}\left[F(\theta_{m+1}) - F(\theta_0)\right] + \frac{3(4+L)^2\sigma^2}{4[BN(m+1)]^{1/2}} \\ &\leq \frac{(4+L)^2\sqrt{3\overline{C}}N^{1/4}}{4\hat{c}\sqrt{2}[B(m+1)]^{3/4}}\left[F^* - F(\theta_0)\right] + \frac{3(4+L)^2\sigma^2}{4[BN(m+1)]^{1/2}},\end{aligned}$$

where we use the fact that $F(\theta_{m+1}) \leq F^*$. □

3.5.4 Proof of Corollary 3.1: trajectory complexity of Algorithms 2 and 3

Proof. If we fix a batch size $B \in \mathbb{N}_+$ and choose $N := \tilde{c}\sigma^{8/3}\left[B(m+1)\right]^{1/3}$ for some $\tilde{c} > 0$, (3.5.3) is equivalent to

$$\begin{aligned}
\mathbb{E}\left[\|\mathcal{G}_\eta(\tilde{\theta}_T)\|^2\right] &\leq \frac{(4+L)^2\sqrt{3\overline{C}}\tilde{c}^{1/4}\sigma^{2/3}}{4\hat{c}\sqrt{2}[B(m+1)]^{2/3}}\left[F^* - F(\overline{\theta}^{(0)})\right] + \frac{3(4+L)^2\sigma^{2/3}}{4\tilde{c}^{1/2}[B(m+1)]^{2/3}} \\
&= \left[\frac{(4+L)^2\sqrt{3\overline{C}}\tilde{c}^{1/4}}{4\hat{c}\sqrt{2}}\left[F^* - F(\overline{\theta}^{(0)})\right] + \frac{3(4+L)^2}{4\tilde{c}^{1/2}}\right]\frac{\sigma^{2/3}}{[B(m+1)]^{2/3}} \\
&= \frac{\Psi_0\sigma^{2/3}}{[B(m+1)]^{2/3}},
\end{aligned}$$

where we define

$$\Psi_0 := \left[\frac{(4+L)^2\sqrt{3\overline{C}}\tilde{c}^{1/4}}{4\hat{c}\sqrt{2}}\left[F^* - F(\overline{\theta}^{(0)})\right] + \frac{3(4+L)^2}{4\tilde{c}^{1/2}}\right]. \tag{3.25}$$

Therefore, for any $\varepsilon > 0$, to guarantee $\mathbb{E}\left[\|\mathcal{G}_\eta(\tilde{\theta}_T)\|^2\right] \leq \varepsilon^2$, we need $\frac{\Psi_0\sigma^{2/3}}{[B(m+1)]^{2/3}} = \varepsilon^2$ which leads to the total number of iterations

$$T = m + 1 = \frac{\Psi_0^{3/2}\sigma}{B\varepsilon^3} = \mathcal{O}\left(\frac{1}{\varepsilon^3}\right).$$

The total number of proximal operations $\mathrm{prox}_{\eta Q}$ is also $\mathcal{O}\left(\frac{1}{\varepsilon^3}\right)$. In addition, the total number of trajectories is at most

$$\begin{aligned}
N + 2B(m+1) &= \tilde{c}\sigma^{8/3}\left[B(m+1)\right]^{1/3} + \frac{2\Psi_0\sigma}{\varepsilon^3} \\
&= \tilde{c}\sigma^{8/3}\frac{\Psi_0^{1/3}\sigma 1/3}{\varepsilon} + \frac{2\Psi_0\sigma}{\varepsilon^3} \\
&= \mathcal{O}\left(\frac{1}{\varepsilon} + \frac{1}{\varepsilon^3}\right) = \mathcal{O}\left(\frac{1}{\varepsilon^3}\right).
\end{aligned}$$

This proves our the complexity of Algorithm 2.

Next, let us denote the superscript $^{(s)}$ when the current stage is s for $s = 0, \cdots, S-1$. Note that from the first inequality of (3.5.3), for any stage $s = 0, \ldots, S-1$, the following holds

$$\frac{1}{m+1}\sum_{t=0}^{m}\mathbb{E}\left[\|\mathcal{G}_\eta(\theta_t^{(s)})\|^2\right] \leq \frac{(4+L)^2\sqrt{3\overline{C}}N^{1/4}}{4\hat{c}\sqrt{2}[B(m+1)]^{3/4}}\left[F(\theta_{m+1}^{(s)}) - F(\theta_0^{(s)})\right] + \frac{3(4+L)^2\sigma^2}{4[BN(m+1)]^{1/2}}.$$

Summing for $s = 0, \cdots, S-1$ and multiply both sides by $\dfrac{1}{S}$ yields

$$\begin{aligned}\frac{1}{S(m+1)} \sum_{s=0}^{S-1} \sum_{t=0}^{m} \mathbb{E}\left[\|\mathcal{G}_\eta(\theta_t^{(s)})\|^2\right] &\leq \frac{(4+L)^2\sqrt{3\overline{C}}N^{1/4}}{4\hat{c}\sqrt{2}[B(m+1)]^{3/4}S}\left[F(\theta_{m+1}^{(S-1)}) - F(\theta_0^{(0)})\right] + \frac{3(4+L)^2\sigma^2}{4[BN(m+1)]^{1/2}S} \\ &\leq \frac{(4+L)^2\sqrt{3\overline{C}}N^{1/4}}{4\hat{c}\sqrt{2}[B(m+1)]^{3/4}S}\left[F^* - F(\theta_0^{(0)})\right] + \frac{3(4+L)^2\sigma^2}{4[BN(m+1)]^{1/2}S},\end{aligned} \tag{3.26}$$

where we use $F(\theta_{m+1}^{(S-1)}) \leq F^*$ again.

If we also fix a batch size $B \in \mathbb{N}_+$ and choose $N := \tilde{c}\sigma^{8/3}\left[B(m+1)\right]^{1/3}$ for some $\tilde{c} > 0$, and select $\tilde{\theta}_T$ uniformly random from $\{\theta_t^{(s)}\}_{t=0,\cdots,m}^{s=1,\cdots,S}$, then, similar to (3.5.4), (3.26) can be written as

$$\begin{aligned}\mathbb{E}\left[\|\mathcal{G}_\eta(\tilde{\theta}_T)\|^2\right] &= \frac{1}{S(m+1)} \sum_{s=0}^{S-1} \sum_{t=0}^{m} \mathbb{E}\left[\|\mathcal{G}_\eta(\theta_t^{(s)})\|^2\right] \\ &\leq \frac{(4+L)^2\sqrt{3\overline{C}}\tilde{c}^{1/4}\sigma^{2/3}}{4\hat{c}\sqrt{2}[B(m+1)]^{2/3}S}\left[F^* - F(\theta_0^{(0)})\right] + \frac{3(4+L)^2\sigma^{2/3}}{4\tilde{c}^{1/2}[B(m+1)]^{2/3}S} \\ &= \left[\frac{(4+L)^2\sqrt{3\overline{C}}\tilde{c}^{1/4}}{4\hat{c}\sqrt{2}}\left[F^* - F(\theta_0^{(0)})\right] + \frac{3(4+L)^2}{4\tilde{c}^{1/2}}\right]\frac{\sigma^{2/3}}{[B(m+1)]^{2/3}S} \\ &\leq \frac{\Psi_0\sigma^{2/3}}{[SB(m+1)]^{2/3}},\end{aligned}$$

where we use Ψ_0 defined in (3.25) and $\frac{1}{S} \leq \frac{1}{S^{2/3}}$ for any $S \geq 1$.

Therefore, to guarantee $\mathbb{E}\left[\|\mathcal{G}_\eta(\tilde{\theta}_T)\|^2\right] \leq \varepsilon^2$ for any $\varepsilon > 0$, we need $\frac{\Psi_0\sigma^{2/3}}{[SB(m+1)]^{2/3}} = \varepsilon^2$ which leads to the total number of iterations

$$T = S(m+1) = \frac{\Psi_0^{3/2}\sigma}{B\varepsilon^3} = \mathcal{O}\left(\frac{1}{\varepsilon^3}\right).$$

The total number of proximal operations $\mathrm{prox}_{\eta Q}$ is also $\mathcal{O}\left(\frac{1}{\varepsilon^3}\right)$. In addition, the total number of trajectories is at most

$$\begin{aligned}S\left[N + 2B(m+1)\right] &= S\left[\tilde{c}\sigma^{8/3}\left[B(m+1)\right]^{1/3} + \frac{2\Psi_0\sigma}{\varepsilon^3}\right] = S\left[\tilde{c}\sigma^{8/3}\frac{\Psi_0^{1/3}\sigma 1/3}{\varepsilon} + \frac{2\Psi_0\sigma}{\varepsilon^3}\right] \\ &= \mathcal{O}\left(\frac{1}{\varepsilon} + \frac{1}{\varepsilon^3}\right) = \mathcal{O}\left(\frac{1}{\varepsilon^3}\right), \text{ for any } S \geq 1.\end{aligned}$$

Therefore, we obtain the conclusion of Corollary 3.1. □

CHAPTER 4

FedDR - Douglas-Rachford Splitting Methods for Federated Learning

4.1 Introduction

Since first introduced in Konečnỳ et al. (2016); McMahan and Ramage (2017), federated learning (FL) has received tremendous attention in the past few years. As the size of datasets and models grow larger, training machine learning model in a centralized fashion becomes more challenging and somewhat inaccessible for a large number of workers. Consequently, training machine learning models using distributed approach comes in as a natural replacement. As the mobile devices have been getting more powerful, they can act as workers to participate in training machine learning models.

In FL, a central server coordinate updates across local workers then perform global model update by averaging models return from local workers. When the number of workers may get extremely large, it creates **communication bottleneck** between the server and workers which eventually slows down the training process. Moreover, the local data stored in each local agent may be different in terms of sizes and distribution which poses another challenge: **data heterogeneity**. Another challenge in FL is the variety of workers with different local storage, computational power, and network connectivity participating into the system also creates a major challenge, as known as **system heterogeneity**. This challenge also causes unstable connection between server and local workers, where worker devices may be disconnected from the server during training so we expect that only a subset of the workers can participate in one round of communication.

Another aspect to consider in FL is the data privacy, distributed methods often communicate the gradient of the local workers whereas private data can be exposed from the shared gradient (Zhu et al., 2019). Therefore, FL methods normally send the global model to each worker at the

start of each communication round, each worker will perform its local update and send back the newly updated local model for aggregation.

4.1.1 Problem of interest

The finite-sum structure is one of the most commonly used to formulate the optimization problem in federated learning which can be written as

$$\min_{x\in\mathbb{R}^p}\Big\{f(x)=\frac{1}{n}\sum_{i=1}^{n}f_i(x)\Big\}. \tag{4.1}$$

We additionally consider another convex function in the objective which can represent constraints or regularizers on the model parameters. In particular, we want to solve the following problem

$$\min_{x\in\mathbb{R}^p}\Big\{F(x):=f(x)+g(x)=\frac{1}{n}\sum_{i=1}^{n}f_i(x)+g(x)\Big\}, \tag{4.2}$$

where n is the number of workers, each f_i is a local loss of the i-th worker, which is assumed to be nonconvex and L-smooth (see Assumptions 4.1 and 4.2 below), and g is a proper, closed, and convex regularizer. Apart from these assumptions, we will not make any additional assumption on (4.2). We emphasize that the use of regularizers g has been motivated in several works, including Yuan et al. (2021).

Let $\mathrm{dom}(F):=\{x\in\mathbb{R}^p:F(x)<+\infty\}$ be the domain of F and ∂g be the subdifferential of g (Bauschke and Combettes, 2017). Since (4.2) is nonconvex, we only expect to find a stationary point, which is characterized by the following optimality condition.

Definition 4.1.1. If $0\in\nabla f(x^*)+\partial g(x^*)$, then x^* is called a stationary point of (4.2).

The algorithms for solving (4.2) developed in this chapter will rely on the following assumptions.

Assumption 4.1 (Boundedness from below)**.** $\mathrm{dom}(F)\neq\emptyset$ and $F^\star:=\inf_{x\in\mathbb{R}^p}F(x)>-\infty$.

Assumption 4.2 (L-smoothness)**.** All functions $f_i(\cdot)$ for $i\in[n]:=\{1,\cdots,n\}$ are L-smooth, i.e., f_i is continuously differentiable and there exists $L\in(0,+\infty)$ such that

$$\|\nabla f_i(x)-\nabla f_i(y)\|\leq L\|x-y\|,\quad\forall x,y\in\mathrm{dom}(f_i). \tag{4.3}$$

Assumptions 4.1 and 4.2 are very standard in nonconvex optimization. Assumption 4.1 guarantees the well-definedness of (4.2) and is independent of algorithms. Assuming the same Lipschitz constant L for all f_i is not restrictive since if f_i is L_i-smooth, then by scaling variables of its constrained formulation (see (4.4) in Supp. Doc.), we can get the same Lipschitz constant L of all f_i.

Let us recall the definition of gradient mapping in (1.5). Then, the optimality condition $0 \in \nabla f(x^*) + \partial g(x^*)$ of (4.2) is equivalent to $\mathcal{G}_\eta(x^*) = 0$. However, in practice, we often wish to find an ε-approximate stationary point to (4.2) defined as follows.

Definition 4.1.2. If $\tilde{x} \in \mathrm{dom}(F)$ satisfies $\mathbb{E}\big[\|\mathcal{G}_\eta(\tilde{x})\|^2\big] \leq \varepsilon^2$, $\tilde{x}$ is called an ε-stationary point of (4.2), where the expectation is taken over all randomness generated by the underlying algorithm.

4.1.2 Related work

On one hand, there are research that focus on the system level to discuss suitable infrastructure design for federated learning systems (Bonawitz et al., 2019; Li et al., 2019a). On the other hand, other researchers also consider FL in specific communication setting (Chen et al., 2020; Amiri and Gündüz, 2020). Zhang et al. (2019); Niknam et al. (2020) provide comprehensive surveys of FL for mobile edge and wireless settings.

Meanwhile, there have been extensive research on developing numerical methods to solve (4.1) to reduce the communication cost. One of the first and well-known method to solve (4.1) is Federated Averaging (FedAvg) or LocalSGD. FedAvg's practical performance has been shown in many early works, e.g., Konečnỳ et al. (2016); McMahan et al. (2017); Zhang et al. (2016).Lin et al. (2018) show that local SGD where local workers perform a number of updates before global communication takes place as in FedAvg may offer benefit over minibatch SGD. Similar comparisons between minibatch SGD and local SGD have been done in Woodworth et al. (2020a,b).

Analyzing convergence of FedAvg was very challenging at its early time due to the complexity in its update as well as data heterogeneity. One of the early attempt to show the convergence of FedAvg is Stich (2018) for convex problems under the iid data setting and a set of assumptions. Yu et al. (2019) also consider local SGD in the nonconvex setting. Without using an additional

bounded gradient assumption as in Stich (2018); Yu et al. (2019), Wang and Joshi (2018) improve the complexity for the general nonconvex setting while Haddadpour et al. (2019) use a Polyak-Łojasiewicz (PL) condition to improve FedAvg's convergence results. In heterogeneous data settings, Khaled et al. (2019) analyze local GD, where workers performs gradient descent (GD) updates instead of SGD. The analysis of FedAvg for non-iid data is given in Li et al. (2019b). The analysis of local GD/SGD for nonconvex problems has been studied in Haddadpour and Mahdavi (2019). However, FedAvg might not converge with non-iid data as shown in Pathak and Wainwright (2020); Zhang et al. (2020); Zhao et al. (2018).

FedProx (Li et al., 2020b) is an extension of FedAvg, which deals with heterogeneity in federated networks by introducing a proximal term to the objective in local updates to improve stability. FedProx has been shown to achieve better performance than FedAvg in heterogeneous setting. Another method to deal with data heterogeneity is SCAFFOLD (Karimireddy et al., 2020b) using a control variate to correct the "client-drift" in local update of FedAvg. MIME (Karimireddy et al., 2020a) is another framework that uses control variate to improve FedAvg for heterogeneous settings. However, SCAFFOLD and MIME require to communicate extra information apart from local models. Compared to aforementioned works, our methods deal with nonconvex problems under standard assumptions and with composite settings.

FedSplit (Pathak and Wainwright, 2020) instead employs a Peaceman-Rachford splitting scheme to solve a constrained reformulation of the original problem. In fact, FedSplit can be viewed as a variant of Tseng's splitting scheme (Bauschke and Combettes, 2017) applied to FL. Pathak and Wainwright (2020) show that FedSplit can find a solution of the FL problem under only convexity without imposing any additional assumptions on system or data homogeneity. Zhang et al. (2020) propose FedPD, which is essentially a variant of the standard augmented Lagrangian method in nonlinear optimization. Other algorithms for FL can be found, e.g., in Charles and Konečnỳ (2021); Gorbunov et al. (2021); Haddadpour et al. (2021); Hanzely et al. (2020); Li et al. (2021); Yu et al. (2020).

4.1.3 Our approach and contribution

Our approach relies on nonconvex DR splitting method, which can handle data heterogeneity as discussed in Pathak and Wainwright (2020). While the DR method is classical, its nonconvex

variants have been recently studied e.g., in Dao and Tam (2019); Li and Pong (2016); Themelis and Patrinos (2020). However, the combination of DR and randomized block-coordinate strategy remains limited (Combettes and Eckstein, 2018; Combettes and Pesquet, 2015) even in the convex settings. Alternatively, asynchronous algorithms have been extensively studied in the literature, also for FL, see, e.g., Bertsekas and Tsitsiklis (1989); Peng et al. (2016); Recht et al. (2011). For instance, a recent work (Xie et al., 2019) analyzes an asynchronous variant of FedAvg under bounded delay assumption and constraint on the number of local updates. Stich (2018) propose an asynchronous local SGD to solve convex problems under iid data. However, to our best knowledge, there exists no asynchronous method using DR splitting techniques with convergence guarantee for FL. In addition, most existing algorithms only focus on non-composite settings. Hence, our work here appears to be the first.

Our contribution can be summarized as follows.

(a) We develop a new FL algorithm, called **FedDR** (**Fed**erated **D**ouglas-**R**achford), by combining the well-known DR splitting technique and randomized block-coordinate strategy for the common nonconvex composite optimization problem in FL. Our algorithm can handle nonsmooth convex regularizers and allows inexact evaluation of the underlying proximal operators as in FedProx or FedPD. It also achieves the best known $\mathcal{O}\left(\varepsilon^{-2}\right)$ communication complexity for finding a stationary point under standard assumptions (Assumptions 4.1-4.2), where ε is a given accuracy. More importantly, unlike FedSplit (Pathak and Wainwright, 2020) and FedPD (Zhang et al., 2020), which require full user participation to achieve convergence, our analysis does allow partial participation by selecting a subset of users to perform update at each communication round.

(b) Next, we propose an asynchronous algorithm, **asyncFedDR**, where each worker can asynchronously perform local update and periodically send the update to the server for proximal aggregation. We show that **asyncFedDR** achieves the same communication complexity $\mathcal{O}\left(\varepsilon^{-2}\right)$ as **FedDR** (up to a constant factor) under the same standard assumptions. Our experiments verify that **asyncFedDR** is more advantageous than **FedDR** when workers have heterogeneous computing power.

4.1.4 Outline

The rest of this chapter is organized as follows. Section 4.2 provides procedure to reformulate (4.1) into a constrained optimization problem and introduce our **FedDR** algorithm along with its convergence guarantee. Section 4.3 introduces **asyncFedDR**, an asynchronous variant of FedDR, and presents its convergence results. Section 4.4 illustrates the performance of our proposed algorithms compared with existing methods using various synthetic and real datasets. Finally, Section 4.5 presents intermediate results and missing proofs for convergence guarantee of **FedDR** and **asyncFedDR** algorithms in Section 4.2 and 4.3.

4.2 FedDR algorithm and its convergence analysis

FedSplit (Pathak and Wainwright, 2020) is a related method which uses the Peaceman-Rachford splitting method (Bauschke and Combettes, 2017) to to solve the convex non-composite problem (4.1). FedSplit is able to address the communication bottleneck and data heterogeneity challenges in federated learning. Following this direction, we first derive a new variant of the Douglas-Rachford splitting method, called **FedDR**, to solve the nonconvex composite problem (4.2). Unlike FedSplit, **FedDR** is a randomized block-coordinate method and consider a more general problem (4.2) instead of (4.1).

4.2.1 The derivation of FedDR

Our first step is to recast (4.2) into a constrained reformulation. After that, we apply the classical Douglas-Rachford (DR) splitting scheme to this reformulation. Finally, we randomize its updates to obtain a randomized block-coordinate DR variant.

(a) Constrained reformulation. We can equivalently write (4.2) into the following constrained minimization problem:

$$\begin{cases} \min\limits_{x_1,\cdots,x_n} & \Big\{F(\mathbf{x}) := f(\mathbf{x}) + g(\mathbf{x}) \equiv \dfrac{1}{n}\displaystyle\sum_{i=1}^{n} f_i(x_i) + g(x_1)\Big\} \\ \text{s.t.} & x_2 = x_1,\ x_3 = x_1,\ \cdots, x_n = x_1. \end{cases} \tag{4.4}$$

where $\mathbf{x} := [x_1, x_2, \cdots, x_n]$ concatenates n copies of $x \in \mathbb{R}^p$ in (4.2) such that it forms a column vector in $\mathbb{R}^{np}$. Such duplications are characterized by $x_2 = x_1, x_3 = x_1, \cdots, x_n = x_1$, which define a consensus set $\mathcal{C} := \{\mathbf{x} \in \mathbb{R}^{np} : x_1 = x_2 = \cdots = x_n\}$.

(b) Unconstrained reformulation. Let $\delta_{\mathcal{C}}$ be the indicator function of the consensus set $\mathcal{C}$, i.e. $\delta_{\mathcal{C}}(\mathbf{x}) = 0$ if $\mathbf{x} \in \mathcal{C}$, and $\delta_{\mathcal{C}}(\mathbf{x}) = +\infty$, otherwise. Then, we can rewrite (4.4) into the following unconstrained setting:

$$\min_{\mathbf{x}\in\mathbb{R}^{np}} \Big\{ F(\mathbf{x}) := f(\mathbf{x}) + g(\mathbf{x}) + \delta_{\mathcal{C}}(\mathbf{x}) \equiv \frac{1}{n}\sum_{i=1}^{n} f_i(x_i) + g(x_1) + \delta_{\mathcal{C}}(\mathbf{x}) \Big\}. \tag{4.5}$$

Note that (4.5) can be viewed as a composite nonconvex minimization problem of $f(\mathbf{x})$ and $g(\mathbf{x}) + \delta_{\mathcal{C}}(\mathbf{x})$. The first-order optimality condition of (4.5) can be written as

$$0 \in \nabla f(\mathbf{x}^\star) + \partial g(\mathbf{x}^\star) + \partial \delta_{\mathcal{C}}(\mathbf{x}^\star), \tag{4.6}$$

where $\partial\delta_{\mathcal{C}}$ is the subdifferential of $\delta_{\mathcal{C}}$, which is the normal cone of $\mathcal{C}$ (or, equivalently, $\partial\delta_{\mathcal{C}}(\mathbf{x}) = \mathcal{C}^\perp$ if $\mathbf{x} \in \mathcal{C}$, the orthogonal subspace of $\mathcal{C}$, and $\partial\delta_{\mathcal{C}}(\mathbf{x}) = \emptyset$, otherwise), and ∂g is the subdifferential of g. Note that since f is nonconvex, (4.6) only provides a necessary condition for $\mathbf{x}^\star := [x_1^\star, \cdots, x_n^\star]$ to be a local minimizer. Any $\mathbf{x}^\star$ satisfying (4.6) is called a (first-order) stationary point of (4.5). In this case, we have $x_i^\star = x_1^\star$ for all $i \in [n]$. Hence, using (4.6), we have $0 \in \nabla f(\mathbf{x}^\star) + \partial g(\mathbf{x}^\star) + \mathcal{C}^\perp$. This condition is equivalent to $0 \in \frac{1}{n}\sum_{i=1}^n \nabla f_i(x_i^\star) + \partial g(x_1^\star)$. However, since $\mathbf{x}^\star \in \mathcal{C}$, $x_i^\star = x_1^\star$ for all $i \in [n]$, then the last inclusion becomes $0 \in \frac{1}{n}\sum_{i=1}^n \nabla f_i(x_1^\star) + \partial g(x_1^\star)$. Equivalently, we have $x^\star := x_1^\star$ to be a stationary point of (4.2).

(c) Full parallel DR variant. Let us apply the DR splitting method to (4.6), which can be written explicitly as follows:

$$\begin{cases} \mathbf{y}^{k+1} &:= \mathbf{x}^k + \alpha(\bar{\mathbf{x}}^k - \mathbf{x}^k), \\ \mathbf{x}^{k+1} &:= \mathrm{prox}_{n\eta f}(\mathbf{y}^{k+1}), \\ \bar{\mathbf{x}}^{k+1} &:= \mathrm{prox}_{n\eta(g+\delta_{\mathcal{C}})}(2\mathbf{x}^{k+1} - \mathbf{y}^{k+1}), \end{cases} \tag{4.7}$$

where $\eta > 0$ is a given such that $n\eta$ is a step-size and $\alpha \in (0, 2]$ is a relaxation parameter (Themelis and Patrinos, 2020). If $\alpha = 1$, we recover the classical Douglas-Rachford scheme

(Lions and Mercier, 1979) and if $\alpha = 2$, we recover the Peaceman-Rachford splitting scheme (Bauschke and Combettes, 2017). Note that the classical DR scheme studied in Lions and Mercier (1979) was developed to solve monotone inclusions, and in our context, convex problems. Recently, it has been extended to solve nonconvex optimization problems, see, e.g., Li and Pong (2015); Themelis and Patrinos (2020).

Let us further take advantage of the special structure of f and $\delta_{\mathcal{C}}$ in (4.5) to obtain a special parallel DR variant.

- First, since $f(\mathbf{x}) = \frac{1}{n}\sum_{i=1}^n f_i(x_i)$, we have

$$\begin{aligned}\min_{\mathbf{x}}\Big\{f(\mathbf{x}) + \tfrac{1}{2n\eta}\|\mathbf{x}-\mathbf{y}^{k+1}\|^2\Big\} &= \min_{\mathbf{x}}\Big\{\tfrac{1}{n}\sum_{i=1}^n\Big[f_i(x_i) + \tfrac{1}{2\eta}\|x_i - y_i^{k+1}\|^2\Big]\Big\}\\ &= \tfrac{1}{n}\sum_{i=1}^n \min_{x_i}\Big\{f_i(x_i) + \tfrac{1}{2\eta}\|x_i - y_i^{k+1}\|^2\Big\}.\end{aligned}$$

  Hence, we can decompose the computation of $\mathbf{x}^{k+1} := \mathrm{prox}_{n\eta f}(\mathbf{y}^{k+1})$ from (4.7) as $x_i^{k+1} := \mathrm{prox}_{\eta f_i}(y_i^{k+1})$ for all $i \in [n]$.
- Next, we denote $\hat{\mathbf{x}}^{k+1} := 2\mathbf{x}^{k+1} - \mathbf{y}^{k+1}$, or equivalently, in component-wise $\hat{x}_i^{k+1} := 2x_i^{k+1} - y_i^{k+1}$ for all $i \in [n]$.
- Finally, the third line of (4.7) $\bar{\mathbf{x}}^{k+1} := \mathrm{prox}_{n\eta(g+\delta_{\mathcal{C}})}(\hat{\mathbf{x}}^{k+1})$ can be rewritten as

$$\bar{\mathbf{x}}^{k+1} := \mathrm{prox}_{n\eta(g+\delta_{\mathcal{C}})}(\hat{\mathbf{x}}^{k+1}) = \begin{cases}\underset{[x_1,\cdots,x_n]}{\mathrm{argmin}}\big\{g(x_1) + \frac{1}{2n\eta}\sum_{i=1}^n\|x_i - \hat{x}_i^{k+1}\|^2\big\}\\ \text{s.t.}\quad x_i = x_1,\quad \text{for all } i = 2,\cdots,n.\end{cases} \tag{4.8}$$

Let us solve(4.8) explicitly. First, we define a Lagrange function associated with (4.8) as

$$\mathcal{C}(\mathbf{x},\mathbf{z}) = g(x_1) + \frac{1}{2n\eta}\sum_{i=1}^n\|x_i - \hat{x}_i^{k+1}\|^2 + \sum_{i=1}^{n-1} z_i^\top(x_{i+1} - x_1),$$

where z_i $(i = 1, \cdots, n-1)$ are the corresponding Lagrange multipliers. Hence, the KKT condition of (4.8) can be written as

$$\begin{cases} \partial g(\bar{x}_1^{k+1}) + \frac{1}{n\eta}(\bar{x}_1^{k+1} - \hat{x}_1^{k+1}) - \sum_{i=1}^{n-1} z_i = 0, \\ \frac{1}{n\eta}(\bar{x}_{i+1}^{k+1} - \hat{x}_{i+1}^{k+1}) + z_i = 0, \quad \text{for all } i = 1, \cdots, n-1, \\ \bar{x}_{i+1}^{k+1} = \bar{x}_1^{k+1}, \quad \text{for all } i = 1, \cdots, n-1. \end{cases}$$

Summing up the second line from $i = 1$ to $i = n-1$ and combining the result with the last line of this KKT condition, we have

$$n\eta \sum_{i=1}^{n-1} z_i = \sum_{i=1}^{n-1} (\hat{x}_{i+1}^{k+1} - \bar{x}_{i+1}^{k+1}) = \sum_{i=2}^{n} \hat{x}_i^{k+1} - (n-1)\bar{x}_1^{k+1}.$$

Substituting this expression into the first line of the KKT condition, we get

$$\sum_{i=1}^{n} \hat{x}_i^{k+1} - (n-1)\bar{x}_1^{k+1} = \hat{x}_1^{k+1} + n\eta \sum_{i=1}^{n-1} z_i \in \bar{x}_1^{k+1} + n\eta \partial g(\bar{x}_1^{k+1}). \tag{4.9}$$

This condition is equivalent to $\sum_{i=1}^{n} \hat{x}_i^{k+1} \in n\bar{x}_1^{k+1} + n\eta \partial g(\bar{x}_1^{k+1})$. By introducing a new notation $\bar{x}^{k+1} := \bar{x}_1^{k+1}$, we eventually obtain from the last inclusion that

$$\bar{\mathbf{x}}^{k+1} := [\bar{x}^{k+1}, \cdots, \bar{x}^{k+1}] \in \mathbb{R}^{np}, \quad \text{where} \quad \bar{x}^{k+1} := \operatorname{prox}_{\eta g}\left(\tfrac{1}{n}\textstyle\sum_{i=1}^{n} \hat{x}_i^{k+1}\right).$$

If we introduce a new variable $\tilde{x}^{k+1} := \frac{1}{n}\sum_{i=1}^{n} \hat{x}_i^{k+1}$, then $\bar{x}^{k+1} := \operatorname{prox}_{\eta g}(\tilde{x}^{k+1})$.

Putting the above steps together, we obtain the following parallel DR variant for solving (4.2):

$$\begin{cases} y_i^{k+1} &:= y_i^k + \alpha(\bar{x}^k - x_i^k), \quad \forall i \in [n] \\ x_i^{k+1} &:= \operatorname{prox}_{\eta f_i}(y_i^{k+1}), \quad \forall i \in [n] \\ \hat{x}_i^{k+1} &:= 2x_i^{k+1} - y_i^{k+1}, \quad \forall i \in [n] \\ \tilde{x}^{k+1} &:= \frac{1}{n}\sum_{i=1}^{n} \hat{x}_i^{k+1}, \\ \bar{x}^{k+1} &:= \operatorname{prox}_{\eta g}(\tilde{x}^{k+1}). \end{cases} \tag{4.10}$$

This variant can be implemented in parallel. It is also known as a special variant of Tseng's splitting method (Bauschke and Combettes, 2017) in the convex case. We note that our algorithms are similar to FedSplit (Pathak and Wainwright, 2020) except that they use $\alpha = 2$ instead of $\alpha < 2$ as in our algorithms. In particular, FedSplit is a variant of the Peaceman-Rachford splitting method and it considers a special case of (4.2) when $g = 0$ and f_i is convex for all $i \in [n]$. If $g = 0$ (i.e., without regularizer), then the last line of (4.10) reduces to $\bar{x}^{k+1} = \tilde{x}^{k+1}$.

(d) Inexact block-coordinate DR variant. Instead of performing update for all workers $i \in [n]$ as in (4.10), we propose a new block-coordinate DR variant, called `FedDR`, where only a subset of workers $\mathcal{S}_k \subseteq [n]$ performs local update then send its local model to server for aggregation. For worker $i \notin \mathcal{S}_k$, the local model is unchanged, i.e., for all $i \notin \mathcal{S}_k$: $y_i^{k+1} = y_i^k$, $x_i^{k+1} = x_i^k$, and $\hat{x}_i^{k+1} = \hat{x}_i^k$. Hence, no communication with the server is needed for these workers. Furthermore, we assume that we can only approximate the proximal operator $\text{prox}_{\eta f_i}$ up to a given accuracy for all $i \in [n]$. In this case, we replace the exact proximal step $x_i^k := \text{prox}_{\eta f_i}(y_i^k)$ by its approximation $x_i^k :\approx \text{prox}_{\eta f_i}(y_i^k)$ up to a given accuracy $\epsilon_{i,k} \geq 0$ such that

$$\|x_i^k - \text{prox}_{\eta f_i}(y_i^k)\| \leq \epsilon_{i,k}. \tag{4.11}$$

Since x_i^k is approximately computed from $\text{prox}_{\eta f_i}(y_i^k)$ as in (4.11), we have

$$x_i^k = z_i^k + e_i^k, \quad \text{where} \quad z_i^k := \text{prox}_{\eta f_i}(y_i^k) \quad \text{and} \quad \|e_i^k\| \leq \epsilon_{i,k}. \tag{4.12}$$

We will use this representation of x_i^k and x_i^{k+1} in our analysis in the sequel.

More specifically, the update of our inexact block-coordinate DR variant can be described as follows.

- **Initialization:** Given an initial vector $x^0 \in \text{dom}(F)$ and accuracies $\epsilon_{i,0} \geq 0$.

  Initialize the server with $\bar{x}^0 := x^0$.

  Initialize all workers $i \in [n]$ with $y_i^0 := x^0$, $x_i^0 :\approx \text{prox}_{\eta f_i}(y_i^0)$, and $\hat{x}_i^0 := 2x_i^0 - y_i^0$.
- **The k-th iteration ($k \geq 0$):** Sample a proper subset $\mathcal{S}_k \subseteq [n]$ so that $\mathcal{S}_k$ presents as the subset of *active workers*.
- (***Communication***) Each worker $i \in \mathcal{S}_k$ receives $\bar{x}^k$ from the server.

- (***Local/worker update***) For each worker $i \in \mathcal{S}_k$, given $\epsilon_{i,k+1} \geq 0$, it updates

$$\begin{cases} y_i^{k+1} & := \quad y_i^k + \alpha(\bar{x}^k - x_i^k) \\ x_i^{k+1} & :\approx \quad \text{prox}_{\eta f_i}(y_i^{k+1}) \\ \hat{x}_i^{k+1} & := \quad 2x_i^{k+1} - y_i^{k+1}. \end{cases}$$

  Each worker $i \notin \mathcal{S}_k$ does nothing, i.e.:

$$\begin{cases} y_i^{k+1} & := \quad y_i^k \\ x_i^{k+1} & := \quad x_i^k \\ \hat{x}_i^{k+1} & := \quad \hat{x}_i^k. \end{cases}$$

- (***Communication***) Each worker $i \in \mathcal{S}_k$ sends only $\hat{x}_i^{k+1}$ to the server.
- (***Global/Server update***) The server aggregates $\tilde{x}^{k+1} := \frac{1}{n}\sum_{i=1}^n \hat{x}_i^{k+1}$, and then compute $\bar{x}^{k+1} := \text{prox}_{\eta g}(\tilde{x}^{k+1})$.

Note that the global update on $\tilde{x}^{k+1}$ can be rewritten as

$$\begin{aligned} \tilde{x}^{k+1} &:= \tfrac{1}{n}\textstyle\sum_{i=1}^n \hat{x}_i^{k+1} = \tfrac{1}{n}\sum_{i\in\mathcal{S}_k}^n \hat{x}_i^{k+1} + \tfrac{1}{n}\sum_{i\notin\mathcal{S}_k}^n \hat{x}_i^k \\ &= \tfrac{1}{n}\textstyle\sum_{i=1}^n \hat{x}_i^k + \tfrac{1}{n}\sum_{i\in\mathcal{S}_k}^n (\hat{x}_i^{k+1} - \hat{x}_i^k) \\ &= \tilde{x}^k + \tfrac{1}{n}\textstyle\sum_{i\in\mathcal{S}_k} \Delta\hat{x}_i^k \end{aligned}$$

to see that only workers in $\mathcal{S}_k$ participate in the global model update.

The complete algorithm is presented in Algorithm 4. Let us make the following remarks. Firstly, **FedDR** mainly updates of three sequences $\{\bar{x}^k\}$, $\{x_i^k\}$ and $\{y_i^k\}$. While $\bar{x}^k$ is an averaged model to approximately minimize the global objective function F, x_i^k act as local models trying to optimize a regularized local loss function w.r.t. its local data distribution, and y_i^k keeps track of the residuals from the local models to the global one. Secondly, we allow x_i^k to be an approximation of $\text{prox}_{\eta f_i}(y_i^k)$ up to an accuracy $\epsilon_{i,k} \geq 0$ as defined in (1.4), i.e., $\|x_i^k - \text{prox}_{\eta f_i}(y_i^k)\| \leq \epsilon_{i,k}$ for all $i \in [n]$ if $k = 0$ and for all $i \in \mathcal{S}_{k-1}$ if $k > 0$. If $\epsilon_{i,k} = 0$, then we get the exact evaluation $x_i^k := \text{prox}_{\eta f_i}(y_i^k)$. Approximately evaluating $\text{prox}_{\eta f_i}$ can be done, e.g., by local SGD as in FedAvg. Thirdly, Algorithm 4 is different from existing randomized proximal

gradient-based methods since we rely on a DR splitting scheme and can handle composite settings. Here, three iterates y_i^k, x_i^k, and $\hat{x}_i^k$ at Step 5 are updated sequentially, making it challenging to analyze convergence. Lastly, the subset of active users $\mathcal{S}_k$ is sampled from a random set-valued mapping $\hat{\mathcal{S}}$. As specified in Assumption 4.3, this sampling mechanism covers a wide range of sampling strategies. Clearly, if $\mathcal{S}_k = [n]$ and $g = 0$, then Algorithm 4 reduces to FedSplit, but for the nonconvex case. Hence, our convergence guarantee below remains applicable, and the guarantee is sure. Note that both our model (4.2) and Algorithm 4 are completely different from Yuan and Ma (2020).

Algorithm 4 (**FedDR** - FL with Randomized DR)

1: **Initialization:** Take $x^0 \in \mathrm{dom}(F)$. Choose $\eta > 0$ and $\alpha > 0$, and accuracies $\epsilon_{i,0} \geq 0$ $(i \in [n])$.
   Initialize the server with $\bar{x}^0 := x^0$ and $\tilde{x}^0 := x^0$.
   Initialize each worker $i \in [n]$ with $y_i^0 := x^0$, $x_i^0 :\approx \mathrm{prox}_{\eta f_i}(y_i^0)$, and $\hat{x}_i^0 := 2x_i^0 - y_i^0$.
2: **For** $k := 0, \cdots, K$ **do**
3: [*Active workers*] Generate a proper realization $\mathcal{S}_k \subseteq [n]$ of $\hat{\mathcal{S}}$ (see Assumption 4.3).
4: [*Communication*] Each worker $i \in \mathcal{S}_k$ receives $\bar{x}^k$ from the server.
5: [*Local update*] **For each worker** $i \in \mathcal{S}_k$ **do**: Choose $\epsilon_{i,k+1} \geq 0$ and update

$$y_i^{k+1} := y_i^k + \alpha(\bar{x}^k - x_i^k), \quad x_i^{k+1} :\approx \mathrm{prox}_{\eta f_i}(y_i^{k+1}), \quad \text{and} \quad \hat{x}_i^{k+1} := 2x_i^{k+1} - y_i^{k+1}.$$

6: [*Communication*] Each worker $i \in \mathcal{S}_k$ sends $\Delta\hat{x}_i^k := \hat{x}_i^{k+1} - \hat{x}_i^k$ back to the server.
7: [*Sever aggregation*] The server aggregates $\tilde{x}^{k+1} := \tilde{x}^k + \frac{1}{n}\sum_{i\in\mathcal{S}_k} \Delta\hat{x}_i^k$.
8: [*Sever update*] Then, the sever updates $\bar{x}^{k+1} := \mathrm{prox}_{\eta g}\big(\tilde{x}^{k+1}\big)$.
9: **End For**

4.2.2 Convergence analysis of FedDR

To analyze convergence of Algorithm 4, we conceptually introduce z_i^0 and z_i^{k+1} for $i \in [n]$ as follows:

$$z_i^0 := \mathrm{prox}_{\eta f_i}(y_i^0), \quad z_i^{k+1} := \begin{cases} \mathrm{prox}_{\eta f_i}(x_i^{k+1}) & \text{if} \quad i \in \mathcal{S}_k \\ z_i^k & \text{if} \quad i \notin \mathcal{S}_k, \end{cases} \quad \text{and} \quad x_i^k := z_i^k + e_i^k. \tag{4.13}$$

Here, e_i^k is the vector of errors. Note that z_i^0 and z_i^{k+1} do not exist in actual implementation of Algorithm 4, and we only have their approximations x_i^0 and x_i^{k+1}, respectively. For any $k \geq 0$,

since $x_i^{k+1} = x_i^k$ and $z_i^{k+1} = z_i^k$ for $i \notin \mathcal{S}_k$, we have $\|x_i^{k+1} - z_i^{k+1}\| = \|e_i^{k+1}\| = \|x_i^k - z_i^k\| = \|e_i^k\|$ for $i \notin \mathcal{S}_k$. To guarantee $\|e_i^{k+1}\| = \|e_i^k\|$ for $i \notin \mathcal{S}_k$, we just choose $\epsilon_{i,k+1} := \epsilon_{i,k}$ for $i \notin \mathcal{S}_k$.

Note that in Algorithm 4, we have not specified the choice of $\mathcal{S}_k$. The subset $\mathcal{S}_k$ is an iid realization of a random set-valued mapping $\hat{\mathcal{S}}$ from $[n]$ to $2^{[n]}$, the collection of all subsets of $[n]$. We first impose the following assumption about the distribution of our sampling scheme $\hat{\mathcal{S}}$.

Assumption 4.3. There exist $\mathbf{p}_1, \cdots, \mathbf{p}_n > 0$ such that $\mathbb{P}\big(i \in \hat{\mathcal{S}}\big) = \mathbf{p}_i > 0$ for all $i \in [n]$.

This assumption covers a large class of sampling schemes as discussed in Richtárik and Takáč (2016), including non-overlapping uniform and doubly uniform. This assumption guarantees that every worker has a non-negligible probability to be updated. Note that $\mathbf{p}_i = \sum_{\mathcal{S}: i \in \mathcal{S}} \mathbb{P}(\mathcal{S})$ due to Assumption 4.3. For the sake of notation, we also denote $\hat{\mathbf{p}} := \min\{\mathbf{p}_i : i \in [n]\} > 0$.

From Assumption 4.3, $\hat{\mathcal{S}}$ is a proper sampling scheme in the sense that $\mathbf{p}_i := \mathbb{P}(i \in \hat{S}) > 0$ for all $i \in [n]$. By specifying this probability distribution $\mathbf{p} := (\mathbf{p}_1, \cdots, \mathbf{p}_n)$, we obtain different sampling strategies ranging from uniform to non-uniform as discussed in Richtárik and Takáč (2016). Our analysis does hold for arbitrary sampling scheme that satisfies Assumption 4.3. Given a proper sampling scheme $\hat{\mathcal{S}}$ of $[n]$, let $\mathcal{S}_k$ be an iid realization of $\hat{\mathcal{S}}$ and $\mathcal{F}_k := \sigma(\mathcal{S}_0, \cdots, \mathcal{S}_k)$ be the σ-algebra generated by $\mathcal{S}_0, \cdots, \mathcal{S}_k$.

To analyze convergence of Algorithm 4, we introduce the following Lyapunov function:

$$V_\eta^k(\bar{x}^k) := g(\bar{x}^k) + \frac{1}{n}\sum_{i=1}^n \Big[f_i(x_i^k) + \langle \nabla f_i(x_i^k), \bar{x}^k - x_i^k\rangle + \frac{1}{2\eta}\|\bar{x}^k - x_i^k\|^2\Big]. \tag{4.14}$$

The following descent lemma which holds surely for any subset $\mathcal{S}_k$ of $[n]$ is key to achieve convergence of FedDR.

Lemma 4.1 (Sure descent lemma)**.** *Suppose that Assumptions 4.1, 4.2, and 4.3 hold. Let* $\{(x_i^k, y_i^k, z_i^k, \hat{x}_i^k, \bar{x}^k)\}$ *be generated by Algorithm 4 and* (4.13)*, and* $V_\eta^k(\cdot)$ *be defined by* (4.14)*. Then, the following estimate holds:*

$$\begin{aligned} V_\eta^{k+1}(\bar{x}^{k+1}) \leq\ & V_\eta^k(\bar{x}^k) - \tfrac{[2-\alpha(L\eta+1)-2L^2\eta^2-4\alpha\gamma_4(1+L^2\eta^2)]}{2\alpha\eta n}\textstyle\sum_{i\in\mathcal{S}_k}\|x_i^{k+1} - x_i^k\|^2 \\ & - \tfrac{(1-\gamma_3)}{2\eta}\|\bar{x}^{k+1} - \bar{x}^k\|^2 + \tfrac{(1+\eta^2L^2)}{\eta\gamma_3}E_{k+1}^2 \\ & + \tfrac{2(1+\eta L)^2}{\gamma_4\eta\alpha^2 n}\textstyle\sum_{i\in\mathcal{S}_k}[\|e_i^k\|^2 + \|e_i^{k+1}\|^2], \end{aligned} \tag{4.15}$$

where $E_{k+1}^2 := \frac{1}{n}\sum_{i\notin\mathcal{S}_k}\|e_i^k\|^2 + \frac{1}{n}\sum_{i\in\mathcal{S}_k}\|e_i^{k+1}\|^2$, *and* $\gamma_3, \gamma_4 > 0$. *In particular, if* $E_{k+1}^2 = 0$, *then we allow* $\gamma_3 = 0$, *and if* $e_i^k = e_i^{k+1} = 0$ *for all* $i \in \mathcal{S}_k$, *then we allow* $\gamma_4 = 0$.

The proof of this lemma is presented in Section 4.5.1.2 which uses several useful lemmas in Section 4.5.1.1. The following lemma specifies the choice of parameters to obtain a descent property on V_η^k when $\mathrm{prox}_{\eta f_i}$ is evaluated approximately. The proof of this lemma is in Section 4.5.1.3.

Lemma 4.2. *Suppose that Assumptions 4.1, 4.2, and 4.3 hold. Let* $V_\eta^k(\cdot)$ *be defined by* (4.14) *and* $\gamma_1, \gamma_2, \gamma_4 > 0$ *be given. Let* $\{(x_i^k, y_i^k, \hat{x}_i^k, \bar{x}^k)\}$ *be generated by Algorithm 4 using*

$$0 < \alpha < \frac{\min\{8, \sqrt{17+64\gamma_4}-1\}}{4(1+4\gamma_4)} \quad \text{and} \quad 0 < \eta < \frac{\sqrt{(4-\alpha)^2 - 16\alpha^2\gamma_4(1+4\gamma_4)} - \alpha}{4L(1+2\alpha\gamma_4)}. \tag{4.16}$$

Then, V_η^k *is bounded from bellow by* $F^\star$, *i.e.* $V_\eta^k \geq F^\star$ *and the following estimate holds:*

$$\frac{\beta}{2n}\sum_{i=1}^n \|\bar{x}^k - x_i^k\|^2 \leq V_\eta^k(\bar{x}^k) - \mathbb{E}\left[V_\eta^{k+1}(\bar{x}^{k+1}) \mid \mathcal{F}_{k-1}\right] + \frac{1}{n}\sum_{i=1}^n(\rho_1\epsilon_{i,k}^2 + \rho_2\epsilon_{i,k+1}^2), \tag{4.17}$$

where

$$\begin{cases} \beta &:= \frac{\hat{\mathbf{p}}\alpha[2-\alpha(L\eta+1)-2L^2\eta^2-4\gamma_4\alpha(1+L^2\eta^2)]}{2\eta(1+\gamma_1)(1+L^2\eta^2)} > 0, \\ \rho_2 &:= \frac{2(1+\eta L)^2}{\gamma_4\eta\alpha^2} + \frac{(1+\eta^2L^2)}{\eta} + \frac{\alpha[2-\alpha(L\eta+1)-2L^2\eta^2-4\alpha\gamma_4(1+L^2\eta^2)]}{2\eta(1+L^2\eta^2)\gamma_1}, \\ \rho_1 &:= \rho_2 + \frac{(1+\eta^2L^2)}{\eta}. \end{cases} \tag{4.18}$$

Here, if $\epsilon_{i,k} = 0$ *for all* $i \in [n]$ *and* $k \geq 0$, *then we allow* $\gamma_1 = \gamma_2 = \gamma_4 = \rho_1 = \rho_2 = 0$.

We are ready to present the convergence of Algorithm 4.

Theorem 4.1. *Suppose that Assumptions 4.1, 4.2, and 4.3 hold. Let* $\{(x_i^k, y_i^k, \hat{x}_i^k, \bar{x}^k)\}$ *be generated by Algorithm 4 using stepsizes* α *and* η *as in* (4.16). *Then, we have*

$$\frac{1}{K+1}\sum_{k=0}^K \mathbb{E}\left[\|\mathcal{G}_\eta(\bar{x}^k)\|^2\right] \leq \frac{C_1[F(x^0)-F^\star]}{K+1} + \frac{1}{n(K+1)}\sum_{k=0}^K\sum_{i=1}^n\left(C_2\epsilon_{i,k}^2 + C_3\epsilon_{i,k+1}^2\right), \tag{4.19}$$

where β, ρ_1, *and* ρ_2 *are defined by* (4.18), *and*

$$C_1 := \frac{2(1+\eta L)^2(1+\gamma_2)}{\eta^2\beta}, \quad C_2 := \rho_1 C_1, \quad \text{and} \quad C_3 := \rho_2 C_1 + \frac{(1+\eta L)^2(1+\gamma_2)}{\eta^2\gamma_2}.$$

Let $\tilde{x}^K$ *be selected uniformly at random from* $\{\bar{x}^0, \cdots, \bar{x}^K\}$ *as the output of Algorithm 4. Let the accuracies* $\epsilon_{i,k}$ *for all* $i \in [n]$ *and* $k \geq 0$ *be chosen such that* $\frac{1}{n}\sum_{i=1}^{n}\sum_{k=0}^{K+1}\epsilon_{i,k}^2 \leq M$ *for all* $K \geq 0$*. Then, if we run Algorithm 4 for at most*

$$K := \left\lfloor \frac{C_1[F(x^0) - F^\star] + (C_2 + C_3)M}{\varepsilon^2} \right\rfloor \equiv \mathcal{O}\left(\varepsilon^{-2}\right)$$

iterations, then $\tilde{x}^K$ *is an* ε*-stationary point of* (4.2) *as in Definition 4.1.2.*

The proof of Theorem 4.1 can be found in Section 4.5.1.4.

Remark 6 (**Comparison**). Since (4.2) is nonconvex, our $\mathcal{O}\left(\varepsilon^{-2}\right)$ communication complexity is the state-of-the-art, matching the lower bound complexity (up to a constant factor) (Zhang et al., 2020). However, unlike FedSplit and FedPD (Zhang et al., 2020), our flexible sampling scheme allows us to update a subset of workers at each round. This can potentially further resolve the communication bottleneck (Li et al., 2020a). We note that FedSplit is a variant of the Peaceman-Rachford splitting method. However, since $\alpha < 2$ in Algorithm 4, we cannot recover FedSplit even in the convex case with full update on all workers.

Remark 7. [**Choice of accuracies** ϵ_i^k] To guarantee $\frac{1}{n}\sum_{i=1}^{n}\sum_{k=0}^{K+1}\epsilon_{i,k}^2 \leq M$ in Theorem 4.1 for a given constant $M > 0$ and for all $K \geq 0$, one can choose, e.g., $\epsilon_{i,k}^2 := \frac{M}{2(k+1)^2}$ for all $i \in [n]$ and $k \geq 0$. In this case, we can easily show that $\frac{1}{n}\sum_{i=1}^{n}\sum_{k=0}^{K+1}\epsilon_{i,k}^2 = \frac{M}{2}\sum_{k=0}^{K+1}\frac{1}{(k+1)^2} \leq M$.

Note that, instead of using decreasing accuracies as in Remark 7, one can also use relative accuracies as in the bounded relative error condition defined as follows.

Definition 4.2.1. For any $i \in \mathcal{S}_k$, given x_i^k and y_i^{k+1}, we say that x_i^{k+1} approximates $\text{prox}_{\eta f_i}(y_i^{k+1})$ up to a **bounded relative error** if there is a constant $\theta_i > 0$ (independent of k) such that

$$\|x_i^{k+1} - \text{prox}_{\eta f_i}(y_i^{k+1})\|^2 \leq \varepsilon_{i,k+1}^2 := \theta_i \|x_i^{k+1} - x_i^k\|^2 \tag{4.20}$$

The condition in Definition 4.2.1 is more practical as it allows solving $\text{prox}_{\eta f_i}$ for a fixed number of iterations. Such an idea has been widely used in the literature, including Liu et al. (2021). The following theorem specifies convergence of Algorithm 4 under the relative bounded error condition (4.20), whose proof can be found in Section 4.5.2.

Theorem 4.2. *Suppose that Assumptions 4.1, 4.2, and 4.3 hold, and the bounded relative error condition* (4.20) *in Definition 4.2.1 holds with* $\theta_i := \hat{\theta}\mathbf{p}_i$ *for a fixed constant* $\hat{\theta} > 0$. *Let* $\{(x_i^k, y_i^k, \hat{x}_i^k, \bar{x}^k)\}$ *be generated by Algorithm 4 using a relaxation stepsize* $\alpha = 1$ *and* $x_i^0 := \mathrm{prox}_{\eta f_i}(y_i^0)$ *for* $i \in [n]$. *If* γ_4 *and* $\hat{\theta}$ *are chosen such that* $1 - 4\gamma_4 - 8\hat{C}\hat{\theta} > 0$ *and* η *is chosen by*

$$0 < \eta < \bar{\eta} := \frac{\sqrt{1+8(1+2\gamma_4)(1-4\gamma_4-8\hat{C}\hat{\theta})}-1}{4L(1+2\gamma_4)}, \tag{4.21}$$

where $\hat{C} := \max\left\{1 + \eta^2 L^2, \frac{2(1+\eta L)^2}{\gamma_4}\right\}$, *then the following bound holds*

$$\frac{1}{K+1}\sum_{k=0}^{K}\mathbb{E}\left[\|\mathcal{G}_\eta(\bar{x}^k)\|^2\right] \leq \frac{\widetilde{C}\left[F(x^0) - F^\star\right]}{(K+1)}, \tag{4.22}$$

where $\widetilde{C} > 0$ *is computed by*

$$\widetilde{C} := \frac{\hat{\mathbf{p}}^2\eta[1 - L\eta - 2L^2\eta^2 - 4\gamma_4(1 + L^2\eta^2) - 8\hat{C}\hat{\theta}]}{4\left[4(1 + L^2\eta^2 + 2\hat{\theta}) + \hat{\mathbf{p}}\hat{\theta}\right](1 + \eta L)^2}. \tag{4.23}$$

The remaining conclusions of this theorem are similar to Theorem 4.1, and we omit them here.

4.3 AsyncFedDR and its convergence guarantee

Although **FedDR** has been shown to converge, it is more practical to account for the system heterogeneity of local workers. Requiring synchronous aggregation at the end of each communication round may lead to slow down in training. It is natural to have asynchronous update from local workers as seen, e.g., in Recht et al. (2011); Stich (2018). However, asynchronous implementation remains limited in FL. Here, we propose **asyncFedDR**, an asynchronous variant of **FedDR**, and analyze its convergence guarantee. For the sake of our analysis, we only consider $\mathcal{S}_k := \{i_k\}$ with exact evaluation of $\mathrm{prox}_{\eta f_i}$ and bounded delay, but extensions to general $\mathcal{S}_k$ and inexact $\mathrm{prox}_{\eta f_i}$ are similar to Algorithm 4.

4.3.1 Derivation of asyncFedDR

Let us first explain the main idea of **asyncFedDR**. At each iteration k, each worker receives a delay copy $\bar{x}^{k-d_{i_k}^k}$ of $\bar{x}^k$ from the server with a delay $d_{i_k}^k$. The active worker i_k will update its

own local model $(y_i^k, x_i^k, \hat{x}_i^k)$ in an asynchronous mode without waiting for others to complete. Once completing its update, worker i_k just sends an increment $\Delta\hat{x}_{i_k}^k$ to the server to update the global model, while others may be reading. Overall, the complete **asyncFedDR** is presented in Algorithm 5.

Algorithm 5 (Asynchronous FedDR (**asyncFedDR**))

1: **Initialization:** Take $x^0 \in \mathrm{dom}(F)$ and choose $\eta > 0$ and $\alpha > 0$.
Initialize the server with $\bar{x}^0 := x^0$ and $\tilde{x}^0 := 0$.
Initialize each worker $i \in [n]$ with $y_i^0 := x^0$, $x_i^0 := \mathrm{prox}_{\eta f_i}(y_i^0)$, and $\hat{x}_i^0 := 2x_i^0 - y_i^0$.
2: **For** $k := 0, \cdots, K$ **do**
3: Select i_k such that (i_k, d^k) is a realization of $(\hat{i}_k, \hat{d}^k)$.
4: [*Communication*] User i_k receives $\bar{x}^{k-d_{i_k}^k}$, a delayed version of $\bar{x}^k$ with the delay $d_{i_k}^k$.
5: [*Local update*] User i_k updates

$$y_{i_k}^{k+1} := y_{i_k}^k + \alpha(\bar{x}^{k-d_{i_k}^k} - x_{i_k}^k), \quad x_{i_k}^{k+1} := \mathrm{prox}_{\eta f_{i_k}}(y_{i_k}^{k+1}), \text{ and } \hat{x}_{i_k}^{k+1} := 2x_{i_k}^{k+1} - y_{i_k}^{k+1}.$$

Other workers maintain $y_i^{k+1} := y_i^k$, $x_i^{k+1} := x_i^k$, and $\hat{x}_i^{k+1} := \hat{x}_i^k$ for $i \neq i_k$.
6: [*Communication*] User i_k sends $\Delta_{i_k}^k := \hat{x}_{i_k}^{k+1} - \hat{x}_{i_k}^k$ back to the server.
7: [*Sever aggregation*] The server aggregates $\tilde{x}^{k+1} := \tilde{x}^k + \frac{1}{n}\Delta_{i_k}^k$.
8: [*Sever update*] Then, the sever updates $\bar{x}^{k+1} := \mathrm{prox}_{\eta g}\big(\tilde{x}^{k+1}\big)$.
9: **End For**

In our analysis below, a transition of iteration from k to $k+1$ is triggered whenever a worker completes its update. Moreover, at Step 3, active worker i_k is chosen from a realization (i_k, d^k) of a joint random vector $(\hat{i}_k, \hat{d}^k)$ at the k-th iteration. Here, we do not assume i_k to be uniformly random or independent of the delay d^k. This allows Algorithm 5 to capture the variety of asynchronous implementations and architectures. Note that $\bar{x}^{k-d_{i_k}^k}$ at Step 4 is a delayed version of $\bar{x}^k$, which only exists on the server when worker i_k is reading. However, right after, $\bar{x}^k$ may be updated by another worker.

Illustrative example. To better understand the update of **asyncFedDR**, Figure 4.1 depicts a simple scenario where there are 4 workers ($C1$ - $C4$) asynchronously perform updates and with $g(\cdot) = 0$. At iteration $k = 4$, worker $C4$ finishes its update so that the server performs updates. During this process, worker $C1$ starts its update by receiving a global model $\bar{x}^{4-d_{i_4}^4}$ from server which is the average of $(\hat{x}_1^4, \hat{x}_2^4, \hat{x}_3^4, \hat{x}_4^4)$. At iteration $t = 7$, $C1$ finishes its update. Although $\hat{x}_1$ and $\hat{x}_4$ do not change during this time, i.e. $\hat{x}_1^6 = \hat{x}_1^4$ and $\hat{x}_4^6 = \hat{x}_4^4$, $\hat{x}_2$ and $\hat{x}_3$ have been updated at $k = 5, 6$ from worker $C2$ and $C3$, respectively. Therefore, the global model $\bar{x}^k$ used

to perform the update at $k = 7$ is actually aggregated from $(\hat{x}_1^6, \hat{x}_2^4, \hat{x}_3^5, \hat{x}_4^6)$ not $(\hat{x}_1^6, \hat{x}_2^6, \hat{x}_3^6, \hat{x}_4^6)$. In other words, each worker receives a delay estimate $\bar{x}^{k-d^k}$ where $d^k = (d_1^k, \cdots, d_n^k)$ is a delay vector and $d_i^k = \max\{t \in [k] : i_t = i\}$, i.e. the last time $\hat{x}_i$ gets updated up to iteration k. Note that when $d_i^k = 0$ for all i, Algorithm 5 reduces to its synchronous variant, i.e. a special variant of Algorithm 4 with $\mathcal{S}_k = \{i_k\}$.

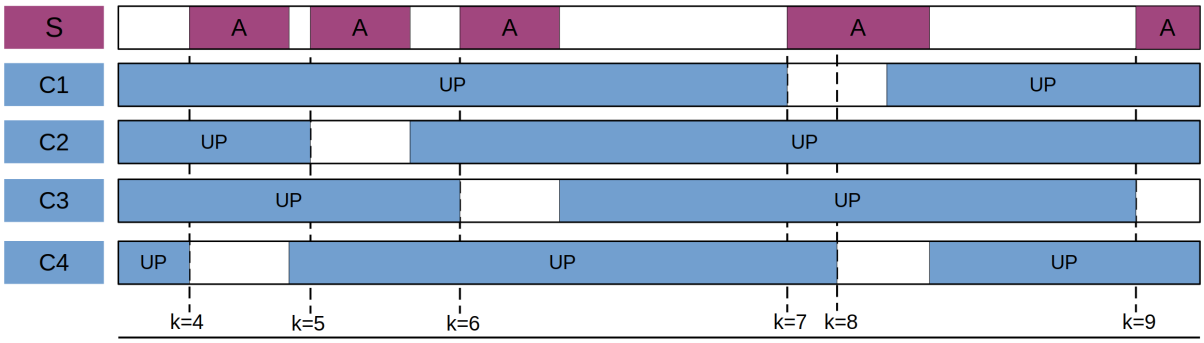


Figure 4.1: Asynchronous update with 4 workers. Here, "A" blocks represent server process and "UP" blocks represent worker process; C_1-C_4 are communication rounds.

Dual-memory approach. Let us provide more details on the implementation of our asynchronous algorithm. When a worker finishes its local update, the updated model (or model difference) is sent to the server for a proximal aggregation step. When the server is performing a proximal aggregation step, other workers might need to read from the global model. To allow concurrent read/write operations, one easy method is to have two models stored on the server, denoted as model 1 and model 2. At any given time, one model is on "read" state (it is supposed to be read from) and the other will be on "write" state (it will be written on when the server finishes aggregation). Suppose model 1 is on a "read" state and model 2 is on a "write" state, then all workers can read from model 1. When the server completes the proximal aggregation, model 2 becomes the latest model and it will change to a "read" state while model 1 is on a "write" state. This implementation detail is also discussed in Peng et al. (2016), which is termed by a *dual-memory approach*.

4.3.2 Probabilistic model

Let $\xi^k := (i_k, d^k)$ be a realization of a joint random vector $\hat{\xi}^k := (\hat{i}_k, \hat{d}^k)$ of the worker index $\hat{i}_k \in [n]$ and the delay vector $\hat{d}^k = (\hat{d}_1^k, \cdots, \hat{d}_n^k) \in \mathcal{D} := \{0, 1, \cdots, \tau\}^n$ presented at the current

iteration k. We consider $k+1$ random vectors $\hat{\xi}^l$ $(0 \leq l \leq k)$ that form a concatenate random vector $\hat{\xi}^{0:k} := (\hat{\xi}^0, \cdots, \hat{\xi}^k)$. We also use $\xi^{0:k} = (\xi^0, \xi^1, \cdots, \xi^k)$ for $k+1$ possible values of the random vector $\hat{\xi}^{0:k}$. Let Ω be the sample space of all sequences $\omega := \{(i_k, d^k)\}_{k\geq 0} \equiv \{\xi^k\}_{k\geq 0}$. We define a cylinder $\mathcal{C}_k(\xi^{0:k}) := \{\omega \in \Omega : (\omega_0, \cdots, \omega_k) = \xi^{0:k}\}$ as a subset in Ω and $\mathcal{C}_k$ is the set of all possible subsets $\mathcal{C}_k(\xi^{0:k})$ when ξ^t, $t = 0, \cdots, k$, take all possible values, where ω_l is the l-th coordinate of ω. Note that $\{\mathcal{C}_k\}_{k\geq 0}$ forms a partition of Ω and measurable. Let $\mathcal{F}_k := \sigma(\mathcal{C}_k)$ be the σ-algebra generated by $\mathcal{C}_k$ and $\mathcal{F} := \sigma(\cup_{k=0}^{\infty}\mathcal{C}_k)$. Clearly, $\{\mathcal{F}_k\}_{k\geq 0}$ forms a filtration such that $\mathcal{F}_k \subseteq \mathcal{F}_{k+1} \subseteq \cdots \subseteq \mathcal{F}$ for $k \geq 0$ that is sufficient to cope with the evolution of Algorithm 5.

For each $\mathcal{C}_k(\xi^{0:k})$ we also equip with a probability $\mathbf{p}(\xi^{0:k}) := \mathbb{P}(\mathcal{C}_k(\xi^{0:k}))$. Then, $(\Omega, \mathcal{F}, \mathbb{P})$ forms a probability space. Our conditional probability is defined as

$$\mathbf{p}((i,d) \mid \xi^{0:k}) := \mathbb{P}(\mathcal{C}_{k+1}(\xi^{0:k+1}))/\mathbb{P}(\mathcal{C}_k(\xi^{0:k})),$$

where we set $\mathbf{p}((i,d) \mid \xi^{0:k}) := 0$ if $\mathbf{p}(\xi^{0:k}) = 0$. We do not need to know these probabilities in advance. They are determined based on the particular system such as hardware architecture, software implementation, asynchrony, and our strategy for selecting active worker.

Now, if X is a random variable defined on Ω, as shown in Cannelli et al. (2019), we have

$$\mathbb{E}[X \mid \mathcal{F}_k] = \sum_{(i,d)\in[n]\times\mathcal{D}} \mathbf{p}((i,d) \mid \xi^{0:k}) X(\xi^{0:k}, (i,d)). \tag{4.24}$$

Note from Assumption 4.4 that

$$\mathbf{p}(i \mid \xi^{0:k}) := \sum_{d\in\mathcal{D}} \mathbf{p}((i,d) \mid \xi^{0:k}) \geq \hat{\mathbf{p}}. \tag{4.25}$$

Our probability model described above allows us to handle a variety class of asynchronous algorithms derived from the DR splitting scheme. Here, we do not make independent assumption between the active worker $\hat{i}_k$ and the delay vector $\hat{d}^k$.

4.3.3 Convergence analysis

Since we treat the active worker i_k and the delay vector d^k jointly at each iteration k as a realization of a joint random vector $(\hat{i}_k, \hat{d}^k)$, we adopt the probabilistic model from Cannelli et al. (2019) to analyze Algorithm 5. This new model allows us to cope with a more general class of asynchronous variants of our method.

To analyze Algorithm 5, we impose Assumption 4.4 on the implementation below.

Assumption 4.4. For all $i \in [n]$ and $\omega \in \Omega$, there exists at least one $t \in \{0, 1, \cdots, T\}$ with $T > 0$, such that

$$\sum_{d \in \mathcal{D}} \mathbf{p}((i, d) \mid \xi^{0:k+t-1}) \geq \hat{\mathbf{p}} \quad \text{if } \mathbf{p}(\xi^{0:k}) > 0, \tag{4.26}$$

for a given $\hat{\mathbf{p}} > 0$ and any $k \geq 0$. Assume also that $d_i^k \leq \tau$ and $d_{i_k}^k = 0$ for all $k \geq 0$ and $i, i_k \in [n]$.

Assumption 4.4 implies that during an interval of T iterations, every worker has a non-negligible positive probability to be updated. Note that if the worker i_k is active, then it uses recent value with no delay, i.e., $d_{i_k}^k = 0$ as in Assumption 4.4. Moreover, the bounded delay assumption $d_i^k \leq \tau$ is standard to analyze convergence of asynchronous algorithms, see e.g., Cannelli et al. (2019); Nguyen et al. (2018); Peng et al. (2016); Recht et al. (2011); Xie et al. (2019).

Let us present the following lemma which is key to establish a sure-descent property. The proof of this lemma can be found in Section 4.5.3.1

Lemma 4.3 (Sure descent)**.** *Suppose that Assumptions 4.1, 4.2, and 4.4 hold for* (4.2)*. Let* $\{(x_i^k, y_i^k, \hat{x}_i^k, \tilde{x}^k, \bar{x}^k)\}$ *be generated by Algorithm 5 and* $V_\eta^k(\cdot)$ *be defined as in* (4.14)*. Then, for all* $k \geq 0$*, the following estimate holds:*

$$\begin{aligned} V_\eta^{k+1}(\bar{x}^{k+1}) \; + \; & \tfrac{\tau}{n\eta} \textstyle\sum_{l=k+1-\tau}^{k} (l - k + \tau) \|\bar{x}^{l+1} - \bar{x}^l\|^2 \leq V_\eta^k(\bar{x}^k) - \tfrac{\rho}{2} \|x_{i_k}^{k+1} - x_{i_k}^k\|^2 \\ & + \; \tfrac{\tau}{n\eta} \textstyle\sum_{l=k-\tau}^{k-1} (l - (k-1) + \tau) \|\bar{x}^{l+1} - \bar{x}^l\|^2, \end{aligned} \tag{4.27}$$

where

$$\rho := \begin{cases} \frac{2(1-\alpha)-(2+\alpha)L^2\eta^2-L\alpha\eta}{\alpha\eta n} & \text{if} \quad 2\tau^2 \leq n, \\ \frac{n^2[2(1-\alpha)-(2+\alpha)L^2\eta^2-L\alpha\eta]-\alpha(1+\eta^2L^2)(2\tau^2-n)}{\alpha\eta n^3} & \text{otherwise.} \end{cases}$$

As a result, we have the following key lemma whose proof is in Section 4.5.3.2.

Lemma 4.4 (Sure descent lemma)**.** *Suppose that Assumptions 4.1, 4.2, and 4.4 hold. Let* $\{(x_i^k, y_i^k, \hat{x}_i^k, \tilde{x}^k, \bar{x}^k)\}$ *be generated by Algorithm 5 and* V_η^k *be defined as in* (4.14)*. Let*

$$\widetilde{V}_\eta^k(\bar{x}^k) := V_\eta^k(\bar{x}^k) + \tfrac{1}{n\eta}\textstyle\sum_{l=k-\tau}^{k-1}[l-(k-\tau)+1]\|\bar{x}^{l+1}-\bar{x}^l\|^2. \tag{4.28}$$

Suppose that we choose $0 < \alpha < \bar{\alpha}$ *and* $0 < \eta < \bar{\eta}$*, where* $c := \frac{2\tau^2-n}{n^2}$*,*

$$\begin{aligned} \bar{\alpha} &:= \begin{cases} 1 & \text{if } 2\tau^2 \leq n, \\ \frac{2}{2+c} & \text{otherwise,} \end{cases} \\ \text{and} \quad \bar{\eta} &:= \begin{cases} \frac{\sqrt{16-8\alpha-7\alpha^2}-\alpha}{2L(2+\alpha)} & \text{if } 2\tau^2 \leq n, \\ \frac{\sqrt{16-8\alpha-(7+4c+4c^2)\alpha^2}-\alpha}{2L[2+(1+c)\alpha]} & \text{otherwise.} \end{cases} \end{aligned} \tag{4.29}$$

Then, the following statement holds:

$$\frac{\rho}{2}\|x_{i_k}^{k+1}-x_{i_k}^k\|^2 \leq \widetilde{V}_\eta^k(\bar{x}^k) - \widetilde{V}_\eta^{k+1}(\bar{x}^{k+1}), \tag{4.30}$$

where

$$\rho := \begin{cases} \frac{2(1-\alpha)-(2+\alpha)L^2\eta^2-L\alpha\eta}{\alpha\eta n} & \text{if} \quad 2\tau^2 \leq n, \\ \frac{n^2[2(1-\alpha)-(2+\alpha)L^2\eta^2-L\alpha\eta]-\alpha(1+\eta^2L^2)(2\tau^2-n)}{\alpha\eta n^3} & \text{otherwise.} \end{cases}$$

Moreover, ρ *is positive.*

The next lemma, whose proof is in Section 4.5.3.3, bounds the term $\sum_{i=1}^n \mathbb{E}\left[\|\bar{x}^k - x_i^k\|^2\right]$ in oredr to connect with the gradient mapping $\mathbb{E}\big[\|\mathcal{G}_\eta(\bar{x}^k)\|^2\big]$.

Lemma 4.5. *Suppose that Assumptions 4.1, 4.2, and 4.4 hold. Let* $\left\{(x_i^k, y_i^k, \hat{x}_i^k, \bar{x}^k)\right\}$ *be generated by Algorithm 5. Then, we have*

$$\sum_{i=1}^{n} \mathbb{E}\left[\|\bar{x}^k - x_i^k\|^2\right] \leq D \sum_{t=k-\tau}^{k+T} \mathbb{E}\left[\|x_{i_t}^{t+1} - x_{i_t}^t\|^2\right], \tag{4.31}$$

where $D := \frac{8\alpha^2(1+L^2\eta^2)(\tau^2+2Tn\hat{\mathbf{p}}) \ + \ 8n^2(1+L^2\eta^2+T\alpha^2\hat{\mathbf{p}})}{\hat{\mathbf{p}}\alpha^2 n^2}$.

Theorem 4.3 presents the convergence of Algorithm 5, whose analysis is in Section 4.5.3.4.

Theorem 4.3. *Suppose that Assumptions 4.1, 4.2, and 4.4 hold for* (4.2). *Let* $\bar{\alpha}$, $\bar{\eta}$, ρ, *and* D *be given by Lemma 4.4 and Lemma 4.5, respectively. Let* $\{(x_i^k, y_i^k, \bar{x}^k)\}$ *be generated by Algorithm 5 with stepsizes* $\alpha \in (0, \bar{\alpha})$ *and* $\eta \in (0, \bar{\eta})$. *Then, the following bound holds:*

$$\frac{1}{K+1}\sum_{k=0}^{K} \mathbb{E}\big[\|\mathcal{G}_\eta(\bar{x}^k)\|^2\big] \leq \frac{\hat{C}\big[F(x^0) - F^\star\big]}{K+1}, \tag{4.32}$$

where $\hat{C} := \ \frac{2(1+\eta L)^2 D}{n\eta^2\rho} > 0$ *depending on* $n, L, \eta, \alpha, \tau, T,$ *and* $\hat{\mathbf{p}}$.

Let $\tilde{x}_K$ *be selected uniformly at random from* $\{\bar{x}^0, \cdots, \bar{x}^K\}$ *as the output of Algorithm 5. Then, after at most* $K := \mathcal{O}\left(\varepsilon^{-2}\right)$ *iterations,* $\tilde{x}^K$ *is an* ε*-stationary point of* (4.2) *as in Definition 4.1.2.*

Remark 8. From Theorem 4.3, we can see that **asyncFedDR** achieves the same worst-case communication complexity $\mathcal{O}\left(\varepsilon^{-2}\right)$ (up to a constant factor) as **FedDR**, but with smaller range of α and η.

4.4 Numerical Experiments

In this section, we first provide details about experiment settings then present numerous results on synthetic and real dataset when comparing with other related methods.

4.4.1 Experiment setup

To evaluate the performance of **FedDR** and **asyncFedDR**, we conduct multiple experiments using both synthetic and real datasets. Since most existing methods are developed for non-composite problems, we also implement three other methods: **FedAvg**, **FedProx**, and **FedPD**

to compare with in this setting. We use training loss, training accuracy, and test accuracy as our performance metrics. The source code is available at

https://github.com/unc-optimization/FedDR.

Implementation. To compare synchronous algorithms, we reuse the implementation of FedAvg and FedProx in Li et al. (2020b) and implement FedDR and FedPD on top of it. To conduct the asynchronous examples, we implement our algorithms based on the asynchronous framework in Cai (2018). All experiments are run on a Linux-based server with multiple nodes and configuration: 24-core 2.50GHz Intel processors, 30M cache, and 256GB RAM.

Parameter selection. We use the learning rate for local solver (SGD) as reported in Li et al. (2020b) to approximately evaluate $\text{prox}_{\eta f_i}(y_i^k)$ at each worker $i \in [n]$. The learning rates are 0.01 for all synthetic datasets, 0.01 for MNIST, and 0.003 for FEMNIST. We also perform a grid-search over multiple values to select the parameter and stepsizes for FedProx, FedPD and FedDR. In particular, we choose $\mu \in [0.001, 1]$ for FedProx, $\eta \in [1, 1000]$ for FedPD, and $\eta \in [1, 1000]$, $\alpha \in [0, 1.99]$ for FedDR. All algorithms perform local SGD updates with 20 epochs to approximately evaluate $\text{prox}_{\eta f_i}(y_i^k)$ before sending the results to server for [proximal] aggregation. For each dataset, we pick the parameters that achieve the best test accuracy for each algorithm and plot their performance on the chosen parameters.

Training model selection. For all datasets, we use fully-connected neural network as training models. For all synthetic datasets, we use a neural network of size $60 \times 32 \times 10$ where we use the format input size $\times$ hidden layer $\times$ output size. For MNIST, we use a network of size $784 \times 128 \times 10$. For FEMNIST used in the main text, we reuse the dataset from Li et al. (2020b) and a $784 \times 128 \times 26$ model.

Composite examples. We test our algorithm under composite setting where we set $g(x) = 0.01 \|x\|_1$. In the first test, we choose $\eta = 500$, $\alpha = 1.95$ and select the local learning rate (lr) for SGD to approximately evaluate $\text{prox}_{\eta f_i}(y_i^k)$ from the set $\{0.0025, 0.005, 0.0075, 0.01, 0.025\}$ for `synthetic-(0,0)` and $\{0.001, 0.003, 0.005, 0.008, 0.01\}$ for `FEMNIST`. Next, we fix the local learning rate at 0.01 for `synthetic-(0,0)` and 0.003 for `FEMNIST` then adjust the number of local epochs in the set $\{5, 10, 15, 20, 30\}$ to evaluate $\text{prox}_{\eta f_i}(y_i^k)$. Finally, we test our algorithm when changing the total number of workers participating at each communication round $|\mathcal{S}_k|$.

For `synthetic-(0,0)` dataset, we set $|\mathcal{S}_k| \in \{5, 10, 15, 20, 25\}$. For `FEMNIST` dataset, we set $|\mathcal{S}_k| \in \{10, 25, 50, 75, 100\}$.

Asynchronous example. To make the sample size larger for each worker, we generate the FEMNIST dataset using Leaf (Caldas et al., 2018). In the new dataset, there are actually 62 classes instead of 26 classes as used in Li et al. (2020b). Therefore, we denote this dataset as **FEMNIST - 62 classes**. In this new dataset, each worker has sample size ranging from 97 to 356. We implement the communication between server and worker using the distributed package in `Pytorch` [1] as in Cai (2018). There are 21 threads created, one acts as server and 20 others are workers. To simulate the case when workers have different computing power, we add a certain amount of delay at the end of each worker's local update such that the total update time varies between all workers. For **FEMNIST - 62 classes** dataset, the model is a fully-connected neural network of the size $784 \times 128 \times 62$.

4.4.2 Results on non-composite example

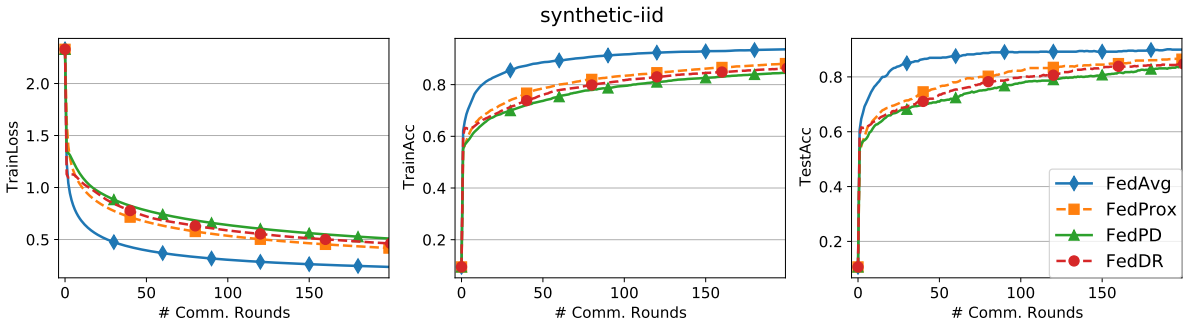


Figure 4.2: The performance of 4 algorithms on iid synthetic dataset without user sampling scheme.

Results on synthetic datasets. We compare four algorithms using synthetic dataset in both iid and non-iid settings. We follow the data generation procedures described in Li et al. (2020b); Shamir et al. (2014) to generate one iid dataset `synthetic-iid` and three non-iid datasets: `synthetic-(`r`,`s`)` for $(r, s) = \{(0, 0), (0.5, 0.5), (1, 1)\}$. We first compare these

[1] see `https://pytorch.org/tutorials/beginner/dist_overview.html` for more details.

algorithms without using the worker sampling scheme, i.e. all workers perform update at each communication round, and for non-composite model of (4.2).

Figure 4.2 illustrates the performance of four algorithms using iid synthetic dataset. From Figure 4.2, FedAvg appears to perform best while the other three algorithms are comparable in the iid setting. Similar behavior is also observed in Li et al. (2020b).

Figure 4.3 depicts the performance of these algorithms using three non-iid synthetic datasets. From Figure 4.3, we observe that the more non-iid the dataset is, the more unstable these algorithms behave. FedDR and FedPD are comparable in these datasets and they both outperform FedProx and FedAvg. FedProx works better than FedAvg which aligns with the results in Li et al. (2020b).

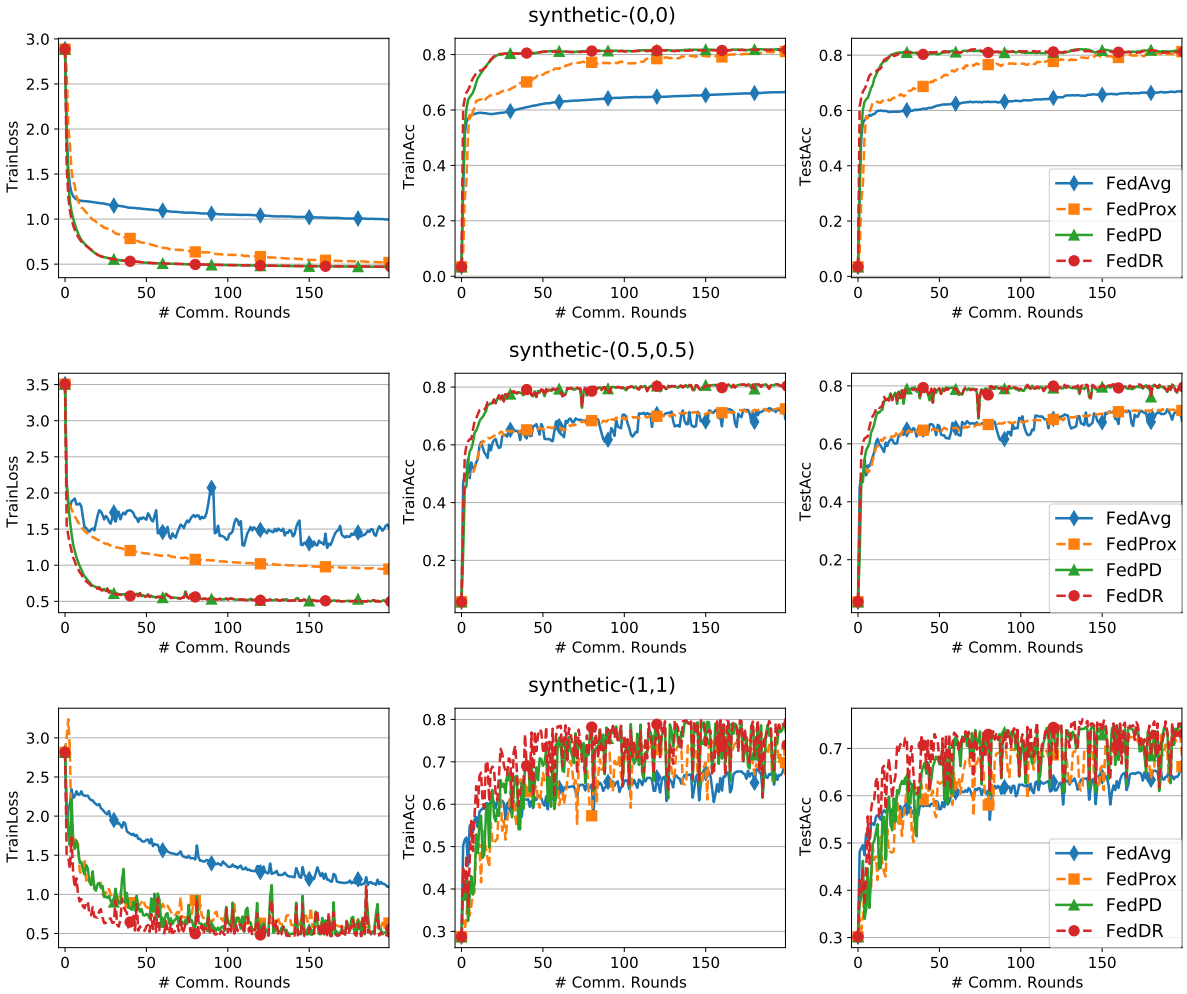


Figure 4.3: The performance of 4 algorithms on non-iid synthetic datasets without worker sampling scheme.

Now we compare these algorithms where we sample 10 workers out of 30 to perform update at each communication round for FedAvg, FedProx, and FedDR while we use all workers for FedPD since FedPD only has convergence guarantee for this setting. In this test, the evaluation metric is plotted in terms of the number of bytes communicated between workers and server at each communication round. Note that using worker sampling scheme in this case can save one-third of communication cost each round. Figure 4.4 depicts the performance of 4 algorithms on one dataset.

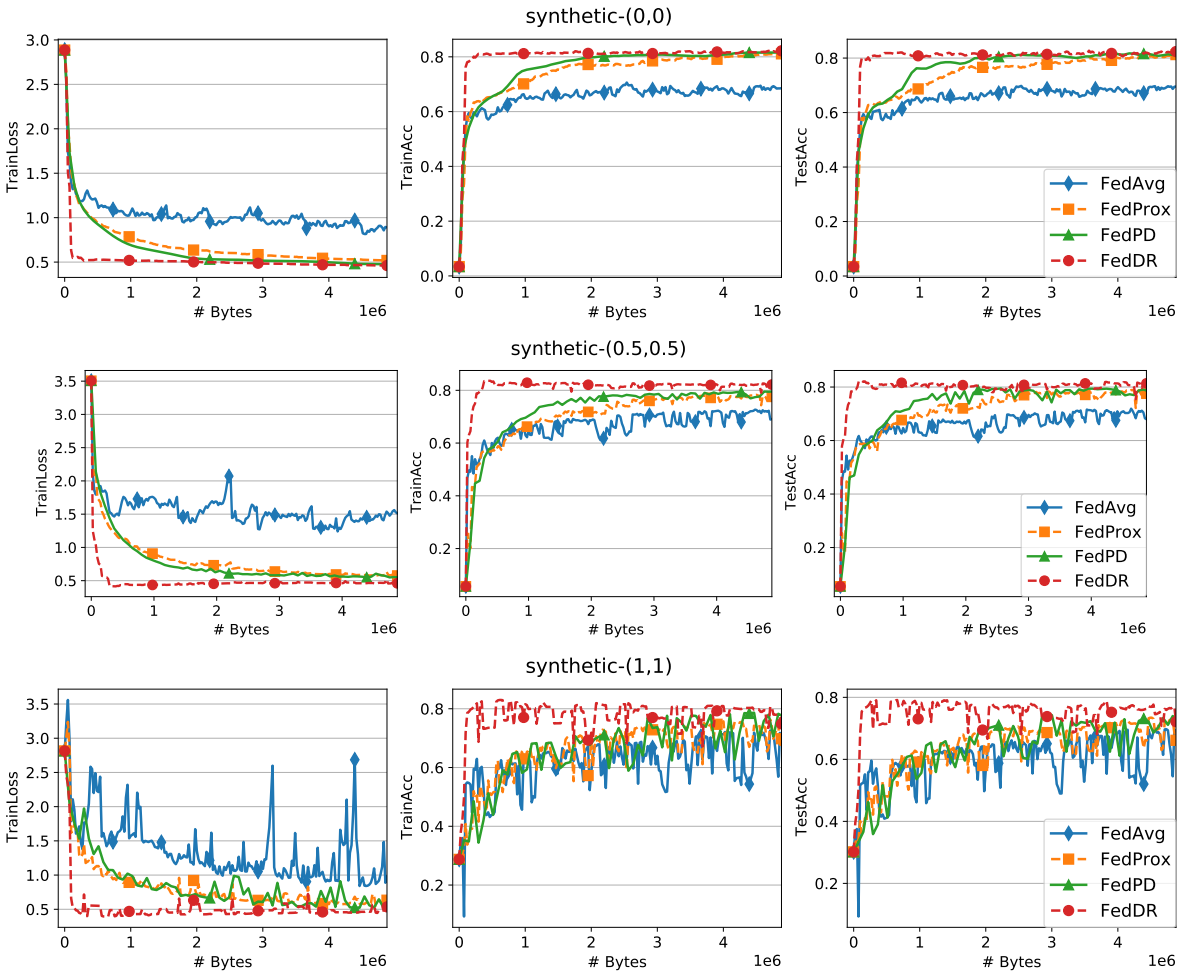


Figure 4.4: The performance of 4 algorithms with worker sampling scheme on non-iid synthetic datasets.

From Figure 4.4, FedDR performs well compared to others. FedProx using worker sampling scheme performs better and is slightly behind FedPD while FedDR, FedPD, and FedProx outperform FedAvg.

Results on FEMNIST datasets. FEMNIST (Caldas et al., 2018) is an extended version of the MNIST dataset (LeCun et al., 1998) where the data is partitioned by the writer of the digit/character. It has a total of 62 classes (10 digits, 26 upper-case and 26 lower-case letters) with over 800,000 samples. In this example, there are total of 200 workers and we sample 50 workers to perform update at each round of communication for FedAvg, FedProx, and FedDR while we use all workers to perform update for FedPD. Figure 4.5 depicts the performance of 4 algorithms in terms of communication cost. From Figure 4.5, FedDR achieves lower loss value and higher training accuracy than other algorithms while FedPD can reach the same test accuracy as ours at the end. Overall, FedDR appears to be better than other algorithms in this test.

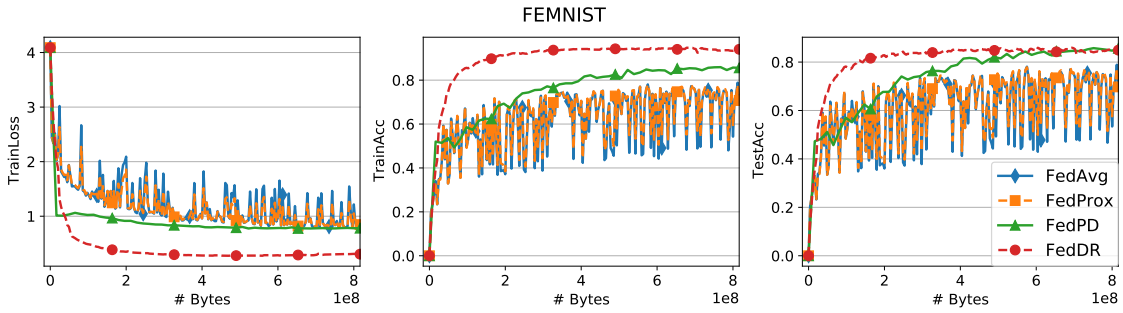


Figure 4.5: The performance of 4 algorithms on the FEMNIST dataset.

4.4.3 Results on composite example using ℓ_1-norm regularizer

We now consider the composite setting with $g(x) := 0.01\,\|x\|_1$ to evaluate the performance of Algorithm 4 on different inexactness levels $\epsilon_{i,k}$ by varying the learning rate (lr) and the number of local SGD epochs to approximately evaluate $\text{prox}_{\eta f_i}(y_i^k)$. We run Algorithm 4 on two datasets: `synthetic-(0,0)` and FEMNIST dataset. The results are shown in Figures 4.6 and 4.7.

From Figure 4.6, we observe that the learning rate (lr) of SGD needs to be tuned for each dataset and the local iteration should be selected carefully to trade-off between local computation cost and inexactness of the evaluation of $\text{prox}_{\eta f_i}(y_i^k)$. From Figure 4.7, Algorithm 4 works best when local learning rate is 0.003 which aligns with Li et al. (2020b) for the non-composite case.

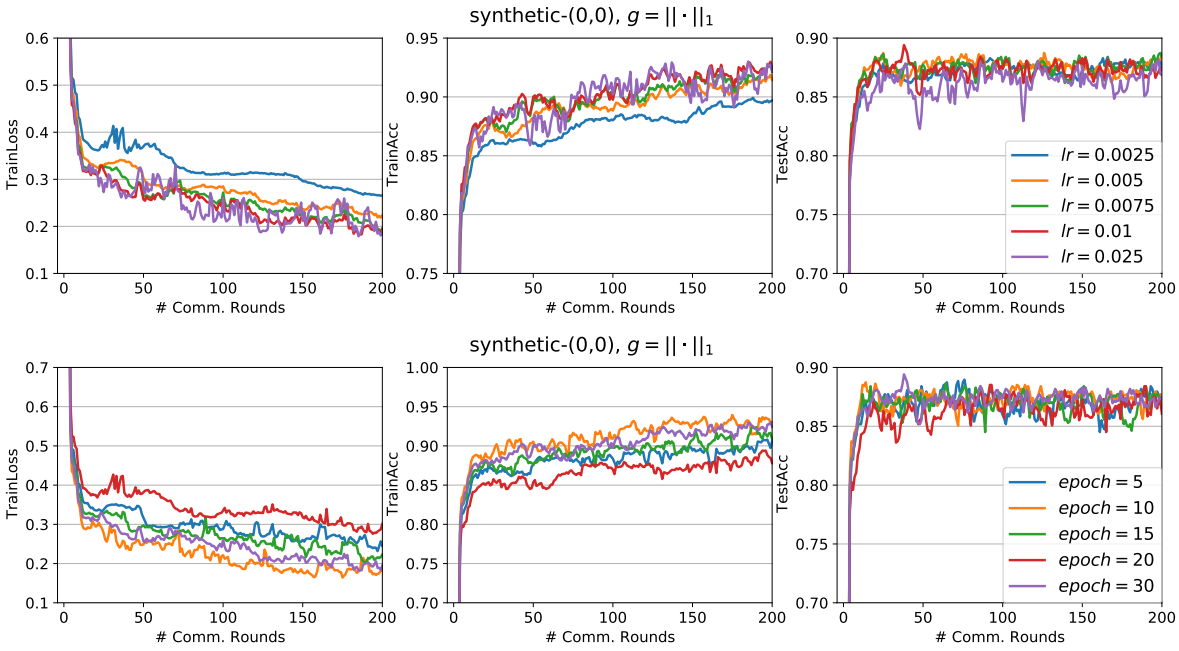


Figure 4.6: The performance of **FedDR** on synthetic dataset in composite setting.

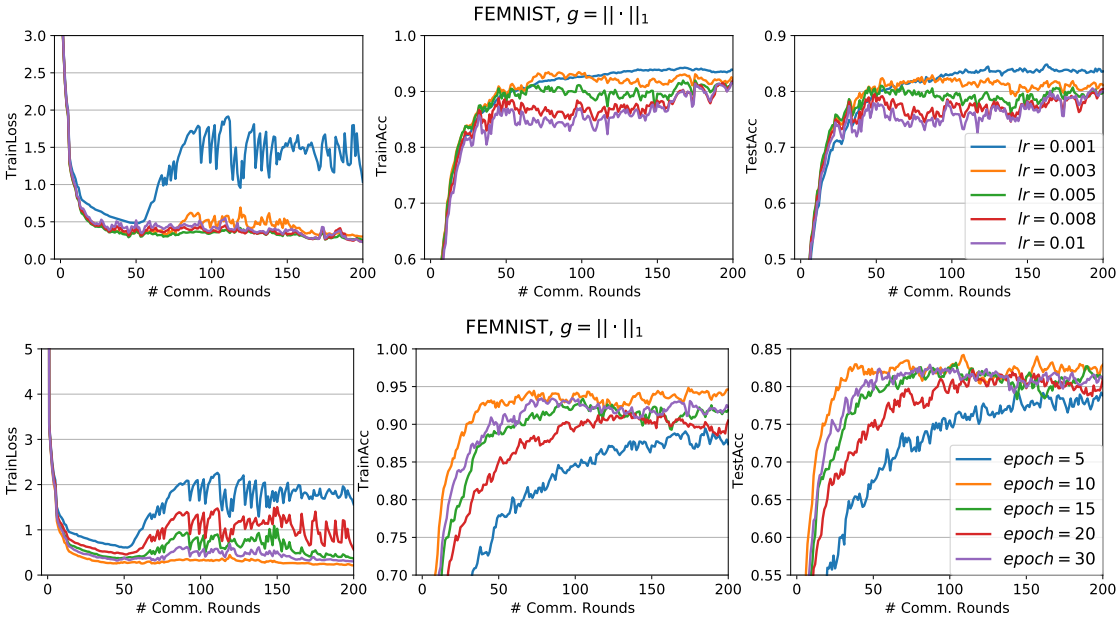


Figure 4.7: The performance of **FedDR** on `FEMNIST` dataset in composite setting.

In addition, it performs better when we decrease $\epsilon_{i,k}$ by increasing the number of epochs when evaluating $\text{prox}_{\eta f_i}$.

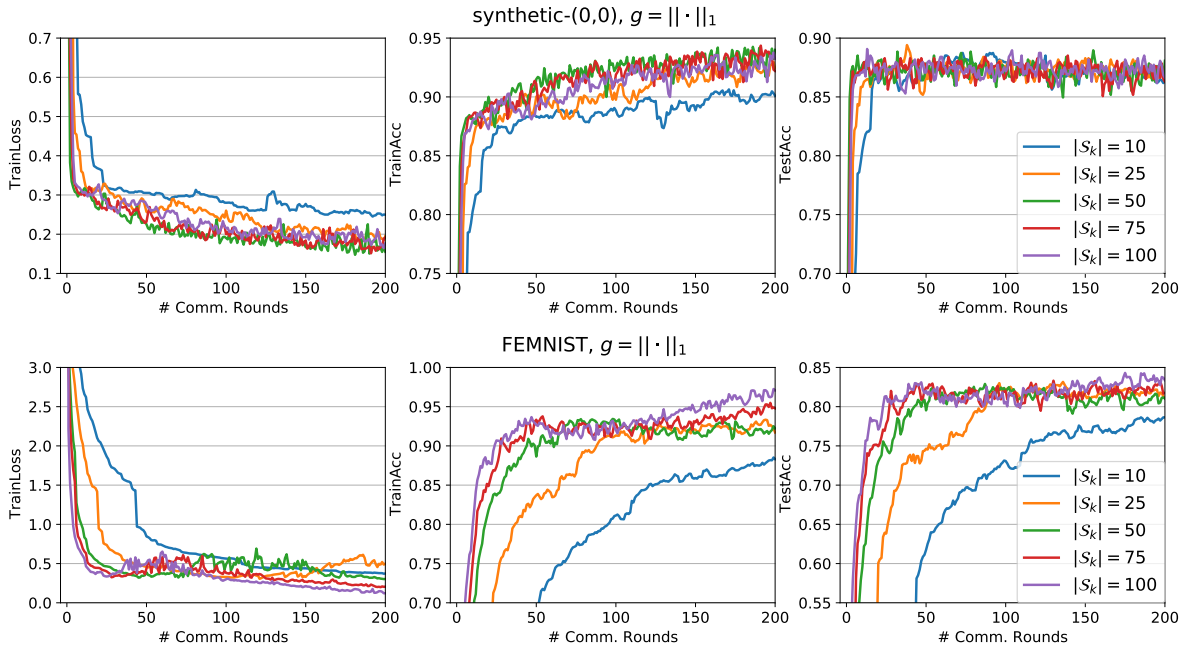


Figure 4.8: The performance of **FedDR** in composite setting in terms of communication rounds.

We also vary the number of users sampled at each communication round. The results are depicted in Figure 4.8 for two datasets. We observe that the performance when we sample smaller number of user per round is not as good as larger ones in terms of communication rounds. However, this might not be a fair comparison since fewer clients also require less communication cost. Therefore, we plot these results in terms of number of bytes communicated. The results are depicted in Figure 4.9. From Figure 4.9, FedDR performs very similarly under different choices of $\mathcal{S}_k$.

4.4.4 Results using asynchronous update

To illustrate the advantage of asyncFedDR over FedDR, we conduct another example to train two datasets: MNIST and FEMNIST using 16 workers. Since we run these experiments on computing nodes with identical configurations, we simulate the case with computing power discrepancy between workers by adding variable delay to each worker's update process such that the difference between the fastest worker may be up to twice as fast as the slowest one.

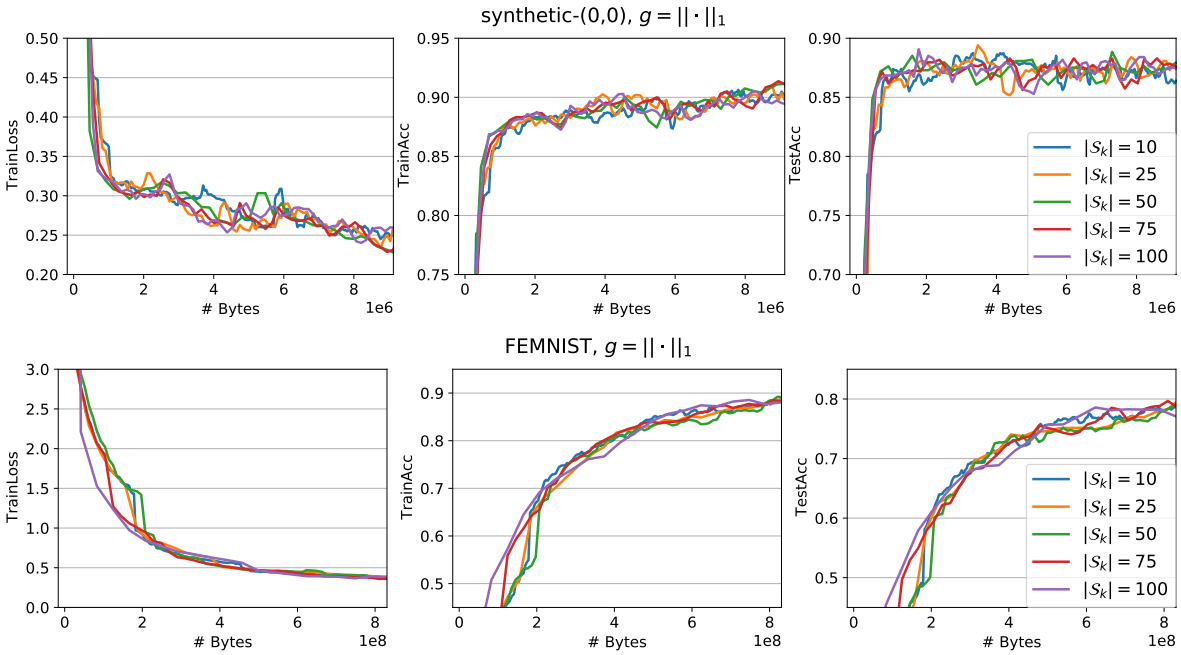


Figure 4.9: The performance of **FedDR** in composite setting in terms of number of bytes.

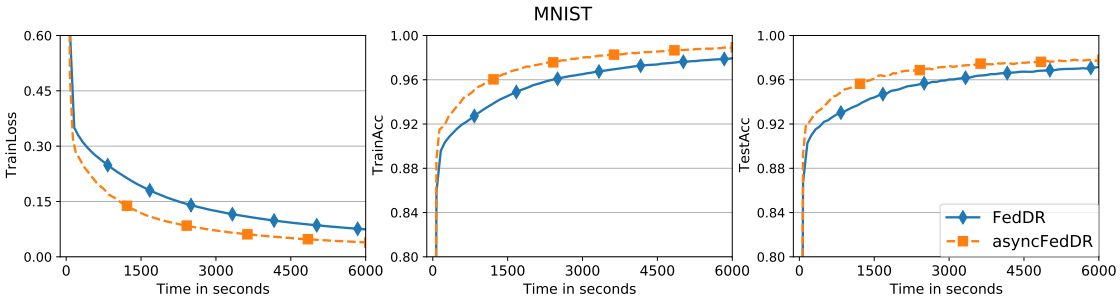


Figure 4.10: The performance of **FedDR** and **asyncFedDR** on the MNIST dataset.

The results of two variants are presented in Figures 4.10 and 4.11. We can see that asyncFedDR can achieve better performance than FedDR in terms of training time which illustrate the advantage of asynchronous update in heterogeneous computing power.

4.5 Appendix

In this appendix, we present intermediate results and missing proof for achieve convergence guarantee of **FedDR** and **asyncFedDR** algorithms in Sections 4.2 and 4.3.

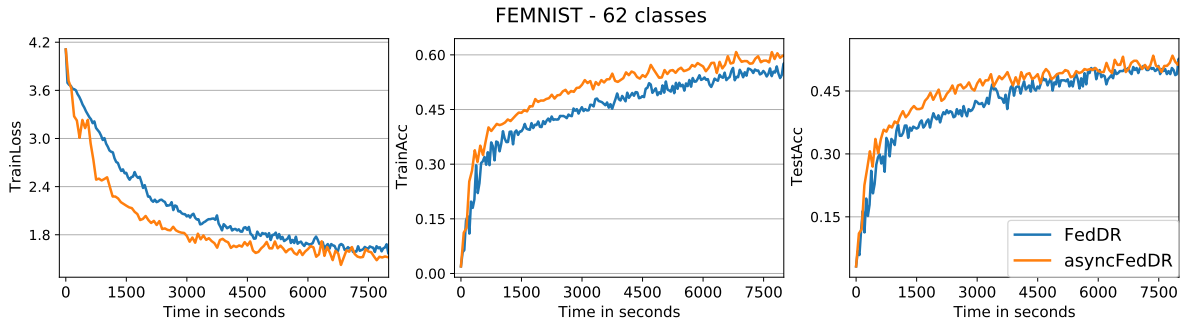


Figure 4.11: The performance of FedDR and asyncFedDR on **FEMNIST - 62 classes** dataset.

4.5.1 Convergence analysis of FedDR

In this section, we introduce several useful lemmas and provide the missing proofs of key results in Section 4.2.

4.5.1.1 Useful lemmas

We provide useful lemmas to show the convergence of FedDR. We first present a useful lemma to characterize the relationship between x_i^k and y_i^k for all iteration k.

Lemma 4.6. *Let* $\{(y_i^k, x_i^k, z_i^k)\}$ *be generated by Algorithm 4 and* (4.13) *starting from* $z_i^0 := \text{prox}_{\eta f_i}(y_i^0)$ *for all* $i \in [n]$ *as in* (4.13). *Then, for all* $i \in [n]$ *and* $k \geq 0$, *we have*

$$y_i^k = z_i^k + \eta \nabla f_i(z_i^k), \quad \text{and} \quad \hat{x}_i^k = 2x_i^k - y_i^k. \tag{4.33}$$

Proof. We prove (4.33) by induction. For $k = 0$, due to the initialization step, Step 1 of Algorithm 4 and (4.13) with $z_i^0 := \text{prox}_{\eta f_i}(y_i^0)$, we have $y_i^0 = z_i^0 + \eta \nabla f_i(z_i^0)$ and $\hat{x}_i^0 = 2x_i^0 - y_i^0$ as in (4.33).

Suppose that (4.33) holds for all $k \geq 0$, i.e., $y_i^k = z_i^k + \eta \nabla f_i(z_i^k)$ and $\hat{x}_i^k = 2x_i^k - y_i^k$. We will show that (4.33) holds for $k+1$, i.e. $y_i^{k+1} = z_i^{k+1} + \eta \nabla f_i(z_i^{k+1})$ and $\hat{x}_i^{k+1} = 2x_i^{k+1} - y_i^{k+1}$ for all $i \in [n]$, respectively. We have two cases:

- For any worker $i \in \mathcal{S}_k$, from the optimality condition of (4.13), we have

$$\nabla f_i(z_i^{k+1}) + \tfrac{1}{\eta}(z_i^{k+1} - y_i^{k+1}) = 0 \quad \Rightarrow \quad y_i^{k+1} = z_i^{k+1} + \eta \nabla f_i(z_i^{k+1}).$$

Moreover, $\hat{x}_i^{k+1} = 2x_i^{k+1} - y_i^{k+1}$ due to Step 5 of Algoritihm 4.

- For any worker $i \notin \mathcal{S}_k$, since $z_i^{k+1} := z_i^k$ due to (4.13), $x_i^{k+1} = x_i^k$, and $y_i^{k+1} = y_i^k$, we can also write y_i^{k+1} as

$$y_i^{k+1} = y_i^k \overset{(*)}{=} z_i^k + \eta \nabla f_i(z_i^k) = z_i^{k+1} + \eta \nabla f_i(z_i^{k+1}).$$

  Here, $(*)$ follows from our induction assumption. Moreover, for $i \notin \mathcal{S}_k$, we maintain $\hat{x}_i^{k+1} = \hat{x}_i^k$ in Algoritihm 4. By our induction assumption, and $x_i^{k+1} = x_i^k$ and $y_i^{k+1} = y_i^k$, we have $\hat{x}_i^{k+1} = \hat{x}_i^k = 2x_i^k - y_i^k = 2x_i^{k+1} - y_i^{k+1}$.

In summary, both cases above imply that $y_i^{k+1} = z_i^{k+1} + \eta \nabla f_i(z_i^{k+1})$ and $\hat{x}_i^k = 2x_i^k - y_i^k$ hold for all $i \in [n]$, which proves (4.33). □

Our next lemma is to bound $\|\bar{x}^k - x_i^k\|^2$ in terms of $\|x_i^{k+1} - x_i^k\|^2$.

Lemma 4.7. *Let $\{(\bar{x}_i^k, z_i^k, x_i^k)\}$ be generated by Algorithm 4 and (4.13), and $\alpha > 0$. Then, for all $i \in \mathcal{S}_k$ and any $\gamma_1 > 0$, we have*

$$\|\bar{x}^k - x_i^k\|^2 \le \tfrac{2(1+\eta^2 L^2)}{\alpha^2}\Big[(1+\gamma_1)\|x_i^{k+1} - x_i^k\|^2 + \tfrac{2(1+\gamma_1)}{\gamma_1}\big(\|e_i^{k+1}\|^2 + \|e_i^k\|^2\big)\Big]. \tag{4.34}$$

In particular, if $e_i^k = e_i^{k+1} = 0$, then $\|\bar{x}^k - x_i^k\|^2 \le \frac{2(1+\eta^2 L^2)}{\alpha^2}\|x_i^{k+1} - x_i^k\|^2$.

Proof. From the update of y_i^{k+1} and Lemma 4.6, for $i \in \mathcal{S}_k$, we have

$$\bar{x}^k - x_i^k = \frac{1}{\alpha}(y_i^{k+1} - y_i^k) \overset{(4.33)}{=} \frac{1}{\alpha}(z_i^{k+1} - z_i^k) + \frac{\eta}{\alpha}(\nabla f_i(z_i^{k+1}) - \nabla f_i(z_i^k)).$$

Using this expression and $\|a+b\|^2 \le 2\|a\|^2 + 2\|b\|^2$, we can bound $\|\bar{x}^k - x_i^k\|^2$ for all $i \in \mathcal{S}_k$ as

$$\begin{aligned}
\|\bar{x}^k - x_i^k\|^2 &= \|\tfrac{1}{\alpha}(z_i^k - z_i^{k+1}) + \tfrac{\eta}{\alpha}(\nabla f_i(z_i^k) - \nabla f_i(z_i^{k+1}))\|^2 \\
&\le \tfrac{2}{\alpha^2}\|z_i^k - z_i^{k+1}\|^2 + \tfrac{2\eta^2}{\alpha^2}\|\nabla f_i(z_i^k) - \nabla f_i(z_i^{k+1})\|^2 \\
&\le \tfrac{2}{\alpha^2}\|z_i^{k+1} - z_i^k\|^2 + \tfrac{2\eta^2 L^2}{\alpha^2}\|z_i^{k+1} - z_i^k\|^2 \quad \text{(by the } L\text{-smoothness of } f_i) \\
&= \tfrac{2(1+\eta^2 L^2)}{\alpha^2}\|x_i^{k+1} - x_i^k - e_i^{k+1} + e_i^k\|^2 \quad \text{(by (4.13))} \\
&\le \tfrac{2(1+\eta^2 L^2)}{\alpha^2}\Big[(1+\gamma_1)\|x_i^{k+1} - x_i^k\|^2 + \tfrac{2(1+\gamma_1)}{\gamma_1}\big(\|e_i^{k+1}\|^2 + \|e_i^k\|^2\big)\Big].
\end{aligned}$$

Here, we have used Young's inequality twice in the last inequality. This proves (4.34). When $e_i^k = e_i^{k+1} = 0$, we can set $\gamma_1 = 0$ in the above estimate to obtain the last statement. $\square$

We still need to link the norm $\sum_{i=1}^n \|x_i^k - \bar{x}^k\|^2$ to the norm of gradient mapping $\|\mathcal{G}_\eta(\bar{x}^k)\|^2$ as in the following lemma.

Lemma 4.8. *Let $\{(\bar{x}_i^k, x_i^k, z_i^k)\}$ be generated by Algorithm 4 and (4.13), and $\alpha > 0$ and $\mathcal{G}_\eta$ be defined by (1.5). Then, for any $\gamma_2 > 0$, we have*

$$\|\mathcal{G}_\eta(\bar{x}^k)\|^2 \le \frac{1}{n\eta^2}\left\{(1+\eta L)^2 \sum_{i=1}^n \left[(1+\gamma_2)\|x_i^k - \bar{x}^k\|^2 + \frac{(1+\gamma_2)}{\gamma_2}\|e_i^k\|^2\right]\right\}. \tag{4.35}$$

In particular, if $e_i^k = 0$ for all $i \in [n]$, then we have $\|\mathcal{G}_\eta(\bar{x}^k)\|^2 \le \frac{(1+\eta L)^2}{n\eta^2}\sum_{i=1}^n \|x_i^k - \bar{x}^k\|^2$.

Proof. From Step 7 of Algorithm 4 and (4.33), we have

$$\tilde{x}^k \overset{\text{Step 7}}{=} \tfrac{1}{n}\textstyle\sum_{i=1}^n \hat{x}_i^k \overset{(4.33)}{=} \tfrac{1}{n}\sum_{i=1}^n (2x_i^k - y_i^k) \overset{(4.33)}{=} \tfrac{1}{n}\sum_{i=1}^n (2x_i^k - z_i^k - \eta\nabla f_i(z_i^k)). \tag{4.36}$$

From the definition (1.5) of $\mathcal{G}_\eta$ and the update of $\bar{x}^k$, we have

$$\begin{aligned}
\eta\|\mathcal{G}_\eta(\bar{x}^k)\| &\overset{(1.5)}{=} \|\bar{x}^k - \mathrm{prox}_{\eta g}(\bar{x}^k - \eta\nabla f(\bar{x}^k))\| \\
&= \|\mathrm{prox}_{\eta g}(\tilde{x}^k) - \mathrm{prox}_{\eta g}(\bar{x}^k - \eta\nabla f(\bar{x}^k))\| \\
&\le \|\tilde{x}^k - \bar{x}^k + \eta\nabla f(\bar{x}^k)\| \\
&\overset{(4.36)}{=} \tfrac{1}{n}\|\textstyle\sum_{i=1}^n [(2x_i^k - z_i^k - \bar{x}^k) + \eta(\nabla f_i(\bar{x}^k) - \nabla f_i(z_i^k))]\|,
\end{aligned}$$

where we have used the non-expansive property of prox_g in the first inequality and $\nabla f(\bar{x}^k) = \frac{1}{n}\sum_{i=1}^n \nabla f_i(\bar{x}^k)$ in the last line.

Finally, using the L-smoothness of f_i, we can derive from the last inequality that

$$\begin{aligned}
\eta^2\|\mathcal{G}_\eta(\bar{x}^k)\|^2 &\le \tfrac{1}{n^2}\Big[\textstyle\sum_{i=1}^n \big(\|2x_i^k - z_i^k - \bar{x}^k\| + \eta L\|z_i^k - \bar{x}^k\|\big)\Big]^2 \\
&\le \tfrac{1}{n}\textstyle\sum_{i=1}^n \big(\|2x_i^k - z_i^k - \bar{x}^k\| + \eta L\|z_i^k - \bar{x}^k\|\big)^2 \\
&\le \tfrac{1}{n}\textstyle\sum_{i=1}^n \big[(1+\eta L)\|x_i^k - \bar{x}^k\| + (1+\eta L)\|e_i^k\|\big]^2 \\
&\le \tfrac{1}{n}(1+\eta L)^2\textstyle\sum_{i=1}^n \big[(1+\gamma_2)\|x_i^k - \bar{x}^k\|^2 + \frac{(1+\gamma_2)}{\gamma_2}\|e_i^k\|^2\big],
\end{aligned}$$

which proves (4.35), where $\gamma_2 > 0$. Here, we have used Young's inequality in the second and the last inequalities, and $x_i^k = z_i^k + e_i^k$ from (4.13) in the third line. □

The following lemma is key to proof Lemma 4.10.

Lemma 4.9. *Suppose that Assumption 4.1, 4.2, and 4.3 hold. Let* $\{(z_i^k, x_i^k, y_i^k, \hat{x}_i^k, \bar{x}^k)\}$ *be generated by Algorithm 4 and* (4.13)*. Let* V_η^k *be defined by* (4.14)*. Then, for any* $\gamma_3 > 0$*, we have*

$$\begin{aligned} V_\eta^{k+1}(\bar{x}^{k+1}) \leq\ & g(\bar{x}^k) + \tfrac{1}{n}\textstyle\sum_{i=1}^n \left[f_i(x_i^{k+1}) + \langle \nabla f_i(x_i^{k+1}), \bar{x}^k - x_i^{k+1}\rangle + \tfrac{1}{2\eta}\|\bar{x}^k - x_i^{k+1}\|^2\right] \\ & - \tfrac{(1-\gamma_3)}{2\eta}\|\bar{x}^{k+1} - \bar{x}^k\|^2 + \tfrac{(1+\eta^2L^2)}{\gamma_3\eta}E_{k+1}^2, \end{aligned} \tag{4.37}$$

where $E_{k+1}^2 := \frac{1}{n}\sum_{i\notin\mathcal{S}_k}\|e_i^k\|^2 + \frac{1}{n}\sum_{i\in\mathcal{S}_k}\|e_i^{k+1}\|^2$*. If* $E_{k+1} = 0$*, then we allow* $\gamma_3 = 0$*.*

Proof. First, from $\bar{x}^{k+1} = \mathrm{prox}_{\eta g}\big(\tilde{x}^{k+1}\big)$ at Step 7 of Algorithm 4, we have $\frac{1}{\eta}(\tilde{x}^{k+1} - \bar{x}^{k+1}) \in \partial g(\bar{x}^{k+1})$. Using this expression and the convexity of g, we obtain

$$g(\bar{x}^{k+1}) \leq g(\bar{x}^k) + \tfrac{1}{\eta}\langle \tilde{x}^{k+1} - \bar{x}^k, \bar{x}^{k+1} - \bar{x}^k\rangle - \tfrac{1}{\eta}\|\bar{x}^{k+1} - \bar{x}^k\|^2. \tag{4.38}$$

Next, since $y_i^{k+1} = z_i^{k+1} + \eta\nabla f_i(z_i^{k+1})$ due to (4.33) and $x_i^{k+1} = z_i^{k+1} + e_i^{k+1}$ due to (4.13), we have

$$\begin{aligned} x_i^{k+1} + \eta\nabla f_i(x_i^{k+1}) &\overset{(4.13)}{=} z_i^{k+1} + \eta\nabla f_i(z_i^{k+1}) + e_i^{k+1} + \eta(\nabla f_i(x_i^{k+1}) - \nabla f_i(z_i^{k+1})) \\ &\overset{(4.33)}{=} y_i^{k+1} + e_i^{k+1} + \eta\xi_i^{k+1}, \end{aligned} \tag{4.39}$$

where $\xi_i^{k+1} := \nabla f_i(x_i^{k+1}) - \nabla f_i(z_i^{k+1})$. Using this relation, we can derive

$$\begin{aligned} \Delta_{k+1} :=\ & \tfrac{1}{n}\textstyle\sum_{i=1}^n \left[f_i(x_i^{k+1}) + \langle \nabla f_i(x_i^{k+1}), \bar{x}^{k+1} - x_i^{k+1}\rangle + \tfrac{1}{2\eta}\|\bar{x}^{k+1} - x_i^{k+1}\|^2\right] \\ =\ & \tfrac{1}{n}\textstyle\sum_{i=1}^n \left[f_i(x_i^{k+1}) + \langle \nabla f_i(x_i^{k+1}), \bar{x}^k - x_i^{k+1}\rangle + \tfrac{1}{2\eta}\|\bar{x}^k - x_i^{k+1}\|^2\right] \\ & + \tfrac{1}{n\eta}\textstyle\sum_{i=1}^n \langle \bar{x}^k - 2x_i^{k+1} + (x_i^{k+1} + \eta\nabla f_i(x_i^{k+1})), \bar{x}^{k+1} - \bar{x}^k\rangle + \tfrac{1}{2\eta}\|\bar{x}^{k+1} - \bar{x}^k\|^2. \end{aligned}$$

Next, using (4.39), we have

$$\begin{aligned}\Delta_{k+1} &\overset{(4.39)}{=} \tfrac{1}{n}\sum_{i=1}^n\Big[f_i(x_i^{k+1}) + \langle\nabla f_i(x_i^{k+1}), \bar{x}^k - x_i^{k+1}\rangle + \tfrac{1}{2\eta}\|\bar{x}^k - x_i^{k+1}\|^2\Big]\\ &\quad + \tfrac{1}{n\eta}\sum_{i=1}^n\langle\bar{x}^k - 2x_i^{k+1} + y_i^{k+1}, \bar{x}^{k+1} - \bar{x}^k\rangle + \tfrac{1}{2\eta}\|\bar{x}^{k+1} - \bar{x}^k\|^2\\ &\quad + \tfrac{1}{n\eta}\sum_{i=1}^n\langle e_i^{k+1} + \eta\xi_i^{k+1}, \bar{x}^{k+1} - \bar{x}^k\rangle\\ &\overset{\text{Step 7}}{=} \tfrac{1}{n}\sum_{i=1}^n\Big[f_i(x_i^{k+1}) + \langle\nabla f_i(x_i^{k+1}), \bar{x}^k - x_i^{k+1}\rangle + \tfrac{1}{2\eta}\|\bar{x}^k - x_i^{k+1}\|^2\Big]\\ &\quad + \tfrac{1}{\eta}\langle\bar{x}^k - \tilde{x}^{k+1}, \bar{x}^{k+1} - \bar{x}^k\rangle + \tfrac{1}{2\eta}\|\bar{x}^{k+1} - \bar{x}^k\|^2\\ &\quad + \tfrac{1}{n\eta}\sum_{i=1}^n\langle e_i^{k+1} + \eta\xi_i^{k+1}, \bar{x}^{k+1} - \bar{x}^k\rangle.\end{aligned}$$

Summing up this expression and (4.38), and using the definition of V_η^k in (4.14), we get

$$\begin{aligned}V_\eta^{k+1}(\bar{x}^{k+1}) &= \tfrac{1}{n}\sum_{i=1}^n\big[f_i(x_i^{k+1}) + \langle\nabla f_i(x_i^{k+1}), \bar{x}^{k+1} - x_i^{k+1}\rangle + \tfrac{1}{2\eta}\|\bar{x}^{k+1} - x_i^{k+1}\|^2\big] + g(\bar{x}^{k+1})\\ &\le g(\bar{x}^k) + \tfrac{1}{n}\sum_{i=1}^n\Big[f_i(x_i^{k+1}) + \langle\nabla f_i(x_i^{k+1}), \bar{x}^k - x_i^{k+1}\rangle + \tfrac{1}{2\eta}\|\bar{x}^k - x_i^{k+1}\|^2\Big]\\ &\quad - \tfrac{1}{2\eta}\|\bar{x}^{k+1} - \bar{x}^k\|^2 + \tfrac{1}{n\eta}\sum_{i=1}^n\langle e_i^{k+1} + \eta\xi_i^{k+1}, \bar{x}^{k+1} - \bar{x}^k\rangle.\end{aligned}$$

By Young's inequality and $e_i^{k+1} = e_i^k$ for $i \notin \mathcal{S}_k$ due to (4.13), for any $\gamma_3 > 0$, we can estimate

$$\begin{aligned}\mathcal{T}_{[1]} &:= \tfrac{1}{n\eta}\sum_{i=1}^n\langle e_i^{k+1} + \eta\xi_i^{k+1}, \bar{x}^{k+1} - \bar{x}^k\rangle\\ &\le \tfrac{1}{2n\eta}\sum_{i=1}^n\big[\tfrac{1}{\gamma_3}\|e_i^{k+1} + \eta\xi_i^{k+1}\|^2 + \gamma_3\|\bar{x}^{k+1} - \bar{x}^k\|^2\big]\\ &\le \tfrac{\gamma_3}{2\eta}\|\bar{x}^{k+1} - \bar{x}^k\|^2 + \tfrac{1}{n\eta\gamma_3}\sum_{i=1}^n\|e_i^{k+1}\|^2 + \tfrac{\eta}{n\gamma_3}\sum_{i=1}^n\|\nabla f_i(x_i^{k+1}) - \nabla f_i(z_i^{k+1})\|^2\\ &\overset{(4.3)}{\le} \tfrac{\gamma_3}{2\eta}\|\bar{x}^{k+1} - \bar{x}^k\|^2 + \tfrac{(1+\eta^2L^2)}{n\eta\gamma_3}\big[\sum_{i\in\mathcal{S}_k}\|e_i^{k+1}\|^2 + \sum_{i\notin\mathcal{S}_k}\|e_i^k\|^2\big].\end{aligned}$$

Substituting this inequality into the last estimate, we eventually obtain (4.37). However, if $E_{k+1}^2 = 0$, then we can deduce from the above inequality that γ_3 can be set to zero. □

4.5.1.2 Proof of Lemma 4.10

First, using (4.37), we can further derive

$$\begin{aligned}V_\eta^{k+1}(\bar{x}^{k+1}) &\overset{(4.37)}{\le} \tfrac{1}{n}\sum_{i=1}^n\big[f_i(x_i^{k+1}) + \langle\nabla f_i(x_i^{k+1}), \bar{x}^k - x_i^{k+1}\rangle + \tfrac{1}{2\eta}\|\bar{x}^k - x_i^{k+1}\|^2\big]\\ &\quad + g(\bar{x}^k) - \tfrac{(1-\gamma_3)}{2\eta}\|\bar{x}^{k+1} - \bar{x}^k\|^2 + \tfrac{(1+\eta^2L^2)}{\eta\gamma_3}E_{k+1}^2.\end{aligned}$$

Using the fact that only users in $\mathcal{S}_k$ perform update and adding/subtracting x_i^k in the term $\langle \nabla f_i(x_i^{k+1}), \bar{x}^k - x_i^{k+1} \rangle$, we have

$$\begin{aligned} V_\eta^{k+1}(\bar{x}^{k+1}) = & \tfrac{1}{n} \textstyle\sum_{i \in \mathcal{S}_k} f_i(x_i^{k+1}) + \tfrac{1}{n} \sum_{i \in \mathcal{S}_k} \langle \nabla f_i(x_i^{k+1}), x_i^k - x_i^{k+1} \rangle \\ & + \tfrac{1}{n} \textstyle\sum_{i \in \mathcal{S}_k} \langle \nabla f_i(x_i^{k+1}), \bar{x}^k - x_i^k \rangle + \frac{1}{2\eta n} \sum_{i \in \mathcal{S}_k} \|\bar{x}^k - x_i^{k+1}\|^2 \\ & + \tfrac{1}{n} \textstyle\sum_{i \notin \mathcal{S}_k} f_i(x_i^k) + \tfrac{1}{n} \sum_{i \notin \mathcal{S}_k} \langle \nabla f_i(x_i^k), \bar{x}^k - x_i^k \rangle + \frac{1}{2\eta n} \sum_{i \notin \mathcal{S}_k} \|\bar{x}^k - x_i^k\|^2 \\ & + g(\bar{x}^k) - \tfrac{(1-\gamma_3)}{2\eta} \|\bar{x}^{k+1} - \bar{x}^k\|^2 + \tfrac{(1+\eta^2 L^2)}{\eta \gamma_3} E_{k+1}^2, \end{aligned} \tag{4.40}$$

On the other hand, from the L-smoothness of f_i, we have

$$f_i(x_i^{k+1}) + \langle \nabla f(x_i^{k+1}), x_i^k - x_i^{k+1} \rangle \leq f_i(x_i^k) + \frac{L}{2} \|x_i^{k+1} - x_i^k\|^2.$$

Substituting this inequality into (4.40), we can further bound it as

$$\begin{aligned} V_\eta^{k+1}(\bar{x}^{k+1}) \leq & \tfrac{1}{n} \textstyle\sum_{i \in \mathcal{S}_k} f_i(x_i^k) + \frac{L}{2n} \sum_{i \in \mathcal{S}_k} \|x_i^{k+1} - x_i^k\|^2 + \tfrac{1}{n} \sum_{i \in \mathcal{S}_k} \langle \nabla f_i(x_i^{k+1}), \bar{x}^k - x_i^k \rangle \\ & + \tfrac{1}{2\eta n} \textstyle\sum_{i \in \mathcal{S}_k} \|\bar{x}^k - x_i^{k+1}\|^2 + \tfrac{1}{n} \sum_{i \notin \mathcal{S}_k} f_i(x_i^k) + \tfrac{1}{n} \sum_{i \notin \mathcal{S}_k} \langle \nabla f_i(x_i^k), \bar{x}^k - x_i^k \rangle \\ & + \tfrac{1}{2\eta n} \textstyle\sum_{i \notin \mathcal{S}_k} \|\bar{x}^k - x_i^k\|^2 + g(\bar{x}^k) \\ & - \tfrac{(1-\gamma_3)}{2\eta} \|\bar{x}^{k+1} - \bar{x}^k\|^2 + \tfrac{(1+\eta^2 L^2)}{\eta \gamma_3} E_{k+1}^2 \\ = & \tfrac{1}{n} \textstyle\sum_{i=1}^n f_i(x_i^k) + \tfrac{1}{n} \sum_{i=1}^n \langle \nabla f_i(x_i^k), \bar{x}^k - x_i^k \rangle + \frac{L}{2n} \sum_{i \in \mathcal{S}_k} \|x_i^{k+1} - x_i^k\|^2 \\ & + \tfrac{1}{2\eta n} \textstyle\sum_{i \in \mathcal{S}_k} \|\bar{x}^k - x_i^{k+1}\|^2 + \tfrac{1}{n} \sum_{i \in \mathcal{S}_k} \langle \nabla f_i(x_i^{k+1}) - \nabla f_i(x_i^k), \bar{x}^k - x_i^k \rangle \\ & + \tfrac{1}{2\eta n} \textstyle\sum_{i \notin \mathcal{S}_k} \|\bar{x}^k - x_i^k\|^2 + g(\bar{x}^k) \\ & - \tfrac{(1-\gamma_3)}{2\eta} \|\bar{x}^{k+1} - \bar{x}^k\|^2 + \tfrac{(1+\eta^2 L^2)}{\eta \gamma_3} E_{k+1}^2, \end{aligned} \tag{4.41}$$

where we have added and subtracted $\frac{1}{n} \sum_{i \in \mathcal{S}_k} \langle \nabla f_i(x_i^k), \bar{x}^k - x_i^k \rangle$ to obtain the last equality.

Note that

$$\|\bar{x}^k - x_i^{k+1}\|^2 = \|\bar{x}^k - x_i^k\|^2 + 2\langle \bar{x}^k - x_i^k, x_i^k - x_i^{k+1} \rangle + \|x_i^k - x_i^{k+1}\|^2.$$

Using the previous expression in (4.41), we can further derive

$$\begin{aligned}
V_\eta^{k+1}(\bar{x}^{k+1}) \leq\ & g(\bar{x}^k) + \tfrac{1}{n}\textstyle\sum_{i=1}^n \Big[f_i(x_i^k) + \langle \nabla f_i(x_i^k), \bar{x}^k - x_i^k\rangle + \tfrac{1}{2\eta}\|\bar{x}^k - x_i^k\|^2\Big] \\
&+ \tfrac{1}{2\eta n}\textstyle\sum_{i\in\mathcal{S}_k}\|x_i^{k+1} - x_i^k\|^2 + \tfrac{1}{\eta n}\sum_{i\in\mathcal{S}_k}\langle x_i^{k+1} - x_i^k, x_i^k - \bar{x}^k\rangle \\
&+ \tfrac{1}{n}\textstyle\sum_{i\in\mathcal{S}_k}\langle \nabla f_i(x_i^{k+1}) - \nabla f_i(x_i^k), \bar{x}^k - x_i^k\rangle + \tfrac{L}{2n}\sum_{i\in\mathcal{S}_k}\|x_i^{k+1} - x_i^k\|^2 \\
&- \tfrac{(1-\gamma_3)}{2\eta}\|\bar{x}^{k+1} - \bar{x}^k\|^2 + \tfrac{(1+\eta^2L^2)}{\eta\gamma_3}E_{k+1}^2 \qquad (4.42)\\
=\ & V_\eta^k(\bar{x}^k) + \tfrac{1+\eta L}{2\eta n}\textstyle\sum_{i\in\mathcal{S}_k}\|x_i^{k+1} - x_i^k\|^2 + \tfrac{1}{\eta n}\sum_{i\in\mathcal{S}_k}\langle x_i^{k+1} - x_i^k, x_i^k - \bar{x}^k\rangle \\
&- \tfrac{(1-\gamma_3)}{2\eta}\|\bar{x}^{k+1} - \bar{x}^k\|^2 + \tfrac{(1+\eta^2L^2)}{\eta\gamma_3}E_{k+1}^2 \\
&+ \tfrac{1}{n}\textstyle\sum_{i\in\mathcal{S}_k}\langle \nabla f_i(x_i^{k+1}) - \nabla f_i(x_i^k), \bar{x}^k - x_i^k\rangle.
\end{aligned}$$

From the update of y_i^{k+1}, for $i \in \mathcal{S}_k$, and similar to the proof of (4.39), we have

$$\begin{aligned}
x_i^k - \bar{x}^k &= \tfrac{1}{\alpha}(y_i^k - y_i^{k+1}) \\
&\overset{(4.39)}{=} \tfrac{1}{\alpha}(z_i^k - z_i^{k+1}) + \tfrac{\eta}{\alpha}(\nabla f_i(z_i^k) - \nabla f_i(z_i^{k+1})) \\
&= \tfrac{1}{\alpha}(x_i^k - x_i^{k+1}) + \tfrac{\eta}{\alpha}(\nabla f_i(x_i^k) - \nabla f_i(x_i^{k+1})) + \tfrac{1}{\alpha}[(e_i^{k+1} + \eta\xi_i^{k+1}) - (e_i^k + \eta\xi_i^k)] \\
&= \tfrac{1}{\alpha}(x_i^k - x_i^{k+1}) + \tfrac{\eta}{\alpha}(\nabla f_i(x_i^k) - \nabla f_i(x_i^{k+1})) + s_i^k,
\end{aligned}$$

where $s_i^k := \frac{1}{\alpha}[e_i^{k+1} + \eta\xi_i^{k+1} - (e_i^k + \eta\xi_i^k))$ with $\xi_i^k := \nabla f_i(x_i^k) - \nabla f_i(z_i^k)$.

Consequently, using the last expression and the L-smoothness of f_i, we can further bound (4.42) as

$$\begin{aligned}
V_\eta^{k+1}(\bar{x}^{k+1}) \leq\ & V_\eta^k(\bar{x}^k) + \tfrac{(1+\eta L)}{2\eta n}\textstyle\sum_{i\in\mathcal{S}_k}\|x_i^{k+1} - x_i^k\|^2 - \tfrac{1}{\alpha\eta n}\sum_{i\in\mathcal{S}_k}\|x_i^{k+1} - x_i^k\|^2 \\
&- \tfrac{1}{\alpha n}\textstyle\sum_{i\in\mathcal{S}_k}\langle x_i^{k+1} - x_i^k, \nabla f_i(x_i^{k+1}) - \nabla f_i(x_i^k)\rangle + \tfrac{1}{\eta n}\sum_{i\in\mathcal{S}_k}\langle s_i^k, x_i^{k+1} - x_i^k\rangle \\
&+ \tfrac{1}{\alpha n}\textstyle\sum_{i\in\mathcal{S}_k}\langle \nabla f_i(x_i^{k+1}) - \nabla f_i(x_i^k), x_i^{k+1} - x_i^k\rangle \\
&+ \tfrac{\eta}{\alpha n}\textstyle\sum_{i\in\mathcal{S}_k}\|\nabla f_i(x_i^{k+1}) - \nabla f_i(x_i^k)\|^2 + \tfrac{1}{n}\sum_{i\in\mathcal{S}_k}\langle s_i^k, \nabla f_i(x_i^{k+1}) - \nabla f_i(x_i^k)\rangle \\
&- \tfrac{(1-\gamma_3)}{2\eta}\|\bar{x}^{k+1} - \bar{x}^k\|^2 + \tfrac{(1+\eta^2L^2)}{\eta\gamma_3}E_{k+1}^2.
\end{aligned}$$

Simplifying gives

$$\begin{aligned} V_\eta^{k+1}(\bar{x}^{k+1}) \leq\ & V_\eta^k(\bar{x}^k) + \tfrac{\eta}{\alpha n}\textstyle\sum_{i\in\mathcal{S}_k}\|\nabla f_i(x_i^{k+1}) - \nabla f_i(x_i^k)\|^2 + \tfrac{[\alpha(L\eta+1)-2]}{2\alpha\eta n}\textstyle\sum_{i\in\mathcal{S}_k}\|x_i^{k+1}-x_i^k\|^2 \\ & + \tfrac{1}{\eta n}\textstyle\sum_{i\in\mathcal{S}_k}\langle s_i^k, (x_i^{k+1}-x_i^k) + \eta(\nabla f_i(x_i^{k+1}) - \nabla f_i(x_i^k))\rangle \\ & - \tfrac{(1-\gamma_3)}{2\eta}\|\bar{x}^{k+1}-\bar{x}^k\|^2 + \tfrac{(1+\eta^2L^2)}{\eta\gamma_3}E_{k+1}^2. \end{aligned}$$

Using (4.3), we obtain

$$\begin{aligned} V_\eta^{k+1}(\bar{x}^{k+1}) \overset{(4.3)}{\leq}\ & V_\eta^k(\bar{x}^k) + \tfrac{\eta L^2}{\alpha n}\textstyle\sum_{i\in\mathcal{S}_k}\|x_i^{k+1}-x_i^k\|^2 + \tfrac{[\alpha(L\eta+1)-2]}{2\alpha\eta n}\textstyle\sum_{i\in\mathcal{S}_k}\|x_i^{k+1}-x_i^k\|^2 \\ & - \tfrac{(1-\gamma_3)}{2\eta}\|\bar{x}^{k+1}-\bar{x}^k\|^2 + \tfrac{(1+\eta^2L^2)}{\eta\gamma_3}E_{k+1}^2 \\ & + \tfrac{1}{n\eta}\textstyle\sum_{i\in\mathcal{S}}\left[\tfrac{1}{\gamma_4}\|s_i^k\|^2 + 2\gamma_4\|x_i^k - x_i^{k+1}\|^2 + 2\gamma_4\eta^2\|\nabla f_i(x_i^k) - \nabla f_i(x_i^{k+1})\|^2\right] \\ =\ & V_\eta^k(\bar{x}^k) - \tfrac{[2-\alpha(L\eta+1)-2L^2\eta^2]}{2\alpha\eta n}\textstyle\sum_{i\in\mathcal{S}_k}\|x_i^{k+1}-x_i^k\|^2 \\ & + \tfrac{1}{n\gamma_4\eta}\textstyle\sum_{i\in\mathcal{S}}\|s_i^k\|^2 + \tfrac{2\gamma_4(1+L^2\eta^2)}{n\eta}\textstyle\sum_{i\in\mathcal{S}_k}\|x_i^{k+1}-x_i^k\|^2 \\ & - \tfrac{(1-\gamma_3)}{2\eta}\|\bar{x}^{k+1}-\bar{x}^k\|^2 + \tfrac{(1+\eta^2L^2)}{\eta\gamma_3}E_{k+1}^2. \end{aligned}$$

Finally, we bound $\|s_i^k\|^2$ as follows:

$$\begin{aligned} \|s_i^k\|^2 &= \tfrac{1}{\alpha^2}\|e_i^{k+1} + \eta\xi_i^{k+1} - (e_i^k + \eta\xi_i^k)\|^2 \\ &\leq \tfrac{1}{\alpha^2}\left[\|e_i^k\| + \|e_i^{k+1}\| + \eta\|\nabla f_i(x_i^k) - \nabla f_i(z_i^k)\| + \eta\|\nabla f_i(x_i^{k+1}) - \nabla f_i(z_i^{k+1})\|\right]^2 \\ &\leq \tfrac{2(1+\eta L)^2}{\alpha^2}(\|e_i^k\|^2 + \|e_i^{k+1}\|^2). \end{aligned}$$

Substituting this inequality into the last estimate, we obtain (4.43). The last statement follows from the last statement of Lemmas 4.8 and 4.9. □

Now, we prove the following key result, which holds surely for any subset $\mathcal{S}_k$ of $[n]$.

Lemma 4.10 (Sure descent lemma)**.** *Suppose that Assumption 4.1, 4.2, and 4.3 hold. Let* $\{(x_i^k, y_i^k, z_i^k, \hat{x}_i^k, \bar{x}^k)\}$ *be generated by Algorithm 4 and* (4.13)*, and* $V_\eta^k(\cdot)$ *be defined by* (4.14)*.*

Then, the following estimate holds:

$$\begin{aligned} V_\eta^{k+1}(\bar{x}^{k+1}) \le\; & V_\eta^k(\bar{x}^k) - \tfrac{[2-\alpha(L\eta+1)-2L^2\eta^2-4\alpha\gamma_4(1+L^2\eta^2)]}{2\alpha\eta n}\textstyle\sum_{i\in\mathcal{S}_k}\|x_i^{k+1}-x_i^k\|^2 \\ & - \tfrac{(1-\gamma_3)}{2\eta}\|\bar{x}^{k+1}-\bar{x}^k\|^2 + \tfrac{(1+\eta^2L^2)}{\eta\gamma_3}E_{k+1}^2 \\ & + \tfrac{2(1+\eta L)^2}{\gamma_4\eta\alpha^2 n}\textstyle\sum_{i\in\mathcal{S}_k}[\|e_i^k\|^2+\|e_i^{k+1}\|^2], \end{aligned} \tag{4.43}$$

where $E_{k+1}^2 := \frac{1}{n}\sum_{i\notin\mathcal{S}_k}\|e_i^k\|^2 + \frac{1}{n}\sum_{i\in\mathcal{S}_k}\|e_i^{k+1}\|^2$, *and* $\gamma_3,\gamma_4>0$. *In particular, if* $E_{k+1}^2=0$, *then we allow* $\gamma_3=0$, *and if* $e_i^k=e_i^{k+1}=0$ *for all* $i\in\mathcal{S}_k$, *then we allow* $\gamma_4=0$.

Proof. First, to guarantee a descent property in (4.43), we need to choose $\eta>0$ and $\alpha>0$ such that $2-\alpha(L\eta+1)-2L^2\eta^2-4\gamma_4\alpha(1+L^2\eta^2)>0$. We first need α such that $0<\alpha<\frac{2}{1+4\gamma_4}$, the condition for η is

$$0<\eta<\bar{\eta} := \tfrac{\sqrt{(4-\alpha)^2+16\alpha^2\gamma_4(1+4\gamma_4)}-\alpha}{4L(1+2\alpha\gamma_4)}.$$

To guarantee $\bar{\eta}>0$, we need to choose $0<\alpha<\frac{\sqrt{17+64\gamma_4}-1}{4(1+4\gamma_4)}$. Combining both conditions on α, we obtain the first condition for α in (4.16).

Now, to show the boundedness of $V_\eta^k(\bar{x}^k)$ from below, we have

$$\begin{aligned} V_\eta^k(\bar{x}^k) &= g(\bar{x}^k) + \tfrac{1}{n}\textstyle\sum_{i=1}^n\Big[f_i(x_i^k) + \langle\nabla f_i(x_i^k),\bar{x}^k-x_i^k\rangle + \tfrac{1}{2\eta}\|\bar{x}^k-x_i^k\|^2\Big] \\ &\ge g(\bar{x}^k) + \tfrac{1}{n}\textstyle\sum_{i=1}^n\Big[f_i(\bar{x}^k) - \tfrac{L}{2}\|\bar{x}^k-x_i^k\|^2 + \tfrac{1}{2\eta}\|\bar{x}^k-x_i^k\|^2\Big] \;\; (\text{the } L\text{-smoothness of } f_i) \\ &\ge f(\bar{x}^k) + g(\bar{x}^k) + \big(\tfrac{1}{\eta}-L\big)\tfrac{1}{2n}\textstyle\sum_{i=1}^n\|\bar{x}^k-x_i^k\|^2 \\ &\ge F^\star \quad (\text{since } \eta\le\tfrac{1}{L} \text{ and Assumption 4.1}). \end{aligned}$$

Next, from (4.34), we have

$$\frac{\alpha^2}{2(1+L^2\eta^2)(1+\gamma_1)}\sum_{i\in\mathcal{S}_k}\|\bar{x}^k-x_i^k\|^2 \le \sum_{i\in\mathcal{S}_k}\Big[\|x_i^{k+1}-x_i^k\|^2 + \frac{\alpha^2}{(1+L^2\eta^2)\gamma_1}\big(\|e_i^{k+1}\|^2+\|e_i^k\|^2\big)\Big].$$

Moreover, from Assumption 4.3, for a nonnegative random variable W_i^k with $i\in\mathcal{S}_k$, by taking expectation of this random variable w.r.t. $\mathcal{S}_k$ conditioned on $\mathcal{F}_{k-1}$, we have

$$\mathbb{E}\Big[\textstyle\sum_{i\in\mathcal{S}_k}W_i^k \mid \mathcal{F}_{k-1}\Big] = \sum_{\mathcal{S}}\mathbb{P}(\mathcal{S}_k=\mathcal{S})\sum_{i\in\mathcal{S}}W_i^k = \sum_{i=1}^n\sum_{\mathcal{S}:i\in\mathcal{S}}\mathbb{P}(\mathcal{S})W_i^k \overset{\text{Ass. (4.3)}}{=} \sum_{i=1}^n\mathbf{p}_iW_i^k.$$

Using this relation with $W_i^k := \|x_i^k - \bar{x}^k\|^2$, $W_i^k := \|e_i^k\|^2$, and $W_i^k := \|e_i^{k+1}\|^2$, and then combining the results with the last inequality, we can derive that

$$\begin{aligned}\mathbb{E}\left[\sum_{i\in\mathcal{S}_k}\|x_i^{k+1}-x_i^k\|^2 \mid \mathcal{F}_{k-1}\right] \geq\ & \tfrac{\alpha^2}{2(1+L^2\eta^2)(1+\gamma_1)}\sum_{i=1}^n \mathbf{p}_i\|\bar{x}^k - x_i^k\|^2 \\ & - \tfrac{\alpha^2}{(1+L^2\eta^2)\gamma_1}\sum_{i=1}^n \mathbf{p}_i\big(\|e_i^{k+1}\|^2 + \|e_i^k\|^2\big) \\ \geq\ & \tfrac{\hat{\mathbf{p}}\alpha^2}{2(1+L^2\eta^2)(1+\gamma_1)}\sum_{i=1}^n \|\bar{x}^k - x_i^k\|^2 \\ & - \tfrac{\alpha^2}{(1+L^2\eta^2)\gamma_1}\sum_{i=1}^n \big(\|e_i^{k+1}\|^2 + \|e_i^k\|^2\big),\end{aligned} \tag{4.44}$$

where we have used $\hat{\mathbf{p}} := \min_{i\in[n]}\mathbf{p}_i > 0$ in Assumption 4.3 and $\mathbf{p}_i \leq 1$ for all $i \in [n]$.

Taking expectation both sides of (4.43) w.r.t. $\mathcal{S}_k$ conditioned on $\mathcal{F}_{k-1}$, and letting $\gamma_3 := 1$, we get

$$\begin{aligned}\mathbb{E}\left[V_\eta^{k+1}(\bar{x}^{k+1}) \mid \mathcal{F}_{k-1}\right] \leq\ & V_\eta^k(\bar{x}^k) + \tfrac{(1+\eta^2L^2)}{\eta n}\sum_{i=1}^n\left[(1+\mathbf{p}_i)\|e_i^k\|^2 + \mathbf{p}_i\|e_i^{k+1}\|^2)\right] \\ & + \tfrac{2(1+\eta L)^2}{\gamma_4\eta\alpha^2 n}\sum_{i=1}^n \mathbf{p}_i\left[\|e_i^k\|^2 + \|e_i^{k+1}\|^2\right] \\ & - \tfrac{[2-\alpha(L\eta+1)-2L^2\eta^2-4\alpha\gamma_4(1+L^2\eta^2)]}{2\eta\alpha n}\mathbb{E}\left[\sum_{i\in\mathcal{S}_k}\|x_i^k-\bar{x}^k\|^2 \mid \mathcal{F}_{k-1}\right].\end{aligned} \tag{4.45}$$

Here, we have written $E_{k+1}^2 = \frac{1}{n}\sum_{i=1}^n\|e_i^k\|^2 + \frac{1}{n}\sum_{i\in\mathcal{S}_k}\left[\|e_i^k\|^2 + \|e_i^{k+1}\|^2\right]$ and used the fact that $\mathbb{E}\left[\sum_{i\in\mathcal{S}_k}\left[\|e_i^k\|^2 + \|e_i^{k+1}\|^2\right] \mid \mathcal{F}_{k-1}\right] = \sum_{i=1}^n \mathbf{p}_i\left[\|e_i^k\|^2 + \|e_i^{k+1}\|^2\right]$.

Combining (4.47) and (4.48) we obtain

$$\begin{aligned}\mathbb{E}\left[V_\eta^{k+1}(\bar{x}^{k+1}) \mid \mathcal{F}_{k-1}\right] \leq\ & V_\eta^k(\bar{x}^k) + \tfrac{(1+\eta^2L^2)}{\eta(n+1)}\sum_{i=1}^n\|e_i^k\|^2 \\ & + \left[\tfrac{2(1+\eta L)^2}{\gamma_4\eta\alpha^2 n} + \tfrac{(1+\eta^2L^2)}{\eta n}\right]\sum_{i=1}^n \mathbf{p}_i\left[\|e_i^k\|^2 + \|e_i^{k+1}\|^2\right] \\ & + \tfrac{\alpha[2-\alpha(L\eta+1)-2L^2\eta^2-4\alpha\gamma_4(1+L^2\eta^2)]}{2\eta(1+L^2\eta^2)\gamma_1 n}\sum_{i=1}^n\left[\|e_i^k\|^2 + \|e_i^{k+1}\|^2\right] \\ & - \tfrac{\hat{\mathbf{p}}\alpha[2-\alpha(L\eta+1)-2L^2\eta^2-4\alpha\gamma_4(1+L^2\eta^2)]}{4\eta(1+L^2\eta^2)(1+\gamma_1)n}\sum_{i=1}^n\|\bar{x}^k - x_i^k\|^2.\end{aligned}$$

Rearranging terms in the last inequality and using $\mathbf{p}_i \leq 1$ and $\|e_i^k\|^2 \leq \epsilon_{i,k}^2$ for all $i \in [n]$ and $k \geq 0$ from (4.12), we obtain (4.17). Note that if $\epsilon_{i,k} = 0$ for all $i \in [n]$ and $k \geq 0$, then we allow to set $\gamma_1 = \gamma_2 = \gamma_4 = \rho_1 = \rho_2 = 0$ as a consequence of the last statement in Lemma 4.7, Lemma 4.8, and Lemma 4.10. □

4.5.1.3 Proof of Lemma 4.2

First, to guarantee a descent property in (4.43), we need to choose $\eta > 0$ and $\alpha > 0$ such that $2 - \alpha(L\eta + 1) - 2L^2\eta^2 - 4\gamma_4\alpha(1 + L^2\eta^2) > 0$. We first need α such that $0 < \alpha < \frac{2}{1+4\gamma_4}$, the condition for η is

$$0 < \eta < \bar{\eta} := \tfrac{\sqrt{(4-\alpha)^2+16\alpha^2\gamma_4(1+4\gamma_4)}-\alpha}{4L(1+2\alpha\gamma_4)}.$$

To guarantee $\bar{\eta} > 0$, we need to choose $0 < \alpha < \frac{\sqrt{17+64\gamma_4}-1}{4(1+4\gamma_4)}$. Combining both conditions on α, we obtain the first condition for α in (4.16).

Now, to show the boundedness of $V_\eta^k(\bar{x}^k)$ from below, we have

$$\begin{aligned}
V_\eta^k(\bar{x}^k) &= g(\bar{x}^k) + \tfrac{1}{n}\textstyle\sum_{i=1}^n \Big[f_i(x_i^k) + \langle \nabla f_i(x_i^k), \bar{x}^k - x_i^k\rangle + \tfrac{1}{2\eta}\|\bar{x}^k - x_i^k\|^2\Big] \\
&\geq g(\bar{x}^k) + \tfrac{1}{n}\textstyle\sum_{i=1}^n \Big[f_i(\bar{x}^k) - \tfrac{L}{2}\|\bar{x}^k - x_i^k\|^2 + \tfrac{1}{2\eta}\|\bar{x}^k - x_i^k\|^2\Big] \quad (L\text{-smoothness of } f_i) \\
&\geq f(\bar{x}^k) + g(\bar{x}^k) + \big(\tfrac{1}{\eta} - L\big)\tfrac{1}{2n}\textstyle\sum_{i=1}^n \|\bar{x}^k - x_i^k\|^2 \\
&\geq F^\star \quad (\text{since } \eta \leq \tfrac{1}{L} \text{ and Assumption 4.1}).
\end{aligned} \tag{4.46}$$

Next, from (4.34), we have

$$\frac{\alpha^2}{2(1+L^2\eta^2)(1+\gamma_1)}\sum_{i\in\mathcal{S}_k}\|\bar{x}^k - x_i^k\|^2 \leq \sum_{i\in\mathcal{S}_k}\Big[\|x_i^{k+1} - x_i^k\|^2 + \frac{\alpha^2}{(1+L^2\eta^2)\gamma_1}\big(\|e_i^{k+1}\|^2 + \|e_i^k\|^2\big)\Big].$$

Moreover, from Assumption 4.3, for a nonnegative random variable W_i^k with $i \in \mathcal{S}_k$, by taking expectation of this random variable w.r.t. $\mathcal{S}_k$ conditioned on $\mathcal{F}_{k-1}$, we have

$$\begin{aligned}
\mathbb{E}\Big[\textstyle\sum_{i\in\mathcal{S}_k} W_i^k \mid \mathcal{F}_{k-1}\Big] &= \textstyle\sum_{\mathcal{S}} \mathbb{P}(\mathcal{S}_k = \mathcal{S}) \sum_{i\in\mathcal{S}} W_i^k = \sum_{i=1}^n \sum_{\mathcal{S}:i\in\mathcal{S}} \mathbb{P}(\mathcal{S}) W_i^k \\
&\overset{\text{Ass. (4.3)}}{=} \textstyle\sum_{i=1}^n \mathbf{p}_i W_i^k.
\end{aligned}$$

Using this relation with $W_i^k := \|x_i^k - \bar{x}^k\|^2$, $W_i^k := \|e_i^k\|^2$, and $W_i^k := \|e_i^{k+1}\|^2$, and then combining the results with the last inequality, we can derive that

$$\begin{aligned}
\mathbb{E}\Big[\textstyle\sum_{i\in\mathcal{S}_k}\|x_i^{k+1} - x_i^k\|^2 \mid \mathcal{F}_{k-1}\Big] \geq\ & \tfrac{\alpha^2}{2(1+L^2\eta^2)(1+\gamma_1)}\textstyle\sum_{i=1}^n \mathbf{p}_i\|\bar{x}^k - x_i^k\|^2 \\
& - \tfrac{\alpha^2}{(1+L^2\eta^2)\gamma_1}\textstyle\sum_{i=1}^n \mathbf{p}_i\big(\|e_i^{k+1}\|^2 + \|e_i^k\|^2\big).
\end{aligned}$$

Using $\hat{\mathbf{p}} := \min_{i\in[n]} \mathbf{p}_i > 0$ in Assumption 4.3 and $\mathbf{p}_i \le 1$ for all $i \in [n]$, we have

$$\begin{aligned}\mathbb{E}\left[\sum_{i\in\mathcal{S}_k} \|x_i^{k+1} - x_i^k\|^2 \mid \mathcal{F}_{k-1}\right] \;\ge\; & \tfrac{\hat{\mathbf{p}}\alpha^2}{2(1+L^2\eta^2)(1+\gamma_1)} \textstyle\sum_{i=1}^n \|\bar{x}^k - x_i^k\|^2 \\ & - \tfrac{\alpha^2}{(1+L^2\eta^2)\gamma_1} \textstyle\sum_{i=1}^n \left(\|e_i^{k+1}\|^2 + \|e_i^k\|^2\right),\end{aligned} \tag{4.47}$$

Taking expectation both sides of (4.43) w.r.t. $\mathcal{S}_k$ conditioned on $\mathcal{F}_{k-1}$, and letting $\gamma_3 := 1$, we obtain

$$\begin{aligned}\mathbb{E}\left[V_\eta^{k+1}(\bar{x}^{k+1}) \mid \mathcal{F}_{k-1}\right] \;\le\; & V_\eta^k(\bar{x}^k) + \tfrac{(1+\eta^2L^2)}{\eta n} \textstyle\sum_{i=1}^n \left[(1+\mathbf{p}_i)\|e_i^k\|^2 + \mathbf{p}_i\|e_i^{k+1}\|^2)\right] \\ & + \tfrac{2(1+\eta L)^2}{\gamma_4\eta\alpha^2 n} \textstyle\sum_{i=1}^n \mathbf{p}_i\left[\|e_i^k\|^2 + \|e_i^{k+1}\|^2\right] \\ & - \tfrac{[2-\alpha(L\eta+1)-2L^2\eta^2-4\alpha\gamma_4(1+L^2\eta^2)]}{2\eta\alpha n}\mathbb{E}\left[\textstyle\sum_{i\in\mathcal{S}_k} \|x_i^k - \bar{x}^k\|^2 \mid \mathcal{F}_{k-1}\right].\end{aligned} \tag{4.48}$$

Here, we have written $E_{k+1}^2 \le \frac{1}{n}\sum_{i=1}^n \|e_i^k\|^2 + \frac{1}{n}\sum_{i\in\mathcal{S}_k}\left[\|e_i^k\|^2 + \|e_i^{k+1}\|^2\right]$ and used the fact that $\mathbb{E}\left[\sum_{i\in\mathcal{S}_k}\left[\|e_i^k\|^2 + \|e_i^{k+1}\|^2\right] \mid \mathcal{F}_{k-1}\right] = \sum_{i=1}^n \mathbf{p}_i\left[\|e_i^k\|^2 + \|e_i^{k+1}\|^2\right]$.

Combining (4.47) and (4.48) yields

$$\begin{aligned}\mathbb{E}\left[V_\eta^{k+1}(\bar{x}^{k+1}) \mid \mathcal{F}_{k-1}\right] \;\le\; & V_\eta^k(\bar{x}^k) + \tfrac{(1+\eta^2L^2)}{\eta(n+1)} \textstyle\sum_{i=1}^n \|e_i^k\|^2 \\ & + \left[\tfrac{2(1+\eta L)^2}{\gamma_4\eta\alpha^2 n} + \tfrac{(1+\eta^2L^2)}{\eta n}\right] \textstyle\sum_{i=1}^n \mathbf{p}_i\left[\|e_i^k\|^2 + \|e_i^{k+1}\|^2\right] \\ & + \tfrac{\alpha[2-\alpha(L\eta+1)-2L^2\eta^2-4\alpha\gamma_4(1+L^2\eta^2)]}{2\eta(1+L^2\eta^2)\gamma_1 n} \textstyle\sum_{i=1}^n \left[\|e_i^k\|^2 + \|e_i^{k+1}\|^2\right] \\ & - \tfrac{\hat{\mathbf{p}}\alpha[2-\alpha(L\eta+1)-2L^2\eta^2-4\alpha\gamma_4(1+L^2\eta^2)]}{4\eta(1+L^2\eta^2)(1+\gamma_1)n} \textstyle\sum_{i=1}^n \|\bar{x}^k - x_i^k\|^2.\end{aligned}$$

Rearranging terms in the last inequality and using $\mathbf{p}_i \le 1$ and $\|e_i^k\|^2 \le \epsilon_{i,k}^2$ for all $i \in [n]$ and $k \ge 0$ from (4.12), we obtain (4.17).

Note that if $\epsilon_{i,k} = 0$ for all $i \in [n]$ and $k \ge 0$, then we allow to set $\gamma_1 = \gamma_2 = \gamma_4 = \rho_1 = \rho_2 = 0$ as a consequence of the last statement in Lemma 4.7, Lemma 4.8, and Lemma 4.10. □

4.5.1.4 Proof of Theorem 4.1

First, from (4.17), we have

$$\begin{aligned}\tfrac{(1+\eta L)^2(1+\gamma_2)}{n\eta^2}\sum_{i=1}^{n}\|x_i^k-\bar{x}^k\|^2 \leq \tfrac{2(1+\eta L)^2(1+\gamma_2)}{\eta^2\beta}\left[V_\eta^k(\bar{x}^k)-\mathbb{E}\left[V_\eta^{k+1}(\bar{x}^{k+1})\mid\mathcal{F}_{k-1}\right]\right],\\ +\ \tfrac{2(1+\eta L)^2(1+\gamma_2)}{n\eta^2\beta}\sum_{i=1}^{n}(\rho_1\epsilon_{i,k}^2+\rho_2\epsilon_{i,k+1}^2).\end{aligned}$$

Substituting these estimates into (4.35) of Lemma 4.8, we have

$$\begin{aligned}\|\mathcal{G}_\eta(\bar{x}^k)\|^2 \leq\ & \tfrac{2(1+\eta L)^2(1+\gamma_2)}{\eta^2\beta}\left[V_\eta^k(\bar{x}^k)-\mathbb{E}\left[V_\eta^{k+1}(\bar{x}^{k+1})\mid\mathcal{F}_{k-1}\right]\right]\\ &+\ \tfrac{2(1+\eta L)^2(1+\gamma_2)}{n\eta^2\beta}\textstyle\sum_{i=1}^{n}(\rho_1\epsilon_{i,k}^2+\rho_2\epsilon_{i,k+1}^2)+\tfrac{(1+\eta L)^2(1+\gamma_2)}{n\eta^2\gamma_2}\sum_{i=1}^{n}\epsilon_{i,k}^2.\end{aligned}$$

Let us introduce three constants

$$C_1 := \tfrac{2(1+\eta L)^2(1+\gamma_2)}{\eta^2\beta},\quad C_2 := \rho_1 C_1,\quad \text{and}\quad C_3 := \rho_2 C_1 + \tfrac{(1+\eta L)^2(1+\gamma_2)}{\eta^2\gamma_2}.$$

Now, taking the total expectation of the last estimate w.r.t. $\mathcal{F}_k$ and using the definition of C_i $(i=1,2,3)$, we have

$$\mathbb{E}\left[\|\mathcal{G}_\eta(\bar{x}^k)\|^2\right] \leq C_1\left(\mathbb{E}\left[V_\eta^k(\bar{x}^k)\right]-\mathbb{E}\left[V_\eta^{k+1}(\bar{x}^{k+1})\right]\right)+\frac{C_2}{n}\sum_{i=1}^{n}\epsilon_{i,k}^2+\frac{C_3}{n}\sum_{i=1}^{n}\epsilon_{i,k+1}^2.$$

Summing up this inequality from $k:=0$ to $k:=K$, and multiplying the result by $\frac{1}{K+1}$, we get

$$\begin{aligned}\tfrac{1}{K+1}\textstyle\sum_{k=0}^{K}\mathbb{E}\left[\|\mathcal{G}_\eta(\bar{x}^k)\|^2\right] \leq\ & C_1\left(\mathbb{E}\left[V_\eta^0(\bar{x}^0)\right]-\mathbb{E}\left[V_\eta^{K+1}(\bar{x}^{K+1})\right]\right)\\ &+\ \tfrac{1}{n(K+1)}\textstyle\sum_{k=0}^{K}\sum_{i=1}^{n}\left(C_2\epsilon_{i,k}^2+C_3\epsilon_{i,k+1}^2\right).\end{aligned}$$

Furthermore, from the initial condition $x_i^0 := x^0$ and $\bar{x}^0 := x^0$, we have $V_\eta^0(\bar{x}^0) = g(x^0) + \frac{1}{n}\sum_{i=1}^{n} f_i(x^0) = F(x^0)$. In addition, $\mathbb{E}\left[V_\eta^{K+1}(\bar{x}^{K+1})\right] \geq F^\star$ due to (4.46). Consequently, the last estimate becomes

$$\frac{1}{K+1}\sum_{k=0}^{K}\mathbb{E}\left[\|\mathcal{G}_\eta(\bar{x}^k)\|^2\right] \leq \frac{C_1}{K+1}\left[F(x^0)-F^\star\right]+\frac{1}{n(K+1)}\sum_{k=0}^{K}\sum_{i=1}^{n}\left(C_2\epsilon_{i,k}^2+C_3\epsilon_{i,k+1}^2\right),$$

which proves (4.19).

Finally, let $\tilde{x}^K$ be selected uniformly at random from $\{\bar{x}^0, \cdots, \bar{x}^K\}$ as the output of Algorithm 4. Then, from (4.19) and $\frac{1}{n}\sum_{i=1}^{n}\sum_{k=0}^{K+1}\epsilon_{i,k}^2 \leq M$ for all $K \geq 0$, we have

$$\mathbb{E}\left[\|\mathcal{G}_\eta(\tilde{x}^K)\|^2\right] = \frac{1}{K+1}\sum_{k=0}^{K}\mathbb{E}\left[\|\mathcal{G}_\eta(\bar{x}^k)\|^2\right] \leq \frac{C_1\left[F(x^0) - F^\star\right] + (C_2+C_3)M}{K+1}.$$

Consequently, to guarantee $\mathbb{E}\left[\|\mathcal{G}_\eta(\tilde{x}^K)\|^2\right] \leq \varepsilon^2$, from the last estimate we need to choose K such that $\frac{C_1[F(x^0)-F^\star]+(C_2+C_3)M}{K+1} \leq \varepsilon^2$. This condition leads to

$$K+1 \geq \frac{C_1[F(x^0) - F^\star] + (C_2+C_3)M}{\varepsilon^2}.$$

Hence, we can take $K := \left\lfloor \frac{C_1[F(x^0)-F^\star]+(C_2+C_3)M}{\varepsilon^2} \right\rfloor \equiv \mathcal{O}\left(\frac{1}{\varepsilon^2}\right)$ as its lower bound. □

4.5.2 Proof of Theorem 4.2

Firstly, starting from (4.43), using $\alpha = 1$, choosing $\gamma_3 = 1$, and noting that $E_{k+1}^2 := \frac{1}{n}\sum_{i\notin\mathcal{S}_k}\|e_i^k\|^2 + \frac{1}{n}\sum_{i\in\mathcal{S}_k}\|e_i^{k+1}\|^2$, we have

$$\begin{aligned} V_\eta^{k+1}(\bar{x}^{k+1}) \leq\ & V_\eta^k(\bar{x}^k) - \tfrac{[1-L\eta-2L^2\eta^2-4\gamma_4(1+L^2\eta^2)]}{2\eta n}\textstyle\sum_{i\in\mathcal{S}_k}\|x_i^{k+1}-x_i^k\|^2 \\ & + \tfrac{(1+\eta^2L^2)}{\eta n}\left(\textstyle\sum_{i\notin\mathcal{S}_k}\|e_i^k\|^2 + \sum_{i\in\mathcal{S}_k}\|e_i^{k+1}\|^2\right) \\ & + \tfrac{2(1+\eta L)^2}{\gamma_4\eta n}\textstyle\sum_{i\in\mathcal{S}_k}[\|e_i^k\|^2 + \|e_i^{k+1}\|^2]. \end{aligned}$$

If we define $\hat{C} := \max\left\{1+\eta^2L^2, \frac{2(1+\eta L)^2}{\gamma_4}\right\}$, then we can further upper bound this estimate as

$$\begin{aligned} V_\eta^{k+1}(\bar{x}^{k+1}) \ \leq\ & V_\eta^k(\bar{x}^k) - \tfrac{[1-L\eta-2L^2\eta^2-4\gamma_4(1+L^2\eta^2)]}{2\eta n}\textstyle\sum_{i\in\mathcal{S}_k}\|x_i^{k+1}-x_i^k\|^2 \\ & + \tfrac{\hat{C}}{n\eta}\left(\textstyle\sum_{i\notin\mathcal{S}_k}\|e_i^k\|^2 + \sum_{i\in\mathcal{S}_k}\|e_i^{k+1}\|^2\right) + \tfrac{\hat{C}}{n\eta}\textstyle\sum_{i\in\mathcal{S}_k}[\|e_i^k\|^2 + \|e_i^{k+1}\|^2] \\ =\ & V_\eta^k(\bar{x}^k) - \tfrac{[1-L\eta-2L^2\eta^2-4\gamma_4(1+L^2\eta^2)]}{2\eta n}\textstyle\sum_{i\in\mathcal{S}_k}\|x_i^{k+1}-x_i^k\|^2 \\ & + \tfrac{\hat{C}}{n\eta}\left(\textstyle\sum_{i=1}^{n}\|e_i^k\|^2 + 2\sum_{i\in\mathcal{S}_k}\|e_i^{k+1}\|^2\right). \end{aligned}$$

Note that $\sum_{i\in\mathcal{S}_k}\|e_i^{k+1}\|^2 \le \sum_{i=1}^n \|e_i^{k+1}\|^2$, we can further bound

$$\begin{aligned} V_\eta^{k+1}(\bar{x}^{k+1}) &\le V_\eta^k(\bar{x}^k) - \tfrac{[1-L\eta-2L^2\eta^2-4\gamma_4(1+L^2\eta^2)]}{2\eta n}\textstyle\sum_{i\in\mathcal{S}_k}\|x_i^{k+1}-x_i^k\|^2 \\ &\quad + \tfrac{\hat{C}}{n\eta}\left(\textstyle\sum_{i=1}^n\|e_i^k\|^2 + 2\sum_{i=1}^n\|e_i^{k+1}\|^2\right) \\ &\le V_\eta^k(\bar{x}^k) - \tfrac{[1-L\eta-2L^2\eta^2-4\gamma_4(1+L^2\eta^2)]}{2\eta n}\textstyle\sum_{i\in\mathcal{S}_k}\|x_i^{k+1}-x_i^k\|^2 \\ &\quad + \tfrac{2\hat{C}}{n\eta}\textstyle\sum_{i=1}^n\left(\|e_i^k\|^2 + \|e_i^{k+1}\|^2\right). \end{aligned}$$

Rearranging terms and noting that $\mathbb{E}\left[\sum_{i\in\mathcal{S}_k}\|x_i^{k+1}-x_i^k\|^2 \mid \mathcal{F}_{k-1}\right] = \sum_{i=1}^n \mathbf{p}_i\|x_i^{k+1}-x_i^k\|^2$, we obtain from the last estimate that

$$\begin{aligned} \tfrac{[1-L\eta-2L^2\eta^2-4\gamma_4(1+L^2\eta^2)]}{2\eta n}\textstyle\sum_{i=1}^n \mathbf{p}_i\|x_i^{k+1}-x_i^k\|^2 &\le V_\eta^k(\bar{x}^k) - V_\eta^{k+1}(\bar{x}^{k+1}) \\ &\quad + \tfrac{2\hat{C}}{n\eta}\textstyle\sum_{i=1}^n\left(\|e_i^k\|^2 + \|e_i^{k+1}\|^2\right). \end{aligned}$$

Now, taking the total expectation of the last inequality w.r.t. $\mathcal{F}_k$, we have

$$\begin{aligned} &\tfrac{[1-L\eta-2L^2\eta^2-4\gamma_4(1+L^2\eta^2)]}{2\eta n}\ \textstyle\sum_{i=1}^n \mathbf{p}_i\mathbb{E}\left[\|x_i^{k+1}-x_i^k\|^2\right] \\ &\qquad \le \mathbb{E}\left[V_\eta^k(\bar{x}^k)\right] - \mathbb{E}\left[V_\eta^{k+1}(\bar{x}^{k+1})\right] + \tfrac{2\hat{C}}{n\eta}\textstyle\sum_{i=1}^n\mathbb{E}\left[\|e_i^k\|^2 + \|e_i^{k+1}\|^2\right]. \end{aligned}$$

Summing this inequality from $k=0$ to $k=K$, we get

$$\begin{aligned} \tfrac{[1-L\eta-2L^2\eta^2-4\gamma_4(1+L^2\eta^2)]}{2\eta n}\ \textstyle\sum_{k=0}^K\sum_{i=1}^n \mathbf{p}_i\mathbb{E}\left[\|x_i^{k+1}-x_i^k\|^2\right] &\le \mathbb{E}\left[V_\eta^0(\bar{x}^0)\right] - \mathbb{E}\left[V_\eta^{K+1}(\bar{x}^{K+1})\right] \\ + \tfrac{2\hat{C}}{n\eta}\textstyle\sum_{k=0}^K\sum_{i=1}^n\mathbb{E}\left[\|e_i^k\|^2 + \|e_i^{k+1}\|^2\right]. \end{aligned}$$

If we choose $\varepsilon_{i,0}=0$ for $i\in[n]$, then the last estimate reduces to

$$\begin{aligned} \tfrac{[1-L\eta-2L^2\eta^2-4\gamma_4(1+L^2\eta^2)]}{2\eta n}\ \sum_{k=0}^K\sum_{i=1}^n \mathbf{p}_i\mathbb{E}\left[\|x_i^{k+1}-x_i^k\|^2\right] &\le \mathbb{E}\left[V_\eta^0(\bar{x}^0)\right] - \mathbb{E}\left[V_\eta^{K+1}(\bar{x}^{K+1})\right] \\ + \tfrac{4\hat{C}}{n\eta}\sum_{k=0}^K\sum_{i=1}^n\mathbb{E}\left[\|e_i^{k+1}\|^2\right]. \end{aligned} \tag{4.49}$$

From (4.20) in Definition 4.2.1, we have $\|e_i^{k+1}\|^2 = \|x_i^{k+1} - \mathrm{prox}_{\eta f_i}(y_i^{k+1})\|^2 \le \varepsilon_{i,k+1}^2 := \theta_i \|x_i^{k+1} - x_i^k\|^2$. Using this condition in (4.49), we have

$$\begin{aligned}\frac{[1-L\eta-2L^2\eta^2-4\gamma_4(1+L^2\eta^2)]}{2\eta n}\sum_{k=0}^{K}\sum_{i=1}^{n}\mathbf{p}_i\mathbb{E}\left[\|x_i^{k+1}-x_i^k\|^2\right] &\le \mathbb{E}\left[V_\eta^0(\bar{x}^0)\right] - \mathbb{E}\left[V_\eta^{K+1}(\bar{x}^{K+1})\right] \\ &\quad + \frac{4\hat{C}}{n\eta}\sum_{k=0}^{K}\sum_{i=1}^{n}\theta_i\mathbb{E}\left[\|x_i^{k+1}-x_i^k\|^2\right].\end{aligned}$$

Now, we can choose θ_i such that $\theta_i = \hat{\theta}\mathbf{p}_i$ for given $\hat{\theta} > 0$. Plugging this choice of θ_i into the last estimate, we have

$$\begin{aligned}\frac{[1-L\eta-2L^2\eta^2-4\gamma_4(1+L^2\eta^2)]}{2\eta n}\sum_{k=0}^{K}\sum_{i=1}^{n}\mathbf{p}_i\mathbb{E}\left[\|x_i^{k+1}-x_i^k\|^2\right] &\le \mathbb{E}\left[V_\eta^0(\bar{x}^0)\right] - \mathbb{E}\left[V_\eta^{K+1}(\bar{x}^{K+1})\right] \\ &\quad + \frac{4\hat{C}\hat{\theta}}{n\eta}\sum_{k=0}^{K}\sum_{i=1}^{n}\mathbf{p}_i\mathbb{E}\left[\|x_i^{k+1}-x_i^k\|^2\right].\end{aligned}$$

Rearranging terms in the above estimate, we arrive at

$$\frac{[1-L\eta-2L^2\eta^2-4\gamma_4(1+L^2\eta^2)-8\hat{C}\hat{\theta}]}{2\eta n}\sum_{k=0}^{K}\sum_{i=1}^{n}\mathbf{p}_i\mathbb{E}\left[\|x_i^{k+1}-x_i^k\|^2\right] \le \mathbb{E}\left[V_\eta^0(\bar{x}^0)\right] - \mathbb{E}\left[V_\eta^{K+1}(\bar{x}^{K+1})\right].$$

From the initial condition $x_i^0 := x^0$ and $\bar{x}^0 := x^0$, we have $V_\eta^0(\bar{x}^0) = g(x^0) + \frac{1}{n}\sum_{i=1}^n f_i(x^0) = F(x^0)$. In addition, $\mathbb{E}\left[V_\eta^{K+1}(\bar{x}^{K+1})\right] \ge F^\star$ due to (4.46). Using these conditions, the last estimate can be further upper bounded by

$$\frac{\hat{\mathbf{p}}[1-L\eta-2L^2\eta^2-4\gamma_4(1+L^2\eta^2)-8\hat{C}\hat{\theta}]}{2\eta n}\sum_{k=0}^{K}\sum_{i=1}^{n}\mathbb{E}\left[\|x_i^{k+1}-x_i^k\|^2\right] \le F(x^0) - F^\star, \quad (4.50)$$

where we have used $\mathbf{p}_i \ge \hat{\mathbf{p}}$ for all $i \in [n]$.

Now, we need to choose η and $\hat{\theta}$ such that $1 - L\eta - 2L^2\eta^2 - 4\gamma_4(1+L^2\eta^2) - 8\hat{C}\hat{\theta} > 0$. First, we need to choose $\gamma_4 > 0$ and $\hat{\theta} > 0$ such that $1 - 4\gamma_4 - 8\hat{C}\hat{\theta} > 0$. Then, the condition for η is

$$0 < \eta < \bar{\eta} := \frac{\sqrt{1+8(1+2\gamma_4)(1-4\gamma_4-8\hat{C}\hat{\theta})}-1}{4L(1+2\gamma_4)}.$$

Next, we connect the term $\|x_i^{k+1} - x_i^k\|^2$ with $\|\mathcal{G}_\eta(\bar{x}^k)\|^2$ as follows. From (4.34) with $\alpha = 1$ and $\gamma_1 = 1$, we have

$$\frac{1}{4(1+L^2\eta^2)} \sum_{i\in\mathcal{S}_k} \|\bar{x}^k - x_i^k\|^2 \leq \sum_{i\in\mathcal{S}_k} \Big[\|x_i^{k+1} - x_i^k\|^2 + \frac{1}{(1+L^2\eta^2)}\big(\|e_i^{k+1}\|^2 + \|e_i^k\|^2\big)\Big].$$

Taking expecatation w.r.t. $\mathcal{S}_k$ given $\mathcal{F}_{k-1}$, and then taking full expectation, we obtain

$$\begin{aligned}
\tfrac{1}{4(1+L^2\eta^2)} \textstyle\sum_{i=1}^n \mathbf{p}_i \mathbb{E}\left[\|\bar{x}^k - x_i^k\|^2\right] &\leq \textstyle\sum_{i=1}^n \mathbf{p}_i \mathbb{E}\left[\|x_i^{k+1} - x_i^k\|^2\right] \\
&\quad + \tfrac{1}{(1+L^2\eta^2)} \textstyle\sum_{i=1}^n \mathbf{p}_i \mathbb{E}\left[\|e_i^{k+1}\|^2 + \|e_i^k\|^2\right] \\
&\leq \textstyle\sum_{i=1}^n \mathbf{p}_i \mathbb{E}\left[\|x_i^{k+1} - x_i^k\|^2\right] \\
&\quad + \tfrac{1}{(1+L^2\eta^2)} \textstyle\sum_{i=1}^n \mathbb{E}\left[\|e_i^{k+1}\|^2 + \|e_i^k\|^2\right].
\end{aligned}$$

Summing this inequality from $k = 0$ to $k = K$, we get

$$\begin{aligned}
\tfrac{1}{4(1+L^2\eta^2)} \textstyle\sum_{k=0}^K \sum_{i=1}^n \mathbf{p}_i \mathbb{E}\left[\|\bar{x}^k - x_i^k\|^2\right] &\leq \textstyle\sum_{k=0}^K \sum_{i=1}^n \mathbf{p}_i \mathbb{E}\left[\|x_i^{k+1} - x_i^k\|^2\right] \\
&\quad + \tfrac{1}{(1+L^2\eta^2)} \textstyle\sum_{k=0}^K \sum_{i=1}^n \mathbb{E}\left[\|e_i^{k+1}\|^2 + \|e_i^k\|^2\right].
\end{aligned}$$

Using the condition that $\epsilon_{i,0} = 0$, similar to (4.49), we have

$$\begin{aligned}
\tfrac{1}{4(1+L^2\eta^2)} \textstyle\sum_{k=0}^K \sum_{i=1}^n \mathbf{p}_i \mathbb{E}\left[\|\bar{x}^k - x_i^k\|^2\right] &\leq \textstyle\sum_{k=0}^K \sum_{i=1}^n \mathbf{p}_i \mathbb{E}\left[\|x_i^{k+1} - x_i^k\|^2\right] \\
&\quad + \tfrac{2}{(1+L^2\eta^2)} \textstyle\sum_{k=0}^K \sum_{i=1}^n \mathbb{E}\left[\|e_i^{k+1}\|^2\right] \\
&\leq \textstyle\sum_{k=0}^K \sum_{i=1}^n \mathbf{p}_i \mathbb{E}\left[\|x_i^{k+1} - x_i^k\|^2\right] \\
&\quad + \tfrac{2}{(1+L^2\eta^2)} \textstyle\sum_{k=0}^K \sum_{i=1}^n \theta_i \mathbb{E}\left[\|x_i^{k+1} - x_i^k\|^2\right] \\
&\leq \tfrac{1+L^2\eta^2+2\hat{\theta}}{(1+L^2\eta^2)} \textstyle\sum_{k=0}^K \sum_{i=1}^n \mathbf{p}_i \mathbb{E}\left[\|x_i^{k+1} - x_i^k\|^2\right].
\end{aligned}$$

In fact, we can further bound this estimate as

$$\frac{\hat{\mathbf{p}}}{4(1+L^2\eta^2)} \sum_{k=0}^K \sum_{i=1}^n \mathbb{E}\left[\|\bar{x}^k - x_i^k\|^2\right] \leq \frac{1+L^2\eta^2+2\hat{\theta}}{(1+L^2\eta^2)} \sum_{k=0}^K \sum_{i=1}^n \mathbb{E}\left[\|x_i^{k+1} - x_i^k\|^2\right],$$

where we have used $\hat{\mathbf{p}} \leq \mathbf{p}_i \leq 1$. Next, multiply both sides of this inequality by $\frac{8(1+L^2\eta^2)(1+\eta L)^2}{\hat{\mathbf{p}}\eta^2 n}$, we obtain

$$\frac{2(1+\eta L)^2}{n\eta^2}\sum_{k=0}^{K}\sum_{i=1}^{n}\mathbb{E}\left[\|\bar{x}^k - x_i^k\|^2\right] \leq \frac{8(1+L^2\eta^2+2\hat{\theta})(1+\eta L)^2}{\hat{\mathbf{p}}\eta^2 n}\sum_{k=0}^{K}\sum_{i=1}^{n}\mathbb{E}\left[\|x_i^{k+1} - x_i^k\|^2\right]. \quad (4.51)$$

Furthermore, from (4.35), choosing $\gamma_2 = 1$ and summing the result from $k = 0$ to $k = K$, we get

$$\begin{aligned}
\textstyle\sum_{k=0}^{K}\mathbb{E}\left[\|\mathcal{G}_\eta(\bar{x}^k)\|^2\right] &\leq \tfrac{2(1+\eta L)^2}{n\eta^2}\textstyle\sum_{k=0}^{K}\sum_{i=1}^{n}\mathbb{E}\left[\|x_i^k - \bar{x}^k\|^2\right] \\
&\quad + \tfrac{2(1+\eta L)^2}{n\eta^2}\textstyle\sum_{k=0}^{K}\sum_{i=1}^{n}\mathbb{E}\left[\|e_i^k\|^2\right] \\
&\leq \tfrac{2(1+\eta L)^2}{n\eta^2}\textstyle\sum_{k=0}^{K}\sum_{i=1}^{n}\mathbb{E}\left[\|x_i^k - \bar{x}^k\|^2\right] \\
&\quad + \tfrac{2(1+\eta L)^2}{n\eta^2}\textstyle\sum_{k=0}^{K}\sum_{i=1}^{n}\theta_i\mathbb{E}\left[\|x_i^{k+1} - x_i^k\|^2\right] \\
&= \tfrac{2(1+\eta L)^2}{n\eta^2}\textstyle\sum_{k=0}^{K}\sum_{i=1}^{n}\mathbb{E}\left[\|x_i^k - \bar{x}^k\|^2\right] \\
&\quad + \tfrac{2(1+\eta L)^2\hat{\theta}}{n\eta^2}\textstyle\sum_{k=0}^{K}\sum_{i=1}^{n}\mathbf{p}_i\mathbb{E}\left[\|x_i^{k+1} - x_i^k\|^2\right]
\end{aligned} \quad (4.52)$$

where the last equality comes from the fact that $\theta_i = \hat{\theta}\mathbf{p}_i$.

Now, plugging (4.51) into (4.52) and using $\mathbf{p}_i \leq 1$, we can get

$$\begin{aligned}
\textstyle\sum_{k=0}^{K}\mathbb{E}\left[\|\mathcal{G}_\eta(\bar{x}^k)\|^2\right] &\leq \left[\tfrac{8\left[1+L^2\eta^2+2\hat{\theta}\right](1+\eta L)^2}{\hat{\mathbf{p}}\eta^2 n} + \tfrac{2(1+\eta L)^2\hat{\theta}}{n\eta^2}\right]\textstyle\sum_{k=0}^{K}\sum_{i=1}^{n}\mathbb{E}\left[\|x_i^{k+1} - x_i^k\|^2\right] \\
&= \tfrac{2\left[4(1+L^2\eta^2+2\hat{\theta})+\hat{\mathbf{p}}\hat{\theta}\right](1+\eta L)^2}{\hat{\mathbf{p}}n\eta^2}\textstyle\sum_{k=0}^{K}\sum_{i=1}^{n}\mathbb{E}\left[\|x_i^{k+1} - x_i^k\|^2\right].
\end{aligned} \quad (4.53)$$

From the definition of $\widetilde{C}$ in (4.23), we can verify that

$$\frac{\hat{\mathbf{p}}[1 - L\eta - 2L^2\eta^2 - 4\gamma_4(1+L^2\eta^2) - 8\hat{C}\hat{\theta}]}{2\eta n\widetilde{C}} = \frac{2\left[4(1+L^2\eta^2+2\hat{\theta}) + \hat{\mathbf{p}}\hat{\theta}\right](1+\eta L)^2}{\hat{\mathbf{p}}n\eta^2}.$$

Next, multiplying both sides of (4.50) by $\frac{1}{\widetilde{C}}$, and then using (4.53), we obtain

$$\begin{aligned}
\textstyle\sum_{k=0}^{K}\mathbb{E}\left[\|\mathcal{G}_\eta(\bar{x}^k)\|^2\right] &\leq \tfrac{2\left[4(1+L^2\eta^2+\hat{\theta})+\hat{\mathbf{p}}\hat{\theta}\right](1+\eta L)^2}{\hat{\mathbf{p}}n\eta^2}\textstyle\sum_{k=0}^{K}\sum_{i=1}^{n}\mathbb{E}\left[\|x_i^{k+1} - x_i^k\|^2\right] \\
&\overset{(4.50)}{\leq} \widetilde{C}\left[F(x^0) - F^\star\right].
\end{aligned}$$

Finally, multiplying both sides of this inequality by $\frac{1}{K+1}$, we obtain (4.22).

□

4.5.3 Convergence analysis of asyncFedDR

For the asynchronous algorithm, Algorithm 5, the following facts hold.

- For x_i^k and y_i^k updated by Algorithm 5, since $\mathcal{S}_k = \{i_k\}$ and the update of y_i^k and x_i^k remain the same as in Algorithm 4 when the error $e_i^k = 0$, the relation (4.33) remains true, i.e. $y_i^k = x_i^k + \eta \nabla f_i(x_i^k)$ and $\hat{x}_i^k = 2x_i^k - y_i^k$ for all $i \in [n]$ and $k \geq 0$.
- Let $\bar{\mathbf{x}}^{k-d^k} := [\bar{x}^{k-d_1^k}, \bar{x}^{k-d_2^k}, \cdots, \bar{x}^{k-d_n^k}]$ be a delayed copy of the vector $\bar{\mathbf{x}}^k := [\bar{x}^k, \cdots, \bar{x}^k] \in \mathbb{R}^{np}$. Since at each iteration k, there is only one block i_k being updated, as shown in Cannelli et al. (2019); Peng et al. (2016), for all $i \in [n]$, we can write

$$\bar{x}^{k-d_i^k} = \bar{x}^k + \sum_{l \in J_i^k} (\bar{x}^l - \bar{x}^{l+1}), \tag{4.54}$$

  where $J_i^k := \{k - d_i^k, k - d_i^k + 1, \cdots, k-1\} \subseteq \{k-\tau, \cdots, k-1\}$.

These facts will be repeatedly used in the sequel.

4.5.3.1 Proof of Lemma 4.3

Let V_η^k be defined by (4.14). For $(x_i^k, \hat{x}_i^k, y_i^k)$ updated as in Algorithm 5, the results of Lemma 4.6 still hold true. Hence, (4.37) still holds for Algorithm 5 with $\gamma_3 = 0$ and $E_{k+1}^2 = 0$, i.e.:

$$\begin{aligned} V_\eta^{k+1}(\bar{x}^{k+1}) \leq\ & g(\bar{x}^k) + \tfrac{1}{n} \textstyle\sum_{i=1}^n \left[f_i(x_i^{k+1}) + \langle \nabla f_i(x_i^{k+1}), \bar{x}^k - x_i^{k+1} \rangle + \tfrac{1}{2\eta} \|\bar{x}^k - x_i^{k+1}\|^2 \right] \\ & - \tfrac{1}{2\eta} \|\bar{x}^{k+1} - \bar{x}^k\|^2. \end{aligned}$$

Using this inequality, the update of $x_{i_k}^{k+1}$ for $i = i_k$, and $x_i^{k+1} = x_i^k$ for $i \neq i_k$, we can expand

$$\begin{aligned} V_\eta^{k+1}(\bar{x}^{k+1}) \leq\ & g(\bar{x}^k) + \tfrac{1}{n} \textstyle\sum_{i \neq i_k} \left[f_i(x_i^k) + \langle \nabla f_i(x_i^k), \bar{x}^k - x_i^k \rangle + \tfrac{1}{2\eta} \|\bar{x}^k - x_i^k\|^2 \right] \\ & + \tfrac{1}{n} \left[f_{i_k}(x_{i_k}^{k+1}) + \langle \nabla f_{i_k}(x_{i_k}^{k+1}), x_{i_k}^k - x_{i_k}^{k+1} \rangle \right] + \tfrac{1}{n} \langle \nabla f_{i_k}(x_{i_k}^{k+1}), \bar{x}^k - x_{i_k}^k \rangle \\ & + \tfrac{1}{2\eta n} \|\bar{x}^k - x_{i_k}^k + x_{i_k}^k - x_{i_k}^{k+1}\|^2 - \tfrac{1}{2\eta} \|\bar{x}^{k+1} - \bar{x}^k\|^2. \end{aligned} \tag{4.55}$$

Now, by the L-smoothness of f_{i_k}, we have

$$f_{i_k}(x_{i_k}^{k+1}) + \langle \nabla f_{i_k}(x_{i_k}^{k+1}), x_{i_k}^k - x_{i_k}^{k+1} \rangle \leq f_{i_k}(x_{i_k}^k) + \frac{L}{2}\|x_{i_k}^k - x_{i_k}^{k+1}\|^2.$$

Plugging this inequality into (4.55) and expanding the third last term of (4.55), we obtain

$$\begin{aligned}
V_\eta^{k+1}(\bar{x}^{k+1}) &\leq g(\bar{x}^k) + \tfrac{1}{n}\textstyle\sum_{i\neq i_k}\left[f_i(x_i^k) + \langle \nabla f_i(x_i^k), \bar{x}^k - x_i^k\rangle + \frac{1}{2\eta}\|\bar{x}^k - x_i^k\|^2\right] \\
&\quad + \tfrac{1}{n} f_{i_k}(x_{i_k}^k) + \tfrac{L}{2n}\|x_{i_k}^{k+1} - x_{i_k}^k\|^2 + \tfrac{1}{n}\langle \nabla f_{i_k}(x_{i_k}^{k+1}), \bar{x}^k - x_{i_k}^k\rangle \\
&\quad + \tfrac{1}{2\eta n}\|\bar{x}^k - x_{i_k}^k\|^2 + \tfrac{1}{2\eta n}\|x_{i_k}^{k+1} - x_{i_k}^k\|^2 + \tfrac{1}{\eta n}\langle x_{i_k}^{k+1} - x_{i_k}^k, x_{i_k}^k - \bar{x}^k\rangle \\
&\quad - \tfrac{1}{2\eta}\|\bar{x}^{k+1} - \bar{x}^k\|^2 \\
&= g(\bar{x}^k) + \tfrac{1}{n}\textstyle\sum_{i=1}^n\left[f_i(x_i^k) + \langle \nabla f_i(x_i^k), \bar{x}^k - x_i^k\rangle + \frac{1}{2\eta}\|\bar{x}^k - x_i^k\|^2\right] \\
&\quad + \tfrac{(1+\eta L)}{2n\eta}\|x_{i_k}^{k+1} - x_{i_k}^k\|^2 + \tfrac{1}{n}\langle \nabla f_{i_k}(x_{i_k}^{k+1}) - \nabla f_{i_k}(x_{i_k}^k), \bar{x}^k - x_{i_k}^k\rangle \\
&\quad + \tfrac{1}{\eta n}\langle x_{i_k}^{k+1} - x_{i_k}^k, x_{i_k}^k - \bar{x}^k\rangle - \tfrac{1}{2\eta}\|\bar{x}^{k+1} - \bar{x}^k\|^2 \\
&\overset{(4.14)}{=} V_\eta^k(\bar{x}^k) + \tfrac{1}{n}\langle \nabla f_{i_k}(x_{i_k}^{k+1}) - \nabla f_{i_k}(x_{i_k}^k), \bar{x}^{k-d_{i_k}^k} - x_{i_k}^k\rangle \\
&\quad + \tfrac{1}{n}\langle \nabla f_{i_k}(x_{i_k}^{k+1}) - \nabla f_{i_k}(x_{i_k}^k), \bar{x}^k - \bar{x}^{k-d_{i_k}^k}\rangle + \tfrac{(1+L\eta)}{2\eta n}\|x_{i_k}^{k+1} - x_{i_k}^k\|^2 \\
&\quad + \tfrac{1}{\eta n}\langle x_{i_k}^{k+1} - x_{i_k}^k, x_{i_k}^k - \bar{x}^{k-d_{i_k}^k}\rangle + \tfrac{1}{\eta n}\langle x_{i_k}^{k+1} - x_{i_k}^k, \bar{x}^{k-d_{i_k}^k} - \bar{x}^k\rangle \\
&\quad - \tfrac{1}{2\eta}\|\bar{x}^{k+1} - \bar{x}^k\|^2.
\end{aligned} \tag{4.56}$$

From $y_{i_k}^{k+1} := y_{i_k}^k + \alpha(\bar{x}^{k-d_{i_k}^k} - x_{i_k}^k)$ at Step 5 of Algorithm 5 and the relation (4.33), we have

$$\bar{x}^{k-d_{i_k}^k} - x_{i_k}^k = \frac{1}{\alpha}(y_{i_k}^{k+1} - y_{i_k}^k) \overset{(4.33)}{=} \frac{1}{\alpha}(x_{i_k}^{k+1} - x_{i_k}^k) + \frac{\eta}{\alpha}(\nabla f_{i_k}(x_{i_k}^{k+1}) - \nabla f_{i_k}(x_{i_k}^k)). \tag{4.57}$$

This relation leads to

$$\begin{aligned}
\tfrac{1}{n}\langle \nabla f_{i_k}(x_{i_k}^{k+1}) - \nabla f_{i_k}(x_{i_k}^k), \bar{x}^{k-d_{i_k}^k} - x_{i_k}^k\rangle &= \tfrac{1}{\alpha n}\langle \nabla f_{i_k}(x_{i_k}^{k+1}) - \nabla f_{i_k}(x_{i_k}^k), x_{i_k}^{k+1} - x_{i_k}^k\rangle \\
&\quad + \tfrac{\eta}{\alpha n}\|\nabla f_{i_k}(x_{i_k}^{k+1}) - \nabla f_{i_k}(x_{i_k}^k)\|^2,
\end{aligned} \tag{4.58}$$

and

$$\begin{aligned}\tfrac{1}{\eta n}\langle x_{i_k}^{k+1}-x_{i_k}^k, x_{i_k}^k-\bar{x}^{k-d_{i_k}^k}\rangle \;=\; &-\tfrac{1}{\alpha n}\langle \nabla f_{i_k}(x_{i_k}^{k+1})-\nabla f_{i_k}(x_{i_k}^k), x_{i_k}^{k+1}-x_{i_k}^k\rangle\\ &-\tfrac{1}{\eta\alpha n}\|x_{i_k}^{k+1}-x_{i_k}^k\|^2.\end{aligned} \tag{4.59}$$

Substituting (4.58) and (4.59) into (4.56), we obtain

$$\begin{aligned}V_\eta^{k+1}(\bar{x}^{k+1}) \;\leq\; & V_\eta^k(\bar{x}^k)+\tfrac{(1+L\eta)}{2\eta n}\|x_{i_k}^{k+1}-x_{i_k}^k\|^2+\tfrac{\eta}{\alpha n}\|\nabla f_{i_k}(x_{i_k}^{k+1})-\nabla f_{i_k}(x_{i_k}^k)\|^2\\ &+\tfrac{1}{n}\langle \nabla f_{i_k}(x_{i_k}^{k+1})-\nabla f_{i_k}(x_{i_k}^k), \bar{x}^k-\bar{x}^{k-d_{i_k}^k}\rangle+\tfrac{1}{\eta n}\langle x_{i_k}^{k+1}-x_{i_k}^k, \bar{x}^{k-d_{i_k}^k}-\bar{x}^k\rangle\\ &-\tfrac{1}{\eta\alpha n}\|x_{i_k}^{k+1}-x_{i_k}^k\|^2-\tfrac{1}{2\eta}\|\bar{x}^{k+1}-\bar{x}^k\|^2\\ \overset{(4.3)}{\leq}\; & V_\eta^k(\bar{x}^k)+\tfrac{\alpha(L\eta+1)-2}{2\eta\alpha n}\|x_{i_k}^{k+1}-x_{i_k}^k\|^2+\tfrac{\eta L^2}{\alpha n}\|x_{i_k}^{k+1}-x_{i_k}^k\|^2\\ &+\tfrac{1}{n}\langle \nabla f_{i_k}(x_{i_k}^{k+1})-\nabla f_{i_k}(x_{i_k}^k), \bar{x}^k-\bar{x}^{k-d_{i_k}^k}\rangle+\tfrac{1}{\eta n}\langle x_{i_k}^{k+1}-x_{i_k}^k, \bar{x}^{k-d_{i_k}^k}-\bar{x}^k\rangle\\ &-\tfrac{1}{2\eta}\|\bar{x}^{k+1}-\bar{x}^k\|^2.\end{aligned}$$

Next, using Young's inequality twice in the above estimate, we can further expand

$$\begin{aligned}V_\eta^{k+1}(\bar{x}^{k+1}) \;\leq\; & V_\eta^k(\bar{x}^k)+\tfrac{\alpha(L\eta+1)+2L^2\eta^2-2}{2\eta\alpha n}\|x_{i_k}^{k+1}-x_{i_k}^k\|^2+\tfrac{\eta}{2n}\|\nabla f_{i_k}(x_{i_k}^{k+1})-\nabla f_{i_k}(x_{i_k}^k)\|^2\\ &+\tfrac{1}{2\eta n}\|\bar{x}^k-\bar{x}^{k-d_{i_k}^k}\|^2+\tfrac{1}{2\eta n}\|x_{i_k}^{k+1}-x_{i_k}^k\|^2+\tfrac{1}{2\eta n}\|\bar{x}^k-\bar{x}^{k-d_{i_k}^k}\|^2\\ &-\tfrac{1}{2\eta}\|\bar{x}^{k+1}-\bar{x}^k\|^2\\ \overset{(4.3)}{\leq}\; & V_\eta^k(\bar{x}^k)+\tfrac{[\alpha(L\eta+2)+2L^2\eta^2-2]}{2\alpha\eta n}\|x_{i_k}^{k+1}-x_{i_k}^k\|^2+\tfrac{L^2\eta}{2n}\|x_{i_k}^{k+1}-x_{i_k}^k\|^2\\ &+\tfrac{1}{\eta n}\|\bar{x}^k-\bar{x}^{k-d_{i_k}^k}\|^2-\tfrac{1}{2\eta}\|\bar{x}^{k+1}-\bar{x}^k\|^2\\ =\; & V_\eta^k(\bar{x}^k)+\tfrac{[\alpha(L^2\eta^2+L\eta+2)+2L^2\eta^2-2]}{2\alpha\eta n}\|x_{i_k}^{k+1}-x_{i_k}^k\|^2-\tfrac{1}{2\eta}\|\bar{x}^{k+1}-\bar{x}^k\|^2\\ &+\tfrac{1}{n\eta}\|\bar{x}^{k-d_{i_k}^k}-\bar{x}^k\|^2.\end{aligned} \tag{4.60}$$

Using (4.54), we can bound $\|\bar{x}^{k-d_{i_k}^k} - \bar{x}^k\|^2$ as follows:

$$\begin{aligned}
\|\bar{x}^{k-d_{i_k}^k} - \bar{x}^k\|^2 &\overset{(4.54)}{=} \big\|\sum_{l\in J_{i_k}^k}(\bar{x}^l - \bar{x}^{l+1})\big\|^2 \\
&\leq d_{i_k}^k \sum_{l=k-d_{i_k}^k}^{k-1} \|\bar{x}^{l+1} - \bar{x}^l\|^2 \quad \text{(Young's inequality and the definition of } J_{i_k}^k) \\
&\leq \tau \sum_{l=k-\tau}^{k-1} \|\bar{x}^{l+1} - \bar{x}^l\|^2 \quad \text{(since } d_{i_k}^k \leq \tau \text{ in Assumption 4.4)} \qquad (4.61) \\
&= \tau\Big[\sum_{l=k-\tau}^{k-1}[l-(k-\tau)+1]\|\bar{x}^{l+1} - \bar{x}^l\|^2 - \sum_{l=k-\tau+1}^{k}(l-(k-\tau))\|\bar{x}^{l+1} - \bar{x}^l\|^2\Big] \\
&\quad + \tau^2\|\bar{x}^{k+1} - \bar{x}^k\|^2.
\end{aligned}$$

Now, we consider two cases as follows.

Case 1: If $n \geq 2\tau^2$, then by plugging (4.61) into (4.60), we finally arrive at

$$\begin{aligned}
V_\eta^{k+1}(\bar{x}^{k+1}) + \tfrac{\tau}{n\eta}\textstyle\sum_{l=k-\tau+1}^{k}[l-(k-\tau)]\|\bar{x}^{l+1} - \bar{x}^l\|^2 &\leq V_\eta^k(\bar{x}^k) \\
+ \tfrac{\tau}{n\eta}\textstyle\sum_{l=k-\tau}^{k-1}[l-(k-\tau)+1]\|\bar{x}^{l+1} - \bar{x}^l\|^2 & \\
- \tfrac{[2(1-\alpha)-(2+\alpha)\eta^2L^2-\alpha\eta L]}{2\alpha\eta n}\|x_{i_k}^{k+1} - x_{i_k}^k\|^2 - \tfrac{(n-2\tau^2)}{2n\eta}\|\bar{x}^{k+1} - \bar{x}^k\|^2. &
\end{aligned}$$

Rearranging the last estimate, we finally arrive at (4.27).

Case 2: if $2\tau^2 > n$, then using (4.33), we can show that

$$\begin{aligned}
\|\bar{x}^{k+1} - \bar{x}^k\|^2 &= \big\|\mathrm{prox}_{\eta g}\big(\tilde{x}^{k+1}\big) - \mathrm{prox}_{\eta g}\big(\tilde{x}^k\big)\big\|^2 \leq \|\tilde{x}^{k+1} - \tilde{x}^k\|^2 \\
&= \|\tfrac{1}{n}\textstyle\sum_{i=1}^n(\hat{x}_i^{k+1} - \hat{x}_i^k)\|^2 \\
&= \tfrac{1}{n^2}\|\hat{x}_{i_k}^{k+1} - \hat{x}_{i_k}^k\|^2 \quad \text{(since only block } i_k \text{ is updated)} \\
&\overset{(4.33)}{=} \tfrac{1}{n^2}\|(x_{i_k}^{k+1} - x_{i_k}^k) - \eta(\nabla f_{i_k}(x_{i_k}^{k+1}) - \nabla f_{i_k}(x_{i_k}^k))\|^2 \\
&\leq \tfrac{2}{n^2}\|x_{i_k}^{k+1} - x_{i_k}^k\|^2 + \tfrac{2\eta^2}{n^2}\|\nabla f_{i_k}(x_{i_k}^{k+1}) - \nabla f_{i_k}(x_{i_k}^k)\|^2 \\
&\leq \tfrac{2(1+\eta^2L^2)}{n^2}\|x_{i_k}^{k+1} - x_{i_k}^k\|^2.
\end{aligned} \qquad (4.62)$$

Substituting this inequality into the previous one, we can get

$$\begin{aligned} V_\eta^{k+1}(\bar{x}^{k+1}) \; + \; & \tfrac{\tau}{n\eta}\textstyle\sum_{l=k-\tau+1}^{k}[l-(k-\tau)]\|\bar{x}^{l+1}-\bar{x}^{l}\|^2 \le V_\eta^{k}(\bar{x}^{k}) \\ & + \; \tfrac{\tau}{n\eta}\textstyle\sum_{l=k-\tau}^{k-1}[l-(k-\tau)+1]\|\bar{x}^{l+1}-\bar{x}^{l}\|^2 \\ & - \; \left[\tfrac{2(1-\alpha)-(2+\alpha)\eta^2L^2-\alpha\eta L}{2\alpha\eta n} - \tfrac{(1+\eta^2L^2)(2\tau^2-n)}{2n^3\eta}\right]\|x_{i_k}^{k+1}-x_{i_k}^{k}\|^2. \end{aligned}$$

Simplifying the coefficients of this estimate, we finally arrive at (4.27). □

4.5.3.2 Proof of Lemma 4.4

If we define $\widetilde{V}_\eta^k$ as in (4.28) of Lemma 4.4, i.e.:

$$\widetilde{V}_\eta^k(\bar{x}^k) := V_\eta^k(\bar{x}^k) + \frac{\tau}{\eta n^2}\sum_{l=k-\tau}^{k-1}[l-(k-\tau)+1]\|\bar{x}^{l+1}-\bar{x}^{l}\|^2,$$

then from (4.27), we have

$$\widetilde{V}_\eta^{k+1}(\bar{x}^{k+1}) \le \widetilde{V}^k(\bar{x}^k) - \frac{\rho}{2}\|x_{i_k}^{k+1}-x_{i_k}^{k}\|^2,$$

which is equivalent to (4.30).

Now, we find conditions of α and η such that ρ and θ are positive. We consider two cases as follows.

Case 1: If $2\tau^2 \le n$, then

$$\rho := \tfrac{2(1-\alpha)-(2+\alpha)L^2\eta^2-L\alpha\eta}{\alpha\eta n}.$$

Let us choose $0<\alpha<1$. To guarantee $\rho>0$, we require $2(1-\alpha) > (2+\alpha)L^2\eta^2 + L\alpha\eta$. In this case, we need to choose $0<\eta<\frac{\sqrt{L^2\alpha^2+8(1-\alpha)(2+\alpha)L^2}-L\alpha}{2L^2(2+\alpha)} = \frac{\sqrt{16-8\alpha-7\alpha^2}-\alpha}{2L(2+\alpha)}$. These are the choices in (4.29) when $2\tau^2 \le n$.

Case 2: If $2\tau^2 > n$, then

$$\rho := \tfrac{n^2[2(1-\alpha)-(2+\alpha)L^2\eta^2-L\alpha\eta]-\alpha(1+\eta^2L^2)(2\tau^2-n)}{\alpha\eta n^3}.$$

Let $c := \frac{2\tau^2-n}{n^2} > 0$. In order to guarantee that $\rho > 0$, we need to choose $0 < \alpha < 1$ and $\eta > 0$ such that

$$2 - 2\alpha - \tfrac{\alpha(2\tau^2-n)}{n^2} > \big[2 + \alpha + \tfrac{\alpha(2\tau^2-n)}{n^2}\big]L^2\eta^2 + L\alpha\eta,$$

$$\text{and} \quad 0 < \alpha < \tfrac{2n^2}{2n^2+(2\tau^2-n)} = \tfrac{2}{2+c}.$$

Using the definition of c, the first condition becomes $2 - 2\alpha - c\alpha > L\alpha\eta + (2 + \alpha + c\alpha)L^2\eta^2$. First, we need to impose $2 - 2\alpha - c\alpha > 0$, leading to $0 < \alpha < \frac{2}{2+c}$. Next, we solve the above inequality w.r.t. $\eta > 0$ to get

$$0 < \eta < \bar{\eta} := \tfrac{\sqrt{16-8\alpha-(7+4c+4c^2)\alpha^2}-\alpha}{2L[2+(1+c)\alpha]}.$$

These are the choices in (4.29) when $2\tau^2 > n$. To guarantee $\bar{\eta} > 0$, we need to choose $\alpha < \frac{4}{1+\sqrt{1+4(2+c+c^2)}}$. Combining four conditions of α, we get $0 < \alpha < \frac{2}{2+c}$. Finally, we conclude that under the choice of α and η as in (4.29), we have $\rho > 0$ and $\theta > 0$. □

4.5.3.3 Proof of Lemma 4.5

Let $t_k(i) := \min\big\{t \in \{0, \cdots, T\} : \mathbf{p}(i \mid \xi^{0:k+t-1}) \geq \hat{\mathbf{p}}\big\}$. In fact, $t_k(i)$ is the first time in the iteration window $[k, k+T]$, worker i is active, i.e. gets updated. For any $\gamma \in (0,1)$, we have

$$\begin{aligned}
\textstyle\sum_{t=k}^{k+T} \mathbb{E}\left[\|\bar{x}^t - x_{\hat{i}_t}^t\|^2 \mid \mathcal{F}_{t-1}\right](\omega) &= \textstyle\sum_{t=k}^{k+T}\sum_{i=1}^n \mathbf{p}(i \mid \xi^{0:t-1})\|\bar{x}^t - x_i^t\|^2 \\
&\overset{(4.25)}{\geq} \textstyle\sum_{i=1}^n \hat{\mathbf{p}}\|\bar{x}^{k+t_k(i)} - x_i^{k+t_k(i)}\|^2 \\
&\overset{(*)}{\geq} \hat{\mathbf{p}}\textstyle\sum_{i=1}^n \Big[\|\bar{x}^k - x_i^k\| - \|\bar{x}^{k+t_k(i)} - x_i^{k+t_k(i)} - (\bar{x}^k - x_i^k)\|\Big]^2 \\
&\geq -2\hat{\mathbf{p}}\textstyle\sum_{i=1}^n \|\bar{x}^k - x_i^k\|\|\bar{x}^{k+t_k(i)} - x_i^{k+t_k(i)} - (\bar{x}^k - x_i^k)\| \\
&\quad + \hat{\mathbf{p}}\textstyle\sum_{i=1}^n \|\bar{x}^k - x_i^k\|^2 \\
&\overset{(**)}{\geq} \hat{\mathbf{p}}\textstyle\sum_{i=1}^n \big[\|\bar{x}^k - x_i^k\|^2 - \tfrac{1}{2}\|\bar{x}^k - x_i^k\|^2\big] \\
&\quad - 4\hat{\mathbf{p}}\textstyle\sum_{i=1}^n \|\bar{x}^{k+t_k(i)} - \bar{x}^k\|^2 - 4\hat{\mathbf{p}}\textstyle\sum_{i=1}^n \|x_i^{k+t_k(i)} - x_i^k\|^2,
\end{aligned}$$

where (*) comes from the reverse triangle inequality $\|a-b\|^2 \geq (\|a\| - \|b\|)^2$ and (**) is due to $4\|v\|^2 + 4\|s\|^2 + \frac{1}{2}\|u\|^2 \geq 2\|u\|\|v+s\|$. Note that the conditional expectation above is only taken w.r.t. $\hat{i}_k$, which is $\sigma(d^k, \mathcal{F}_{k-1})$-measurable. For simplicity of notation, we drop (ω) in the sequel.

Rearranging the last inequality, we obtain

$$\begin{aligned}\frac{\hat{\mathbf{p}}}{2}\sum_{i=1}^n \|\bar{x}^k - x_i^k\|^2 \leq \sum_{t=k}^{k+T} \mathbb{E}\left[\|\bar{x}^t - x_{\hat{i}_t}^t\|^2 \mid \mathcal{F}_{t-1}\right] + 4\hat{\mathbf{p}}\sum_{i=1}^n \|\bar{x}^{k+t_k(i)} - \bar{x}^k\|^2 \\ + 4\hat{\mathbf{p}}\sum_{i=1}^n \|x_i^{k+t_k(i)} - x_i^k\|^2.\end{aligned} \tag{4.63}$$

Next, we bound the term $\sum_{i=1}^n \|\bar{x}^{k+t_k(i)} - \bar{x}^k\|^2$ as follows:

$$\begin{aligned}\sum_{i=1}^n \|\bar{x}^{k+t_k(i)} - \bar{x}^k\|^2 &= \sum_{i=1}^n \|\sum_{t=k}^{k+t_k(i)-1}(\bar{x}^{t+1} - \bar{x}^t)\|^2 \\ &\leq \sum_{i=1}^n t_k(i)\sum_{t=k}^{k+t_k(i)-1}\|\bar{x}^{t+1} - \bar{x}^t\|^2 \qquad \text{(Young's inequality)} \\ &\leq T\sum_{i=1}^n \sum_{t=k}^{k+t_k(i)-1}\|\bar{x}^{t+1} - \bar{x}^t\|^2 \qquad \text{(since } t_k(i) \leq T\text{)} \\ &= nT\sum_{t=k}^{k+T}\|\bar{x}^{t+1} - \bar{x}^t\|^2 \\ &\overset{(4.62)}{\leq} \frac{2T(1+\eta^2L^2)}{n}\sum_{t=k}^{k+T}\|x_{i_t}^{t+1} - x_{i_t}^t\|^2.\end{aligned} \tag{4.64}$$

We can also bound $\sum_{i=1}^n \|x_i^{k+t_k(i)} - x_i^k\|^2$ as follows:

$$\begin{aligned}\sum_{i=1}^n \|x_i^{k+t_k(i)} - x_i^k\|^2 &= \sum_{i=1}^n \|\sum_{t=k}^{k+t_k(i)-1}(x_i^{t+1} - x_i^t)\|^2 \\ &\leq \sum_{i=1}^n t_k(i)\sum_{t=k}^{k+t_k(i)-1}\|x_i^{t+1} - x_i^t\|^2 \qquad \text{(Young's inequality)} \\ &\leq T\sum_{t=k}^{k+T-1}\sum_{i=1}^n \|x_i^{t+1} - x_i^t\|^2 \qquad \text{(since } t_k(i) \leq T\text{)} \\ &= T\sum_{t=k}^{k+T}\|x_{i_t}^{t+1} - x_{i_t}^t\|^2 \quad \text{(since only worker } i_t \text{ is updated).}\end{aligned} \tag{4.65}$$

Let us bound the first term on the right-hand side of (4.63) as follows:

$$\begin{aligned}\sum_{t=k}^{k+T}\mathbb{E}\left[\|\bar{x}^t - x_{\hat{i}_t}^t\|^2 \mid \mathcal{F}_{t-1}\right] \leq 2\sum_{t=k}^{k+T}\mathbb{E}\left[\|\bar{x}^{t-d_{\hat{i}_t}^t} - x_{\hat{i}_t}^t\|^2 \mid \mathcal{F}_{t-1}\right] \\ + 2\sum_{t=k}^{k+T}\mathbb{E}\left[\|\bar{x}^t - \bar{x}^{t-d_{\hat{i}_t}^t}\|^2 \mid \mathcal{F}_{t-1}\right].\end{aligned} \tag{4.66}$$

However, similar to the proof of (4.61) and (4.62), we can show that

$$\begin{aligned}\sum_{t=k}^{k+T}\|\bar{x}^t - \bar{x}^{t-d_{\hat{i}_t}^t}\|^2 &\overset{(4.61)}{\leq} \tau\sum_{t=k}^{k+T}\sum_{l=t-\tau}^{t-1}\|\bar{x}^{l+1} - \bar{x}^l\|^2 \\ &\leq \tau^2\sum_{t=k-\tau}^{k+T}\|\bar{x}^{t+1} - \bar{x}^t\|^2 \\ &\overset{(4.62)}{\leq} \sum_{t=k-\tau}^{k+T}\frac{2\tau^2(1+\eta^2L^2)}{n^2}\|x_{i_t}^{t+1} - x_{i_t}^t\|^2.\end{aligned} \tag{4.67}$$

On the other hand, by using (4.57), we have

$$\begin{aligned}
\|\bar{x}^{t-d^t_{\hat{i}_t}} - x^t_{\hat{i}_t}\|^2 &= \|y^{t+1}_{\hat{i}_t} - y^t_{\hat{i}_t}\|^2 \qquad \text{(by the update of } y^k_{i_k} \text{ in Algorithm 5)} \\
&\overset{(4.57)}{=} \left\|\tfrac{1}{\alpha}(x^{t+1}_{\hat{i}_t} - x^t_{\hat{i}_t}) + \tfrac{\eta}{\alpha}(\nabla f_{\hat{i}_t}(x^{t+1}_{\hat{i}_t}) - \nabla f_{\hat{i}_t}(x^t_{\hat{i}_t}))\right\|^2 \\
&\leq \tfrac{2}{\alpha^2}\|x^{t+1}_{\hat{i}_t} - x^t_{\hat{i}_t}\|^2 + \tfrac{2\eta^2}{\alpha^2}\|\nabla f_{\hat{i}_t}(x^{t+1}_{\hat{i}_t}) - \nabla f_{\hat{i}_t}(x^t_{\hat{i}_t})\|^2 \\
&\leq \tfrac{2(1+\eta^2L^2)}{\alpha^2}\|x^{t+1}_{\hat{i}_t} - x^t_{\hat{i}_t}\|^2.
\end{aligned} \tag{4.68}$$

Therefore, plugging (4.67) and (4.68) into (4.66), we have

$$\begin{aligned}
\textstyle\sum_{t=k}^{k+T} \mathbb{E}\left[\|\bar{x}^t - x^t_{\hat{i}_t}\|^2 \mid \mathcal{F}_{t-1}\right] &\leq \tfrac{4\tau^2(1+\eta^2L^2)}{n^2}\textstyle\sum_{t=k-\tau}^{k+T}\mathbb{E}\left[\|x^{t+1}_{\hat{i}_t} - x^t_{\hat{i}_t}\|^2 \mid \mathcal{F}_{t-1}\right] \\
&\quad + \tfrac{4(1+\eta^2L^2)}{\alpha^2}\textstyle\sum_{t=k-\tau}^{k+T}\mathbb{E}\left[\|x^{t+1}_{\hat{i}_t} - x^t_{\hat{i}_t}\|^2 \mid \mathcal{F}_{t-1}\right] \\
&= \tfrac{4(1+\eta^2L^2)[\tau^2\alpha^2+n^2]}{n^2\alpha^2}\textstyle\sum_{t=k-\tau}^{k+T}\mathbb{E}\left[\|x^{t+1}_{\hat{i}_t} - x^t_{\hat{i}_t}\|^2 \mid \mathcal{F}_{t-1}\right].
\end{aligned} \tag{4.69}$$

Substituting (4.64), (4.65), and (4.69) into (4.63), we obtain

$$\begin{aligned}
\tfrac{\hat{\mathbf{p}}}{2}\textstyle\sum_{i=1}^n \|\bar{x}^k - x^k_i\|^2 &\leq \tfrac{4(1+\eta^2L^2)[\tau^2\alpha^2+n^2]}{n^2\alpha^2}\textstyle\sum_{t=k-\tau}^{k+T}\mathbb{E}\left[\|x^{t+1}_{\hat{i}_t} - x^t_{\hat{i}_t}\|^2 \mid \mathcal{F}_{t-1}\right] \\
&\quad + \tfrac{8\hat{\mathbf{p}}T(1+\eta^2L^2)}{n}\textstyle\sum_{t=k}^{k+T}\|x^{t+1}_{i_t} - x^t_{i_t}\|^2 + 4\hat{\mathbf{p}}T\textstyle\sum_{t=k}^{k+T}\|x^{t+1}_{i_t} - x^t_{i_t}\|^2 \\
&\leq \tfrac{4(1+\eta^2L^2)[\tau^2\alpha^2+n^2]}{n^2\alpha^2}\textstyle\sum_{t=k-\tau}^{k+T}\mathbb{E}\left[\|x^{t+1}_{\hat{i}_t} - x^t_{\hat{i}_t}\|^2 \mid \mathcal{F}_{t-1}\right] \\
&\quad + \tfrac{4\hat{\mathbf{p}}T[2(1+\eta^2L^2)+n]}{n}\textstyle\sum_{t=k-\tau}^{k+T}\|x^{t+1}_{i_t} - x^t_{i_t}\|^2.
\end{aligned}$$

Finally, taking full expectation both sides of the last inequality w.r.t. $\sigma(d^k, \mathcal{F}_{k-1})$, and multiplying the result by $\frac{2}{\hat{\mathbf{p}}}$, we arrive at

$$\sum_{i=1}^n \mathbb{E}\left[\|\bar{x}^k - x^k_i\|^2\right] \leq D\sum_{t=k-\tau}^{k+T}\mathbb{E}\left[\|x^{t+1}_{i_t} - x^t_{i_t}\|^2\right],$$

where $D := \frac{8(1+\eta^2L^2)(\tau^2\alpha^2+n^2)}{\hat{\mathbf{p}}n^2\alpha^2} + \frac{8T[2(1+\eta^2L^2)+n]}{n}$. This inequality is exactly (4.31). □

4.5.3.4 Proof of Theorem 4.3

By Assumption 4.4, for each T iterations, the probability of each worker i getting updated is at least $\hat{\mathbf{p}} > 0$. Hence, from (4.30) of Lemma 4.4, we sum up from $t := k - \tau$ to $t := k + T$,

and have

$$\frac{\rho}{2}\sum_{t=k-\tau}^{k+T}\|x_{i_t}^{t+1}-x_{i_t}^{t}\|^2 \le \sum_{t=k-\tau}^{k+T}\big[\widetilde{V}_\eta^t(\bar{x}^t)-\widetilde{V}_\eta^{t+1}(\bar{x}^{t+1})\big],$$

where $\rho>0$ is given in Lemma 4.4. Now, take full expectation both sides of this inequality w.r.t. $\mathcal{F}_k$, we obtain

$$\frac{\rho}{2}\sum_{t=k-\tau}^{k+T}\mathbb{E}\big[\|x_{i_t}^{t+1}-x_{i_t}^{t}\|^2\big] \le \sum_{t=k-\tau}^{k+T}\Big[\mathbb{E}\big[\widetilde{V}_\eta^t(\bar{x}^t)\big]-\mathbb{E}\big[\widetilde{V}_\eta^{t+1}(\bar{x}^{t+1})\big]\Big]. \tag{4.70}$$

Next, using (4.35) from Lemma 4.8 with $\gamma_2=0$, we have

$$\|\mathcal{G}_\eta(\bar{x}^k)\|^2 \le \frac{(1+\eta L)^2}{n\eta^2}\sum_{i=1}^{n}\|x_i^k-\bar{x}^k\|^2.$$

Taking full expectation both sides of this inequality, and then combining the result and (4.31), we obtain

$$\mathbb{E}\left[\|\mathcal{G}_\eta(\bar{x}^k)\|^2\right] \le \tfrac{(1+\eta L)^2 D}{n\eta^2}\textstyle\sum_{t=k-\tau}^{k+T}\mathbb{E}\left[\|x_{i_t}^{t+1}-x_{i_t}^{t}\|^2\right],$$

where D is given in Lemma 4.5.

Combining the last inequality and (4.70), we arrive at

$$\mathbb{E}\left[\|\mathcal{G}_\eta(\bar{x}^k)\|^2\right] \le \tfrac{2(1+\eta L)^2 D}{n\eta^2\rho}\textstyle\sum_{t=k-\tau}^{k+T}\Big(\mathbb{E}\big[\widetilde{V}_\eta^t(\bar{x}^t)\big]-\mathbb{E}\big[\widetilde{V}_\eta^{t+1}(\bar{x}^{t+1})\big]\Big).$$

Averaging this inequality from $k:=0$ to $k:=K$, we get

$$\begin{aligned}\tfrac{1}{K+1}\textstyle\sum_{k=0}^{K}\mathbb{E}\left[\|\mathcal{G}_\eta(\bar{x}^k)\|^2\right] &\le \tfrac{\hat{C}}{K+1}\textstyle\sum_{k=0}^{K}\sum_{t=k-\tau}^{k+T}\Big[\mathbb{E}\big[\widetilde{V}_\eta^t(\bar{x}^t)\big]-\mathbb{E}\big[\widetilde{V}_\eta^{t+1}(\bar{x}^{t+1})\big]\Big]\\ &\le \tfrac{\hat{C}}{K+1}\big[\widetilde{V}_\eta^0(\bar{x}^0)-\mathbb{E}\big[\widetilde{V}_\eta^{K+T+1}(\bar{x}^{K+T+1})\big]\big],\end{aligned} \tag{4.71}$$

where $\hat{C}:=\frac{2(1+\eta L)^2 D}{n\rho\eta^2}$. Here, we have used the monotonicity of $\{\mathbb{E}\big[\widetilde{V}_\eta^k(\bar{x}^k)\big]\}_{k\ge 0}$ and $\mathbb{E}\big[\widetilde{V}_\eta^0(\bar{x}^0)\big]=\widetilde{V}_\eta^0(\bar{x}^0)$ in the last equality.

Now, recall from the definition of $\widetilde{V}_\eta^k(\cdot)$ and $V_\eta^k(\cdot)$ that

$$\widetilde{V}_\eta^0(\bar{x}^0)=V_\eta^0(\bar{x}^0)=F(x^0) \qquad \text{and} \qquad \mathbb{E}\left[\widetilde{V}_\eta^k(\bar{x}^k)\right]\ge\mathbb{E}\left[V_\eta^k(\bar{x}^k)\right]\overset{(4.46)}{\ge}F^\star.$$

Substituting these relations into (4.71), we eventually get

$$\frac{1}{K+1}\sum_{k=0}^{K}\mathbb{E}\left[\|\mathcal{G}_\eta(\bar{x}^k)\|^2\right] \leq \frac{\hat{C}}{(K+1)}\big[F(x^0)-F^\star\big],$$

which is exactly (4.32). Using the definition of ρ, θ, and D into $\hat{C}$, we obtain its simplified formula as in Theorem 4.3. The final conclusion of the theorem is a direct consequence of (4.32).

□

CHAPTER 5
Conclusions and Future Research

5.1 Conclusions

In this dissertation, we have proposed new stochastic and randomized algorithms to solve three common classes of nonconvex optimization problems in machine learning. These algorithms are shown to achieve the best-known convergence rate while exhibiting advantageous performance compared to existing methods using common numerical examples.

Firstly, we propose ProxSARAH, a stochastic proximal gradient framework using the SARAH estimator, to solve the composite expectation or finite sum problems which cover many problems in supervised learning. Our algorithm is different from existing stochastic proximal gradient methods in which we have an additional averaging step right after the proximal gradient step. Our algorithm also allows using single sample or mini-batch when estimating the SARAH estimator. Our convergence analysis not only works with constant step-size but also increasing step-size which is different from diminishing stepsize schedule in ProxSGD. Our algorithms are shown to not only achieve the best-known convergence rate but also match existing complexity lower bound for both expectation and finite-sum cases. We have also demonstrated that our methods are comparable or even outperform existing methods in various numerical experiments using real datasets.

Secondly, we study the policy optimization problem in reinforcement learning and consider a more general model with convex regularizer. In particular, we propose a new proximal hybrid stochastic policy gradient algorithm (ProxHSPGA) that uses a novel policy gradient estimator by combining an unbiased policy gradient estimator with a biased one. Theoretical results show that our algorithm achieves the best-known trajectory complexity to attain an approximate first-order solution under standard assumptions. In addition, our numerical experiments not

only help confirm the benefit of our algorithm compared to other closely related policy gradient methods but also verify the effectiveness of regularization in policy gradient methods.

Finally, we explore the composite finite sum problem in federated learning, a distributed training framework that has received tremendous attention in the past few years. We combine the classical Douglas-Rachford splitting technique with randomized strategy and asynchronous implementation to develop two new algorithms, called FedDR and asyncFedDR, which achieve the best-known communication complexity under standard assumptions. Different from existing methods including FedSplit and FedDR, our methods allow partial participation which selects a subset of workers to perform local update at each communication round. The asynchronous variant, asyncFedDR, can handle heterogeneity in workers' computing power which further improves the practicality of our methods. Numerical experiments on both synthetic and real datasets illustrate that our algorithms are not only communication-efficient by also able to handle heterogeneous data setting.

5.2 Ongoing and Future Research

Ongoing research. Following the success of FedDR for composite nonconvex problems, we further investigate whether we can improve the convergence rate of FedDR using acceleration techniques. Acceleration techniques have been used to improve the convergence rate of many existing methods resulting new accelerated algorithms in different areas. Acceleration has been applied to full gradient methods to obtain its accelerated variants such as Nesterov (1983); Beck and Teboulle (2009); Tseng (2008). In addition, acceleration is also extended to other settings such as stochastic gradient methods (Deng et al., 2018; Nitanda, 2014; Allen-Zhu, 2017b), coordinate gradient method (Lin et al., 2014; Shalev-Shwartz and Zhang, 2014), saddle-point problems (Chen et al., 2014; Mokhtari et al., 2020), monotone inclusion problems (Boţ and Csetnek, 2016; Rosasco et al., 2015), variational inequalities (Chen et al., 2017). Recently, there has been an active line of research of using the Halpern iteration (Halpern, 1967) to achieve acceleration. In federated learning, there are not many methods using acceleration techniques. Recent works include Yuan and Ma (2020); Ozfatura et al. (2021); Hanzely et al. (2020) where they mostly apply acceleration on the well-known FedAvg or LocalSGD algorithm

(McMahan et al., 2017; Zhou and Cong, 2017). Motivated by this fact, our goal is to to develop a new accelerated variant of FedDR, called **Acc-FedDR**, to solve convex problems in federated learning. The new method is expected to achieve faster convergence rate compared to its non-accelerated variant.

Moreover, similar to FedDR, we also aim at developing an asynchronous variant of **Acc-FedDR** for federated learning to deal with system heterogeneity, when there is large variance in workers computational power. Apart from **AsyncFedDR**, there are only two other works (Stich, 2018; Xie et al., 2019) that consider asynchronous algorithms for federated learning with convergence guarantee. The success of the asynchronous variant of **Acc-FedDR** will make significant contribution to this line of research.

In addition, our convergence analysis should allow inexact proximal evaluation when performing local update as **FedDR**. In practice, it is preferable to run the local update for a fixed number of iterations, establishing convergence with inexact evaluation of proximal operators as in Theorem 4.1 and 4.2 will further improve the practicality of **Acc-FedDR**.

Future research. As a continuation of my Ph.D. research projects in stochastic optimization methods and federated learning, the following ideas are promising to consider in the near future.

Similar to Mishchenko et al. (2019), we analyze the case where we can apply compression to our new algorithm to further reduce the communication cost. There have been many recent works discussing different type of compression (see Beznosikov et al., 2020; Albasyoni et al., 2020). We plan to study the results in Khaled and Richtárik (2019); Chraibi et al. (2019); Albasyoni et al. (2020) for our convergence analysis. We expect to achieve the same rate of convergence of our algorithm with compression as in the non-compressed case under mild additional assumptions.

Among these directions, we are also interested in perturbation techinques (Mania et al., 2015, 2017; Lu et al., 2019; Li, 2019; Chen et al., 2021) in order to obtain second-order optimality by only using first-order oracles. Gradient noise plays an importance role for stochastic gradient methods to escape saddle-point to reach second-order optimality (Vlaski and Sayed, 2019). While our proposed methods in earlier chapters only guarantee first-order stationarity, incorporating

perturbed stochastic gradient into our algorithms might further improve the convergence guarantee to a local optimum.

BIBLIOGRAPHY

Abadi, M., Agarwal, A., Barham, P., Brevdo, E., Chen, Z., Citro, C., Corrado, G. S., Davis, A., Dean, J., Devin, M., Ghemawat, S., Goodfellow, I., Harp, A., Irving, G., Isard, M., Jia, Y., Jozefowicz, R., Kaiser, L., Kudlur, M., Levenberg, J., Mané, D., Monga, R., Moore, S., Murray, D., Olah, C., Schuster, M., Shlens, J., Steiner, B., Sutskever, I., Talwar, K., Tucker, P., Vanhoucke, V., Vasudevan, V., Viégas, F., Vinyals, O., Warden, P., Wattenberg, M., Wicke, M., Yu, Y., and Zheng, X. (2016). Tensorflow: Large-scale machine learning on heterogeneous distributed systems. In *12th USENIX symposium on operating systems design and implementation (OSDI 16)*, pages 265–283.

Agarwal, A., Bartlett, P. L., Ravikumar, P., and Wainwright, M. J. (2012). Information-theoretic lower bounds on the oracle complexity of stochastic convex optimization. *IEEE Transactions on Information Theory*, 58(5):3235–3249.

Albasyoni, A., Safaryan, M., Condat, L., and Richtárik, P. (2020). Optimal gradient compression for distributed and federated learning. *arXiv preprint arXiv:2010.03246*.

Allen-Zhu, Z. (2017a). Katyusha: The first direct acceleration of stochastic gradient methods. In *Proceedings of the 49th Annual Symposium on Theory of Computing*, pages 1200–1205, New York, NY, USA.

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

Allen-Zhu, Z. (2018). Natasha 2: Faster non-convex optimization than SGD. In *Advances in Neural Information Processing Systems*, pages 2675–2686.

Allen-Zhu, Z. and Li, Y. (2018). Neon2: Finding local minima via first-order oracles. In Bengio, S., Wallach, H., Larochelle, H., Grauman, K., Cesa-Bianchi, N., and Garnett, R., editors, *Advances in Neural Information Processing Systems 31*, pages 3716–3726. Curran Associates, Inc.

Allen-Zhu, Z. and Yuan, Y. (2016). Improved svrg for non-strongly-convex or sum-of-non-convex objectives. In *Proceedings of the 33rd International Conference on International Conference on Machine Learning - Volume 48*, pages 1080–1089.

Amiri, M. M. and Gündüz, D. (2020). Federated learning over wireless fading channels. *IEEE Transactions on Wireless Communications*, 19(5):3546–3557.

Arjevani, Y., Carmon, Y., Duchi, J. C., Foster, D. J., Srebro, N., and Woodworth, B. (2019). Lower bounds for non-convex stochastic optimization. *arXiv preprint arXiv:1912.02365*.

Bauschke, H. H. and Combettes, P. (2011). *Convex analysis and monotone operator theory in Hilbert spaces*, volume 408. Springer.

Bauschke, H. H. and Combettes, P. (2017). *Convex analysis and monotone operators theory in Hilbert spaces*. Springer-Verlag, 2nd edition.

Baxter, J. and Bartlett, P. L. (2001). Infinite-horizon policy-gradient estimation. *J. Artif. Int. Res.*, 15(1):319–350.

Beck, A. and Teboulle, M. (2009). A fast iterative shrinkage-thresholding algorithm for linear inverse problems. *SIAM journal on imaging sciences*, 2(1):183–202.

Bertsekas, D. and Tsitsiklis, J. N. (1989). *Parallel and distributed computation: Numerical methods*. Prentice Hall.

Beznosikov, A., Horváth, S., Richtárik, P., and Safaryan, M. (2020). On biased compression for distributed learning. *arXiv preprint arXiv:2002.12410*.

Bonawitz, K., Eichner, H., Grieskamp, W., Huba, D., Ingerman, A., Ivanov, V., Kiddon, C., Konečný, J., Mazzocchi, S., McMahan, H. B., Overveldt, T. V., Petrou, D., Ramage, D., and Roselander, J. (2019). Towards federated learning at scale: System design.

Boţ, R. I. and Csetnek, E. R. (2016). An inertial forward-backward-forward primal-dual splitting algorithm for solving monotone inclusion problems. *Numerical Algorithms*, 71(3):519–540.

Bridle, J. S. (1990). Training stochastic model recognition algorithms as networks can lead to maximum mutual information estimation of parameters. In *Advances in Neural Information Processing Systems*, pages 211–217. Morgan-Kaufmann.

Brockman, G., Cheung, V., Pettersson, L., Schneider, J., Schulman, J., Tang, J., and Zaremba, W. (2016). OpenAI Gym. *arXiv preprint arXiv:1606.01540*.

Cai, J. (2018). Implementing DistBelief. `https://jcaip.github.io/Distbelief/`.

Caldas, S., Duddu, S. M. K., Wu, P., Li, T., Konečnỳ, J., McMahan, H. B., Smith, V., and Talwalkar, A. (2018). LEAF: A benchmark for federated settings. *arXiv preprint arXiv:1812.01097*.

Cannelli, L., Facchinei, F., Kungurtsev, V., and Scutari, G. (2019). Asynchronous parallel algorithms for nonconvex optimization. *Mathematical Programming*, pages 1–34.

Chambolle, A., Ehrhardt, M. J., Richtárik, P., and Schönlieb, C.-B. (2018). Stochastic primal-dual hybrid gradient algorithm with arbitrary sampling and imaging applications. *SIAM Journal on Optimization*, 28(4):2783–2808.

Chang, C. C. and Lin, C. J. (2011). LIBSVM: A library for support vector machines. *ACM transactions on intelligent systems and technology (TIST)*, 2(3):1–27.

Charles, Z. and Konečnỳ, J. (2021). Convergence and accuracy trade-offs in federated learning and meta-learning. In *International Conference on Artificial Intelligence and Statistics*, pages 2575–2583. PMLR.

Chen, M., Yang, Z., Saad, W., Yin, C., Poor, H. V., and Cui, S. (2020). A joint learning and communications framework for federated learning over wireless networks.

Chen, Y., Lan, G., and Ouyang, Y. (2014). Optimal primal-dual methods for a class of saddle point problems. *SIAM Journal on Optimization*, 24(4):1779–1814.

Chen, Y., Lan, G., and Ouyang, Y. (2017). Accelerated schemes for a class of variational inequalities. *Mathematical Programming*, 165(1):113–149.

Chen, Z., Zhou, D., and Gu, Q. (2021). Faster perturbed stochastic gradient methods for finding local minima. *arXiv preprint arXiv:2110.13144*.

Choromanska, A., Henaff, M., Mathieu, M., Arous, G. B., and LeCun, Y. (2015). The loss surfaces of multilayer networks. In *Artificial intelligence and statistics*, pages 192–204.

Chraibi, S., Khaled, A., Kovalev, D., Richtárik, P., Salim, A., and Takáč, M. (2019). Distributed fixed point methods with compressed iterates. *arXiv preprint arXiv:1912.09925*.

Combettes, P. L. and Eckstein, J. (2018). Asynchronous block-iterative primal-dual decomposition methods for monotone inclusions. *Mathematical Programming*, 168(1):645–672.

Combettes, P. L. and Pesquet, J.-C. (2015). Stochastic quasi-Fejér block-coordinate fixed point iterations with random sweeping. *SIAM Journal on Optimization*, 25(2):1221–1248.

Cortes, C., Mansour, Y., and Mohri, M. (2010). Learning bounds for importance weighting. In *Advances in Neural Information Processing Systems*, pages 442–450.

Cutkosky, A. and Orabona, F. (2019). Momentum-based variance reduction in non-convex sgd. In *Advances in Neural Information Processing Systems*, pages 15210–15219.

Dao, M. N. and Tam, M. K. (2019). A Lyapunov-type approach to convergence of the Douglas-Rachford algorithm for a nonconvex setting. *Journal of Global Optimization*, 73(1):83–112.

DeepMind (2019). AlphaStar: Mastering the Real-Time Strategy Game StarCraft II. https://deepmind.com/blog.

Defazio, A., Bach, F., and Lacoste-Julien, S. (2014). SAGA: A fast incremental gradient method with support for non-strongly convex composite objectives. In *Advances in neural information processing systems*, pages 1646–1654.

Deng, Q., Cheng, Y., and Lan, G. (2018). Optimal adaptive and accelerated stochastic gradient descent. *arXiv preprint arXiv:1810.00553*.

Duan, Y., Chen, X., Houthooft, R., Schulman, J., and Abbeel, P. (2016). Benchmarking deep reinforcement learning for continuous control. In *Proceedings of the 33rd International Conference on International Conference on Machine Learning - Volume 48*, pages 1329–1338.

Fang, C., Li, C. J., Lin, Z., and Zhang, T. (2018). Spider: Near-optimal non-convex optimization via stochastic path integrated differential estimator. *arXiv preprint arXiv:1807.01695*.

Frostig, R., Ge, R., Kakade, S. M., , and Sidford, A. (2015). Competing with the empirical risk minimizer in a single pass. In *Conference on learning theory*, pages 728–763.

Ghadimi, S. and Lan, G. (2012). Optimal stochastic approximation algorithms for strongly convex stochastic composite optimization I: A generic algorithmic framework. *SIAM Journal on Optimization*, 22(4):1469–1492.

Ghadimi, S. and Lan, G. (2013). Stochastic first-and zeroth-order methods for nonconvex stochastic programming. *SIAM Journal on Optimization*, 23(4):2341–2368.

Ghadimi, S., Lan, G., and Zhang, H. (2016). Mini-batch stochastic approximation methods for nonconvex stochastic composite optimization. *Mathematical Programming*, 155(1-2):267–305.

Gorbunov, E., Hanzely, F., and Richtárik, P. (2021). Local SGD: Unified theory and new efficient methods. In *International Conference on Artificial Intelligence and Statistics*, pages 3556–3564. PMLR.

Haarnoja, T., Zhou, A., Abbeel, P., and Levine, S. (2018). Soft actor-critic: Off-policy maximum entropy deep reinforcement learning with a stochastic actor. In *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pages 1861–1870, Stockholmsmässan, Stockholm Sweden. PMLR.

Haddadpour, F., Kamani, M. M., Mahdavi, M., and Cadambe, V. (2019). Local SGD with periodic averaging: Tighter analysis and adaptive synchronization. In *Advances in Neural Information Processing Systems*, pages 11082–11094.

Haddadpour, F., Kamani, M. M., Mokhtari, A., and Mahdavi, M. (2021). Federated learning with compression: Unified analysis and sharp guarantees. In *International Conference on Artificial Intelligence and Statistics*, pages 2350–2358. PMLR.

Haddadpour, F. and Mahdavi, M. (2019). On the convergence of local descent methods in federated learning. *arXiv preprint arXiv:1910.14425*.

Halpern, B. (1967). Fixed points of nonexpanding maps. *Bulletin of the American Mathematical Society*, 73(6):957–961.

Hanzely, F., Hanzely, S., Horváth, S., and Richtárik, P. (2020). Lower bounds and optimal algorithms for personalized federated learning. *arXiv preprint arXiv:2010.02372*.

Harikandeh, R., Ahmed, M. O., Virani, A., M. Schmidt, J. K., and Sallinen, S. (2015). Stopwasting my gradients: Practical SVRG. In *Advances in Neural Information Processing Systems*, pages 2251–2259.

Hasselt, H. v., Guez, A., and Silver, D. (2016). Deep reinforcement learning with Double Q-learning. In *Proceedings of the 30th Conference on Artificial Intelligence*, pages 2094–2100.

Johnson, R. and Zhang, T. (2013). Accelerating stochastic gradient descent using predictive variance reduction. In *Advances in Neural Information Processing Systems*, pages 315–323.

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

Karimireddy, S. P., Jaggi, M., Kale, S., Mohri, M., Reddi, S. J., Stich, S. U., and Suresh, A. T. (2020a). Mime: Mimicking centralized stochastic algorithms in federated learning. *arXiv preprint arXiv:2008.03606*.

Karimireddy, S. P., Kale, S., Mohri, M., Reddi, S., Stich, S., and Suresh, A. T. (2020b). Scaffold: Stochastic controlled averaging for federated learning. In *International Conference on Machine Learning*, pages 5132–5143. PMLR.

Khaled, A., Mishchenko, K., and Richtárik, P. (2019). First analysis of local GD on heterogeneous data. *arXiv preprint arXiv:1909.04715*.

Khaled, A. and Richtárik, P. (2019). Gradient descent with compressed iterates. *arXiv preprint arXiv:1909.04716*.

Kingma, D. P. and Ba, J. (2014). Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Klimov, O. and Schulman, J. (2017). Roboschool. https://openai.com/blog/roboschool/.

Konečnỳ, J., McMahan, H. B., Ramage, D., and Richtárik, P. (2016). Federated optimization: Distributed machine learning for on-device intelligence. *arXiv preprint arXiv:1610.02527*.

LeCun, Y., Bottou, L., Bengio, Y., and Haffner, P. (1998). Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324.

Lei, L. and Jordan, M. (2017). Less than a single pass: Stochastically controlled stochastic gradient. In *International Conference on Artificial Intelligence and Statistics*, pages 148–156.

Levine, S. (2017). CS 294-112: Deep reinforcement learning lecture notes.

Li, G. and Pong, T. K. (2015). Global convergence of splitting methods for nonconvex composite optimization. *SIAM Journal on Optimization*, 25(4):2434–2460.

Li, G. and Pong, T. K. (2016). Douglas-Rachford splitting for nonconvex optimization with application to nonconvex feasibility problems. *Mathematical programming*, 159(1-2):371–401.

Li, Q., Wen, Z., Wu, Z., Hu, S., Wang, N., Li, Y., Liu, X., and He, B. (2019a). A survey on federated learning systems: vision, hype and reality for data privacy and protection. *arXiv preprint arXiv:1907.09693*.

Li, T., Sahu, A. K., Talwalkar, A., and Smith, V. (2020a). Federated learning: Challenges, methods, and future directions. *IEEE Signal Processing Magazine*, 37(3):50–60.

Li, T., Sahu, A. K., Zaheer, M., Sanjabi, M., Talwalkar, A., and Smith, V. (2020b). Federated optimization in heterogeneous networks. In Dhillon, I., Papailiopoulos, D., and Sze, V., editors, *Proceedings of Machine Learning and Systems*, volume 2, pages 429–450.

Li, X., Huang, K., Yang, W., Wang, S., and Zhang, Z. (2019b). On the convergence of FedAvg on non-iid data. *arXiv preprint arXiv:1907.02189*.

Li, X., Jiang, M., Zhang, X., Kamp, M., and Dou, Q. (2021). FedBN: Federated learning on non-iid features via local batch normalization. *arXiv preprint arXiv:2102.07623*.

Li, Z. (2019). SSRGD: Simple stochastic recursive gradient descent for escaping saddle points. *arXiv preprint arXiv:1904.09265*.

Li, Z. and Li, J. (2018). A simple proximal stochastic gradient method for nonsmooth nonconvex optimization. In *Proceedings of the 32Nd International Conference on Neural Information Processing Systems*, pages 5569–5579, USA.

Lihua, L., Ju, C., Chen, J., , and Jordan, M. (2017). Non-convex finite-sum optimization via scsg methods. In *Advances in Neural Information Processing Systems*, pages 2348–2358.

Lillicrap, T. P., Hunt, J. J., Pritzel, A., Heess, N., Erez, T., Tassa, Y., Silver, D., and Wierstra, D. (2016). Continuous control with deep reinforcement learning. In *4th International Conference on Learning Representations, San Juan, Puerto Rico, May 2-4, 2016, Conference Track Proceedings*.

Lin, H., Mairal, J., and Harchaoui, Z. (2015). A universal catalyst for first-order optimization. In *Advances in Neural Information Processing Systems*, pages 3384–3392.

Lin, Q., Lu, Z., and Xiao, L. (2014). An accelerated proximal coordinate gradient method. *Advances in Neural Information Processing Systems*, 27:3059–3067.

Lin, T., Stich, S. U., Patel, K. K., and Jaggi, M. (2018). Don't use large mini-batches, use local SGD. *arXiv preprint arXiv:1808.07217*.

Lions, P. L. and Mercier, B. (1979). Splitting algorithms for the sum of two nonlinear operators. *SIAM Journal on Numerical Analysis*, 16:964–979.

Liu, Y., Xu, Y., and Yin, W. (2021). Acceleration of primal–dual methods by preconditioning and simple subproblem procedures. *Journal of Scientific Computing*, 86(2):1–34.

Liu, Z., Li, X., Kang, B., and Darrell, T. (2019). Regularization matters in policy optimization. *arXiv preprint arXiv:1910.09191*.

Lohr, S. L. (2009). *Sampling: design and analysis*. Nelson Education.

Lu, S., Hong, M., and Wang, Z. (2019). PA-GD: On the convergence of perturbed alternating gradient descent to second-order stationary points for structured nonconvex optimization. In *International Conference on Machine Learning*, pages 4134–4143. PMLR.

Mania, H., Pan, X., Papailiopoulos, D., Recht, B., Ramchandran, K., and Jordan, M. I. (2015). Perturbed iterate analysis for asynchronous stochastic optimization. *arXiv preprint arXiv:1507.06970*.

Mania, H., Pan, X., Papailiopoulos, D., Recht, B., Ramchandran, K., and Jordan, M. I. (2017). Perturbed iterate analysis for asynchronous stochastic optimization. *SIAM Journal on Optimization*, 27(4):2202–2229.

McMahan, B., Moore, E., Ramage, D., Hampson, S., and y Arcas, B. A. (2017). Communication-efficient learning of deep networks from decentralized data. In *Artificial Intelligence and Statistics*, pages 1273–1282. PMLR.

McMahan, B. and Ramage, D. (2017). Federated learning: Collaborative machine learning without centralized training data.

Mishchenko, K., Gorbunov, E., Takáč, M., and Richtárik, P. (2019). Distributed learning with compressed gradient differences. *arXiv preprint arXiv:1901.09269*.

Mnih, V., Badia, A. P., Mirza, M., Graves, A., Lillicrap, T., Harley, T., Silver, D., and Kavukcuoglu, K. (2016). Asynchronous methods for deep reinforcement learning. In *Proceedings of The 33rd International Conference on Machine Learning*, volume 48, pages 1928–1937.

Mnih, V., Kavukcuoglu, K., Silver, D., Graves, A., Antonoglou, I., Wierstra, D., and Riedmiller, M. A. (2013). Playing Atari with deep reinforcement learning. *arXiv preprint arXiv:1312.5602*.

Mnih, V., Kavukcuoglu, K., Silver, D., Rusu, A. A., Veness, J., Bellemare, M. G., Graves, A., Riedmiller, M. A., Fidjeland, A., Ostrovski, G., Petersen, S., Beattie, C., Sadik, A., Antonoglou, I., King, H., Kumaran, D., Wierstra, D., Legg, S., and Hassabis, D. (2015). Human-level control through deep reinforcement learning. *Nature*, 518:529–533.

Mokhtari, A., Ozdaglar, A., and Pattathil, S. (2020). A unified analysis of extra-gradient and optimistic gradient methods for saddle point problems: Proximal point approach. In *International Conference on Artificial Intelligence and Statistics*, pages 1497–1507. PMLR.

Nemirovski, A., Juditsky, A., Lan, G., and Shapiro, A. (2009). Robust stochastic approximation approach to stochastic programming. *SIAM Journal on optimization*, 19(4):1574–1609.

Nemirovskii, A. and Yudin, D. (1983). Problem complexity and method efficiency in optimization.

Nesterov, Y. (2013). *Introductory lectures on convex optimization: A basic course*, volume 87. Springer Science & Business Media.

Nesterov, Y. and Polyak, B. (2006). Cubic regularization of Newton method and its global performance. *Mathematical Programming*, 108(1):177–205.

Nesterov, Y. E. (1983). A method for solving the convex programming problem with convergence rate o (1/kˆ 2). In *Dokl. akad. nauk Sssr*, volume 269, pages 543–547.

Neyshabur, B., Bhojanapalli, S., Mcallester, D., and Srebro, N. (2017). Exploring generalization in deep learning. In *Advances in Neural Information Processing Systems*, pages 5947–5956.

Nguyen, L., Nguyen, P. H., Dijk, M., Richtárik, P., Scheinberg, K., and Takác, M. (2018). SGD and Hogwild! convergence without the bounded gradients assumption. In *International Conference on Machine Learning*, pages 3750–3758. PMLR.

Nguyen, L. M., Liu, J., Scheinberg, K., and Takáč, M. (2017a). SARAH: A novel method for machine learning problems using stochastic recursive gradient. In *Proceedings of the 34th International Conference on Machine Learning*, pages 2613–2621.

Nguyen, L. M., Liu, J., Scheinberg, K., and Takáč, M. (2017b). Stochastic recursive gradient algorithm for nonconvex optimization. *arXiv preprint arXiv:1705.07261*.

Nguyen, L. M., Scheinberg, K., and Takáč, M. (2020). Inexact SARAH algorithm for stochastic optimization. *Optimization Methods and Software*, pages 1–22.

Nguyen, L. M., van Dijk, M., Phan, D. T., Nguyen, P. H., Weng, T.-W., and Kalagnanam, J. R. (2019). Finite-sum smooth optimization with SARAH. *arXiv preprint arXiv:1901.07648*.

Niknam, S., Dhillon, H. S., and Reed, J. (2020). Federated learning for wireless communications: Motivation, opportunities, and challenges. *IEEE Communications Magazine*, 58:46–51.

Nitanda, A. (2014). Stochastic proximal gradient descent with acceleration techniques. In *Advances in Neural Information Processing Systems*, pages 1574–1582.

OpenAI (2018). OpenAI Five. `https://blog.openai.com/openai-sfive/`.

Ozfatura, E., Ozfatura, K., and Gündüz, D. (2021). FedADC: Accelerated federated learning with drift control. In *2021 IEEE International Symposium on Information Theory (ISIT)*, pages 467–472. IEEE.

Papini, M., Binaghi, D., Canonaco, G., Pirotta, M., and Restelli, M. (2018). Stochastic variance-reduced policy gradient. In *Proceedings of the 35th International Conference on Machine Learning*, volume 80, pages 4026–4035.

Parikh, N. and Boyd, S. (2014). Proximal algorithms. *Found. Trends Optim.*, 1(3):127–239.

Pathak, R. and Wainwright, M. J. (2020). FedSplit: An algorithmic framework for fast federated optimization. *arXiv preprint arXiv:2005.05238*.

Peng, Z., Xu, Y., Yan, M., and Yin, W. (2016). ARock: an algorithmic framework for asynchronous parallel coordinate updates. *SIAM Journal on Scientific Computing*, 38(5):2851–2879.

Pham, N., Nguyen, L., Phan, D., Nguyen, P. H., van Dijk, M., and Tran-Dinh, Q. (2020a). A hybrid stochastic policy gradient algorithm for reinforcement learning. volume 108 of *Proceedings of Machine Learning Research*, pages 374–385. PMLR.

Pham, N. H., Nguyen, L. M., Phan, D. T., and Tran-Dinh, Q. (2020b). ProxSARAH: An efficient algorithmic framework for stochastic composite nonconvex optimization. *Journal of Machine Learning Research*, 21(110):1–48.

Recht, B., Re, C., Wright, S., and Niu, F. (2011). Hogwild!: A lock-free approach to parallelizing stochastic gradient descent. In Shawe-Taylor, J., Zemel, R. S., Bartlett, P. L., Pereira, F., and Weinberger, K. Q., editors, *Advances in Neural Information Processing Systems 24*, pages 693–701. Curran Associates, Inc.

Reddi, S. J., Hefny, A., Sra, S., Póczós, B., and Smola, A. (2016a). Stochastic variance reduction for nonconvex optimization. In *Proceedings of the 33rd International Conference on International Conference on Machine Learning - Volume 48*, pages 314–323.

Reddi, S. J., Sra, S., Póczos, B., and Smola, A. J. (2016b). Proximal stochastic methods for nonsmooth nonconvex finite-sum optimization. In *Advances in Neural Information Processing Systems*, pages 1145–1153.

Reddi, S. J., Sra, S., Póczos, B., and Smola, A. J. (2016c). Stochastic Frank-Wolfe methods for nonconvex optimization. In *2016 54th Annual Allerton Conference on Communication, Control, and Computing (Allerton)*, pages 1244–1251. IEEE.

Richtárik, P. and Takáč, M. (2016). Parallel coordinate descent methods for big data optimization. *Mathematical Programming*, 156(1-2):433–484.

Robbins, H. and Monro, S. (1951). A stochastic approximation method. *The annals of mathematical statistics*, pages 400–407.

Rosasco, L., Villa, S., and Vu, B. C. (2015). Stochastic inertial primal-dual algorithms. *arXiv preprint arXiv:1507.00852*.

Schmidt, M., Roux, N. L., and Bach, F. (2017). Minimizing finite sums with the stochastic average gradient. *Mathematical Programming*, 162(1-2):83–112.

Schulman, J., Levine, S., Abbeel, P., Jordan, M., and Moritz, P. (2015). Trust region policy optimization. In *Proceedings of the 32nd International Conference on Machine Learning*, volume 37, pages 1889–1897, Lille, France.

Schulman, J., Wolski, F., Dhariwal, P., Radford, A., and Klimov, O. (2017). Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*.

Shalev-Shwartz, S. and Zhang, T. (2013). Stochastic dual coordinate ascent methods for regularized loss minimization. *Journal of Machine Learning Research*, 14(Feb):567–599.

Shalev-Shwartz, S. and Zhang, T. (2014). Accelerated proximal stochastic dual coordinate ascent for regularized loss minimization. In *International conference on machine learning*, pages 64–72.

Shamir, O., Srebro, N., and Zhang, T. (2014). Communication-efficient distributed optimization using an approximate newton-type method. In *International conference on machine learning*, pages 1000–1008.

Shapiro, A., Dentcheva, D., and Ruszczynski, A. (2014). *Lectures on stochastic programming: modeling and theory*. SIAM.

Shen, Z., Ribeiro, A., Hassani, H., Qian, H., and Mi, C. (2019). Hessian aided policy gradient. In *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 5729–5738, Long Beach, California, USA.

Silver, D., Huang, A., Maddison, C. J., Guez, A., Sifre, L., v. d. Driessche, G., Schrittwieser, J., Antonoglou, I., Panneershelvam, V., Lanctot, M., Dieleman, S., Grewe, D., Nham, J., Kalchbrenner, N., Sutskever, I., Lillicrap, T., Leach, M., Kavukcuoglu, K., Graepel, T., and Hassabis, D. (2016). Mastering the game of Go with deep neural networks and tree search. *Nature*, 529:484–503.

Silver, D., Hubert, T., Schrittwieser, J., Antonoglou, I., Lai, M., Guez, A., Lanctot, M., Sifre, L., Kumaran, D., Graepel, T., Lillicrap, T., Simonyan, K., and Hassabis, D. (2018). A general reinforcement learning algorithm that masters chess, shogi, and go through self-play. *Science*, 362(6419):1140–1144.

Silver, D., Lever, G., Heess, N., Degris, T., Wierstra, D., and Riedmiller, M. (2014). Deterministic policy gradient algorithms. In *Proceedings of the 31st International Conference on International Conference on Machine Learning - Volume 32*, pages I–387–I–395.

Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., and Salakhutdinov, R. (2014). Dropout: a simple way to prevent neural networks from overfitting. *The journal of machine learning research*, 15(1):1929–1958.

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

Sutton, R. S. and Barto, A. G. (2018). *Introduction to Reinforcement Learning, 2nd Edition*. MIT Press.

Sutton, R. S., McAllester, D., Singh, S., and Mansour, Y. (1999). Policy gradient methods for reinforcement learning with function approximation. In *Proceedings of the 12th International Conference on Neural Information Processing Systems*, pages 1057–1063.

Themelis, A. and Patrinos, P. (2020). Douglas-Rachford splitting and ADMM for nonconvex optimization: Tight convergence results. *SIAM Journal on Optimization*, 30(1):149–181.

Tieleman, T. and Hinton, G. (2012). Lecture 6.5—RmsProp: Divide the gradient by a running average of its recent magnitude. COURSERA: Neural Networks for Machine Learning.

Todorov, E., Erez, T., and Tassa, Y. (2012). MuJoCo: A physics engine for model-based control. *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pages 5026–5033.

Tran-Dinh, Q., Pham, N. H., Phan, D. T., and Nguyen, L. M. (2019a). Hybrid stochastic gradient descent algorithms for stochastic nonconvex optimization. *arXiv preprint arXiv:1905.05920*.

Tran-Dinh, Q., Pham, N. H., Phan, D. T., and Nguyen, L. M. (2019b). A hybrid stochastic optimization framework for stochastic composite nonconvex optimization. *arXiv preprint arXiv:1907.03793*.

Tran-Dinh, Q., Pham, N. H., Phan, D. T., and Nguyen, L. M. (2021). FedDR–randomized Douglas-Rachford splitting algorithms for nonconvex federated composite optimization. *arXiv preprint arXiv:2103.03452*.

Tseng, P. (2008). On accelerated proximal gradient methods for convex-concave optimization. *submitted to SIAM Journal on Optimization*, 2(3).

Tziotis, I., Caramanis, C., and Mokhtari, A. (2020). Second-order optimality in non-convex decentralized optimization via perturbed gradient tracking. *Advances in Neural Information Processing Systems*.

Vlaski, S. and Sayed, A. H. (2019). Second-order guarantees of stochastic gradient descent in non-convex optimization. *arXiv preprint arXiv:1908.07023*.

Wang, J. and Joshi, G. (2018). Cooperative SGD: A unified framework for the design and analysis of communication-efficient SGD algorithms. *arXiv preprint arXiv:1808.07576*.

Wang, Z., Bapst, V., Heess, N., Mnih, V., Munos, R., Kavukcuoglu, K., and d. Freitas, N. (2017). Sample efficient actor-critic with experience replay. In *5th International Conference on Learning Representations, Toulon, France, April 24-26, 2017, Conference Track Proceedings*.

Wang, Z., Ji, K., Zhou, Y., Liang, Y., and Tarokh, V. (2019). Spiderboost and momentum: Faster variance reduction algorithms. In *Advances in Neural Information Processing Systems*, pages 2406–2416. Curran Associates, Inc.

Wang, Z., Schaul, T., Hessel, M., Van Hasselt, H., Lanctot, M., and De Freitas, N. (2016). Dueling network architectures for deep reinforcement learning. In *Proceedings of the 33rd International Conference on International Conference on Machine Learning - Volume 48*, pages 1995–2003.

Watkins, C. J. C. H. and Dayan, P. (1992). Q-learning. *Machine Learning*, 8(3):279–292.

Williams, R. J. (1992). Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine Learning*, 8(3):229–256.

Woodworth, B., Patel, K. K., and Srebro, N. (2020a). Minibatch vs local SGD for heterogeneous distributed learning. *arXiv preprint arXiv:2006.04735*.

Woodworth, B., Patel, K. K., Stich, S. U., Dai, Z., Bullins, B., McMahan, H. B., Shamir, O., and Srebro, N. (2020b). Is local SGD better than minibatch SGD? *arXiv preprint arXiv:2002.07839*.

Wu, Y., Mansimov, E., Liao, S., Grosse, R., and Ba, J. (2017). Scalable trust-region method for deep reinforcement learning using kronecker-factored approximation. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, pages 5285–5294, USA.

Xiao, L. and Zhang, T. (2014). A proximal stochastic gradient method with progressive variance reduction. *SIAM Journal on Optimization*, 24(4):2057–2075.

Xie, C., Koyejo, S., and Gupta, I. (2019). Asynchronous federated optimization. *arXiv preprint arXiv:1903.03934*.

Xu, P., Gao, F., and Gu, Q. (2019a). An improved convergence analysis of stochastic variance-reduced policy gradient. *Conference on Uncertainty in Artificial Intelligence*.

Xu, P., Gao, F., and Gu, Q. (2019b). Sample efficient policy gradient methods with recursive variance reduction. *arXiv preprint arXiv:1909.08610*.

Yang, L. and Zhang, Y. (2019). Policy optimization with stochastic mirror descent. *arXiv preprint arXiv:1906.10462*.

Yu, H., Yang, S., and Zhu, S. (2019). Parallel restarted SGD with faster convergence and less communication: Demystifying why model averaging works for deep learning. In *Proceedings of the Conference on Artificial Intelligence*, volume 33, pages 5693–5700.

Yu, P., Kundu, A., Wynter, L., and Lim, S. H. (2020). Fed+: A unified approach to robust personalized federated learning. *arXiv preprint arXiv:2009.06303*.

Yuan, H., Li, C. J., Tang, Y., and Zhou, Y. (2019). Policy optimization via stochastic recursive gradient algorithm.

Yuan, H. and Ma, T. (2020). Federated accelerated stochastic gradient descent. *arXiv preprint arXiv:2006.08950*.

Yuan, H., Zaheer, M., and Reddi, S. (2021). Federated composite optimization. In *International Conference on Machine Learning*, pages 12253–12266. PMLR.

Zhang, C., Bengio, S., Hardt, M., Recht, B., and Vinyals, O. (2017). Understanding deep learning requires rethinking generalization.

Zhang, C., Patras, P., and Haddadi, H. (2019). Deep learning in mobile and wireless networking: A survey. *IEEE Communications Surveys & Tutorials*, 21(3):2224–2287.

Zhang, J., De Sa, C., Mitliagkas, I., and Ré, C. (2016). Parallel SGD: When does averaging help? *arXiv preprint arXiv:1606.07365*.

Zhang, X., Hong, M., Dhople, S., Yin, W., and Liu, Y. (2020). FedPD: A federated learning framework with optimal rates and adaptivity to non-iid data. *arXiv preprint arXiv:2005.11418*.

Zhao, L., Mammadov, M., and Yearwood, J. (2010). From convex to nonconvex: A loss function analysis for binary classification. In *2010 IEEE International Conference on Data Mining Workshops*, pages 1281–1288.

Zhao, T., Hachiya, H., Niu, G., and Sugiyama, M. (2011). Analysis and improvement of policy gradient estimation. In *Advances in Neural Information Processing Systems*, pages 262–270.

Zhao, Y., Li, M., Lai, L., Suda, N., Civin, D., and Chandra, V. (2018). Federated learning with non-iid data. *arXiv preprint arXiv:1806.00582*.

Zhou, D. and Gu, Q. (2019). Lower bounds for smooth nonconvex finite-sum optimization. volume 97 of *Proceedings of Machine Learning Research*, pages 7574–7583, Long Beach, California, USA. PMLR.

Zhou, D., Xu, P., and Gu, Q. (2018a). Stochastic nested variance reduction for nonconvex optimization. In *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, page 3925–3936, Red Hook, NY, USA. Curran Associates Inc.

Zhou, D., Xu, P., and Gu, Q. (2018b). Stochastic nested variance reduction for nonconvex optimization. *arXiv preprint arXiv:1806.07811*.

Zhou, F. and Cong, G. (2017). On the convergence properties of a k-step averaging stochastic gradient descent algorithm for nonconvex optimization. *arXiv preprint arXiv:1708.01012*.

Zhou, Y., Wang, Z., Ji, K., Liang, Y., and Tarokh, V. (2019). Momentum schemes with stochastic variance reduction for nonconvex composite optimization. *arXiv preprint arXiv:1902.02715*.

Zhu, L., Liu, Z., and Han, S. (2019). Deep leakage from gradients. In *Advances in Neural Information Processing Systems*, pages 14774–14784.