A Framework for Analyzing Stochastic Optimization Algorithms Under Dependence

Chaoxu Zhou

Submitted in partial fulfillment of the
requirements for the degree of
Doctor of Philosophy
under the Executive Committee
of the Graduate School of Arts and Sciences

COLUMBIA UNIVERSITY

2020

ABSTRACT

A Framework for Analyzing Stochastic Algorithms Under Dependence

Chaoxu Zhou

In this dissertation, a theoretical framework based on concentration inequalities for empirical processes is developed to better design iterative optimization algorithms and analyze their convergence properties in the presence of complex dependence between directions and step-sizes. Based on this framework, we proposed a stochastic away-step Frank-Wolfe algorithm and a stochastic pairwise-step Frank-Wolfe algorithm for solving strongly convex problems with polytope constraints and proved that both of those algorithms converge linearly to the optimal solution in expectation and almost surely. Numerical results showed that the proposed algorithms are faster and more stable than most of their competitors.

This framework can be applied for designing and analyzing stochastic algorithms with adaptive step-sizes that are based on local curvature for self-concordant optimization problems. Notably, we proposed and analyzed a stochastic BFGS algorithm without line-search, and proved that it converges linearly globally and super-linearly locally using the framework mentioned above. This is the first work that analyzes a fully stochastic BFGS algorithm, which also avoids time consuming or even impossible line-search steps.

A third class of problems that the empirical processes framework can be applied to is to study the optimization of compositions of stochastic functions. A multi-level Monte Carlo based unbiased gradient generation method is introduced into stochastic optimization algorithms for minimizing function compositions. Based on this, standard stochastic optimization algorithms can be applied to these problems directly.

Table of Contents

List of Tables

List of Figures

Acknowledgements

First and foremost, I would like to express my sincere gratitude to my academic advisors Professor Donald Goldfarb and Garud Iyengar. This thesis would not have been possible without their constant guidance and support during my graduate study. They not only provided valuable advice on my research, but also cared about my personal and career development, for which I am truly indebted and grateful.

I would also like to thank my committee members, Professors John Wright, Henry Lam, and Krzysztof Choromanski for their advice and helpful insights, and for careful reading of my thesis manuscript.

I would like to thank all the faculty and staff members of the Department of Industrial Engineering and Operations Research for creating such a supportive community. I am also obliged to my friends and fellow students for their sincere friendship.

I especially thank my family for their unconditional support and understanding throughout the years.

Chapter 1: Introduction and Background

1.1 Overview

In the era of big data, the main challenge that the field of optimization faces is the trade-off between solution accuracy and algorithm running time. To address this issue, a large number of stochastic optimization algorithms have been developed, especially for convex problems. When the directions and step sizes are both stochastic and depend on each other, analyzing the convergence properties of such algorithms poses great technical difficulty. To address this issue, we developed a theoretical framework based on concentration inequalities for empirical processes to better design algorithms and analyze their convergence properties in the presence of complex dependence.

Based on this framework, we proposed a stochastic away-step Frank-Wolfe algorithm and a stochastic pairwise-step Frank-Wolfe algorithm in [1] for solving strongly convex problems with polytope constraints and proved that both of them converge linearly to the optimal solution in expectation and almost surely. Numerical results showed that the proposed algorithms are faster and more stable than most of their competitors.

Another important issue for current stochastic optimization algorithms is step-size tuning. Most currently available stochastic algorithms are provably convergent only if either diminishing or infinitesimal step-sizes are used. As a result, practitioners have to put a lot of effort into tuning step-sizes and other parameters in most of the algorithms that are used. To address this topic, we proposed an adaptive step-size framework based on local curvature for a number of stochastic algorithms for self-concordant optimization problems in [2]. Most notably, we proposed a stochastic BFGS algorithm without line-search, and proved that it converges linearly globally and super-linearly locally using the techniques mentioned above that we developed to resolve the dependence issue. This is the first work that analyzes a fully stochastic BFGS algorithm, which also

avoids time consuming or even impossible line-search steps.

A third class of problems that we studied addresses the optimization of compositions of stochastic functions. Problems that have this structure arise in many statistics and machine learning applications, such as parameter estimation for conditional random fields and for maximizing the partial likelihood in proportional-hazards models. In these problems, obtaining a stochastic gradient is already computationally difficult. Therefore most current approaches use biased stochastic gradients in their algorithmic design, which results in non-optimal iteration complexities. To solve this problem, we introduced multi-level Monte Carlo methods into optimization algorithms for minimizing function compositions in [3]. We proposed simulation algorithms that generate unbiased gradient estimates with finite variance and finite expected computational cost. As a result, standard stochastic optimization algorithms can be applied to these problems directly. We also modified our simulation algorithms to enable them to incorporate various acceleration schemes.

1.2 The Empirical Processes Framework

Empirical processes generalizes the Glivenko-Cantelli theorem and the Donsker theorem to more general function classes. It has been widely used in analyzing large-sample properties of statistical estimators, especially M-estimators, in parametric, non-parametric, and semi-parametric statistical models. The uniform convergence results in the theory of empirical processes naturally resolve the complex dependence between the estimator and the samples when analyzing its properties. The complexity of the underlying function class, which is measured by the covering number, packing number, or bracketing number, plays an important role in the development of empirical processes theory. However, this set of tools has not been previously used in analyzing the properties of stochastic optimization algorithms. In chapter 2 of this thesis, we propose and develop a theoretical framework that is based on concentration inequalities for empirical processes for proving the convergence results for stochastic optimization algorithms under dependence.

1.3 The Frank-Wolfe Algorithm and Its Variants

The Frank-Wolfe algorithm, which is also known as the conditional gradient algorithm, was proposed in 1956 to minimize a convex function over a convex and compact feasible region. More specifically, for solving $\min_{x\in\mathcal{D}} F(x)$, where $F(\cdot)$ is a convex function and $\mathcal{D}$ is a convex and compact set, the Frank-Wolfe algorithm proceeds as follows:

Algorithm 1 The Frank-Wolfe Algorithm

Input: Initial solution $x^{(1)} \in \mathcal{D}$.

for $k = 1, 2, \ldots$ **do**

Set $p^{(k)} = \arg\min_{s\in\mathcal{D}} \langle \nabla F(x^{(k)}), s\rangle$.

Set $d^{(k)} = p^{(k)} - x^{(k)}$.

Set $x^{(k+1)} = x^{(k)} + \gamma^{(k)} d^{(k)}$, where $\gamma^{(k)} = \frac{2}{k+2}$ or obtain by line-search.

end for

Return: $x^{(k+1)}$.

The Frank-Wolfe Algorithm has become popular recently because it performs a sparse update at each step. For a good review of what was known about the FW algorithm until a few years ago, see [4]. When the feasible region $\mathcal{D}$ is a polytope, it is well-known that this algorithm converges sub-linearly with rate $O(1/k)$ because of the so-called zig-zagging phenomenon [5]. Especially if the optimal solution x_* does not lie in the relative interior of $\mathcal{D}$, the FW algorithm tends to zig-zag amongst the vertices that define the facet containing x_*. One way to overcome this zig-zagging problem is to keep track of the "active“ vertices (the vertices discovered previously in the FW algorithm) and move away from the “worst” of these in some iterations. The Away-step Frank-Wolfe algorithm (AFW) and the Pairwise Frank-Wolfe algorithm (PFW) in [5] are two notable variants based on this idea.

In chapter 3 of this thesis, we will discuss in details and analyze the convergence properties of stochastic versions of these two algorithms. Moreover, in large-scale numerical experiments, the proposed algorithms perform as well as or better than their stochastic competitors in actual CPU

time.

1.4 Local Curvature Based Adaptive Step-size Algorithms

Many stochastic algorithms have been proposed to solve the generic stochastic optimization problem

$$\min_{x\in\mathbb{R}^d} \mathbb{E}_\xi f(x,\xi),$$

and its finite sample version

$$\min_{x\in\mathbb{R}^d} \frac{1}{n}\sum_{i=1}^{n} f(x,\xi_i),$$

including stochastic gradient descent (SGD), and variance-reduced extensions of SGD, such as SVRG [6], SAG [7], and SAGA [8]. These first-order methods extend gradient descent to the stochastic setting. It is natural to consider stochastic extensions of quasi-Newton and second-order methods. One such method, the Newton Incremental Method (NIM) [9], combines cyclic updating of a fixed collection of functions $f(x,\xi_1),\ldots,f(x,\xi_m)$ with Newton's method, and attains local superlinear convergence.

One of the key obstacles in developing stochastic extensions of quasi-Newton methods is the necessity of selecting appropriate step sizes. The analysis of the global convergence of the BFGS method [10] and other members of Broyden's convex class [11] assumes that Armijo-Wolfe inexact line search is used. This is rather undesirable for a stochastic algorithm, as line search is both computationally expensive and difficult to analyze in a probabilistic setting. However, there is a special class of functions, the self-concordant functions, whose properties allow us to compute an adaptive step size based on local curvature and thereby avoid performing line searches. In [12], it is shown that the BFGS [13][14][15][16] method with adaptive step sizes converges superlinearly when applied to self-concordant functions.

In chapter 4 of this thesis, we will introduce class of stochastic, adaptive methods for minimizing self-concordant functions which can be expressed as an expected value. These methods generate an estimate of the true objective function by taking the empirical mean over a sample

drawn at each step, making the problem tractable. The use of adaptive step sizes, which are based on local curvature, eliminates the need for the user to supply a step size. Methods in this class include extensions of gradient descent (GD) and BFGS. Based on the empirical processes framework, we can show that, given a suitable amount of sampling, our stochastic adaptive GD method attains linear convergence in expectation, and with further sampling, our stochastic adaptive BFGS method attains R-superlinear convergence.

1.5 Unbiased Simulation Method for Stochastic Composition Optimization Problems

Most of the algorithms for solving the generic stochastic optimization problem,

$$\min_{x\in\mathcal{D}} \mathbb{E}_\xi f(x;\xi),$$

implicitly assume the gradient of each member function $f(\cdot;\xi)$ is easy to compute. But this assumption does not hold in the so-called stochastic composition optimization (SCO) problem [17]:

$$\min_{x\in\mathcal{D}} F(x) \triangleq \mathbb{E}_v f_v(\mathbb{E}_w g_w(x)),$$

where v and w are random variables with certain known joint distributions nor its finite sample version:

$$\min_{x\in\mathcal{D}} F_n(x) \triangleq \frac{1}{n}\sum_{i=1}^{n} f_i\{\frac{1}{m_i}\sum_{j=1}^{m_i} g_{ij}(x)\}. \tag{1.1}$$

As far as we know, all current algorithms that are used to solve SCO problems are based on *biased* stochastic gradient oracles.

In chapter 5 of this thesis, we introduce unbiased gradient simulation algorithms that are based on a multilevel Monte Carlo technique for solving smooth SCO problems. Based on our unbiased gradient simulation algorithms, a stochastic composition optimization problem can be considered as a generic stochastic optimization problem.

Chapter 2: An Empirical Processes Framework

2.1 The Framework

The goal of developing the empirical prcoesses framework described in this section is to unify the convergence analysis of stochastic optimization algorithms under dependence. These results originate in empirical process theory [18]. The problem to be minimized has the form

$$\min_{x\in\mathbb{R}^d} F(x) \equiv \mathbb{E}_\xi f(x,\xi). \tag{2.1}$$

We require the following assumptions on F and f for the analysis.

Assumptions:

1. There exist compact sets $\mathcal{D}_0$ and $\mathcal{D}$ with $x^* \in \mathcal{D}$ and $\mathcal{D}_0 \subseteq \mathcal{D} \subset \mathbb{R}^d$, such that if x_0 is chosen in $\mathcal{D}_0$, then for all possible realizations of the samples $\xi_1, \ldots, \xi_{m^{(k)}}$ for every k, the sequence of iterates $\{x_k\}_{k=0}^\infty$ produced by the algorithm is contained within $\mathcal{D}$. We write $D = \sup\{\|x-y\| : x, y \in \mathcal{D}\}$ for the diameter of $\mathcal{D}$.

   Furthermore, we assume that the objective values and gradients are bounded:

$$u = \sup_\xi \sup_{x\in\mathcal{D}} f(x,\xi) < \infty$$

$$l = \inf_\xi \inf_{x\in\mathcal{D}} f(x,\xi) > -\infty$$

$$\gamma = \sup_\xi \sup_{x\in\mathcal{D}} \|\nabla f(x,\xi)\| < \infty$$

2. There exists $0 < L < \infty$, such that $\sup_\xi |f(x,\xi) - f(y,\xi)| < L\|x-y\|$.

The key theorem of this framework is a concentration bound which limits the divergence of a

sub-sampled function $F^{(k)}(x)$ from $F(x)$.

Theorem 2.1.1. *Let $m^{(k)} \in \mathbb{N}_+$, and $F^{(k)}(x) \equiv \frac{1}{m^{(k)}} \sum_{i=1}^{m^{(k)}} f(x, \xi_i^{(k)})$, where $\xi_1^{(k)}, \ldots, \xi_{m^{(k)}}^{(k)}$ are i.i.d. following the distribution of ξ. For any $\delta > 0$ and $0 < \epsilon < \min\{D, \frac{\delta}{2L}\}$, we have*

$$\mathbb{P}(\sup_{x \in \mathcal{D}} |F^{(k)}(x) - F(x)| \geq \delta) \leq 2d^{d/2} \frac{D^d}{\epsilon^d} \exp\{ - \frac{m^{(k)}(\delta - 2L\epsilon)^2}{2(u-l)^2}\}. \tag{2.2}$$

Moreover, let $x_* = \arg\min_{x \in \mathcal{D}} F(x)$ and $x_*^{(k)} = \arg\min_{x \in \mathcal{D}} F^{(k)}(x)$. For $m^{(k)} \geq 3$, we have

$$\mathbb{E} \sup_{x \in \mathcal{D}} |F^{(k)}(x) - F(x)| \leq C \sqrt{\frac{\log m^{(k)}}{m^{(k)}}} \tag{2.3}$$

and

$$\mathbb{E}|F^{(k)}(x_*^{(k)}) - F(x_*)| \leq C \sqrt{\frac{\log m^{(k)}}{m^{(k)}}}, \tag{2.4}$$

where

$$C = 4(|u| + |l|) d^{d/2} D^d \exp\{ - d \log \frac{u-l}{2\sqrt{2}L}\} + (u-l)\sqrt{d+1}.$$

Proof of this theorem can be found in next section.

Based on this theorem, we are ready to state our theoretical framework for proving the convergence of stochastic algorithms. Let $\{y^{(k)}\}$ be a sequence of iterates that are generated by an iterative and deterministic algorithm which solves problem (2.1). Assume the iterates satisfies the property that

$$F(y^{(k+1)}) - F(y_*) \leq \rho^{(k)} \{F(y^{(k)}) - F(y_*)\}, \tag{2.5}$$

where $\rho^{(k)} \in (0, 1)$ and only depends on the smoothness properties of $F(\cdot)$ and other deterministic

properties such as iteration count and hyper-parameters. Now consider a stochastic version of the deterministic algorithm which uses an average i.i.d. sub-sampled quantity to substitute for the original deterministic quantity. Let $\{x^{(k)}\}$ be the sequence of iterates generated by this stochastic algorithm. Then we have

$$F(x^{(k+1)}) - F(x_*) = \{F(x^{(k+1)}) - F^{(k)}(x^{(k+1)})\} + \{F^{(k)}(x^{(k+1)}) - F^{(k)}(x_*^{(k)})\} + \{F^{(k)}(x_*^{(k)}) - F(x_*)\}.$$

For the terms in the first set of brackets on the right hand side of the equation above, we have

$$\begin{aligned}\mathbb{E}\{F(x^{(k+1)}) - F^{(k)}(x^{(k+1)})\} &\leq \mathbb{E}|F(x^{(k+1)}) - F^{(k)}(x^{(k+1)})| \\ &\leq \mathbb{E}\sup_{x\in\mathcal{D}}|F^{(k)}(x) - F(x)| \\ &\leq C\sqrt{\frac{\log m^{(k)}}{m^{(k)}}}. \end{aligned} \tag{2.6}$$

Similarly, for the terms in the third set of brackets on the right hand side, we have

$$\mathbb{E}\{F^{(k)}(x_*^{(k)}) - F(x_*)\} \leq \mathbb{E}|F^{(k)}(x_*^{(k)}) - F(x_*)| \leq C\sqrt{\frac{\log m^{(k)}}{m^{(k)}}}. \tag{2.7}$$

For the second term, note that $x^{(k+1)}$ only depends on the samples generated in k-th iteration and $x^{(k)}$. As a result, we may consider it as running the deterministic algorithms on a deterministic function $F^{(k)}(\cdot)$. Thus the property (2.5) can be directly applied here; that is,

$$\begin{aligned}&F^{(k)}(x^{(k+1)}) - F^{(k)}(x_*^{(k)}) \\ &\leq \rho^{(k)}\{F^{(k)}(x^{(k)}) - F^{(k)}(x_*^{(k)})\} \\ &= \rho^{(k)}\{F(x^{(k)}) - F(x_*)\} + \rho^{(k)}\{F^{(k)}(x^{(k)}) - F(x^{(k)})\} + \rho^{(k)}\{F(x_*) - F(x_*^{(k)})\}.\end{aligned}$$

Taking expectation on both sides of the inequality above and using (2.6) and (2.7), we have

$$\begin{aligned}\mathbb{E}\{F^{(k)}(x^{(k+1)}) - F^{(k)}(x_*^{(k)})\} &\leq \rho^{(k)}\mathbb{E}\{F^{(k)}(x^{(k)}) - F^{(k)}(x_*^{(k)})\} + 2\rho^{(k)}C\sqrt{\frac{\log m^{(k)}}{m^{(k)}}} \\ &\leq \rho^{(k)}\mathbb{E}\{F^{(k)}(x^{(k)}) - F^{(k)}(x_*^{(k)})\} + 2C\sqrt{\frac{\log m^{(k)}}{m^{(k)}}},\end{aligned}$$

where the last inequality follows from $\rho^{(k)} < 1$.

Combining these three inequalities, we have

$$\begin{aligned}\mathbb{E}F(x^{(k+1)}) - F(x_*) &\leq \rho^{(k)}\mathbb{E}\{F(x^{(k)}) - F(x_*)\} + 4C\sqrt{\frac{\log m^{(k)}}{m^{(k)}}} \\ &\leq (F(x^{(1)}) - F(x_*))\prod_{i=1}^{k}\rho^{(i)} + 4C\sum_{i=1}^{k}\sqrt{\frac{\log m^{(i)}}{m^{(i)}}}\prod_{j=i+1}^{k}\rho^{(j)}.\end{aligned}$$

Therefore, the rate of convergence of this stochastic algorithm is determined by the rate of of convergence of its deterministic version $\rho^{(k)}$ and the sampling rate, $m^{(k)}$, in every iteration.

Remark. Our analysis above focuses on the iteration complexities instead of sample complexities. In machine learning, the sample complexity of an algorithm represents the total number of training samples needed in order to learn the target function with arbitrarily high probability. This concept is also important in the context of optimization algorithms, since many stochastic algorithms, such as Stochastic Gradient Descent (SGD), use only one sample in every iteration and obtaining such a sample is the most time consuming step. In this case, the sample complexity is a good indicator of the performance of these algorithms. However, in many other algorithms such as the stochastic Frank-Wolfe (conditional gradient) algorithms and stochastic Quasi-Newton algorithms, this is not the case. In stochastic Frank-Wolfe algorithms, in every iteration, one needs to solve a linear programming problem which is typically much more time consuming than sampling. Similarly, the matrix vector multiplications in the Quasi-Newton algorithms can be more time consuming than sampling. In such cases, the iteration complexity is a much more reasonable indicator of the

performance of an algorithm.

2.2 Proof of Theorem 2.1.1

We need the following definition and lemma to prove the Theorem 2.1.1.

Definition [Bracketing Number] Let $\mathcal{F}$ be a class of functions. Given two functions l and u, the bracket $[l, u]$ is the set of all function f with $l \leq f \leq u$. An ϵ-bracket in L_1 is a bracket $[l, u]$ with $\mathbb{E}|u - l| < \epsilon$. The bracketing number $N_{[]}(\epsilon, \mathcal{F}, L_1)$ is the minimum number of ϵ-brackets needed to cover $\mathcal{F}$. (The bracketing functions l and u must have finite L_1-norms but need not belong to $\mathcal{F}$).

The bracketing number is a quantity that measures the complexity of a function class. The lemma below provides an upper bound for a function class indexed by a finite dimensional bounded set. This result can be found in any empirical processes textbook such as [18]. For completeness, we provide a proof.

Lemma 2.2.1. *Let $\mathcal{F} = \{f_\theta \mid \theta \in \Theta\}$ be a collection of measurable functions indexed by a bounded subset $\Theta \subset \mathbb{R}^d$. Denote $D_\Theta = \sup\{\|\theta_1 - \theta_2\| \mid \theta_1, \theta_2 \in \Theta\}$. Suppose that there exists a measurable function g such that*

$$|f_{\theta_1}(\xi) - f_{\theta_2}(\xi)| \leq g(\xi)\|\theta_1 - \theta_2\| \tag{2.8}$$

for every $\theta_1, \theta_2 \in \Theta$. If $\|g(\xi)\|_1 \equiv \int |g(\xi)|dP < \infty$, then the bracketing numbers satisfy

$$N_{[]}(\epsilon\|g\|_1, \mathcal{F}, L_1) \leq (\frac{\sqrt{d}D_\Theta}{\epsilon})^d$$

for every $0 < \epsilon < D_\Theta$.

Proof. To prove the result, we use brackets of the type $[f_\theta - \epsilon g/2, f_\theta + \epsilon g/2]$ for θ that ranging over a suitably chosen subset of Θ and these brackets have L_1-size $\epsilon\|g\|_1$. If $\|\theta_1 - \theta_2\| \leq \epsilon/2$, then by the Lipschitz condition (2.8), we have $f_{\theta_1} - \epsilon g/2 \leq f_{\theta_2} \leq f_{\theta_1} + \epsilon g/2$. Therefore, the brackets

cover $\mathcal{F}$ if θ ranges over a grid of meshwidth $\epsilon/\sqrt{d}$ over Θ. This grid has at most $(\sqrt{d}D_\Theta/\epsilon)^d$ grid points. Therefore the bracketing number $N_{[]}(\epsilon\|g\|_1, \mathcal{F}, L_1)$ can be bounded by $(\sqrt{d}D_\Theta/\epsilon)^d$. □

Remark: The bracketing number has a very close relationship with the covering number, which is a better known quantity in machine learning. Let $N(\epsilon, \mathcal{F}, L_1)$ be the covering number of the set $\mathcal{F}$; that is, the minimal number of balls of L_1-radius ϵ needs to cover the set $\mathcal{F}$. Then the relation, $N(\epsilon, \mathcal{F}, L_1) \leq N_{[]}(2\epsilon, \mathcal{F}, L_1)$, between covering number and bracketing number always holds. Moreover, this concept is also closely related to the VC-dimension. Usually, constructing and counting the number of brackets for a class of functions is easier to do than computing the minimum number of balls that covers the class.

Now, we are ready to prove Theorem 2.1.1.

Proof. Consider the function class $\mathcal{F} = \{f(x, \cdot) \mid x \in \mathcal{D}\}$ as defined in (2.1). Since $f(\cdot, \xi)$ each is assumed to be Lipschitz continuous with Lipschitz constant L, we must have $|f(x, \xi) - f(y, \xi)| \leq L\|x - y\|$. Moreover, the index set $\mathcal{D} \in \mathbb{R}^p$ for the function class $\mathcal{F}$ is assume to be bounded. Therefore all conditions for Lemma 2.2.1 are satisfied and hence the number of brackets of the type $[f(x, \cdot) - \epsilon L, f(x, \cdot) + \epsilon L]$ satisfies

$$N_{[]}(\epsilon L, \mathcal{F}, L_1) \leq (\sqrt{d})^d (\frac{D}{\epsilon})^d,$$

for every $0 < \epsilon < D$, where $D = \sup\{\|x - y\| \mid x, y \in \mathcal{D}\}$. Let $\Gamma \subset \mathcal{D}$ denote the set of indices of the centers of these brackets and $\xi_1, \ldots \xi_{m^{(k)}}$ be the i.i.d. samples drawn at the k-th iteration of the algorithm. Since the brackets centered at Γ cover $\mathcal{F}$, we must have

$$\sup_{x \in \mathcal{D}} |\frac{1}{m^{(k)}} \sum_{i=1}^{m^{(k)}} f(x, \xi_i) - \mathbb{E}f(x, \xi_i)| \leq \max\{|\frac{1}{m^{(k)}} \sum_{i=1}^{m^{(k)}} f(y, \xi_i) - \mathbb{E}f(y, \xi_i)| \mid y \in \Gamma\} + 2\epsilon L.$$

Consequently, for every $\delta \geq 0$ and $\epsilon < \min\{\delta/(2L), D\}$,

$$
\begin{aligned}
&\mathbb{P}\{\sup_{x\in\mathcal{D}} |\frac{1}{m^{(k)}}\sum_{i=1}^{m^{(k)}} f(x,\xi_i) - \mathbb{E}f(x,\xi_i)| \geq \delta\} \\
&\leq \mathbb{P}\{\max\{|\frac{1}{m^{(k)}}\sum_{i=1}^{m^{(k)}} f(y,\xi_i) - \mathbb{E}f(y,\xi_i)| \mid y \in \Gamma\} + 2\epsilon L \geq \delta\} \\
&\leq \sum_{y\in\Gamma} \mathbb{P}\{|\frac{1}{m^{(k)}}\sum_{i=1}^{m^{(k)}} f(y,\xi_i) - \mathbb{E}f(y,\xi_1)| \geq \delta - 2\epsilon L\} && \text{(union bound)} \\
&\leq \sum_{y\in\Gamma} 2\exp\{-\frac{2m^{(k)}(\delta - 2L\epsilon)^2}{(u-l)^2}\} && \text{(Hoeffding inequality)} \\
&\leq 2(\sqrt{d})^d(\frac{D}{\epsilon})^d \exp\{-\frac{2m^{(k)}(\delta - 2L\epsilon)^2}{(u-l)^2}\}. && (|\Gamma| \leq (\sqrt{d})^d(\tfrac{D}{\epsilon})^d)
\end{aligned}
$$

Since by definition, $F^{(k)}(x) = \frac{1}{m^{(k)}}\sum_{i=1}^{m^{(k)}} f(\xi_i, x)$ and $F(x) = \mathbb{E}f(\xi_i, x)$, then (2.2) follows.

To show (2.3), first note that both $F^{(k)}(\cdot)$ and $F(\cdot)$ are bounded by l and u; hence, $\sup_{x\in\mathcal{D}} |F^{(k)}(x) - F(x)| \leq 2(|u| + |l|)$. Then for every $\delta \geq 0$, we have,

$$
\begin{aligned}
&\mathbb{E}\sup_{x\in\mathcal{D}} |F^{(k)}(x) - F(x)| \\
&\leq 2(|u| + |l|)\mathbb{P}\{\sup_{x\in\mathcal{D}} |F^{(k)}(x) - F(x)| \geq \delta\} + \delta\, \mathbb{P}\{\sup_{x\in\mathcal{D}} |F^{(k)}(x) - F(x)| < \delta\} \\
&\leq 4(|u| + |l|)(\sqrt{d})^d(\frac{D}{\epsilon})^d \exp\{-\frac{2m^{(k)}(\delta - 2L\epsilon)^2}{(u-l)^2}\} + \delta \\
&\leq 4(|u| + |l|)(\sqrt{d})^d D^d \exp\{-\frac{2m^{(k)}(\delta - 2L\epsilon)^2}{(u-l)^2} + d\log\frac{1}{\epsilon}\} + \delta.
\end{aligned}
$$

Now let $\delta = \frac{(u-l)\sqrt{4(d+1)\log\sqrt{m^{(k)}}}}{\sqrt{m^{(k)}}\sqrt{2}}$, $\epsilon = \frac{(u-l)}{2L\sqrt{m^{(k)}}\sqrt{2}}$. Then

$$
\begin{aligned}
\mathbb{E}\sup_{x\in\mathcal{P}} |F^{(k)}(x) - F(x)| \leq 4(|u| + |l|)(\sqrt{d})^d D^d \exp\{&-(\sqrt{4(d+1)\log\sqrt{m^{(k)}}} - 1)^2 - d(\log\frac{u-l}{2\sqrt{2}L}) \\
&+ d\log\sqrt{m^{(k)}}\} + \frac{(u-l)\sqrt{4(d+1)\log\sqrt{m^{(k)}}}}{\sqrt{m^{(k)}}\sqrt{2}}.
\end{aligned}
$$

Note that $(x-1)^2 \geq x^2/4$ when $x \geq 2$. Thus, for $m^{(k)} \geq 3$ and $d \geq 1$, $\sqrt{4(d+1)\log\sqrt{m^{(k)}}} \geq 2$.

Therefore

$$
\begin{aligned}
&\mathbb{E}\sup_{x\in\mathcal{D}}|F^{(k)}(x)-F(x)| \\
&\leq 4(|u|+|l|)(\sqrt{d})^d D^d \exp\{-(d+1)\log(\sqrt{m^{(k)}})+d\log\sqrt{m^{(k)}}-d(\log\frac{u-l}{2\sqrt{2}L})\} \\
&\quad+\frac{(u-l)\sqrt{4(d+1)\log\sqrt{m^{(k)}}}}{\sqrt{m^{(k)}}\sqrt{2}} \\
&\leq C\sqrt{\frac{\log m^{(k)}}{m^{(k)}}},
\end{aligned}
$$

where $C = 4(|u|+|l|)(\sqrt{d})^d D^d \exp\{-d(\log\frac{u-l}{2\sqrt{2}L})\} + (u-l)\sqrt{d+1}$.

Next, we will obtain a bound for $\mathbb{E}|F^{(k)}(x_*^{(k)}) - F(x^*)|$. (2.2) implies both

$$F(x_*^{(k)}) - \delta \leq F^{(k)}(x_*^{(k)}) \leq F(x_*^{(k)}) + \delta \tag{2.9}$$

and

$$F(x^*) - \delta \leq F^{(k)}(x^*) \leq F(x^*) + \delta \tag{2.10}$$

happen with probability at least $1-2(\sqrt{d})^d(\frac{D}{\epsilon})^d \exp\{-\frac{m^{(k)}(\delta-2L\epsilon)^2}{2(u-l)^2}\}$. Consequently, on the one hand

$$
\begin{aligned}
F^{(k)}(x_*^{(k)}) &\geq F(x_*^{(k)}) - \delta && \text{(by 2.9)} \\
&\geq F(x^*) - \delta && \text{(optimality of } x^* \text{ for } F(\cdot))
\end{aligned}
$$

On the other hand,

$$
\begin{aligned}
F^{(k)}(x_*^{(k)}) &\leq F^{(k)}(x^*) && \text{(optimiality of } x_*^{(k)} \text{ for } F^{(k)}(\cdot)) \\
&\leq F(x^*) + \delta && \text{(by 2.10)}
\end{aligned}
$$

Therefore, we have

$$\mathbb{P}\{|F^{(k)}(x_*^{(k)}) - F(x_*)| \geq \delta\} \leq 2(\sqrt{d})^d (\frac{D}{\epsilon})^d \exp\{-\frac{m^{(k)}(\delta - 2L\epsilon)^2}{2(u-l)^2}\},$$

and hence, $\mathbb{E}|F^{(k)}(x_*^{(k)}) - F(x^*)| = C\sqrt{\frac{\log m^{(k)}}{m^{(k)}}}$. □

2.3 From Convergence in Expectation to Almost Sure Convergence.

In this section, we will discuss a simple technique that enables us to derive the almost sure convergence of the solutions of a stochastic algorithm from its convergence rate in expectation. Let $\{x^{(k)}\}$ be the solutions generated by a stochastic algorithm for solving problem (2.1). Assume that $\mathbb{E}F(x^{(k)}) - F(x_*) \leq \alpha^{(k)}$ and $\sum_{k=1}^{\infty} \alpha^{(k)} < \infty$. Then $F(x^{(k)}) \to F(x_*)$ almost surely as $k \to \infty$.

To prove this, for every $\epsilon > 0$, let $E^{(k)}$ denote the event that $F(x^{(k)}) - F(x_*) > \epsilon$. By the Markov inequality

$$\sum_{k=2}^{\infty} \mathbb{P}(E^{(k)}) = \sum_{k=2}^{\infty} \mathbb{P}((F(x^{(k)}) - F(x_*)) > \epsilon) \leq \sum_{k=2}^{\infty} \frac{\mathbb{E}\{F(x^{(k)}) - F^*\}}{\epsilon} < \frac{1}{\epsilon} \sum_{k=2}^{\infty} \alpha^{(k)} < \infty.$$

Therefore the Borel-Cantelli lemma implies that $\mathbb{P}(\limsup_{k\to\infty} E^{(k)}) = 0$, and hence, $F(x^{(k)}) - F(x_*) \to 0$ almost surely as $k \to \infty$.

Remark. As we have shown, when the rate of convergence in expectation satisfies certain conditions, we can get almost sure convergence for free. This result guarantees the global convergence of every individual sample path, which is a key component for analyzing local convergence properties of the stochastic quasi-Newton algorithms.

Chapter 3: Linear Convergence of Stochastic Frank Wolfe Variants

3.1 Motivation

The recent trend of using a large number of parameters to model large datasets in machine learning and statistics has created a strong demand for optimization algorithms that have low computational cost per iteration and exploit model structure. Regularized empirical risk minimization (ERM) is an important class of problems in this area that can be formulated as smooth constrained optimization problems. A popular approach for solving such ERM problems is the proximal gradient method which solves a projection sub-problem in each iteration. The major drawback of this method is that the projection step can be expensive in many situations. As an alternative, the Frank-Wolfe (FW) algorithm [19], also known as the conditional gradient method, solves a linear optimization sub-problem in each iteration, which is much faster than the standard projection technique when the feasible set is a simple polytope [20]. When the number of observations in ERM is large, calculating the gradient in every FW iteration becomes a computationally intensive task. The question of whether 'cheap' stochastic gradients can be used as a surrogate in FW immediately arises.

3.2 Contribution

In this chapter, we show that the Away-step Stochastic Frank-Wolfe (ASFW) algorithm converges linearly in expectation and on each sample path, the algorithm converges linearly. We also show that if an algorithm converges linearly in expectation then it converges linearly almost surely. The major technical difficulty of analyzing the ASFW algorithm is the lack of tools that combine stochastic arguments and combinatorial arguments. In order to solve this problem and prove our convergence results, a novel proof technique based on the empirical processes framework, that we

introduced in Chapter 2, is developed. This technique is then applied to prove the linear convergence in expectation and almost sure convergence of each sample path of another Frank-Wolfe variant, the Pairwise Stochastic Frank-Wolfe (PSFW) algorithm. In our large-scale numerical experiments, the proposed algorithms outperform their competitors in all different settings.

3.3 Related Work

The Frank-Wolfe algorithm was proposed sixty years ago ([19]) for minimizing a convex function over a polytope and is known to converge at an $O(1/k)$ rate. In [21] the same convergence rate was proved for compact convex constraints. When both objective function and the constraint set are strongly convex, [22] proved that the Frank-Wolfe algorithm has an $O(1/k^2)$ rate of convergence with a properly chosen step size. Motivated by removing the influence of "bad" visited vertices, the away-steps variant of the Frank-Wolfe algorithm was proposed in [23]. Later, [24] showed that this variant converges linearly under the assumption that the objective function is strongly convex and the optimum lies in the interior of the constraint polytope. Recently, [25] and [26] extended the linear convergence result by removing the assumption of the location of the optimum and [27] extended it further by relaxing the strongly convex objective function assumption. Stochastic Frank-Wolfe algorithms were studied by [28] and [29] and an $O(1/k)$ rate of convergence in expectation were established. [30] considered the Stochastic Varianced-Reduced Frank-Wolfe method (SVRF) which also has convergence rate $O(1/k)$ in expectation. In addition, the Frank-Wolfe algorithm has been applied to solve several different classes of problems, including non-linear SVM ([31]), structural SVM ([32, 33]), and comprehensive principal component pursuit ([34]) among many others. We compare FW variants and other useful algorithms such as the Prox-SVRG of [35] and the stochastic variance reduced FW algorithm of [30] in Table 3.1, where we summarize the required conditions for convergence and the given complexity bounds, the number of exact and stochastic gradient oracle calls, the number of linear optimization oracle (LO) calls and the number of projection calls in order to obtain an ϵ-approximate solution.

Algorithm	Extra conditions	FG	SG	LO	Projection
FW	bounded constraint	$O(\frac{1}{\epsilon})$	NA	$O(\frac{1}{\epsilon})$	NA
Away-step FW	polytope constraint strongly convex objective	$O(\log \frac{1}{\epsilon})$	NA	$O(\log \frac{1}{\epsilon})$	NA
Pairwise FW	polytope constraint strongly convex objective	$O(\log \frac{1}{\epsilon})$	NA	$O(\log \frac{1}{\epsilon})$	NA
SVRF	bounded constraint	$O(\log \frac{1}{\epsilon})$	$O(\frac{1}{\epsilon^2})$	$O(\frac{1}{\epsilon})$	NA
Prox-SVRG	strongly convex objective	$O(\log \frac{1}{\epsilon})$	$O(m \log \frac{1}{\epsilon})$	NA	$O(m \log \frac{1}{\epsilon})$
ASFW	polytope constraint strongly convex objective	NA	$O(1/\epsilon^{4\eta})$, $0 < \eta < 1$	$O(\log \frac{1}{\epsilon})$	NA
PSFW	polytope constraint strongly convex objective	NA	$O(1/\epsilon^{(6\|V\|!+2)\zeta})$, $0 < \zeta < 1$	$O(\log \frac{1}{\epsilon})$	NA

Table 3.1: Comparisons of algorithms in terms of their requirements and theoretical performance to get an ϵ-approximate solution. In Table 3.1, FG denotes full gradient; SG denotes stochastic gradients; and LO denotes linear optimizations. In Prox-SVRG, m is the number of iterations in each epoch. In PSFW, $|V|$ is the number of vertices in the polytope constraint.

3.4 Problem description.

Consider the minimization problem

$$\min_{x \in \mathcal{P}} \Big\{ F(x) \equiv \frac{1}{n} \sum_{i=1}^{n} f_i(x) \Big\}, \tag{P1}$$

where $\mathcal{P}$ is a polytope, i.e., a non-empty compact polyhedron given by $\mathcal{P} = \{x \in \mathbb{R}^p : Cx \leq d\}$ for some $C \in \mathbb{R}^{m\times p}$, $d \in \mathbb{R}^m$. Therefore, the set of vertices V of the polytope $\mathcal{P}$ has finitely many elements. Let $D = \sup\{\|x-y\| \mid x, y \in \mathcal{P}\}$ be the diameter of $\mathcal{P}$. For every $i = 1, \ldots, n$, $f_i : \mathbb{R} \to \mathbb{R}$ is a strongly convex function with parameter σ_i with an L_i Lipschitz continuous gradient. From another point of view, (P1) can be reformulated as the stochastic optimization problem

$$\min_{x\in\mathcal{P}} \Big\{\frac{1}{n}\sum_{i=1}^{n} f_i(x) \equiv \mathbb{E}f(\xi, x)\Big\}, \tag{SP1}$$

where ξ is a random variable that follows a discrete uniform distribution on $\{1, \ldots, n\}$, $f(i, x) = f_i(x)$ for every $i = 1, \ldots, n$ and $x \in \mathcal{P}$. Furthermore, define $\nabla f(\xi, x) = \nabla f_\xi(x)$.

3.5 The Frank-Wolfe Algorithms.

In contrast to the projected gradient algorithm, the Frank-Wolfe algorithm calls a linear optimization oracle instead of a projection oracle in every iteration.

Algorithm 2 The Frank-Wolfe Algorithm

Input: $x^{(1)} \in \mathcal{P}$, $F(\cdot)$

for $k = 1, 2, \ldots$ **do**

Set $p^{(k)} = \arg\min_{s\in\mathcal{P}}\langle\nabla F(x^{(k)}), s\rangle$.

Set $d^{(k)} = p^{(k)} - x^{(k)}$.

Set $x^{(k+1)} = x^{(k)} + \gamma^{(k)}d^{(k)}$, where $\gamma^{(k)} = \frac{2}{k+2}$ or obtain by line-search.

end for

Return: $x^{(k+1)}$.

The Frank-Wolfe Algorithm has become popular recently because it performs a sparse update at each step. See [4] for a good review of the classical results on the FW algorithm. It is well-known that this algorithm converges sub-linearly with rate $O(1/k)$ because of the so-called zig-zagging phenomenon ([5]). Especially when the optimal solution x^* does not lie in the relative interior of

$\mathcal{P}$, the FW algorithm tends to zig-zag amongst the vertices that define the facet containing x^*. One way to overcome this zig-zagging problem is to keep tracking of the "active“ vertices (the vertices discovered previously in the FW algorithm) and move away from the “worst” of these in some iterations.

The Away-step Frank-Wolfe algorithm (AFW) and the Pairwise Frank-Wolfe algorithm (PFW) are two notable variants based on this idea. After computing the vertex
$p^{(k)} = \arg\min_{x\in\mathcal{P}}\langle\nabla F(x^{(k)}), x\rangle$ by the linear optimization oracle and the vertex
$u^{(k)} = \arg\max_{x\in U^{(k)}}\langle\nabla F(x^{(k)}), x\rangle$, where $U^{(k)}$ is the set of active vertices at iteration k, the AFW algorithm moves away from the one that maximizes the potential increase in $F(x)$; i.e. the increase in the linearized function, while the PFW algorithm tries to take advantages of both vertices and moves in the direction $p^{(k)} - u^{(k)}$. Details of the algorithms can be found in [5].

3.5.1 Variants of Stochastic Frank-Wolfe Algorithm

When the exact gradients are expensive to compute and an unbiased stochastic gradient is easy to obtain, it may be advantageous to use stochastic gradients in AFW and PFW. We describe the Away-step Stochastic Frank-Wolfe Algorithm (ASFW) and the Pairwise Stochastic Frank-Wolfe Algorithm(PSFW) below.

Algorithm 3 Away-step Stochastic Frank-Wolfe algorithm

1: **Input:** $x^{(1)} \in V$, f_i and L_i

2: Set $\mu^{(1)}_{x^{(1)}} = 1$, $\mu^{(1)}_v = 0$, for all $v \in V/\{x^{(1)}\}$ and $U^{(1)} = \{x^{(1)}\}$.

3: **for** $k = 1, 2, \ldots$ **do**

4: Sample $\xi_1, \ldots, \xi_{m^{(k)}} \overset{\text{i.i.d.}}{\sim} \xi$ and set $g^{(k)} = \frac{1}{m^{(k)}} \sum_{i=1}^{m^{(k)}} \nabla_x f(\xi_i, x^{(k)})$, $L^{(k)} = \frac{1}{m^{(k)}} \sum_{i=1}^{m^{(k)}} L_{\xi_i}$.

5: Compute $p^{(k)} \in \arg\min_{x \in \mathcal{P}} \langle g^{(k)}, x \rangle$.

6: Compute $u^{(k)} \in \operatorname{argmax}_{v \in U^{(k)}} \langle g^{(k)}, v \rangle$.

7: **if** $\langle g^{(k)}, p^{(k)} + u^{(k)} - 2x^{(k)} \rangle \leq 0$, **then**

8: Set $d^{(k)} = p^{(k)} - x^{(k)}$ and $\gamma^{(k)}_{\max} = 1$.

9: **else**

10: Set $d^{(k)} = x^{(k)} - u^{(k)}$ and $\gamma^{(k)}_{\max} = \frac{\mu^{(k)}_{u^{(k)}}}{1 - \mu^{(k)}_{u^{(k)}}}$.

11: **end if**

12: Set $\gamma^{(k)} = \min\{-\frac{\langle g^{(k)}, d^{(k)} \rangle}{L^{(k)} \|d^{(k)}\|^2}, \gamma^{(k)}_{\max}\}$ or determine it by line-search.

13: Set $x^{(k+1)} = x^{(k)} + \gamma^{(k)} d^{(k)}$.

14: Update $U^{(k+1)}$ and $\mu^{(k+1)}$ by VRU Procedure.

15: **end for**

16: **Return:** $x^{(k+1)}$.

Algorithm 4 Pairwise Stochastic Frank-Wolfe algorithm

1: Replace line 7 to 11 in Algorithm 3 by: $d^{(k)} = p^{(k)} - u^{(k)}$ and $\gamma^{(k)}_{\max} = \mu^{(k)}_{u^{(k)}}$.

The following algorithm updates a vertex representation of the current iterate and is called in Algorithms 3 and 4.

Algorithm 5 Procedure Vertex Representation Update (VRU)

Input: $x^{(k)}$, $(U^{(k)}, \mu^{(k)})$, $d^{(k)}$, $\gamma^{(k)}$, $p^{(k)}$ and $v^{(k)}$.

if $d^{(k)} = x^{(k)} - u^{(k)}$ **then**

  Update $\mu_v^{(k)} = \mu_v^{(k)}(1 + \gamma^{(k)})$, for all $v \in U^{(k)}/\{u^{(k)}\}$.

  Update $\mu_{u^{(k)}}^{(k+1)} = \mu_{u^{(k)}}^{(k)}(1 + \gamma^{(k)}) - \gamma^{(k)}$.

  if $\mu_{u^{(k)}}^{(k+1)} = 0$ **then**

    Update $U^{(k+1)} = U^{(k)}/\{u^{(k)}\}$

  else

    Update $U^{(k+1)} = U^{(k)}$

  end if

end if

Update $\mu_v^{(k+1)} = \mu_v^{(k)}(1 - \gamma^{(k)})$, for any $v \in U^{(k)}/\{p^{(k)}\}$.

Update $\mu_{p^{(k)}}^{(k+1)} = \mu_{p^{(k)}}^{(k)}(1 - \gamma^{(k)}) + \gamma^{(k)}$.

if $\mu_{p^{(k)}}^{(k+1)} = 1$ **then**

  Update $U^{(k+1)} = \{p^{(k)}\}$.

else

  Update $U^{(k+1)} = U^{(k)} \cup \{p^{(k)}\}$.

end if

(Optional) Carathéodory's Theorem can be applied for the vertex representation of $x^{(k+1)}$ so that $|U^{(k+1)}| = p + 1$ and $\mu^{(k+1)} \in \mathbb{R}^{p+1}$.

Return: $(U^{(k+1)}, \mu^{(k+1)})$

3.6 Convergence Proof

In this section, we first introduce some lemmas and notation and then prove the main theorems in this chapter. Note that, at the k-th iteration of Algorithms 3 and 4, $m^{(k)}$ i.i.d. samples of ξ are obtained. Define $F^{(k)}(x) = \frac{1}{m^{(k)}} \sum_{i=1}^{m^{(k)}} f_{\xi_i}(x)$. Clearly, $F^{(k)}$ is Lipschitz continuous with Lipschitz constant $L^{(k)} = \frac{1}{m^{(k)}} \sum_{i=1}^{m^{(k)}} L_{\xi_i}$ and strongly convex with constant $\sigma^{(k)} = \frac{1}{m^{(k)}} \sum_{i=1}^{m^{(k)}} \sigma_{\xi_i}$.

The following ancillary problem is used in our analysis.

$$\min_{x\in\mathcal{P}} F^{(k)}(x), \tag{H1}$$

Let $x_*^{(k)}$ denote the optimal solution of problem (H1), i.e., $x_*^{(k)} = \text{argmin}_{x\in\mathcal{P}} F^{(k)}(x)$. The lemma below plays an important role in our proof. We refer to [27] for a detailed proof of this lemma.

Lemma 3.6.1. *For any $x \in \mathcal{P}/\{x_*^{(k)}\}$ that can be represented as $x = \sum_{v\in U^{(k)}} \mu_v v$ for some $U^{(k)} \subset V$, where $\sum_{v\in U^{(k)}} \mu_v = 1$ and $\mu_v > 0$ for every $v \in U^{(k)}$, it holds that,*

$$\max_{u\in U, p\in V} \langle \nabla F^{(k)}(x), u - p\rangle \geq \frac{\Omega_{\mathcal{P}}}{|U|} \frac{\langle \nabla F^{(k)}(x), x - x_*^{(k)}\rangle}{\|x - x_*^{(k)}\|},$$

where $|U^{(k)}|$ denotes the cardinality of $U^{(k)}$, V is the set of extreme points of $\mathcal{P}$ and

$$\Omega_{\mathcal{P}} = \frac{\zeta}{\phi}$$

for

$$\zeta = \min_{v\in V, i\in\{1,\ldots,m\}: a_i > C_i v} (d_i - C_i v),$$
$$\phi = \max_{i\in\{1,\ldots,m\}/I(V)} \|C_i\|.$$

Lemma 3.6.2. *Let $c_i \geq 0$ and $b_i \in \{0, 1\}$ for $i = 1, \ldots, n$. Assume that $\sum_{j=1}^n b_j = m < n$. Then for $0 < a < 1$, we have*

$$\sum_{k=1}^{n} a^{\sum_{j=k}^n b_j} c_k \leq \sum_{k=1}^{m} a^{m-k+1} c_k + \sum_{k=m+1}^{n} c_k. \tag{3.1}$$

Proof. The right hand side of (3.1) is obtained by setting $b_i = 1$ for $i \leq m$ and $b_i = 0$ for $i > m$. We will show that this choice of $\{b_i\}$ maximizes $\sum_{k=1}^n a^{\sum_{j=k}^n b_j} c_k$. Consider an assignment of b_i such that there is a $b_r = 0$ for $r \leq m$ and $b_s = 1$ for $s > m$. Define a new assignment b_i' such that there

is $b'_i = b_i$ for $i \neq r, s$, $b'_r = 1$ and $b'_s = 0$. Then

$$\begin{aligned}
\sum_{k=1}^{n} a^{\sum_{j=k}^{n} b_j} c_k &= \sum_{k=s+1}^{n} a^{\sum_{j=k}^{n} b_j} c_k + \sum_{k=r}^{s} a^{\sum_{j=k}^{n} b_j} c_k + \sum_{k=1}^{r-1} a^{\sum_{j=k}^{n} b_j} c_k \\
&= \sum_{k=s+1}^{n} a^{\sum_{j=k}^{n} b'_j} c_k + \sum_{k=r+1}^{s} a^{\sum_{j=k}^{n} b_j} c_k + \sum_{k=1}^{r} a^{\sum_{j=k}^{n} b'_j} c_k \\
&= \sum_{k=s+1}^{n} a^{\sum_{j=k}^{n} b'_j} c_k + a \sum_{k=r+1}^{s} a^{\sum_{j=k}^{n} b'_j} c_k + \sum_{k=1}^{r} a^{\sum_{j=k}^{n} b'_j} c_k \\
&\leq \sum_{k=s+1}^{n} a^{\sum_{j=k}^{n} b'_j} c_k + \sum_{k=r+1}^{s} a^{\sum_{j=k}^{n} b'_j} c_k + \sum_{k=1}^{r} a^{\sum_{j=k}^{n} b'_j} c_k \\
&= \sum_{k=1}^{n} a^{\sum_{j=k}^{n} b'_j} c_k.
\end{aligned}$$

Therefore, such interchanges will always increase the value of $\sum_{k=1}^{n} a^{\sum_{j=k}^{n} b_j} c_k$ and hence, setting $b_i = 1$ for $i \leq m$ and $b_i = 0$ for $i > m$ maximizes it. □

Using the above lemmas we are ready to state and prove the main results.

Theorem 3.6.3. *Let $\{x^{(k)}\}_{k\geq 1}$ be the sequence generated by Algorithm 3 for solving Problem (P1), N be the number of vertices used to represent $x^{(k)}$ (if VRU is implemented by using Carathéodory's theorem, $N = p + 1$, otherwise $N = |V|$) and F^* be the optimal value of the problem. Let $\rho = \min\{\frac{1}{2}, \frac{\Omega_{\mathcal{P}}^2 \sigma_F}{16N^2 L_F D^2}\}$, where $\sigma_F = \min\{\sigma_1, \ldots, \sigma_n\}$, $L_F = \max\{L_1, \ldots, L_n\}$. Set $m^{(i)} = \lceil 1/(1-\rho)^{2i+2} \rceil$. Then for every $k \geq 1$,*

$$\mathbb{E}\{F(x^{(k+1)}) - F^*\} \leq C_2(1-\beta)^{(k-1)/2}, \tag{3.2}$$

where C_2 is a deterministic constant and $0 < \beta < \rho \leq 1/2$.

Proof. At iteration k, let $x^{(k)}$ denote the current solution, $\xi_1, \ldots, \xi_{m^{(k)}}$ denote the samples used by Algorithm 3, $d^{(k)}$ denote the direction that Algorithm 3 takes and $\gamma^{(k)}$ denote the step length. Define $F^{(k)}(x) = \frac{1}{m^{(k)}} \sum_{i=1}^{m^{(k)}} f(\xi_i, x)$, $x_*^{(k)} = \arg\min_{x\in\mathcal{P}} F^{(k)}(x)$ and $F_*^{(k)} = F^{(k)}(x_*^{(k)})$. Note that $F^{(k)}$ is Lipschitz continuous with Lipschitz constant $L^{(k)} = \frac{1}{m^{(k)}} \sum_{i=1}^{m^{(k)}} L_{\xi_i}$ and strongly convex

with constant $\sigma^{(k)} = \frac{1}{m^{(k)}} \sum_{i=1}^{m^{(k)}} \sigma_{\xi_i}$. In addition, the stochastic gradient $g^{(k)} = \nabla F^{(k)}(x)$. From the choice of $d^{(k)}$ in the algorithm,

$$\langle g^{(k)}, d^{(k)} \rangle \leq \frac{1}{2}(\langle g^{(k)}, p^{(k)} - x^{(k)} \rangle + \langle g^{(k)}, x^{(k)} - u^{(k)} \rangle) = \frac{1}{2}\langle g^{(k)}, p^{(k)} - u^{(k)} \rangle \leq 0.$$

Hence, we can bound $\langle g^{(k)}, d^{(k)} \rangle^2$ below by

$$\begin{aligned}
\langle g^{(k)}, d^{(k)} \rangle^2 &\geq \frac{1}{4}\langle g^{(k)}, u^{(k)} - p^{(k)} \rangle^2 \\
&\geq \frac{1}{4} \max_{p \in V, u \in U^{(k)}} \langle g^{(k)}, u - p \rangle^2 && \text{(definition of } p^{(k)} \text{ and } u^{(k)}) \\
&= \frac{1}{4} \max_{p \in V, u \in U^{(k)}} \langle \nabla F^{(k)}(x^{(k)}), u - p \rangle^2 && (g^{(k)} = \nabla F^{(k)}(x^{(k)})) \\
&\geq \frac{1}{4} \frac{\Omega_{\mathcal{P}}^2}{|U^{(k)}|^2} \frac{\langle \nabla F^{(k)}(x^{(k)}), x^{(k)} - x_*^{(k)} \rangle^2}{\|x^{(k)} - x_*^{(k)}\|^2} && \text{(by Lemma 3.6.1)} \\
&\geq \frac{\Omega_{\mathcal{P}}^2}{4N^2} \frac{\{F^{(k)}(x^{(k)}) - F_*^{(k)}\}^2}{\|x^{(k)} - x_*^{(k)}\|^2} && \text{(Convexity of } F^{(k)}(\cdot)) \\
&\geq \frac{\Omega_{\mathcal{P}}^2 \sigma^{(k)}}{8N^2} \{F^{(k)}(x^{(k)}) - F_*^{(k)}\} && \text{(by strong convexity of } F^{(k)}(\cdot)) \\
&\geq \frac{\Omega_{\mathcal{P}}^2 \sigma_F}{8N^2} \{F^{(k)}(x^{(k)}) - F_*^{(k)}\}.
\end{aligned}$$

Similarly, we can bound $\langle g^{(k)}, d^{(k)} \rangle$ above by

$$\begin{aligned}
\langle g^{(k)}, d^{(k)} \rangle &\leq \frac{1}{2}\langle g^{(k)}, p^{(k)} - u^{(k)} \rangle \\
&\leq \frac{1}{2}\langle g^{(k)}, x_*^{(k)} - x^{(k)} \rangle && \text{(definition of } p^{(k)} \text{ and } u^{(k)}) \\
&= \frac{1}{2}\langle \nabla F^{(k)}(x^{(k)}), x_*^{(k)} - x^{(k)} \rangle && (g^{(k)} = \nabla F^{(k)}(x^{(k)})) \\
&\leq \frac{1}{2}\{F_*^{(k)} - F^{(k)}(x^{(k)})\}. && \text{(Convexity of } F(\cdot))
\end{aligned}$$

With the above bounds, we can separate our analysis into the following four cases at iteration k

$(A^{(k)})$ $\quad \gamma_{\max}^{(k)} \geq 1$ and $\gamma^{(k)} \leq 1$.

$(B^{(k)})$ $\quad \gamma_{\max}^{(k)} \geq 1$ and $\gamma^{(k)} \geq 1$.

$(C^{(k)})$ $\quad \gamma_{\max}^{(k)} < 1$ and $\gamma^{(k)} < \gamma_{\max}^{(k)}$.

$(D^{(k)})$ $\quad \gamma_{\max}^{(k)} < 1$ and $\gamma^{(k)} = \gamma_{\max}^{(k)}$.

By the descent lemma, we have

$$\begin{aligned} F^{(k)}(x^{(k+1)}) &= F^{(k)}(x^{(k)} + \gamma^{(k)} d^{(k)}) \\ &\leq F^{(k)}(x^{(k)}) + \gamma^{(k)} \langle \nabla F^{(k)}(x^{(k)}), d^{(k)} \rangle + \frac{L^{(k)}(\gamma^{(k)})^2}{2} \|d^{(k)}\|^2 \\ &= F^{(k)}(x^{(k)}) + \gamma^{(k)} \langle g^{(k)}, d^{(k)} \rangle + \frac{L^{(k)}(\gamma^{(k)})^2}{2} \|d^{(k)}\|^2. \end{aligned} \tag{3.3}$$

In case $(A^{(k)})$, let $\delta_{A^{(k)}}$ denote the indicator function for this case. Then

$$\begin{aligned} &\delta_{A^{(k)}} \{F^{(k)}(x^{(k+1)}) - F_*^{(k)}\} \\ &\leq \delta_{A^{(k)}} \{F^{(k)}(x^{(k)}) - F_*^{(k)} + \gamma^{(k)} \langle g^{(k)}, d^{(k)} \rangle + \frac{L^{(k)}(\gamma^{(k)})^2}{2} \|d^{(k)}\|^2\} \\ &= \delta_{A^{(k)}} \{F^{(k)}(x^{(k)}) - F_*^{(k)} - \frac{\langle g^{(k)}, d^{(k)} \rangle^2}{2L^{(k)} \|d^{(k)}\|^2}\} \qquad \text{(definition of } \gamma^{(k)} \text{ in case } A^{(k)}) \\ &\leq \delta_{A^{(k)}} \{(1 - \frac{\Omega_{\mathcal{P}}^2 \sigma_F}{16N^2 L^{(k)} D^2})(F^{(k)}(x^{(k)}) - F_*^{(k)})\} \\ &\leq \delta_{A^{(k)}} \{(1 - \frac{\Omega_{\mathcal{P}}^2 \sigma_F}{16N^2 L_F D^2})(F^{(k)}(x^{(k)}) - F_*^{(k)})\} \end{aligned}$$

In case $(B^{(k)})$, since $\gamma^{(k)} > 1$, we have

$$-\langle g^{(k)}, d^{(k)} \rangle > L^{(k)} \|d^{(k)}\|^2 \qquad \text{and} \tag{3.4}$$

$$\gamma^{(k)} \langle g^{(k)}, d^{(k)} \rangle + \frac{L^{(k)}(\gamma^{(k)})^2}{2} \|d^{(k)}\|^2 \leq \langle g^{(k)}, d^{(k)} \rangle + \frac{L^{(k)}}{2} \|d^{(k)}\|^2. \tag{3.5}$$

Use $\delta_{B^{(k)}}$ to denote the indicator function for this case. Then,

$$
\begin{aligned}
&\delta_{B^{(k)}}\{F^{(k)}(x^{(k+1)}) - F_*^{(k)}\} \\
&\leq \delta_{B^{(k)}}\{F^{(k)}(x^{(k)}) - F_*^{(k)} + \gamma^{(k)}\langle \nabla F^{(k)}(x^{(k)}), d^{(k)}\rangle + \frac{L^{(k)}(\gamma^{(k)})^2}{2}\|d^{(k)}\|^2\} \\
&= \delta_{B^{(k)}}\{F^{(k)}(x^{(k)}) - F_*^{(k)} + \gamma^{(k)}\langle g^{(k)}, d^{(k)}\rangle + \frac{L^{(k)}(\gamma^{(k)})^2}{2}\|d^{(k)}\|^2 \\
&\leq \delta_{B^{(k)}}\{F^{(k)}(x^{(k)}) - F_*^{(k)} + \langle g^{(k)}, d^{(k)}\rangle + \frac{L^{(k)}}{2}\|d^{(k)}\|^2\} \qquad \text{(by (3.5))} \\
&\leq \delta_{B^{(k)}}\{F^{(k)}(x^{(k)}) - F_*^{(k)} + \frac{1}{2}\langle g^{(k)}, d^{(k)}\rangle\} \qquad \text{(by (3.4))} \\
&\leq \delta_{B^{(k)}}\{\frac{1}{2}(F^{(k)}(x^{(k)}) - F_*^{(k)})\}
\end{aligned}
$$

In case $(C^{(k)})$, let $\delta_{C^{(k)}}$ be the indicator function for this case. Using exactly the same argument as in case $(A^{(k)})$, we obtain the following inequality

$$
\begin{aligned}
\delta_{C^{(k)}}\{F^{(k)}(x^{(k+1)}) - F_*^{(k)}\} &\leq \delta_{C^{(k)}}\{F^{(k)}(x^{(k)}) - F_*^{(k)} - \frac{\langle g^{(k)}, d^{(k)}\rangle^2}{2L^{(k)}\|d^{(k)}\|^2}\} \\
&\leq \delta_{C^{(k)}}\{(1 - \frac{\Omega_{\mathcal{P}}^2\sigma_F}{16N^2L_FD^2})(F^{(k)}(x^{(k)}) - F_*^{(k)})\}
\end{aligned}
$$

Case $(D^{(k)})$ is the so called “drop step" in the conditional gradient algorithm with away-steps. Use $\delta_{D^{(k)}}$ to denote the indicator function for this case. Note that $\gamma^{(k)} = \gamma_{\max}^{(k)} \leq -\langle g^{(k)}, d^{(k)}\rangle/(L^{(k)}\|d^{(k)}\|^2)$ in this case. Hence, we have

$$
\begin{aligned}
&\delta_{D^{(k)}}\{(F^{(k)}(x^{(k+1)}) - F_*^{(k)})\} \\
&\leq \delta_{D^{(k)}}\{F^{(k)}(x^{(k)}) - F_*^{(k)} + \gamma^{(k)}\langle \nabla F^{(k)}(x^{(k)}), d^{(k)}\rangle + \frac{L^{(k)}(\gamma^{(k)})^2}{2}\|d^{(k)}\|^2\} \\
&= \delta_{D^{(k)}}\{F^{(k)}(x^{(k)}) - F_*^{(k)} + \gamma^{(k)}\langle g^{(k)}, d^{(k)}\rangle + \frac{L^{(k)}(\gamma^{(k)})^2}{2}\|d^{(k)}\|^2\} \\
&\leq \delta_{D^{(k)}}\{F^{(k)}(x^{(k)}) - F_*^{(k)} + \frac{\gamma^{(k)}}{2}\langle g^{(k)}, d^{(k)}\rangle\} \\
&\leq \delta_{D^{(k)}}\{F^{(k)}(x^{(k)}) - F_*^{(k)}\}.
\end{aligned}
$$

Define $\rho = \min\{\frac{1}{2}, \frac{\Omega_{\mathcal{P}}^2 \sigma_F}{16N^2 L_F D^2}\}$. Note that ρ is a deterministic constant between 0 and 1. Therefore we have

$$
\begin{aligned}
& F^{(k)}(x^{(k+1)}) - F_*^{(k)} \\
\leq\ & (\{1-\rho)^{\{1-\delta_{D^{(k)}}\}}(F^{(k)}(x^{(k)}) - F_*^{(k)}) \\
=\ & (1-\rho)^{\{1-\delta_{D^{(k)}}\}}(F^{(k-1)}(x^{(k)}) - F_*^{(k-1)}) \\
& + (1-\rho)^{\{1-\delta_{D^{(k)}}\}}\{F^{(k)}(x^{(k)}) - F_*^{(k)} - F^{(k-1)}(x^{(k)}) + F_*^{(k-1)}\} \\
=\ & (1-\rho)^{\{1-\delta_{D^{(k)}}\}}(F^{(k-1)}(x^{(k)}) - F_*^{(k-1)}) \\
& + (1-\rho)^{\{1-\delta_{D^{(k)}}\}}\{F^{(k)}(x^{(k)}) - F(x^{(k)}) + F(x^{(k)}) - F^{(k-1)}(x^{(k)}) + F^* - F_*^{(k)} + F_*^{(k-1)} - F^*\} \\
\leq\ & (1-\rho)^{\{1-\delta_{D^{(k)}}\}}(F^{(k-1)}(x^{(k)}) - F_*^{(k-1)}) \\
& + (1-\rho)^{\{1-\delta_{D^{(k)}}\}}\{|F^{(k)}(x^{(k)}) - F(x^{(k)})| + |F^{(k-1)}(x^{(k)}) - F(x^{(k)})| + |F_*^{(k)} - F^*| \\
& + |F_*^{(k-1)} - F^*|\} \\
\leq\ & (1-\rho)^{\sum_{i=1}^{k}\{1-\delta_{D^{(i)}}\}}(F^{(0)}(x^{(1)}) - F_*^{(0)}) + \\
& \sum_{i=1}^{k}(1-\rho)^{\sum_{j=i}^{k}\{1-\delta_{D^{(j)}}\}}\{|F^{(i)}(x^{(i)}) - F(x^{(i)})| + |F^{(i-1)}(x^{(i)}) - F(x^{(i)})| + |F_*^{(i)} - F^*| \\
& + |F_*^{(i-1)} - F^*|\}.
\end{aligned}
$$

At iteration k, there are at most $(k+1)/2$ drop steps, i.e., at most $(k+1)/2$ $\delta_{D^{(i)}}$'s equal to 1. Then, letting $a = 1-\rho$, $b_i = 1 - \delta_{D^{(i)}}$, and $c_i = \{|F^{(i)}(x^{(i)}) - F(x^{(i)})| + |F^{(i-1)}(x^{(i)}) - F(x^{(i)})| + |F_*^{(i)} -$

$F^*| + |F_*^{(i-1)} - F^*|\}$, it follows from Lemma 3.6.2 that

$$
\begin{aligned}
&\sum_{i=1}^{k}(1-\rho)^{\sum_{j=i}^{k}\{1-\delta_{D^{(j)}}\}}\{|F^{(i)}(x^{(i)}) - F(x^{(i)})| + |F^{(i-1)}(x^{(i)}) - F(x^{(i)})| + |F_*^{(i)} - F^*| \\
&+ |F_*^{(i-1)} - F^*|\} \\
\leq &\sum_{i=k/2}^{k}\{|F^{(i)}(x^{(i)}) - F(x^{(i)})| + |F^{(i-1)}(x^{(i)}) - F(x^{(i)})| + |F_*^{(i)} - F^*| + |F_*^{(i-1)} - F^*|\} \\
+ &\sum_{i=1}^{k/2-1}(1-\rho)^{k/2-i}\{|F^{(i)}(x^{(i)}) - F(x^{(i)})| + |F^{(i-1)}(x^{(i)}) - F(x^{(i)})| + |F_*^{(i)} - F^*| + |F_*^{(i-1)} - F^*|\}.
\end{aligned}
$$

Therefore

$$
\begin{aligned}
&F^{(k)}(x^{(k+1)}) - F_*^{(k)} \\
\leq &(1-\rho)^{\frac{k-1}{2}}(u_F - l_F) \\
+ &\sum_{i=k/2}^{k}\{|F^{(i)}(x^{(i)}) - F(x^{(i)})| + |F^{(i-1)}(x^{(i)}) - F(x^{(i)})| + |F_*^{(i)} - F^*| + |F_*^{(i-1)} - F^*|\} \\
+ &\sum_{i=1}^{k/2-1}(1-\rho)^{k/2-i}\{|F^{(i)}(x^{(i)}) - F(x^{(i)})| + |F^{(i-1)}(x^{(i)}) - F(x^{(i)})| + |F_*^{(i)} - F^*| + |F_*^{(i-1)} - F^*|\}.
\end{aligned}
$$

In addition, $F^{(k)}(x^{(k+1)}) - F_*^{(k)} = F(x^{(k+1)}) - F^* + (F^{(k)}(x^{(k+1)}) - F(x^{(k+1)})) + (F^* - F_*^{(k)})$. Thus

$$
\begin{aligned}
&F(x^{(k+1)}) - F^* \\
\leq &(1-\rho)^{\frac{k-1}{2}}(u_F - l_F) \\
+ &\sum_{i=k/2}^{k+1}\{|F^{(i)}(x^{(i)}) - F(x^{(i)})| + |F^{(i-1)}(x^{(i)}) - F(x^{(i)})| + |F_*^{(i)} - F^*| + |F_*^{(i-1)} - F^*|\} \\
+ &\sum_{i=1}^{k/2-1}(1-\rho)^{k/2-i}\{|F^{(i)}(x^{(i)}) - F(x^{(i)})| + |F^{(i-1)}(x^{(i)}) - F(x^{(i)})| + |F_*^{(i)} - F^*| + |F_*^{(i-1)} - F^*|\}.
\end{aligned}
$$

Note that for any deterministic $x \in \mathcal{P}$, we have $\mathbb{E}F^{(k)}(x) = F(x)$. In addition, by Theorem 2.1.1,

the following bound holds for every iteration k

$$\mathbb{E}|F^{(k)}(x^{(k)}) - F(x^{(k)})| \leq \mathbb{E}\sup_{x\in\mathcal{P}}|F^{(k)}(x) - F(x)| \leq C_1\sqrt{\frac{\log m^{(k)}}{m^{(k)}}}$$

and

$$\mathbb{E}|F_*^{(k)} - F^*| \leq C_1\sqrt{\frac{\log m^{(k)}}{m^{(k)}}}.$$

Combining all above bounds and using $m^{(i)} = \lceil 1/(1-\rho)^{2i+2}\rceil$, we have

$$\begin{aligned}
&\mathbb{E}\{F(x^{(k+1)}) - F^*\} \\
&\leq (1-\rho)^{\frac{k-1}{2}}(u_F - l_F) \\
&\quad + 2C_1\{\sum_{i=k/2}^{k+1}(\sqrt{\frac{\log m^{(i)}}{m^{(i)}}} + \sqrt{\frac{\log m^{(i-1)}}{m^{(i-1)}}}) + \sum_{i=1}^{k/2-1}(1-\rho)^{k/2-i}(\sqrt{\frac{\log m^{(i)}}{m^{(i)}}} + \sqrt{\frac{\log m^{(i-1)}}{m^{(i-1)}}})\} \\
&\leq (1-\rho)^{\frac{k-1}{2}}(u_F - l_F) + 4C_1\{\sum_{i=k/2}^{k+1}\sqrt{\frac{\log m^{(i-1)}}{m^{(i-1)}}} + \sum_{i=1}^{k/2-1}(1-\rho)^{k/2-i}\sqrt{\frac{\log m^{(i-1)}}{m^{(i-1)}}}\} \\
&\qquad\qquad (\tfrac{\log x}{x} \text{ decreases for } x > e) \\
&\leq (1-\rho)^{\frac{k-1}{2}}(u_F - l_F) + 4C_1\sqrt{2\log\frac{1}{1-\rho}}\{\sum_{i=k/2}^{k+1}(1-\rho)^i\sqrt{i} + \sum_{i=1}^{k/2-1}(1-\rho)^{k/2}\sqrt{i}\} \\
&\leq C_2(1-\beta)^{\frac{k-1}{2}}
\end{aligned}$$

for some constant C_2 and $0 < \beta < \rho < 1$. □

Remark: The proof of Theorem 3.6.3 does not use any stochastic arguments until the very end and uses Lemma 3.1 to get rid of the indicator function for the 'drop-steps' so that the stochastic arguments based on concentration inequalities can be applied. Note that we cannot take expectation on the stochastic gradients and utilize their unbiasedness property because of the presence of the indicator functions. This proof technique is specifically designed for the 'drop-step' in ASFW and can be useful in analyzing other similar algorithms.

Corollary 3.6.4. *Let* $\{x^{(k)}\}_{k\geq 1}$ *be the sequence generated by Algorithm 3 for solving Problem* $(P1)$. *Then*

$$\frac{F(x^{(k)}) - F^*}{(1-\omega)^{\frac{k-1}{2}}} \to 0$$

almost surely as k *tends to infinity for any* $0 < \omega < \beta$. *Therefore* $F(x^{(k)})$ *linearly converges to* F^* *almost surely.*

Proof. For every $\epsilon > 0$, let $E^{(k)}$ denotes the event that $(F(x^{(k)}) - F^*)/(1-\omega)^{(k-1)/2} > \epsilon$. By Markov's inequality

$$\begin{aligned}
\sum_{k=2}^{\infty} \mathbb{P}(E^{(k)}) &= \sum_{k=1}^{\infty} \mathbb{P}((F(x^{(k)}) - F^*)/(1-\omega)^{(k-1)/2} > \epsilon) \\
&\leq \sum_{k=2}^{\infty} \frac{\mathbb{E}\{F(x^{(k)}) - F^*\}}{\epsilon(1-\omega)^{(k-1)/2}} \\
&\leq \frac{C_2}{\epsilon} \sum_{k=2}^{\infty} (\frac{1-\beta}{1-\omega})^{\frac{k-1}{2}} \\
&< \infty.
\end{aligned}$$

Therefore $\sum_{k=2}^{\infty} \mathbb{P}(E^{(k)}) < \infty$ and the Borel-Cantelli lemma implies that $\mathbb{P}(\limsup_{k\to\inf} E^{(k)}) = 0$, which implies $(F(x^{(k)}) - F^*)/(1-\omega)^{(k-1)/2}$ converges to 0 almost surely. This implies that every sequence generated by Algorithm 3 linearly converges to the optimal function value almost surely. □

Remark: Note that the result in Corollary 3.6.4 only relies on the property that an algorithm converges linearly in expectation. Therefore, we can apply exactly the same argument to show that every sequence generated by the algorithm in [6] converges linearly almost surely.

Corollary 3.6.5. *To obtain an* ϵ*-accurate solution, Algorithm 3 requires* $O((1/\epsilon)^{4\eta})$ *of stochastic gradient evaluations, where* $0 < \eta = \log(1-\rho)/\log(1-\beta) < 1$.

Proof. Let k be the total number of iterations performed by Algorithm 3 so that an ϵ-accurate solution is obtained for the first time. Theorem 3.6.3 implies $C_2(1-\beta)^{\frac{k-1}{2}} < \epsilon$ and hence $k \geq$

$1 + 2\log\epsilon/\log(1-\beta)$. In iteration i of Algorithm 3, $m^{(i)} = 1/(1-\rho)^{2i+2}$ stochastic gradient evaluations are performed. Thus, the total number of stochastic gradient evaluations until iteration k is

$$\begin{aligned}
\sum_{i=1}^{k} m^{(i)} &= \sum_{i=1}^{k} \frac{1}{(1-\rho)^{(2i+2)}} \\
&= \frac{1}{(1-\rho)^2} \frac{1/(1-\rho)^2 - 1/(1-\rho)^{2k+2}}{1 - 1/(1-\rho)^2} \\
&\leq \frac{2}{(1-\rho)^{2k+4}} \leq \frac{2}{(1-\rho)^4} \exp\{-2k\log(1-\rho)\} \\
&\leq \frac{2}{(1-\rho)^4} \exp\{-2\log(1-\rho) - 4\frac{\log\epsilon\log(1-\rho)}{\log(1-\beta)}\} \\
&= O((\frac{1}{\epsilon})^{\frac{4\log(1-\rho)}{\log(1-\beta)}}) \\
&= O((\frac{1}{\epsilon})^{4\eta}).
\end{aligned}$$

□

Theorem 3.6.6. *Let $\{x^{(k)}\}_{k\geq 1}$ be the sequence generated by Algorithm 4 for solving Problem (P1), N be the number of vertices used to represent $x^{(k)}$ (if VRU is implemented by using Carathéodory's theorem, $N = p+1$, otherwise $N = |V|$) and F^* be the optimal value of the problem. Let $\kappa = \min\{\frac{1}{2}, \frac{\Omega_{\mathcal{P}}^2\sigma_F}{8N^2L_FD^2}\}$ where $\sigma_F = \min\{\sigma_1,\ldots,\sigma_n\}$, $L_F = \max\{L_1,\ldots,L_n\}$. Set $m^{(i)} = \lceil 1/(1-\kappa)^{2i+2}\rceil$. Then for every $k \geq 1$*

$$\mathbb{E}\{F(x^{(k+1)}) - F^*\} \leq C_3(1-\phi)^{k/(3|V|!+1)} \tag{3.6}$$

where C_3 is a deterministic constant and $0 < \phi < \kappa \leq 1/2$.

Proof. Since $d^{(k)} = p^{(k)} - u^{(k)}$, similar to the proof of Theorem 3.6.3, we have

$$\begin{aligned}
\langle g^{(k)}, d^{(k)}\rangle^2 &\geq \frac{\Omega_{\mathcal{P}}^2\sigma_F}{4N^2}\{F^{(k)}(x^{(k)}) - F_*^{(k)}\} \\
\langle g^{(k)}, d^{(k)}\rangle &\leq \frac{1}{2}(F_*^{(k)} - F^{(k)}(x^{(k)})).
\end{aligned}$$

The remaining proof for Theorem 3.6.3 could also apply here except that the case $D^{(k)}$ can be either a 'drop step' or a so-called 'swap step'. A swap step moves the weight of a active vertex to another active vertex. There are at most $(1 - \frac{1}{3|V|!+1})k$ drop steps and swap steps after k iteration. The same argument as in Theorem 3.6.3 implies

$$\mathbb{E}\{F(x^{(k+1)}) - F^*\} \leq C_3(1-\phi)^{k/(3|V|!+1)}$$

for a deterministic constant C_3 and $0 < \phi < \kappa \leq 1/2$. □

Corollary 3.6.7. *Let $\{x^{(k)}\}_{k\geq 1}$ be the sequence generated by Algorithm 4 for solving Problem $(P1)$. Then*

$$\frac{F(x^{(k)}) - F^*}{(1-\psi)^{\frac{k}{3|V|!+1}}} \to 0$$

almost surely as k tends to infinity for some $0 < \psi < \phi$. Therefore $F(\mathbf{x}^{(k)})$ linearly converges to F^* almost surely.

Proof of this Corollary is almost the same as the proof of Corollary 3.6.4.

Corollary 3.6.8. *To obtain an ϵ-accurate solution, Algorithm 4 requires $O((1/\epsilon)^{(6|V|!+2)\xi})$ of stochastic gradient evaluations, where $0 < \zeta = \log(1-\rho)/\log(1-\phi) < 1$.*

Proof of this Corollary is the same as the proof of Corollary 3.6.5.

3.7 Numerical Experiments

3.7.1 Simulated Data

We apply the proposed algorithms to the synthetic problem:

$$\begin{aligned} &\text{minimize} \quad \|Ax - b\|_2^2 + \frac{1}{2}\|x\|_2^2 \\ &\text{such that} \quad l \leq x_1 \leq x_2 \leq \cdots \leq x_p \leq u, \end{aligned}$$

where $A \in \mathbb{R}^{n\times p}$, $b \in \mathbb{R}^n$ and $x \in \mathbb{R}^p$. We generated the entries of A and b from the standard normal distribution and set $n = 10^6$, $p = 1000$, $l = -1$ and $u = 1$. This problem can be viewed as minimizing a sum of strongly convex functions subject to a polytope constraint. Such problems can be found in the shape restricted regression literature. We compared the ASFW and PSFW with two variance-reduced stochastic methods, the variance-reduced stochastic Frank-Wolfe (SVRF) method [30] and the proximal variance-reduced stochastic gradient (Prox-SVRG) method [6, 35]. Both Prox-SVRG and SVRF are epoch based algorithms. They first fix a reference point and compute the exact gradient at the reference point at the beginning of each epoch. Within each epoch, both algorithms compute variance reduced gradients in every step using the control variates technique based on the reference point. The major difference between them is that in every iteration, the Prox-SVRG takes a proximal gradient step and the SVRF takes a Frank-Wolfe step. For detailed implementations of SVRF, we followed Algorithm 1 in [30] and chose the parameters according to Theorem 1 in [30]. For the Prox-SVRG, we followed the Algorithm in [35] and set the number of iterations in each epoch to be $m = 2n$ and set the step size to be $\gamma = 0.1/L$ found by [35] to give the best results for Prox-SVRG, where n is the sample size and L is the Lipschitz constant of the gradient of the objective function. For ASFW and PSFW implementations, we followed Algorithm 3 and Algorithm 4 and used adaptive step sizes since we know the Lipschitz constants of the gradients of the objective functions. The number of samples that we used to compute stochastic gradients for ASFW and PSFW was set to be $1.04^k + 100$ at the iteration k. The linear optimization sub-problems in the Frank-Wolfe algorithms and the projection step in Prox-SVRG were solved by using the GUROBI solver. We summarize the parameters that were used in the algorithms at iteration k and epoch t in Table 3.2. In this table, $g^{(k)}$ is the stochastic gradient, $L^{(k)}$ is the Lipschitz constant of the stochastic gradient at iteration k, $d^{(k)}$ is the direction the algorithms take at iteration k and $\gamma_{\max}$ is the maximum of the possible step sizes (see Algorithm 3 and 4). In Prox-SVRG, L is the Lipschitz constant of the gradient of the objection function and n is the sample size.

To make fair comparisons, we used the same starting point for all four algorithms. The loss

	step-size	batch-size	#iterations
ASFW	$\min\{-\langle g^{(k)}, d^{(k)}\rangle/(L^{(k)}\|d^{(k)}\|^2), \gamma_{\max}\}$	$100 + 1.04^k$	N/A
PSFW	$\min\{-\langle g^{(k)}, d^{(k)}\rangle/(L^{(k)}\|d^{(k)}\|^2), \gamma_{\max}\}$	$100 + 1.04^k$	N/A
SVRF	$2/(k+1)$	$96(k+1)$	$2^{t+3} - 2$
SVRG	$0.1/L$	1	$2n$

Table 3.2: Parameter choices in the algorithms

function values obtained by using ASFW, PSFW and Prox-SVRG and the running minimum values obtained by SVRF are plotted against CPU time. From the plot, we can see that ASFW and PSFW performed as well as or slightly better than their stochastic competitors. At the very beginning, both Prox-SVRG and SVRF descents rapidly, while ASFW and PSFW obtains lower function values later on. We also observe big swings in SVRF periodically. This is because at the beginning of each epoch, SVRF proceeds with noisy gradients and very large step sizes. According to Theorem 1 in [30], the step size of the first step in every epoch can be as large as 1.

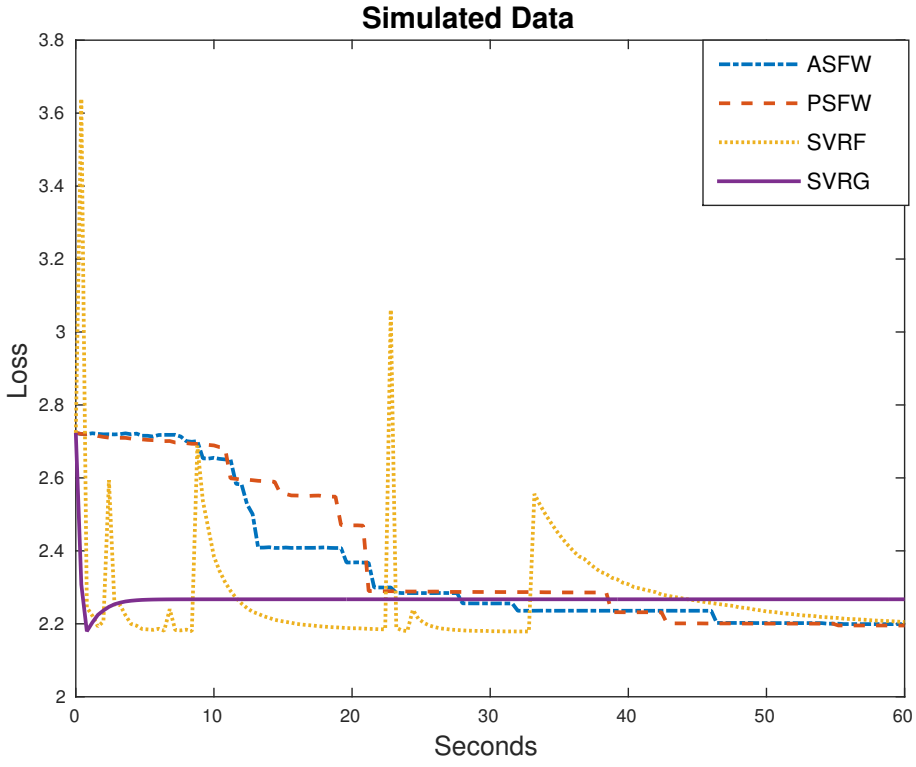


Figure 3.1: Comparison between algorithms on simulated data.

3.7.2 Million Song Dataset

We implemented ASFW and PSFW for solving least squares problems with elastic-net regularization and tested them on the Million Song Dataset (YearPredictionMSD) [36][37], which is a dataset of songs with the goal of predicting the release year of a song from its audio features. There are $n = 463,715$ training samples and $p = 90$ features in this dataset. The dataset is the one with largest number of training samples available in the UCI machine learning data repository. Therefore it is interesting to examine the actual performance of stochastic algorithms on such a massive dataset. The least squares with elastic-net regularization model that we used was,

$$\min_{x \in \mathbb{R}^p} \frac{1}{n} \|Ax - b\|_2^2 + \lambda \|x\|_1 + \mu \|x\|_2^2$$

where $A \in \mathbb{R}^{n \times p}$ and $b \in \mathbb{R}^n$. $\mu \geq 0$ and $\lambda \geq 0$ are regularization parameters. In the numerical experiments, we considered the constrained version of the problem, that is,

$$\begin{aligned} \text{minimize} \quad & \frac{1}{n} \|Ax - b\|_2^2 + \mu \|x\|_2^2 \\ \text{subject to} \quad & \|x\|_1 \leq \alpha \end{aligned}$$

where $\alpha > 0$ is inversely related to λ.

We also compared the ASFW and PSFW with SVRF and Prox-SVRG. We followed the same settings in this real data experiment as that in the simulated data experiment except that we used explicit solutions for solving linear optimizations over an l_1-balls in FW algorithms and we used the algorithm in [38] for the solving projections onto l_1-balls in the Prox-SVRG algorithm instead of using GUROBI for solving linear optimizations and projections. To make fair comparisons, we used the same starting point for all four algorithms. The logarithm of the loss function values obtained by ASFW, PSFW and Prox-SVRG and the running minimum value obtained by SVRF are plotted against CPU time. The figures indicate that the performance of ASFW and PFW was as good as or better than Prox-SVRG and SVRF under different regularization parameter settings.

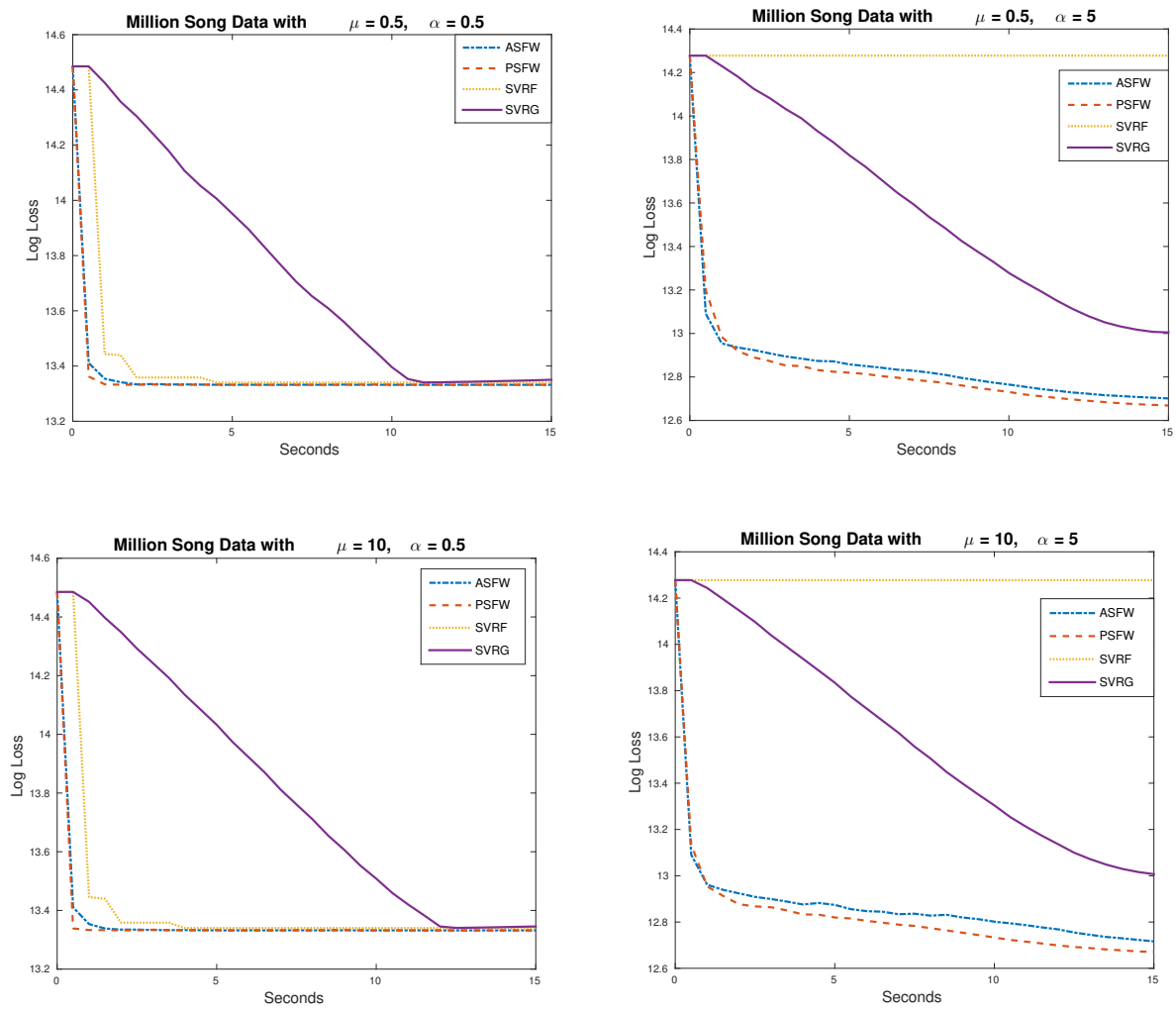


Figure 3.2: Comparisons between algorithms on million song dataset.

We also observed huge swings in SVRF periodically in these experiments. Therefore, we plot the running minimums instead of the most recent function values for SVRF.

3.8 Conclusion and Future Work

In this chapter, we proved linear convergence almost surely and in expectation of the Away-step Stochastic Frank-Wolfe algorithm and the Pairwise Stochastic Frank-Wolfe algorithm by using a novel proof technique. We tested these algorithms by training a least squares model with elastic-net regularization on the million song dataset and on a synthetic problem. The proposed algorithms performed as well as or better than their stochastic competitors for various choices of the

regularization parameters. Future work includes extending the proposed algorithms to problems with block-coordinate structures and non-strongly convex objective functions and using variance reduced stochastic gradients to reduce the number of stochastic gradient oracle calls.

Chapter 4: Local Curvature Based Adaptive Step-size Algorithms

4.1 Introduction

We are concerned with minimizing functions of the form

$$\min_{x\in\mathbb{R}^n} F(x) = \mathbb{E}_\xi f(x,\xi). \tag{4.1}$$

Many common problems in statistics and machine learning can be put into this form. For instance, in the empirical risk minimization framework, a model is learned from a set $\{y_1,\dots,y_m\}$ of training data by minimizing an empirical loss function of the form

$$\min_x L(x) = \frac{1}{m}\sum_{i=1}^{m} f(x,y_i). \tag{4.2}$$

It is easy to see that this formulation is equivalent to taking ξ to be the uniform distribution on the points $\{y_1,\dots,y_m\}$.

An objective function of the form (4.1) is often impractical, as the distribution of ξ is generally unavailable, making it infeasible to analytically compute $\mathbb{E}f(x,\xi)$. This can be resolved by replacing the expectation $\mathbb{E}f(x,\xi)$ by the estimate (4.2). The strong law of large numbers implies that the sample mean $L(x)$ converges almost surely to $F(x)$ as the number of samples m increases, provided that the samples y_i are drawn independently from the distribution of ξ. However, even the concrete problem (4.2) is not a good target for classical optimization algorithms, as the amount of data m is frequently extremely large. Thus, a better strategy when optimizing (4.2) is to consider *subsamples* of the data to reduce the computational cost. This leads to *stochastic algorithms* where the objective function changes at each iteration by randomly selecting subsamples.

Many stochastic algorithms have been proposed which use this approach for solving (4.2), no-

tably *stochastic gradient descent* (SGD), and variance-reduced extensions of SGD, such as SVRG [6], SAG [7], and SAGA [8]. These methods are *first-order* methods, extending gradient descent to the stochastic setting, and the latter three (variance-reduced) methods can be shown to converge linearly for strongly convex objectives. Linearly convergent stochastic Limited Memory BFGS algorithms [39][40] have also been proposed. It is then natural to consider stochastic extensions of quasi-Newton and second-order methods. One such method, the Newton Incremental Method (NIM) [9], combines cyclic updating of a fixed collection of functions $f(x, y_1), \ldots, f(x, y_m)$ with Newton's method, and attains local superlinear convergence.

One of the key obstacles to developing stochastic extensions of quasi-Newton methods is the necessity of selecting appropriate *step sizes*. The analysis of the global convergence of the BFGS method [10] and other members of *Broyden's convex class* [11] assumes that Armijo-Wolfe inexact line search is used. This is rather undesirable for a stochastic algorithm, as line search is both computationally expensive and difficult to analyze in a probabilistic setting. However, there is a special class of functions, the *self-concordant functions*, whose properties allow us to compute an *adaptive step size* and thereby avoid performing line searches. In [12], it is shown that the BFGS [13][14][15][16] method with adaptive step sizes converges superlinearly when applied to self-concordant functions.

In this chapter, our goal is to develop a stochastic quasi-Newton algorithm for self-concordant functions. We propose an iterative method of the following form. At the k-th iteration, we draw m_k i.i.d samples $\xi_1, \ldots, \xi_{m_k}$. Define the *empirical objective function* at the k-th iteration to be

$$F_k(x) = \frac{1}{m_k} \sum_{i=1}^{m_k} f(x, \xi_i). \tag{4.3}$$

Let H_k be a positive definite matrix. The next step direction is given by

$$d_k = -H_k \nabla F_k(x_k),$$

and the step size by

$$t_k = \frac{\alpha_k}{1 + \alpha_k \delta_k},$$

where

$$\alpha_k = \frac{\nabla F_k(x_k)^T H_k \nabla F_k(x_k)}{\delta_k^2} \text{ and} \tag{4.4}$$
$$\delta_k = \sqrt{\nabla F_k(x_k)^T H_k \nabla^2 F_k(x_k) H_k \nabla F_k(x_k)}.$$

The motivation for this step size is described in Section 4.4.

A key feature of these methods is that the step size t_k can be computed analytically, using only local information, and adapts itself to the local curvature. A fixed step size η is typically used in the SGD variants, and this step size must be determined experimentally. The theoretical analysis that has been provided for these methods is of little help in choosing η, as often η is constrained to be impractically small, and moreover, is related to unknown constants such as the Lipschitz parameter of the gradient. Furthermore, a fixed η which was effective in one phase may become ineffective as the algorithm progresses, and enters regions of varying curvature.

Our new methods are also capable of solving general problems of the form (4.1). This is in contrast to incremental-type methods such as SAG, SAGA, and NIM, which, because of their stored updating scheme, can only be applied to problems of the form (4.2) with a fixed data set $\{y_1, \ldots, y_m\}$. This opens up new avenues for the solutions of problems where new data can be sampled as the algorithm progresses, as opposed to having a fixed training set throughout.

By choosing the matrices H_k appropriately, we obtain stochastic extensions of classical methods. In particular, two choices of H_k will be of interest:

1. Taking $H_k = I$ yields the stochastic adaptive gradient descent method (SA-GD).

2. Fixing H_0, and then taking H_{k+1} to be the BFGS update of H_k, yields the stochastic adaptive BFGS method (SA-BFGS).

When the number of samples m_k is fixed, and large enough, both SA-GD and SA-BFGS con-

verge R-linearly in expectation to an ϵ-optimal solution. The SA-BFGS algorithm can also be shown to converge *R-superlinearly* to the true optimal solution if the number of samples is increased so that m_k^{-1} converges R-superlinearly to 0.

This chapter is organized as follows. In Section 4.2, we introduce our notation, and the technical assumptions needed for the analysis of our methods. In Section 4.3, we describe the relevant results from stochastic analysis which motivate our algorithms. In Section 4.4, we describe the required theory of self-concordant functions. In Section 4.5, we formally define stochastic adaptive methods, and prove the convergence results. In Section 4.6, we present preliminary numerical experiments, and conclude with a discussion in Section 4.7.

4.2 Assumptions and Notation

The number of variables is n. We use $g_k(x) = \nabla F_k(x)$ and $G_k(x) = \nabla^2 F_k(x)$ for the gradient and Hessian of the empirical objective function. In the context of a sequence of iterates $\{x_k\}_{k=0}^{\infty}$ generated by an algorithm, we also write g_k with no argument to denote $g_k(x_k)$, and G_k for $G_k(x_k)$. In the context of BFGS, H_k denotes the approximation to the inverse Hessian, and $B_k = H_k^{-1}$.

The optimal solution of $\min_x F(x)$ is denoted x^*, and the optimal solution of the empirical problem $\min_x F_k(x)$ is denoted x_k^*. Note that x_k^* is random.

Unless otherwise specified, the norm $\|\cdot\|$ is the 2-norm, or the operator 2-norm. The Frobenius norm is indicated as $\|\cdot\|_F$.

We make the following technical assumptions on $F(x)$ and $f(x, \xi)$. We will explain the motivation behind these assumptions at the relevant points in the discussion.

Assumptions:

1. There exist constants $L \geq \ell > 0$ such that for every $x \in \mathbb{R}^n$ and every realization of ξ, the Hessian of f with respect to x satisfies

$$\ell I \preceq \nabla_x^2 f(x, \xi) \preceq LI$$

That is, $f(x, \xi)$ is strongly convex for all ξ, with the eigenvalues of $\nabla_x^2 f(x, \xi)$ bounded by ℓ and L.

2. $F_k(x)$ is standard self-concordant for every possible sampling $\xi_1, \ldots, \xi_{m_k}$.

3. There exist compact sets $\mathcal{D}_0$ and $\mathcal{D}$ with $x^* \in \mathcal{D}$ and $\mathcal{D}_0 \subseteq \mathcal{D}$, such that if x_0 is chosen in $\mathcal{D}_0$, then for all possible realizations of the samples $\xi_1, \ldots, \xi_{m_k}$ for every k, the sequence of iterates $\{x_k\}_{k=0}^{\infty}$ produced by the algorithm is contained within $\mathcal{D}$. We use $D = \sup\{\|x - y\| : x, y \in \mathcal{D}\}$ for the diameter of $\mathcal{D}$.

   Furthermore, we assume that the objective values and gradients are bounded:

$$u = \sup_{\xi} \sup_{x \in \mathcal{D}} f(x, \xi) < \infty$$

$$l = \inf_{\xi} \inf_{x \in \mathcal{D}} f(x, \xi) > -\infty$$

$$\gamma = \sup_{\xi} \sup_{x \in \mathcal{D}} \|\nabla f(x, \xi)\| < \infty$$

4. (For BFGS only) The Hessian $G(x)$ is Lipschitz continuous with constant L_H.

Note that Assumption 1 is standard when analyzing stochastic algorithms. The function $f(x, \xi)$ is commonly a loss function, and either $f(x, \xi)$ is itself strongly convex, or each $f(x, \xi)$ is weakly convex and strong convexity is ensured by adding a quadratic regularization term to $F(x)$.

4.3 Stochastic Framework

Our analysis is based on a uniform convergence law, a standard technique in learning theory and empirical processes. The central idea is that $F_k(x)$ is an empirical mean estimating $F(x)$, and we can closely control the error $|F_k(x) - F(x)|$ over all $x \in \mathcal{D}$ by varying the sample size m_k. The relevant stochastic analysis can be found in [1]. [41] and [42] also consider strategies of increasing sample size during the computation of batch gradients that are based on pointwise variance estimate or pointwise tail bounds instead of a bound that is uniform over all possible

points. To be consistent with the notation in this chapter, we restate the Theorem 2.1.1 below as Theorem 4.3.1.

Theorem 4.3.1. *Let $\delta > 0$ and $0 < \epsilon < \min\{D, \frac{\delta}{2L}\}$. Then*

$$\mathbb{P}(\sup_{x \in \mathcal{D}} |F_k(x) - F(x)| > \delta) \leq c(\epsilon) \exp\left(-\frac{m_k(\delta - 2L\epsilon)^2}{2(u-l)^2}\right)$$

where $c(\epsilon) = 2n^{n/2}D^n\epsilon^{-n}$. If $m_k \geq 3$, then

$$\mathbb{E} \sup_{x \in \mathcal{D}} |F_k(x) - F(x)| \leq C\sqrt{\frac{\log m_k}{m_k}}$$
$$\mathbb{E}|F_k(x_k^*) - F(x^*)| \leq C\sqrt{\frac{\log m_k}{m_k}}$$

where C is given by

$$4(|u| + |l|)n^{n/2}D^n \exp\left[-n\left(\log \frac{u-l}{2\sqrt{2}L}\right)\right] + (u-l)\sqrt{n+1}$$

Assumption 3 is required for the uniform law of Theorem 4.3.1 to hold. The set $\mathcal{D}_0$ is assumed to be a region where, if x_0 is chosen within $\mathcal{D}_0$, the path of the stochastic algorithm will remain within the larger set $\mathcal{D}$. For practical purposes, we may take $\mathcal{D}$ to be an arbitrarily large bounded set in order to ensure convergence, though this worsens the complexity bounds provided by Theorem 4.3.1. We note that the constant C is very much a 'worst-case' bound, and for almost every problem arising in practice, the expected difference will be much smaller than Theorem 4.3.1 suggests. Rather, the crucial implication is that the expected difference diminishes at the rate $\sqrt{\frac{\log m_k}{m_k}}$, allowing a sharp level of control by adjusting m_k.

We will also use Theorem 4.3.1 to bound g_k and G_k. Assumption 1 implies that the partial derivatives $\frac{\partial F}{\partial x_i}(x)$ and $\frac{\partial^2 F}{\partial x_i x_j}(x)$ are also uniformly bounded for $x \in \mathcal{D}$. Hence, we can apply Theorem 4.3.1 to each of the n entries $\frac{\partial F}{\partial x_i}$ of the gradient, and each of the n^2 entries $\frac{\partial^2 F}{\partial x_i \partial x_j}$ of the Hessian. Taking a union bound over the resulting $n^2 + n$ inequalities, we obtain the following

concentration inequality for the sampled gradients and Hessians:

Corollary 4.3.2. *For any $\delta > 0$ and $0 < \epsilon < \min\{D, \frac{\delta}{2L}\}$,*

$$\mathbb{P}(\sup_{x \in \mathcal{D}} \|G_k(x) - G(x)\| > \delta \text{ or } \sup_{x \in \mathcal{D}} \|g_k(x) - g(x)\| > \delta) \leq C_1 \epsilon^{-n} \exp\left[-C_2 m_k (\delta - C_3 \epsilon)^2\right]$$

where C_1, C_2, C_3 are constants depending only on F.

Recall the definitions of $\delta_k, \rho_k, \alpha_k$, and η_k above. In our analysis of stochastic methods, the gradients and Hessians are those of the *empirical* objective function. That is to say, $\rho_k = g_k^T H_k g_k$ and $\delta_k = \sqrt{d_k^T G_k d_k}$, where g_k and G_k are the gradient and Hessian of F_k.

We say that a constant c is *global* if it depends only on the properties of the function F, and is completely independent of the realization of the samples $\xi_1, \ldots, \xi_{m_k}$.

4.4 Self-Concordant Functions and Adaptive Methods

A convex function $f : \mathbb{R}^n \to \mathbb{R}$ is *self-concordant* if there exists a constant κ such that for every $x \in \mathbb{R}^n$ and every $h \in \mathbb{R}^n$, we have

$$|\nabla^3 f(x)[h, h, h]| \leq \kappa (\nabla^2 f(x)[h, h])^{3/2}$$

If $\kappa = 2$, f is *standard self-concordant*. Self-concordant functions were introduced by Nesterov and Nemirovski in the context of interior point methods [43].

Many common problems have self-concordant formulations. At least one method, the DiSCO algorithm of [44], is tailored for distributed self-concordant optimization and has been applied to many regression problems. Convex quadratic objective functions have third derivatives equal to zero, and are therefore trivially standard self-concordant. In particular, least squares regression is self-concordant. In [44], it is also shown that regularized regression, with either logistic loss or hinge loss, is self-concordant.

For self-concordant functions, the notion of a *local norm* is especially useful. Given a convex

function f, the local norm with respect to f at the point x is given by

$$\|h\|_x = \sqrt{h^T \nabla^2 f(x) h}$$

Consider an iterative method for minimizing self-concordant functions with steps given as follows. On the k-th step, the step direction d_k is given by $d_k = -H_k \nabla f(x_k)$ for some positive definite matrix H_k, and the step size is given by $t_k = \frac{\alpha_k}{1+\alpha_k \delta_k}$, where $\delta_k = \|d_k\|_{x_k}$ and $\alpha_k = \frac{g_k^T H_k g_k}{\delta_k^2}$.

Methods of this type have been analyzed in [45] and [12]. In [12], the above choice of α_k is shown to guarantees a decrease in the function value.

Theorem 4.4.1 (Lemma 4.1, [12])**.** *Let $\rho_k = \nabla f(x_k)^T H_k \nabla f(x_k)$. If α_k is chosen to be $\alpha_k = \frac{\rho_k}{\delta_k^2}$, then*

$$f(x_k + t_k d_k) \leq f(x_k) - \omega(\eta_k),$$

where $\eta_k = \frac{\rho_k}{\delta_k}$ and the function $\omega(z) = z - \log(1+z)$.

We make Assumption 2 in order to apply Theorem 4.4.1 to the empirical objective functions $F_k(x)$. A natural question is whether Assumption 2 can be relaxed to the assumption that $f(x, \xi)$ is self-concordant for all ξ, and $F(x)$ is standard self-concordant. In the case where ξ is finitely supported, it is possible to scale $F(x)$ so that we may use this weaker assumption.

Lemma 4.4.2. *Suppose that ξ is finitely supported on $\{a_1, \ldots, a_m\}$, with $p_i = \mathbb{P}(\xi = a_i)$. Suppose that $f(x, a_i)$ is self-concordant with constant κ_i. Let $\theta = \frac{1}{4} \frac{\max \kappa_i^2}{\min p_i}$. Then the scaled function $\overline{F}(x) = \mathbb{E}[\theta f(x, \xi)]$ is standard self-concordant, and every empirical objective function $\overline{F}_k(x)$ is standard self-concordant.*

Proof. Observe that $\theta f(x, a_i) p_i$ is self-concordant with constant $\theta^{-1/2} p_i^{-1/2} \kappa_i \leq 2$. We deduce that $\mathbb{E}\overline{F}(x) = \sum_{i=1}^m \theta f(x, a_i) p_i$ is standard self-concordant. Furthermore, since $\sum_{i=1}^m p_i = 1$, we have $\min p_i \leq \frac{1}{m}$. Thus, $\frac{1}{m}\theta f(x, a_i)$ is self-concordant with constant $\theta^{-1/2} m^{-1/2} \kappa_i \leq 2$, which implies that $\overline{F}_k(x)$ is standard self-concordant for every possible $\overline{F}_k(x)$. □

However, if we do not assume that each $f(x, \xi)$ is self-concordant, or if ξ is not compactly supported, it is unclear whether every possible $F_k(x)$ will be standard self-concordant, even when $F(x)$ is standard self-concordant. Thus, we impose Assumption 2 in our analysis, while observing that it is unnecessary in the practical case where ξ is derived from a finite data set.

4.5 Stochastic Adaptive Methods

Our basic approach is to sample an empirical objective function $F_k(x)$ at each step, and then compute the step direction and adaptive step size (4.4) using F_k. For particular choices of the matrices H_k, we recover analogues of classical methods such as gradient descent, L-BFGS, and BFGS.

4.5.1 Stochastic Adaptive GD

When $H_k = I$ for every k, the resulting method is *stochastic adaptive gradient descent* (SA-GD), which is given as Algorithm 6. The number of samples drawn at each iteration is left as an input to the algorithm, and (weak) bounds on the required number m_k can be inferred from the convergence analysis.

Algorithm 6 SA-GD

Input: x_0, $\{m_0, m_1, \ldots\}$
for $k = 0, 1, 2, \ldots$ **do**
  Sample $\xi_1, \ldots, \xi_{m_k}$ i.i.d from ξ
  Compute:

$$g_k = \nabla F_k(x_k), \quad d_k = -g_k$$
$$\delta_k = \sqrt{g_k^T G_k(x_k) g_k}$$
$$\alpha_k = \frac{g_k^T g_k}{\delta_k^2}, t_k = \frac{\alpha_k}{1 + \alpha_k \delta_k}$$

  where $F_k(x) = \frac{1}{m_k} \sum_{i=1}^{m_k} f(x, \xi_i)$.
  Set $x_{k+1} = x_k + t_k g_k$.
end for

Theorem 4.5.1. *Let $\epsilon > 0$ be fixed. At each iteration, we draw m i.i.d samples $\xi_1, \ldots, \xi_m$, where the size of m satisfies*

$$\frac{\log m}{m} \leq \left(\frac{1-r}{4C}\right)^2 \epsilon^2$$

and C is the constant in Theorem 4.3.1 and $r = 1 - \frac{\lambda^2 \ell^{3/2}}{(\sqrt{\ell}+\gamma)\Lambda^2 L}$. Then we have

$$\mathbb{E}F(x_{k+1}) - F(x^*) \leq \epsilon$$

when $k = \log(\epsilon^{-1}2(u-l))/\log r$.

We need the following theorem and lemma to prove our main theorem (Theorem 4.5.1) in this subsection.

For matrices H_k with bounded eigenvalues, $\eta_k = \rho_k/\delta_k$ can readily be bounded in terms of the empirical gradients, and the sequence $\{\eta_k\}_{k=0}^{\infty}$ is bounded.

Theorem 4.5.2. *There exists a global constant $\Gamma = \frac{\gamma}{\sqrt{\ell}}$ such that $\eta_k \leq \Gamma$ for all k. Furthermore, $\eta_k \geq \frac{\lambda}{\Lambda\sqrt{L}}\|g_k\|$ for all k.*

Proof. By Assumption 1 (strong convexity), G_k satisfies $\ell I \preceq G_k \preceq LI$. Thus, from the definition of η_k, we have

$$\eta_k = \frac{g_k^T H_k g_k}{\sqrt{g_k^T H_k G_k H_k g_k}} \leq \frac{\|g_k\|\|H_k g_k\|}{\sqrt{\ell}\|H_k g_k\|} = \frac{1}{\sqrt{\ell}}\|g_k\|$$

By Assumption 3, we find that $\|g_k\| = \|g_k(x_k)\| \leq \gamma$. Hence, we may take $\Gamma = \frac{\gamma}{\sqrt{\ell}}$. We also find that

$$\eta_k = \frac{g_k^T H_k g_k}{\sqrt{g_k^T H_k G_k H_k g_k}} \geq \frac{\lambda}{\Lambda\sqrt{L}}\|g_k\|$$

□

Lemma 4.5.3. *The empirical objective function $F_k(x)$ satisfies*

$$F_k(x_{k+1}) - F_k(x_k^*) \leq r(F_k(x_k) - F_k(x_k^*))$$

for the global constant $r = 1 - \frac{\ell}{(1+\Gamma)L} < 1$.

Proof. Observe that the function $\omega(z)$ satisfies $\omega(z) \geq \frac{1}{2}(1+\Gamma)^{-1}z^2$ for all $z \in [0, \Gamma]$. Also, recall that the strongly convex function F_k satisfies $\|g_k(x)\|^2 \geq 2\ell(F_k(x) - F_k(x_k^*))$. By Theorem 4.4.1 and Theorem 4.5.2, we find that

$$\begin{aligned} F_k(x_{k+1}) - F_k(x_k^*) \leq F_k(x_k) - F_k(x_k^*) - \omega(\eta_k) &\leq F_k(x_k) - F_k(x_k^*) - \frac{1}{2}(1+\Gamma)^{-1}\eta_k^2 \\ &\leq F_k(x_k) - F_k(x_k^*) - \frac{1}{2}(1+\Gamma)^{-1}\frac{\lambda^2}{\Lambda^2 L}\|g_k\|^2 \\ &\leq \left(1 - \frac{\lambda^2\ell}{(1+\Gamma)\Lambda^2 L}\right)(F_k(x_k) - F_k(x_k^*)) \end{aligned}$$

Thus, we may take $r = 1 - \frac{\lambda^2\ell}{(1+\Gamma)\Lambda^2 L}$. For SA-GD in particular, $\lambda = \Lambda = 1$, so $r = 1 - \frac{\ell}{(1+\Gamma)L}$. □

We are now ready to prove Theorem 4.5.1.

Proof. By Lemma 4.5.3, we calculate that

$$\begin{aligned} F_k(x_{k+1}) - F_k(x_k^*) &\leq r(F_k(x_k) - F_k(x_k^*)) \\ &= r(F_{k-1}(x_k) - F_{k-1}(x_{k-1}^*)) \\ &\quad + r(F_k(x_k) - F(x_k) - F_{k-1}(x_k) + F(x_k)) \\ &\quad + r(F_{k-1}(x_{k-1}^*) - F(x^*) - F_k(x_k^*) + F(x^*)) \\ &\leq r(F_{k-1}(x_k) - F_{k-1}(x_{k-1}^*)) \\ &\quad + r(\sup_{x\in\mathcal{D}}|F_k(x) - F(x)| + \sup_{x\in\mathcal{D}}|F_{k-1}(x) - F(x)|) \\ &\quad + r(|F_k(x_k^*) - F(x^*)| + |F_{k-1}(x_{k-1}^*) - F(x^*)|) \end{aligned}$$

By iterating this expansion, we find that

$$\begin{aligned} F_k(x_{k+1}) - F_k(x_k^*) \leq\ & r^k(F_0(x_1) - F_0(x_0^*)) \\ & + \sum_{j=1}^{k} r^j(\sup_{x\in\mathcal{D}} |F_{k+1-j}(x) - F(x)| + \sup_{x\in\mathcal{D}} |F_{k-j}(x) - F(x)|) \\ & + \sum_{j=1}^{k} r^j(|F_{k+1-j}(x_{k+1-j}^*) - F(x^*)| + |F_{k-j}(x_{k-j}^*) - F(x^*)|) \end{aligned}$$

Decompose $F_k(x_{k+1}) - F_k(x_k^*)$ as

$$F_k(x_{k+1}) - F_k(x_k^*) = F(x_{k+1}) - F(x^*) + [F_k(x_{k+1}) - F(x_{k+1})] + [F(x^*) - F_k(x_k^*)]$$

We can move the terms in square brackets to the right hand side, and upper bound them, to obtain

$$\begin{aligned} F(x_{k+1}) - F(x^*) \leq\ & r^k(F_0(x_1) - F_0(x_0^*)) \\ & + \sup_{x\in\mathcal{D}} |F_k(x) - F(x)| \\ & + \sum_{j=1}^{k} r^j(\sup_{x\in\mathcal{D}} |F_{k+1-j}(x) - F(x)| + \sup_{x\in\mathcal{D}} |F_{k-j}(x) - F(x)|) \\ & + |F_k(x_k^*) - F(x^*)| \\ & + \sum_{j=1}^{k} r^j(|F_{k+1-j}(x_{k+1-j}^*) - F(x^*)| + |F_{k-j}(x_{k-j}^*) - F(x^*)|) \end{aligned} \tag{4.5}$$

Suppose that we draw a constant number of samples $m_k = m$ at each iteration. Taking expectations on both sides of equation (4.5) and applying the concentration bound of Theorem 4.3.1, we obtain

$$\begin{aligned} \mathbb{E}F(x_{k+1}) - F(x^*) &\leq r^k(u - l) + 2C\sqrt{\frac{\log m}{m}} \sum_{j=0}^{k} r^j \\ &\leq r^k(u - l) + \frac{2C}{1 - r}\sqrt{\frac{\log m}{m}} \end{aligned}$$

In order to obtain an ϵ-optimal solution, we may use sufficiently large samples, and take sufficiently

many iterations, so that

$$r^k(u-l) \leq \frac{\epsilon}{2}$$
$$\frac{2C}{1-r}\sqrt{\frac{\log m}{m}} \leq \frac{\epsilon}{2}$$

This yields the given bounds on m and k in Theorem 4.5.1. □

In particular, it suffices to take $m = O(\epsilon^{-2}\log\epsilon^{-1})$ and $k = O(\log\epsilon^{-1})$.

4.5.2 Stochastic Adaptive BFGS

By updating H_k using the BFGS formula, we obtain the *stochastic adaptive BFGS* (SA-BFGS) method, which is given in Algorithm 7.

Algorithm 7 SA-BFGS

Input: x_0, H_0, $\{m_0, m_1, \ldots\}$, $\beta < 1$
for $k = 0, 1, 2, \ldots$ **do**
  Sample $\xi_1, \ldots, \xi_{m_k}$ i.i.d from ξ
  Compute

$$g_k = \nabla F_k(x_k), d_k = -H_k g_k$$
$$\delta_k = \sqrt{d_k^T G_k(x_k) d_k}$$
$$\alpha_k = \frac{g_k^T H_k g_k}{\delta_k^2}, t_k = \frac{\alpha_k}{1+\alpha_k\delta_k}$$

  where $F_k(x) = \frac{1}{m_k}\sum_{i=1}^{m_k} f(x, \xi_i)$.
  Set $g_{k+1} = \nabla F_k(x_k + t_k d_k)$
  Set $y_k = g_{k+1} - g_k$
  if $g_{k+1}^T d_k < \beta g_k^T d_k$ **then**
    Set $d_k = g_k$
    Recompute δ_k, α_k, t_k
    Set $H_{k+1} = H_k$
  else
    Set $H_{k+1} = (I - \frac{s_k y_k^T}{s_k^T y_k})H_k(I - \frac{y_k s_k^T}{s_k^T y_k}) + \frac{s_k s_k^T}{s_k^T y_k}$
  end if
  Set $x_{k+1} = x_k + t_k d_k$.
end for

In Algorithm 7, we use the standard BFGS update with $y_k = g_k(x_{k+1}) - g_k(x_k)$. Another option is to replace y_k with the action of the Hessian on s_k, so $y_k = G_k(x_k)s_k$. In general, we must compute $G_k(x_k)d_k$ when finding the adaptive step size, so we can re-use the result of that computation instead of computing an extra gradient $g_k(x_{k+1})$.

For technical reasons, our SA-BFGS procedure tests whether the Wolfe condition is satisfied for the adaptive step size t_k. If not, we revert to taking a SA-GD step. This is an artifact of our analysis, and under suitable conditions on the growth of the samples m_k, there will be some point after which the Wolfe condition is necessarily satisfied on every step. In practice, omitting this test does not impact performance.

There are also two possible ways to implement SA-BFGS. For problems with n at most medium-sized, it is possible to explicitly store the matrix H_k, and compute d_k by a matrix product $-H_k g_k$. For n very large, it is infeasible to store H_k, and we can instead store the pairs (s_k, y_k) and compute $-H_k g_k$ using a two-loop recursion [46]. This corresponds to stochastic adaptive L-BFGS (SA-LBFGS) if we limit the number of past pairs (s_k, y_k) to only the h most recent, and to SA-BFGS if we store everything. The amount of storage used by SA-LBFGS surpasses that of SA-BFGS as h approaches n.

Theorem 4.5.1 holds for SA-LBFGS, since it holds for any method where $\{H_k\}$ has uniformly bounded eigenvalues. Thus, SA-LBFGS also converges in expectation to an ϵ-optimal solution after $k = O(\log \epsilon^{-1})$ steps given samples of size $m = O(\epsilon^{-2} \log \epsilon^{-1})$, though now the constants within the big-O are dependent on h.

Now, we will layout our framework for proving that SA-BFGS converges superlinearly with probability 1.

Theorem 4.5.4. *Suppose that we draw m_k samples on the k-th step, where m_k^{-1} converges R-superlinearly to 0. Then SA-BFGS converges to the optimal solution x^* almost surely.*

Our arguments closely follow the proofs given in [10] and [47] for the deterministic BFGS method. Along the way, we will also consider the behavior of SA-BFGS when ϵ-optimality suffices, and m_k is held constant. Note that the results preceding Lemma 4.5.13 do not depend on any

particular choice of sample sizes m_k.

We introduce the following assumption in this section:

Assumption:

4. The Hessian $G(x)$ is Lipschitz continuous with constant L_H.

The adaptive step size is known to satisfy the Armijo-Wolfe conditions in the deterministic setting. A similar property holds for the empirical objective functions.

Theorem 4.5.5 (Theorem 6.2, [12])**.** *The adaptive step size t_k satisfies the Armijo condition for $\alpha = \frac{1}{2}$, for the empirical objective function $F_k(x)$.*

Recall that the SA-BFGS algorithm performs a BFGS update at step k only if t_k satisfies the Wolfe condition. If t_k does not satisfy the Wolfe condition, then we take a SA-GD step instead. In this case, the direction is $-g_k$ and the step size is the adaptive step size for SA-GD.

We use $q(j)$ to denote the the index of the j-th BFGS step, or equivalently, the index at which the j-th BFGS update is performed. The steps $\{q(j)\}_{j=1}^{\infty}$ where we perform BFGS updates will be referred to as *update times*. Later on, we will see that if m_k grows at a sufficient rate, then all $q(j)$ exist with probability 1.

The following technical lemma is used in the analysis of BFGS; it can also be found in [11] and [10].

Lemma 4.5.6. *Let $k = q(j)$ be an update time. Let $\overline{G}_k = \int_0^1 G_k(x_k + \tau s_k)d\tau$, and let θ_k denote the angle between the vectors $-g_k$ and s_k. Then*

1. $y_k = \overline{G}_k s_k$, *and* $s_k^T y_k \leq L\|s_k\|^2$.

2. $\|s_k\| \leq \frac{1}{\ell}\|g_k\| \cos\theta_k$

3. *If the Wolfe condition is satisfied on step k, then $\langle y_k, s_k\rangle \geq (1-\beta)\langle -g_k, s_k\rangle$ and $\|s_k\| \geq \frac{(1-\beta)}{L}\|g_k\| \cos\theta_k$.*

Proof. The first statement follows from the definition $y_k = g_k(x_{k+1}) - g_k(x_k)$. Since $G_k(x) \preceq LI$ for all x, we also have $\overline{G}_k \preceq LI$, and hence $s_k^T y_k = s_k^T \overline{G}_k s_k \leq L\|s_k\|^2$.

The second statement follows from the Armijo condition (Theorem 4.5.5) and Taylor's theorem. Let $\overline{x}$ be a point on the line $[x_k, x_{k+1}]$ with $F_k(x_{k+1}) = F_k(x_k) + \langle g_k, s_k\rangle + \frac{1}{2}s_k^T G_k(\overline{x})s_k$. Since $F_k(x_{k+1}) - F_k(x_k) \leq \frac{1}{2}\langle g_k, s_k\rangle$, we have $\frac{1}{2}\langle -g_k, s_k\rangle \geq \frac{1}{2}s_k^T G_k(\overline{x})s_k \geq \frac{1}{2}m\|s_k\|^2$ as desired.

The Wolfe condition implies that $\langle y_k, s_k\rangle = \langle g_k(x_{k+1}) - g_k(x_k), s_k\rangle \geq (1-\beta)\langle -g_k, s_k\rangle$. Writing $\langle -g_k, s_k\rangle = \|g_k\|\|s_k\|\cos\theta_k$, we have $L\|s_k\|^2 \geq (1-\beta)\|g_k\|\|s_k\|\cos\theta_k$, which gives the last statement. □

The next result is the key technical lemma in proving that SA-BFGS converges R-linearly. Its proof is identical to the deterministic case [10].

Lemma 4.5.7. *There exists a global constant c such that*

$$\prod_{j=1}^{k} \frac{\|g_{q(j)}\|^2}{\langle -g_{q(j)}, s_{q(j)}\rangle} \leq c^k.$$

Proof. By considering the BFGS update formula, we have

$$\mathrm{Tr}(B_{j+1}) = \mathrm{Tr}(B_j) - \frac{s_j^T B_j^2 s_j}{s_j^T B_j s_j} + \frac{y_j^T y_j}{s_j^T y_j},$$

Recall from Lemma 4.5.6 that $y_j = \overline{G}_j s_j$. Therefore, writing $z_j = \overline{G}_j^{1/2} s_j$, we have

$$\frac{y_j^T y_j}{s_j^T y_j} = \frac{z_j^T \overline{G}_j z_j}{z_j^T z_j} \leq L,$$

where the last inequality follows from Assumption 1. Let $c_1 = \mathrm{Tr}(B_0) + kL$. The BFGS formula implies that $\mathrm{Tr}(B_{q(k+1)}) \leq \mathrm{Tr}(B_0) + kL \leq c_1 k$, and since $B_{q(k+1)}$ is positive definite, we also have

$$\sum_{j=1}^{k} \frac{s_{q(j)}^T B_{q(j)}^2 s_{q(j)}}{s_{q(j)}^T B_{q(j)} s_{q(j)}} \leq \mathrm{Tr}(B_0) + kL \leq c_1 k.$$

Observe that $s_j^T B_j^2 s_j = t_j^2 \|g_j\|^2$ and that $s_j^T B_j s_j = t_j \langle -g_j, s_j \rangle$. By the arithmetic mean-geometric mean (AM-GM) inequality,

$$\prod_{j=1}^{k} \frac{t_{q(j)} \|g_{q(j)}\|^2}{\langle -g_{q(j)}, s_{q(j)} \rangle} \leq c_1^k. \tag{4.6}$$

Next, we use the recursive formula for the determinant:

$$\det(B_{j+1}) = \frac{y_j^T s_j}{s_j^T B_j s_j} \det(B_j).$$

Since the Wolfe condition is satisfied, we have

$$y_j^T s_j = (g_j(x_{j+1}) - g_j(x_j))^T s_j \geq (1 - \beta) \langle -g_j, s_j \rangle.$$

Therefore,

$$\det(B_{q(k+1)}) \geq \det(B_0) \prod_{j=1}^{k} \frac{1 - \beta}{t_{q(j)}}.$$

By the AM-GM inequality applied to the eigenvalues of $B_{q(k+1)}$, we find that $\det(B_{q(k+1)}) \leq (c_1 k/n)^n \leq c_2^k$ for a global constant c_2. Hence, $\prod_{j=1}^{k} \frac{1-\beta}{t_{q(j)}} \leq c_2^k$. Multiplying this together with inequality (4.6), and taking $c = \frac{c_1}{(1-\beta)c_2}$, we find that

$$\prod_{j=1}^{k} \frac{\|g_{q(j)}\|^2}{\langle -g_{q(j)}, s_{q(j)} \rangle} \leq c^k$$

as desired. □

Lemma 4.5.8. *At least $\frac{1}{2}k$ of the angles $\theta_{q(1)}, \ldots, \theta_{q(k)}$ satisfy $\cos^2 \theta_{q(j)} > (\ell/c)^2$, where c is the constant of Lemma 4.5.7.*

Proof. By Lemma 4.5.6, $\|s_j\| \leq \frac{1}{\ell} \|g_j\| \cos \theta_j$. Substituting this in Lemma 4.5.7 yields

$$c^k \geq \prod_{j=1}^{k} \frac{\|g_{q(j)}\|^2}{\langle -g_{q(j)}, s_{q(j)} \rangle} \geq \prod_{j=1}^{k} \frac{\ell}{\cos^2 \theta_{q(j)}} = \ell^{k+1} \prod_{j=1}^{k} \frac{1}{\cos^2 \theta_{q(j)}}.$$

Hence, $\prod_{j=1}^{k} \cos^2 \theta_{q(j)} \geq (\ell/c)^k$. It follows that at least $\frac{1}{2}k$ of the angles must satisfy $\cos^2 \theta_{q(j)} \geq (\ell/c)^2$. □

We can proceed to show that stochastic adaptive BFGS converges R-linearly. The argument proceeds by showing that if k is not an update time, then SA-BFGS inherits the Q-linear convergence rate of SA-GD, and if $k = q(j)$, then we can measure the decrement with Lemma 4.5.7.

Lemma 4.5.9. *If k is not an update time, then*

$$F_k(x_{k+1}) - F_k(x_k^*) \leq r(F_k(x_k) - F_k(x_k^*)),$$

where $r = 1 - \frac{\ell^{3/2}}{(\sqrt{\ell}+\gamma)L}$.

Proof. This follows from Lemma 4.5.3 for SA-GD. □

Lemma 4.5.10. *Let $k = q(j)$. Then*

$$F_k(x_{k+1}) - F_k(x_k^*) \leq \left(1 - (1-\beta)\ell L^{-1} \cos^2 \theta_k\right)(F_k(x_k) - F_k(x_k^*)).$$

Proof. Since the adaptive step size t_k satisfies the Armijo condition for $\alpha = \frac{1}{2}$, we have

$$F_k(x_{k+1}) - F_k(x_k) \leq \frac{1}{2}\langle g_k, s_k \rangle = -\frac{1}{2}\|g_k\|\|s_k\| \cos \theta_k.$$

Using Lemma 4.5.6, we rewrite $\|s_k\|$ in terms of $\|g_k\|, \cos \theta_k$ to obtain

$$F_k(x_{k+1}) - F_k(x_k) \leq -\frac{1}{2}(1-\beta)L^{-1}\|g_k\|^2 \cos^2 \theta_k.$$

Since $\|g_k\|^2 \geq 2\ell(F_k(x_k) - F_k(x_k^*))$, we rearrange to obtain

$$F_k(x_{k+1}) - F_k(x_k^*) \leq (1 - (1-\beta)\ell L^{-1} \cos^2 \theta_k)(F_k(x_k) - F_k(x_k^*)).$$

□

Theorem 4.5.11. *Suppose that we draw samples of size m_k at step k, where m_k^{-1} converges superlinearly to 0. With probability 1, SA-BFGS converges R-linearly.*

Proof. Let $\nu = \max\{1 - (1-\beta)\ell L^{-1}(\ell/c)^2, r\} < 1$. Let $\mathbb{I}_1(k)$ be the 0-1 indicator variable for the event that k is a BFGS update time, and let $\mathbb{I}_2(k)$ be the indicator for the event that k is a BFGS update time *and* $\cos^2\theta_k \geq (\ell/c)^2$. Combining Lemma 4.5.9 and Lemma 4.5.10 by using these indicator variables, we have

$$\begin{aligned} F_k(x_{k+1}) - F_k(x_k^*) &\leq (1 - (1-\beta)\ell L^{-1}\cos^2\theta_k)^{\mathbb{I}_1(k)} r^{1-\mathbb{I}_1(k)}(F_k(x_k) - F_k(x_k^*)) \\ &\leq (1 - (1-\beta)\ell L^{-1}(\ell/c)^2)^{\mathbb{I}_2(k)} r^{1-\mathbb{I}_1(k)}(F_k(x_k) - F_k(x_k^*)) \\ &\leq \nu^{\mathbb{I}_2(k)+1-\mathbb{I}_1(k)}(F_k(x_k) - F_k(x_k^*)). \end{aligned}$$

For any $t \leq k$, let $b(t) = \sum_{j=0}^{t} \mathbb{I}_1(j)$. Rewritten with indicators, Lemma 4.5.8 states that $\sum_{j=0}^{t} \mathbb{I}_2(j) \geq \frac{1}{2}b(t)$. Therefore,

$$\sum_{j=0}^{k}(\mathbb{I}_2(j) + 1 - \mathbb{I}_1(j)) \geq k - \frac{1}{2}b.$$

Define $\mathbb{I}_3(k) = \mathbb{I}_2(k) + 1 - \mathbb{I}_1(k)$. Iterating the above expansion, we have

$$\begin{aligned}
F_k(x_{k+1}) - F_k(x_k^*) &\le \nu^{\mathbb{I}_3(k)}(F_k(x_k) - F_k(x_k^*)) \\
&\le \nu^{\mathbb{I}_3(k)}(F_{k-1}(x_k) - F_{k-1}(x_{k-1}^*) + (F_k(x_k) - F_{k-1}(x_k)) + (F_{k-1}(x_{k-1}^*) - F_k(x_k^*)) \\
&\le \nu^{\sum_{i=0}^k \mathbb{I}_3(i)}(F_0(x_0) - F_0(x_0^*)) \\
&\quad + \sum_{j=1}^k \nu^{\sum_{i=j}^k \mathbb{I}_3(i)}[\sup_{x\in\mathcal{D}} |F_j(x) - F(x)| + \sup_{x\in\mathcal{D}} |F_{j-1}(x) - F(x)|] \\
&\quad + \sum_{j=1}^k \nu^{\sum_{i=j}^k \mathbb{I}_3(i)}[|F_j(x_j^*) - F(x^*)| + |F_{j-1}(x_{j-1}^*) - F(x^*)|] \\
&\le \nu^{k-b/2}(F_0(x_0) - F_0(x_0^*)) \\
&\quad + 2\sum_{0\le j\le k-b/2} \nu^{k-b/2-j}(\sup_{x\in\mathcal{D}} |F_j(x) - F(x)| + |F_j(x_j^*) - F(x^*)|) \\
&\quad + 2\sum_{j>k-b/2}^k (\sup_{x\in\mathcal{D}} |F_j(x) - F(x)| + |F_j(x_j^*) - F(x^*)|).
\end{aligned}$$

In the last inequality, we have simply split the sums into two sums, one running over the indices $0 \le j \le k - b/2$ and the other over $k - b/2 < j \le k$. Writing the left side as

$$F_k(x_{k+1}) - F_k(x_k^*) = F(x_{k+1}) - F(x^*) + (F_k(x_{k+1}) - F(x_{k+1})) + (F(x^*) - F_k(x_k^*)),$$

we can move terms to the right to obtain

$$\begin{aligned}
F(x_{k+1}) - F(x^*) &\le \nu^{k-b/2}(F_0(x_0) - F_0(x_0^*)) \\
&\quad + \sup_{x\in\mathcal{D}} |F_k(x) - F(x)| + |F_k(x_k^*) - F(x^*)| \\
&\quad + 2\sum_{0\le j\le k-b/2} \nu^{k-b/2-j}(\sup_{x\in\mathcal{D}} |F_j(x) - F(x)| + |F_j(x_j^*) - F(x^*)|) \\
&\quad + 2\sum_{j>k-b/2}^k (\sup_{x\in\mathcal{D}} |F_j(x) - F(x)| + |F_j(x_j^*) - F(x^*)|).
\end{aligned}$$

Taking expectations, and applying Theorem 4.3.1 on the right, we have

$$\mathbb{E}F(x_{k+1}) - F(x^*) \leq \nu^{k-b/2}(u-l) + 4C \sum_{0 \leq j \leq k-b/2} \nu^{k-b/2-j} \sqrt{\frac{\log m_j}{m_j}} + 4C \sum_{j > k-b/2}^{k} \sqrt{\frac{\log m_j}{m_j}}. \quad (4.7)$$

Our choice of m_j satisfies $m_j = \Omega(\nu^{-2j})$, so $\sqrt{\frac{\log m_j}{m_j}} = O(\nu^j \sqrt{j})$. Hence, by bounding each term with a multiple of $\nu^{k-b/2}$, we may find a global constant ϕ, with $1 > \phi > \nu$, and a global constant c_3, such that

$$\mathbb{E}F(x_{k+1}) - F(x^*) \leq c_3 \phi^{k-b/2}.$$

Clearly $b \leq k$, and thus we find that

$$\mathbb{E}F(x_{k+1}) - F(x^*) \leq c_3 \phi^{k/2}.$$

Now, fix any constant φ with $\phi < \varphi < 1$. By Markov's inequality,

$$\mathbb{P}(F(x_k) - F(x^*) \geq \varphi^{k/2}) \leq \frac{\mathbb{E}(F(x_k) - F(x^*))}{\varphi^{k/2}} \leq c_3 \left(\frac{\phi}{\varphi}\right)^{k/2}.$$

Since $\sum_{k=0}^{\infty} \left(\frac{\phi}{\varphi}\right)^{k/2} < \infty$, the Borel-Cantelli Lemma implies that the sequence of events A_k with

$$A_k = \{F(x_k) - F(x^*) > \varphi^{k/2}\}$$

occurs finitely often with probability 1. Therefore, with probability 1, SA-BFGS converges R-linearly. □

Before proceeding further, let us digress briefly to consider the behavior of SA-BFGS when we are satisfied with an ϵ-optimal solution, and wish to hold the number of samples constant.

Lemma 4.5.12. *Let $\epsilon > 0$. Suppose we draw m i.i.d samples at each step, where $m = O(\epsilon^2(\log \epsilon^{-1})^3)$. Then SA-BFGS converges in expectation to an ϵ-optimal solution after k steps, where $k = O(\epsilon^{-1})$.*

Proof. Note that equation (4.7) in the proof of Theorem 4.5.11 holds in the absence of any as-

sumptions on the sample sizes m_k. Suppose that we take $m_k = m$. Then we have

$$\begin{aligned}\mathbb{E}F(x_{k+1}) - F(x^*) &\leq \nu^{k-b/2}(u-l) + 4C \sum_{0\leq j\leq k-b/2} \nu^{k-b/2-j}\sqrt{\frac{\log m_j}{m_j}} + 4C \sum_{j>k-b/2}^{k} \sqrt{\frac{\log m_j}{m_j}} \\ &\leq \nu^{k/2}(u-l) + 4C\sqrt{\frac{\log m}{m}}\left(\frac{1}{1-\nu} + k/2\right).\end{aligned}$$

Therefore, in order to obtain an ϵ-optimal solution from SA-BFGS, we may take

$$\begin{aligned}\nu^{k/2}(u-l) &\leq \frac{\epsilon}{2} \\ \frac{4C}{1-r}\sqrt{\frac{\log m}{m}}\left(\frac{1}{1-\nu} + k/2\right) &\leq \frac{\epsilon}{2}.\end{aligned}$$

Thus, it suffices to take $k = \log(\epsilon^{-1}2(u-l))/\log\nu$. Substituting this value of k into the second inequality, we see that it suffices to take $m = O(\epsilon^2(\log\epsilon^{-1})^3)$. □

We now concern ourselves with R-superlinear convergence to the true optimal solution. Henceforth, we assume that the sample sizes grow so that m_k^{-1} converges R-superlinearly to 0.

Lemma 4.5.13. *We have $\sum_{k=0}^{\infty}\omega(\eta_k) < \infty$ with probability 1. In particular, $\eta_k \to 0$ almost surely.*

Proof. By Theorem 4.4.1, we find that

$$\begin{aligned}F_k(x_{k+1}) &\leq F_k(x_k) - \omega(\eta_k) \\ &= F_{k-1}(x_k) + (F_k(x_k) - F_{k-1}(x_k)) - \omega(\eta_k) \\ &\leq F_0(x_0) + \sum_{j=1}^{k}(F_j(x_j) - F_{j-1}(x_j)) - \sum_{j=0}^{k}\omega(\eta_j) \\ &\leq F_0(x_0) + \sum_{j=1}^{k}\sup_{x\in\mathcal{D}}|F_j(x) - F_{j-1}(x)| - \sum_{j=0}^{k}\omega(\eta_j) \\ &\leq F_0(x_0) + 2\sum_{j=1}^{k}\sup_{x\in\mathcal{D}}|F_j(x) - F(x)| - \sum_{j=0}^{k}\omega(\eta_j) \\ &\leq F_0(x_0) + 2\sum_{j=1}^{\infty}\sup_{x\in\mathcal{D}}|F_j(x) - F(x)| - \sum_{j=0}^{k}\omega(\eta_j).\end{aligned}$$

Let $Y = \sum_{j=1}^{\infty} \sup_{x \in \mathcal{D}} |F_j(x) - F(x)|$. By the monotone convergence theorem and Theorem 4.3.1, we have

$$\mathbb{E}Y = \sum_{j=1}^{\infty} \mathbb{E} \sup_{x \in \mathcal{D}} |F_j(x) - F(x)| \leq C \sum_{j=1}^{\infty} \sqrt{\frac{\log m_j}{m_j}}.$$

By our choice of m_j, the latter sum is finite. This implies that $\mathbb{P}(Y < \infty) = 1$. Since $F_k(x)$ is bounded below on $\mathcal{D}$ by Assumption 3, we necessarily have $\sum_{k=0}^{\infty} \omega(\eta_k) < \infty$ whenever $Y < \infty$. Thus $\eta_k \to 0$ with probability 1. □

Theorem 4.5.14. *Fix any $\beta < 1$. With probability 1, there exists a finite index k_0 such that the Wolfe condition is satisfied for all $k \geq k_0$.*

Proof. This follows from Theorem 6.3 in [12], for any realization of the empirical objective functions $F_0, F_1, \ldots$ such that $\eta_k \to 0$. By Lemma 4.5.13, the event $\eta_k \to 0$ occurs with probability 1. □

In particular, this implies that with probability 1, there exists a finite time k_0 after which every step is a BFGS step, and BFGS updates are always performed.

Corollary 4.5.15. *With probability 1, we have $\sum_{k=0}^{\infty} \|x_k - x^*\| < \infty$.*

Proof. This follows from Theorem 4.5.11. Let $\{x_k\}_{k=0}^{\infty}$ be any instance of the algorithm where $F(x_k) \leq F(x^*) + \varphi^{k/2}$ for all $k \geq k_0$, for some index k_0. Since $F(x)$ is strongly convex,

$$\|x_k - x^*\| \leq \frac{2}{\ell}(F(x_k) - F(x^*)) \leq \frac{2}{\ell}\varphi^{k/2}$$

for all $k \geq k_0$. Hence $\sum_{k=0}^{\infty} \|x_k - x^*\| < \infty$. By Theorem 4.5.11, this occurs with probability 1. □

Let us define $e_k = \max\{\|x_k - x^*\|, \|x_{k+1} - x^*\|\}$. Corollary 4.5.15 implies that $\sum_{k=0}^{\infty} e_k < \infty$.

Next, we perform a detailed analysis of the evolution of H_{k+1}. By applying Corollary 4.3.2, we can use a modified form of the classical argument ([47]) on a path-by-path basis.

Corollary 4.5.16. *Let $\sigma_k = m_k^{-2/5}$. By taking $\delta = \sigma_k$ in Corollary 4.3.2, we can find global constants c_4 and $\omega < 1$ such that*

$$\mathbb{P}(\sup_{x\in\mathcal{D}} \|G_k(x) - G(x)\| > \sigma_k \text{ or } \sup_{x\in\mathcal{D}} \|g_k(x) - g(x)\| > \sigma_k) \leq c_4\omega^k.$$

Hence, with probability 1, there exists an index k_0 such that for all $k \geq k_0$, we have both $\sup_{x\in\mathcal{D}} \|G_k(x) - G(x)\| < \sigma_k$ and $\sup_{x\in\mathcal{D}} \|g_k(x) - g(x)\| < \sigma_k$.

By construction, $\{\sigma_k\}$ converges to 0 at a R-superlinear rate.

Proof. The first part follows by Corollary 4.3.2. Taking $\epsilon = \frac{\delta}{2L+1}$, our probability bound is

$$\begin{aligned}&\mathbb{P}(\sup_{x\in\mathcal{D}} \|G_k(x) - G(x)\| > \sigma_k \text{ or } \sup_{x\in\mathcal{D}} \|g_k(x) - g(x)\| > \sigma_k)\\ &\leq C_1 \exp(\frac{2}{5}n\log m_k - C_2(1 - \frac{C_3}{2L+1})^2 m_k^{1/5}).\end{aligned}$$

Since $\frac{m_k^{1/5}}{\log m_k} \to \infty$ and $m_k = \Omega(k^5)$ by construction, we can find the desired $\omega < 1$. The second statement then follows immediately from the Borel-Cantelli Lemma. □

Let Ω denote the space of paths where $\sum_{k=0}^{\infty} e_k < \infty$ and for some k_0, $\sup_{x\in\mathcal{D}} \|G_k(x) - G(x)\| \leq \sigma_k$ and $\sup_{x\in\mathcal{D}} \|g_k(x) - g(x)\| \leq \sigma_k$ for all $k \geq k_0$. By Corollary 4.5.15 and Corollary 4.5.16, $\mathbb{P}(\Omega) = 1$. Henceforth, we restrict our analysis to the paths belonging to Ω.

The BFGS algorithm is invariant under a linear change of variables, so without loss of generality, we may assume that $G(x^*) = I$. This corresponds to the change of variables $\widetilde{F}(y) = F(G(x^*)^{-1/2}y)$, $y = G(x^*)^{1/2}x$. Define two 'hypothetical' updates:

$$\begin{aligned}\widehat{B}_{k+1} &= B_k - \frac{B_k s_k s_k^T B_k}{s_k^T B_k s_k} + \frac{G_k(x^*)s_k s_k^T G_k(x^*)}{s_k^T G_k(x^*) s_k}\\ \widetilde{B}_{k+1} &= B_k - \frac{B_k s_k s_k^T B_k}{s_k^T B_k s_k} + \frac{G(x^*)s_k s_k^T G(x^*)}{s_k^T G(x^*) s_k}.\end{aligned}$$

Lemma 4.5.17. *We have*

$$\|\widetilde{B}_{k+1} - I\|_F^2 \leq \|B_k - I\|_F^2$$

and

$$\|\widetilde{H}_{k+1} - I\|_F^2 \leq \|H_k - I\|_F^2.$$

Proof. For brevity, we write $s = s_k, B = B_k, H = H_k$. By a routine calculation (see §4 of [47]), we have

$$\|\widetilde{B}_{k+1} - I\|_F^2 - \|B_{k+1} - I\|_F^2 = -\left[\left(1 - \frac{s^T B^2 s}{s^T B s}\right)^2 + 2\left(\frac{s^T B^3 s}{s^T B s} - \left(\frac{s^T B^2 s}{s^T B s}\right)^2\right)\right]$$

and

$$\|\widetilde{H}_{k+1} - I\|_F^2 - \|H_{k+1} - I\|_F^2 = -\left[\left(1 - \frac{s^T H s}{s^T s}\right)^2 + 2\left(\frac{s^T H^2 s}{s^T s} - \left(\frac{s^T H s}{s^T s}\right)^2\right)\right].$$

The Cauchy-Schwarz inequality implies that the latter terms in the brackets are non-positive, which gives the desired result. □

Lemma 4.5.18. *Every path in* Ω *satisfies*

$$\|B_{k+1} - \widetilde{B}_{k+1}\| \leq O(e_k + \sigma_k)$$

and

$$\|H_{k+1} - \widetilde{H}_{k+1}\| \leq (\|H_k - I\| + 1)O(e_k + \sigma_k).$$

Proof. We again write $s = s_k, y = y_k, B = B_k, H = H_k$ for brevity.

We can bound the difference $\|B_{k+1} - \widehat{B}_{k+1}\|$, as both updates are performed with sampled gradients, and then use Corollary 4.5.16 to bound $\|\widehat{B}_{k+1} - \widetilde{B}_{k+1}\}$.

Take $\Delta = G_k(x^*)s - y$. By Lemma 4.5.6, we can write $y = G_k(\widehat{x})s$ for some $\widehat{x}$ on the line segment $[x_k, x_{k+1}]$, and we deduce that:

1. $\ell\|s\|^2 \leq y^T s \leq L\|s\|^2$
2. $\|\Delta\| \leq L_H e_k \|s\|$.

3. $\frac{y^T\Delta}{s^Ty} \leq LL_He_k$

Hence, writing $\frac{1}{s^Ty+\Delta^Ts} = \frac{1}{s^Ty} - \frac{y^T\Delta}{s^Ty+y^T\Delta}$, we have

$$\begin{aligned}\|B_{k+1} - \widehat{B}_{k+1}\| &= \left\|\frac{yy^T}{s^Ty} - \frac{(y+\Delta)(y+\Delta)^T}{(y+\Delta)^Ts}\right\| \\ &= \left\|-\frac{y\Delta^T + \Delta y^T + \Delta\Delta^T}{s^Ty} + \frac{y^T\Delta(yy^T + y\Delta^T + \Delta y^T + \Delta\Delta^T)}{s^Ty + y^T\Delta}\right\| \\ &\leq O(e_k).\end{aligned}$$

Next, write $\widehat{y} = G_k(x^*)s$ and $\widetilde{y} = G(x^*)s$. Since our path lies in Ω, we know that $\|G_k(x^*) - G(x^*)\| \leq \sigma_k$. Let $\Delta = \widehat{y} - \widetilde{y}$, so $\|\Delta\| \leq \sigma_k\|s\|$, and perform the same calculation as above to obtain

$$\begin{aligned}\|\widehat{B}_{k+1} - \widetilde{B}_{k+1}\| &= \left\|-\frac{\widetilde{y}\Delta^T + \Delta\widetilde{y}^T + \Delta\Delta^T}{s^T\widetilde{y}} + \frac{\widetilde{y}^T\Delta(\widetilde{y}\widetilde{y}^T + \widetilde{y}\Delta^T + \Delta\widetilde{y}^T + \Delta\Delta^T)}{s^T\widetilde{y} + \widetilde{y}^T\Delta}\right\| \\ &\leq O(\sigma_k).\end{aligned}$$

Hence, $\|B_{k+1} - \widetilde{B}_{k+1}\| \leq O(e_k + \sigma_k)$.

A similar calculation holds for H.

$$\begin{aligned}\|H_{k+1} - \widehat{H}_{k+1}\| = \|&\frac{ss^T}{(y+\Delta)^Ts} - \frac{ss^T}{s^Ty} \\ &+ \left(\frac{s(y+\Delta)^T}{(y+\Delta)^Ts} - \frac{sy^T}{s^Ty}\right)H + H\left(\frac{(y+\Delta)s^T}{(y+\Delta)^Ts} - \frac{ys^T}{s^Ty}\right) \\ &+ \frac{s(y+\Delta)^TH(y+\Delta)s^T}{((y+\Delta)^Ts)^2} - \frac{sy^THys^T}{(s^Ty)^2}\|.\end{aligned}$$

It is elementary, though tedious, to verify that $\frac{ss^T}{(y+\Delta)^Ts} - \frac{ss^T}{s^Ty} \leq O(e_k)$ and that the other terms are bounded by $O(\|H\|e_k)$. The same calculation shows that $\|\widehat{H}_{k+1} - \widetilde{H}_{k+1}\| \leq O(\sigma_k + \|H\|\sigma_k)$. Thus, we have $\|H_{k+1} - \widetilde{H}_{k+1}\| \leq (\|H_k - I\| + 1)O(e_k + \sigma_k)$. □

Corollary 4.5.19. *By Lemma 4.5.18, Lemma 4.5.17, and the triangle inequality,*

$$\|B_{k+1} - I\| \leq \|B_{k+1} - \widetilde{B}_{k+1}\| + \|\widetilde{B}_{k+1} - I\| \leq \|B_k - I\| + O(e_k + \sigma_k)$$

and

$$\|H_{k+1} - I\| \leq \|H_{k+1} - \widetilde{H}_{k+1}\| + \|\widetilde{H}_{k+1} - I\| \leq (\|H_k - I\| + 1)O(e_k + \sigma_k).$$

A lemma of Griewank and Toint shows that this forces the convergence of $\{\|B_k - I\|\}$ and $\{\|H_k - I\|\}$.

Lemma 4.5.20 (Lemma 3.3 of [47])**.** *Let $\{\phi_k\}$ and $\{\delta_k\}$ be sequences of non-negative numbers such that $\phi_{k+1} \leq (1 + \delta_k)\phi_k + \delta_k$ and $\sum_{k=1}^{\infty} \delta_k < \infty$. Then $\{\phi_k\}$ converges.*

In our case, we take $\delta_k = e_k + \sigma_k$, as $\sum_{k=0}^{\infty}(e_k + \sigma_k) < \infty$ by Corollary 4.5.15 and Corollary 4.5.16.

Following §4 of [47], our previous results yield the Dennis-Moré ([48]) condition:

$$\lim_{k\to\infty} \frac{\|(B_k - I)s_k\|}{\|s_k\|} = 0.$$

It only remains to show that this implies R-superlinear convergence in the stochastic setting.

Since $I = G(x^*)$, we have

$$\begin{aligned}
&\|B_k s_k - G(x^*)s_k\| \\
&= \| - g_k - G(x^*)s_k + g_k(x_{k+1}) - g_k(x_{k+1})\| \\
&= \|g_k(x_{k+1}) - g_k - G(x^*)s_k - g_k(x_{k+1})\| \\
&= \| \int_0^1 (G_k(x_k + \tau s_k) - G(x^*))s_k d\tau - g_k(x_{k+1})\| \\
&= \| \int_0^1 (G(x_k + \tau s_k) - G(x^*))s_k d\tau + \int_0^1 (G_k(x + \tau s_k) - G(x_k + \tau s_k))s_k d\tau - g_k(x_{k+1})\| \\
&\geq \|g_k(x_{k+1})\| - (L_H e_k + \sigma_k)\|s_k\|
\end{aligned}$$

and therefore $\frac{\|g_k(x_{k+1})\|}{\|s_k\|} \to 0$. By Assumption 1, the empirical objective function $F_k(x)$ is strongly convex, and therefore

$$\frac{\|g_k(x_{k+1})\|}{\|s_k\|} \geq \frac{|\|g_k(x_{k+1}) - g_k(x^*)\| - \|g_k(x^*) - g(x^*)\||}{\|x_{k+1} - x^*\| + \|x_k - x^*\|}. \tag{4.8}$$

To complete the analysis, let $a_k = \frac{\|g_{k+1}\|}{\|s_k\|}, b_k = \|g_k(x^*) - g(x^*)\|$, and $z_k = \|x_k - x^*\|$. Our above results show that $a_k \to 0$, and $b_k \leq \sigma_k$ tends to 0 R-superlinearly. For convenience, we assume without loss of generality that $\{b_k\}$ converges Q-superlinearly, by replacing $\{b_k\}$ by the Q-superlinear sequence bounding σ_k if necessary.

Rearrange inequality (4.8) to obtain

$$\ell z_{k+1} = \ell\|x_{k+1} - x^*\| \leq \|g_k(x_{k+1}) - g_k(x^*)\| \leq a_k(z_{k+1} + z_k) + b_k.$$

Eventually, $a_k < \frac{1}{2}\ell$, as $a_k \to 0$. Beyond that point, we find that

$$z_{k+1} \leq \frac{a_k}{\ell - a_k} z_k + b_k \leq \frac{2}{\ell} a_k z_k + b_k. \tag{4.9}$$

Let $c_k = \max\{a_k z_k, b_k\}$. Clearly $z_{k+1} \leq (2 + \frac{2}{\ell})c_k$, so it suffices to prove that $\{c_k\}$ converges

superlinearly. There are two cases to consider. If $c_{k+1} = a_{k+1}z_{k+1}$, then

$$\frac{c_{k+1}}{c_k} = \frac{a_{k+1}z_{k+1}}{c_k} \leq a_{k+1}\frac{(2+\frac{2}{\ell})c_k}{c_k} = \left(2+\frac{2}{\ell}\right)a_{k+1}$$

and $a_k \to 0$. Otherwise, if $c_{k+1} = b_{k+1}$, then

$$\frac{c_{k+1}}{c_k} = \frac{b_{k+1}}{c_k} \leq \frac{b_{k+1}}{b_k}$$

and by construction, $\{b_k\}$ converges to 0 superlinearly, so $\frac{b_{k+1}}{b_k} \to 0$.

This proves that z_k converges R-superlinearly, and completes the proof of Theorem 4.5.4.

Increasing m_k at this rate is clearly infeasible in reality, and it is non-trivial to determine a level of sampling which produces (iteration-wise) superlinear convergence in a reasonable amount of real time. However, the problem of choosing m_k is exactly analogous to choosing the batch size in SGD or SVRG, and similar heuristics can be used. In the early phase of the algorithm, there is less value in attempting to compute the gradient particularly precisely, and taking more steps with small batches/samples is efficient. As the algorithm approaches optimality, the sample size should increase, or else the objective value will simply fluctuate.

4.6 Numerical Experiments

We compared several implementations of SA-GD and SA-BFGS against the original SGD method on a penalized least squares problem with random design. That is, the objective function has the form

$$\min_{w\in\mathbb{R}^p} L(w) = \mathbb{E}(Y - X^T w)^2 + \frac{1}{2}\|w\|_2^2,$$

where $X \in \mathbb{R}^p$ and $Y \in \mathbb{R}$ are random variables with finite second moments. We base our model on a standard linear regression problem with Gaussian errors, so $Y = X^T\beta + \epsilon$ for a deterministic vector $\beta \in \mathbb{R}^p$ and $\epsilon \sim N(0,1)$ is a noise component. X was drawn according to a multivariate $N(0, \Sigma(\rho))$, where $\Sigma(\rho) = (1-\rho^2)I_p + \rho^2 J$ (here J is the all-ones matrix). By varying ρ, we

control the condition number of the expected Hessian. We tested problems of size $p = 100$ and $p = 500$.

The following six algorithms were tested:

SGD: SGD with fixed $m_k = p$ and diminishing step sizes $t_k = \frac{1}{k+1000}$ for problems with $p = 100$, and $t_k = \frac{1}{k+5000}$ for $p = 500$.

SA-GD: SA-GD with fixed $m_k = p$.

SA-GD-I: SA-GD with increasing samples $m_k = \frac{1}{2}p + 1.01^k$.

SA-BFGS: SA-BFGS with fixed $m_k = p$. We do not test for the Wolfe condition at each step, and instead always take the adaptive BFGS step and perform a BFGS update.

SA-BFGS-I: SA-BFGS with increasing samples $m_k = \frac{1}{2}p + 1.01^k$. We do not test for the Wolfe condition at each step, and instead always take the adaptive BFGS step and perform a BFGS update.

SA-BFGS-GD: SA-BFGS with increasing samples $m_k = \frac{1}{2}p + 1.01^k$. We test for the Wolfe condition and switch to taking a SA-GD step when the adaptive step t_k for SA-BFGS fails the test.

R-S-GD-C: Robust SGD with constant step size $t_k = \frac{1}{\sqrt{N}}$ where N is the total number of iteration that the algorithm will perform. At iteration N, output the average of the last $N/2$ solutions. For details, see [49].

R-S-GD-V: Robust SGD with diminishing step size $t_k = \frac{1}{\sqrt{k}}$. At iteration T, output the average of the last $T/2$ solutions. For details, see [49].

We also implemented the Streaming SVRG algorithm using parameter values specified in Corollary 4 in [50]. However, the performance of this algorithm is not comparable to the algorithms listed above. Therefore we didn't include it in the figures.

All algorithms whose “identifier" begin with "SA-" use the adaptive step size. For SGD, we followed the standard practice of using a diminishing and non-summable step size $t_k = \frac{a}{k+b}$. The values $a = 1, b = \{1000, 5000\}$ in our test were chosen experimentally.

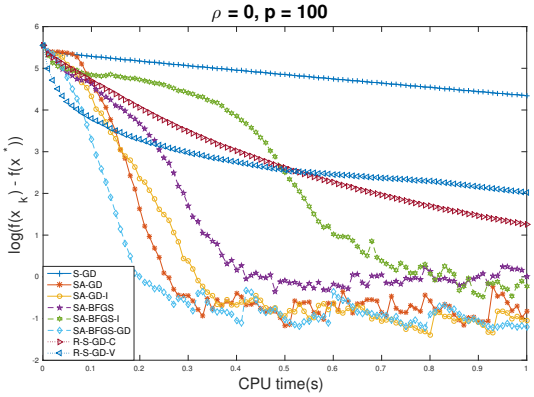
The SA-BFGS algorithms were implemented using a two-loop recursion to compute $H_k g_k$ when $p = 500$. This significantly improved their performance compared to storing the matrix H_k explicitly.

Figure 4.1 shows the performance of each algorithm on a series of problems with varying problem size p and parameter ρ. The y-axis measures the gap $\log(f(x(t)) - f(x^*))$ of the solution $x(t)$ obtained by the algorithm after using t seconds of CPU time. We used Matlab 2015a to implement the algorithms. The hardware was an Intel i5-5200U CPU running Ubuntu.

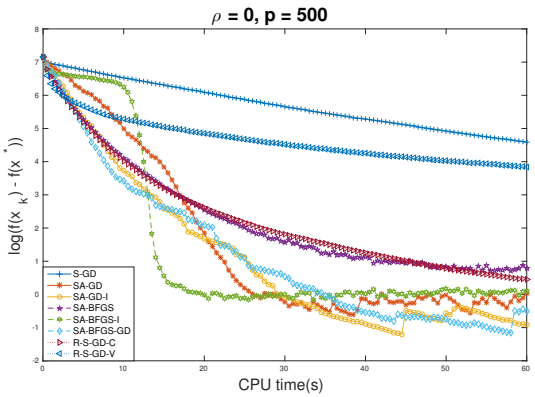
The increasing sample sizes used in SA-GD-I do not appear to reduce its variance, compared to SA-GD with m_k fixed, which goes against our initial expectations. In plots (a), (b), (e), and (f), SA-GD-I appears to exhibit the same fluctuations as SA-GD when the points approach optimality. In plot (c), SA-GD-I briefly surpasses SA-GD before the objective value jumps again. It is only in plot (d) that SA-GD-I appears to descend more consistently than SA-GD.

In contrast, SA-BFGS-I performed substantially better than SA-BFGS with m_k fixed. On the larger problem ($p = 500$), SA-BFGS-I rapidly approaches the optimal solution, until its progress slows. This suggests that the growth rate 1.01^k in sample sizes is too slow. This is not as apparent in the smaller problem $p = 100$, but a closer inspection reveals that all algorithms reach a comparable objective value after only 0.2s of CPU time.

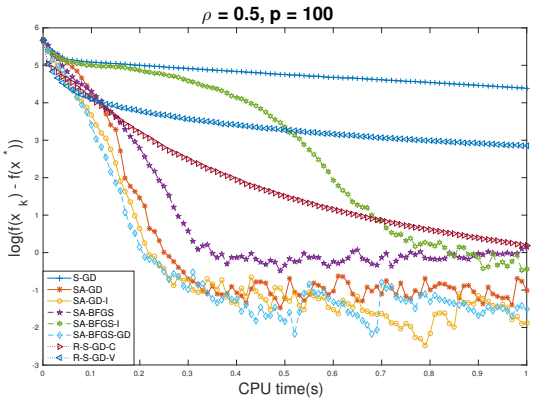
What is also interesting is that SA-GD often outperforms SA-BFGS if both algorithms use the same fixed sample size. While somewhat disappointing, there is a natural reason for this. BFGS optimizes a local *quadratic* model of the objective, and is performant when its approximation H_k resembles the true Hessian. The Hessian exhibits greater variance than the gradient, simply by virtue of having n^2 components compared to n, and we generally expect that more sampling is needed to accurately estimate the Hessian. With the same amount of sampling, SA-BFGS is therefore ‘noisier’ than SA-GD relative to the true function.



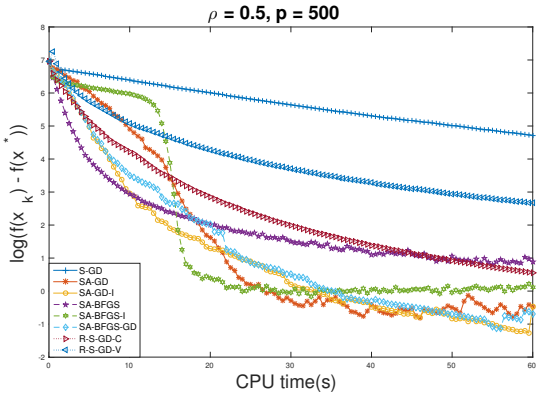
(a) $\rho = 0, p = 100$



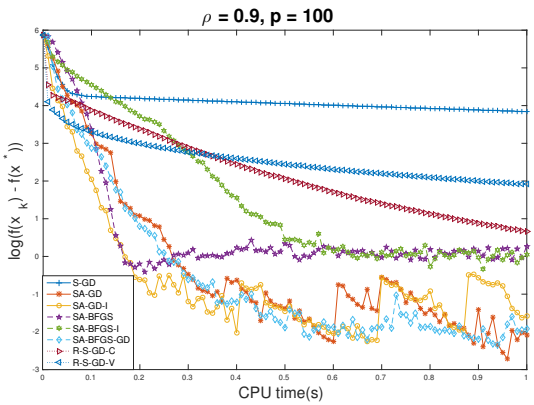
(b) $\rho = 0, p = 500$



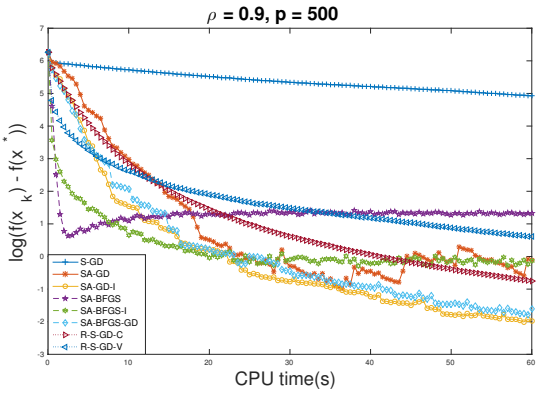
(c) $\rho = 0.5, p = 100$



(d) $\rho = 0.5, p = 500$



(e) $\rho = 0.9, p = 100$



(f) $\rho = 0.9, p = 500$

Figure 4.1: Experimental results for $p = 100, 500$ and varying ρ. The x-axis is the elapsed CPU time and the y-axis measures $\log(f(x) - f(x^*))$.

4.7 Conclusion and Future works

For self-concordant objective functions, adaptive methods eliminate the need to choose a step size, which is a big advantage. This is independent of a new difficulty which arises for *stochastic* methods, which is the choice of the sample size m_k. From our experiments, we see that choosing m_k appropriately is crucial. Unlike SGD and SVRG, where it is often most effective to use mini-batches of size 1, SA-BFGS and SA-GD work best with comparatively larger samples.

Note that SA-GD and SA-BFGS are not *purely* first-order methods, as computing the adaptive step size requires calculating the Hessian-vector product $G_k(x_k)d_k$. However, it is often possible to calculate Hessian-vector products efficiently, and at far less cost than computing the full Hessian $\nabla^2 F_k(x)$. Consider, for example, a a logistic regression problem where the empirical loss function is

$$L(x) = \frac{1}{m}\sum_{i=1}^{m} \log(1 + e^{-y_i a_i^T x})$$

for sampled data $\{(a_i, y_i) : a_i \in \mathbb{R}^n, y_i \in \{-1, 1\}\}$. Let A denote the $n \times m$ matrix with columns a_i. The gradient is given by $A^T\beta$ where β is the vector

$$\beta_i = \frac{-y_i e^{-y_i a_i^T x}}{1 + e^{-y_i a_i^T x}}$$

and the Hessian is given by ABA^T where B is the diagonal matrix with entries

$$B_{ii} = \frac{e^{-y_i a_i^T x}}{(1 + e^{-y_i a_i^T x})^2}$$

To compute the product $d^T\nabla^2 L(x)d$, it suffices to compute $\sum_{i=1}^{m} B_{ii}(a_i^T d)^2$, and this requires approximately the same number of arithmetic operations as computing $\nabla L(x)$. Thus, computing δ_k for the adaptive step size requires roughly the same amount of work as one additional gradient calculation. Also, as mentioned in Section 4.5, we may replace the BFGS update in SA-BFGS with a modified update using the Hessian action $y_k = G_k(x_k)d_k$ to save effort.

This work represents only a preliminary step in the development of stochastic quasi-Newton

methods. There are several key questions that remain open:

1. Theorem 4.5.4 partially resolves a question posed by Moritz et al. [39] by proving superlinear convergence under rather restrictive conditions. It would be strengthened greatly if we could prove superlinear convergence under weaker conditions on m_k.

2. The theory developed for adaptive step sizes only applies to self-concordant functions, but the adaptive step size itself can be interpreted for non-self-concordant functions, as an adjustment based on the local curvature. It would be of great interest to extend adaptive methods to general convex functions, thereby replacing both inexact line search and fixed step sizes on a large class of problems.

3. How can variance-reduction be applied to SA-GD and SA-BFGS? The control variates used in SVRG are effective, though costly, and perhaps difficult to compute for functions of the form $\mathbb{E}f(x, \xi)$. However, SA-GD and SA-BFGS could potentially be improved by incorporating some form of variance reduction, which would also allow us to reduce the number of samples needed on most iterations.

4. What heuristics can be developed for the sample sizes m_k to improve accuracy and speed up performance of SA-GD and SA-BFGS?

Chapter 5: Using Unbiased Simulation for Solving Stochastic Composition Optimization Problems

5.1 Introduction

In statistics and machine learning, we often encounter the generic stochastic optimization problem

$$\min_{x\in\mathcal{D}} F(x) \triangleq \mathbb{E}_v f_v(x), \tag{5.1}$$

where f_v is a convex function indexed by random variable v, $\mathbb{E}_v$ denotes expectation with respect to v, and $\mathcal{D} \subset \mathbb{R}^d$ is a compact convex set. A special case of (5.1) is the empirical risk minimization (ERM) problem when v is from the uniform random variable on $\{1, 2, \ldots, n\}$, that is,

$$\min_{x\in\mathcal{D}} F_n(x) \triangleq \frac{1}{n}\sum_{i=1}^{n} f_i(x). \tag{5.2}$$

When obtaining the full gradient is computationally intensive, a popular method for solving these problems is the (projected) stochastic gradient descent (SGD) algorithm, which can be described by the following update rule for $t = 1, 2, \ldots$

$$x_t = \Pi_{\mathcal{D}}\{x_{t-1} - \lambda_t \nabla f_{v_t}(x_{t-1})\}, \tag{5.3}$$

where v_t is sampled from the distribution of v for generic optimization problems and from the uniform distribution on $\{1, 2, ..., n\}$ for ERM problems, λ_t is the step size, and $\Pi_{\mathcal{D}}$ is the projection operator on to $\mathcal{D}$. It is well known that convergence of SGD requires a diminishing step size λ_t and thus results in a worse convergence rate than gradient descent algorithms. [6] observed that

the inferior rate of SGD is caused by the fact that stochastic gradients do not converge to 0 as the iterates converge to the optimal solution. Based on this observation, they improved the SGD by applying a control variate variance reduction technique to the stochastic gradient generation which is known as the SVRG algorithm. SVRG has been shown to converge linearly to the optimal solution for strongly convex ERM problems and performs well in practice. These algorithms implicitly assume that the gradient of each member function $f_v(\cdot)$ is easy to compute. But this assumption does not hold in the so-called stochastic composition optimization (SCO) problem [17]:

$$\min_{x\in\mathcal{D}} F(x) \triangleq \mathbb{E}_v f_v(\mathbb{E}_w g_w(x)), \tag{5.4}$$

where v and w are random variables with certain known joint distributions or in its finite sample version:

$$\min_{x\in\mathcal{D}} F_n(x) \triangleq \frac{1}{n}\sum_{i=1}^{n} f_i\{\frac{1}{m_i}\sum_{j=1}^{m_i} g_{ij}(x)\}. \tag{5.5}$$

Problems of this form arise in many areas such as reinforcement learning and risk-averse learning to graphical models, econometrics and survival analysis. As far as we know, all current algorithms that are used to solve SCO problems are based on *biased* stochastic gradient oracles. The convergence rates for these algorithms are unsatisfactory compared to the algorithms for solving generic stochastic optimization problems, except for the Comp-SVRG algorithms in [51]. These algorithms are also based on *biased* stochastic gradients, but the modified variance reduced gradients vanish as the iterates converge to the optimal solution. Therefore, linear convergence can be proved for the finite sum version of SCO when strong convexity is present. However, the number of samples that are needed to construct a variance reduced gradient depends on the condition number of the objective function. All these drawbacks are the result of biased stochastic gradients. If unbiased stochastic gradients can be generated for SCO problems, we can treat SCO problems in the same way that we treat generic stochastic optimization problems and apply SGD and its

variants to solve them.

5.1.1 Contributions

Our contributions in this section can be summarized as follows.

- We introduce unbiased gradient simulation algorithms that are based on a multilevel Monte Carlo technique for solving smooth SCO problems. We also show that the output of these algorithms has finite variance and its expected computational cost is finite.

- Based on our unbiased gradient simulation algorithms, a stochastic composition optimization problem can be considered as a generic stochastic optimization problem. This is because we can simply apply SGD to solve SCO problems and achieve the same iteration complexity as using SGD to solve generic stochastic optimization problems.

- We also show that our unbiased gradient simulation algorithm can be combined with variance reduction techniques including SVRG [6] and SCSG [52], yielding variance reduced optimization algorithms that converge linearly to the optimum of a SCO problem.

5.1.2 Related work

Using *biased* stochastic gradients, [17] proposed a generic algorithm for solving (5.4) with an iteration complexity of $O(\epsilon^{-3/2})$ for strongly convex objectives and $O(\epsilon^{-4})$ for general convex objectives. This result was improved to $O(\epsilon^{-5/4})$ for strongly convex objectives and $O(\epsilon^{-7/2})$ for general convex objectives in [53]. For strongly convex objectives with finite sum structure, ([51]) modified the SVRG algorithm and achieved a sample complexity $O((m+n)\log(1/\epsilon))$. Stochastic algorithms using biased gradient methods also appeared in [54] for non-convex SCOs.

We propose unbiased gradient simulation methods that are based on a multilevel Monte Carlo technique for solving smooth SCO problems. Unbiased simulation methods for functions of expectations using multilevel Monte Carlo techniques were developed in [55] and [56]. Such techniques

have been heavily used in simulation algorithms to solve problems that require high accuracy estimates such as stochastic differential equation [57, 58, 59], stochastic partial differential equations [60], and Markov Chains [61]. They also have been used to reduce computational cost through variance reduction techniques [62, 63, 64, 65].

We also consider variance reduced stochastic gradient algorithms that are based on unbiased gradient simulation. A number of variance reduction techniques have been proposed for strongly convex ERM problems in the literature including the use of control variates; see SVRG in ([6]) and SDCA in ([66]), incremental gradients in [67] and SAGA in [8], and importance sampling in [68]. The analysis of these methods and their variants can be found in [69, 70, 52, 71, 72].

The *iteration complexity* for current algorithms on smooth SCO is presented in Table 1. In particular, SimGD, SimVRG and SCSimG are proposed by us in this chapter. We report iteration complexity instead of sample complexity due to the special randomization component in the gradient estimator construction. This component is critical for our estimator to be unbiased, but the trade-off that it is difficult to analyze sample complexity. We will discuss related issues in detail in later sections.

Table 5.1: Iteration complexity of different algorithms for solving smooth SCO problems.

	Convex	Strongly Convex
Basic SCGD [17]	$O(1/\epsilon^4)$	$O(1/\epsilon^{3/2})$
Accelerating SCGD [53]	$O(1/\epsilon^{7/2})$	$O(1/\epsilon^{5/4})$
Compositional SVRG-1 [51]	N.A.	$O(\log(1/\epsilon))$
Compositional SVRG-2 [51]	N.A.	$O(\log(1/\epsilon))$
SimGD (our variant of SGD)	$O(1/\epsilon^2)$	$O(1/\epsilon)$
SimVRG (our variant of SVRG)	N.A.	$O(\log(1/\epsilon))$
SCSimG (our variant of SCSG)	N.A.	$O(\log(1/\epsilon))$

Both basic SCGD and accelerating SCGD make 2 sampling queries in every iteration, Compositional SVRG-1 and Compositional SVRG-2 make $\sum_{i=1}^{n} m_i$ and an additional constant number of sampling queries in every iteration. SimGD makes a random number of sampling queries in every iteration and the expectation of this random number is finite. SimSVRG makes $\sum_{i=1}^{n} m_i$ and

additional random number of sampling queries in every iteration and the the expectation of this random number is finite. SCSimG makes $\min\{\sum_{i=1}^{n} m_i, 1/\epsilon\}$ and an additional random number of sampling queries in every iteration and the the expectation of this random number is finite.

5.1.3 Organization

The rest of this chapter is organized as follows. In section 5.2, we describe the problem formulations and introduce the notation that we will use. We then present our unbiased gradient simulation algorithms and the optimization algorithms that are based on these unbiased simulations. In section 5.3, we give concrete examples of SCO problems that arise in a variety of areas and explain how our algorithms are well-suited to solve them. In section 5.4, we prove several important theoretical properties of our gradient simulation algorithm, in particular, its unbiasedness, finite variance and finite expected computational cost. We also show it has a certain "Lipschitz" property that enable it to be combined with variance reduction algorithms such as SVRG and SCSG. Finally, we prove the convergence properties of our algorithms. In section 5.5, we present numerical results obtained using our algorithms for maximizing Cox's partial likelihood and training conditional random fields.

5.2 Problem Description and Algorithms

5.2.1 Problem Description and Notation

Throughout this chapter, we consider the following smooth stochastic composition optimization problem (5.4). We define the support of the distributions v and w to be Ω_v and Ω_w. Note that the following two problems can be considered as special cases of (5.4); the first one is the finite sum problem:

$$\min_{x\in\mathcal{D}} F_n(x) \triangleq \frac{1}{n}\sum_{i=1}^{n} f_i\Big(\frac{1}{m_i}\sum_{j=1}^{m_i} g_{ij}(x)\Big), \tag{5.6}$$

and the second one is the mixed problem:

$$\min_{x\in\mathcal{D}} \frac{1}{n}\sum_{i=1}^{n} f_i(\mathbb{E}_w g_w(x)). \tag{5.7}$$

Later, we will discuss algorithms for these two special cases.

As for notation, for a vector $v \in \mathbb{R}^n$, we use $[v]_i$ to denote the i-th entry for $1 \le i \le n$ and use $\|v\|_p$ to denote its L_p-norm. For a matrix $A \in \mathbb{R}^{m\times n}$, we use $[A]_{ij}$, $[A]_{:j}$ and $[A]_{i:}$ to denote the (i,j)-th entry, j-th column and i-th row for every $1 \le i \le m$ and $1 \le j \le n$. We use $\|A\|_2$ and $\|A\|_F$ to denote its spectral norm and Frobenius norm, respectively. We use $\|A\|_\infty$ to denote the maximum absolute value of the entries of A, that is, $\|A\|_\infty = \max\{|[A]_{ij}| \mid 1 \le i \le m, 1 \le j \le n\}$. For a multi-linear map $B \in \mathbb{R}^{m\times n\times p}$, we use $[B]_{ijk} \in \mathbb{R}$ to denote its (i,j,k)-th entry, $[B]_{:jk} \in \mathbb{R}^m$,$[B]_{i:k} \in \mathbb{R}^n$, and $[B]_{ij:} \in \mathbb{R}^{1\times p}$ to denote its (j,k)-th column fiber,(i,k)-th row fiber, and (i,j)-th tube fiber, and $[B]_{::k} \in \mathbb{R}^{m\times n}$, $[B]_{:j:} \in \mathbb{R}^{m\times p}$ and $[B]_{i::} \in \mathbb{R}^{n\times p}$ to denote its k-th frontal slice, j-th lateral slice and i-th horizontal slice, where $1 \le i \le m$, $1 \le j \le n$ and $1 \le k \le p$. We define $\|B\|_\infty = \{|[B]_{ijk}| \mid 1 \le i \le m, 1 \le j \le n, 1 \le k \le p\}$. Moreover, we use $\mathrm{vec}(\cdot)$ to denote the vectorize operation for one matrix or a multi-linear map. When there are multiple arguments in vec(), it vectorize each component and stack them into another vector.

We write the Jacobian (with respect to x) of the vector valued $g_w(\cdot)$ as

$$\nabla g_w(x) = \begin{pmatrix} \frac{\partial [g_w]_1}{\partial [x]_1}(x) & \cdots & \frac{\partial [g_w]_1}{\partial [x]_p}(x) \\ \vdots & \ddots & \vdots \\ \frac{\partial [g_w]_d}{\partial [x]_1}(x) & \cdots & \frac{\partial [g_w]_d}{\partial [x]_p}(x) \end{pmatrix},$$

where

$$g_w(x) = ([g_w]_1(x), [g_w]_2(x), \ldots, [g_w]_d(x))^\top.$$

It then follows from the chain rule that the gradient (with respect of x) of $f_v(\cdot)$ for the stochastic

problem is $\{\mathbb{E}_w \nabla g_w(x)\} \nabla f_v \{\mathbb{E}_w g_w(x)\}$ and

$$\nabla F(x) = \{\mathbb{E}_w \nabla g_w(x)\}^\intercal \mathbb{E}_v \{\nabla f_v(\mathbb{E}_w g_w(x))\}. \tag{5.8}$$

We use $\nabla^2 g_w(x) \in \mathbb{R}^{d \times p \times p}$ to denote the Hessian (with respect to x) of the vector valued $g_w(\cdot)$ and use $\nabla^2 g_w(x)[u, v] \in \mathbb{R}^d$ to denote the vector produced by $\nabla^2 g_w(x)$ acting on $u, v \in \mathbb{R}^p$, that is,

$$[\nabla^2 g_w(x)[u, v]]_i = \sum_{j=1}^{p} \sum_{k=1}^{p} [\nabla^2 g_w(x)]_{ijk} [u]_j [v]_k = \sum_{j=1}^{p} \sum_{k=1}^{p} [\nabla^2 [g_w]_i(x)]_{jk} [u]_j [v]_k.$$

Finally, we introduce the following notation used in our gradient simulation algorithms. Let $I_n(v_1) = \{w_i\}_{i=1}^n$ be a collection of random variables that are i.i.d. generated from the distribution of w given $v = v_1$, where v and w are the random variables in problem (5.4). Given the samples $I_n(v_1)$, let

$$\begin{aligned}
\overline{g}(x; n_1, n_2) &= \frac{1}{n_2 - n_1 + 1} \sum_{i=n_1}^{n_2} g_{w_i}(x), \\
\overline{\nabla g}(x; n_1, n_2) &= \frac{1}{n_2 - n_1 + 1} \sum_{i=n_1}^{n_2} \nabla g_{w_i}(x), \quad \text{and} \\
\overline{\nabla^2 g}(x; n_1, n_2) &= \frac{1}{n_2 - n_1 + 1} \sum_{i=n_1}^{n_2} \nabla^2 g_{w_i}(x),
\end{aligned}$$

for $x \in \mathcal{D} \subset \mathbb{R}^p$ and $1 \leq n_1 \leq n_2 \leq n$. These quantities are *unbiased* estimates of $\mathbb{E}_w g_w(x)$, $\mathbb{E}_w \nabla g_w(x)$ and $\mathbb{E}_w \nabla^2 g_w(x)$. In addition, let

$$\bar{y}(x; n_1, n_2) = \overline{\nabla g}(x; n_1, n_2)^\top \nabla f_v(\bar{g}(x; n_1, n_2)),$$

which is the gradient of $f_{v_1}(\bar{g}(x; n_1, n_2))$. This is an estimate of $\nabla\{\mathbb{E}_v f_v(\mathbb{E}_w g_w(x))\}$. However, it is a biased estimate, that is,

$$\mathbb{E}\bar{y}(x; n_1, n_2) \neq \nabla\big(\mathbb{E}_v f_v\{\mathbb{E}_w g_w(x)\}\big).$$

Since the samples are i.i.d., the expectation of $\bar{y}(x; n_1, n_2)$ only depends on the distribution of w conditioned on $v = v_1$, and the number of samples that are used to construct $\bar{y}(x; n_1, n_2)$, we write

$$s(x; n_2 - n_1 + 1, v_1) = \mathbb{E}\{\bar{y}(x; n_1, n_2)|v = v_1\}.$$

We also let

$$\begin{aligned}[\bar{z}(x; n_1; n_2)]_i = &\{[\overline{\nabla^2 g}(x; n_1, n_2)]_{::i}\}^\top \nabla f_{v_1}\{\bar{g}(x; n_1, n_2)\} \\ &+ \{\overline{\nabla g}(x; n_1, n_2)\}^\top \nabla^2 f_{v_1}\{\bar{g}(x; n_1, n_2)\}[\overline{\nabla g}(x; n_1, n_2)]_{:i},\end{aligned}$$

which is the i-th row of the Hessian of $f_{v_1}(\bar{g}(x; n_1, n_2))$ for $1 \le i \le p$. Similarly, it is also a *biased* estimate of $\nabla^2(\mathbb{E}_v f_v\{\mathbb{E}_w g_w(x)\})$.

5.2.2 Unbiased Stochastic Gradient Simulation

We first present Algorithm 8 to simulate unbiased gradients for the stochastic problems (5.4) and (5.7) while fixing a component v_1 for $f_{v_1}(\mathbb{E}_w g_w(x))$. It can be considered as a variant of [56] based on a multilevel randomization technique.

Algorithm 8 UnbiasedGradient(x, v_1, n_0, γ)

Input: $x \in \mathcal{D}$,$v_1 \in \Omega_v$, base level $n_0 \ge 0 \in \mathbb{Z}$, rate parameter $1 < \gamma < 2$.
Output: $G(x, v_1) \in \mathbb{R}^p$, an unbiased estimate of the gradient of $f_{v_1}(\mathbb{E}_w g_w(x))$ at point x and component v_1.
Sample N from a geometric distribution with success probability $1 - p$ where $p = 0.5^\gamma$.
Independently sample $I_{2^{n_0+N+1}}(v_1) = \{w_i\}_{i=1}^{2^{n_0+N+1}}$ from the distribution of w given v_1.
Compute $Y_1(x) = \bar{y}(x; 1, 2^{n_0+N+1})$.
Compute $Y_2(x) = \bar{y}(x; 1, 2^{n_0+N})$.
Compute $Y_3(x) = \bar{y}(x; 2^{n_0+N} + 1, 2^{n_0+N+1})$.
Compute $Y_4(x) = \bar{y}(x; 1, 2^{n_0})$.
Compute $G(x, v_1) = \frac{Y_1(x) - 0.5(Y_2(x)+Y_3(x))}{\tilde{p}_N} + Y_4(x)$, where $\tilde{p}_N = (1 - p)p^N$.
Output: $G(x, v_1)$

We shall prove in Section 4 that the output of Algorithm 8 is indeed an unbiased estimate of $f_{v_1}(\mathbb{E}_w g_w(x))$ for fixed v_1. It follows that if we sample $v_1 \sim v$, then $G(x, v_1)$ would be an unibased

estimate of the gradient of $\mathbb{E}_v f_v(\mathbb{E}_w g_w(x))$.

Remark: We note that Algorithm 8 requires conditional sampling of w given v. It is difficult to obtain such samples in a very general setting. However, in many applications, obtaining such samples can be relatively easy. We will discuss this in detail in Section 5.3. Moreover, Algorithm 8 uses a random number of samples to construct an unbiased estimate. We will show later that the number of samples needed is finite in expectation and independent of the problem sample size. However, for problems such as (5.6), computing an unbiased estimate using this algorithm may need the same number of samples as computing the true gradient in a worst case scenario.

5.2.3 Optimization Algorithms

We now present our optimization algorithms to solve problem (5.4), (5.7) and (5.6) based on unbiased gradient simulation. First, in Algorithm 9, we present our SGD (SimGD) algorithm with a simple averaging techinique (see [73]). Convergence of our SimGD algorithm under different conditions will be analyzed in Section 5.4. It is worth noting that our SGD algorithm is an analogue of the standard stochastic gradient descent algorithm that substitutes simulated unbiased gradients for sampled stochastic gradients. Therefore, the unbiased gradient simulation algorithm enables us to solve SCO problems in the same way as generic stochastic optimization problems.

Algorithm 9 Simulated Gradient Descent (SimGD)

Input: Number of iterations T, step size $\{\lambda_t\}_{t=1}^{\infty}$, initial point x_0, base level n_0 and rate parameter $1 < \gamma < 2$.
for $t = 0, 1, 2, \ldots, T-1$ **do**
  Sample v_t follows the distribution of v and let $\rho_t = \text{UnbiasedGradient}(x_t, v_t, n_0, \gamma)$
  $x_{t+1} = \Pi_{\mathcal{D}}(x_t - \lambda_t \rho_t)$
end for
option I Output $\tilde{x}_T = \frac{2}{(T)(T+1)} \sum_{t=0}^{T-1} (t+1) x_t$
option II Output x_T

In contrast to SGD, where a diminishing step size is used, we also introduce an SVRG type of control variate variance reduced algorithm as mentioned in [6] with constant step size for SCO problems. As described in [6] for ERM problems (5.2) and in [50] for generic stochastic optimiza-

tion problems (5.1), a variance reduced stochastic gradient at point x with respect to the reference point $\tilde{x}$ is defined as $\nabla f_{v'}(x) - \nabla f_{v'}(\tilde{x}) + \nabla F(\tilde{x})$ where v' is sampled from v for the generic stochastic optimization problem (5.1) and defined similarly for the ERM problem. We adopt these variance reduction techniques in our setting of unbiased gradient simulation. Specifically, we will simulate the unbiased gradients at x and $\tilde{x}$ simultaneously, using the same set of simulated data, to reduce variance. The details of generating such variance reduced gradients are specified in Algorithm 10. For ease of presentation, Algorithm 10 is built on the setting of Algorithm 8 and it can be modified by using Algorithm 9 for solving problem (5.6).

Algorithm 10 SimulatedGradient(x, $\tilde{x}$, $G(\tilde{x})$, v_1, n_0, γ)

Input: $x \in \mathbb{R}^d$, $v_1 \in \Omega_v$, reference point $\tilde{x} \in \mathbb{R}^d$, an estimate of gradient at point $\tilde{x}$ $\hat{G}(\tilde{x}) \in \mathbb{R}^p$, base level $n_0 \geq 0$ and rate parameter $1 < \gamma < 2$.
Output: $W \in \mathbb{R}^p$, a variance reduced unbiased estimator of the gradient of $\mathbb{E}_v f(\mathbb{E}_w g_w(x), v)$ at point x.
Sample N from a geometric distribution with success rate $1 - p$, where $p = 0.5^\gamma$.
Compute $\tilde{p}_N = (1-p)p^N$.
Independently sample $I_{2^{n_0+N+1}}(v_1) = \{w_i\}_{i=1}^{2^{n_0+N+1}}$ from the conditional distribution of w, given $v = v_1$.
Compute $Y_1(x) = \bar{y}(x; 1, 2^{n_0+N+1})$ and $Y_1(\tilde{x}) = \bar{y}(\tilde{x}; 1, 2^{n_0+N+1})$.
Compute $Y_2(x) = \bar{y}(x; 1, 2^{n_0+N})$ and $Y_2(\tilde{x}) = \bar{y}(\tilde{x}; 1, 2^{n_0+N})$.
Compute $Y_3(x) = \bar{y}(x; 2^{n_0+N}+1, 2^{n_0+N+1})$ and $Y_3(\tilde{x}) = \bar{y}(\tilde{x}; 2^{n_0+N}+1, 2^{n_0+N+1})$.
Compute $Y_4(x) = \bar{y}(x; 1, 2^{n_0})$ and $Y_4(\tilde{x}) = \bar{y}(\tilde{x}; 1, 2^{n_0})$.
Compute $W(x, v_1) = \frac{Y_1(x) - 0.5\{Y_2(x)+Y_3(x)\}}{\tilde{p}_N} + Y_4(x)$.
Compute $W(\tilde{x}, v_1) = \frac{Y_1(\tilde{x}) - 0.5\{Y_2(\tilde{x})+Y_3(\tilde{x})\}}{\tilde{p}_N} + Y_4(\tilde{x})$.
Set $W(x, \tilde{x}, v_1) = W(x, v_1) - W(\tilde{x}, v_1) + \hat{G}(\tilde{x})$.
Output: $W(x, \tilde{x}, v_1)$.

In Algorithm 10, the reference gradient $G(\tilde{x})$ can either be the full gradient at $\nabla F(\tilde{x})$ or an estimate of the full gradient $\nabla F(\tilde{x})$. For example, when it is efficient to compute full gradients of the objective function for problem (5.4) and (5.7), we propose to use the following method in Algorithm 11 to solve this problem. Since it can be considered as a variant of SVRG, we refer to it as Simulated Variance Reduced Gradient Descent.

However, when the full gradient $\nabla F(\tilde{x})$ of the objective function (5.4) is difficult to compute, we estimate $\nabla F(\tilde{x})$ by sampling the unbiased gradient within a batch of the indices and take the

Algorithm 11 Simulated Variance Reduced Gradient Descent(SimVRG)

Inputs: Number of epochs T, number of steps in each epoch M, step size λ and initial point $\tilde{x}_0$, base level $n_0 \geq 0$, and parameter $1 < \gamma < 2$.
for $s = 0, 1, 2, \ldots, T-1$ **do**
  Compute the full gradient $\nabla F(\tilde{x}_s)$
  $x_0 = \tilde{x}_s$
  for $t = 0, 1, 2, \ldots, M-1$ **do**
    Sample v_t from the distribution of v.
    Compute $\rho_t = \text{SimulatedGradient}(x_t, \tilde{x}_s, \hat{G}(x_s), v_t, n_0, \gamma)$.
    Update $x_{t+1} = \Pi_{\mathcal{D}}(x_t - \lambda \rho_t)$.
  end for
  option I Output $\tilde{x}_{s+1} = x_M$
  option II Output $\tilde{x}_{s+1} = x_t$ for randomly chosen $t \in \{1, ..., M\}$
end for

average. This method is related to another variant of SVRG, namely SCSG in [52] and we present this approach in Algorithm 12. Convergence properties of Algorithm 11 and Algorithm 12 will be analyzed in Section 5.4.

Algorithm 12 Stochastically Controlled Simulated Gradient Descent(SCSimG)

Inputs: Number of epochs T, number of steps in each epoch M, batch size B, sample size K, step size λ, initial point $\tilde{x}_0$, base level $n_0 \geq 0$ and parameter $1 < \gamma < 2$.
for $s = 0, 1, \ldots, T-1$ **do**
  $x_0 = \tilde{x}_s$
  Uniformly sample a batch $\mathcal{I}_s \subset \Omega_v$ according to the distribution of v with $|\mathcal{I}_s| = B$
  for $k = 1, 2, \ldots, K$ **do**
    Compute $h_k(\tilde{x}_s) = \frac{1}{B} \sum_{v_i \in \mathcal{I}_s} \text{UnbiasedGradient}(\tilde{x}_s, v_i, n_0, \gamma)$
  end for
  Compute $\tilde{h}(\tilde{x}_s) = \frac{1}{K} \sum_{i=1}^{K} h_i(\tilde{x}_s)$
  for $t = 0, 1, \ldots, M-1$ **do**
    Sample v_t from the distribution of v.
    Set $\rho_t = \text{SimulatedGradient}(x_t, \tilde{x}_s, \tilde{h}(\tilde{x}_s), v_t, n_0, \gamma)$.
    Update $x_{t+1} = \Pi_{\mathcal{D}}(x_t - \lambda \rho_t)$.
  end for
  option I Output $\tilde{x}_s = x_M$
  option II Output $\tilde{x}_s = x_t$ for randomly chosen $t \in \{1, ..., M\}$
end for

5.3 Examples

We now present some important examples that can be formulated as SCO problems.

5.3.1 Conditional Random Fields (CRF)

Conditional random fields (CRF) [74] is a popular probabilistic model used for structural prediction. It has been used in a number of natural language processing (NLP) problems including part-of-speech tagging [74], noun-phrase chunking [75, 76], named identity recognition [77] and image segmentation in computer vision [78]. In the CRF models, the conditional probability of a structured outcome $y \in \mathcal{Y}$, given an observation $x \in \mathcal{X}$ is:

$$p(y \mid z; x) = \frac{\exp\{x^\top F(z, y)\}}{\sum_{y' \in \mathcal{Y}} \exp\{x^\top F(z, y')\}}, \tag{5.9}$$

where $x \in \mathbb{R}^p$ is the parameter for estimation and $F(z, y) \in \mathbb{R}^p$ is a vector of pre-specified feature functions depending on the underlying structure of $\mathcal{Y}$. Based on the set of training data $\{(z_i, y_i), i = 1, \ldots, n\}$, the parameter x can be estimated by maximizing the log likelihood function

$$\max_{x \in \mathbb{R}^p} \frac{1}{n} \sum_{i=1}^{n} \log p(y_i \mid z_i, x). \tag{5.10}$$

As we shall see, the practical difficulty of computing the objective function value or its gradient lies in the exponential cardinality of $\mathcal{Y}$. The hardness of computing log-likelihood and gradients for CRFs has been considered in [79] and [80]. When the underlying structure of $\mathcal{Y}$ is a linear chain or a tree, both the objective function value and the gradient can be efficiently computed through dynamic programming (the Viterbi algorithm in [81]). For these structural cases, a number of methods can be used to solve (5.10); for example, deterministic methods such as the iterative scaling algorithm in [74] , L-BFGS in [76], stochastic methods such as stochastic gradient descent in [82] and SAG in [83]. However, when the underlying structure is more general (no linear chain or tree structure), computing a full gradient or even a stochastic gradient for problem (5.10) is difficult due to the exponential cardinality of $\mathcal{Y}$. In our setting, we can formulate (5.10) as a

composition optimization problem as in (5.4) by noticing that (5.10) is equivalent to

$$\min_x \frac{1}{n}\sum_{i=1}^{n}\big(\log\big[\sum_{y'\in\mathcal{Y}}\exp\{x^\top F(z_i,y')\}\big]-x^\top F(z_i,y_i)\big), \tag{5.11}$$

whose gradient can be written as

$$\frac{1}{n}\sum_{i=1}^{n}\Big[\frac{\sum_{y'\in\mathcal{Y}}\exp\{x^\top F(z_i,y')\}F(z_i,y')}{\sum_{y'\in\mathcal{Y}}\exp\{x^\top F(z_i,y')\}}-F(z_i,y_i)\Big].$$

Note that this problem is equivalent to

$$\min_x \frac{1}{n}\sum_{i=1}^{n}\big(\log\big[\frac{1}{|\mathcal{Y}|}\sum_{y'\in\mathcal{Y}}\exp\{x^\top F(z_i,y')\}\big]-x^\top F(z_i,y_i)+\log|\mathcal{Y}|\big).$$

Therefore we can view it as a form of problem (5.4) and apply our optimization algorithms to solve (5.11).

To obtain a sample y' uniformly from $\mathcal{Y}$, we first let (V,E) be the underlying graph of the CRF. We assume that each vertex $v\in V$ takes value from $\{1,2,\ldots,K\}$. Under this setting, we can generate a discrete uniform random number over $\{1,2,\ldots,K\}$ for each vertex, and hence repeat this $|V|$ times to obtain a sample y' uniformly, where $|V|$ is the cardinality of V. This sampling scheme avoids sampling y' from a set of cardinality $K^{|V|}$ directly.

5.3.2 Softmax Optimization

Softmax optimization problems naturally arise when applying maximum likelihood estimation to the multinomial logistic model with application in many fields such as economics [84] and and network flows [85]. Specifically, the multinomial logistic model assumes the conditional probability mass of a discrete response $Y\in\{1,\ldots,K\}$, given covariates $X\in\mathbb{R}^p$ and parameters $\beta=[\beta_1,\ldots\beta_K]\in\mathbb{R}^{p\times K}$, satisfies

$$\mathbb{P}(Y=k|X,\beta)=\frac{\exp(X^\top\beta_k)}{\sum_{i=1}^{K}\exp(X^\top\beta_i)}.$$

Given n observations (X_i, Y_i), the log-likelihood function can be written as

$$l(\beta) = \sum_{i=1}^{n} \Big[X_i^\top \beta_{Y_i} - \log\{\sum_{j=1}^{K} \exp(X_j^\top \beta_j)\} \Big].$$

Therefore, maximizing the log-likelihood function, which is known as the Softmax optimization problem, can be viewed as a compositional optimization problem, where the β here corresponds to the x in problem (5.6). To obtain a sample w_i in Algorithm 1 for this problem, we only need to generate a discrete uniform random variable over $\{1, \ldots, K\}$.

5.3.3 Cox's Partial Likelihood

The Cox's partial likelihood model [86, 87] is a widely used in survival analysis for censored data. It belongs to a class of survival models in statistics called the proportional-hazards models in [88]. In particular, the Cox's model assumes there is a hazard function for an observation with covariates $X \in \mathbb{R}^p$ and coefficient $\beta \in \mathbb{R}^p$ as:

$$\lambda(t|X) = \lambda_0(t) \exp(\beta^\top X),$$

where $\lambda_0(t)$ is the baseline hazard function. In Cox's model, for each data point, we have two variables T_i denoting the true life time and C_i denoting the censoring time independent of T_i, which are not observed. Instead, we can only observe $(X_i, Y_i, \Delta_i)_{1 \le i \le n}$ assumed to be i.i.d. observations, where $X_i \in \mathbb{R}^p$ are the covariates, $Y_i \in \mathbb{R}$ are the observed times determined by $Y_i = \min(T_i, C_i)$, and $\Delta_i = \mathbb{I}\{Y_i = T_i\}$ are the indications for the censoring. Moreover, for a particular observation i, we define its risk set as the index set $\{j : Y_j \ge Y_i\}$. Cox's model aims to maximize the partial likelihood function as follows:

$$\max_{\beta \in \mathbb{R}^p} -\frac{1}{n} \sum_{i=1}^{n} \Delta_i [-X_i^\top \beta + \log\{\sum_{j=1}^{n} \mathbb{I}(Y_j \ge Y_i) \exp(X_j^\top \beta)\}], \tag{5.12}$$

which is equivalent to

$$\min_{\beta\in\mathbb{R}^p} \frac{1}{n}\sum_{i=1}^{n}\Delta_i[-X_i^\top\beta + \log\{\frac{1}{n}\sum_{j=1}^{n}\mathbb{I}(Y_j \geq Y_i)\exp(X_j^\top\beta)\}],$$

whose gradient can be written as

$$\frac{1}{n}\sum_{i=1}^{n}\Delta_i[-X_i + \frac{\sum_{j=1}^{n}\mathbb{I}(Y_j \geq Y_i)\exp(X_j^\top\beta)X_j}{\sum_{j=1}^{n}\mathbb{I}(Y_j \geq Y_i)\exp(X_i^\top\beta)}]. \tag{5.13}$$

This problem is of the form of (5.4); hence we can apply our proposed algorithms to solve it.

5.4 Theory

In this section we present the analysis of our algorithms applied to problem (5.4), that is, $\min_{x\in\mathcal{D}} F(x) \triangleq \mathbb{E}_v f_v\{\mathbb{E}_w g_w(x)\}$. We omit the cases (5.6) and (5.7) as they can be analyzed similarly. We first give our assumptions.

5.4.1 Definitions, Assumptions and Lemmas

Assumption 1 In the compact set $\mathcal{D}$, each $f_v(\cdot)$ in the objective function of (5.4) is three times continuously differentiable. Its first-order, second-order and third-order derivatives are Lipschitz continuous with constants $L_{f,1}$,$L_{f,2}$, and $L_{f,3}$, respectively.

Assumption 2 In the compact set $\mathcal{D}$, each $g_w(\cdot)$ is twice continuously differentiable. Its first-order and second-order derivatives are Lipschitz continuous with constant $L_{g,1}$ and, $L_{g,2}$, respectively.

Assumption 3 $F(\cdot)$ in (5.4) is strongly convex with parameter μ and its gradient is Lipstchitz continuous with constant L.

Definition. *Define* $\mathcal{G} = \{y \in \mathbb{R}^d \mid y = g_w(x), x \in \mathcal{D}, w \in \Omega_w\}$ $\mathcal{H} = \{y \in \mathbb{R}^{d\times p} \mid y = \nabla g_w(x), x \in \mathcal{D}, w \in \Omega_w\}$ *and* $\mathcal{J} = \{z \in \mathbb{R}^{d\times p\times p} \mid z = \nabla^2 g_w(x), x \in \mathcal{D}, w \in \Omega_w\}$.

Assumption 4 $l_{g,0} = \sup\{\|y\|_\infty \mid y \in \mathcal{G} \subset \mathbb{R}^d\} < \infty$, $l_{g,1} = \sup\{\|y\|_\infty \mid y \in \mathcal{H} \subset \mathbb{R}^{d\times p}\} < \infty$, and $l_{g,2} = \sup\{\|z\|_\infty \mid z \in \mathcal{J} \subset \mathbb{R}^{d\times p\times p}\}$.

Assumption 5 $l_{f,0} = \sup\{|y| \mid y = f_v(x), x \in \mathcal{G}, v \in \Omega_v\} < \infty$, $l_{f,1} = \sup\{\|y\|_\infty \mid y = \nabla f_v(x), x \in \mathcal{G}, v \in \Omega_v\} < \infty$, $l_{f,2} = \sup\{\|y\|_\infty \mid y = \nabla^2 f_v(x), x \in \mathcal{G}, v \in \Omega_v\} < \infty$, and $l_{f,3} = \sup\{\|y\|_\infty \mid y = \nabla^3 f_v(x), x \in \mathcal{G}, v \in \Omega_v\} < \infty$.

Before we proceed, we state two elementary lemmas used in our proofs.

Lemma 5.4.1. *Let $f : \mathbb{R}^d \to \mathbb{R}$ be a continuously differentiable function with L-Lipschitz continuous gradients, then*

$$|f(y) - f(x) - \langle \nabla f(x), y - x \rangle| \leq \frac{L}{2}\|y - x\|_2^2.$$

We omit the proof of Lemma 5.4.1 since it is a well known result.

Lemma 5.4.2. *Given a positive integer N and a sequence of real number a_i, $1 \leq i \leq N$, we have, for all $p \geq 1$, that*

$$|\sum_{i=1}^{N} a_i|^p \leq N^{p-1} \sum_{i=1}^{N} |a_i|^p, \tag{5.14}$$

Proof. This is a consequence of Jensen's inequality. □

5.4.2 Properties of the Unbiased Gradient Simulation Algorithm

In this subsection, we analysis the properties of Algorithm 8. We first prove the unbiasedness of $G(x, v_1)$.

Proposition 5.4.3 (Unbiasedness)**.** *For any $x \in \mathcal{D}$, sample $v_1 \sim v$, $G(x, v_1)$ is an unbiased estimate of $\nabla \mathbb{E}_v f_v\{\mathbb{E}_w g_w(x)\}$, that is, $\mathbb{E}G(x, v_1) = \nabla \mathbb{E}_v f_v\{\mathbb{E}_w g_w(x)\}$.*

Proof. Fix v_1 and $x \in \mathcal{D}$. We first show that the output $G(x, v_1)$ is an unbiased estimate of

$\nabla f_{v_1}\{\mathbb{E}_w g_w(x)\}$. According to Algorithm 8, we have,

$$\begin{aligned}\mathbb{E}G(x, v_1) &= \sum_{n=0}^{\infty} \mathbb{E}\{G(x, v_1)|N = n\}\mathbb{P}(N = n) \\ &= \sum_{n=0}^{\infty} \frac{\mathbb{E}\{Y_1(x) - 0.5(Y_2(x) + Y_3(x))|N = n\}}{\tilde{p}_n}\tilde{p}_n + \mathbb{E}Y_4(x) \\ &= \sum_{n=0}^{\infty} \mathbb{E}\{Y_1(x) - 0.5(Y_2(x) + Y_3(x))|N = n\} + \mathbb{E}Y_4(x).\end{aligned}$$

Note that condition on $N = n$, we assume there is hypothetically a set of i.i.d. samples $I_{2^{n_0+n+1}}(v_1) = \{w_i\}_{i=1}^{2^{n+n_0+1}}$ that follows the distribution of w given $v = v_1$ that $Y_1(x)$, $Y_2(x)$ and $Y_3(x)$ are constructed. Therefore

$$\begin{aligned}\mathbb{E}\{Y_2(x)|N = n\} &= \mathbb{E}\{\bar{y}(x, 1, 2^{n_0+n})\} = s(x; 2^{n+n_0}) \\ &= \mathbb{E}\{\bar{y}(x; 2^{n_0+n} + 1, 2^{n_0+n+1})\} = \mathbb{E}\{Y_3(x)|N = n\},\end{aligned}$$

$\mathbb{E}Y_4(x) = s(x; 2^{n_0}, v_1)$ and $\mathbb{E}\{Y_1(x)|N = n\} = s(x; 2^{n_0+n+1}, v_1)$. Therefore,

$$\begin{aligned}&\mathbb{E}G(x, v_1) \\ &= \sum_{n=0}^{\infty} \left(s(x; 2^{n_0+n+1}, v_1) - 0.5\{s(x; 2^{n_0+n}, v_1) + s(x; 2^{n_0+n}, v_1)\}\right) + s(x; 2^{n_0}, v_1) \\ &= \sum_{n=0}^{\infty} \{s(x; 2^{n_0+n+1}, v_1) - s(x; 2^{n_0+n}, v_1)\} + s(x; 2^{n_0}, v_1).\end{aligned}$$

Note that the above sum is a telescoping sum. Therefore

$$\begin{aligned}
\mathbb{E}G(x, v_1) &= \lim_{n\to\infty} s(x; 2^{n_0+n}, v_1) - s(x; 2^{n_0}, v_1) + s(x; 2^{n_0}, v_1) \\
&= \lim_{n\to\infty} s(x; 2^{n_0+n}, v_1) = \lim_{n\to\infty} \mathbb{E}\bar{y}(x; 1, 2^{n_0+n}, v_1) \\
&= \lim_{n\to\infty} \mathbb{E}(\overline{\nabla g}(x; 1, 2^{n_0+n})^\top \nabla f_{v_1}\{\bar{g}(x; 1, 2^{n_0+n})\}) \\
&= \lim_{n\to\infty} \mathbb{E}(\{\frac{1}{2^{n_0+n}} \sum_{i=1}^{2^{n_0+n}} \nabla g_{w_i}(x)\}^\top f_{v_1}\{\frac{1}{2^{n_0+n}} \sum_{i=1}^{2^{n_0+n}} g_{w_i}(x)\}).
\end{aligned}$$

Note that

$$\begin{aligned}
&\|\{\frac{1}{2^{n_0+n}} \sum_{i=1}^{2^{n_0+n}} \nabla g_{w_i}(x)\}^\top f_{v_1}\{\frac{1}{2^{n_0+n}} \sum_{i=1}^{2^{n_0+n}} g_{w_i}(x)\}\|_2 \\
\leq\ &\|\{\frac{1}{2^{n_0+n}} \sum_{i=1}^{2^{n_0+n}} \nabla g_{w_i}(x)\}\|_F \|f_{v_1}\{\frac{1}{2^{n_0+n}} \sum_{i=1}^{2^{n_0+n}} g_{w_i}(x)\}\|_2 \\
\leq\ &(\sqrt{pd}\|\{\frac{1}{2^{n_0+n}} \sum_{i=1}^{2^{n_0+n}} \nabla g_{w_i}(x)\}\|_\infty)(\sqrt{d}\|f_{v_1}\{\frac{1}{2^{n_0+n}} \sum_{i=1}^{2^{n_0+n}} g_{w_i}(x)\}\|_\infty) \\
\leq\ &\sqrt{p}dl_{g,0}l_{g,1},
\end{aligned}$$

where the last inequality utilizes Assumption 4 and Assumption 5. Then, by the bounded convergence theorem, we can exchange the expectation and limit and hence

$$\mathbb{E}G(x, v_1) = \mathbb{E} \lim_{n\to\infty} (\{\frac{1}{2^{n_0+n}} \sum_{i=1}^{2^{n_0+n}} \nabla g_{w_i}(x)\}^\top f_{v_1}\{\frac{1}{2^{n_0+n}} \sum_{i=1}^{2^{n_0+n}} g_{w_i}(x)\}).$$

By continuity of $\nabla f_{v_1}(\cdot)$, we have

$$\lim_{n\to\infty} \nabla f_{v_1}\{\frac{1}{2^{n_0+n}} \sum_{i=1}^{2^{n_0+n}} g_{w_i}(x)\} = \nabla f_{v_1}\{\lim_{n\to\infty} \frac{1}{2^{n_0+n}} \sum_{i=1}^{2^{n_0+n}} g_{w_i}(x)\}.$$

Since the samples are i.i.d., by the strong law of large numbers, we have

$$\lim_{n\to\infty} \frac{1}{2^{n_0+n}} \sum_{i=1}^{2^{n_0+n}} g_{w_i}(x) = \mathbb{E}_w g_w(x) \text{ almost surely.}$$

By a similar argument,

$$\lim_{n\to\infty} \frac{1}{2^{n_0+n}} \sum_{i=1}^{2^{n_0+n}} \nabla g_{w_i}(x) = \mathbb{E}_w \nabla g_w(x) \text{ almost surely.}$$

Therefore

$$\begin{aligned}
\mathbb{E}G(x, v_1) &= \mathbb{E} \lim_{n\to\infty} (\{\frac{1}{2^{n_0+n}} \sum_{i=1}^{2^{n_0+n}} \nabla g_{w_i}(x)\}^\top f_{v_1}\{\frac{1}{2^{n_0+n}} \sum_{i=1}^{2^{n_0+n}} g_{w_i}(x)\}) \\
&= \mathbb{E}(\{\mathbb{E}_w \nabla g_w(x)\}^\top \nabla f_{v_1}\{\mathbb{E}_w g_w(x)\}) \\
&= \{\mathbb{E}_w \nabla g_w(x)\}^\top \nabla f_{v_1}\{\mathbb{E} g_w(x)\} = \nabla\{f_{v_1}(\mathbb{E}_w g_w(x))\}.
\end{aligned}$$

Finally, taking expectation w.r.t v_1, we obtain that

$$\mathbb{E}G(x, v_1) = \mathbb{E}_v \nabla(f_v\{\mathbb{E}_w g_w(x)\}) = \nabla \mathbb{E}_v f_v\{\mathbb{E}_w g_w(x)\}.$$

□

We now state two ancillary lemmas that will be used in proving the finite variance of $G(x, v_1)$.

Lemma 5.4.4. *For every $s \in \mathcal{H} \subset R^{d\times p}$, $t \in \mathcal{G} \subset R^d$, and $v_1 \in \Omega_v$, define $H : \mathcal{H} \times \mathcal{G} \to \mathbb{R}^p$ by $H(s,t) = s^\top \nabla f_{v_1}(t)$. Then every component function of $H(s,t)$ has a Lipschitz continuous gradient with constant $L_H = \sqrt{L_{f,1}^2 + 2dl_{f,2}^2 + 2dl_{g,1}^2 L_{f,2}^2}$, i.e., for every $1 \le i \le p$, we have*

$$\|\nabla[H]_i(s_1, t_1) - \nabla[H]_i(s_2, t_2)\|_F \le L_H \|vec([s_1]_{:i}, t_1) - vec([s_2]_{:i}, t_2)\|_2.$$

Proof. Before proving this lemma, we introduce the notation for partial derivatives of $H(s,t)$, i.e.,

each component of the gradient $\nabla H(s,t) \in \mathbb{R}^p \times (\mathbb{R}^{d\times p} \times \mathbb{R}^d)$. Let

$$\frac{\partial [H]_i}{\partial [s]_{kj}}(s,t) = \delta_{ij}\frac{\partial f_{v_1}}{\partial [t]_k}(t), \text{ and} \frac{\partial [H]_i}{\partial [t]_h}(s,t) = \sum_{k=1}^{d}[s]_{ki}\frac{\partial [\nabla f_{v_1}]_k}{\partial [t]_h} = \sum_{k=1}^{d}[s]_{ki}\frac{\partial^2 f_{v_1}}{\partial [t]_k \partial [t]_h}(t),$$

where $1 \le i \le p, 1 \le j \le p$, $1 \le k \le d$, $1 \le h \le d$, and δ_{ij} is the Kronecker delta, i.e., $\delta_{ij} = 1$ when $i = j$; $\delta_{ij} = 0$ otherwise. Note that by Assumption 1, ∇f_{v_1} is Lipschitz continuous with constant $L_{f,1}$; therefore $\frac{\partial [H]_i}{\partial [s]_{kj}}(s,t)$, which is the partial derivative of ∇f_{v_1}, is Lipschitz continuous with constant $L_{f,1}$. By Assumption 1, $\nabla^2 f_{v_1}$ is Lipschitz continuous with constant $L_{f,2}$; therefore $\frac{\partial [H]_i}{\partial [t]_h}(s,t)$ is Lipschitz continuous with constant $L_{f,2}$. Therefore

$$\|\nabla [H]_i(s_1,t_1) - \nabla [H]_i(s_2,t_2)\|_F \le$$
$$\sqrt{\sum_{k=1}^{d}\sum_{j=1}^{p}(\delta_{ij}\frac{\partial f_{v_1}}{\partial [t]_k}(t_1) - \delta_{ij}\frac{\partial f_{v_1}}{\partial [t]_k}(t_2))^2 + \sum_{h=1}^{d}(\sum_{k=1}^{d}[s_1]_{ki}\frac{\partial^2 f_{v_1}}{\partial [t]_k \partial [t]_h}(t_1) - \sum_{k=1}^{d}[s_2]_{ki}\frac{\partial^2 f_{v_1}}{\partial [t]_k \partial [t]_h}(t_2))^2}.$$

Since

$$\sum_{k=1}^{d}\sum_{j=1}^{p}\{\delta_{ij}\frac{\partial f_{v_1}}{\partial [t]_k}(t_1) - \delta_{ij}\frac{\partial f_{v_1}}{\partial [t]_k}(t_2)\}^2 = \sum_{k=1}^{d}\{\frac{\partial f_{v_1}}{\partial [t]_k}(t_1) - \frac{\partial f_{v_1}}{\partial [t]_k}(t_2)\}_2^2 = \|\nabla f_{v_1}(t_1) - \nabla f_{v_2}(t_2)\|_2^2$$
$$\le L_{f,1}^2\|t_1 - t_2\|_2^2$$

using the fact that $|[s_2]_{ki}| \leq l_{g,1}\,|\frac{\partial^2 f_{v_1}}{\partial[t]_k\partial[t]_h}(t_2)| \leq l_{f,2}$ for all k and h,

$$
\begin{aligned}
&\sum_{h=1}^{d}\{\sum_{k=1}^{d}[s_1]_{ki}\frac{\partial^2 f_{v_1}}{\partial[t]_k\partial[t]_h}(t_1) - \sum_{k=1}^{d}[s_2]_{ki}\frac{\partial^2 f_{v_1}}{\partial[t]_k\partial[t]_h}(t_2)\}^2 \\
\leq\ & 2\sum_{h=1}^{d}\{\sum_{k=1}^{d}[s_1]_{ki}\frac{\partial^2 f_{v_1}}{\partial[t]_k\partial[t]_h}(t_1) - \sum_{k=1}^{d}[s_2]_{ki}\frac{\partial^2 f_{v_1}}{\partial[t]_k\partial[t]_h}(t_1)\}^2 \\
&+ 2\sum_{h=1}^{d}\{\sum_{k=1}^{d}[s_2]_{ki}\frac{\partial^2 f_{v_1}}{\partial[t]_k\partial[t]_h}(t_1) - \sum_{k=1}^{d}[s_2]_{ki}\frac{\partial^2 f_{v_1}}{\partial[t]_k\partial[t]_h}(t_2)\}^2 \\
\leq\ & 2l_{f,2}^2\sum_{h=1}^{d}\{\sum_{k=1}^{d}[s_1]_{ki} - [s_2]_{ki}\}^2 + 2l_{g,1}^2\sum_{h=1}^{d}\{\sum_{k=1}^{d}\frac{\partial^2 f_{v_1}}{\partial[t]_k\partial[t]_h}(t_1) - \frac{\partial^2 f_{v_1}}{\partial[t]_k\partial[t]_h}(t_2)\}^2 \\
\leq\ & 2l_{f,2}^2 d\|[s_1]_{:i} - [s_2]_{:i}\|_2^2 + 2l_{g,1}^2 dL_{f,2}^2\|t_1 - t_2\|_2^2.
\end{aligned}
$$

Hence,

$$
\begin{aligned}
&\|\nabla[H]_i(s_1,t_1) - \nabla[H]_i(s_2,t_2)\|_F \\
\leq\ & \sqrt{L_{f,1}^2\|t_1 - t_2\|_2^2 + 2l_{f,2}^2 d\|[s_1]_{\cdot i} - [s_2]_{\cdot i}\|_2^2 + 2dl_{g,1}^2 L_{f,2}^2\|t_1 - t_2\|_2^2} \\
\leq\ & \sqrt{L_{f,1}^2 + 2dl_{f,2}^2 + 2dl_{g,1}^2 L_{f,2}^2}\|\mathrm{vec}([s_1]_{:i}, t_1) - \mathrm{vec}([s_2]_{:i}, t_2)\|_2 \\
=\ & L_H\|\mathrm{vec}([s_1]_{:i}, t_1) - \mathrm{vec}([s_2]_{:i}, t_2)\|_2.
\end{aligned}
$$

□

Lemma 5.4.5. *For every $s, s_0 \in \mathcal{H} \subset \mathbb{R}^{d\times p}$ and $t, t_0 \in \mathcal{G} \subset \mathbb{R}^p$, define*

$$R(s, s_0, t, t_0) = H(s,t) - H(s_0,t_0) - \nabla H(s_0,s_0)[s - s_0, t - t_0].$$

Then we have

$$\|R(s, s_0, t, t_0)\| \leq \frac{L_H}{2}(\|s - s_0\|_F^2 + p\|t - t_0\|_2^2).$$

Proof. Recall that $\nabla H(s,t)[u,v] \in \mathbb{R}^p$, $u \in \mathbb{R}^{d\times p}$, $v \in \mathbb{R}^d$ and each component of $\nabla H(s,t)$ is

defined as

$$[\nabla H(s,t)[u,v]]_i = \nabla[H]_i(s,t)[u,v] = \sum_{k=1}^{d}\sum_{j=1}^{p}\frac{\partial[H]_i}{\partial[s]_{kj}}(s,t)\cdot[u]_{kj} + \sum_{h=1}^{d}\frac{\partial[H]_i}{\partial[t]_h}(s,t)\cdot[v]_h.$$

Note that $R(s, s_0, t, t_0)$ can be considered as the remainder of the first-order Taylor expansion of $H(s,t)$ at (s_0, t_0). Now using Lemma 5.4.1, we have

$$\begin{aligned}
\|R(s,s_0,t,t_0)\|_2 &= \|H(s,t) - H(s_0,t_0) - \nabla H(s_0,t_0)[(s-s_0),(t-t_0)]\|_2 \\
&= \sqrt{\sum_{i=1}^{p}|[H]_i(s,t) - [H]_i(s_0,t_0) - \nabla[H]_i(s_0,t_0)[s-s_0,t-t_0]|^2} \\
&\le \sum_{i=1}^{p}|[H]_i(s,t) - [H]_i(s_0,t_0) - \nabla[H]_i(s_0,t_0)[s-s_0,t-t_0]| \\
&\le \sum_{i=1}^{p}\frac{1}{2}\sqrt{L_{f,1}^2 + 2dl_{f,2}^2 + 2dl_{g,1}^2L_{f,2}^2}\,\|\mathrm{vec}([s]_i,t) - \mathrm{vec}([s_0]_i,t_0)\|_2^2 \\
&= \frac{1}{2}\sqrt{L_{f,1}^2 + 2dl_{f,2}^2 + 2dl_{g,1}^2L_{f,2}^2}(\|s-s_0\|_F^2 + p\|t-t_0\|_2^2) \\
&= \frac{L_H}{2}(\|s-s_0\|_F^2 + p\|t-t_0\|_2^2) \qquad (5.15)
\end{aligned}$$

for any $x, x_0 \in \mathcal{H}$ and $y, y_0 \in \mathcal{G}$.

□

Proposition 5.4.6 (Finite second moment)**.** *Fix any $x \in \mathcal{D}$ and $v_1 \in \Omega_v$, we have*

$$\mathbb{E}\|G(x,v_1)\|_2^2 \le C'_{\mathcal{D}},$$

where

$$C'_{\mathcal{D}} = 2pd^2l_{g,1}^2l_{f,1}^2 + \frac{108p^2d^2(L_{f,1}^2 + 2df_{f,2}^2 + 2dl_{g,1}^2L_{f,2}^2)(l_{g,0}^4 + l_{g,1}^4)}{4^{n_0}(1-0.5^{\gamma})(1-0.5^{2-\gamma})}$$

and $1 < \gamma < 2$ is from the unbiased gradient simulation algorithm. Therefore $G(x,v_1)$ has finite variance.

Proof. First, by (5.14),

$$\begin{aligned}\|G(x,v_1)\|_2^2 &= \|\frac{(Y_1(x)-0.5\{Y_2(x)+Y_3(x)\})}{\tilde{p}_N}+Y_4(x)\|_2^2\\ &\le 2\|\frac{(Y_1(x)-0.5\{Y_2(x)+Y_3(x)\})}{\tilde{p}_N}\|_2^2+2\|Y_4(x)\|_2^2.\end{aligned}$$

To obtain an upper bound of $\mathbb{E}\|G(x,v_1)\|_2^2$, we first take expectation with respect to N. Therefore

$$\begin{aligned}&\mathbb{E}\|G(x,v_1)\|_2^2\\ &\le 2\sum_{n=0}^{\infty}\mathbb{E}(\frac{\|Y_1(x)-0.5\{Y_2(x)+Y_3(x)\}\|_2^2}{\tilde{p}_n^2}|N=n)\mathbb{P}(N=n)+2\mathbb{E}\|Y_4(x)\|_2^2\\ &\le 2\sum_{n=0}^{\infty}\frac{\mathbb{E}(\|Y_1(x)-0.5\{Y_2(x)+Y_3(x)\}\|_2^2|N=n)}{\tilde{p}_n}+2\mathbb{E}\|Y_4(x)\|_2^2. \qquad (5.16)\end{aligned}$$

To bound $\|Y_4(x)\|_2^2$, we first note that

$$\begin{aligned}\|Y_4(x)\|_2^2 &= \|\{\frac{1}{2^{n_0}}\sum_{i=1}^{2^{n_0}}\nabla g_{w_i}(x)\}^\top\nabla f_{v_1}\{\frac{1}{2^{n_0}}\sum_{i=1}^{2^{n_0}}g_{w_i}(x)\}\|_2^2\\ &\le \|\frac{1}{2^{n_0}}\sum_{i=1}^{2^{n_0}}\nabla g_{w_i}(x)\|_2^2\,\|\nabla f_{v_1}\{\frac{1}{2^{n_0}}\sum_{i=1}^{2^{n_0}}g_{w_i}(x)\}\|_2^2\\ &\le \|\frac{1}{2^{n_0}}\sum_{i=1}^{2^{n_0}}\nabla g_{w_i}(x)\|_F^2\,\|\nabla f_{v_1}\{\frac{1}{2^{n_0}}\sum_{i=1}^{2^{n_0}}g_{w_i}(x)\}\|_2^2.\end{aligned}$$

Then by Assumptions 4 and 5,

$$\|\frac{1}{2^{n_0}}\sum_{i=1}^{2^{n_0}}\nabla g_{w_i}(x)\|_F\le\sqrt{pd}\|\frac{1}{2^{n_0}}\sum_{i=1}^{2^{n_0}}\nabla g_{w_i}(x)\|_\infty\le\sqrt{pd}l_{g,1},\text{ and}$$

$$\|\nabla f_{v_1}\{\frac{1}{2^{n_0}}\sum_{i=1}^{2^{n_0}}g_{w_i}(x)\}\|_2\le\sqrt{d}\|\nabla f_{v_1}\{\frac{1}{2^{n_0}}\sum_{i=1}^{2^{n_0}}g_{w_i}(x)\}\|_\infty\le\sqrt{d}l_{f,1}.$$

Therefore

$$\mathbb{E}\|Y_4(x)\|_2^2 \leq pd^2 l_{g,1}^2 l_{f,1}^2. \tag{5.17}$$

To bound the first term on the right hand side of (5.16), we first define the following vector-valued function: for $s \in \mathcal{H} \subseteq \mathbb{R}^{d\times p}$ and $t \in \mathcal{G} \subseteq \mathbb{R}^d$, define $H : \mathcal{H} \times \mathcal{G} \to \mathbb{R}^p$ by $H(s,t) \triangleq s^\intercal \nabla f_{v_1}(t)$. Moreover, to simplify the notation, let $\bar{n}_0 = n_0 + n$ and $\bar{n}_0^+ = n_0 + n + 1$. Therefore given that $N = n$, we can write

$$\begin{aligned}
&Y_1(x) - 0.5\{Y_2(x) + Y_3(x)\} \\
&= \bar{y}(x; 1, 2^{\bar{n}_0^+}) - 0.5\{\bar{y}(x; 1, 2^{\bar{n}_0}) + \bar{y}(x; 2^{\bar{n}_0} + 1, 2^{\bar{n}_0^+})\} \\
&= H\{\overline{\nabla g}(x; 1, 2^{\bar{n}_0^+}), \bar{g}(x; 1, 2^{\bar{n}_0^+})\} - 0.5H\{\overline{\nabla g}(x; 1, 2^{\bar{n}_0}), \bar{g}(x; 1, 2^{\bar{n}_0})\} \\
&- 0.5H\{\overline{\nabla g}(x; 2^{\bar{n}_0} + 1, 2^{\bar{n}_0^+}), \bar{g}(x; 2^{\bar{n}_0} + 1, 2^{\bar{n}_0^+}).
\end{aligned} \tag{5.18}$$

Since $\bar{g}(x; 1, 2^{\bar{n}_0^+}) = 0.5\{\bar{g}(x; 1, 2^{\bar{n}_0}) + \bar{g}(x; 2^{\bar{n}_0} + 1, 2^{\bar{n}_0^+})\}$, and $\overline{\nabla g}(x; 1, 2^{\bar{n}_0^+}) = 0.5\{\overline{\nabla g}(x; 1, 2^{\bar{n}_0}) + \overline{\nabla g}(x; 2^{\bar{n}_0} + 1, 2^{\bar{n}_0^+})\}$, when expanding the three functions in (5.18) at $(\mathbb{E}_w \nabla g_w(x), \mathbb{E}_w g_w(x))$, the zeroth order terms and first order terms vanish. Therefore condition on $N = n$,

$$\begin{aligned}
&Y_1(x) - 0.5(Y_2(x) + Y_3(x)) \\
&= R\{\overline{\nabla g}(x; 1, 2^{\bar{n}_0^+}), \mathbb{E}_w \nabla g_w(x), \bar{g}(x; 1, 2^{\bar{n}_0^+}), \mathbb{E}_w g_w(x)\} \\
&- 0.5R\{\overline{\nabla g}(x; 1, 2^{\bar{n}_0}), \mathbb{E}_w \nabla g_w(x), \bar{g}(x; 1, 2^{\bar{n}_0}), \mathbb{E}_w g_w(x)\} \\
&- 0.5R\{\overline{\nabla g}(x; 2^{\bar{n}_0} + 1, 2^{\bar{n}_0^+}), \mathbb{E}_w \nabla g_w(x), \bar{g}(x; 2^{\bar{n}_0} + 1, 2^{\bar{n}_0^+}), \mathbb{E}_w g_w(x)\}.
\end{aligned}$$

As a result, using (5.14) and (5.15), we have

$$
\begin{aligned}
&\sum_{n=0}^{\infty} \frac{\mathbb{E}[\|Y_1 - 0.5(Y_2+Y_3)\|_2^2 | N=n]}{\tilde{p}_n} \\
&\le \sum_{n=0}^{\infty} \frac{3}{\tilde{p}_n} \Big(\mathbb{E}\|R\{\overline{\nabla g}(x;1,2^{\bar{n}_0^+}), \mathbb{E}_w \nabla g_w(x), \bar{g}(x;1,2^{\bar{n}_0^+}), \mathbb{E}_w g_w(x)\}\|_2^2 \\
&\quad + \frac{1}{4}\mathbb{E}\|R\{\overline{\nabla g}(x;1,2^{\bar{n}_0}), \mathbb{E}_w \nabla g_w(x), \bar{g}(x;1,2^{\bar{n}_0}), \mathbb{E}_w g_w(x)\}\|_2^2 \\
&\quad + \frac{1}{4}\mathbb{E}\|R\{\overline{\nabla g}(x;2^{\bar{n}_0}+1,2^{\bar{n}_0^+}), \mathbb{E}_w \nabla g_w(x), \bar{g}(x;2^{\bar{n}_0}+1,2^{\bar{n}_0^+}), \mathbb{E}_w g_w(x)\}\|_2^2 \Big) \\
&\le \frac{3L_H^2}{4} \sum_{n=0}^{\infty} \frac{1}{\tilde{p}_n} \Big(\mathbb{E}(\|\overline{\nabla g}(x;1,2^{\bar{n}_0^+}) - \mathbb{E}_w \nabla g_w(x)\|_F^2 + p\|\bar{g}(x;1,2^{\bar{n}_0^+}) - \mathbb{E}_w g_w(x)\|_2^2)^2 \\
&\quad + \frac{1}{4}\mathbb{E}(\|\overline{\nabla g}(x;2^{\bar{n}_0}+1,2^{\bar{n}_0^+}) - \mathbb{E}_w \nabla g_w(x)\|_F^2 + p\|\bar{g}(x;2^{\bar{n}_0}+1,2^{\bar{n}_0^+}) - \mathbb{E}_w g_w(x)\|_2^2)^2 \Big) \\
&\quad + \frac{1}{4}\mathbb{E}(\|\overline{\nabla g}(x;1,2^{\bar{n}_0}) - \mathbb{E}_w \nabla g_w(x)\|_F^2 + p\|\bar{g}(x;1,2^{\bar{n}_0}) - \mathbb{E}_w g_w(x)\|_2^2)^2. \qquad (5.19)
\end{aligned}
$$

Then, by (5.14),

$$
\begin{aligned}
&\mathbb{E}(\|\overline{\nabla g}(x;1,2^{\bar{n}_0^+}) - \mathbb{E}_w \nabla g_w(x)\|_F^2 + p\|\bar{g}(x;1,2^{\bar{n}_0^+}) - \mathbb{E}_w g_w(x)\|_2^2)^2 \\
&\le 2\mathbb{E}\|\overline{\nabla g}(x;1,2^{\bar{n}_0^+}) - \mathbb{E}_w \nabla g_w(x)\|_F^4 + 2p^2\mathbb{E}\|\bar{g}(x;1,2^{\bar{n}_0^+}) - \mathbb{E}_w g_w(x)\|_2^2)^4.
\end{aligned}
$$

Next, we will analyze the two terms on the right hand side of the inequality above.

Since $\overline{\nabla g}(x;1,2^{\bar{n}_0^+}) = \frac{1}{2^{\bar{n}_0^+}} \sum_{i=1}^{2^{\bar{n}_0^+}} \nabla g_{w_i}(x)$, and $\mathbb{E}\overline{\nabla g}(x;1,2^{\bar{n}_0^+}) = \mathbb{E}_w \nabla g_w(x)$, we can write

$$
\begin{aligned}
&\mathbb{E}\|\overline{\nabla g}(x;1,2^{\bar{n}_0^+}) - \mathbb{E}_w \nabla g_w(x)\|_F^4 \\
&= \mathbb{E}\{\sum_{k=1}^{d}\sum_{h=1}^{p}(\frac{1}{2^{\bar{n}_0^+}}\sum_{i=1}^{2^{\bar{n}_0^+}}\{[\nabla g_{w_i}(x)]_{kh} - \mathbb{E}_w[\nabla g_w(x)]_{kh}\})^2\}^2 \\
&\le pd\sum_{k=1}^{d}\sum_{h=1}^{p}\mathbb{E}(\frac{1}{2^{\bar{n}_0^+}}\sum_{i=1}^{2^{\bar{n}_0^+}}\{[\nabla g_{w_i}(x)]_{kh} - \mathbb{E}_w[\nabla g_w(x)]_{kh}\})^4,
\end{aligned}
$$

where the last inequality is obtained by using (5.14). Note that for i.i.d. $\{X\}_{i=1}^n$'s that $\mathbb{E}X_i = 0$, and

$|X| \le c_0$ we have

$$\begin{aligned}
&\mathbb{E}(\frac{1}{n}\sum_{i=1}^{n} X_i)^4 \\
&= \frac{1}{n^4}\mathbb{E}\{\sum_{i=1}^{n} X_i^4 + \sum_{i\neq j}(4X_i^3X_j + 3X_i^2X_j^2) + \sum_{i\neq j\neq k} 6X_i^2X_jX_k + \sum_{i\neq j\neq k\neq h} X_iX_jX_kX_h\} \\
&= \frac{1}{n^4}\{n\mathbb{E}X_1^4 + 3n(n-1)\mathbb{E}X_1^2X_2^2\} \le \frac{3c_0^4}{n^2}.
\end{aligned}$$

Since $|[\nabla g_{w_i}(x)]_{kh} - \mathbb{E}_w[\nabla g_w(x)]_{kh}| \le 2l_{g,1}$ and $\mathbb{E}\{[\nabla g_{w_i}(x)]_{kh} - \mathbb{E}_w[\nabla g_w(x)]_{kh}\} = 0$, we have $\mathbb{E}(\frac{1}{2^{\bar{n}_0^+}}\sum_{i=1}^{2^{\bar{n}_0^+}}\{[\nabla g_{w_i}(x)]_{kh} - \mathbb{E}_w[\nabla g_w(x)]_{kh}\})^4 \le \frac{48l_{g,1}^4}{4^{\bar{n}_0^+}}$ and hence $\mathbb{E}\|\overline{\nabla g}(x;1,2^{\bar{n}_0^+}) - \mathbb{E}_w\nabla g_w(x)\|_F^4 \le \frac{48p^2d^2l_{g,1}^4}{4^{\bar{n}_0^+}}$. By the same argument, we also have $\mathbb{E}\|\overline{\nabla g}(x;1,2^{\bar{n}_0}) - \mathbb{E}_w\nabla g_w(x)\|_F^4 \le \frac{48p^2d^2l_{g,1}^4}{4^{\bar{n}_0}}$ and $\mathbb{E}\|\overline{\nabla g}(x;2^{\bar{n}_0}+1,2^{\bar{n}_0^+}) - \mathbb{E}_w\nabla g_w(x)\|_F^4 \le \frac{48p^2d^2l_{g,1}^4}{4^{\bar{n}_0}}$. Similarly, since $\mathbb{E}\bar{g}(x;1,2^{\bar{n}_0^+}) = \mathbb{E}_w g_w(x)$ and $|[g_{w_i}(x)]_j - \mathbb{E}_w[g_w(x)]_j| \le 2l_{g,0}$, we have

$$\begin{aligned}
&\mathbb{E}\|\bar{g}(x;1,2^{\bar{n}_0^+}) - \mathbb{E}_w g_w(x)\|_2^4 = \mathbb{E}(\sum_{j=1}^{d}\{\frac{1}{2^{\bar{n}_0^+}}\sum_{i=1}^{2^{\bar{n}_0^+}}([g_{w_i}(x)]_j - \mathbb{E}_w[g_w(x)]_j)\}^2)^2 \\
&\le d\sum_{j=1}^{d}\mathbb{E}\{\frac{1}{2^{\bar{n}_0^+}}\sum_{i=1}^{2^{\bar{n}_0^+}}([g_{w_i}(x)]_j - \mathbb{E}_w[g_w(x)]_j)\}^4 \le \frac{48d^2l_{g,0}^4}{4^{\bar{n}_0^+}}.
\end{aligned}$$

Using the same argument, we also have $\mathbb{E}\|\bar{g}(x;1,2^{\bar{n}_0}) - \mathbb{E}_w g_w(x)\|_2^4 \le \frac{48d^2l_{g,0}^4}{4^{\bar{n}_0}}$, and $\mathbb{E}\|\bar{g}(x;2^{\bar{n}_0}+1,2^{\bar{n}_0^+}) - \mathbb{E}_w g_w(x)\|_2^4 \le \frac{48d^2l_{g,0}^4}{4^{\bar{n}_0}}$. Therefore

$$\begin{aligned}
&\mathbb{E}(\|\overline{\nabla g}(x;1,2^{\bar{n}_0^+}) - \mathbb{E}_w\nabla g_w(x)\|_F^2 + p\|\bar{g}(x;1,2^{\bar{n}_0^+}) - \mathbb{E}_w g_w(x)\|_2^2)^2 \\
&\le 2\mathbb{E}\|\overline{\nabla g}(x;1,2^{\bar{n}_0^+}) - \mathbb{E}_w\nabla g_w(x)\|_F^4 + 2p^2\mathbb{E}\|\bar{g}(x;1,2^{\bar{n}_0^+}) - \mathbb{E}_w g_w(x)\|_2^2)^4 \\
&\le \frac{96p^2d^2(l_{g,0}^4 + l_{g,1}^4)}{4^{\bar{n}_0^+}}.
\end{aligned}$$

Hence

$$\mathbb{E}(\|\overline{\nabla g}(x;1,2^{\bar{n}_0}) - \mathbb{E}_w \nabla g_w(x)\|_F^2 + p\|\bar{g}(x;1,2^{\bar{n}_0}) - \mathbb{E}_w g_w(x)\|_2^2)^2 \leq \frac{96p^2d^2(l_{g,0}^4 + l_{g,1}^4)}{4^{\bar{n}_0}} \quad \text{and}$$

$$\mathbb{E}(\|\overline{\nabla g}(x;2^{\bar{n}_0}+1,2^{\bar{n}_0^+}) - \mathbb{E}_w \nabla g_w(x)\|_F^2 + p\|\bar{g}(x;2^{\bar{n}_0}+1,2^{\bar{n}_0^+}) - \mathbb{E}_w g_w(x)\|_2^2)^2 \leq \frac{96p^2d^2(l_{g,0}^4 + l_{g,1}^4)}{4^{\bar{n}_0}}.$$

Now we continue with the analysis of (5.19)

$$\begin{aligned}
&\sum_{n=0}^{\infty} \frac{1}{\tilde{p}_n} \mathbb{E}\{(Y_1(x) - 0.5(Y_2(x) + Y_3(x)))^2 \mid N = n\} \\
&\leq \frac{3L_H}{4} \sum_{n=0}^{\infty} \frac{1}{\tilde{p}_n} \Big(\mathbb{E}(\|\overline{\nabla g}(x;1,2^{\bar{n}_0^+}) - \mathbb{E}_w \nabla g_w(x)\|_F^2 + p\|\bar{g}(x;1,2^{\bar{n}_0^+}) - \mathbb{E}_w g_w(x)\|_2^2)^2 \\
&\quad + \frac{1}{4}\mathbb{E}(\|\overline{\nabla g}(x;1,2^{\bar{n}_0}) - \mathbb{E}_w \nabla g_w(x)\|_F^2 + p\|\bar{g}(x;1,2^{\bar{n}_0}) - \mathbb{E}_w g_w(x)\|_2^2)^2 \\
&\quad + \frac{1}{4}\mathbb{E}(\|\overline{\nabla g}(x;2^{\bar{n}_0}+1,2^{\bar{n}_0^+}) - \mathbb{E}_w \nabla g_w(x)\|_F^2 + p\|\bar{g}(x;2^{\bar{n}_0}+1,2^{\bar{n}_0^+}) - \mathbb{E}_w g_w(x)\|_2^2)^2 \Big) \\
&\leq 72(L_{f,1}^2 + 2df_{f,2}^2 + 2dl_{g,1}^2 L_{f,2}^2)p^2d^2(l_{g,0}^4 + l_{g,1}^4) \sum_{n=0}^{\infty} \frac{3}{\tilde{p}_n 4^{n+n_0+1}},
\end{aligned}$$

since $L_H = \sqrt{L_1^2 + 2df_{f,2}^2 + 2dl_{g,1}^2 L_2^2}$. Note that $\tilde{p}_n = (1 - 0.5^{\gamma})0.5^{\gamma n}$ and $1 < \gamma < 2$; therefore

$$\sum_{n=0}^{\infty} \frac{3}{\tilde{p}_n 4^{n+n_0+1}} = \frac{3}{4^{n_0+1}(1-0.5^{\gamma})} \sum_{n=0}^{\infty} 2^{n(\gamma-2)} = \frac{3}{4^{n_0+1}(1-0.5^{\gamma})(1-0.5^{2-\gamma})} < \infty.$$

Hence

$$\sum_{n=0}^{\infty} \frac{\{\|Y_1(x) - 0.5(Y_2(x) + Y_3(x))\|_2^2}{\tilde{p}_n} \leq \frac{54p^2d^2(L_{f,1}^2 + 2df_{f,2}^2 + 2dl_{g,1}^2 L_{f,2}^2)(l_{g,0}^4 + l_{g,1}^4)}{4^{n_0}(1-0.5^{\gamma})(1-0.5^{2-\gamma})} \quad (5.20)$$

Combining (5.17) and (5.20), we can bound (5.16) by

$$\begin{aligned}
\mathbb{E}\|G(x,v_1)\|_2^2 &\leq 2\mathbb{E}\|Y_4(x)\|_2^2 + 2\sum_{n=0}^{\infty} \frac{\{\|Y_1(x) - 0.5(Y_2(x) + Y_3(x))\|_2^2}{\tilde{p}_n} \\
&\leq 2pd^2 l_{g,1}^2 l_{f,1}^2 + \frac{108p^2d^2(L_{f,1}^2 + 2df_{f,2}^2 + 2dl_{g,1}^2 L_{f,2}^2)(l_{g,0}^4 + l_{g,1}^4)}{4^{n_0}(1-0.5^{\gamma})(1-0.5^{2-\gamma})} = C'_{\mathcal{D}}.
\end{aligned}$$

□

Proposition 5.4.7 (Finite expected computational cost)**.** *For any $x \in \mathcal{D}$ and $v_1 \in \Omega_v$, the number of random numbers one needs to generate (simulation cost) to construct $G(x, v_1)$ has finite expectation.*

Proof. Fix $v_1 \in \Omega_v$ and $x \in \mathcal{D}$, and denote by $cost_G$ the number of random variables one needs to generate to construct $G(x, v_1)$. In Algorithm 8, we generate one geometric random variable N and 2^{n_0+n+1} w_i's that follows the distribution of w conditioned on $v = v_1$. Thus we have $cost_G = 1 + 2^{n_0+N+1}$. Taking expectation w.r.t. N, we conclude

$$\begin{aligned}\mathbb{E}(cost_G) = \mathbb{E}\{\mathbb{E}(cost_G|N)\} &= \sum_{n=0}^{\infty} \mathbb{E}(cost_G|N=n)\mathbb{P}(N=n) \\ &= \sum_{n=0}^{\infty} (1+2^{n_0+n+1})(1-0.5^{\gamma})0.5^{\gamma n} \\ &= 1 + 2^{n_0+1}(1-0.5^{\gamma})(1-2^{1-\gamma})^{-1} < \infty,\end{aligned}$$

where the convergence of the series above relies on $\gamma > 1$. □

Remark: Note that the choices of both the base level n_0 and γ affect both the variance of the simulated estimator and its computational cost. By choosing a larger n_0, the variance of the simulated gradient will be lower but it will also have a higher computational cost. Similarly, choosing a smaller γ will result in an estimator that has lower variance but higher computational cost.

5.4.3 Convergence of the Simulated Gradient Descent Algorithm

In this subsection, we establish the convergence properties of Algorithm 9 when $F(\cdot)$ is either μ-strongly convex or non-strongly convex. Note that with the unbiasedness and finite second-order moment properties of the simulated gradients, convergence properties of the Simulated Gradient

Descent (SimGD) algorithm for SCO problems follow from the classical theory of SGD for generic stochastic optimization problems.

Lemma 5.4.8. *[Almost Sure Convergence] If $F(\cdot)$ is μ-strongly convex, assume $\mathbb{E}\|x_t - x_\star\|_2^2 \le D$ for all $t \ge 0$. When $\sum_t \lambda_t = \infty$ and $\sum_t \lambda_t^2 < \infty$, $\|x_t - x_\star\|_2^2$ converges to 0 almost surely.*

Proof. Define $Y_t = \|x_t - x_\star\|_2^2$. By the contraction property of projection operators, we have $Y_{t+1} = \|x_{t+1} - x_\star\|_2^2 = \|\Pi_{\mathcal{D}}(x_t - \lambda_t \rho_t) - \Pi_{\mathcal{D}}(x_\star)\|_2^2 \le \|x_t - \lambda_t \rho_t - x_\star\|_2^2$. Thus

$$Y_{t+1} - Y_t \le \|x_{t+1} - x_\star\|^2 = \|x_t - x_\star\|^2 = -2\lambda_t (x_t - x_\star)^\intercal \rho_t + \lambda_t^2 \|\rho_t\|_2^2, \tag{5.21}$$

Moreover, with respect to the natural filtration $\{\mathcal{F}_t\}_{t\ge 0}$, we can obtain, using Proposition 5.4.3 and 5.4.6, $\mathbb{E}\{\rho_t \,|\, \mathcal{F}_t\} = \nabla F(x_t)$ and $\mathbb{E}\{\|\rho_t\|_2^2 \,|\, \mathcal{F}_t\} \le C'_{\mathcal{D}}$ and by convexity of $F(\cdot)$, we have $0 \ge F(x_\star) - F(x) \ge (x_\star - x)^\intercal \nabla F(x)$. Therefore

$$\mathbb{E}[Y_{t+1} - Y_t | \mathcal{F}_t] \le -2\lambda_t (x_t - x_\star)^\intercal \nabla F(x_t) + \lambda_t^2 C'_{\mathcal{D}} \le \lambda_t^2 C'_{\mathcal{D}}. \tag{5.22}$$

Define $M_t = Y_t + \sum_t^\infty \lambda_s^2 C'_{\mathcal{D}}$ with respect to the natural filtration $\mathcal{F}_t$. Then it can be checked that M_t is a positive supermartingale with finite expected values. Thus, it follows from the martingale convergence theorem that M_t and consequently $Y_t = \|x_t - x_\star\|_2^2$ converges almost surely. To show that $\|x_t - x_\star\|_2^2 \to 0$, we define $Z_t = \sum_{s=0}^t 2\lambda_t (x_t - x_\star)^\intercal \nabla F(x_t)$, and notice that $0 \le Z_t \le Z_{t+1}$ due to convexity of $F(\cdot)$. Therefore, using the monotone convergence theorem and (5.22) we have

$$\begin{aligned}\mathbb{E}[\sum_t 2\lambda_t (x_t - x_\star)^\intercal \nabla F(x_t)] &\le \sum_t \mathbb{E}[2\lambda_t (x_t - x_\star)^\intercal \nabla F(x_t)] \\ &= \sum_t \mathbb{E}[Y_t] - \mathbb{E}[Y_{t+1}] + \lambda_t^2 \mathbb{E}[\rho_t^2] \le D + \sum_t \lambda_t^2 C'_{\mathcal{D}} < \infty. \end{aligned} \tag{5.23}$$

Thus the monotone series $Z_t = \sum_{s\le t} 2\lambda_s (x_s - x_\star)^\intercal \nabla F(x_s)$ converges almost surely. It follows from $\sum_t \lambda_t = \infty$ and $(x_t - x_\star)^\intercal \nabla F(x_t) \ge 0$ that $(x_t - x_\star)^\intercal \nabla F(x_t) \to 0$. Since $F(\cdot)$ is μ-strongly convex, we have $(x_t - x_\star)^\intercal \nabla F(x_t) \ge \mu \|x_t - x_\star\|_2^2$, which implies $\|x_t - x_\star\|_2^2 \to 0$. □

The techniques of our proof for the Lemma below come mostly from [73].

Lemma 5.4.9. *[Rate of Convergence] In the presence of μ-strong convexity for $F(\cdot)$, with $\lambda_t = \frac{2}{\mu(t+1)}$, we can show that $\mathbb{E}\|x_T - x_\star\|_2^2 \leq \frac{4C'_{\mathcal{D}}}{\mu^2(T+1)}$ and $\mathbb{E}\|\tilde{x}_T - x_\star\|_2^2 \leq \frac{4C'_{\mathcal{D}}}{\mu^2(T+1)}$. In the case where $F(\cdot)$ is not strongly convex, if we have $\mathbb{E}\|x_t - x_\star\|_2^2 \leq D$ for all t, then with $\lambda_t = \frac{c}{\sqrt{t+1}}$ and $c > 0$, we can show that $\mathbb{E}F(\tilde{x}_T) - F(x_\star) \leq \frac{2\sqrt{2}C'_{\mathcal{D}} + c^{-1}4\sqrt{2}D}{\sqrt{T}}$.*

Proof. By the contraction property of projection operators, we have

$$\begin{aligned}
\mathbb{E}[\|x_t - x_\star\|^2 | x_{t-1}] &\leq \mathbb{E}[\|x_{t-1} - \lambda_t \rho_{t-1} - x_\star\|^2 | x_{t-1}] \\
&= \|x_{t-1} - x_\star\|_2^2 + \lambda_t^2 \mathbb{E}[\|\rho_{t-1}\|_2^2 | x_{t-1}] - 2\lambda_t (x_{t-1} - x_\star)^\intercal \mathbb{E}[\rho_{t-1} | x_{t-1}] \\
&= \|x_{t-1} - x_\star\|_2^2 + \lambda_t^2 \mathbb{E}[\|\rho_{t-1}\|_2^2 | x_{t-1}] - 2\lambda_t (x_{t-1} - x_\star)^\intercal \nabla F(x_{t-1}) \\
&\leq \|x_{t-1} - x_\star\|_2^2 + \lambda_t^2 C'_{\mathcal{D}} - 2\lambda_t (F(x_{t-1}) - F(x_\star) + \frac{\mu}{2}\|x_{t-1} - x_\star\|_2^2). \quad (5.24)
\end{aligned}$$

The third line follows from the Proposition 5.4.3 and the fourth line follows from Proposition 5.4.6 and strong convexity. Now we have

$$\mathbb{E}[F(x_{t-1})] - F(x_\star) \leq \frac{\lambda_t C'_{\mathcal{D}}}{2} + \frac{\lambda_t^{-1} - \mu}{2}\mathbb{E}\|x_{t-1} - x_\star\|_2^2 - \frac{\lambda_t^{-1}}{2}\mathbb{E}\|x_t - x_\star\|_2^2. \quad (5.25)$$

Finally, with $\lambda_t = \frac{2}{\mu(t+1)}$, it follows from the convexity of $F(\cdot)$ that

$$\begin{aligned}
0 \leq \mathbb{E}[F(\tilde{x}_T)] - F(x_\star) &\leq \frac{2}{(T)(T+1)} \sum_{t=0}^{T-1} (t+1)\big(\mathbb{E}[F(x_t)] - F(x_\star)\big) \\
&\leq \frac{2}{(T)(T+1)} \sum_{t=0}^{T-1} \frac{t+1}{t+2} \frac{C'_{\mathcal{D}}}{\mu} + \frac{\mu}{4}\big((t)(t+1)\mathbb{E}\|x_{t-1} - x_\star\|_2^2 - (t+1)(t+2)\mathbb{E}\|x_t - x_\star\|_2^2\big) \\
&\leq \frac{2C'_{\mathcal{D}}}{\mu(T+1)} - \frac{\mu}{2}\mathbb{E}\|x_T - x_\star\|_2^2. \quad (5.26)
\end{aligned}$$

The last inequality implies that both $\mathbb{E}\|x_T - x_\star\|_2^2 \leq \frac{4C'_{\mathcal{D}}}{\mu^2(T+1)}$ and $\mathbb{E}\|\tilde{x}_T - x_\star\|_2^2 \leq \frac{4C'_{\mathcal{D}}}{\mu^2(T+1)}$ (using strong convexity).

When $F(\cdot)$ is non-strongly convex, we can use the convexity of $F(\cdot)$ so that the last inequality

of (5.24) becomes

$$\mathbb{E}[\|x_t - x_\star\|^2 | x_{t-1}] \leq \|x_{t-1} - x_\star\|_2^2 + \lambda_t^2 C'_{\mathcal{D}} - 2\lambda_t (F(x_{t-1}) - F(x_\star)), \tag{5.27}$$

Thus we have

$$\mathbb{E}[F(x_{t-1})] - F(x_\star) \leq \frac{\lambda_t C'_{\mathcal{D}}}{2} + \frac{\lambda_t^{-1}}{2}\mathbb{E}\|x_{t-1} - x_\star\|_2^2 - \frac{\lambda_t^{-1}}{2}\mathbb{E}\|x_t - x_\star\|_2^2. \tag{5.28}$$

Finally, with $\lambda_t = \frac{c}{\sqrt{(t+1)}}$, it follows from the convexity of $F(\cdot)$ and the assumption that $\mathbb{E}\|x_t - x_\star\|_2^2 \leq D$ that

$$\begin{aligned}
0 \leq \mathbb{E}[F(\tilde{x}_T)] - F(x_\star) &\leq \frac{2}{(T)(T+1)} \sum_{t=0}^{T-1} (t+1)(\mathbb{E}[F(x_t)] - F(x_\star)) \\
&\leq \frac{\sqrt{2}}{2c(T)(T+1)} D + \frac{2}{(T)(T+1)} \sum_{t=0}^{T-1} (t+1) \frac{c C'_{\mathcal{D}}}{2\sqrt{t+2}} \\
&\quad + \sum_{t=1}^{T-1} \left(\frac{\sqrt{t+2}(t+1)}{2c} - \frac{\sqrt{t+1}(t)}{2c}\right) \mathbb{E}\|x_t - x_\star\|_2^2 \\
&\leq \frac{\sqrt{2}}{2c(T)(T+1)} D + \frac{2}{(T)(T+1)} \sum_{t=0}^{T-1} \sqrt{t+1} \frac{c C'_{\mathcal{D}}}{2} + \sum_{t=1}^{T-1} \frac{\sqrt{t+1}}{2c} \left(\frac{3t+2}{\sqrt{(t+2)(t+1)} + t}\right) D \\
&\leq \frac{\sqrt{2}}{2c(T)(T+1)} D + \frac{2}{(T)(T+1)} (T+1)^{\frac{3}{2}} \left(\frac{c C'_{\mathcal{D}}}{2} + \frac{3D}{2c}\right) \leq \frac{2\sqrt{2} C'_{\mathcal{D}} + c^{-1} 4\sqrt{2} D}{\sqrt{T}}
\end{aligned} \tag{5.29}$$

□

Corollary 5.4.10. *The iteration complexity of Algorithm 9 is $O(\epsilon^{-1})$ when $F(\cdot)$ is μ-strongly convex and the iteration complexit is $O(\epsilon^{-2})$ when $F(\cdot)$ is non-strongly convex.*

5.4.4 Lipschitz Continuity of the Simulated Variance Reduced Gradient

In this subsection, we will present the convergence properties of the Simulated Variance Reduced Gradient (SVRG) algorithm. In contrast to the stochastic variance reduced gradient algo-

rithm for the ERM problem (5.2) , the property that

$$\mathbb{E}\|\nabla f_i(x) - f_i(\tilde{x}) + \nabla F_n(\tilde{x})\|_2^2 \leq 4L\{F_n(x) - F_n(x_\star) + F_n(\tilde{x}) - F_n(x_\star)\},$$

where i is uniformly sampled from $\{1, \ldots, n\}$ and L is the Lipschitz constant for $\nabla F_n(x)$, may no longer hold because of the variance introduced by the simulation procedure. Instead, we establish a Lipschitz continuity property for the output $W = W(x, v_1) - W(\tilde{x}, v_1) + G(\tilde{x})$, where $G(\tilde{x})$ can be full gradient or a subsampled gradient at $\tilde{x}$, from Algorithm 10 that is important in the proof of the convergence rate of Algorithms 11 and 12. We need the following lemma to prove the results.

Lemma 5.4.11. *For all $n \geq 1$, we have*

$$\mathbb{E}\Big[\sup_{x\in\mathcal{D}}|[\overline{\nabla g}(x;1,n)]_{kj} - [\mathbb{E}_w \nabla g_w(x)]_{kj}|^4\Big] \leq C_1(\frac{log(4n^2)}{n})^2 \tag{5.30}$$

$$\mathbb{E}\Big[\sup_{x\in\mathcal{D}}|[\bar{g}(x;1,n)]_h - [\mathbb{E}_w g_w(x)]_h|^4\Big] \leq C_0(\frac{log(4n^2)}{n})^2 \tag{5.31}$$

$$\mathbb{E}\Big[\sup_{x\in\mathcal{D}}|[\overline{\nabla^2 g}(x;1,n)]_{kij} - [\mathbb{E}_w \nabla^2 g_w(x)]_{kij}|^4\Big] \leq C_2(\frac{log(4n^2)}{n})^2 \tag{5.32}$$

for any $n \geq 1$, $1 \leq k, h \leq d$ and $1 \leq i, j \leq p$, where $C_1 = 8l_{g,1}^{4-p}(4diam(\mathcal{D})^p)p^{p/2}L_{g,1}^p + 64l_{g,1}^4(p+1)^2$, $C_0 = 8l_{g,0}^{4-p}(4diam(\mathcal{D})^p)p^{p/2}L_{g,0}^p + 64l_{g,0}^4(p+1)^2$ and $C_2 = 8l_{g,2}^{4-p}(4diam(\mathcal{D})^p)p^{p/2}L_{g,2}^p + 64l_{g,2}^4(p+1)^2$.

We also need the following ancillary functions to develop our theory. For $x \in \mathcal{H} \subset \mathbb{R}^{d\times p}$, $y \in \mathcal{G} \subset \mathbb{R}^d$ and $z \in \mathcal{J} \subset \mathbb{R}^{d\times p\times p}$, for every $1 \leq i \leq p$ and $1 \leq j \leq p$, define $J(x, y, z) : \mathcal{H} \times \mathcal{G} \times \mathcal{J} \to \mathbb{R}^{p\times p}$ that

$$[J]_{ij}(x, y, z) = z_{:ij}^\top \nabla f_v(y) + [x]_{i:}\nabla^2 f_v(y)[x]_{:j}.$$

Lemma 5.4.12. *$[J]_{ij}(x, y, z)$ has Lipschitz continuous gradient with constant L_J; that is, for*

$x_1, x_2 \in \mathcal{H}$, $y_1, y_2 \in \mathcal{G}$ *and* $z_1, z_2 \in \mathcal{J}$,

$$\|\nabla[J]_{ij}(x_1, y_1, z_1) - \nabla[J]_{ij}(x_2 y_2, z_2)\|_F \leq L_J \|vec(x_1, y_1, z_1) - vec(x_2, y_2, z_1)\|_2,$$

where

$$L_J = \{12d^2 L_{f,2}^2 l_{g,1}^2 + 4d(\sqrt{d}_{g,2} L_{f,2} + d^2 l_{g,1}^2 L_{f,3})^2 + dL_{f,1}^2 + 4d^2 l_{f,2}^2 L_{g,2}^2 + 2d^2 l_{f,2}^2 + 4d^3 l_{f,3}^2\}^{1/2}.$$

Proof. Note that

$$\begin{aligned}
[J]_{ij}(x, y, z) &= z_{:ij}^\top \nabla f_v(y) + [x]_{i:} \nabla^2 f_v(y) [x]_{:j} \\
&= \sum_{k=1}^{d} \Big([z]_{kij} \frac{\partial f_{v_1}}{\partial [y]_k}(y) + [x]_{ki} \Big(\sum_{h=1}^{d} \frac{\partial f_{v_1}}{\partial [y]_k \partial [y]_h}(y) [x]_{hj} \Big) \Big).
\end{aligned}$$

We can then compute each component of the gradient $\nabla[J]_{ij}(x, y, z) \in \mathbb{R}^{(d\times p)\times d\times(d\times p\times p)}$ as

$$\begin{aligned}
\frac{\partial [J]_{ij}}{\partial [x]_{k'j'}}(x, y, z) &= \delta_{ij'} \sum_{h=1}^{d} \frac{\partial f_{v_1}}{\partial [y]_{k'} \partial [y]_h}(y) [x]_{hj} + \delta_{jj'} \sum_{k=1}^{d} \frac{\partial f_{v_1}}{\partial [y]_k \partial [y]_{k'}}(y) [x]_{ki} \\
&= \delta_{ij'} [\nabla^2 f_{v_1}]_{k':}(y) [x]_{:j} + \delta_{jj'} [\nabla^2 f_{v_1}]_{k':}(y) x_{:i} \\
\frac{\partial [J]_{ij}}{\partial [y]_{h'}}(x, y, z) &= \sum_{k=1}^{d} \Big([z]_{kij} \frac{\partial f_{v_1}}{\partial [y]_k \partial [y]_{h'}}(y) + [x]_{ki} \Big(\sum_{h=1}^{d} \frac{\partial f_{v_1}}{\partial [y]_k \partial [y]_h \partial [y]_{h'}}(y) [x]_{hj} \Big) \Big) \\
&= [z]_{:ij}^\top [\nabla^2 f_{v_1}(y)]_{:h'} + [x]_{:i}^\top [\nabla^3 f_{v_1}(y)]_{::h'} [x]_{:j} \\
\frac{\partial [J]_{ij}}{\partial [z]_{k''i''j''}}(x, y, z) &= \delta_{ii''} \delta_{jj''} \frac{\partial f_{v_1}}{\partial [y]_{k''}}(y) = \delta_{ii''} \delta_{jj''} [\nabla f_{v_1}(y)]_{k''}.
\end{aligned}$$

where $1 \leq i', j', i'', j'' \leq p$,$1 \leq k', h', k'' \leq d$ and δ_{ij} is the Kronecker delta. Note that by

Assumptions 1, 2, 4, and 5, we have

$$
\begin{aligned}
&|\frac{\partial [J]_{ij}}{\partial [x]_{k'j'}}(x_1, y_1, z_1) - \frac{\partial [J]_{ij}}{\partial [x]_{k'j'}}(x_2, y_2, z_2)| \\
&\leq \delta_{ij'}|[\nabla^2 f_{v_1}]_{k':}(y_1)[x_1]_{:j} - [\nabla^2 f_{v_1}]_{k':}(y_2)[x_2]_{:j}| + \delta_{jj'}|[\nabla^2 f_{v_1}]_{k':}(y_1)[x_1]_{:i} - [\nabla^2 f_{v_1}]_{k':}(y_2)[x_2]_{:i}| \\
&\leq \delta_{ij'}\sqrt{d}\{l_{f,2}\|[x_1]_{:j} - [x_2]_{:j}\|_2 + L_{f,2}l_{g,1}\|y_1 - y_2\|_2\} + \delta_{jj'}\sqrt{d}\{l_{f,2}\|[x_1]_{:i} - [x_2]_{:i}\|_2 \\
&\quad + L_{f,2}l_{g,1}\|y_1 - y_2\|_2\} \\
&= (\delta_{ij'} + \delta_{jj'})\sqrt{d}L_{f,2}l_{g,1}\|y_1 - y_2\|_2 + \delta_{ij'}\sqrt{d}l_{f,2}\|[x_1]_{:j} - [x_2]_{:j}\|_2 + \delta_{jj'}\sqrt{d}l_{f,2}\|[x_1]_{:i} - [x_2]_{:i}\|_2
\end{aligned}
$$

$$
\begin{aligned}
&|\frac{\partial [J]_{ij}}{\partial [y]_{h'}}(x_1, y_1, z_1) - \frac{\partial [J]_{ij}}{\partial [y]_{h'}}(x_2, y_2, z_2)| \\
&\leq |[z_1]_{:ij}^\top [\nabla^2 f_{v_1}(y_1)]_{:h'} - [z_2]_{:ij}^\top [\nabla^2 f_{v_1}(y_2)]_{:h'}| \\
&\quad + |[x_1]_{:i}^\top [\nabla^3 f_{v_1}(y_1)]_{::h'}[x_1]_{:j} - [x_2]_{:i}^\top [\nabla^3 f_{v_1}(y_2)]_{::h'}[x_2]_{:j}| \\
&\leq \sqrt{d}l_{g,2}L_{f,2}\|y_1 - y_2\|_2 + \sqrt{d}l_{f,2}L_{g,2}\|[z_1]_{:ij} - [z_2]_{:ij}\|_2 + dl_{g,1}^2 L_{f,3}\|y_1 - y_2\|_2 \\
&\quad + dl_{g,1}l_{f,3}\|[x_1]_{:j} - [x_2]_{:j}\|_2 + dl_{g,1}l_{f,3}\|[x_1]_{:i} - [x_2]_{:i}\|_2 \\
&= (\sqrt{d}l_{g,2}L_{f,2} + dl_{g,1}^2 L_{f,3})\|y_1 - y_2\|_2 + \sqrt{d}l_{f,2}L_{g,2}\|[z_1]_{:ij} - [z_2]_{:ij}\|_2 \\
&\quad + dl_{g,1}l_{f,3}\|[x_1]_{:j} - [x_2]_{:j}\|_2 + dl_{g,1}l_{f,3}\|[x_1]_{:i} - [x_2]_{:i}\|_2
\end{aligned}
$$

$$
\begin{aligned}
&|\frac{\partial [J]_{ij}}{\partial [z]_{k''i''j''}}(x_1, y_1, z_1) - \frac{\partial [J]_{ij}}{\partial [z]_{k''i''j''}}(x_2, y_2, z_2)| \\
&\leq |\delta_{ii''}\delta_{jj''}[\nabla f_{v_1}(y_1)]_{k''} - \delta_{ii''}\delta_{jj''}[\nabla f_{v_1}(y_1)]_{k''}| \\
&\leq \delta_{ii''}\delta_{jj''}L_{f,1}\|y_1 - y_2\|_2.
\end{aligned}
$$

Note that

$$
\begin{aligned}
&\|\nabla[J]_{ij}(x_1, y_1, z_1) - \nabla[J]_{ij}(x_2, y_2, z_2)\|_F^2 \\
&= \sum_{k'=1}^{d}\sum_{j'=1}^{p} |\frac{\partial[J]_{ij}}{\partial[x]_{k'j'}}(x_1, y_1, z_1) - \frac{\partial[J]_{ij}}{\partial[x]_{k'j'}}(x_2, y_2, z_2)|^2 \\
&\quad + \sum_{h'=1}^{d} |^2\frac{\partial[j]_{ij}}{\partial[y]_{h'}}(x_1, y_1, z_1) - \frac{\partial[j]_{ij}}{\partial[y]_{h'}}(x_2, y_2, z_2)|^2 \\
&\quad + \sum_{k''=1}^{d}\sum_{i''=1}^{p}\sum_{j''=1}^{p} |\frac{\partial[J]_{ij}}{\partial[z]_{k''i''j''}}(x_1, y_1, z_1) - \frac{\partial[J]_{ij}}{\partial[z]_{k''i''j''}}(x_2, y_2, z_2)|^2.
\end{aligned}
$$

Then based on our previous computation and using (5.14), we have

$$
\begin{aligned}
&\sum_{k'=1}^{d}\sum_{j'=1}^{p} |\frac{\partial[J]_{ij}}{\partial[x]_{k'j'}}(x_1, y_1, z_1) - \frac{\partial[J]_{ij}}{\partial[x]_{k'j'}}(x_2, y_2, z_2)|^2 \\
&\leq \sum_{k'=1}^{d}\sum_{j'=1}^{p} 3\{(2\delta_{ij'} + 2\delta_{jj'})dL_{f,2}^2 l_{g,1}^2\|y_1 - y_2\|_2^2 + \delta_{ij'}dl_{f,2}^2\|[x_1]_{:j} - [x_2]_{:j}\|_2^2 \\
&\quad + \delta_{jj'}dl_{f,2}^2\|[x_1]_{:i} - [x_2]_{:i}\|_2^2\} \\
&= 12d^2L_{f,2}^2 l_{g,1}^2\|y_1 - y_2\|_2^2 + d^2l_{f,2}^2\|[x_1]_{:j} - [x_2]_{:j}\|_2^2 + d^2l_{f,2}^2\|[x_1]_{:i} - [x_2]_{:i}\|_2^2\},
\end{aligned}
$$

$$
\begin{aligned}
&\sum_{h'=1}^{d} |\frac{\partial[j]_{ij}}{\partial[y]_{h'}}(x_1, y_1, z_1) - \frac{\partial[j]_{ij}}{\partial[y]_{h'}}(x_2, y_2, z_2)|^2 \\
&\leq 4d(\sqrt{d}l_{g,2}L_{f,2} + dl_{g,1}^2L_{f,3})^2\|y_1 - y_2\|_2^2 + 4d^2l_{f,2}^2L_{g,2}^2\|[z_1]_{:ij} - [z_2]_{:ij}\|_2^2 \\
&\quad + 4d^3l_{f,3}^2\|[x_1]_{:j} - [x_2]_{:j}\|_2^2 + 4d^3l_{g,1}^2l_{f,3}^2\|[x_1]_{:i} - [x_2]_{:i}\|, \text{ and}
\end{aligned}
$$

$$
\sum_{k''=1}^{d}\sum_{i''=1}^{p}\sum_{j''=1}^{p} |\frac{\partial[J]_{ij}}{\partial[z]_{k''i''j''}}(x_1, y_1, z_1) - \frac{\partial[J]_{ij}}{\partial[z]_{k''i''j''}}(x_2, y_2, z_2)|^2 \leq dL_{f,1}^2\|y_1 - y_2\|_2^2.
$$

Therefore

$$
\begin{aligned}
&\|\nabla[J]_{ij}(x_1, y_1, z_1) - \nabla[J]_{ij}(x_2, y_2, z_2)\|_F^2 \\
\leq\ &\{12d^2L_{f,2}^2l_{g,1}^2 + 4d(\sqrt{d}_{g,2}L_{f,2} + d^2l_{g,1}^2L_{f,3})^2 + dL_{f,1}^2\}\|y_1 - y_2\|_2^2 + 4d^2l_{f,2}^2L_{g,2}^2\|[z_1]_{:ij} - [z_2]_{:ij}\|_2^2 \\
&+ (d^2l_{f,2}^2 + 4d^3l_{f,3}^2)\|[x_1]_{:j} - [x_2]_{:j}\|_2^2 + (d^2l_{f,2}^2 + 4d^3l_{f,3}^2)\|[x_1]_{:i} - [x_2]_{:i}\|_2^2 \\
\leq\ &\{12d^2L_{f,2}^2l_{g,1}^2 + 4d(\sqrt{d}_{g,2}L_{f,2} + d^2l_{g,1}^2L_{f,3})^2 + dL_{f,1}^2 + 4d^2l_{f,2}^2L_{g,2}^2 \\
&+ 2d^2l_{f,2}^2 + 4d^3l_{f,3}^2\}\|\mathrm{vec}(x_1 - x_2, y_1 - y_2, z_1 - z_2)\|_2^2 \\
=\ &L_J^2\|\mathrm{vec}(x_1 - x_2, y_1 - y_2, z_1 - z_2)\|_2^2
\end{aligned}
$$

□

Based on the ancillary function $J(x, y, z)$, for $x, x_0 \in \mathcal{H} \subset \mathbb{R}^{d\times p}$, $y, y_0 \in \mathcal{G} \subset \mathbb{R}^d$ and $z, z_0 \in \mathcal{J} \subset \mathbb{R}^{d\times p\times p}$, we define

$$
\begin{aligned}
&[R]_{ij}(x, x_0, y, y_0, z, z_0) \\
=\ &[J]_{ij}(x, y, z) - [J]_{ij}(x_0, y_0, z_0) - \{\nabla[J]_{ij}(x_0, y_0, z_0)\}[x - x_0, y - y_0, z - z_0],
\end{aligned}
$$

where

$$
\begin{aligned}
&\{\nabla[J]_{ij}(x_0, y_0, z_0)\}[x - x_0, y - y_0, z - z_0] \\
=\ &(\mathrm{vec}\{\nabla[J]_{ij}(x_0, y_0, z_0)\})^\top \mathrm{vec}(x - x_0, y - y_0, z - z_0) \\
=\ &\sum_{k'=1}^{d}\sum_{j'=1}^{d}\frac{\partial[J]_{ij}}{\partial[x]_{k'j'}}(x_0, y_0, z_0)([x]_{k'j'} - [x_0]_{k'j'}) + \sum_{h'=1}^{d}\frac{\partial[J]_{ij}}{\partial[y]_{h'}}([y]_{h'} - [y_0]_{h'}) \\
&+ \sum_{k''=1}^{d}\sum_{i''=1}^{p}\sum_{j''=1}^{p}\frac{\partial[J]_{ij}}{\partial[z]_{k''i''j''}}(x_0, y_0, z_0)([z]_{k''i''j''} - [z_0]_{k''i''j''}).
\end{aligned}
$$

Lemma 5.4.13. *For all $x, x_0 \in \mathcal{H}$, $y, y_0 \in \mathcal{G}$ and $z, z_0 \in \mathcal{J}$, we have*

$$
|[R]_{ij}(x, x_0, y, y_0, z, z_0)| \leq \frac{L_J}{2}\|vec(x, y, z) - vec(x_0, y_0, z_0)\|_2^2,
$$

where

$$L_J = \{12d^2L_{f,2}^2l_{g,1}^2 + 4d(\sqrt{d}_{g,2}L_{f,2} + d^2l_{g,1}^2L_{f,3})^2 + dL_{f,1}^2 + 4d^2l_{f,2}^2L_{g,2}^2 + 2d^2l_{f,2}^2 + 4d^3l_{f,3}^2\}^{1/2}.$$

Proof. This result is a direct consequence of Lemma 5.4.1. □

Now we proceed with the main lemma of this section. This lemma will be used for proving convergence results for our SimVRG (Algorithm 11) and SCSimG (Algorithm 12) algorithms.

Lemma 5.4.14. *There exist a constant $C_{\mathcal{D}} < \infty$ such that for any $v_1 \in \Omega_v$ and $x, \tilde{x} \in \mathcal{D}$, $W(x, v_1)$ and $W(\tilde{x}, v_1)$ from the variance reduced unbiased gradient $W(x, \tilde{x}, v_1) = W(x, v_1) - W(\tilde{x}, v_1) + \nabla G(\tilde{x})$ in Algorithm 10 satisfies*

$$\mathbb{E}\|W(x, v_1) - W(\tilde{x}, v_1)\|_2^2 \leq C_{\mathcal{D}}\|x - \tilde{x}\|_2^2, \tag{5.33}$$

where

$$\begin{aligned} C_{\mathcal{D}} = {} & 4p^2d^2f_{g,2}^2l_{f,1}^2 + 4p^2d^4l_{g,1}^4l_{f,2}^2 \\ & + 9L_J^2p^2(C_0 + C_1 + C_2)\left(\frac{(n_0+1)^2}{1-2^{\gamma-2}} + \frac{2(n_0+1)2^{\gamma-2}}{(1-2^{\gamma-2})^2} + \frac{2^{3\gamma-6}+2^{\gamma-2}}{(1-2^{\gamma-2})^3}\right). \end{aligned}$$

Proof. Fixing $v_1 \in \Omega_v$ and $x, \tilde{x} \in \mathcal{D}$, we have

$$W(x, v_1) - W(\tilde{x}, v_1) = \frac{1}{\tilde{p}_N}\Big(Y_1(x) - Y_1(\tilde{x}) - \frac{1}{2}\Big(Y_2(x) - Y_2(\tilde{x}) + Y_3(x) - Y_3(\tilde{x})\Big)\Big) + Y_4(x) - Y_4(\tilde{x}).$$

Similar to the proof of Proposition 5.4.6, we first take expectation with respect to N. Then,

$$\begin{aligned}
&\mathbb{E}\|W(x,v_1)-W(\tilde{x},v_1)\|_2^2 = \sum_{n=0}^{\infty}\mathbb{E}\{\|W(x,v_1)-W(\tilde{x},v_1)\|_2^2|N=n\}\tilde{p}_n \\
&= \sum_{n=0}^{\infty}\sum_{i=1}^{p}\mathbb{E}\big(\{[W(x,v_1)]_i-[W(\tilde{x},v_1)]_i\}^2|N=n\big)\tilde{p}_n \le \sum_{i=1}^{p}2\mathbb{E}\{[Y_4(x)]_i-[Y_4(\tilde{x})]_i\}^2 \\
&\quad+\sum_{n=0}^{\infty}\sum_{i=1}^{p}\frac{2}{\tilde{p}_n}\mathbb{E}\{\big([Y_1(x)]_i-[Y_1(\tilde{x})]_i-0.5\{[Y_2(x)]_i-[Y_2(\tilde{x})]_i+[Y_3(x)]_i-[Y_3(\tilde{x})]_i\}\big)^2|N=n\},
\end{aligned}$$

where the last inequality comes from (5.14). Since $[Y_4(\cdot)]_i$ and $[Y_1(\cdot)]_i-0.5\{[Y_2(\cdot)]_i+[Y_3(\cdot))]_i\}$ are continuous for every $1\le i\le p$, the mean value theorem implies that there exist ζ_i and ξ_i that lie between x and $\tilde{x}$ such that $[Y_4(x)]_i-[Y_4(\tilde{x})]_i=\nabla[Y_4(\zeta_i)]_i^\top(x-\tilde{x})$ and

$$\begin{aligned}
&\big([Y_1(x)]_i-0.5\{[Y_2(x)]_i+[Y_3(x)]_i\}\big)-\big([Y_1(\tilde{x})]_i-0.5\{[Y_2(\tilde{x})]_i+[Y_3(\tilde{x})]_i\}\big) \\
&=\{\nabla\big([Y_1(\xi_i)]_i-0.5\{[Y_2(\xi_i)]_i+[Y_3(\xi_i)]_i\}\big)\}^\top(x-\tilde{x}).
\end{aligned}$$

Therefore, we may write

$$\begin{aligned}
&\mathbb{E}\|W(x,v_1)-W(\tilde{x},v_1)\|_2^2 = \sum_{i=1}^{p}2\mathbb{E}\{\nabla[Y_4(\zeta_i)]_i^\top(x-\tilde{x})\}^2 \\
&\quad+\sum_{n=0}^{\infty}\sum_{i=1}^{p}\frac{2}{\tilde{p}_n}\mathbb{E}\Big(\Big\{\nabla\big([Y_1(\xi_i)]_i-0.5\{[Y_2(\xi_i)]_i+[Y_3(\xi_i)]_i\}\big)^\top(x-\tilde{x})\Big\}^2|N=n\Big) \\
&\le\sum_{i=1}^{p}2\|x-\tilde{x}\|_2^2\mathbb{E}\|\nabla[Y_4(\zeta_i)]_i\|_2^2 \\
&\quad+\sum_{n=0}^{\infty}\sum_{i=1}^{p}\frac{2\|x-\tilde{x}\|_2^2}{\tilde{p}_n}\mathbb{E}\Big\{\|\nabla\big([Y_1(\xi_i)]_i-0.5\{[Y_2(\xi_i)]_i+[Y_3(\xi_i)]_i\}\big)\|_2^2|N=n\Big\} \\
&=\sum_{i=1}^{p}2\|x-\tilde{x}\|_2^2\mathbb{E}\|\nabla[Y_4(\zeta_i)]_i\|_2^2 \\
&\quad+\sum_{n=0}^{\infty}\sum_{i=1}^{p}\sum_{j=1}^{p}\frac{2\|x-\tilde{x}\|_2^2}{\tilde{p}_n}\mathbb{E}\Big\{\|[\nabla Y_1(\xi_i)]_{ij}-0.5\{[\nabla Y_2(\xi_i)]_{ij}+[\nabla Y_3(\xi_i)]_{ij}\}\|_2^2|N=n\Big\}, \qquad (5.34)
\end{aligned}$$

where the last inequality uses the Cauchy-Shwartz inequality. Next, we first obtain an upper bound for $\mathbb{E}\|\nabla[Y_4(\zeta_i)]_i\|_2^2$ and then bound $\mathbb{E}\Big\{\|\nabla([Y_1(\xi_i)]_i - 0.5\{[Y_2(\xi_i)]_i + [Y_3(\xi_i)]_i\})\|_2^2 | N = n\Big\}$ using a function of n in order to analyze the infinite sum above.

To obtain an upper bound for $\mathbb{E}\|[\nabla Y_4(\zeta_i)]_i\|_2^2$, we first note that

$$\begin{aligned}
&\nabla\{[Y_4(\zeta_i)]_i\} \\
&= \{[\overline{\nabla^2 g}(x; 1, 2^{n_0})]_{::i}\}^\top \nabla f_{v_1}\{\bar{g}(x; 1, 2^{n_0})\} + \{\overline{\nabla g}(x; 1, 2^{n_0})\}^\top \nabla^2 f_{v_1}(\bar{g}(x; 1, 2^{n_0}))[\overline{\nabla g}(x; 1, 2^{n_0})]_{i:}.
\end{aligned}$$

Therefore by (5.14),

$$\begin{aligned}
\|\nabla\{[Y_4(\zeta_i)]_i\}\|_2^2 &\leq 2\|\{[\overline{\nabla^2 g}(x; 1, 2^{n_0})]_{::i}\}^\top \nabla f_{v_1}\{\bar{g}(x; 1, 2^{n_0})\}\|_2^2 \\
&\quad + 2\|\{\overline{\nabla g}(x; 1, 2^{n_0})\}^\top \nabla^2 f_{v_1}(\bar{g}(x; 1, 2^{n_0}))[\overline{\nabla g}(x; 1, 2^{n_0})]_{i:}\|_2^2 \\
&\leq 2\|[\overline{\nabla^2 g}(x; 1, 2^{n_0})]_{::i}\|_F^2 \|\nabla f_{v_1}\{\bar{g}(x; 1, 2^{n_0})\}\|_2^2 \\
&\quad + 2\|\overline{\nabla g}(x; 1, 2^{n_0})\|_F^2 \|\nabla^2 f_{v_1}(\bar{g}(x; 1, 2^{n_0}))\|_F^2 \|[\overline{\nabla g}(x; 1, 2^{n_0})]_{i:}\|_2^2.
\end{aligned}$$

By Assumptions 4 and 5,

$$\begin{aligned}
&\|[\overline{\nabla^2 g}(\zeta_i; 1, 2^{n_0})]_{::i}\|_F^2 \leq pd\|[\overline{\nabla^2 g}(\zeta_i; 1, 2^{n_0})]_{::i}\|_\infty^2 \leq pdl_{g,2}^2, \\
&\|\nabla f_{v_1}\{\bar{g}(\zeta_i; 1, 2^{n_0})\}\|_2^2 \leq p\|f_{v_1}\{\bar{g}(\zeta_i; 1, 2^{n_0})\}\|_\infty^2 \leq dl_{f,1}^2, \\
&\|\overline{\nabla g}(\zeta_i; 1, 2^{n_0})\|_F^2 \leq pd\|\overline{\nabla g}(\zeta_i; 1, 2^{n_0})\|_\infty^2 \leq pdl_{g,1}^2, \\
&\|\nabla^2 f_{v_1}(\bar{g}(\zeta_i; 1, 2^{n_0}))\|_F^2 \leq d^2\|\nabla^2 f_{v_1}(\bar{g}(\zeta_i; 1, 2^{n_0}))\|_\infty^2 \leq d^2 l_{f,2}^2, \text{ and} \\
&\|[\overline{\nabla g}(\zeta_i; 1, 2^{n_0})]_{i:}\|_2^2 \leq d\|[\overline{\nabla g}(\zeta_i; 1, 2^{n_0})]_{i:}\|_\infty^2 \leq dl_{g,1}^2.
\end{aligned}$$

Therefore

$$\|\nabla\{[Y_4(\zeta_i)]_i\}\|_2^2 \leq 2pd^2 f_{g,2}^2 l_{f,1}^2 + 2pd^4 l_{g,1}^4 l_{f,2}^2.$$

Hence

$$2\|x-\tilde{x}\|_2^2\mathbb{E}\|\nabla[Y_4(\zeta_i)]_i\|_2^2 \leq \{4pd^2f_{g,2}^2l_{f,1}^2 + 4pd^4l_{g,1}^4l_{f,2}^2\}\|x-\tilde{x}\|_2^2. \tag{5.35}$$

To bound the second term in (5.34), we let $\bar{n}_0 = n + n_0$ and $\bar{n}_0^+ = n + n_0 + 1$ and note that conditioned on $N = n$,

$$\begin{aligned}
[\nabla Y_1(\xi_i)]_{ij} &= [J]_{ij}\{\overline{\nabla g}(\xi_i; 1, 2^{\bar{n}_0^+}), \bar{g}(\xi_i; 1, 2^{\bar{n}_0^+}), \overline{\nabla^2 g}(\xi_i; 1, 2^{\bar{n}_0^+})\} \\
&= [J]_{ij}\{\mathbb{E}_w\nabla g_w(\xi_i), \mathbb{E}_w g_w(\xi_i), \mathbb{E}_w\nabla^2 g_w(\xi_i)\} \\
&\quad + \nabla[J]_{ij}\{\mathbb{E}_w\nabla g_w(\xi_i), \mathbb{E}_w g_w(\xi_i), \mathbb{E}_w\nabla^2 g_w(\xi_i)\}[\overline{\nabla g}(\xi_i; 1, 2^{\bar{n}_0^+}) - \mathbb{E}_w\nabla g_w(\xi_i), \\
&\quad \bar{g}(\xi_i; 1, 2^{\bar{n}_0^+}) - \mathbb{E}_w g_w(\xi_i), \overline{\nabla^2 g}(\xi_i; 1, 2^{\bar{n}_0^+}) - \mathbb{E}_w\nabla^2 g_w(\xi_i)] + \\
&\quad R\{\overline{\nabla g}(\xi_i; 1, 2^{\bar{n}_0^+}), \mathbb{E}_w\nabla g_w(\xi_i), \bar{g}(\xi_i; 1, 2^{\bar{n}_0^+}), \mathbb{E}_w g_w(\xi_i), \overline{\nabla^2 g}(\xi_i; 1, 2^{\bar{n}_0^+}), \mathbb{E}_w\nabla^2 g_w(\xi_i)\},
\end{aligned}$$

$$\begin{aligned}
[\nabla Y_2(\xi_i)]_{ij} &= [J]_{ij}\{\overline{\nabla g}(\xi_i; 1, 2^{\bar{n}_0}), \bar{g}(\xi_i; 1, 2^{\bar{n}_0}), \overline{\nabla^2 g}(\xi_i; 1, 2^{\bar{n}_0})\} \\
&= [J]_{ij}\{\mathbb{E}_w\nabla g_w(\xi_i), \mathbb{E}_w g_w(\xi_i), \mathbb{E}_w\nabla^2 g_w(\xi_i)\} \\
&\quad + \nabla[J]_{ij}\{\mathbb{E}_w\nabla g_w(\xi_i), \mathbb{E}_w g_w(\xi_i), \mathbb{E}_w\nabla^2 g_w(\xi_i)\}[\overline{\nabla g}(\xi_i; 1, 2^{\bar{n}_0}) - \mathbb{E}_w\nabla g_w(\xi_i), \\
&\quad \bar{g}(\xi_i; 1, 2^{\bar{n}_0}) - \mathbb{E}_w g_w(\xi_i), \overline{\nabla^2 g}(\xi_i; 1, 2^{\bar{n}_0}) - \mathbb{E}_w\nabla^2 g_w(\xi_i)] + \\
&\quad R\{\overline{\nabla g}(\xi_i; 1, 2^{\bar{n}_0}), \mathbb{E}_w\nabla g_w(\xi_i), \bar{g}(\xi_i; 1, 2^{\bar{n}_0}), \mathbb{E}_w g_w(\xi_i), \overline{\nabla^2 g}(\xi_i; 1, 2^{\bar{n}_0}), \mathbb{E}_w\nabla^2 g_w(\xi_i)\}, \text{ and}
\end{aligned}$$

$$\begin{aligned}
[\nabla Y_3(\xi_i)]_{ij} &= [J]_{ij}\{\overline{\nabla g}(\xi_i; 2^{\bar{n}_0} + 1, 2^{\bar{n}_0^+}), \bar{g}(\xi_i; 2^{\bar{n}_0} + 1, 2^{\bar{n}_0^+}), \overline{\nabla^2 g}(\xi_i; 2^{\bar{n}_0} + 1, 2^{\bar{n}_0^+})\} \\
&\quad + \nabla[J]_{ij}\{\mathbb{E}_w\nabla g_w(\xi_i), \mathbb{E}_w g_w(\xi_i), \mathbb{E}_w\nabla^2 g_w(\xi_i)\}[\overline{\nabla g}(\xi_i; 2^{\bar{n}_0} + 1, 2^{\bar{n}_0^+}) - \mathbb{E}_w\nabla g_w(\xi_i), \\
&\quad \bar{g}(\xi_i; 2^{\bar{n}_0} + 1, 2^{\bar{n}_0^+}) - \mathbb{E}_w g_w(\xi_i), \ \overline{\nabla^2 g}(\xi_i; 2^{\bar{n}_0} + 1, 2^{\bar{n}_0^+}) - \mathbb{E}_w\nabla^2 g_w(\xi_i)] \\
&\quad + R\{\overline{\nabla g}(\xi_i; 2^{\bar{n}_0} + 1, 2^{\bar{n}_0^+}), \ \mathbb{E}_w\nabla g_w(\xi_i), \ \bar{g}(\xi_i; 2^{\bar{n}_0} + 1, 2^{\bar{n}_0^+}), \ \mathbb{E}_w g_w(\xi_i), \\
&\quad \overline{\nabla^2 g}(\xi_i; 2^{\bar{n}_0} + 1, 2^{\bar{n}_0^+}), \ \mathbb{E}_w\nabla^2 g_w(\xi_i)\}.
\end{aligned}$$

Therefore, condition on $N = n$, we have

$$
\begin{aligned}
&[\nabla Y_1(\xi_i)]_{ij} - 0.5\{[\nabla Y_2(\xi_i)]_{ij} + [\nabla Y_3(\xi_i)]_{ij}\} \\
&= R\{\overline{\nabla g}(\xi_i; 1, 2^{\bar{n}_0^+}), \mathbb{E}_w \nabla g_w(\xi_i), \bar{g}(\xi_i; 1, 2^{\bar{n}_0^+}), \mathbb{E}_w g_w(\xi_i), \overline{\nabla^2 g}(\xi_i; 1, 2^{\bar{n}_0^+}), \mathbb{E}_w \nabla^2 g_w(\xi_i)\} \\
&- \frac{1}{2} R\{\overline{\nabla g}(\xi_i; 1, 2^{\bar{n}_0}), \mathbb{E}_w \nabla g_w(\xi_i), \bar{g}(\xi_i; 1, 2^{\bar{n}_0}), \mathbb{E}_w g_w(\xi_i), \overline{\nabla^2 g}(\xi_i; 1, 2^{\bar{n}_0}), \mathbb{E}_w \nabla^2 g_w(\xi_i)\} \\
&- \frac{1}{2} R\{\overline{\nabla g}(\xi_i; 2^{\bar{n}_0} + 1, 2^{\bar{n}_0^+}), \ \mathbb{E}_w \nabla g_w(\xi_i), \ \bar{g}(\xi_i; 2^{\bar{n}_0} + 1, 2^{\bar{n}_0^+}), \\
&\qquad \mathbb{E}_w g_w(\xi_i), \overline{\nabla^2 g}(\xi_i; 2^{\bar{n}_0} + 1, 2^{\bar{n}_0^+}), \ \mathbb{E}_w \nabla^2 g_w(\xi_i)\}.
\end{aligned}
$$

Then, by (5.14)

$$
\begin{aligned}
&\mathbb{E}\Big\{\Big([\nabla Y_1(\xi_i)]_{ij} - 0.5\{[\nabla Y_2(\xi_i)]_{ij} + [\nabla Y_3(\xi_i)]_{ij}\}\Big)^2 | N = n\Big\} \\
&\le 3\mathbb{E}\Big(R\{\overline{\nabla g}(\xi_i; 1, 2^{\bar{n}_0^+}), \mathbb{E}_w \nabla g_w(\xi_i), \bar{g}(\xi_i; 1, 2^{\bar{n}_0^+}), \mathbb{E}_w g_w(\xi_i), \overline{\nabla^2 g}(\xi_i; 1, 2^{\bar{n}_0^+}), \mathbb{E}_w \nabla^2 g_w(\xi_i)\}^2\Big) \\
&\quad + \frac{3}{4}\mathbb{E}\Big(R\{\overline{\nabla g}(\xi_i; 1, 2^{\bar{n}_0}), \mathbb{E}_w \nabla g_w(\xi_i), \bar{g}(\xi_i; 1, 2^{\bar{n}_0}), \mathbb{E}_w g_w(\xi_i), \overline{\nabla^2 g}(\xi_i; 1, 2^{\bar{n}_0}), \mathbb{E}_w \nabla^2 g_w(\xi_i)\}^2\Big) \\
&+ \frac{3}{4}\mathbb{E}\Big(R\{\overline{\nabla g}(\xi_i; 2^{\bar{n}_0} + 1, 2^{\bar{n}_0^+}), \mathbb{E}_w \nabla g_w(\xi_i), \ \bar{g}(\xi_i; 2^{\bar{n}_0} + 1, 2^{\bar{n}_0^+}), \\
&\qquad \mathbb{E}_w g_w(\xi_i), \overline{\nabla^2 g}(\xi_i; 2^{\bar{n}_0} + 1, 2^{\bar{n}_0^+}), \ \mathbb{E}_w \nabla^2 g_w(\xi_i)\}^2\Big).
\end{aligned}
$$

Now, applying Lemma 5.4.13 on the three terms on the right-hand-side of the inequality above,

$$
\begin{aligned}
&\mathbb{E}\Big\{\Big([\nabla Y_1(\xi_i)]_{ij} - 0.5\{[\nabla Y_2(\xi_i)]_{ij} + [\nabla Y_3(\xi_i)]_{ij}\}\Big)^2 | N = n\Big\} \\
&\le \frac{3L_J^2}{4}\{\mathbb{E}\|\overline{\nabla g}(\xi_i; 1, 2^{n_0+n+1}) - \mathbb{E}_w \nabla g_w(\xi_i)\|_F^4 + \mathbb{E}\|\bar{g}(\xi_i; 1, 2^{n_0+n+1}) - \mathbb{E}_w g_w(\xi_i)\|_F^4 \\
&\quad + \mathbb{E}\|\overline{\nabla^2 g}(\xi_i; 1, 2^{n_0+n+1}) - \mathbb{E}_w g_w(\xi_i)\|_F^4\} + \frac{3L_J^2}{16}\{\mathbb{E}\|\overline{\nabla g}(\xi_i; 1, 2^{n_0+n}) - \mathbb{E}_w \nabla g_w(\xi_i)\|_F^4 + \\
&\mathbb{E}\|\bar{g}(\xi_i; 1, 2^{n_0+n}) - \mathbb{E}_w g_w(\xi_i)\|_F^4 + \mathbb{E}\|\overline{\nabla^2 g}(\xi_i; 1, 2^{n_0+n}) - \mathbb{E}_w \nabla^2 g_w(\xi_i)\|_F^4\} + \\
&\frac{3L_J^2}{16}\{\mathbb{E}\|\overline{\nabla g}(\xi_i; 2^{n_0+n} + 1, 2^{n_0+n+1}) - \mathbb{E}_w \nabla g_w(\xi_i)\|_F^4 \\
&\quad + \mathbb{E}\|\bar{g}(\xi_i; 2^{n_0+n} + 1, 2^{n_0+n+1}) - \mathbb{E}_w g_w(\xi_i)\|_F^4 \\
&\quad + \mathbb{E}\|\overline{\nabla^2 g}(\xi_i; 2^{n_0+n} + 1, 2^{n_0+n+1}) - \mathbb{E}_w \nabla^2 g_w(\xi_i)\|_F^4\}.
\end{aligned}
$$

Then applying Lemma 5.4.11 to the right-hand-side of the above inequality, we have

$$\begin{aligned}
&\mathbb{E}\Big\{\Big([\nabla Y_1(\xi_i)]_{ij} - 0.5\{[\nabla Y_2(\xi_i)]_{ij} + [\nabla Y_3(\xi_i)]_{ij}\}\Big)^2 | N = n\Big\} \\
&\leq \frac{3L_J^2}{4}\Big((C_0 + C_1 + C_2)\frac{\{\log(4^{n+n_0+1})\}^2}{4^{n+n_0+1}}\Big) + \frac{3L_J^2}{8}\Big((C_0 + C_1 + C_2)\frac{\{\log(4^{n_0+n})\}^2}{4^{n_0+n}}\Big) \\
&= \frac{3L_J^2(C_0 + C_1 + C_2)}{4^{n_0+n+1}}\{\frac{1}{4}\{\log(4^{n+n_0+1})\}^2 + \frac{1}{2}\{\log(4^{n_0+n})\}^2\} \\
&\leq \frac{3L_J^2(C_0 + C_1 + C_2)}{4^{n_0+n+1}}\{\frac{3}{2}(n_0 + n + 1)^2\}, \qquad (5.36)
\end{aligned}$$

where the last inequality is the result of $\log 4 < 2$.

Now we are ready to obtain a bound for (5.34). Using (5.35) and (5.36), we have

$$\begin{aligned}
&\mathbb{E}\|W(x, v_1) - W(\tilde{x}, v_1)\|_2^2 \\
&\leq \sum_{i=1}^{p} 2\|x - \tilde{x}\|_2^2 \mathbb{E}\|\nabla[Y_4(\zeta_i)]_i\|_2^2 \\
&+ \sum_{n=0}^{\infty}\sum_{i=1}^{p}\sum_{j=1}^{p}\frac{2\|x - \tilde{x}\|_2^2}{\tilde{p}_n}\mathbb{E}\Big\{\|[\nabla Y_1(\xi_i)]_{ij} - 0.5\{[\nabla Y_2(\xi_i)]_{ij} + [\nabla Y_3(\xi_i)]_{ij}\}\|_2^2 | N = n\Big\} \\
&\leq \sum_{i=1}^{p}\{4pd^2 f_{g,2}^2 l_{f,1}^2 + 4pd^4 l_{g,1}^4 l_{f,2}^2\}\|x - \tilde{x}\|_2^2 \\
&\quad + \sum_{n=0}^{\infty}\sum_{i=1}^{p}\sum_{j=1}^{p}\frac{2\|x - \tilde{x}\|_2^2}{\tilde{p}_n}\frac{3L_J^2(C_0 + C_1 + C_2)}{4^{n_0+n+1}}\{\frac{3}{2}(n_0 + n + 1)^2\} \\
&= \|x - \tilde{x}\|_2^2\Big\{4p^2d^2 f_{g,2}^2 l_{f,1}^2 + 4p^2 d^4 l_{g,1}^4 l_{f,2}^2 + 9L_J^2 p^2(C_0 + C_1 + C_2)\sum_{n=0}^{\infty}\frac{(n_0 + n + 1)^2}{\tilde{p}_n 4^{n_0+n+1}}\Big\}.
\end{aligned}$$

Since $\tilde{p}_n = (1 - 0.5^\gamma)0.5^{\gamma n}$ and $1 < \gamma < 2$, we have

$$\begin{aligned}
\sum_{n=0}^{\infty}\frac{(n_0 + n + 1)^2}{\tilde{p}_n 4^{n_0+n+1}} &= \frac{1}{(1 - 0.5^\gamma)4^{n_0+1}}\sum_{n=0}^{\infty}\frac{(n_0 + n + 1)^2}{2^{(2-\gamma)n}} \\
&= \frac{(n_0 + 1)^2}{1 - 2^{\gamma-2}} + \frac{2(n_0 + 1)2^{\gamma-2}}{(1 - 2^{\gamma-2})^2} + \frac{2^{3\gamma-6} + 2^{\gamma-2}}{(1 - 2^{\gamma-2})^3}.
\end{aligned}$$

Therefore

$$\mathbb{E}\|W(x, v_1) - W(\tilde{x}, v_1)\|_2^2 \leq C_{\mathcal{D}}\|x - \tilde{x}\|_2^2,$$

where

$$\begin{aligned} C_{\mathcal{D}} = & 4p^2d^2f_{g,2}^2l_{f,1}^2 + 4p^2d^4l_{g,1}^4l_{f,2}^2 \\ & + 9L_J^2p^2(C_0 + C_1 + C_2)\left(\frac{(n_0+1)^2}{1-2^{\gamma-2}} + \frac{2(n_0+1)2^{\gamma-2}}{(1-2^{\gamma-2})^2} + \frac{2^{3\gamma-6}+2^{\gamma-2}}{(1-2^{\gamma-2})^3}\right). \end{aligned}$$

□

5.4.5 Convergence of the Simulated Variance Reduced Gradient Algorithm

In this section we prove the convergence of Algorithm 11. We make use of the constant $C_{\mathcal{D}}$ defined in Lemma 5.4.14 and Assumption 3 that $F(\cdot)$ is μ-strongly convex.

Lemma 5.4.15. *Let $F : \mathbb{R}^p \to \mathbb{R}$ be a convex function with L-Lipschitz gradient and $x_\star = \arg\min_{x\in\mathbb{R}^p} F(x)$ be the global minimizer of $F(\cdot)$. Then for any $x \in \mathbb{R}^p$,*

$$\frac{1}{2L}\|\nabla F(x)\|_2^2 \leq F(x) - F(x_\star).$$

We omit the proof for this lemma since it is a well known result.

Theorem 5.4.16. *Consider Algorithm 11 with option II. Let λ be sufficiently small and M be sufficiently large so that*

$$\alpha = \frac{1}{\mu(1-\frac{4}{\mu}C_{\mathcal{D}}\lambda)\lambda M} + \frac{(\frac{4}{\mu}C_{\mathcal{D}} + 2L)\lambda}{1-\frac{4}{\mu}C_{\mathcal{D}}\lambda} < 1. \tag{5.37}$$

Then under Assumptions 1-5, we have geometric convergence in expectation for the SimVRG :

$$\mathbb{E}[F(\tilde{x}_s)] \leq F(x_\star) + \alpha^s[F(\tilde{x}_0) - F(x_\star)]$$

Proof. It follows from Lemma 5.4.15 that

$$\|\nabla F(x) - \nabla F(x_\star)\|_2^2 = \|\nabla F(x)\|_2^2 \leq 2L[F(x) - F(x_\star)] \tag{5.38}$$

. Now conditioning on x_t, we can take expectation with respect to $v_t \in \Omega_v$ to obtain

$$\begin{aligned}
\mathbb{E}[\|\rho_t\|_2^2 \mid x_t] \leq & 2\mathbb{E}[\|W(x_t, v_t) - W(\tilde{x}_s, v_t)\|_2^2 \mid x_t] + 2\nabla\|F(\tilde{x}_s)\|_2^2 \\
\leq & 2C_{\mathcal{D}}\|x_t - \tilde{x}_s\|_2^2 + 4L[F(\tilde{x}_s) - F(x_\star)] \\
\leq & 4C_{\mathcal{D}}(\|x_t - x_\star\|_2^2 + \|\tilde{x}_s - x_\star\|_2^2) + 4L[F(\tilde{x}_s) - F(x_\star)] \\
\leq & \frac{8}{\mu}C_{\mathcal{D}}[F(x_t) - F(x_\star)] + (\frac{8}{\mu}C_{\mathcal{D}} + 4L)[F(\tilde{x}_s) - F(x_\star)].
\end{aligned} \tag{5.39}$$

where the second inequality follows from Lemma 5.4.14 and equation (5.38). The last inequality follows from the strong convexity of $F(\cdot)$. Thus, by the contraction property of the projection operator $\Pi_{\mathcal{D}}$,

$$\begin{aligned}
& \mathbb{E}[\|x_{t+1} - x_\star\|_2^2 \mid x_t] \\
\leq & \|x_t - x_\star\|_2^2 - 2\lambda(x_t - x_\star)^\intercal \mathbb{E}[\rho_t \mid x_t] + \lambda^2\mathbb{E}[\|\rho_t\|_2^2 \mid x_t] \\
\leq & \|x_t - x_\star\|_2^2 - 2\lambda(x_t - x_\star)^\intercal \nabla F(x_t) + \frac{8}{\mu}C_{\mathcal{D}}\lambda^2[F(x_t) - F(x_\star)] + (\frac{8}{\mu}C_{\mathcal{D}} + 4L)\lambda^2[F(\tilde{x}_s) - F(x_\star)] \\
\leq & \|x_t - x_\star\|_2^2 - 2\lambda[F(x_t) - F(x_\star)] + \frac{8}{\mu}C_{\mathcal{D}}\lambda^2[F(x_t) - F(x_\star)] + (\frac{8}{\mu}C_{\mathcal{D}} + 4L)\lambda^2[F(\tilde{x}_s) - F(x_\star)] \\
= & \|x_t - x_\star\|_2^2 - 2\lambda(1 - \frac{4}{\mu}C_{\mathcal{D}}\lambda)[F(x_t) - F(x_\star)] + (\frac{8}{\mu}C_{\mathcal{D}} + 4L)\lambda^2[F(\tilde{x}_s) - F(x_\star)].
\end{aligned} \tag{5.40}$$

where the third line follows from the unbiasedness of the simulated gradient and the fourth line follows from the convexity of $F(\cdot)$. Since $\tilde{x}_{s+1}$ is selected uniformly after all M updates are

completed and $x_0 = \tilde{x}_s$, summing over the previous inequality over $t = 0, ..., M-1$ and taking expectation and using option II at stage s, we obtain

$$\begin{aligned}
&\mathbb{E}[\|x_M - x_\star\|_2^2] + 2\lambda(1 - \frac{4}{\mu}C_{\mathcal{D}}\lambda)M\mathbb{E}[F(\tilde{x}_{s+1}) - F(x_\star)] \\
\leq &\mathbb{E}[\|x_0 - x_\star\|_2^2] + (\frac{8}{\mu}C_{\mathcal{D}} + 4L)\lambda^2 M\mathbb{E}[F(\tilde{x}_s) - F(x_\star)] \\
= &\mathbb{E}[\|\tilde{x}_s - x_\star\|_2^2] + (\frac{8}{\mu}C_{\mathcal{D}} + 4L)\lambda^2 M\mathbb{E}[F(\tilde{x}_s) - F(x_\star)] \\
\leq &\frac{2}{\mu}\mathbb{E}[F(\tilde{x}_s) - F(x_\star)] + (\frac{8}{\mu}C_{\mathcal{D}} + 4L)\lambda^2 M\mathbb{E}[F(\tilde{x}_s) - F(x_\star)] \\
= &(\frac{2}{\mu} + (\frac{8}{\mu}C_{\mathcal{D}} + 4L)\lambda^2 M)\mathbb{E}[F(\tilde{x}) - F(x_\star)].
\end{aligned} \tag{5.41}$$

Thus we obtain

$$\mathbb{E}[F(\tilde{x}_{s+1}) - F(x_\star)] \leq \Big[\frac{1}{\mu(1 - \frac{4}{\mu}C_{\mathcal{D}}\lambda)\lambda M} + \frac{(\frac{4}{\mu}C_{\mathcal{D}} + 2L)\lambda}{1 - \frac{4}{\mu}C_{\mathcal{D}}\lambda}\Big]\mathbb{E}[F(\tilde{x}_s) - F(x_\star)]. \tag{5.42}$$

This implies that $\mathbb{E}[F(\tilde{x}_s) - F(x_\star)] \leq \alpha^s \mathbb{E}[F(\tilde{x}_0) - F(x_\star)]$. The conclusion follows. □

As we mentioned, the sample complexity becomes difficult to analyze in the presence of batch size randomization. However, the corollary below provides an estimate of the total number of samples that are needed to achieve and ϵ-accurate solution for the finite sample SCO problems using Algorithm 11.

Corollary 5.4.17. *In Algorithm 11, let $T_\epsilon = \min\{n \geq 0 \mid F(\tilde{x}_k) - F(x_\star) \leq \epsilon\}$ and let $N_{k,t}$ be the geometric random number that is generated when calling SimulatedGradient procedure at t-th epoch and k-th iteration. Then we have*

$$\mathbb{E}\{\sum_{k=1}^{T_\epsilon}\sum_{t=1}^{M}(2^{n_0+N_{k,t}+1} + 1)\} = O(\log(1/\epsilon)).$$

Proof. Since T_ϵ is a stopping time, by Wald's identity and Proposition 5.4.7, we have

$$\begin{aligned}\mathbb{E}\{\sum_{k=1}^{T_\epsilon}\sum_{t=1}^{M} 2^{N_{k,t}+1}\} &= M\ \mathbb{E}T_\epsilon\ \mathbb{E}(2^{n_0+N_{k,t}+1}+1)\\ &= M\{1+2^{n_0+1}(1-0.5^\gamma)(1-2^{1-\gamma})^{-1}\}\mathbb{E}T_\epsilon.\end{aligned}$$

Next, we analyze $\mathbb{E}T_\epsilon$. Since T_ϵ is non- negative, we have

$$\begin{aligned}\exp(\mathbb{E}T_\epsilon) \le \mathbb{E}\exp(T_\epsilon) &= \int_0^\infty \mathbb{P}\{\exp(T_\epsilon)\ge x\}dx = 1+\int_1^\infty \mathbb{P}\{T_\epsilon \ge \log(x)\}dx\\ &\le 1+\int_1^\infty \mathbb{P}\{T_\epsilon \ge \lfloor\log(x)\rfloor\}dx \le 3+\int_3^\infty \mathbb{P}\{T_\epsilon \ge \lfloor\log(x)\rfloor\}dx.\end{aligned}$$

By the definition of T_ϵ,Markov's inequality and Theorem 5.4.16 , we have

$$\mathbb{P}(T_\epsilon \ge k) \le \mathbb{P}\{F(\tilde{x}_k)-F(x_\star)\ge\epsilon\} \le \frac{1}{\epsilon}\mathbb{E}\{F(\tilde{x}_k)-F(x_\star)\} \le \frac{1}{\epsilon}\alpha^k\{F(\tilde{x}_0)-F(x_\star)\}.$$

Therefore,

$$\exp\{\mathbb{E}T_\epsilon\} \le 3+\frac{1}{\epsilon}\int_3^\infty \{F(\tilde{x}_0)-F(x_\star)\}\alpha^{\lfloor\log(x)\rfloor}dx \le 3+\frac{F(\tilde{x}_0)-F(x_\star)}{\alpha\epsilon}\int_3^\infty x^{\log(\alpha)}dx.$$

If we choose M and λ in Algorithm 11 such that $\log\alpha < -1$, we have

$$\exp\{\mathbb{E}T_\epsilon\} \le 3+\frac{F(\tilde{x}_0)-F(x_\star)}{\alpha\epsilon(-\log\alpha-1)}3^{\log\alpha+1}.$$

Therefore $\mathbb{E}T_\epsilon = O(\log(1/\epsilon))$. Consequently, $\mathbb{E}\{\sum_{k=1}^{T_\epsilon}\sum_{t=1}^{M}(2^{n_0+N_{k,t}+1}+1)\} = O(1/\epsilon)$. □

Corollary 5.4.18. *Let $\{\tilde{x}_s\}_{s\ge0}$ be the sequence of outputs from each epoch of the SimVRG algorithm. Then, with probability 1, $\tilde{x}_s$ converges exponentially fast to $x_\star$.*

Proof. It follows from Theorem 5.4.16 that we can find $0<\alpha<1$ such that $\mathbb{E}[F(\tilde{x}_s)] \le F(\tilde{x}_\star) + \alpha^s[F(\tilde{x}_0)-F(\tilde{x}_\star)]$. Pick any $\alpha<\rho<1$. Define the set $\mathcal{A}_s = \{F(\tilde{x}_s)-F(x_\star) > \rho^s\}$. We have $\mathbb{P}(\mathcal{A}_s) \le (\frac{\alpha}{\rho})^s\mathbb{E}[F(\tilde{x}_0)-F(x_\star)]$, which implies that $\sum_{s\ge0}\mathbb{P}(\mathcal{A}_s)<\infty$. It then follows from the

Borel-Cantelli lemma that

$$\mathbb{P}(\mathcal{A}_s \text{ occurs infinitely often}) = \mathbb{P}\Big(\limsup_{s\to\infty} \mathcal{A}_s\Big) = \mathbb{P}(\bigcap_{t=0}^{\infty}\bigcup_{s=t}^{\infty} \mathcal{A}_s) = \inf_{t\geq 0} \mathbb{P}(\bigcup_{s=t}^{\infty} \mathcal{A}_s)$$
$$\leq \inf_{t\geq 0} \sum_{s\geq t} \mathbb{P}(\mathcal{A}_s) = 0. \tag{5.43}$$

Thus with probability 1, $F(\tilde{x}_s) - F(x_\star) < \rho^s$ for s large enough (depending on each the probability path), which implies $\|\tilde{x}_s - x_\star\|_2^2 \leq \frac{2}{\mu}\rho^s$ in the presence of μ-strong convexity. □

5.4.6 Convergence of the Stochastically Controlled Simulated Gradient Algorithm

In this section we prove the convergence of Algorithm 12.

Lemma 5.4.19. *Fix $x \in \mathcal{D}$ and $K, B \geq 1$, and sample a batch $\mathcal{I} \subset \Omega_v$ with $|\mathcal{I}| = B$ following the distribution of v and independently generate*

$$h_k(x) = \frac{1}{B}\sum_{v_i\in\mathcal{I}} UnibasedGradient(x, v_i, n_0, \gamma)$$

for $1 \leq k \leq K$. Let $C'_{\mathcal{D}}$ be the constant in the proof of Proposition 5.4.6, where $\mathbb{E}\|W(x,v)\|_2^2 \leq C'_{\mathcal{D}}$ for arbitary $v \in \Omega_v$. Defining $\tilde{h}(x) = \frac{1}{K}\sum_{i=1}^{K} h_i(x)$, we have

$$\mathbb{E}[\tilde{h}(x)] = \nabla F(x) \qquad and \qquad Var[\tilde{h}(x)] \leq \frac{C'_{\mathcal{D}}}{KB} + 4pd^2 l_{\mathcal{D}}^4(\frac{1}{K} + \frac{1}{B}), \tag{5.44}$$

so $Var[\tilde{h}(x)]$ can be made arbitrarily small for any $x \in \mathcal{D}$ by making K and B sufficiently large.

Proof. First we have

$$\mathbb{E}[\tilde{h}(x)] = \mathbb{E}[h_1(x)] = \mathbb{E}[\mathbb{E}[h_1(x)|\mathcal{I}]] = \frac{1}{B}\mathbb{E}[\mathbb{E}[\sum_{v_i\in\mathcal{I}} \text{UnibasedGradient}(x, v_i)|\mathcal{I}]]$$
$$= \frac{1}{B}\mathbb{E}[\sum_{v_i\in\mathcal{I}} \nabla(f_{v_i}(\mathbb{E}_w g_w(x)))] = \nabla F(x).$$

Second, for any $v \in \Omega_v$, denote $W_i = \text{UnbiasedGradient}(x, v_i)$, $h_v = \nabla(f_v(\mathbb{E}_w g_w(x)))$ and $h(\mathcal{I}) =$

$\mathbb{E}[h_1(x)|\mathcal{I}] = \frac{1}{B}\sum_{v_i\in\mathcal{I}} h_{v_i}$. We have

$$\begin{aligned}
Var[\tilde{h}(x)] =& \mathbb{E}[Var[\tilde{h}(x)|\mathcal{I}]] + Var[\mathbb{E}[\tilde{h}(x)|\mathcal{I}]] = \frac{1}{K}\mathbb{E}[Var[h_1(x)|\mathcal{I}]] + Var_{\mathcal{I}}[h(\mathcal{I})] \\
=& \frac{1}{K}\mathbb{E}\big[\mathbb{E}[(h_1(x) - h(\mathcal{I}))^\intercal (h_1(x) - h(\mathcal{I}))|\mathcal{I}]\big] + \frac{1}{B}Var_v[h_v] \\
=& \frac{1}{KB^2}\mathbb{E}\big[\mathbb{E}[(\sum_{i=1}^{B} W_i - h_{v_i} + h_{v_i} - h(\mathcal{I}))^\intercal (\sum_{i=1}^{B} W_i - h_{v_i} + h_{v_i} - h(\mathcal{I}))|\mathcal{I}]\big] + \frac{1}{B}Var_v[h_v] \\
=& \frac{1}{KB^2}\mathbb{E}\big[\mathbb{E}[\sum_{i=1}^{B} \|W_i - h_{v_i}\|_2^2 + \sum_{i=1}^{B}\sum_{j=1}^{B}(h_{v_i} - h(\mathcal{I}))^\intercal (h_{v_j} - h(\mathcal{I}))|\mathcal{I}]\big] + \frac{1}{B}Var_v[h_v] \\
\leq& \frac{C'_{\mathcal{D}}}{KB} + 4pd^2 l_{f,1}^2 l_{g,1}^2 (\frac{1}{K} + \frac{1}{B}),
\end{aligned}$$

where the last inequality follows from the definition of $C'_{\mathcal{D}}$ and the fact that each component of h_v is bounded by $dl_{f,1}l_{g,1}$ for any $v \in \Omega_v$, according to the definition of $l_{\mathcal{D}}$ and h_v. The equality above it follows from the independence between the W_i's given $\mathcal{I}$. □

Theorem 5.4.20. *Consider the Simulated SCSG Algorithm 12 with option II. Fix $\epsilon > 0$ as the level of accuracy. Let λ be sufficiently small and M be sufficiently large so that*

$$\alpha = \frac{2}{\mu(1 - \frac{8}{\mu}C_{\mathcal{D}}\lambda)\lambda M} + \frac{(\frac{8}{\mu}C_{\mathcal{D}} + 8L)\lambda}{1 - \frac{8}{\mu}C_{\mathcal{D}}\lambda} < 1, \tag{5.45}$$

while making either K or B large enough so that

$$\frac{4(\lambda + \frac{1}{2\mu})}{1 - \frac{8}{\mu}C_{\mathcal{D}}\lambda} Var[\tilde{h}(\tilde{x}_s)] < \epsilon. \tag{5.46}$$

Then

$$\mathbb{E}[F(\tilde{x}_s) - F(x_\star)] \leq \alpha^s \mathbb{E}[F(\tilde{x}_0) - F(x_\star)] + \frac{1}{1-\alpha}\epsilon. \tag{5.47}$$

Proof. Conditioning on x_t, we can take expectation with respect to $v_t \in \Omega_v$ to obtain

$$
\begin{aligned}
&\mathbb{E}[\|\rho_t\|_2^2 \mid x_t] \\
\leq& 2\mathbb{E}[\|W(x_t, v_t) - W(\tilde{x}_s, v_t)\|_2^2 \mid x_t] + 4\|\nabla F(\tilde{x}_s)\|_2^2 + 4\|\tilde{h}(\tilde{x}_s) - \nabla F(\tilde{x}_s)\|_2^2 \\
\leq& 2C_{\mathcal{D}}\|x_t - \tilde{x}_s\|_2^2 + 8L[F(\tilde{x}_s) - F(x_\star)] + 4\|\tilde{h}(\tilde{x}_s) - \nabla F(\tilde{x}_s)\|_2^2 \\
\leq& 4C_{\mathcal{D}}(\|x_t - x_\star\|_2^2 + \|\tilde{x}_s - x_\star\|_2^2) + 8L[F(\tilde{x}_s) - F(x_\star)] + 4\|\tilde{h}(\tilde{x}_s) - \nabla F(\tilde{x}_s)\|_2^2 \\
\leq& \frac{8}{\mu}C_{\mathcal{D}}[F(x_t) - F(x_\star)] + (\frac{8}{\mu}C_{\mathcal{D}} + 8L)[F(\tilde{x}_s) - F(x_\star)] + 4\|\tilde{h}(\tilde{x}_s) - \nabla F(\tilde{x}_s)\|_2^2. \qquad (5.48)
\end{aligned}
$$

where the second inequality follows from Lemma 5.4.14 and equation (5.38). The last inequality follows from the strong convexity of $F(\cdot)$. Now following (5.48), using the distance contraction property of projection operator $\Pi_{\mathcal{D}}(\cdot)$ we can write

$$
\begin{aligned}
&\mathbb{E}[\|x_{t+1} - x_\star\|_2^2 \mid x_t] \\
\leq& \|x_t - x_\star\|_2^2 - 2\lambda(x_t - x_\star)^\intercal \mathbb{E}[\rho_t \mid x_t] + \lambda^2 \mathbb{E}[\|\rho_t\|_2^2 \mid x_t] \\
\leq& \|x_t - x_\star\|_2^2 - 2\lambda(x_t - x_\star)^\intercal(\nabla F(x_t) - \nabla F(\tilde{x}_s) + \tilde{h}(\tilde{x}_s)) + \frac{8}{\mu}C_{\mathcal{D}}\lambda^2[F(x_t) - F(x_\star)] \\
&+ (\frac{8}{\mu}C_{\mathcal{D}} + 8L)\lambda^2[F(\tilde{x}_s) - F(x_\star)] + 4\lambda^2\|\tilde{h}(\tilde{x}_s) - \nabla F(\tilde{x}_s)\|_2^2 \\
\leq& \|x_t - x_\star\|_2^2 - 2\lambda[F(x_t) - F(x_\star)] + 2\lambda(x_t - x_\star)^\intercal(\tilde{h}(\tilde{x}_s) - \nabla F(\tilde{x}_s)) \\
&+ \frac{8}{\mu}C_{\mathcal{D}}\lambda^2[F(x_t) - F(x_\star)] + (\frac{8}{\mu}C_{\mathcal{D}} + 8L)\lambda^2[F(\tilde{x}_s) - F(x_\star)] + 4\lambda^2\|\tilde{h}(\tilde{x}_s) - \nabla F(\tilde{x}_s)\|_2^2 \\
=& \|x_t - x_\star\|_2^2 - 2\lambda(1 - \frac{4}{\mu}C_{\mathcal{D}}\lambda)[F(x_t) - F(x_\star)] + (\frac{8}{\mu}C_{\mathcal{D}} + 8L)\lambda^2[F(\tilde{x}_s) - F(x_\star)] \\
&+ 4\lambda^2\|\tilde{h}(\tilde{x}_s) - \nabla F(\tilde{x}_s)\|_2^2 + 2\lambda(x_t - x_\star)^\intercal(\tilde{h}(\tilde{x}_s) - \nabla F(\tilde{x}_s)), \qquad (5.49)
\end{aligned}
$$

where the third line follows from the convexity of $F(\cdot)$. Now we consider a fixed stage s, so that $x_0 = \tilde{x}_s$ and $\tilde{x}_{s+1}$ is selected uniformly after all M updates are completed. Summing the previous

inequality over $t = 1, ..., M$, taking expectation and using option II at stage s, we obtain

$$\begin{aligned}
&\mathbb{E}[\|x_M - x_\star\|_2^2] + 2\lambda(1 - \frac{4}{\mu}C_{\mathcal{D}}\lambda)M\mathbb{E}[F(\tilde{x}_{s+1}) - F(x_\star)] \\
\leq&\mathbb{E}[\|x_0 - x_\star\|_2^2] + (\frac{8}{\mu}C_{\mathcal{D}} + 8L)\lambda^2 M\mathbb{E}[F(\tilde{x}_s) - F(x_\star)] \\
&+ 4\lambda^2 M\|\tilde{h}(\tilde{x}_s) - \nabla F(\tilde{x}_s)\|_2^2 + 2\lambda M\mathbb{E}[(\tilde{x}_{s+1} - x_\star)^\intercal(\tilde{h}(\tilde{x}_s) - \nabla F(\tilde{x}_s))] \\
\leq&\mathbb{E}[\|\tilde{x}_s - x_\star\|_2^2] + (\frac{8}{\mu}C_{\mathcal{D}} + 8L)\lambda^2 M\mathbb{E}[F(\tilde{x}_s) - F(x_\star)] \\
&+ 4\lambda M(\lambda + \frac{1}{2\mu})\|\tilde{h}(\tilde{x}_s) - \nabla F(\tilde{x}_s)\|_2^2 + \frac{\mu}{2}\lambda M\mathbb{E}[\|\tilde{x}_{s+1} - x_\star\|_2^2] \\
\leq&\mathbb{E}[\|\tilde{x}_s - x_\star\|_2^2] + (\frac{8}{\mu}C_{\mathcal{D}} + 8L)\lambda^2 M\mathbb{E}[F(\tilde{x}_s) - F(x_\star)] \\
&+ 4\lambda M(\lambda + \frac{1}{2\mu})\|\tilde{h}(\tilde{x}_s) - \nabla F(\tilde{x}_s)\|_2^2 + \lambda M\mathbb{E}[F(\tilde{x}_{s+1}) - F(x_\star)],
\end{aligned} \tag{5.50}$$

where the second inequality follows from $2a^\intercal b \leq \beta\|a\|_2^2 + \frac{1}{\beta}\|b\|_2^2$, while $\beta = \frac{\mu}{2}$. The last inequality follows from the strong convexity of $F(\cdot)$. Finally, taking expectation over the randomness of $\tilde{h}(\tilde{x}_s)$, we have

$$\begin{aligned}
&\lambda(1 - \frac{8}{\mu}C_{\mathcal{D}}\lambda)M\mathbb{E}[F(\tilde{x}_{s+1}) - F(x_\star)] \\
\leq&\mathbb{E}[\|\tilde{x}_s - x_\star\|_2^2] + (\frac{8}{\mu}C_{\mathcal{D}} + 8L)\lambda^2 M\mathbb{E}[F(\tilde{x}_s) - F(x_\star)] + 4\lambda M(\lambda + \frac{1}{2\mu})Var[\tilde{h}(\tilde{x_s})] \\
\leq&\frac{2}{\mu}\mathbb{E}[F(\tilde{x}_s) - F(x_\star)] + (\frac{8}{\mu}C_{\mathcal{D}} + 8L)\lambda^2 M\mathbb{E}[F(\tilde{x}_s) - F(x_\star)] + 4\lambda M(\lambda + \frac{1}{2\mu})Var[\tilde{h}(\tilde{x}_s)] \\
=&(\frac{2}{\mu} + (\frac{8}{\mu}C_{\mathcal{D}} + 8L)\lambda^2 M)\mathbb{E}[F(\tilde{x}_s) - F(x_\star)] + 4\lambda M(\lambda + \frac{1}{2\mu})Var[\tilde{h}(\tilde{x}_s)].
\end{aligned} \tag{5.51}$$

Thus we obtain

$$\begin{aligned}
&\mathbb{E}[F(\tilde{x}_{s+1}) - F(x_\star)] \\
\leq&\Big[\frac{2}{\mu(1 - \frac{8}{\mu}C_{\mathcal{D}}\lambda)\lambda M} + \frac{(\frac{8}{\mu}C_{\mathcal{D}} + 8L)\lambda}{1 - \frac{8}{\mu}C_{\mathcal{D}}\lambda}\Big]\mathbb{E}[F(\tilde{x}_s) - F(x_\star)] + \frac{4(\lambda + \frac{1}{2\mu})}{1 - \frac{8}{\mu}C_{\mathcal{D}}\lambda}Var[\tilde{h}(\tilde{x}_s)] \\
\leq&\alpha\mathbb{E}[F(\tilde{x}_s) - F(x_\star)] + \epsilon.
\end{aligned} \tag{5.52}$$

This implies that $\mathbb{E}[F(\tilde{x}_s) - F(x_\star)] \leq \alpha^s \mathbb{E}[F(\tilde{x}_0) - F(x_\star)] + \frac{\epsilon}{1-\alpha}$. The conclusion follows. □

Corollary 5.4.21. *Let $\{\tilde{x}_s\}_{s\geq 0}$ be the sequence of outputs from each epoch of the SCSimG algorithm (Algorithm 12) and define $\tilde{y}_s = \min_{t\leq s}\{F(\tilde{x}_t) - F(x_\star)\}$ for $s \geq 0$ to be the lowest objective value after epoch s. Then, with probability 1, we have $\inf_{s\geq 0} \tilde{y}_s \leq \frac{\epsilon}{1-\alpha}$.*

Proof. It follows from Theorem 5.4.20 that we can find $0 < \alpha < 1$ where $\mathbb{E}[F(\tilde{x}_s) - F(x_\star)] \leq \alpha^s \mathbb{E}[F(\tilde{x}_0) - F(x_\star)] + \frac{\epsilon}{1-\alpha}$. We also have $\sup_{x\in\mathcal{D}}\{F(x) - F(x_\star)\} \leq 2l_{\mathcal{D}}$ from the definition of $l_{\mathcal{D}}$. It follows that for any $\tilde{x}_0 \in \mathcal{D}$, we have that $\mathbb{E}[F(\tilde{x}_s) - F(x_\star)|\tilde{x}_0] \leq \alpha^s \cdot 2l_{\mathcal{D}} + \frac{\epsilon}{1-\alpha}$. For any $\rho > 0$, picking N large enough so that $\delta = (\alpha^N \cdot 2l_{\mathcal{D}} + \frac{\epsilon}{1-\alpha})(\frac{\epsilon}{1-\alpha} + \rho)^{-1} < 1$, we have

$$\mathbb{P}(\tilde{y}_N \geq \frac{\epsilon}{1-\alpha} + \rho) \leq \mathbb{P}(F(\tilde{x}_N) - F(x) \geq \frac{\epsilon}{1-\alpha} + \rho) \leq \mathbb{E}[F(\tilde{x}_0) - F(x)](\frac{\epsilon}{1-\alpha} + \rho)^{-1} \leq \delta.$$

However, if we denote $\mathcal{X}_N$ to be the distribution of $\tilde{x}_N$ conditioned on $\tilde{y}_N \geq \frac{\epsilon}{1-\alpha} + \rho$, then it follows from the Markov Property that

$$\begin{aligned}
&\mathbb{P}(\tilde{y}_{2N} \geq \frac{\epsilon}{1-\alpha} + \rho) \\
=&\mathbb{P}(\tilde{y}_{2N} \geq \frac{\epsilon}{1-\alpha} + \rho | \tilde{y}_N \geq \frac{\epsilon}{1-\alpha} + \rho)\mathbb{P}(\tilde{y}_N \geq \frac{\epsilon}{1-\alpha} + \rho) \\
=&\mathbb{P}(\min_{N+1\leq s\leq 2N}\{F(\tilde{x}_s) - F(x_\star)\} \geq \frac{\epsilon}{1-\alpha} + \rho | \tilde{y}_N \geq \frac{\epsilon}{1-\alpha} + \rho)\mathbb{P}(\tilde{y}_N \geq \frac{\epsilon}{1-\alpha} + \rho) \\
=&(\mathbb{P}_{\tilde{x}_N\sim\mathcal{X}_N}\mathbb{P}(\min_{N+1\leq s\leq 2N}\{F(\tilde{x}_s) - F(x_\star)\} \geq \frac{\epsilon}{1-\alpha} + \rho|\tilde{x}_N)) \cdot \mathbb{P}(\tilde{y}_N \geq \frac{\epsilon}{1-\alpha} + \rho) \\
\leq&(\mathbb{P}_{\tilde{x}_N\sim\mathcal{X}_N}\mathbb{P}(F(\tilde{x}_{2N}) - F(x_\star) \geq \frac{\epsilon}{1-\alpha} + \rho|\tilde{x}_N)) \cdot \delta \\
\leq&(\mathbb{P}_{\tilde{x}_N\sim\mathcal{X}_N}\mathbb{E}[F(\tilde{x}_{2N}) - F(x_\star)|\tilde{x}_N]) \cdot (\frac{\epsilon}{1-\alpha} + \rho)^{-1} \cdot \delta \\
=&(\mathbb{P}_{\tilde{x}_0\sim\mathcal{X}_N}\mathbb{E}[F(\tilde{x}_N) - F(x_\star)|\tilde{x}_0]) \cdot (\frac{\epsilon}{1-\alpha} + \rho)^{-1} \cdot \delta \\
\leq&\mathbb{P}_{\tilde{x}_0\sim\mathcal{X}_N}(\alpha^N \cdot 2l_{\mathcal{D}} + \frac{\epsilon}{1-\alpha}) \cdot (\frac{\epsilon}{1-\alpha} + \rho)^{-1} \cdot \delta \leq \delta^2.
\end{aligned}$$

We can prove that $\mathbb{P}(\tilde{y}_{kN} \geq \frac{\epsilon}{1-\alpha} + \rho) \leq \delta^k$. Thus if we define the set $\mathcal{A}_\rho = \{\inf_{s\geq 0} \tilde{y}_s \geq \frac{\epsilon}{1-\alpha} + \rho\}$ and

$\mathcal{A} = \{\inf_{s\geq 0} \tilde{y}_s > \frac{\epsilon}{1-\alpha}\}$ in probability space, we have

$$\mathbb{P}(\mathcal{A}_\rho) = \mathbb{P}(\inf_{s\geq 0} \tilde{y}_s \geq \frac{\epsilon}{1-\alpha} + \rho) \leq \mathbb{P}(\tilde{y}_{kN} \geq \frac{\epsilon}{1-\alpha} + \rho) \leq \delta^k, \tag{5.53}$$

for any $k \geq 1$. Since $\delta < 1$, we have $\mathbb{P}(\mathcal{A}_\rho) = 0$ for any $\rho > 0$, which implies $\mathbb{P}(\mathcal{A}) = \mathbb{P}(\bigcup_{n\geq 1} \mathcal{A}_{\frac{1}{n}}) \leq \sum_{n=1}^{\infty} \mathbb{P}(\mathcal{A}_{\frac{1}{n}}) = 0$. So, with probability 1, $\inf_{s\geq 0} \tilde{y}_s \leq \frac{\epsilon}{1-\alpha}$. □

5.5 Numerical Experiments

In our numerical experiments, all algorithms were implemented in C++, and were performed on an Intel i5-5200U processor using Ubuntu 16.04.

5.5.1 Cox's Partial Likelihood

We implemented Algorithms 9(SGD),11(SimVRG) and 12(SCSimG) to minimize a regularized Cox's negative partial log- likelihood and compared their performance with the Compositional-SVRG-1 algorithm (Comp-SVRG-1) in [51], the Stochastic Compositional Gradient Descent algorithm (SCGD) in [17] and Gradient Descent(GD) algorithm. The optimization problem in Section 5.3 combined with L_2 regulation can be written as:

$$\min_{\beta\in\mathbb{R}^p} \quad \frac{1}{n}\sum_{i=1}^{n} \Delta_i[-X_i^\top \beta + \log\{\sum_{j=1}^{n} \mathbb{I}(Y_j \geq Y_i)\exp(X_j^\top \beta)\}] + \frac{1}{2}\|\beta\|_2^2,, \tag{5.54}$$

where (X_i, Y_i, Δ_i) and T_i, C_i for $i = 1, \ldots, n$ come from the Cox's model as in the setting of Section 5.3. Here, we generated our dataset by seting $n = 10^4$, $p = 10^3$ and letting X_i follow the i.i.d. standard normal distribution. Moreover, T_i was generated according to the standard exponential base line hazard function and C_i was generated independent of T_i with a 30% censoring rate. One can check that each component function is strongly convex with Lipschitz continuous gradients. The numerical results are presented below in Figure 5.1.

In Figure 5.1, the upper plot is the logarithm of the objective value minus the optimal value versus the number of iterations while the lower plot is the logarithm of the same difference versus the CPU running time. We compare both the running time and the iteration number to give a more comprehensive review of each algorithm since the iteration time for each algorithm could be drastically different due to different update rules. Moreover, the parameters in each algorithm were selected and tuned to achieve a relatively optimal performance without heavily increasing the computational cost. In Algorithm 11, we set $\lambda = 0.01$, $\gamma = 3/2$ $M = 100$ and $n_0 = 0$, in Algorithm 12, $\lambda = 0.0005$, $M = 100$, $B = 100$, $K = 50$ and $n_0 = 2$, in Compositional SVRG-1, we set $\lambda = 0.001$, $M = 100$ and $B = 500$, and in Gradient Descent we set $\lambda = 0.01$.

As we can see, in the upper plot, the SimVRG and Compositional-SVRG-1 algorithms performed best amongst all algorithms while SimVRG also had better performance in the lower plot. Algorithm 12, SCSimG was slightly less effective due to the lack of full gradient computation, but, as expected from the theorems in Section 5.4, Algorithm 11 also converged linearly to the optimal solution. Algorithm SimGD is ploted for every 50 iterations for the sake of fairness (to account for the inner loop in the other algorithms) and it also showed satisfactory performance without the presence of variance reduction techinques.

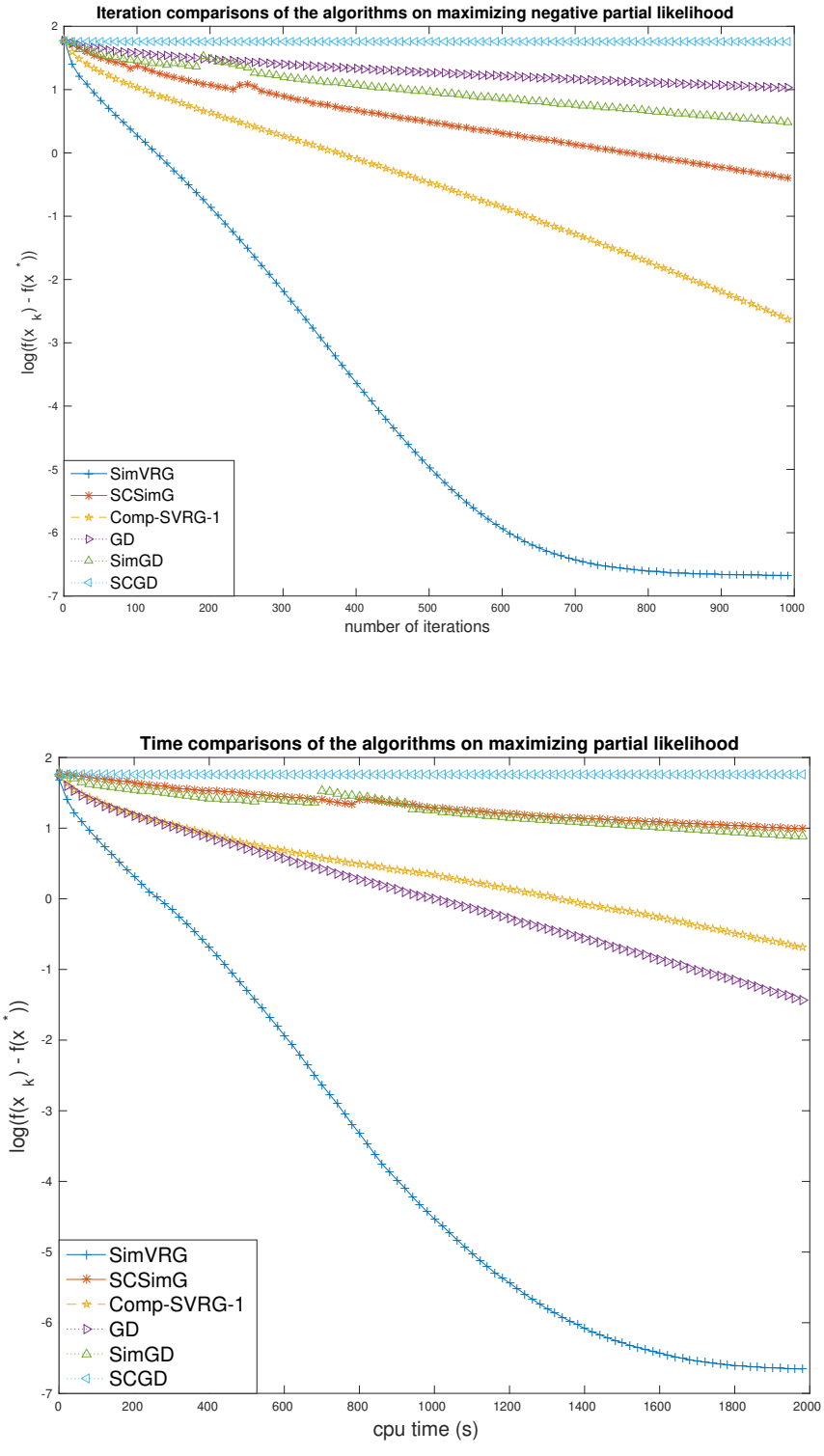


Figure 5.1: Performance plots for different algorithms on Cox's partial likelihood dataset.

5.5.2 Conditional Random Fields

We implemented Algorithms 9 (SimGD),11 (SimVRG) and 12 (SCSimG) to train conditional random field models and compared their performance with the Compositional-SVRG-1 algorithm (Comp-SVRG-1) in [51], the Stochastic Compositional Gradient Descent algorithm (SCGD) in [17] and Gradient Descent(GD) algorithm. In our tests, we used the optical character recognition (OCR) data in [89], which provides labelling for letters in a image composed of words. The numerical results are summarized in Figure 5.2.

Once again, to make comparisons fair, the performance of algorithms was measured both by the number of iterations and CPU time. For the parameters, in Algorithm 11, we set $\lambda = 0.001$,$\gamma = 3/2$ $M = 200$ and $n_0 = 0$, in Algorithm 12, $\lambda = 0.0001$, $M = 200$, $B = 100$, $K = 10$ and $n_0 = 2$, in Gradient Descent, $\lambda = 0.01$. In the other algorithms, the parameters were chosen according to their convergence theorem with scaling factor 0.5. For example, basic SCGD corresponds to Theorem 6 in [17].

As we can see from the figures, once again, the SimVRG (Algorithm 11) has the best performance amongst the group. However, in this example, the gradient descent algorithm actually outperforms Algorithm 12 in terms iteration complexity. This is possibly due to the lack of accurate gradient estimation in Algorithm 12. Specifically, as the dataset grows large, it becomes more costly to obtain accurate gradient estimate. On the other hand, the SimGD in Algorithm 9 outperforms SCGD in terms of iterations and CPU time for both datasets. We note that the occasional increase of function value in some executions of the SimGD algorithm is caused by the variance of our gradient simulation.

5.6 Conclusion and Future Work.

In this chapter, we introduced unbiased gradient simulation algorithms that are based on a multilevel Monte Carlo technique for solving stochastic compositional optimization (SCO) problems

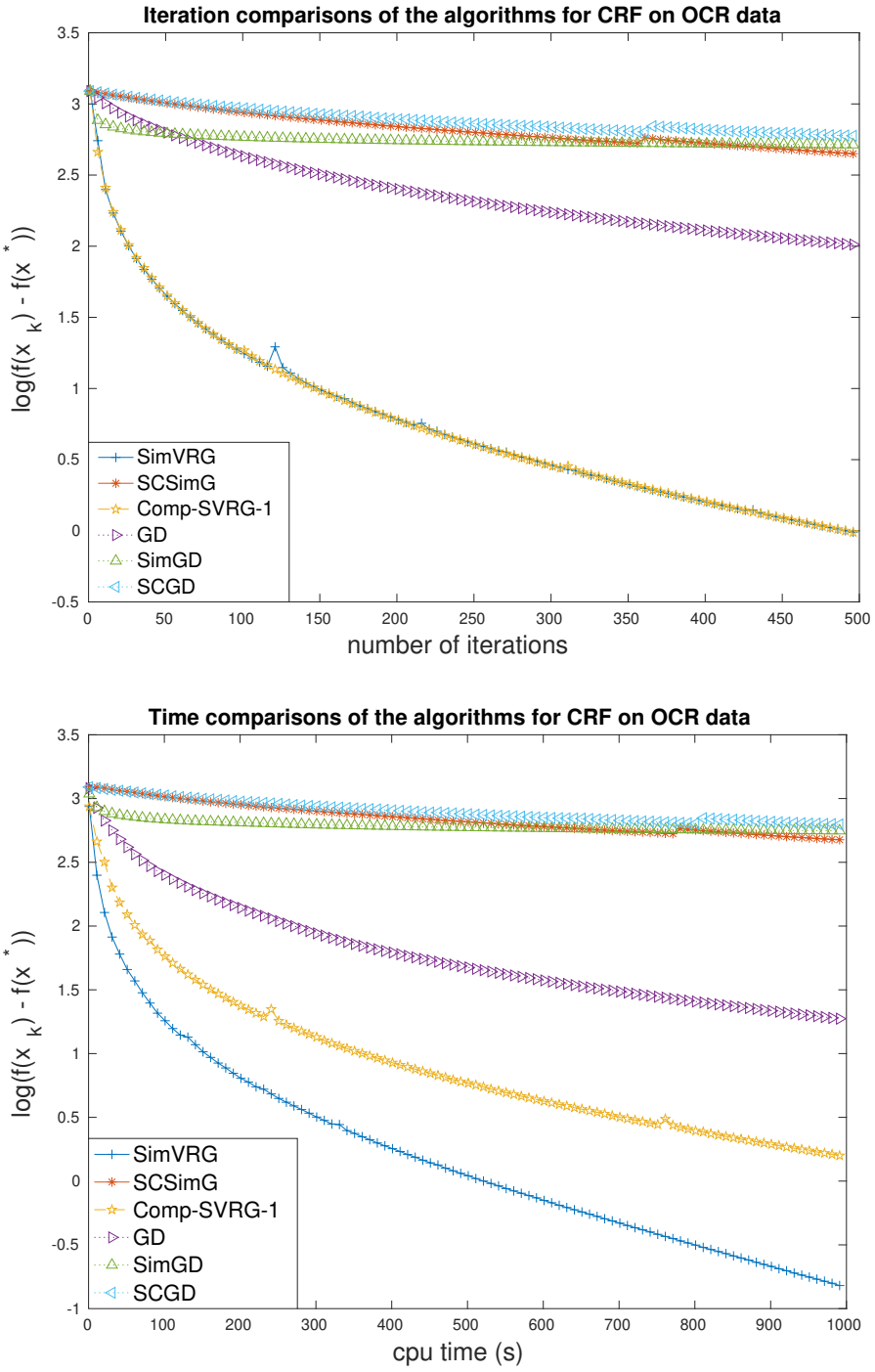


Figure 5.2: Performance plots for different algorithms on the OCR dataset.

and proved convergence of our algorithms and applied them to a number of different statistical and machine learning problems.

There are several directions where we can expand upon our work. For example, different accelerating schemes and second-order methods usually show fast convergence in practice, and can be extended using simulated gradients for SCO problems. Another direction is to extend our approach to adaptive step size schemes. A limitation of our unbiased gradient simulation algorithm is the requirement for smoothness of the objective function. Therefore, developing unbiased simulation of sub-gradient methods and utilizing them for optimizing non-smooth functions is also of great interest. Analyzing the sample complexity of our algorithms and the optimal choice of the parameters are also interesting problems for future work.

References

[1] D. Goldfarb, G. Iyengar, and C. Zhou, "Linear convergence of stochastic Frank-Wolfe variants," *ArXiv preprint arXiv:1703.07269*,

[2] C. Zhou, W. Gao, and D. Goldfarb, "Stochastic adaptive quasi-Newton methods for minimizing expected values," in *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, JMLR. org, 2017, pp. 4150–4159.

[3] J. Blanchet, D. Goldfarb, G. Iyengar, F. Li, and C. Zhou, "Unbiased simulation for optimizing stochastic function compositions," *ArXiv preprint arXiv:1711.07564*, 2017.

[4] M. Jaggi, "Revisiting Frank-Wolfe: Projection-free sparse convex optimization.," in *ICML (1)*, 2013, pp. 427–435.

[5] S. Lacoste-Julien and M. Jaggi, "On the global linear convergence of Frank-Wolfe optimization variants," in *Advances in Neural Information Processing Systems*, 2015, pp. 496–504.

[6] R. Johnson and T. Zhang, "Accelerating stochastic gradient descent using predictive variance reduction," in *Advances in Neural Information Processing Systems 26*, C. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K. Weinberger, Eds., 2013, pp. 315–323.

[7] M. Schmidt, N. Le Roux, and F. Bach, "Minimizing finite sums with the stochastic average gradient," *Mathematical Programming*, pp. 1–30, 2013.

[8] A. Defazio, F. Bach, and S. Lacoste-Julien, "SAGA: A fast incremental gradient method with support for non-strongly convex composite objectives," in *Advances in Neural Information Processing Systems*, 2014, pp. 1646–1654.

[9] A. Rodomanov and D. Kropotov, "A superlinearly-convergent proximal Newton-type method for the optimization of finite sums," in *International Conference on Machine Learning*, 2016, pp. 2597–2605.

[10] M. J. D. Powell, "Some global convergence properties of a variable metric algorithm for minimization without exact line searches," vol. IX, R. Cottle and C. Lemke, Eds., 1976.

[11] R. H. Byrd, J. Nocedal, and Y.-X. Yuan, "Global convergence of a class of quasi-Newton methods on convex problems," *SIAM. J. Numer. Anal.*, no. 5, pp. 1171–1190, 1987.

[12] W. Gao and D. Goldfarb, "Quasi-Newton methods: Superlinear convergence without line search for self-concordant functions," *In review. arXiv:1612.06965*, 2016.

[13] C. G. Broyden, "Quasi-Newton methods and their application to function minimisation," *Mathematics of Computation*, vol. 21, no. 99, pp. 368–381, 1967.

[14] R. Fletcher, "A new approach to variable metric algorithms," *The Computer Journal*, vol. 13, no. 3, pp. 317–322, 1970.

[15] D. Goldfarb, "A family of variable-metric methods derived by variational means," *Mathematics of computation*, vol. 24, no. 109, pp. 23–26, 1970.

[16] D. F. Shanno, "Conditioning of quasi-Newton methods for function minimization," *Mathematics of computation*, vol. 24, no. 111, pp. 647–656, 1970.

[17] M. Wang, E. X. Fang, and H. Liu, "Stochastic compositional gradient descent: Algorithms for minimizing compositions of expected-value functions," *Mathematical Programming*, vol. 161, no. 1-2, pp. 419–449, 2017.

[18] A. W. van der Vaart and J. A. Wellner, Eds., *Weak Convergence and Empirical Processes With Applications to Statistics*. Springer, 1996.

[19] M. Frank and P. Wolfe, "An algorithm for quadratic programming," *Naval Research Logistics Quarterly*, vol. 3, 95–110, 1956.

[20] Y. Nesterov, "Complexity bounds for primal-dual methods minimizing the model of objective function," Université catholique de Louvain, Center for Operations Research and Econometrics ´ (CORE), Tech. Rep., 2015.

[21] E. Levitin and B. T. Polyak, "Constrained minimization methods," *USSR Computational Mathematics and Mathematical Physics*, vol. 6, no. 5, 787–823, 1966.

[22] D. Garber and E. Hazan, "Faster rates for the Frank-Wolfe method over strongly-convex sets," in *Proceedings of the 32nd International Conference on Machine Learning (ICML 2015)*, 2015, 541–549.

[23] J. Abadie, Ed., *Integer and Nonlinear Programming*. Amsterdam: North-Holland, 1970.

[24] J. Guelat and P. Marcotte, "Some comments on Wolfe's 'away step'," *Mathematical Programming*, vol. 35, no. 1, 110–119, 1986.

[25] D. Garber and E. Hazan, "A linearly convergent conditional gradient algorithm with applications to online and stochastic optimization," *ArXiv:1301.4666*, 2013.

[26] S. Lacoste-Julien and M. Jaggi, "An affine invariant linear convergence analysis for Frank-Wolfe algorithms," *ArXiv:1312.7864v2*, 2014.

[27] A. Beck and S. Shtern, "Linearly convergent away-step conditional gradient for non-strongly convex functions," *ArXiv: 1504.05002*, 2015.

[28] G. Lan, "The complexity of large-scale convex programming under a linear optimization oracle," *ArXiv:1309.5550*, 2013.

[29] J. Lafond, H. Wai, and E. Moulines, "Convergence analysis of a stochastic projection-free algorithm," *ArXiv:1510.01171*, 2015.

[30] H. Luo and E. Hazan, "Variance-reduced and projection-free stochastic optimization," in *Proceedings of the 33rd International Conference on Machine Learning (ICML 2016)*, vol. 48, 2016.

[31] H. Ouyang and A. Gray, "Fast stochastic Frank-Wolfe algorithms for nonlinear SVMs," in *SDM*, 2010, 245–256.

[32] S. Lacoste-Julien, M. Jaggi, M. Schmidt, and P. Pletscher, "Block-coordinate Frank-Wolfe optimization for structural svms," *In Proceedings of the 30th International Conference on Machine Learning (ICML-13)*, vol. 28, 53–61, 2013.

[33] A. Osokin, J. B. Alayrac, I. Lukasewitz, P. K. Dokania, and S. Lacoste-Julien, "Minding the gaps for block Frank-Wolfe optimization of structured svms," *ArXiv:1605.09346*, 2016.

[34] C. Mu, Y. Zhang, J. Wright, and D. Goldfarb, "Scalable robust matrix recovery: Frank-Wolfe meets proximal methods," *ArXiv:1403.7588*, 2015.

[35] X. Lin and T. Zhang, "A proximal stochastic gradient method with progressive variance reduction," *SIAM Journal on Optimization*, vol. 24, no. 4, pp. 2057–2075, 2014.

[36] M. Lichman, *UCI machine learning repository*, 2013.

[37] T. Bertin-Mahieux, D. P. Ellis, B. Whitman, and P. Lamere, "The million song dataset," in *Proceedings of the 12th International Conference on Music Information Retrieval (ISMIR 2011)*, 2011.

[38] J. Duchi, S. Shalev-Shwartz, Y. Singer, and T. Chandra, "Efficient projections onto the l 1-ball for learning in high dimensions," in *Proceedings of the 25th international conference on Machine learning*, ACM, 2008, pp. 272–279.

[39] M. Philipp, N. Robert, and I. J. Michael, "A linearly-convergent stochastic L-BFGS algorithm," in *Proceedings of the Nineteenth International Conference on Artificial Intelligence and Statistics*, 2016, pp. 249–258.

[40] R. Gower, D. Goldfarb, and P. Richtárik, "Stochastic block BFGS: Squeezing more curvature out of data," in *International Conference on Machine Learning*, 2016, pp. 1869–1878.

[41] R. H. Byrd, G. M. Chin, J. Nocedal, and Y. Wu, "Sample size selection in optimization methods for machine learning," *Mathematical Programming*, vol. 134, no. 1, pp. 127–155, 2012.

[42] M. P. Friedlander and G. Goh, "Tail bounds for stochastic approximation," *ArXiv preprint arXiv:1304.5586*, 2013.

[43] Y. Nesterov and A. Nemirovski, "Interior-point polynomial algorithms in convex programming," 1994.

[44] Y. Zhang and L. Xiao, "DiSCO: Distributed optimization for self-concordant empirical loss," vol. 32, pp. 362–370, 2015.

[45] Q. Tran-Dinh, A. Kyrillidis, and V. Cevher, "Composite self-concordant minimization," *Journal of Machine Learning Research*, vol. 16, no. Mar, pp. 371–416, 2015.

[46] J. Nocedal, "Updating quasi-Newton matrices with limited storage," *Math. Comp.*, vol. 35, no. 151, pp. 773–782, 1980.

[47] A. Griewank and P. L. Toint, "Local convergence analysis for partitioned quasi-Newton updates," *Numer. Math.*, vol. 39, pp. 429–448, 1982.

[48] J. E. Dennis Jr. and J. J. Moré, "Characterization of superlinear convergence and its application to quasi-Newton methods," *Math. Comp.*, vol. 28, no. 106, pp. 549–560, 1974.

[49] A. Nemirovski, A. Juditsky, G. Lan, and A. Shapiro, "Robust stochastic approximation approach to stochastic programming," *SIAM Journal on optimization*, vol. 19, no. 4, pp. 1574–1609, 2009.

[50] R. Frostig, R. Ge, S. M. Kakade, and A. Sidford, "Competing with the empirical risk minimizer in a single pass," in *Conference on learning theory*, 2015, pp. 728–763.

[51] X. Lian, M. Wang, and J. Liu, "Finite-sum composition optimization via variance reduced gradient descent," in *Proceedings of the 20th International Conference on Artificial Intelligence and Statistics*, A. Singh and J. Zhu, Eds., ser. Proceedings of Machine Learning Research, vol. 54, Fort Lauderdale, FL, USA: PMLR, 2017, pp. 1159–1167.

[52] L. Lei and M. I. Jordan, "Less than a single pass: Stochastically controlled stochastic gradient method," *ArXiv preprint arXiv:1609.03261*, 2016.

[53] M. Wang and J. Liu, "Accelerating stochastic composition optimization," in *Advances In Neural Information Processing Systems*, 2016, pp. 1714–1722.

[54] S. Ghadimi, A. Ruszczyński, and M. Wang, "A single time-scale stochastic approximation method for nested stochastic optimization," *ArXiv preprint arXiv:1812.01094*, 2018.

[55] C.-H. Rhee and P. W. Glynn, "Unbiased estimation with square root convergence for sde models," *Operations Research*, vol. 63, no. 5, pp. 1026–1043, 2015. eprint: http://dx.doi.org/10.1287/opre.2015.1404.

[56] J. H. Blanchet and P. W. Glynn, "Unbiased monte carlo for optimization and functions of expectations via multi-level randomization," in *Proceedings of the 2015 Winter Simulation Conference*, IEEE Press, 2015, pp. 3656–3667.

[57] M. B. Giles, L. Szpruch, *et al.*, "Antithetic multilevel monte carlo estimation for multi-dimensional sdes without lévy area simulation," *The Annals of Applied Probability*, vol. 24, no. 4, pp. 1585–1620, 2014.

[58] X. Li and J. Liu, "A multilevel approach towards unbiased sampling of random elliptic partial differential equations," *ArXiv preprint arXiv:1605.06349*, 2016.

[59] S. Dereich and F. Heidenreich, "A multilevel monte carlo algorithm for lévy-driven stochastic differential equations," *Stochastic Processes and their Applications*, vol. 121, no. 7, pp. 1565–1587, 2011.

[60] M. B. Giles and C. Reisinger, "Stochastic finite differences and multilevel monte carlo for a class of spdes in finance," *SIAM Journal on Financial Mathematics*, vol. 3, no. 1, pp. 572–592, 2012.

[61] D. F. Anderson and D. J. Higham, "Multilevel monte carlo for continuous time markov chains, with applications in biochemical kinetics," *Multiscale Modeling & Simulation*, vol. 10, no. 1, pp. 146–179, 2012.

[62] M. B. Giles, "Multilevel monte carlo path simulation," *Operations Research*, vol. 56, no. 3, pp. 607–617, 2008.

[63] ——, "Multilevel monte carlo methods," in *Monte Carlo and Quasi-Monte Carlo Methods 2012*, Springer, 2013, pp. 83–103.

[64] K. A. Cliffe, M. B. Giles, R. Scheichl, and A. L. Teckentrup, "Multilevel monte carlo methods and applications to elliptic pdes with random coefficients," *Computing and Visualization in Science*, vol. 14, no. 1, p. 3, 2011.

[65] J. Charrier, R. Scheichl, and A. L. Teckentrup, "Finite element error analysis of elliptic pdes with random coefficients and its application to multilevel monte carlo methods," *SIAM Journal on Numerical Analysis*, vol. 51, no. 1, pp. 322–352, 2013.

[66] S. Shalev-Shwartz and T. Zhang, "Stochastic dual coordinate ascent methods for regularized loss minimization," *Journal of Machine Learning Research*, vol. 14, no. Feb, pp. 567–599, 2013.

[67] N. L. Roux, M. Schmidt, and F. R. Bach, "A stochastic gradient method with an exponential convergence rate for finite training sets," in *Advances in Neural Information Processing Systems*, 2012, pp. 2663–2671.

[68] P. Zhao and T. Zhang, "Stochastic optimization with importance sampling for regularized loss minimization," in *International Conference on Machine Learning*, 2015, pp. 1–9.

[69] Z. Allen-Zhu and Y. Yuan, "Improved SVRG for non-strongly-convex or sum-of-non-convex objectives," arXiv preprint, Tech. Rep., 2016.

[70] R. Harikandeh, M. O. Ahmed, A. Virani, M. Schmidt, J. Konečnỳ, and S. Sallinen, "Stop-wasting my gradients: Practical SVRG," in *Advances in Neural Information Processing Systems*, 2015, pp. 2251–2259.

[71] P. Gong and J. Ye, "Linear convergence of variance-reduced stochastic gradient without strong convexity," *ArXiv preprint arXiv:1406.1102*, 2014.

[72] A. Nitanda, "Stochastic proximal gradient descent with acceleration techniques," in *Advances in Neural Information Processing Systems*, 2014, pp. 1574–1582.

[73] S. Lacoste-Julien, M. Schmidt, and F. Bach, "A simpler approach to obtaining an o (1/t) convergence rate for the projected stochastic subgradient method," *ArXiv preprint arXiv:1212.2002*, 2012.

[74] J. Lafferty, A. McCallum, and F. C. Pereira, "Conditional random fields: Probabilistic models for segmenting and labeling sequence data," 2001.

[75] C. Sutton, A. McCallum, and K. Rohanimanesh, "Dynamic conditional random fields: Factorized probabilistic models for labeling and segmenting sequence data," *Journal of Machine Learning Research*, vol. 8, no. Mar, pp. 693–723, 2007.

[76] F. Sha and F. Pereira, "Shallow parsing with conditional random fields," in *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology-Volume 1*, Association for Computational Linguistics, 2003, pp. 134–141.

[77] A. McCallum and W. Li, "Early results for named entity recognition with conditional random fields, feature induction and web-enhanced lexicons," in *Proceedings of the seventh conference on Natural language learning at HLT-NAACL 2003-Volume 4*, Association for Computational Linguistics, 2003, pp. 188–191.

[78] S. Nowozin, C. H. Lampert, *et al.*, "Structured learning and prediction in computer vision," *Foundations and Trends® in Computer Graphics and Vision*, vol. 6, no. 3–4, pp. 185–365, 2011.

[79] F. Barahona, "On the computational complexity of ising spin glass models," *Journal of Physics A: Mathematical and General*, vol. 15, no. 10, p. 3241, 1982.

[80] V. Chandrasekaran, N. Srebro, and P. Harsha, "Complexity of inference in graphical models," *ArXiv preprint arXiv:1206.3240*, 2012.

[81] G. D. Forney, "The Viterbi algorithm," *Proceedings of the IEEE*, vol. 61, no. 3, pp. 268–278, 1973.

[82] S. Vishwanathan, N. N. Schraudolph, M. W. Schmidt, and K. P. Murphy, "Accelerated training of conditional random fields with stochastic gradient methods," ACM, 2006, pp. 969–976.

[83] M. Schmidt, R. Babanezhad, M. Ahmed, A. Defazio, A. Clifton, and A. Sarkar, "Non-uniform stochastic average gradient method for training conditional random fields," pp. 819–828, 2015.

[84] R. T. Rust and A. J. Zahorik, "Customer satisfaction, customer retention, and market share," *Journal of Retailing*, vol. 69, no. 2, pp. 193–215, 1993.

[85] F. Shahrokhi and D. W. Matula, "The maximum concurrent flow problem," *Journal of the ACM (JACM)*, vol. 37, no. 2, pp. 318–334, 1990.

[86] R. D. Cox, "Regression models and life tables (with discussion)," *Journal of the Royal Statistical Society*, vol. 34, pp. 187–220, 1972.

[87] D. R. Cox, "Partial likelihood," *Biometrika*, vol. 62, no. 2, pp. 269–276, 1975.

[88] ——, "Regression models and life-tables," in, Springer, 1992, pp. 527–541.

[89] B. Taskar, C. Guestrin, and D. Koller, "Max-margin Markov networks," in *Advances in Neural Information Processing Systems*, 2004, pp. 25–32.