UNIVERSITY OF CALIFORNIA

Los Angeles

Sample-Efficient Nonconvex Optimization Algorithms in Machine Learning and Reinforcement Learning

A dissertation submitted in partial satisfaction
of the requirements for the degree
Doctor of Philosophy in Computer Science

by

Pan Xu

2021

ABSTRACT OF THE DISSERTATION

Sample-Efficient Nonconvex Optimization Algorithms in Machine Learning and Reinforcement Learning

by

Pan Xu
Doctor of Philosophy in Computer Science
University of California, Los Angeles, 2021
Professor Quanquan Gu, Chair

Machine learning and reinforcement learning have achieved tremendous success in solving problems in various real-world applications. Many modern learning problems boil down to a nonconvex optimization problem, where the objective function is the average or the expectation of some loss function over a finite or infinite dataset. Solving such nonconvex optimization problems, in general, can be NP-hard. Thus one often tackles such a problem through incremental steps based on the nature and the goal of the problem: finding a first-order stationary point, finding a second-order stationary point (or a local optimum), and finding a global optimum. With the size and complexity of the machine learning datasets rapidly increasing, it has become a fundamental challenge to design efficient and scalable machine learning algorithms that can improve the performance in terms of accuracy and save computational cost in terms of sample efficiency at the same time. Though many algorithms based on stochastic gradient descent have been developed and widely studied theoretically and empirically for nonconvex optimization, it has remained an open problem whether we can achieve the optimal sample complexity for finding a first-order stationary point and for finding local optima in nonconvex optimization.

In this thesis, we start with the stochastic nested variance reduced gradient (SNVRG) algorithm, which is developed based on stochastic gradient descent methods and variance reduction techniques. We prove that SNVRG achieves the near-optimal convergence rate among its type for finding a first-order stationary point of a nonconvex function. We further build algorithms to efficiently find the local optimum of a nonconvex objective function by examining the curvature information at the stationary point found by SNVRG. With the ultimate goal of finding the global optimum in nonconvex optimization, we then provide a unified framework to analyze the global convergence of stochastic gradient Langevin dynamics-based algorithms for a nonconvex objective function. In the second part of this thesis, we generalize the aforementioned sample-efficient stochastic nonconvex optimization methods to reinforcement learning problems, including policy gradient, actor-critic, and Q-learning. For these problems, we propose novel algorithms and prove that they enjoy state-of-the-art theoretical guarantees on the sample complexity. The works presented in this thesis form an incomplete collection of the recent advances and developments of sample-efficient nonconvex optimization algorithms for both machine learning and reinforcement learning.

The dissertation of Pan Xu is approved.

University of California, Los Angeles

2021

To my family.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ACKNOWLEDGMENTS

First and foremost, I would like to thank my advisor, Quanquan Gu, for his constant support and encouragement during my Ph.D. study. He guided me into academia and helped me develop my research style. Quanquan is not only a great mentor but is also an exceptional researcher who has been my biggest inspiration. His endless passion and profound insight for research have led me through difficult times in research.

I am grateful for my doctoral committee members, Animashree Anandkumar, Cho-Jui Hsieh, Alexander Sherstov, and Lieven Vandenberghe, who have provided valuable feedbacks for me towards the completion of my thesis. I also want to extend my heartfelt gratitude to Jian Ma and Farzad Farnoud for their mentoring and support through multiple collaborations during my Ph.D. study. I am greatly thankful to Georgios Theocharous, Zheng Wen, and Handong Zhao for their help and support, who were my mentors when I did a wonderful internship at Adobe Research.

I have been extremely fortunate to collaborate with many brilliant people, including Aditya Chaudhry, Jinghui Chen, Felicia Gao, Tianyuan Jin, Tao Jin, Lu Tian, Lingxiao Wang, Tianhao Wang, Yue Wu, Xiaokui Xiao, Yaodong Yu, Tingting Zhang, Weitong Zhang, Dongruo Zhou, and Difan Zou. Four chapters are based on our collaborated projects: Chapter 2 is based on work [ZXG18a, ZXG20] with Dongruo, Chapter 3 is based on work [XCZG18] with Jinghui and Difan, Chapter 4 is based on work [XGG20] with Felicia, and Chapter 5 is based on work [WZXG20] with Yue and Weitong. Special thanks to Lu, Lingxiao, Dongruo, and Difan, who had been both fantastic collaborators and amazing roommates to me.

Lastly, I am deeply grateful to my family members, Xiaojia, Huiqiong, Shiyi, and Yang, for their firmest and unconditional love. I could not have completed my Ph.D. and this dissertation without their encouragement, comfort, and support. Therefore, I dedicate this thesis to them.

VITA

2011–2015	Bachelor of Science in Mathematics, University of Science and Technology of China, Hefei, China.
2015–2018	Ph.D. candidate (transferred out) in Computer Science and Systems and Information Engineering, University of Virginia, Virginia, USA.
2018–2021	Ph.D. candidate in Computer Science, UCLA, California, USA.

PUBLICATIONS

This list contains selected publications that are the most relevant to the theme of this thesis.

Speeding Up Latent Variable Gaussian Graphical Model Estimation via Nonconvex Optimization. Pan Xu, Jian Ma, and Quanquan Gu. Advances in Neural Information Processing Systems, 2017.

Accelerated Stochastic Mirror Descent: From Continuous-Time Dynamics to Discrete-Time Algorithms. Pan Xu, Tianhao Wang, and Quanquan Gu. International Conference on Artificial Intelligence and Statistics, 2018.

Continuous and Discrete-Time Accelerated Stochastic Mirror Descent for Strongly Convex Functions. Pan Xu, Tianhao Wang, and Quanquan Gu. International Conference on Machine Learning, 2018.

Global Convergence of Langevin Dynamics Based Algorithms for Nonconvex Optimization. Pan Xu, Jinghui Chen, Difan Zou, and Quanquan Gu. Advances in Neural Information Processing Systems, 2018.

Third-Order Smoothness Helps: Faster Stochastic Optimization Algorithms for Finding Local Minima. Yaodong Yu, Pan Xu, and Quanquan Gu. Advances in Neural Information Processing Systems, 2018

Stochastic Variance-Reduced Hamilton Monte Carlo Methods. Difan Zou, Pan Xu, and Quanquan Gu. International Conference on Machine Learning, 2018

An Improved Convergence Analysis of Stochastic Variance-Reduced Policy Gradient. Pan Xu, Felicia Gao, and Quanquan Gu. Uncertainty in Artificial Intelligence, 2019.

Dongruo Zhou, Pan Xu, and Quanquan Gu. Stochastic Nested Variance Reduction for Nonconvex Optimization. Journal of Machine Learning Research, 2020.

Sample Efficient Policy Gradient Methods with Recursive Variance Reduction. Pan Xu, Felicia Gao, and Quanquan Gu. International Conference on Learning Representations, 2020.

A Finite-Time Analysis of Q-Learning with Neural Network Function Approximation. Pan Xu and Quanquan Gu. International Conference on Machine Learning, 2020.

A Finite-Time Analysis of Two Time-Scale Actor-Critic Methods. Yue Frank Wu, Weitong Zhang, Pan Xu, and Quanquan Gu. Advances in Neural Information Processing Systems, 2020

CHAPTER 1

Introduction

1.1 Overview and Background

We study the following nonconvex optimization problem: $\min_{\mathbf{x}\in\mathbb{R}^d} F(\mathbf{x})$, where F is a nonconvex smooth function. A popular example of this problem is the finite-sum optimization, where the loss function is a sum of n nonconvex component functions:

$$\min_{\mathbf{x}\in\mathbb{R}^d} F(\mathbf{x}) = \frac{1}{n}\sum_{i=1}^{n} f_i(\mathbf{x}), \tag{1.1.1}$$

where each f_i is defined on a different data point. The finite-sum optimization problem (1.1.1) is often regarded as the offline learning setting in the literature [AZL18, FLLZ18]. A closely related variant of the finite-sum optimization problem in (1.1.1) is the following general stochastic optimization problem:

$$\min_{\mathbf{x}\in\mathbb{R}^d} F(\mathbf{x}) = \mathbb{E}_{\xi\sim\mathcal{D}}[F(\mathbf{x};\xi)], \tag{1.1.2}$$

where ξ is a random variable drawn from some fixed but unknown distribution $\mathcal{D}$ and $F(\mathbf{x};\xi)$ is a nonconvex smooth function indexed by ξ. The general stochastic optimization problem defined in (1.1.2) encloses innumerable large-scale machine learning applications which keep generating oceans of data samples. Therefore, (1.1.2) is also referred to as the online learning setting [AZL18].

For either (1.1.1) or (1.1.2), finding the global minimum of such nonconvex optimization problems can be generally NP hard [HL13]. Therefore, instead of finding the global minimum, various optimization methods have been developed to find an ϵ-approximate first-order stationary point of (1.1.1) and (1.1.2), i.e., a point $\mathbf{x}$ satisfying

$$\|\nabla F(\mathbf{x})\|_2 \le \epsilon, \tag{1.1.3}$$

where $\epsilon > 0$ is a predefined precision parameter. This vast body of literature consists of gradient descent (GD), stochastic gradient descent (SGD) [RM51], stochastic variance reduced gradient (SVRG) [RHS$^+$16, AZH16], StochAstic Recursive grAdient algoritHm (SARAH) [NLST17b], stochastically controlled stochastic gradient (SCSG) [LJCJ17] and many others. To avoid unsatisfactory stationary points (saddle points or local maxima), one can further pursue an (ϵ, ϵ_H)-approximate second-order stationary point [NP06] of (1.1.1) and (1.1.2), namely a point $\mathbf{x}$ that satisfies

$$\|\nabla F(\mathbf{x})\|_2 \leq \epsilon, \text{ and } \lambda_{\min}\big(\nabla^2 F(\mathbf{x})\big) \geq -\epsilon_H, \tag{1.1.4}$$

where $\epsilon, \epsilon_H \in (0,1)$ are predefined precision parameters and $\lambda_{\min}(\cdot)$ denotes the minimum eigenvalue of a matrix. An $(\epsilon, \sqrt{\epsilon})$-approximate second-order stationary point is considered as an approximate local minimum of the optimization problem [NP06]. In many tasks such as training a deep neural network, matrix completion and matrix sensing, one have found that local minima have a very good generalization performance [CHM$^+$15, DPG$^+$14] or all local minima are global minima [GLM16, BNS16, ZWYG18]. However, when local minima are also unsatisfactory, one needs to develop global optimization methods. One popular and promising type of approach is sampling from a distribution that concentrates around the global minimum of $F(\mathbf{x})$ [Dal17b, Dal17a]. In this way, we can transform the (global) nonconvex optimization problem into a Bayesian posterior sampling problem, where we can use plentiful methods such as Monte Carlo Markov Chain (MCMC), variational inference, and many others to generate random samples from the desired distribution.

Nonconvex optimization algorithms are being used almost everywhere in machine learning. One of the most exciting but challenging nonconvex optimization problems lies in reinforcement learning (RL) [SB18], which has received significant success in solving various complex problems such as learning robotic motion skills [LWA15], autonomous driving [SSSS16] and Go game [SSS$^+$17], where the agent progressively interacts with the environment to learn a good policy to solve the task. In RL, the agent chooses the action based on the current state and the historical rewards it has received so far. After performing the chosen action, the agent's state will change according to some transition probability model.

A new reward would be revealed to the agent by the environment based on the action and the new state. Then the agent continues to choose its next action until it reaches a terminal state. The agent aims to maximize its expected cumulative rewards. Therefore, the pivotal problem in RL is to efficiently find a good policy which is a function that maps the state space to the action space and thus informs the agent which action to take at each state. Most successful RL algorithms can be categorized into two types: (1) policy gradient methods [SMSM00] that parameterize the policy by an unknown parameter $\boldsymbol{\theta} \in \mathbb{R}^d$ and directly optimizes the policy by finding the optimal $\boldsymbol{\theta}$; and (2) action-value function (Q-function) based methods such as Q-learning that assign a value to each state-action pair and infer the optimal policy based on the action values, both of which can be formulated as a general stochastic nonconvex optimization problem as stated in (1.1.2). In particular, policy gradient methods try to directly optimize the performance function, which is the expected return under a specific policy, while action value-based methods often try to optimize the mean-squared Bellman error, which is based on the optimal Bellman equation for the action-value function.

Compared with the general nonconvex optimization problem, RL problems often bear more unique challenges, and thus the efficient optimization algorithms developed for solving the general problem in (1.1.2) may not be easily generalized to reinforcement learning. In particular, we have to deal with the following challenges when we develop nonconvex optimization algorithms for reinforcement learning: (1) *The data distribution in reinforcement learning is changing over time.* Unlike the optimization problem in supervised learning, where data are drawn from an unknown but fixed distribution, in reinforcement learning, we are learning to optimize the policy while using the policy to generate data for the learning process. Thus the data distribution changes at every step once we update the policy parameter. (2) *The objective function in reinforcement learning is changing over time.* This is especially true if we use gradient-based algorithms to optimize the policy. Since, in practice, we do not know the data distribution or the transition probability in the environment, we can only sample a batch of data to approximate the gradient of the optimization problem. (3) *The optimization trajectories are highly dependent in reinforcement learning.* Since the

Bellman equation involves the current state-action pair and further state-action pairs, the gradients used in action-value function-based methods such as temporal difference learning and Q-learning are not i.i.d. even conditioned on the current iterate. Therefore, it is an exciting but challenging task to develop efficient reinforcement learning methods based on tools from nonconvex optimization.

1.2 Organization of Chapters

In the first part of this thesis, we present a series of works addressing the problems of finding the first-order stationary points, finding the local minima, and finding the global minima in nonconvex optimization. In particular, in Chapter 2, we develop the stochastic nested variance reduced gradient algorithm (SNVRG) and prove that SNVRG enjoys the state-of-the-art gradient complexity for finding a stationary point in nonconvex optimization. We also extend it to an algorithm that finds local minima only using first-order information and prove that it enjoys a faster convergence rate to local minima when the objective function satisfies the third-order smoothness condition. To find the global minima in nonconvex optimization, in Chapter 3, we present a unified framework for analyzing the global convergence of Langevin dynamics-based algorithms. We prove that stochastic gradient Langevin dynamics (SGLD), variance reduced stochastic gradient Langevin dynamics (SVRG-LD), and other variants of SGLD converge to an almost global minimizer of a nonconvex objective function.

In the second part of the thesis, we generalize the efficient nonconvex optimization algorithms developed for general stochastic optimization problems to reinforcement learning, where we address the challenges mentioned above with a diverse set of techniques. In Chapter 4, we propose the stochastic recursive variance reduced policy gradient (SRVR-PG) algorithm, which significantly improves the sample efficiency of previous policy gradient methods by using similar variance reduction techniques in the last part for general nonconvex optimization problems and using importance sampling to control the changes in data distribution in RL. In Chapter 5, we study the finite-time analysis of the two time-scale actor-critic methods, which use linear function approximation to learn the value function of the current policy that changes over time. Finally, in Chapter 6, we study the finite-time convergence of Q-learning with neural network function approximation, where non-i.i.d. data with Markov noises are considered in the analysis.

1.3 Notations

Throughout this thesis, unless particularly specified, we will use the following notations. We denote $[n] = \{1, \ldots, n\}$ for any $n \in \mathbb{N}^+$. $\|\mathbf{x}\|_2$ is the Euclidean norm of a vector $\mathbf{x} \in \mathbb{R}^d$. For a matrix $\mathbf{W} \in \mathbb{R}^{m\times n}$, we denote by $\|\mathbf{W}\|_2$ and $\|\mathbf{W}\|_F$ its operator norm and Frobenius norm respectively. We denote by $\mathrm{vec}(\mathbf{W})$ the vectorization of $\mathbf{W}$, which converts $\mathbf{W}$ into a column vector. For a semi-definite matrix $\boldsymbol{\Sigma} \in \mathbb{R}^{d\times d}$ and a vector $\mathbf{x} \in \mathbb{R}^d$, $\|\mathbf{x}\|_{\boldsymbol{\Sigma}} = \sqrt{\mathbf{x}^\top \boldsymbol{\Sigma} \mathbf{x}}$ denotes the Mahalanobis norm. We write $a_n = O(b_n)$ (or $a_n = \Omega(b_n)$) if $a_n \le C b_n$ (or $a_n \ge C b_n$) for some constant $C > 0$. We use $\tilde{O}(\cdot)$ notation to hide logarithmic terms and constants, which means $a_n = \tilde{O}(b_n)$ if and only if $a_n = O(b_n)$ up to some logarithmic term or constant. We also denote $a_n \lesssim b_n$ ($a_n \gtrsim b_n$) if a_n is less than (larger than) b_n up to a constant.

The Dirac delta function $\delta(x)$ satisfies $\delta(0) = +\infty$ and $\delta(x) = 0$ if $x \neq 0$. Note that $\delta(x)$ satisfies $\int_{-\infty}^{+\infty} \delta(x) \mathrm{d}x = 1$. For any $\alpha > 0$, we define the Rényi divergence [Ro61] between distributions P and Q as $D_\alpha(P||Q) = 1/(\alpha - 1) \log_2 \int_x P(x)(P(x)/Q(x))^{\alpha-1} \mathrm{d}x$, which is non-negative for all $\alpha > 0$. The exponentiated Rényi divergence is $d_\alpha(P||Q) = 2^{D_\alpha(P||Q)}$. $d_{TV}(P, Q)$ is the total variation norm between two probability measure P and Q, which is defined as $d_{TV}(P, Q) = 1/2 \int_{\mathcal{X}} |P(dx) - Q(dx)|$.

Part I

Efficient Algorithms for General Nonconvex Optimization Problems

CHAPTER 2

Local Convergence of Stochastic Algorithms for Nonconvex Optimization

2.1 Introduction

In this chapter, we focus on the finite-sum optimization problem defined in (1.1.1). Among all first-order methods such as gradient descent (GD), stochastic gradient descent (SGD) [RM51], stochastic variance reduced gradient (SVRG) [RHS+16, AZH16], StochAstic Recursive grAdient algoritHm (SARAH) [NLST17b], and stochastically controlled stochastic gradient (SCSG) [LJCJ17], SCSG achieves the lowest gradient complexity[1] $O(n \wedge \epsilon^{-2} + \epsilon^{-10/3} \wedge (n^{2/3}\epsilon^{-2}))$ for finding the first-order stationary point (defined in (1.1.3)) in nonconvex optimization under the smoothness (i.e., gradient Lipschitzness) and bounded stochastic gradient variance assumptions. The key idea behind variance reduction is that the gradient complexity can be saved if the algorithm use history information as *reference*. For instance, the representative variance reduction method SVRG is based on a semi-stochastic gradient that is defined by two reference points. Since the the variance of this semi-stochastic gradient will diminish when the iterate gets closer to the minimizer, it therefore accelerates the convergence of stochastic gradient method. A natural and long standing question is:

Is there still room for improvement in nonconvex finite-sum optimization without making additional assumptions beyond smoothness and bounded stochastic gradient variance?

In this chapter, we provide an affirmative answer to the above question, by showing that

[1]We usually use gradient complexity, the number of stochastic gradient evaluations, to measure the convergence speed of different first-order algorithms.

the dependence on n in the gradient complexity of SVRG [RHS+16, AZH16] and SCSG [LJCJ17] can be further reduced. We propose a novel algorithm namely Stochastic Nested Variance-Reduced Gradient descent (SNVRG). Similar to SVRG and SCSG, our proposed algorithm works in a multi-epoch way. Nevertheless, the technique we developed is highly nontrivial. At the core of our algorithm is the multiple reference points-based variance reduction technique in each iteration. In detail, inspired by SVRG and SCSG, which uses two reference points to construct a semi-stochastic gradient with diminishing variance, our algorithm uses $K+1$ reference points to construct a semi-stochastic gradient, whose variance decays faster than that of the semi-stochastic gradient used in SVRG and SCSG.

Based on the SNVRG algorithm we proposed for finding the first-order stationary point in nonconvex optimization, we take a step further to propose faster algorithms for finding the second-order stationary point (defined in (1.1.4)). More specifically, we present a novel algorithm that can find local minima faster than existing algorithms [XRY18, AZL18, YXG18] in a wide regime for the finite-sum optimization problem (1.1.1). The proposed algorithms essentially use Neon2 [AZL18] to extract the negative curvature direction based on gradient evaluation, which saves Hessian-vector computation.

2.1.1 Contribution

We summarize the major contributions of this chapter as follows:

- We propose a stochastic nested variance reduced gradient (SNVRG) algorithm for nonconvex optimization, which reduces the dependence of the gradient complexity on n compared with SVRG and SCSG.

- We show that our proposed algorithm is able to find an ϵ-approximate stationary point with $\tilde{O}(n \wedge \epsilon^{-2} + \epsilon^{-3} \wedge n^{1/2}\epsilon^{-2})$ stochastic gradient evaluations, which outperforms all existing first-order algorithms such as GD, SGD, SVRG and SCSG. A detailed comparison is demonstrated in Figure 2.1.

- We further propose an algorithm, SNVRG + Neon2$^{\text{finite}}$, that can find an (ϵ, ϵ_H) second-order stationary point of the finite-sum problem (1.1.1) within $\tilde{O}(n^{1/2}\epsilon^{-2}+n\epsilon_H^{-3}+n^{3/4}\epsilon_H^{-7/2})$ stochastic gradient evaluations, which is faster than the best existing algorithm SVRG + Neon2$^{\text{finite}}$ [AZL18] that attains $\tilde{O}(n^{2/3}\epsilon^{-2} + n\epsilon_H^{-3} + n^{3/4}\epsilon_H^{-7/2})$ gradient complexity in a wide regime. A thorough comparison is illustrated in Figure 2.3.

- We also show that our proposed algorithms can find local minima even faster when the objective function enjoys the third-order smoothness property. We prove that our proposed algorithms achieve faster convergence rates to a local minimum than the FLASH algorithm proposed in [YXG18], which also exploits the third-order smoothness of the objective function.

2.2 Background and Related Work

In this section, we review and discuss the relevant work in the literature of nonconvex optimization for solving the finite-sum problem (1.1.1).

Finding first-order stationary points For nonconvex optimization, it is well-known that Gradient Descent (GD) can converge to an ϵ-approximate stationary point with $O(n\cdot\epsilon^{-2})$ [Nes13] number of stochastic gradient evaluations. GD needs to calculate the full gradient at each iteration, which is a heavy load when $n \gg 1$. Stochastic gradient descent (SGD) [RM51, Nes13] and its variants [GL13, GL16, GLZ16] achieve $O(1/\epsilon^4)$ gradient complexity under the assumption that the stochastic gradient has a bounded variance. Inspired by the great success of various variance reduced techniques in convex finite-sum optimization such as Stochastic Average Gradient (SAG) [RSB12], Stochastic Variance Reduced Gradient (SVRG) [JZ13, XZ14], SAGA [DBLJ14], Stochastic Dual Coordinate Ascent (SDCA) [SSZ13], Finito [DDo14] and Batching SVRG [HAV$^+$15], [GH15, SS16] first analyzed the convergence of SVRG under nonconvex setting, where F is still convex but each component function f_i can be nonconvex. The analysis for the general nonconvex function F was done by [RHS$^+$16, AZH16], which shows that SVRG can converge to an ϵ-approximate stationary point with $O(n^{2/3}\cdot\epsilon^{-2})$ number of stochastic gradient evaluations. This result is strictly better than that of GD. [NLST17a, NLST17b] proposed StochAstic Recursive grAdient algoritHm (SARAH) with recursive estimators for finding first-order stationary points with $O(n + L^2/\epsilon^4)$ stochastic gradient evaluations. [LJCJ17] proposed a new variance reduction algorithm, i.e., the stochastically controlled stochastic gradient (SCSG) algorithm, which finds a first-order stationary point within $O(\min\{\epsilon^{-10/3}, n^{2/3}\epsilon^{-2}\})$ stochastic gradient evaluations for finite-sum optimization in (1.1.1), and outperforms SVRG when the number of component functions n is large.

The literature of finding local minima in nonconvex optimization can be roughly divided into three categories according to the oracles they use: Hessian-based, Hessian-vector product-based and gradient-based (Hessian-free). We review each category in the sequel accordingly.

Finding local minima using Hessian matrix The most popular algorithm using Hessian matrix to find an $(\epsilon, \sqrt{\epsilon})$-approximate local minimum is the cubic regularized Newton's method [NP06], which attains $O(\epsilon^{-3/2})$ iteration complexity. The trust region method is proved to achieve the same iteration complexity [CRS17]. To alleviate the computation burden of evaluating full gradients and Hessian matrices in large-scale optimization problems, subsampled cubic regularization and trust-region methods [KL17, XRM20] were proposed and proved to enjoy the same iteration complexity as their original versions with full gradients and Hessian matrices. Recently, stochastic variance reduced cubic regularization method (SVRC) [ZXG18b] was proposed, which achieves the best-known second-order oracle complexity among existing cubic regularization methods.

Finding local minima using Hessian-vector product Another line of research uses Hessian-vector products to find the second-order stationary points. [CDHS16, AAB$^+$17] independently proposed two algorithms that can find an $(\epsilon, \sqrt{\epsilon})$-approximate local minimum within $O(\epsilon^{-7/4})$ full gradient and Hessian-product evaluations. [AAB$^+$17] also showed that their algorithm only needs $O(n\epsilon^{-3/2} + n^{3/4}\epsilon^{7/4})$ stochastic gradient and Hessian-vector product evaluations for finite-sum optimization problems (1.1.1). [RZS$^+$18] proposed a generic algorithmic framework that uses both first-order and second-order methods to find the local minimum within $O(n^{2/3}\epsilon^{-2} + n\epsilon^{-3/2} + n^{3/4}\epsilon^{7/4})$ stochastic gradient and Hessian-product evaluations. [AZ18] proposed the Natasha2 algorithm which finds an $(\epsilon, \sqrt{\epsilon})$-approximate second-order stationary point within $O(\epsilon^{-7/2})$ stochastic gradient and Hessian-vector product evaluations.

Finding local minima using gradient The last line of research uses purely gradient information to find the local minima. The local minima finding algorithms proposed in this chapter also fall into this category. [GHJY15, Lev16] studied the perturbed GD and SGD algorithms for escaping saddle points, where isotropic noise is added into the gradient or stochastic gradient at each iteration or whenever the gradient is sufficiently small. [JGN$^+$17] further proposed a perturbed accelerated gradient descent, which can finds the second-order stationary point even faster. [XRY18] showed that perturbed gradient or stochastic gradient descent can help find the negative curvature direction without using Hessian matrix and proposed the NEON algorithm that extracts the negative curvature using only first-order information. Later [AZL18] developed the Neon2 algorithm, which improves upon on Neon, and turns Natasha2 [AZ18] into a first-order method to find the local minima. [YZG17] proposed the gradient descent with one-step escaping algorithm (GOSE) that saves negative curvature computation and [YXG18] proposed the FLASH algorithm that exploits the third-order smoothness of the objective function. Very recently, [DKLH18] proved that SGD with periodically changing step size can escape from saddle points under an additional correlated negative curvature (CNC) assumption on the stochastic gradient.

To give a thorough comparison of our proposed SNVRG algorithm with existing algorithms for nonconvex finite-sum optimization, we summarize the gradient complexity of the most relevant algorithms in Table 2.1 for finding first-order stationary points and in Table 2.2 for finding local minimum using first-order information. We also present the gradient complexities of first-order local minimum finding algorithms in Table 2.2. According to Table 2.1, the proposed SNVRG algorithm achieves the lowest gradient complexity to find an ϵ-approximate first-order stationary point for nonconvex smooth functions. We can also see from Table 2.2 that our proposed algorithm SNVRG + Neon2$^{\text{finite}}$ outperforms all other first-order algorithms in finding an (ϵ, ϵ_H)-approximate second-order stationary point for finite-sum nonconvex optimization problems in a wide regime.

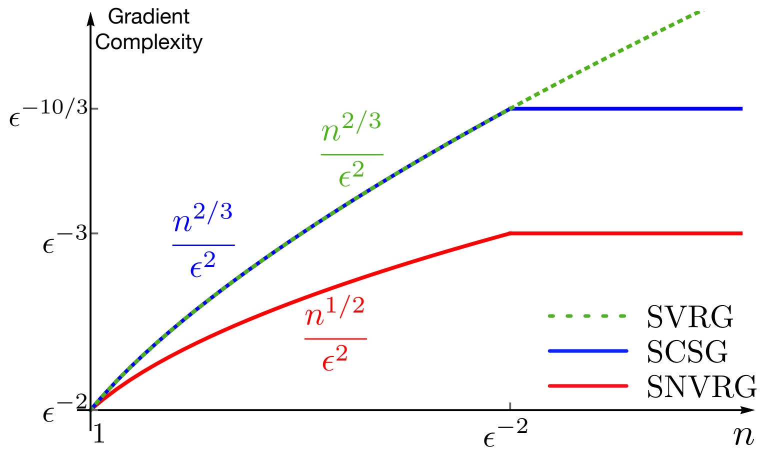


Figure 2.1: Comparison of gradient complexities.

After the first appearance of our SNVRG algorithm in a conference paper [ZXG18a], there have emerged a considerable amount of exciting work on this topic. [FLLZ18] concurrently proposed the Stochastic Path-Integrated Differential EstimatoR (SPIDER), which uses recursive update to define the semi-stochastic gradient in the variance reduction algorithm. They proved that SPIDER achieves $O(n^{1/2}\epsilon^{-2} \wedge \epsilon^{-3})$ gradient complexity for finding an ϵ-approximate stationary point in nonconvex optimization. [WJZ$^+$19] proposed an improved analysis for SPIDER (also called SpiderBoost) and SPIDER with momentum. Note that all the aforementioned algorithms enjoy a similar convergence rate to SPIDER [FLLZ18]. [FLLZ18, ZG19a] also showed that both SPIDER and SNVRG are near optimal with respect to the gradient complexity. In a recent work, [FLZ19] proposed a tighter analysis of the gradient complexity for SGD to escape saddle points.

2.3 Preliminaries

In this section, we present some definitions that will be used throughout this chapter.

Definition 2.3.1 (Smoothness). *$f : \mathbb{R}^d \to \mathbb{R}$ is L_1-smooth for some constant $L_1 > 0$, if it is differentiable and satisfies*

$$\|\nabla f(\mathbf{x}) - \nabla f(\mathbf{y})\|_2 \leq L_1\|\mathbf{x} - \mathbf{y}\|_2, \quad \textit{for any } \mathbf{x}, \mathbf{y} \in \mathbb{R}^d. \tag{2.3.1}$$

Table 2.1: Comparisons on gradient complexity of different algorithms. The second column shows the gradient complexity for a nonconvex and smooth function to achieve an ϵ-approximate stationary point (i.e., $\|\nabla F(\mathbf{x})\|_2 \le \epsilon$). The last column indicates whether the algorithm needs the Hessian Lipschitz condition in their analyses.

Algorithm	Gradient complexity	Need Hessian Lipschitz?
GD	$O\big(\frac{n}{\epsilon^2}\big)$	✗
SGD	$O\big(\frac{1}{\epsilon^4}\big)$	✗
SVRG [RHS+16]	$O\big(\frac{n^{2/3}}{\epsilon^2}\big)$	✗
SCSG [LJCJ17]	$O\big(\frac{1}{\epsilon^{10/3}} \wedge \frac{n^{2/3}}{\epsilon^2}\big)$	✗
GNC-AGD [CDHS17]	$\tilde{O}\big(\frac{n}{\epsilon^{1.75}}\big)$	✓
Natasha 2 [AZ18]	$\tilde{O}\big(\frac{1}{\epsilon^{3.25}}\big)$	✓
SNVRG (Algorithm 2)	$\tilde{O}\big(\frac{1}{\epsilon^3} \wedge \frac{n^{1/2}}{\epsilon^2}\big)$	✗

Definition 2.3.1 implies that if f is L-smooth, we have for any $\mathbf{x}, \mathbf{h} \in \mathbb{R}^d$

$$f(\mathbf{x}+\mathbf{h}) \le f(\mathbf{x}) + \langle \nabla f(\mathbf{x}), \mathbf{h}\rangle + \frac{L}{2}\|\mathbf{h}\|_2^2. \tag{2.3.2}$$

Definition 2.3.2 (Hessian Lipschitzness)**.** *$f : \mathbb{R}^d \to \mathbb{R}$ is L_2-Hessian Lipschitz for some constant $L_2 > 0$, if it is twice-differentiable and satisfies*

$$\|\nabla^2 f(\mathbf{x}) - \nabla^2 f(\mathbf{y})\|_2 \le L_2 \|\mathbf{x}-\mathbf{y}\|_2, \quad \textit{for any } \mathbf{x}, \mathbf{y} \in \mathbb{R}^d.$$

The above two smoothness conditions are widely used in nonconvex optimization problems [NP06]. We will call them first-order smoothness and second-order smoothness respectively in this chapter. As shown in [CDHS17, YXG18], when the objective function has additionally third-order smoothness, one can design algorithms that find local minima even faster. Following [YXG18], we denote the three-way tensor $\nabla^3 f(\mathbf{x}) \in \mathbb{R}^{d\times d\times d}$ as the third-order derivative of f.

Definition 2.3.3 (Third-order Derivative)**.** *The third-order derivative of function $f : \mathbb{R}^d \to$*

Table 2.2: Comparisons on gradient complexities to find an (ϵ, ϵ_H)-approximate second-order stationary point in finite-sum nonconvex optimization. The last column indicates whether the algorithm exploits the third-order smoothness of the objective function.

Algorithm	Gradient complexity	Need 3rd-order smooth?
PGD [JGN+17]	$\tilde{O}\big(\frac{n}{\epsilon^2}\big)$ (for $\epsilon_H \geq \epsilon^{1/2}$)	✗
SVRG + Neon2$^{\text{finite}}$ [AZL18]	$\tilde{O}\Big(\frac{n^{2/3}}{\epsilon^2} + \frac{n}{\epsilon_H^3} + \frac{n^{3/4}}{\epsilon_H^{7/2}}\Big)$	✗
FLASH [YXG18]	$\tilde{O}\Big(\frac{n^{2/3}}{\epsilon^2} + \frac{n}{\epsilon_H^2} + \frac{n^{3/4}}{\epsilon_H^{5/2}}\Big)$	✓
SNVRG + Neon2$^{\text{finite}}$ (Algorithm 3)	$\tilde{O}\Big(\frac{n^{1/2}}{\epsilon^2} + \frac{n}{\epsilon_H^3} + \frac{n^{3/4}}{\epsilon_H^{7/2}}\Big)$	✗
SNVRG + Neon2$^{\text{finite}}$ (Algorithm 3)	$\tilde{O}\Big(\frac{n^{1/2}}{\epsilon^2} + \frac{n}{\epsilon_H^2} + \frac{n^{3/4}}{\epsilon_H^{5/2}}\Big)$	✓

$\mathbb{R}$ *is defined as a three-way tensor* $\nabla^3 f(\mathbf{x}) \in \mathbb{R}^{d\times d\times d}$, *where*

$$[\nabla^3 f(\mathbf{x})]_{ijk} = \frac{\partial}{\partial x_i \partial x_j \partial x_k} f(\mathbf{x}), \quad i, j, k = 1, \ldots, d \textit{ and } \mathbf{x} \in \mathbb{R}^d.$$

Now we are ready to present the formal definition of third-order smoothness, which has been explored in [AG16, CDHS17, YXG18]. It is also called third-order derivative Lipschitzness in [CDHS17].

Definition 2.3.4 (Third-order Smoothness). $f : \mathbb{R}^d \to \mathbb{R}$ *is* L_3*-third-order smooth for some constant* $L_3 > 0$*, if it is thrice-differentiable and satisfies*

$$\|\nabla^3 f(\mathbf{x}) - \nabla^3 f(\mathbf{y})\|_F \leq L_3 \|\mathbf{x} - \mathbf{y}\|_2, \quad \textit{for any } \mathbf{x}, \mathbf{y} \in \mathbb{R}^d.$$

The following definition characterizes the distance between the initial point of an algorithm and the minimizer of function f.

Definition 2.3.5 (Optimal Gap). *The optimal gap of* f *at point* $\mathbf{x}_0$ *is denoted by* Δ_f *and*

$$f(\mathbf{x}_0) - \min_{\mathbf{x}\in\mathbb{R}^d} f(\mathbf{x}) \le \Delta_f.$$

W.L.O.G., we assume $\Delta_f < +\infty$.

Definition 2.3.6. $f : \mathbb{R}^d \to \mathbb{R}$ *is* λ*-strongly convex for some constant* $\lambda > 0$*, if it satisfies*

$$f(\mathbf{x}+\mathbf{h}) \ge f(\mathbf{x}) + \langle \nabla f(\mathbf{x}), \mathbf{h}\rangle + \frac{\lambda}{2}\|\mathbf{h}\|_2^2, \quad \textit{for any } \mathbf{x},\mathbf{y} \in \mathbb{R}^d. \tag{2.3.3}$$

While the above definitions are based on a general function f, the following two definitions rely on the finite-sum structure of F defined in (1.1.1).

Definition 2.3.7. *A function* F *with finite-sum structure in* (1.1.1) *is said to have stochastic gradients with bounded variance* σ^2*, if for any* $\mathbf{x} \in \mathbb{R}^d$*, we have*

$$\mathbb{E}_i\|\nabla f_i(\mathbf{x}) - \nabla F(\mathbf{x})\|_2^2 \le \sigma^2, \tag{2.3.4}$$

where i *a random index uniformly chosen from* $[n]$ *and* $\mathbb{E}_i$ *denotes the expectation over such* i*.*

σ^2 is called the upper bound on the variance of stochastic gradients [LJCJ17].

Definition 2.3.8. *A function* F *with finite-sum structure in* (1.1.1) *is said to have averaged* L*-Lipschitz gradient, if for any* $\mathbf{x},\mathbf{y} \in \mathbb{R}^d$*, we have*

$$\mathbb{E}_i\|\nabla f_i(\mathbf{x}) - \nabla f_i(\mathbf{y})\|_2^2 \le L^2\|\mathbf{x}-\mathbf{y}\|_2^2, \tag{2.3.5}$$

where i *is a random index uniformly chosen from* $[n]$ *and* $\mathbb{E}_i$ *denotes the expectation over the choice.*

It should be noted that the smoothness condition of each f_i in Definition 2.3.1 will directly imply the averaged L-Lipschitz gradient for F.

Inspired by the SCSG algorithm [LJCJ17], we will use the property of geometric distribution in our algorithm design. The definition of geometric random variable is as follows.

Definition 2.3.9 (Geometric Distribution)**.** *A random variable* X *follows a geometric distribution with parameter* p*, denoted as* $Geom(p)$*, if it holds that*

$$\mathbb{P}(X = k) = p(1-p)^k, \quad \forall k = 0, 1, \ldots.$$

Definition 2.3.10 (Sub-Gaussian Stochastic Gradient)**.** *We say a function F has σ^2-sub-Gaussian stochastic gradient $\nabla F(\mathbf{x};\xi)$ for any $\mathbf{x} \in \mathbb{R}^d$ and random variable $\xi \sim \mathcal{D}$, if it satisfies*

$$\mathbb{E}\bigg[\exp\bigg(\frac{\|\nabla F(\mathbf{x};\xi) - \nabla f(\mathbf{x})\|_2^2}{\sigma^2}\bigg)\bigg] \leq \exp(1).$$

Note that Definition 2.3.10 implies $\mathbb{E}[\|\nabla F(\mathbf{x};\xi) - \nabla f(\mathbf{x})\|_2^2] \leq 2\sigma^2$ [Ver10]. In the finite-sum optimization setting (1.1.1), we call $\nabla f_i(\mathbf{x})$ a stochastic gradient of function F for a randomly chosen index $i \in [n]$, and we say F has σ^2-sub-Gaussian stochastic gradient if $\mathbb{E}[\|\nabla f_i(\mathbf{x}) - \nabla F(\mathbf{x})\|_2^2] \leq 2\sigma^2$.

2.4 Stochastic Nested Variance-Reduced Gradient Descent

In this section, we present our nested stochastic variance reduction algorithm, namely, SNVRG for finding first-order stationary points in nonconvex optimization.

One-epoch-SNVRG: We first present the key component of our main algorithm, One-epoch-SNVRG, which is displayed in Algorithm 1. The most innovative part of Algorithm 1 attributes to the $K+1$ *reference points* and $K+1$ *reference gradients.* Note that when $K=1$, Algorithm 1 reduces to one epoch of SVRG algorithm [JZ13, RHS$^+$16, AZH16]. To better understand our One-epoch-SNVRG algorithm, it would be helpful to revisit the original SVRG which is a special case of our algorithm. For the finite-sum optimization problem in (1.1.1), the original SVRG takes the following updating formula

$$\mathbf{x}_{t+1} = \mathbf{x}_t - \eta\mathbf{v}_t = \mathbf{x}_t - \eta\big(\nabla F(\tilde{\mathbf{x}}) + \nabla f_{i_t}(\mathbf{x}_t) - \nabla f_{i_t}(\tilde{\mathbf{x}})\big),$$

where $\eta > 0$ is the step size, i_t is a random index uniformly chosen from $[n]$ and $\tilde{\mathbf{x}}$ is a snapshot for $\mathbf{x}_t$ after every T_1 iterations. There are two reference points in the update formula at $\mathbf{x}_t$: $\mathbf{x}_t^{(0)} = \tilde{\mathbf{x}}$ and $\mathbf{x}_t^{(1)} = \mathbf{x}_t$. Note that $\tilde{\mathbf{x}}$ is updated every T_1 iterations, namely, $\tilde{\mathbf{x}}$ is set to be $\mathbf{x}_t$ only when $(t \mod T_1) = 0$. Moreover, in the semi-stochastic gradient $\mathbf{v}_t$, there are also two reference gradients and we denote them by $\mathbf{g}_t^{(0)} = \nabla F(\tilde{\mathbf{x}})$ and $\mathbf{g}_t^{(1)} = \nabla f_{i_t}(\mathbf{x}_t) - \nabla f_{i_t}(\tilde{\mathbf{x}}) = \nabla f_{i_t}(\mathbf{x}_t^{(1)}) - \nabla f_{i_t}(\mathbf{x}_t^{(0)})$.

Back to our One-epoch-SNVRG, we can define similar reference points and reference

Algorithm 1 One-epoch-SNVRG($\mathbf{x}_0$, $F, K, M, \{T_l\}, \{B_l\}, B_0$)

```
1: Input: initial point $\mathbf{x}_{-1}^{(l)} \leftarrow \mathbf{x}_0$, $l \in [K]$; function $F$; loop number $K$; step size parameter $M$; loop parameters $\{T_l\}$; batch parameters $\{B_l\}$, base batch size $B_0$.
2: Option I  $T = \prod_{l=1}^{K} T_l$
3: Option II $T \sim \text{Geom}(1/(1 + \prod_{l=1}^{K} T_l))$
4: for $t = 0, \ldots, T-1$ do
5:     $r = \min\{j : 0 = (t \mod \prod_{l=j+1}^{K} T_l),\ 0 \le j \le K\}$
6:     $\{\mathbf{x}_t^{(l)}\} \leftarrow \text{Update\_reference\_points}(\{\mathbf{x}_{t-1}^{(l)}\}, \mathbf{x}_t, r), 0 \le l \le K$.
7:     $\{\mathbf{g}_t^{(l)}\} \leftarrow \text{Update\_reference\_gradients}(\{\mathbf{g}_{t-1}^{(l)}\}, \{\mathbf{x}_t^{(l)}\}, r), 0 \le l \le K$.
8:     $\mathbf{v}_t \leftarrow \sum_{l=0}^{K} \mathbf{g}_t^{(l)}$
9:     $\mathbf{x}_{t+1} \leftarrow \mathbf{x}_t - 1/(10M) \cdot \mathbf{v}_t$
10: end for
11: $\mathbf{x}_{\text{out}} \leftarrow$ uniformly random choice from $\{\mathbf{x}_t\}$, where $0 \le t < \prod_{l=1}^{K} T_l$
12: Output: $[\mathbf{x}_{\text{out}}, \mathbf{x}_T]$

13: Function: Update_reference_points($\{\mathbf{x}_{\text{old}}^{(l)}\}, \mathbf{x}, r$)
14: $\mathbf{x}_{\text{new}}^{(l)} \leftarrow \mathbf{x}_{\text{old}}^{(l)}, 0 \le l \le r-1$; $\mathbf{x}_{\text{new}}^{(l)} \leftarrow \mathbf{x}, r \le l \le K$
15: return $\{\mathbf{x}_{\text{new}}^{(l)}\}$

16: Function: Update_reference_gradients($\{\mathbf{g}_{\text{old}}^{(l)}\}, \{\mathbf{x}_{\text{new}}^{(l)}\}, r$)
17: if $r > 0$ then
18:     $\mathbf{g}_{\text{new}}^{(l)} \leftarrow \mathbf{g}_{\text{old}}^{(l)}, 0 \le l < r$; $\mathbf{g}_{\text{new}}^{(l)} \leftarrow 0, r+1 \le l \le K$
19:     Uniformly generate index set $I \subset [n]$ without replacement, $|I| = B_r$
20:     $\mathbf{g}_{\text{new}}^{(r)} \leftarrow 1/B_r \sum_{i \in I} \big[\nabla f_i(\mathbf{x}_{\text{new}}^{(r)}) - \nabla f_i(\mathbf{x}_{\text{new}}^{(r-1)})\big]$
21: else
22:     Uniformly generate index set $I \subset [n]$ without replacement, $|I| = B_0$
23:     $\mathbf{g}_{\text{new}}^{(0)} \leftarrow 1/B_0 \sum_{i \in I} \nabla f_i(\mathbf{x}_{\text{new}}^{(0)})$; $\mathbf{g}_{\text{new}}^{(l)} \leftarrow 0, 1 \le l \le K$
24: end if
25: return $\{\mathbf{g}_{\text{new}}^{(l)}\}$.
```

gradients as that in the special case of SVRG. Specifically, for $t = 0, \ldots, \prod_{l=1}^{K} T_l - 1$, each point $\mathbf{x}_t$ has $K+1$ reference points $\{\mathbf{x}_t^{(l)}\}, l = 0, \ldots, K$, which is set to be $\mathbf{x}_t^{(l)} = \mathbf{x}_{t^l}$ with index t^l defined as

$$t^l = \left\lfloor \frac{t}{\prod_{k=l+1}^{K} T_k} \right\rfloor \cdot \prod_{k=l+1}^{K} T_k. \tag{2.4.1}$$

Specially, note that we have $\mathbf{x}_t^{(0)} = \mathbf{x}_0$ and $\mathbf{x}_t^{(K)} = \mathbf{x}_t$ for all $t = 0, \ldots, \prod_{l=1}^{K} T_l - 1$. Similarly, $\mathbf{x}_t$ also has $K+1$ reference gradients $\{\mathbf{g}_t^{(l)}\}$, which can be defined based on the reference points $\{\mathbf{x}_t^{(l)}\}$:

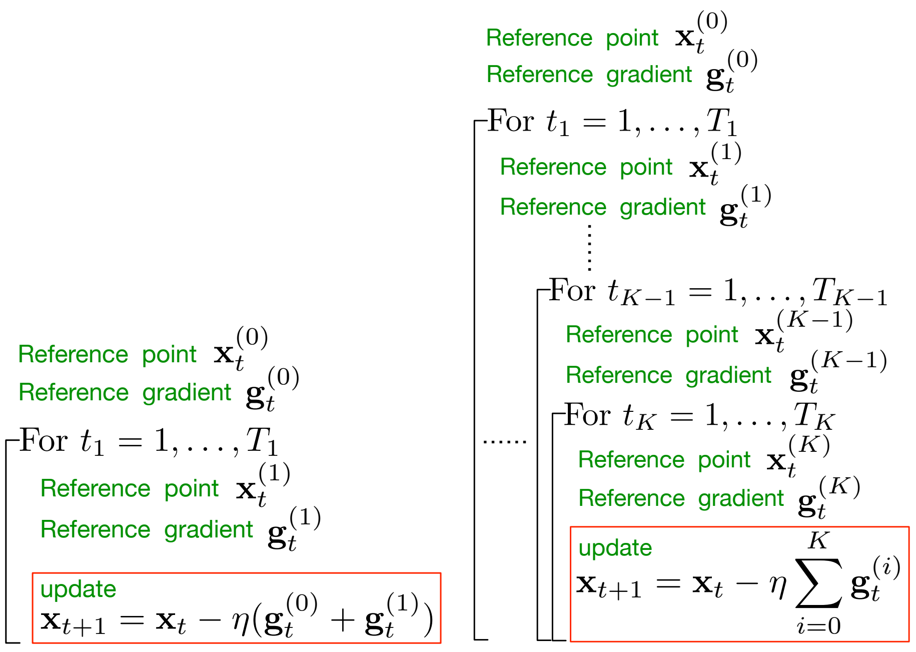
$$\mathbf{g}_t^{(0)} = \frac{1}{B} \sum_{i \in I} \nabla f_i(\mathbf{x}_0), \qquad \mathbf{g}_t^{(l)} = \frac{1}{B_l} \sum_{i \in I_l} \big[\nabla f_i(\mathbf{x}_t^{(l)}) - \nabla f_i(\mathbf{x}_t^{(l-1)})\big], l = 1, \ldots, K, \tag{2.4.2}$$

where I, I_l are random index sets with $|I| = B, |I_l| = B_l$ and are uniformly generated from $[n]$ without replacement. Based on the reference points and reference gradients, we then update $\mathbf{x}_{t+1} = \mathbf{x}_t - 1/(10M) \cdot \mathbf{v}_t$, where $\mathbf{v}_t = \sum_{l=0}^{K} \mathbf{g}_t^{(l)}$ and M is the step size parameter. The illustration of reference points and gradients of SNVRG is displayed in Figure 2.2(b).

We remark that it would be a huge waste for us to re-evaluate $\mathbf{g}_t^{(l)}$ at each iteration. Fortunately, due to the fact that each reference point is only updated after a long period, we can maintain $\mathbf{g}_t^{(l)} = \mathbf{g}_{t-1}^{(l)}$ and only need to update $\mathbf{g}_t^{(l)}$ when $\mathbf{x}_t^{(l)}$ has been updated as is suggested by Line 20 in Algorithm 1.

SNVRG: Using One-epoch-SNVRG (Algorithm 1) as a building block, we now present our main algorithm: Algorithm 2, for finding an ϵ-approximate stationary point in nonconvex finite-sum optimization. At each iteration of Algorithm 2, it executes One-epoch-SNVRG (Algorithm 1) which takes $\mathbf{z}_{s-1}$ as its input and outputs $[\mathbf{y}_s, \mathbf{z}_s]$. We choose $\mathbf{y}_{\text{out}}$ as the output of Algorithm 2 uniformly from $\{\mathbf{y}_s\}$, for $s = 1, \ldots, S$.

Space complexity: We briefly compare the space complexity between our algorithms and other variance reduction based algorithms. SVRG and SCSG needs $O(d)$ space complexity to store one reference gradient, SAGA [DBLJ14] needs to store reference gradients for each component functions, and its space complexity is $O(nd)$ without using any trick. For our algorithm SNVRG, we need to store K reference gradients, thus its space complexity is


Reference point $\mathbf{x}_t^{(0)}$
Reference gradient $\mathbf{g}_t^{(0)}$
For $t_1 = 1, \ldots, T_1$
Reference point $\mathbf{x}_t^{(1)}$
Reference gradient $\mathbf{g}_t^{(1)}$
update
$\mathbf{x}_{t+1} = \mathbf{x}_t - \eta(\mathbf{g}_t^{(0)} + \mathbf{g}_t^{(1)})$
Reference point $\mathbf{x}_t^{(0)}$
Reference gradient $\mathbf{g}_t^{(0)}$
For $t_1 = 1, \ldots, T_1$
Reference point $\mathbf{x}_t^{(1)}$
Reference gradient $\mathbf{g}_t^{(1)}$
For $t_{K-1} = 1, \ldots, T_{K-1}$
Reference point $\mathbf{x}_t^{(K-1)}$
Reference gradient $\mathbf{g}_t^{(K-1)}$
For $t_K = 1, \ldots, T_K$
Reference point $\mathbf{x}_t^{(K)}$
Reference gradient $\mathbf{g}_t^{(K)}$
update
$\mathbf{x}_{t+1} = \mathbf{x}_t - \eta \sum_{i=0}^{K} \mathbf{g}_t^{(i)}$

(a) SVRG (b) SNVRG

Figure 2.2: Illustration of reference points and gradients in SVRG and SNVRG.

$O(Kd)$. In our theory, we will show that $K = O(\log \log n)$. Therefore, the space complexity of our algorithm is actually $\tilde{O}(d)$, which is almost comparable to that of SVRG and SCSG.

Algorithm 2 SNVRG$(\mathbf{z}_0, F, K, M, \{T_l\}, \{B_l\}, B_0, S)$

```
1: Input: initial point $\mathbf{z}_0$; function $F$; loop numbers $K, S$; step size parameter $M$; loop
   parameters $\{T_l\}$; batch parameters $\{B_l\}$; base batch size $B_0$.
2: for $s = 1, \ldots, S$ do
3:     $[\mathbf{y}_s, \mathbf{z}_s] = $ One-epoch-SNVRG$(\mathbf{z}_{s-1}, F, K, M, \{T_l\}, \{B_l\}, B_0)$    ▷ Algorithm 1 with
       Option I
4: end for
5: Output: Uniformly choose $\mathbf{y}_{\text{out}}$ from $\{\mathbf{y}_s\}, 1 \le s \le S$.
```

2.4.1 Convergence of SNVRG

The following theorem shows the gradient complexity for Algorithm 2 to find an ϵ-approximate stationary point with a constant base batch size B_0.

Theorem 2.4.1. *Suppose that F has averaged L-Lipschitz gradient and stochastic gradients with bounded variance σ^2. In Algorithm 2, let $B_0 = n \wedge (2C\sigma^2/\epsilon^2)$ and suppose $B_0 > 4$, $S = 1 \vee (2CL\Delta_F/(B_0^{1/2}\epsilon^2))$ and $C = 6000$. The rest parameters $(K, M, \{B_l\}, \{T_l\})$ are chosen as follows:*

$$\begin{aligned}
&K = \lfloor \log\log B_0 \rfloor, \\
&M = 6L_1, \\
&T_1 = \lfloor B_0^{2^{-K}} \rfloor, \quad T_l = \lfloor B_0^{2^{l-K-2}} \rfloor, \text{ for } 2 \le l \le K, \\
&B_l = 6^{K-l+1}\bigg(\prod_{s=l}^{K} T_s\bigg)^2, \text{ for } 1 \le l \le K.
\end{aligned} \tag{2.4.3}$$

Then the output $\mathbf{y}_{out}$ of Algorithm 2 satisfies $\mathbb{E}[\|\nabla F(\mathbf{y}_{out})\|_2^2] \le \epsilon^2$ with less than

$$O\bigg(\log^3\bigg(\frac{\sigma^2}{\epsilon^2} \wedge n\bigg)\bigg[\frac{\sigma^2}{\epsilon^2} \wedge n + \frac{L\Delta_F}{\epsilon^2}\bigg[\frac{\sigma^2}{\epsilon^2} \wedge n\bigg]^{1/2}\bigg]\bigg) \tag{2.4.4}$$

stochastic gradient computations, where $\Delta_F = F(\mathbf{z}_0) - F^*$.

Remark 2.4.2. *If we treat σ^2, L and Δ_F as constants, and assume $\epsilon \ll 1$, then (2.4.4) can be simplified to $\tilde{O}(\epsilon^{-3} \wedge n^{1/2}\epsilon^{-2})$. This gradient complexity is strictly better than $O(\epsilon^{-10/3} \wedge n^{2/3}\epsilon^{-2})$, which is achieved by SCSG [LJCJ17]. Specifically, when $n \lesssim 1/\epsilon^2$, our proposed SNVRG is faster than SCSG by a factor of $n^{1/6}$; when $n \gtrsim 1/\epsilon^2$, SNVRG is faster than SCSG by a factor of $\epsilon^{-1/3}$. Moreover, SNVRG also outperforms Natasha 2 [AZ18] which attains $\tilde{O}(\epsilon^{-3.25})$ gradient complexity and needs the additional Hessian Lipschitz condition.*

2.5 SNVRG for Finding Local Minima

In this section, we present our algorithm that is built upon One-epoch-SNVRG (Algorithm 1) and Neon2 [AZL18] to find a local minimum in nonconvex optimization faster than existing methods. It is worth noting that to find local minima, we employ a different choice of the number of iteration T which is chosen to be a random variable following a geometric distribution (Algorithm 1 with Option II) rather than fixed. We will show in the next section that these differences are essential in the theoretical analysis of finding local minima.

2.5.1 SNVRG + Neon2: Finding Local Minima

In particular, to solve the finite-sum optimization problem (1.1.1), we propose the SNVRG+ Neon2$^{\text{finite}}$ algorithm to find the local minimum, which is displayed in Algorithm 3. At each iteration of 3, it first determines whether the current point is a first-order stationary point (Line 4) or not. If not, it will run Algorithm 1 (One-epoch-SNVRG) in order to find a first-order stationary point. Once obtaining a first-order stationary point, it will call Neon2$^{\text{finite}}$ to find the negative curvature direction to escape any potential non-degenerate saddle point. According to [XRY18, AZL18], Neon-type algorithms can output such a direction with probability $1-\delta$ for some failure probability $\delta \in (0,1)$. If Neon2$^{\text{finite}}$ does not find such a direction, it will output $\hat{\mathbf{v}} = \perp$ and Algorithm 3 terminates and outputs $\mathbf{z}_{u-1}$ (Line 9) since it has already reached a second-order stationary point according to (1.1.4). If Neon2$^{\text{finite}}$ finds a negative curvature direction $\hat{\mathbf{v}} \neq \perp$, Algorithm 3 will perform one step of negative curvature descent in the direction of $\hat{\mathbf{v}}$ or $-\hat{\mathbf{v}}$ (Line 12) to escape the non-degenerate saddle point. The direction can also be chosen in the same way as in [CDHS17] via comparing the function values at the two resulting points. Here to reduce the computational complexity, we follow [XRY18] and generate a Rademacher random variable to decide the direction, which leads to the same result in expectation. Note that Algorithm 3 is only based on the gradient information of the objective function and therefore belongs to first-order optimization algorithms.

2.5.2 Convergence Analysis of SNVRG + Neon2

In this section, we provide the main theoretical results for finding local minima using SNVRG. The following theorem provides the gradient complexity of Algorithm 3 in finding an approximate local minimum.

Theorem 2.5.1. *Suppose that $F = 1/n \sum_{i=1}^{n} f_i$, where each f_i is L_1-smooth and L_2-Hessian Lipschitz continuous. Let $0 < \epsilon, \epsilon_H < 1$, $\delta = \epsilon_H^3/(144L_2^2\Delta_F)$ and $U = 24L_2^2\Delta_F\epsilon_H^{-3} + 1800L_1\Delta_F\epsilon^{-2}n^{-1/2}$. Set $B_0 = n, M = 6L_1$ and all the rest parameters of One-epoch-SNVRG as in (2.4.3) of Theorem 2.4.1. Choose step size $\eta = \epsilon_H/L_2$. Then with probability at least*

Algorithm 3 SNVRG + Neon2$^{\text{finite}}(\mathbf{z}_0, F, K, M, \{T_l\}, \{B_l\}, B_0, U, \epsilon, \epsilon_H, \delta, \eta, L_1, L_2)$

```
1: Input: initial point $\mathbf{z}_0$; function $F$; loop number $K$; step size parameter $M$; loop parameters $\{T_l\}$; batch parameters $\{B_l\}$; base batch size $B_0$; gradient accuracy $\epsilon$; Hessian accuracy $\epsilon_H$; failure probability $\delta$; negative curvature descent step size $\eta$; gradient Lipschitz parameter $L_1$; Hessian Lipschitz parameter $L_2$.
2: for $u = 1, \ldots, U$ do
3:   $\mathbf{g}_{u-1} = \nabla F(\mathbf{z}_{u-1})$
4:   if $\|\mathbf{g}_{u-1}\|_2 \geq \epsilon$ then
5:     $\mathbf{z}_u$ = One-epoch-SNVRG($\mathbf{z}_{u-1}$,$F, K, M, \{T_l\}, \{B_l\}, B_0$) ▷ Algorithm 1 with Option II
6:   else
7:     $\mathbf{v} = \text{Neon2}^{\text{finite}}(F, \mathbf{z}_{u-1}, L_1, L_2, \delta, \epsilon_H)$
8:     if $\mathbf{v} = \perp$ then
9:       return $\mathbf{z}_{u-1}$
10:    else
11:      Generate a Rademacher random variable $\zeta$
12:      $\mathbf{z}_u \leftarrow \mathbf{z}_{u-1} + \zeta\eta\hat{\mathbf{v}}$
13:    end if
14:  end if
15: end for
16: return
```

1/4, SNVRG + Neon2$^{\text{finite}}$ will find an (ϵ, ϵ_H)-second-order stationary point within

$$\tilde{O}\bigg(\frac{\Delta_F n L_2^2}{\epsilon_H^3} + \frac{\Delta_F n^{3/4} L_1^{1/2} L_2^2}{\epsilon_H^{7/2}} + \frac{\Delta_F n^{1/2} L_1}{\epsilon^2}\bigg) \tag{2.5.1}$$

stochastic gradient evaluations.

Remark 2.5.2. *Note that the gradient complexity in Theorem 2.5.1 holds with constant probability* $1/4$*. In practice, we can repeatedly run Algorithm 3 for* $\log(1/p)$ *times to achieve a result that holds with probability at least* $1-p$ *for any* $p \in (0,1)$*. Similar boosting techniques have also been used in [YZG17, AZL18, YXG18].*

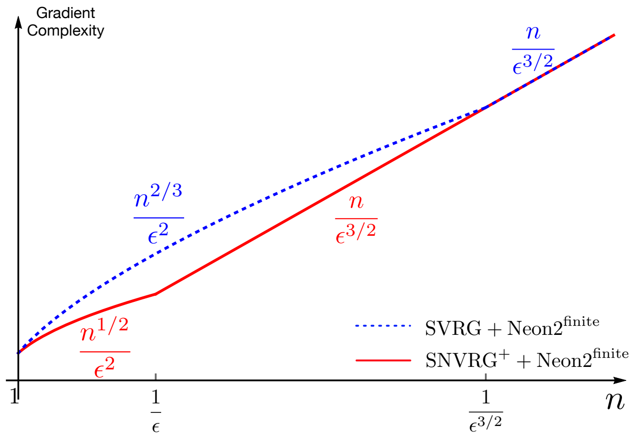


Figure 2.3: Comparison of gradient complexities between SNVRG + Neon2$^{\text{finite}}$ and SVRG + Neon2$^{\text{finite}}$ for finding an $(\epsilon, \sqrt{\epsilon})$-approximate local minimum in finite-sum optimization problems.

Remark 2.5.3. *For finite-sum nonconvex optimization, Theorem 2.5.1 suggests that the gradient complexity of Algorithm 3 (SNVRG + Neon2$^{\text{finite}}$) is* $\tilde{O}(n^{1/2}\epsilon^{-2} + n\epsilon_H^{-3} + n^{3/4}\epsilon_H^{-7/2})$*. In contrast, the gradient complexity of other state-of-the-art local minimum finding algorithms (SVRG + Neon2$^{\text{finite}}$) [AZL18] is* $\tilde{O}(n^{2/3}\epsilon^{-2} + n\epsilon_H^{-3} + n^{3/4}\epsilon_H^{-7/2})$*. Our algorithm is strictly better than that of [AZL18] in terms of the first term in the big O notation.*

If we choose $\epsilon_H = \sqrt{\epsilon}$*, the gradient complexity of our algorithm to find an* $(\epsilon, \sqrt{\epsilon})$*-approximate local minimum turns out to be* $O(n^{1/2}\epsilon^{-2} + n\epsilon^{-3/2} + n^{3/4}\epsilon^{-7/4})$ *and that of SVRG + Neon2$^{\text{finite}}$ is* $O(n^{2/3}\epsilon^{-2} + n\epsilon^{-3/2} + n^{3/4}\epsilon^{-7/4})$*. We compare these two algorithms in Figure 2.3 when* $\epsilon_H = \sqrt{\epsilon}$ *and make the following comments:*

- *When $n \gtrsim \epsilon^{-3/2}$, the gradient complexities of both algorithms are in the same order of $\tilde{O}(n\epsilon^{-3/2})$.*
- *When $\epsilon^{-1} \lesssim n \lesssim \epsilon^{-3/2}$, SNVRG + Neon2$^{\text{finite}}$ enjoys $\tilde{O}(n\epsilon^{-3/2})$ gradient complexity, which is strictly better than that of SVRG + Neon2$^{\text{finite}}$, i.e., $\tilde{O}(n^{2/3}\epsilon^{-2})$.*
- *Lastly, when $n \lesssim \epsilon^{-1}$, SNVRG + Neon2$^{\text{finite}}$ achieves $\tilde{O}(n^{1/2}\epsilon^{-2})$ gradient complexity, which is again better than the gradient complexity of SVRG + Neon2$^{\text{finite}}$, $\tilde{O}(n^{2/3}\epsilon^{-2})$, by a factor of $\tilde{O}(n^{1/6})$.*

In short, our algorithms beats SVRG + Neon2$^{\text{finite}}$ when $n \lesssim \epsilon^{-3/2}$.

2.5.3 Finding Local Minima with the Third-Order Smoothness Condition

As we mentioned before, it has been shown that the third-order smoothness of the objective function F can help accelerate the convergence of nonconvex optimization [CDHS17, YXG18]. For the intuition of the acceleration by third-order smoothness, we refer readers to the detailed exhibition and discussion in [YXG18]. In this section, we will show that our local minimum finding algorithm (Algorithm 3) can find local minima faster provided this additional condition.

Theorem 2.5.4. *Suppose that $F = 1/n\sum_{i=1}^n f_i$, where each f_i is L_1-smooth, L_2-Hessian Lipschitz continuous and F is L_3-third-order smooth. Let $0 < \epsilon, \epsilon_H < 1$, $\delta = \epsilon_H^2/(72L_3\Delta_F)$ and $U = 12L_3\Delta_F\epsilon_H^{-2} + 1800CL_1\Delta_F\epsilon^{-2}n^{-1/2}$. Set $B_0 = n, M = 6L_1$ and all the rest parameters of One-epoch-SNVRG as in (2.4.3). Choose the step size as $\eta = \sqrt{3\epsilon_H/L_3}$. Then with probability at least $1/4$, SNVRG+ Neon2$^{\text{finite}}$ will find an (ϵ, ϵ_H)-second-order stationary point within*

$$\tilde{O}\bigg(\frac{\Delta_F n L_3}{\epsilon_H^2} + \frac{\Delta_F n^{3/4} L_1^{1/2} L_3}{\epsilon_H^{5/2}} + \frac{\Delta_F n^{1/2} L_1}{\epsilon^2}\bigg) \tag{2.5.2}$$

stochastic gradient evaluations.

Similar to previous discussions, we can repeatedly run Algorithm 3 for $\log(1/p)$ times to boost its confidence to $1 - p$ for any $p \in (0, 1)$.

Remark 2.5.5. *Compared with step size $\eta = \epsilon_H/L_2$ used in the negative curvature descent step (Line 12) of Algorithm 3 in Theorem 2.5.1 without third-order smoothness, the step size in Theorem 2.5.4 is chosen to be $\eta = \sqrt{\epsilon_H/L_3}$ where L_3 is the third-order smoothness parameter. Note that when $\epsilon_H \ll 1$, the step size we choose under third-order smoothness assumption is much bigger than that under only second-order smoothness assumption. As is pointed out by [YXG18], the key advantage of third-order smoothness condition is that it enables us to choose a larger step size and therefore achieve much more function value decrease in the negative curvature descent step (Line 12 of Algorithm 3).*

Remark 2.5.6. *Theorem 2.5.4 suggests that the gradient complexity of SNVRG+ Neon2$^{\text{finite}}$ under third-order smoothness is $\tilde{O}(n^{1/2}\epsilon^{-2} + n\epsilon_H^{-2} + n^{3/4}\epsilon_H^{-5/2})$. In stark contrast, the gradient complexity of the state-of-the-art finite-sum local minimum finding algorithm with third-order smoothness assumption (FLASH) [YXG18] is $\tilde{O}(n^{2/3}\epsilon^{-2} + n\epsilon_H^{-2} + n^{3/4}\epsilon_H^{-5/2})$. Clearly, our algorithm is strictly better than the FLASH algorithm [YXG18] in the first term of the gradient complexity.*

Specifically, if we choose $\epsilon_H = \sqrt{\epsilon}$, SNVRG+ Neon2$^{\text{finite}}$ is faster for finding an $(\epsilon, \sqrt{\epsilon})$-approximate local minimum than FLASH by a factor of $O(1/\epsilon^{1/6})$ when $n \lesssim \epsilon^{-2}$. SNVRG+ Neon2$^{\text{finite}}$ is also strictly faster than FLASH when $\epsilon^{-2} \lesssim n \lesssim \epsilon^{-3}$ and will match FLASH when $n \gtrsim \epsilon^{-3}$. We show this comparison in Figure 2.4, which clearly demonstrates that the gradient complexity of SNVRG+ Neon2$^{\text{finite}}$ is much smaller than that of FLASH in a very wide regime.

2.6 Experiments

In this section, we conduct experiments to validate the superiority of the proposed algorithms. In the first part of this section, we compare our algorithm SNVRG with existing baseline algorithms on training a convolutional neural network for image classification. In the second part of this section, we consider a symmetric matrix sensing problem, where many saddle points exist and thus the proposed SNVRG+Neon2 is compared with vanilla SNVRG, SGD+NEON and SCSG+Neon2.

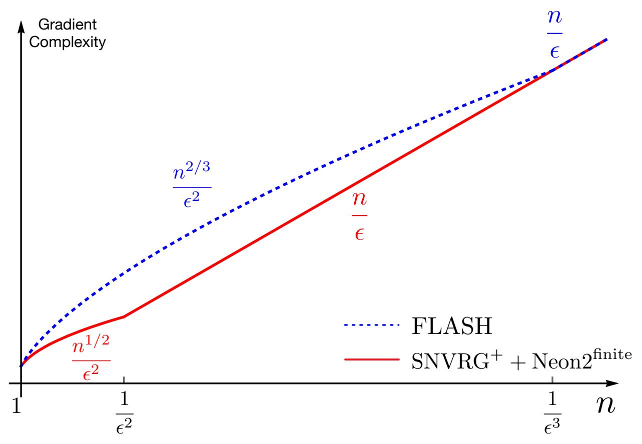


Figure 2.4: Comparison of gradient complexities between SNVRG + Neon2$^{\text{finite}}$ and FLASH for finding an $(\epsilon, \sqrt{\epsilon})$-approximate local minimum in finite-sum nonconvex optimization problems.

2.6.1 SNVRG for Training CNNs for Image Classification

We compare the performance of the following algorithms: SGD; SGD with momentum [Qia99] (denoted by SGD-momentum); ADAM [KB14]; SCSG [LJCJ17]. It is worth noting that SCSG is a special case of SNVRG when the number of nested loops $K = 1$. Due to the memory cost, we did not compare Gradient Descent (GD) or SVRG which need to calculate the full gradient. Although our theoretical analysis holds for general K nested loops, it suffices to choose $K = 2$ in SNVRG to illustrate the effectiveness of the nested structure for the simplification of implementation. In this case, we have 3 reference points and gradients. All experiments are conducted on Amazon AWS p2.xlarge servers which comes with Intel Xeon E5 CPU and NVIDIA Tesla K80 GPU (12G GPU RAM). All algorithm are implemented in Pytorch platform version 0.4.0 within Python 3.6.4.

Datasets We use three image datasets: (1) The MNIST dataset [SS02] consists of handwritten digits and has $50,000$ training examples and $10,000$ test examples. The digits have been size-normalized to fit the network, and each image is 28 pixels by 28 pixels. (2) CIFAR10 dataset [Kri09] consists of images in 10 classes and has $50,000$ training examples and $10,000$ test examples. The digits have been size-normalized to fit the network, and each image is 32 pixels by 32 pixels. (3) SVHN dataset [NWC$^+$11] consists of images of digits and has

$531,131$ training examples and $26,032$ test examples. The digits have been size-normalized to fit the network, and each image is 32 pixels by 32 pixels.

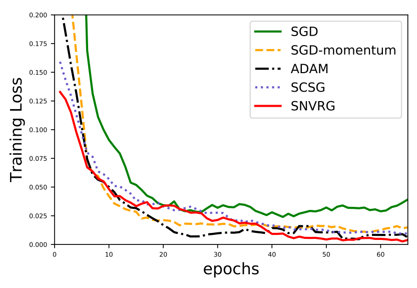
CNN Architecture We use the standard LeNet [LBBH98], which has two convolutional layers with 6 and 16 filters of size 5 respectively, followed by three fully-connected layers with output size 120, 84 and 10. We apply max pooling after each convolutional layer.

Implementation Details & Parameter Tuning We did not use the random data augmentation which is set as default by Pytorch, because it will apply random transformation (e.g., clip and rotation) at the beginning of each epoch on the original image dataset, which will ruin the finite-sum structure of the loss function. We set our grid search rules for all three datasets as follows. For SGD, we search the batch size from $\{256, 512, 1024, 2048\}$ and the initial step sizes from $\{1, 0.1, 0.01\}$. For SGD-momentum, we set the momentum parameter as 0.9. We search its batch size from $\{256, 512, 1024, 2048\}$ and the initial learning rate from $\{1, 0.1, 0.01\}$. For ADAM, we search the batch size from $\{256, 512, 1024, 2048\}$ and the initial learning rate from $\{0.01, 0.001, 0.0001\}$. For SCSG and SNVRG, we choose loop parameters $\{T_l\}$ which satisfy $B_l \cdot \prod_{j=1}^{l} T_j = B$ automatically. In addition, for SCSG, we set the batch sizes $(B, B_1) = (B, B/b)$, where b is the batch size ratio parameter. We search B from $\{256, 512, 1024, 2048\}$ and we search b from $\{2, 4, 8\}$. We search its initial learning rate from $\{1, 0.1, 0.01\}$. For our proposed SNVRG algorithm, we set the batch sizes $(B, B_1, B_2) = (B, B/b, B/b^2)$, where b is the batch size ratio parameter. We search B from $\{256, 512, 1024, 2048\}$ and b from $\{2, 4, 8\}$. We search its initial learning rate from $\{1, 0.1, 0.01\}$.

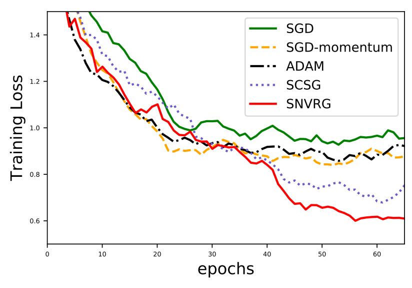
2.6.1.1 Experimental Results with Learning Rate Decaying

In this section, we first present the experimental results with learning rate decay. In particular, following the convention of deep learning practice, we apply learning rate decay schedule to each algorithm with the learning rate decayed by 0.1 every 20 epochs.

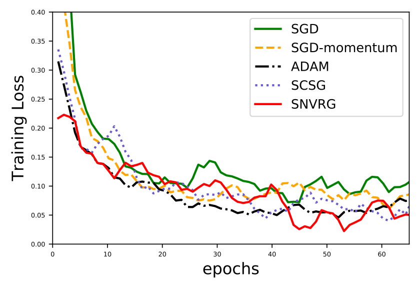
We plotted the training loss and test error for different algorithms on each dataset in Figure 2.5. The results on MNIST are presented in Figures 2.5(a) and 2.5(d); the results on



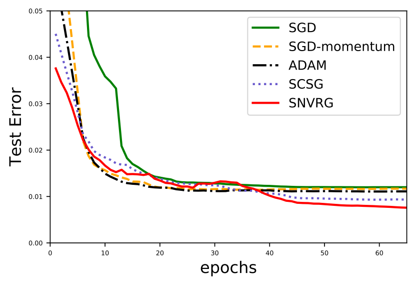
(a) training loss (*MNIST*)



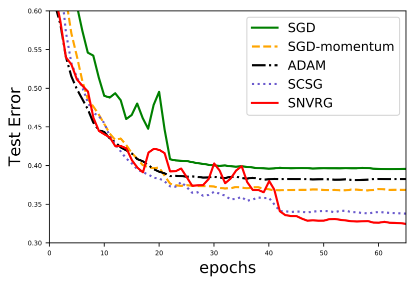
(b) training loss (*CIFAR10*)



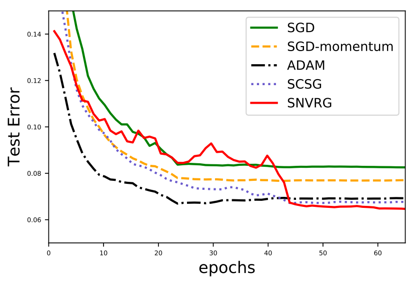
(c) training loss (*SVHN*)



(d) test error (*MNIST*)



(e) test error (*CIFAR10*)



(f) test error (*SVHN*)

Figure 2.5: Experiment results on different datasets with learning rate decay. (a) and (d) depict the training loss and test error (top-1 error) v.s. data epochs for training LeNet on MNIST dataset. (b) and (e) depict the training loss and test error v.s. data epochs for training LeNet on CIFAR10 dataset. (c) and (f) depict the training loss and test error v.s. data epochs for training LeNet on SVHN dataset.

CIFAR10 are in Figures 2.5(b) and 2.5(e); and the results on SVHN dataset are shown in Figures 2.5(c) and 2.5(f). It can be seen that with learning rate decay schedule, our algorithm SNVRG outperforms all baseline algorithms, which confirms that the use of nested reference points and gradients can accelerate the nonconvex finite-sum optimization.

We would like to emphasize that, while this experiment is on training convolutional neural networks, the major goal of this experiment is to illustrate the advantage of our algorithm and corroborate our theory, rather than claiming a state-of-the-art algorithm for training deep neural networks.

2.6.1.2 Experimental Results without Learning Rate Decay

We also conducted experiments comparing different algorithms without the learning rate decay schedule. The parameters are tuned by the same grid search described in Section 2.6. In particular, we summarize the parameters of different algorithms used in our experiments with and without learning rate decay for MNIST in Table 2.3, CIFAR10 in Table 2.4, and SVHN in Table 2.5. We plotted the training loss and test error for each dataset without learning rate decay in Figure 2.6. The results on MNIST are presented in Figures 2.6(a) and 2.6(d); the results on CIFAR10 are in Figures 2.6(b) and 2.6(e); and the results on SVHN dataset are shown in Figures 2.6(c) and 2.6(f). It can be seen that without learning decay, our algorithm SNVRG still outperforms all the baseline algorithms except for the training loss on SVHN dataset. However, SNVRG still performs the best in terms of test error on SVHN dataset. These results again suggest that SNVRG can beat the state-of-the-art in practice, which backups our theory.

Table 2.3: Parameter settings of all algorithms on MNIST dataset.

Algorithm	With Learning Rate Decay			Without Learning Rate Decay		
	Initial learning rate η	Batch size B	Batch size ratio b	learning rate η	Batch size B	Batch size ratio b
SGD	0.1	1024	N/A	0.01	1024	N/A
SGD-momentum	0.01	1024	N/A	0.1	1024	N/A
ADAM	0.001	1024	N/A	0.001	1024	N/A
SCSG	0.01	512	8	0.01	512	8
SNVRG	0.01	512	8	0.01	512	8

2.6.2 Experimental Results for Escaping Saddle Points

In this section, we conduct experiments to validate the superiority of our proposed algorithms for escaping from saddle points. We consider the matrix sensing problem, which is defined

Table 2.4: Parameter settings of all algorithms on CIFAR10 dataset.

Algorithm	With Learning Rate Decay			Without Learning Rate Decay		
	Initial learning rate η	Batch size B	Batch size ratio b	learning rate η	Batch size B	Batch size ratio b
SGD	0.1	1024	N/A	0.01	512	N/A
SGD-momentum	0.01	1024	N/A	0.01	2048	N/A
ADAM	0.001	1024	N/A	0.001	2048	N/A
SCSG	0.01	512	8	0.01	512	8
SNVRG	0.01	1024	8	0.01	512	4

as follows:

$$\min_{\mathbf{U}\in\mathbb{R}^{d\times r}} f(\mathbf{U}) = \frac{1}{2n}\sum_{i=1}^{n}(\langle \mathbf{A}_i, \mathbf{U}\mathbf{U}^\top\rangle - b_i)^2, \tag{2.6.1}$$

where $\{\mathbf{A}_i\}_{i=1}^n$ are sensing matrices, $b_i = \langle \mathbf{A}_i, \mathbf{M}^*\rangle$ is the i-th observation, and $\mathbf{M}^* = \mathbf{U}^*(\mathbf{U}^*)^\top$ is the underlying unknown low-rank matrix. Following the same setting in [YXG18], we consider two matrix sensing problems: (1) $d = 50, r = 3$ and (2) $d = 100, r = 3$. We generate $n = 20d$ sensing matrices $\{\mathbf{A}_i\}_{i=1}^n$, where each entry of $\mathbf{A}_i$ follows the standard normal distribution. We generate $\mathbf{U}^*$ randomly where each row of $\mathbf{U}^*$ follows the standard normal distribution. We generate $\mathbf{u}_0$ from standard normal distribution and set the initial point as $\mathbf{U}_0 = [\mathbf{u}_0, \mathbf{0}, \ldots, \mathbf{0}]$.

We compare our algorithm SNVRG+Neon with following baselines for nonconvex optimization problems: SNVRG, noisy stochastic gradient descent (NSGD) [GHJY15], and Stochastically Controlled Stochastic Gradient with Neon (SCSG-Neon) [XRY18, AZL18]. For the simplicity, we choose the gradient batch size to be 100 for all algorithms. For SCSG-Neon, we set the outer batch size to be n. For SNVRG and SNVRG+Neon, we choose $K = 2$ and set $(B, B_1) = (n, n/5)$. We apply Oja's algorithm [Oja82] to calculate the negative curvature with a Hessian mini-batch size of 100. We perform a grid search over step sizes for all algorithms. We report the objective function value versus CPU running time.

Table 2.5: Parameter settings of all algorithms on SVHN dataset.

Algorithm	With Learning Rate Decay			Without Learning Rate Decay		
	Initial learning rate η	Batch size B	Batch size ratio b	learning rate η	Batch size B	Batch size ratio b
SGD	0.1	2048	N/A	0.01	1024	N/A
SGD-momentum	0.01	2048	N/A	0.01	2048	N/A
ADAM	0.001	1024	N/A	0.001	512	N/A
SCSG	0.01	512	4	0.1	1024	4
SNVRG	0.01	512	8	0.01	512	4

The experimental results are shown in Figures 2.7(a) and 2.7(b). From the figures we can see that without adding additional noise or using negative curvature information, SNVRG tends to get stuck in saddle points. In sharp contrast, NSGD, SCSG-Neon and SNVRG-Neon are able to escape from saddle points. We also notice that SNVRG-Neon outperforms all other baseline algorithms in both problem settings.

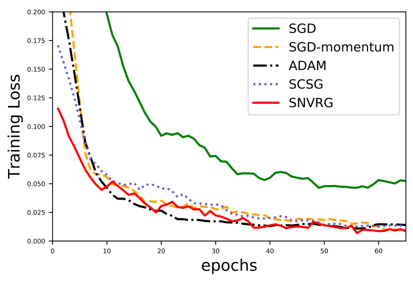
2.7 Proof of Main Theory

In this section, we provide the proofs of our theoretical analysis in omitted in previous sections.

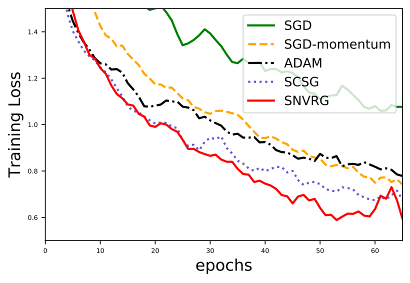
2.7.1 Proof of Main Theory for Finding Stationary Points

We start with the following supporting lemma that characterizes the function value decrease of One-epoch-SNVRG (Algorithm 1).

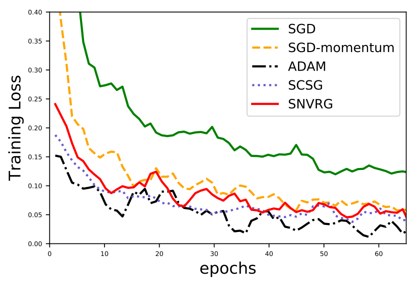
Lemma 2.7.1. *Suppose that F has averaged L-Lipschitz gradient. Suppose that $B_0 \geq 4$ and the rest parameters $(K, M, \{B_l\}, \{T_l\})$ of Algorithm 1 are chosen the same as in* (2.4.3).



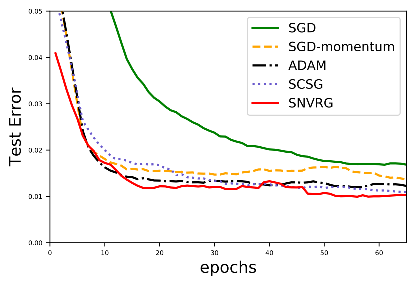
(a) training loss (*MNIST*)



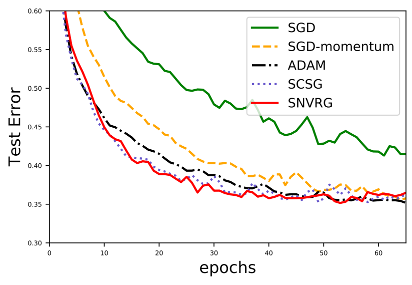
(b) training loss (*CIFAR10*)



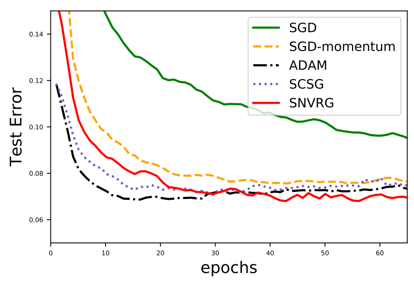
(c) training loss (*SVHN*)



(d) test error (*MNIST*)



(e) test error (*CIFAR10*)



(f) test error (*SVHN*)

Figure 2.6: Experimental results on different datasets without learning rate decay. (a) and (d) depict the training loss and test error (top-1 error) v.s. data epochs for training LeNet on MNIST dataset. (b) and (e) depict the training loss and test error v.s. data epochs for training LeNet on CIFAR10 dataset. (c) and (f) depict the training loss and test error v.s. data epochs for training LeNet on SVHN dataset.

Then Algorithm 1 with Option I satisfies

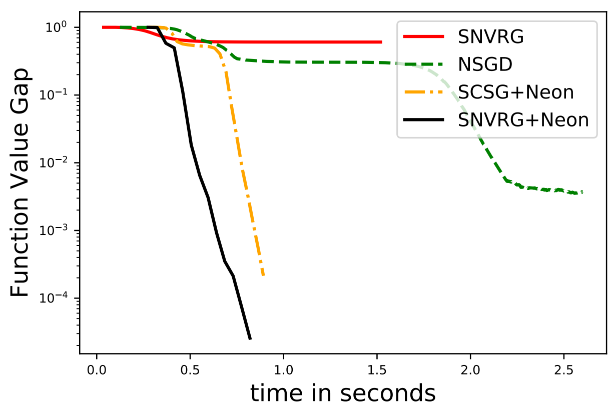
$$\mathbb{E}\|\nabla F(\mathbf{x}_{out})\|_2^2 \le C\bigg(\frac{L}{B_0^{1/2}}\cdot\mathbb{E}\big[F(\mathbf{x}_0)-F(\mathbf{x}_T)\big]+\frac{\sigma^2}{B_0}\cdot\mathbb{1}(B_0<n)\bigg) \tag{2.7.1}$$

within $1 \vee (10B_0\log^3 B_0)$ *stochastic gradient computations, where* $T=\prod_{l=1}^K T_l$, $C=6000$ *is a constant and* $\mathbb{1}(\cdot)$ *is the indicator function.*

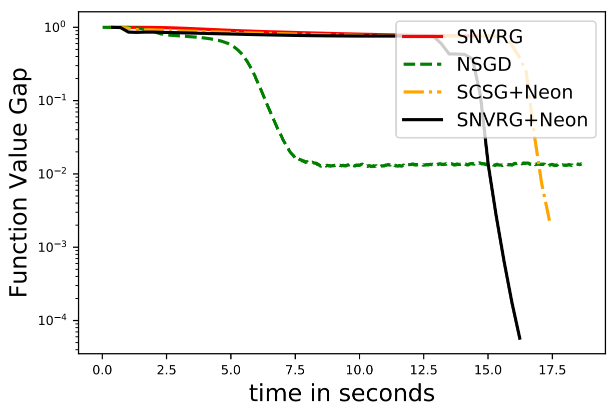
Now we prove our main theorem which spells out the gradient complexity of SNVRG.

Proof of Theorem 2.4.1. By (2.7.1) we have

$$\mathbb{E}\|\nabla F(\mathbf{y}_s)\|_2^2 \le C\bigg(\frac{L}{B_0^{1/2}}\cdot\mathbb{E}\big[F(\mathbf{z}_{s-1})-F(\mathbf{z}_s)\big]+\frac{\sigma^2}{B_0}\cdot\mathbb{1}(B_0<n)\bigg), \tag{2.7.2}$$



(a) $d = 50, r = 3$



(b) $d = 100, r = 3$

Figure 2.7: Experimental results on matrix sensing problems. (a) depicts matrix sensing problem with $d = 50, r = 3$. (b) depicts matrix sensing problem with $d = 100, r = 3$.

where $C = 6000$. Taking summation for (2.7.2) over s from 1 to S, we have

$$\sum_{s=1}^{S} \mathbb{E}\|\nabla F(\mathbf{y}_s)\|_2^2 \le C\bigg(\frac{L}{B_0^{1/2}} \cdot \mathbb{E}\big[F(\mathbf{z}_0) - F(\mathbf{z}_S)\big] + \frac{\sigma^2}{B_0} \cdot \mathbb{1}(B_0 < n) \cdot S\bigg). \tag{2.7.3}$$

Dividing both sides of (2.7.3) by S, we immediately obtain

$$\mathbb{E}\|\nabla F(\mathbf{y}_{\text{out}})\|_2^2 \le C\bigg(\frac{L\mathbb{E}\big[F(\mathbf{z}_0) - F^*\big]}{SB_0^{1/2}} + \frac{\sigma^2}{B_0} \cdot \mathbb{1}(B_0 < n)\bigg) \tag{2.7.4}$$

$$= C\bigg(\frac{L\Delta_F}{SB_0^{1/2}} + \frac{\sigma^2}{B_0} \cdot \mathbb{1}(B_0 < n)\bigg), \tag{2.7.5}$$

where (2.7.4) holds because $F(\mathbf{z}_S) \ge F^*$ and by the definition $\Delta_F = F(\mathbf{z}_0) - F^*$. By the choice of parameters in Theorem 2.4.1, we have $B_0 = n \wedge (2C\sigma^2/\epsilon^2), S = 1 \vee (2CL\Delta_F/(B_0^{1/2}\epsilon^2))$, which implies

$$\mathbb{1}(B_0 < n) \cdot \sigma^2/B_0 \le \epsilon^2/(2C), \quad \text{and} \quad L\Delta_F/(SB_0^{1/2}) \le \epsilon^2/(2C). \tag{2.7.6}$$

Submitting (2.7.6) into (2.7.5), we have $\mathbb{E}\|\nabla F(\mathbf{y}_{\text{out}})\|_2^2 \le 2C\epsilon^2/(2C) = \epsilon^2$. By Lemma 2.7.1, we have that each One-epoch-SNVRG takes less than $7B_0 \log^3 B_0$ stochastic gradient computations. Since we have total S epochs, so the total gradient complexity of Algorithm 2 is less than

$$S \cdot 7B_0 \log^3 B_0 \le 7B_0 \log^3 B_0 + \frac{L\Delta_F}{\epsilon^2} \cdot 7B_0^{1/2} \log^3 B_0$$

$$= O\bigg(\log^3 \left(\frac{\sigma^2}{\epsilon^2} \wedge n \right) \left[\frac{\sigma^2}{\epsilon^2} \wedge n + \frac{L\Delta_F}{\epsilon^2} \left[\frac{\sigma^2}{\epsilon^2} \wedge n \right]^{1/2} \right] \bigg),$$

which leads to the conclusion. □

2.7.2 Proof of Main Theory for Finding Local Minima

In this section, we prove the gradient complexity of SNVRG + Neon2$^{\text{finite}}$. It is worth noting that in order to find local minima we apply One-epoch-SNVRG with Option II which samples the total number of epochs T from a geometric distribution. Similar to the analysis for finding first-order stationary points, we also have the following supporting lemma about the function value decrease of Algorithm 1.

Lemma 2.7.2. *Suppose that each f_i is L_1-smooth and F has σ^2-sub-Gaussian stochastic gradient. In Algorithm 1, suppose that $B_0 \geq 4$ and the rest parameters $(K, M, \{B_l\}, \{T_l\})$ of Algorithm 1 are chosen the same as in* (2.4.3)*. Then Algorithm 1 with Option II satisfies*

$$\mathbb{E}\|\nabla F(\mathbf{x}_T)\|_2^2 \leq C\bigg(\frac{M}{B_0^{1/2}} \cdot \mathbb{E}[F(\mathbf{x}_0) - F(\mathbf{x}_T)] + \frac{2\sigma^2}{B_0} \cdot \mathbb{1}\{B_0 < n\} \bigg), \tag{2.7.7}$$

where $C = 1000$. In addition, the total number of stochastic gradient computations $\mathcal{T}$ by Algorithm 1 satisfies $\mathbb{E}\mathcal{T} \leq 10 B_0 \log^3 B_0$.

Remark 2.7.3. *Note that Lemma 2.7.1 is regarding $\mathbf{x}_{out}$, which is a uniformly chosen iterate from $\mathbf{x}_1, \ldots, \mathbf{x}_T$. In contrast, Lemma 2.7.2 is regarding the last iterate of $\mathbf{x}_T$ in Algorithm 1. This difference leads to the nonergodic-type and ergodic-type guarantees of One-epoch-SNVRG which plays different roles in the analysis of stationary point finding algorithms and local minimum finding algorithms.*

Remark 2.7.4. *For simplicity, we use $\nabla f_i(\mathbf{x})$ to denote the stochastic gradient at point $\mathbf{x}$ in our One-epoch-SNVRG algorithm (Lines 20 and 23 in Algorithm 1) and the analysis of Lemma 2.7.2. However, we emphasize that One-epoch-SNVRG also works in the general stochastic optimization setting if we replace $\nabla f_i(\mathbf{x})$ with $\nabla F(\mathbf{x}; \xi_i)$ for any index i. And the theoretical result in Lemma 2.7.2 still holds.*

When $F(\mathbf{x})$ has the finite-sum structure in (1.1.1), we choose $B_0 = n, M = 6L_1$ in One-epoch-SNVRG. Lemma 2.7.2 straightforwardly implies the following corollary.

Corollary 2.7.5. *Suppose that each f_i is L_1-smooth. We choose $B_0 = n$, and let other parameters be chose as in Lemma 2.7.2. Then the output of Algorithm 1 with Option II satisfies*

$$\mathbb{E}\|\nabla F(\mathbf{x}_T)\|_2^2 \leq \frac{CL_1}{n^{1/2}} \cdot \mathbb{E}\big[F(\mathbf{x}_0) - F(\mathbf{x}_T)\big],$$

where $C = 6000$. Let $\mathcal{T}$ be the total amount of stochastic gradient computations of Algorithm 1, then we have $\mathbb{E}\mathcal{T} \leq 10n\log^3 n$.

The following lemma shows that based on Neon2$^{\text{finite}}$ the negative curvature descent step of Algorithm 3 (Line 12) enjoys sufficient function value decrease. The proof can be found in Theorem 5 and Claim C.2 in [AZL18].

Lemma 2.7.6 ([AZL18])**.** *Suppose $F = 1/n\sum_{i=1}^n f_i$, each f_i is L_1-smooth and L_2-Hessian Lipschitz continuous. Let $\epsilon_H \in (0,1)$ and set $\eta = \epsilon_H/L_2$. Assume $\lambda_{min}(\nabla^2 F(\mathbf{z}_{u-1})) < -\epsilon_H$ and that at the u-th iteration Algorithm 3 executes the Neon2^{finite} algorithm (Line 7). Then with probability $1-\delta$ it holds that*

$$\mathbb{E}_\zeta\big[F(\mathbf{z}_u) - F(\mathbf{z}_{u-1})\big] \leq -\epsilon_H^3/(12L_2^2).$$

In addition, Neon2^{finite} takes $O\big((n + n^{3/4}\sqrt{L_1/\epsilon_H})\log^2(d/\delta)\big)$ stochastic gradient computations.

Proof of Theorem 2.5.1. Let $\mathcal{I} = \{1, \dots, U\}$ be the index set of all iterations. We denote $\mathcal{I}_1$ and $\mathcal{I}_2$ as the index sets such that $\mathbf{z}_u$ is obtained from Neon2$^{\text{finite}}$ for all $u \in \mathcal{I}_1$ and $\mathbf{z}_{u'}$ is the output by SNVRG for all $u' \in \mathcal{I}_2$. Obviously we have $U = |\mathcal{I}_1| + |\mathcal{I}_2|$. We will calculate $|\mathcal{I}_1|, |\mathcal{I}_2|$ separately. For $|\mathcal{I}_1|$, by Lemma 2.7.6, with probability $1-\delta$, we have

$$\mathbb{E}\big[F(\mathbf{z}_u) - F(\mathbf{z}_{u-1})\big] \leq -\epsilon_H^3/(12L_2^2), \qquad \text{for } u \in \mathcal{I}_1. \tag{2.7.8}$$

Summing up (2.7.8) over $u \in \mathcal{I}_1$, then with probability $1 - \delta \cdot |\mathcal{I}_1|$ we have

$$|\mathcal{I}_1| \cdot \epsilon_H^3/(12L_2^2) \leq \sum_{u\in\mathcal{I}_1} \mathbb{E}\big[F(\mathbf{z}_{u-1}) - F(\mathbf{z}_u)\big] \leq \sum_{u\in\mathcal{I}} \mathbb{E}\big[F(\mathbf{z}_{u-1}) - F(\mathbf{z}_u)\big] \leq \Delta_F, \tag{2.7.9}$$

where the second inequality holds because by Corollary 2.7.5 it holds that

$$0 \le \mathbb{E}\|\nabla F(\mathbf{z}_u)\|_2^2 \le \frac{CL_1}{n^{1/2}}\mathbb{E}\big[F(\mathbf{z}_{u-1}) - F(\mathbf{z}_u)\big], \qquad \text{for all } u \in \mathcal{I}_2. \tag{2.7.10}$$

By (2.7.9), we have

$$|\mathcal{I}_1| \le 12L_2^2\Delta_F/\epsilon_H^3.$$

To calculate $|\mathcal{I}_2|$, we further decompose $\mathcal{I}_2$ into two disjoint sets such that $\mathcal{I}_2 = \mathcal{I}_2^1 \cup \mathcal{I}_2^2$, where $\mathcal{I}_2^1 = \{u \in \mathcal{I}_2 : \|\mathbf{g}_u\|_2 > \epsilon\}$, $\mathcal{I}_2^2 = \{u \in \mathcal{I}_2 : \|\mathbf{g}_u\|_2 \le \epsilon\}$. It is worth noting that if $u \in \mathcal{I}_2^2$ such that $\|\mathbf{g}_u\|_2 \le \epsilon$, then Algorithm 3 will execute Neon2$^{\text{finite}}$ and a negative curvature descent step, which means $u+1 \in \mathcal{I}_1$ by definition. Thus, it always holds that $|\mathcal{I}_2^2| \le |\mathcal{I}_1|$. For $|\mathcal{I}_2^1|$, note that $\mathbf{x}_0 = \mathbf{z}_{u-1}$ and $\mathbf{x}_T = \mathbf{z}_u$ in Corollary 2.7.5, which directly implies

$$\begin{aligned}\sum_{u\in\mathcal{I}_2^1} \mathbb{E}\|\nabla F(\mathbf{z}_u)\|_2^2 &\le \sum_{u\in\mathcal{I}_2^1} \frac{CL_1}{n^{1/2}}\mathbb{E}\big[F(\mathbf{z}_{u-1}) - F(\mathbf{z}_u)\big] \\ &\le \sum_{u\in\mathcal{I}} \frac{CL_1}{n^{1/2}}\mathbb{E}\big[F(\mathbf{z}_{u-1}) - F(\mathbf{z}_u)\big] \\ &\le \frac{CL_1}{n^{1/2}} \cdot \Delta_F,\end{aligned} \tag{2.7.11}$$

where the second inequality holds because $\mathbb{E}[F(\mathbf{z}_{u-1}) - F(\mathbf{z}_u)] \ge 0$ for $u \in \mathcal{I}_1 \cup \mathcal{I}_2$ by (2.7.8) and (2.7.10). Applying Markov's inequality, with probability at least 2/3, we have

$$\sum_{u\in\mathcal{I}_2^1} \|\nabla F(\mathbf{z}_u)\|_2^2 \le \frac{3CL_1\Delta_F}{n^{1/2}}.$$

Since for any $u \in \mathcal{I}_2^1$, we have $\|\nabla F(\mathbf{z}_u)\|_2 = \|\mathbf{g}_u\|_2 > \epsilon$, with probability at least 2/3 it holds that

$$|\mathcal{I}_2^1| \le \frac{3CL_1\Delta_F}{\epsilon^2 n^{1/2}}.$$

Thus, the total number of iterations is $U = |\mathcal{I}_1| + |\mathcal{I}_2| \le 2|\mathcal{I}_1| + |\mathcal{I}_2^1| \le 24L_2^2\Delta_F\epsilon_H^{-3} + 3CL_1\Delta_F\epsilon^{-2}n^{-1/2}$.

We now calculate the gradient complexity of Algorithm 3. By Corollary 2.7.5 one single call of One-epoch-SNVRG needs at most $20n\log^3 n$ stochastic gradient computations and by

Lemma 2.7.6 one single call of Neon2$^{\text{finite}}$ needs $O\big((n + n^{3/4}\sqrt{L_1/\epsilon_H})\log^2(d/\delta)\big)$ stochastic gradient computations. In addition, we need to compute $\mathbf{g}_u$ at each iteration of Algorithm 3 (Line 3), which takes $O(n)$ stochastic gradient computations. Thus, the expectation of the total amount of stochastic gradient computations, denoted by $\mathbb{E}T_{\text{total}}$, can be upper bounded by

$$\begin{aligned}
&|\mathcal{I}_1| \cdot O\big((n + n^{3/4}\sqrt{L_1/\epsilon_H})\log^2(d/\delta)\big) + |\mathcal{I}_2| \cdot O(n\log^3 n) + |\mathcal{I}| \cdot O(n)\\
&= |\mathcal{I}_1| \cdot \tilde{O}\big(n + n^{3/4}\sqrt{L_1/\epsilon_H}\big) + (|\mathcal{I}_2^1| + |\mathcal{I}_2^2|) \cdot \tilde{O}(n)\\
&= |\mathcal{I}_1| \cdot \tilde{O}\big(n + n^{3/4}\sqrt{L_1/\epsilon_H}\big) + (|\mathcal{I}_2^1| + |\mathcal{I}_1|) \cdot \tilde{O}(n).
\end{aligned} \tag{2.7.12}$$

We further plug the upper bound for $|\mathcal{I}_1|$ and $|\mathcal{I}_2^1|$ into (2.7.12) and obtain

$$\begin{aligned}
\mathbb{E}T_{\text{total}} &= O(L_2^2\Delta_F\epsilon_H^{-3}) \cdot \tilde{O}\big(n + n^{3/4}\sqrt{L_1/\epsilon_H}\big) + O(L_1\Delta_F\epsilon^{-2}n^{-1/2})\tilde{O}(n)\\
&= \tilde{O}\big(\Delta_F n L_2^2\epsilon_H^{-3} + \Delta_F n^{3/4}L_1^{1/2}L_2^2\epsilon_H^{-7/2} + \Delta_F n^{1/2}L_1\epsilon^{-2}\big).
\end{aligned}$$

Finally, applying Markov inequality, with probability 2/3, it holds that

$$T_{\text{total}} = \tilde{O}\big(\Delta_F n L_2^2\epsilon_H^{-3} + \Delta_F n^{3/4}L_1^{1/2}L_2^2\epsilon_H^{-7/2} + \Delta_F n^{1/2}L_1\epsilon^{-2}\big).$$

Since $|\mathcal{I}_1|\delta = |\mathcal{I}_1|/(144 \cdot L_2^2\Delta_F\epsilon_H^{-3}) \le 1/12$, then by the union bound, with probability $1 - 1/3 - 1/3 - |\mathcal{I}_1|\delta \ge 1/4$, SNVRG + Neon2$^{\text{finite}}$ will find an (ϵ, ϵ_H)-second order stationary point within

$$\tilde{O}\big(\Delta_F n L_2^2\epsilon_H^{-3} + \Delta_F n^{3/4}L_1^{1/2}L_2^2\epsilon_H^{-7/2} + \Delta_F n^{1/2}L_1\epsilon^{-2}\big)$$

stochastic gradient computations. $\square$

2.7.3 Proof of Main Theory with Third-order Smoothness

In this section, we prove the theoretical results of our proposed algorithms under third-order smoothness condition. The following lemma shows that the negative curvature descent step (Line 12) of Algorithm 3 achieves more function value decrease under third-order smoothness assumption. The proof can be found in Lemma 4.3 of [YXG18].

Lemma 2.7.7 ([YXG18]). *Suppose that $F = 1/n\sum_{i=1}^n f_i$, each f_i is L_1-smooth, L_2-Hessian Lipschitz continuous and F is L_3-third-order smooth. Let $\epsilon_H \in (0,1)$ and $\eta = \sqrt{3\epsilon_H/L_3}$. Suppose that $\lambda_{min}(\nabla^2 F(\mathbf{z}_{u-1})) < -\epsilon_H$ and that at the u-th iteration Algorithm 3 executes the Neon2^{finite} algorithm (Line 7). Then with probability $1-\delta$ it holds that*

$$\mathbb{E}_\zeta\big[F(\mathbf{z}_u) - F(\mathbf{z}_{u-1})\big] \le -\epsilon_H^2/(6L_3).$$

In addition, Neon2^{finite} takes $O\big((n + n^{3/4}\sqrt{L_1/\epsilon_H})\log^2(d/\delta)\big)$ stochastic gradient computations.

Proof of Theorem 2.5.4. Denote $\mathcal{I} = \{1,\dots,U\}$ as the index of iteration. Let $\mathcal{I} = \{1,\dots,U\}$ be the index set of iteration. We use $\mathcal{I}_1$ and $\mathcal{I}_2$ to represent the index set of iterates where the $\mathbf{z}_u$ is obtained from Neon2$^{\text{finite}}$ and One-epoch-SNVRG. Since $U = |\mathcal{I}_1| + |\mathcal{I}_2|$, we calculate $|\mathcal{I}_1|, |\mathcal{I}_2|$ separately. For $|\mathcal{I}_1|$, by Lemma 2.7.7, with probability at least $1-\delta$, we have

$$\mathbb{E}\big[F(\mathbf{z}_u) - F(\mathbf{z}_{u-1})\big] \le -\epsilon_H^2/(6L_3), \qquad \text{for } u \in \mathcal{I}_1. \tag{2.7.13}$$

Summing up (2.7.13) over $u \in \mathcal{I}_1$ and applying union bound, then with probability at least $1 - \delta \cdot |\mathcal{I}_1|$ we have

$$|\mathcal{I}_1| \cdot \epsilon_H^2/(6L_3) \le \sum_{u\in\mathcal{I}_1}\mathbb{E}\big[F(\mathbf{z}_{u-1}) - F(\mathbf{z}_u)\big] \le \sum_{u\in\mathcal{I}}\mathbb{E}\big[F(\mathbf{z}_{u-1}) - F(\mathbf{z}_u)\big] \le \Delta_F, \tag{2.7.14}$$

where the second inequality holds due to the fact that by Corollary 2.7.5 we have

$$0 \le \mathbb{E}\|\nabla F(\mathbf{z}_u)\|_2^2 \le \frac{CL_1}{n^{1/2}}\mathbb{E}\big[F(\mathbf{z}_{u-1}) - F(\mathbf{z}_u)\big], \qquad \text{for } u \in \mathcal{I}_2. \tag{2.7.15}$$

(2.7.14) directly implies

$$|\mathcal{I}_1| \le 6L_3\Delta_F/\epsilon_H^2.$$

For $|\mathcal{I}_2|$, we decompose $\mathcal{I}_2 = \mathcal{I}_2^1 \cup \mathcal{I}_2^2$, where $\mathcal{I}_2^1 = \{u \in \mathcal{I}_2 : \|\mathbf{g}_u\|_2 > \epsilon\}$ and $\mathcal{I}_2^2 = \{u \in \mathcal{I}_2 : \|\mathbf{g}_u\|_2 \le \epsilon\}$. If $u \in \mathcal{I}_2^2$, then at the $(u+1)$-th iteration, Algorithm 3 will execute Neon2$^{\text{finite}}$. Thus, we have $|\mathcal{I}_2^2| \le |\mathcal{I}_1|$. For $|\mathcal{I}_2^1|$, note that $\mathbf{x}_0 = \mathbf{z}_{u-1}$ and $\mathbf{x}_T = \mathbf{z}_u$ in Corollary 2.7.5 and summing up over $u \in \mathcal{I}_2^1$ yields

$$\sum_{u\in\mathcal{I}_2^1}\mathbb{E}\|\nabla F(\mathbf{z}_u)\|_2^2 \le \sum_{u\in\mathcal{I}_2^1}\frac{CL_1}{n^{1/2}}\mathbb{E}\big[F(\mathbf{z}_{u-1}) - F(\mathbf{z}_u)\big]$$

$$
\begin{aligned}
&\le \sum_{u\in\mathcal{I}} \frac{CL_1}{n^{1/2}}\mathbb{E}\big[F(\mathbf{z}_{u-1}) - F(\mathbf{z}_u)\big]\\
&\le \frac{CL_1}{n^{1/2}}\cdot\Delta_F, \qquad (2.7.16)
\end{aligned}
$$

where the second inequality follows from (2.7.14) and (2.7.15). Applying Markov's inequality, with probability at least 2/3, we have

$$\sum_{u\in\mathcal{I}_2^1} \|\nabla F(\mathbf{z}_u)\|_2^2 \le \frac{3CL_1\Delta_F}{n^{1/2}}.$$

by definition for any $u\in\mathcal{I}_2^1$, we have $\|\nabla F(\mathbf{z}_u)\|_2 = \|\mathbf{g}_u\|_2 > \epsilon$. Then we have with probability at least 2/3 that

$$|\mathcal{I}_2^1| \le \frac{3CL_1\Delta_F}{\epsilon^2 n^{1/2}}.$$

Total number of iteration is $U = |\mathcal{I}_1| + |\mathcal{I}_2| \le 2|\mathcal{I}_1| + |\mathcal{I}_2^1| \le 12L_3\Delta_F\epsilon_H^{-2} + 3CL_1\Delta_F\epsilon^{-2}n^{-1/2}$. We now calculate the gradient complexity of Algorithm 3. By Lemma 2.7.7 one single call of Neon2$^{\text{finite}}$ needs $O\big((n + n^{3/4}\sqrt{L_1/\epsilon_H})\log^2(d/\delta)\big)$ stochastic gradient computations and by Corollary 2.7.5 one single call of One-epoch-SNVRG needs $20n\log^3 n$ stochastic gradient computations. Moreover, we need to compute $\mathbf{g}_u$ at each iteration, which takes $O(n)$ stochastic gradient computations. Thus, the expectation of the total amount of stochastic gradient computations $\mathbb{E}T_{\text{total}}$ can be bounded by

$$
\begin{aligned}
&|\mathcal{I}_1|\cdot O\big((n + n^{3/4}\sqrt{L_1/\epsilon_H})\log^2(d/\delta)\big) + |\mathcal{I}_2|\cdot O(n\log^3 n) + |\mathcal{I}|\cdot O(n)\\
&= |\mathcal{I}_1|\cdot \tilde{O}\big(n + n^{3/4}\sqrt{L_1/\epsilon_H}\big) + (|\mathcal{I}_2^1| + |\mathcal{I}_2^2|)\cdot\tilde{O}(n)\\
&= |\mathcal{I}_1|\cdot \tilde{O}\big(n + n^{3/4}\sqrt{L_1/\epsilon_H}\big) + (|\mathcal{I}_2^1| + |\mathcal{I}_1|)\cdot\tilde{O}(n). \qquad (2.7.17)
\end{aligned}
$$

We further plug the upper bound of $|\mathcal{I}_1|$ and $|\mathcal{I}_2^1|$ into (2.7.17) and obtain

$$
\begin{aligned}
\mathbb{E}T_{\text{total}} &= O(L_3\Delta_F\epsilon_H^{-2})\cdot\tilde{O}\big(n + n^{3/4}\sqrt{L_1/\epsilon_H}\big) + O(L_1\Delta_F\epsilon^{-2}n^{-1/2})\tilde{O}(n)\\
&= \tilde{O}\big(\Delta_F nL_3\epsilon_H^{-2} + \Delta_F n^{3/4}L_1^{1/2}L_3\epsilon_H^{-5/2} + \Delta_F n^{1/2}L_1\epsilon^{-2}\big).
\end{aligned}
$$

Using Markov inequality, with probability at least 2/3, we have

$$T_{\text{total}} = \tilde{O}\big(\Delta_F nL_3\epsilon_H^{-2} + \Delta_F n^{3/4}L_1^{1/2}L_3\epsilon_H^{-5/2} + \Delta_F n^{1/2}L_1\epsilon^{-2}\big).$$

Note that $|\mathcal{I}_1|\delta = |\mathcal{I}_1|/(72 \cdot L_3\Delta_F\epsilon_H^{-2}) \leq 1/12$. By union bound, with probability at least $1-1/3-1/3-|\mathcal{I}_1|\delta \geq 1/4$, SNVRG + Neon2$^{\text{finite}}$ will find an (ϵ, ϵ_H)-second order stationary point within

$$\tilde{O}\big(\Delta_F nL_3\epsilon_H^{-2} + \Delta_F n^{3/4}L_1^{1/2}L_3\epsilon_H^{-5/2} + \Delta_F n^{1/2}L_1\epsilon^{-2}\big)$$

stochastic gradient computations. $\square$

2.8 Proof of Supporting Lemmas

2.8.1 Proof of Lemma 2.7.1

We first prove our key lemma on One-epoch-SNVRG. In order to prove Lemma 2.7.1, we need the following supporting lemma, which shows that with any chosen epoch length T, the summation of expectation of the square of gradient norm $\sum_{j=0}^{T-1}\mathbb{E}\|\nabla F(\mathbf{x}_j)\|_2^2$ can be bounded.

Lemma 2.8.1. *Suppose we arbitrarily fix the amount of epochs $T > 1$ in Algorithm 1. In other words, we do not bother with Options I or II for the present. If the step size and batch size parameters in Algorithm 1 satisfy $M \geq 6L$ and $B_l \geq 6^{K-l+1}(\prod_{s=l}^{K} T_s)^2$ for any $1 \leq l \leq K$, then the iterates of Algorithm 1 satisfies*

$$\sum_{j=0}^{T-1}\mathbb{E}\|\nabla F(\mathbf{x}_j)\|_2^2 \leq C\bigg(M\mathbb{E}\big[F(\mathbf{x}_0) - F(\mathbf{x}_T)\big] + \frac{2\sigma^2 T}{B_0}\cdot \mathbb{1}\{B_0 < n\}\bigg), \tag{2.8.1}$$

where $C = 100$.

Proof of Lemma 2.7.1. We can check that $2 \leq B_0^{2^{-K}} < 4$, and we can check that the choice of $M, \{T_l\}, \{B_l\}$ in Lemma 2.7.1 satisfies the assumption of Lemma 2.8.1. Moreover, we have

$$\begin{aligned}T &= \prod_{l=1}^{K} T_l \\ &> (B_0^{2^{-K}} - 1)\prod_{l=2}^{K}(B_0^{2^{l-K-2}} - 1)\end{aligned}$$

$$
\begin{aligned}
&> \frac{1}{2} B_0^{2^{-K}} \cdot \prod_{l=2}^{K} B_0^{2^{l-K-2}} \cdot \bigg(1 - \bigg(\sum_{l=2}^{K} \frac{1}{B_0^{2^{l-K-2}}}\bigg)\bigg) \\
&\geq \frac{1}{2} B_0^{1/2} \bigg(1 - \bigg(\sum_{l=2}^{K} \frac{1}{2^{2^{l-2}}}\bigg)\bigg) \\
&> \frac{1}{10} B_0^{1/2}, \qquad (2.8.2)
\end{aligned}
$$

where the first inequality holds due to the fact $\lfloor x \rfloor > x - 1$ for any $x > 1$, the second inequality holds since $2 \leq B_0^{2^{-K}} < 4$ and the fact $\prod_{l=2}^{K}(x_l - 1) > \prod_{l=2}^{K} x_l(1 - \sum_{l=2}^{K} x_l^{-1})$ for any sequence $\{x_l\}_{l=2}^{K}$ satisfying $\forall 2 \leq l \leq K, x_l \geq 2$, the third inequality holds since $2^{2^K} \leq B_0$, the last inequality holds due to the fact that $\sum_{l=2}^{K} 2^{-2^{l-2}} < 4/5$. We now submit (2.8.2) into (2.8.1), which immediately implies (2.7.1). Next we compute how many stochastic gradient computations we need in total after we run One-epoch-SNVRG once. According to the update of reference gradients in Algorithm 1, we only update $\mathbf{g}_t^{(0)}$ once at the beginning of Algorithm 1 (Line 23 is only reached when $r = 0$), which needs B_0 stochastic gradient computations. For $\mathbf{g}_t^{(l)}$, we only need to update it when $0 = (t \mod \prod_{j=l+1}^{K} T_j)$, and thus we need to sample $\mathbf{g}_t^{(l)}$ for $T/\prod_{j=l+1}^{K} T_j = \prod_{j=1}^{l} T_j$ times. We need $2B_l$ stochastic gradient computations for each sampling procedure (Line 20 in Algorithm 1). We use $\mathcal{T}$ to represent the total number of stochastic gradient computations, then based on above arguments we have

$$
\mathcal{T} = B_0 + 2\sum_{l=1}^{K} B_l \cdot \prod_{j=1}^{l} T_j. \qquad (2.8.3)
$$

Now we calculate $\mathcal{T}$ under the parameter choice of Lemma 2.7.1. Note that we can easily verify the following inequalities:

$$
\begin{aligned}
&\prod_{j=1}^{l} T_j \leq B_0^{2^{-K}} \prod_{j=2}^{l} B_0^{2^{j-K-2}} = B_0^{\frac{2^l}{2^{K+1}}}, \\
&\bigg(\prod_{j=l}^{K} T_j\bigg)^2 \leq \bigg(\prod_{j=l}^{K} B_0^{2^{j-K-2}}\bigg)^2 = B_0^{1-2^{K+1-l}}, \qquad \forall 2 \leq l \leq K, \\
&\bigg(\prod_{j=1}^{K} T_j\bigg)^2 \leq \bigg(B_0^{2^{-K}} \cdot \prod_{j=2}^{K} B_0^{2^{j-K-2}}\bigg)^2 = B_0,
\end{aligned}
$$

which implies that

$$B_1 \cdot \prod_{j=1}^{1} T_j = 6^K \bigg(\prod_{j=1}^{K} T_j \bigg)^2 T_1 \le 6^K B_0 \cdot 4,$$

$$B_l \cdot \prod_{j=1}^{l} T_j = 6^{K-l+1} \bigg(\prod_{j=l}^{K} T_j \bigg)^2 \prod_{j=1}^{l} T_j \le 6^{K-l+1} B_0. \tag{2.8.4}$$

Submit (2.8.4) into (2.8.3) yields the following results:

$$\begin{aligned}
\mathcal{T} &= B_0 + 2\bigg(4 \times 6^K B_0 + \sum_{l=2}^{K} 6^{K-l+1} B_0\bigg) \\
&< B_0 + 9 \times 6^K B_0 \\
&\le B_0 + 9 \times 6^{\log\log B_0} B_0 \\
&< B_0 + 9 B_0 \log^3 B_0.
\end{aligned}$$

Therefore, the total gradient complexity $\mathcal{T}$ is bounded as follows.

$$\mathcal{T} = B_0 + 2 \sum_{l=1}^{K} B_l \cdot \prod_{j=1}^{l} T_j \le B_0 + 9 B_0 \log^3 B_0 \le 10 B_0 \log^3 B_0. \tag{2.8.5}$$

□

2.8.2 Proof of Lemma 2.7.2

Now we prove Lemma 2.7.2 about the function value decrease of Algorithm 1 with Option II. Note that Lemma 2.8.1 shows that with any chosen epoch length T, the summation of expectation of the square of gradient norm $\sum_{j=0}^{T-1} \mathbb{E}\|\nabla F(\mathbf{x}_j)\|_2^2$ can be bounded. In order to prove the upper bound on $\mathbb{E}\|\nabla F(\mathbf{x}_T)\|_2^2$, we need the following technical lemma about geometric distribution.

Lemma 2.8.2. *Suppose that $G \sim$ Geom(p), where $\mathbb{P}(G = k) = p(1-p)^k, k \ge 0$. Let $a(j), b(j)$ be two series and $b(0) \ge 0$. If for any $k \ge 1$, it holds that $\sum_{j=0}^{k-1} a(j) \le b(k)$, then we have*

$$\frac{1-p}{p} \mathbb{E}_G a(G) \le \mathbb{E}_G b(G).$$

Proof of Lemma 2.7.2. We can easily check that the choice of $M, \{T_l\}, \{B_l\}$ in Lemma 2.7.2 satisfies the assumption of Lemma 2.8.1. By Algorithm 1, we have $T \sim \text{Geom}(p)$ where $p = 1/(1 + \prod_{j=1}^K T_j)$. Let

$$a(j) = \mathbb{E}\|\nabla F(\mathbf{x}_j)\|_2^2,\ b(j) = C\bigg(M\mathbb{E}\big[F(\mathbf{x}_0) - F(\mathbf{x}_j)\big] + \frac{\sigma^2 j}{B_0} \cdot \mathbb{1}\{B_0 < n\}\bigg).$$

Then by Lemma 2.8.1, for any $T \geq 1$, we have $\sum_{j=0}^{T-1} a(j) \leq b(T)$ and $b(0) = 0$. Thus, by Lemma 2.8.2, we have

$$\frac{1-p}{p}\mathbb{E}_T\mathbb{E}\|\nabla F(\mathbf{x}_T)\|_2^2 \leq C\bigg(M\mathbb{E}_T\mathbb{E}\big[F(\mathbf{x}_0) - F(\mathbf{x}_T)\big] + \frac{2\sigma^2\mathbb{E}_T T}{B_0} \cdot \mathbb{1}\{B_0 < n\}\bigg).$$

Since $\mathbb{E}_T T = (1-p)/p = \prod_{j=1}^K T_j > B_0^{1/2}/10$ due to (2.8.2) , we have

$$\begin{aligned}\mathbb{E}\|\nabla F(\mathbf{x}_T)\|_2^2 &\leq C\bigg(\frac{M}{\prod_{j=1}^K T_j}\mathbb{E}\big[F(\mathbf{x}_0) - F(\mathbf{x}_T)\big] + \frac{2\sigma^2}{B_0} \cdot \mathbb{1}\{B_0 < n\}\bigg)\\ &\leq 10C\bigg(\frac{M}{B_0^{1/2}}\mathbb{E}\big[F(\mathbf{x}_0) - F(\mathbf{x}_T)\big] + \frac{2\sigma^2}{B_0} \cdot \mathbb{1}\{B_0 < n\}\bigg),\end{aligned}$$

which immediately implies (2.7.7).

Finally we consider how many stochastic gradient computations for us to run One-epoch-SNVRG once. According to the update of reference gradients in Algorithm 1, for $\mathbf{g}_t^{(l)}$, we need to update it when $0 = (t \mod \prod_{j=l+1}^K T_j)$, and thus we need to sample $\mathbf{g}_t^{(l)}$ for $T/\prod_{j=l+1}^K T_j$ times. We need B_0 stochastic gradient computations to update $\mathbf{g}_t^{(0)}$ and $2B_l$ stochastic gradient computations for $\mathbf{g}_t^{(l)}$ (Lines 20 and 23 in Algorithm 1 respectively). If we use $\mathcal{T}$ to represent the total number of stochastic gradient computations, then based on above arguments, we have

$$\begin{aligned}\mathbb{E}\mathcal{T} &\leq B_0 \cdot \frac{\mathbb{E}T}{\prod_{j=1}^K T_j} + 2\sum_{l=1}^K B_l \cdot \frac{\mathbb{E}T}{\prod_{j=l+1}^K T_j}\\ &= B_0 + 2\sum_{l=1}^K B_l \prod_{j=1}^l T_j\\ &\leq 10B_0 \log^3 B_0,\end{aligned}$$

where the last inequality holds due to (2.8.5). □

2.9 Proof of Key Lemma 2.8.1

In this section, we focus on proving Lemma 2.8.1, which holds for any fixed T and plays a pivotal role in the analyses of Algorithm 1 with both Option I and Option II. Let $M, \{T_i\}, \{B_i\}, B_0$ be the parameters as defined in Algorithm 1. We define filtration $\mathcal{F}_t = \sigma(\mathbf{x}_0, \dots, \mathbf{x}_t)$. Let $\{\mathbf{x}_t^{(l)}\}, \{\mathbf{g}_t^{(l)}\}$ be the reference points and reference gradients in Algorithm 1. We define $\mathbf{v}_t^{(l)}$ as

$$\mathbf{v}_t^{(l)} := \sum_{j=0}^{l} \mathbf{g}_t^{(j)}, \quad \text{for } 0 \le l \le K. \tag{2.9.1}$$

We first present the following definition and two technical lemmas for the purpose of our analysis.

Definition 2.9.1. *We define constant series $\{c_j^{(s)}\}$ as the following. For each s, we define $c_{T_s}^{(s)}$ as*

$$c_{T_s}^{(s)} = \frac{M}{6^{K-s+1} \prod_{l=s}^{K} T_l}. \tag{2.9.2}$$

When $0 \le j < T_s$, we define $c_j^{(s)}$ by induction:

$$c_j^{(s)} = \left(1 + \frac{1}{T_s}\right) c_{j+1}^{(s)} + \frac{3L^2}{M} \cdot \frac{\prod_{l=s+1}^{K} T_l}{B_s}. \tag{2.9.3}$$

Lemma 2.9.2. *For any p, s, where $1 \le s \le K$, $p \cdot \prod_{j=s}^{K} T_j < T$ and $q \prod_{j=1}^{K} T_j \le p \cdot \prod_{j=s}^{K} T_j < (p+1) \cdot \prod_{j=s}^{K} T_j \le (q+1) \prod_{j=1}^{K} T_j$, we define*

$$\mathit{start} = p \cdot \prod_{j=s}^{K} T_j, \ \ \mathit{end} = \min\left\{\mathit{start} + \prod_{j=s}^{K} T_j, T\right\}$$

for simplification. Then we have the following results:

$$\mathbb{E}\left[\sum_{j=\mathit{start}}^{\mathit{end}-1} \frac{\|\nabla F(\mathbf{x}_j)\|_2^2}{100M} + F(\mathbf{x}_{\mathit{end}}) + c_{T_s}^{(s)} \cdot \|\mathbf{x}_{\mathit{end}} - \mathbf{x}_{\mathit{start}}\|_2^2 \big| \mathcal{F}_{\mathit{start}}\right]$$
$$\le F(\mathbf{x}_{\mathit{start}}) + \frac{2}{M} \cdot \mathbb{E}\left[\left\|\nabla F(\mathbf{x}_{\mathit{start}}) - \mathbf{v}_{\mathit{start}}\right\|_2^2 \big| \mathcal{F}_{\mathit{start}}\right] \cdot (\mathit{end} - \mathit{start}).$$

Lemma 2.9.3 ([LJCJ17]). *Let $\mathbf{a}_i$ be vectors satisfying $\sum_{i=1}^{N} \mathbf{a}_i = 0$. Let $\mathcal{J}$ be a uniform random subset of $\{1, \dots, N\}$ with size m, then*

$$\mathbb{E}\left\|\frac{1}{m}\sum_{j \in \mathcal{J}} \mathbf{a}_j\right\|_2^2 \le \frac{\mathbb{1}(|\mathcal{J}| < N)}{mN} \sum_{j=1}^{N} \|\mathbf{a}_j\|_2^2.$$

Proof of Lemma 2.8.1. We have

$$\sum_{j=0}^{T-1}\frac{\mathbb{E}\|\nabla F(\mathbf{x}_j)\|_2^2}{100M}+\mathbb{E}\big[F(\mathbf{x}_T)\big] \le \sum_{j=0}^{T-1}\frac{\mathbb{E}\|\nabla F(\mathbf{x}_j)\|_2^2}{100M}+\mathbb{E}\big[F(\mathbf{x}_T)+c_{T_1}^{(1)}\cdot\|\mathbf{x}_T-\mathbf{x}_0\|_2^2\big]$$
$$\le \mathbb{E}\big[F(\mathbf{x}_0)\big]+\frac{2}{M}\cdot\mathbb{E}\|\nabla F(\mathbf{x}_0)-\mathbf{g}_0\|_2^2\cdot T, \tag{2.9.4}$$

where the second inequality comes from Lemma 2.9.2 with we take $s=1, p=0$. Moreover we have

$$\mathbb{E}\|\nabla F(\mathbf{x}_0)-\mathbf{g}_0\|_2^2 = \mathbb{E}\bigg\|\frac{1}{B_0}\sum_{i\in I}\big[\nabla f_i(\mathbf{x}_0)-\nabla F(\mathbf{x}_0)\big]\bigg\|_2^2$$
$$\le \mathbb{1}(B_0<n)\cdot\frac{1}{B_0}\cdot\frac{1}{n}\sum_{i=1}^{n}\big\|\nabla f_i(\mathbf{x}_0)-\nabla F(\mathbf{x}_0)\big\|_2^2 \tag{2.9.5}$$
$$\le \mathbb{1}(B_0<n)\cdot\frac{\sigma^2}{B_0}, \tag{2.9.6}$$

where (2.9.5) holds because of Lemma 2.9.3. Plug (2.9.6) into (2.9.4) and note that we have $M=6L$, and then we obtain

$$\sum_{j=0}^{T-1}\mathbb{E}\|\nabla F(\mathbf{x}_j)\|_2^2 \le C\bigg(M\mathbb{E}\big[F(\mathbf{x}_0)-F(\mathbf{x}_T)\big]+\frac{2T\sigma^2}{B_0}\cdot\mathbb{1}(B_0<n)\bigg), \tag{2.9.7}$$

where $C=100$, which complete the proof of Lemma 2.8.1. □

2.10 Proof of Technical Lemmas

In this section, we provide the proofs of technical lemmas used in Appendix 2.9.

2.10.1 Proof of Lemma 2.9.2

Let $M, \{T_l\}, \{B_l\}, B_0$ be the parameters defined in Algorithm 1 and $\{\mathbf{x}_t^{(l)}\}, \{\mathbf{g}_t^{(l)}\}$ be the reference points and reference gradients defined in Algorithm 1. Let $\mathbf{v}_t^{(l)}, \mathcal{F}_t$ be the variables and filtration defined in Appendix 2.9 and let $c_j^{(s)}$ be the constant series defined in Definition 2.9.1.

In order to prove Lemma 2.9.2, we will need the following supporting propositions and lemmas. We first state the proposition about the relationship among $\mathbf{x}_t^{(s)}, \mathbf{g}_t^{(s)}$ and $\mathbf{v}_t^{(s)}$:

Proposition 2.10.1. *Let $\mathbf{v}_t^{(l)}$ be defined as in (2.9.1). Let p, s satisfy $0 \le p \cdot \prod_{j=s+1}^{K} T_j < (p+1) \cdot \prod_{j=s+1}^{K} T_j < T$. For any t, t' satisfying $p \cdot \prod_{j=s+1}^{K} T_j \le t < t' < (p+1) \cdot \prod_{j=s+1}^{K} T_j$, it holds that*

$$\mathbf{x}_t^{(s)} = \mathbf{x}_{t'}^{(s)} = \mathbf{x}_{p \prod_{j=s+1}^{K} T_j}, \tag{2.10.1}$$

$$\mathbf{g}_t^{(s')} = \mathbf{g}_{t'}^{(s')}, \qquad \textit{for any } s' \textit{ that satisfies } 0 \le s' \le s, \tag{2.10.2}$$

$$\mathbf{v}_t^{(s)} = \mathbf{v}_{t'}^{(s)} = \mathbf{v}_{p \prod_{j=s+1}^{K} T_j}. \tag{2.10.3}$$

The following lemma spells out the relationship between $c_j^{(s-1)}$ and $c_{T_s}^{(s)}$. In a word, $c_j^{(s-1)}$ is about $1 + T_{s-1}$ times less than $c_{T_s}^{(s)}$:

Lemma 2.10.2. *If $B_s \ge 6^{K-s+1} (\prod_{l=s}^{K} T_l)^2, T_l \ge 1$ and $M \ge 6L$, then it holds that*

$$c_j^{(s-1)} \cdot (1 + T_{s-1}) < c_{T_s}^{(s)}, \qquad \textit{for } 2 \le s \le K, 0 \le j \le T_{s-1}, \tag{2.10.4}$$

and

$$c_j^{(K)} \cdot (1 + T_K) < M, \qquad \textit{for } 0 \le j \le T_K. \tag{2.10.5}$$

Next lemma is a special case of Lemma 2.9.2 with $s = K$:

Lemma 2.10.3. *Suppose p satisfies $q \prod_{i=1}^{K} T_i \le pT_K < (p+1)T_K \le (q+1) \prod_{i=1}^{K} T_i$ for some q and $pT_K < T$. For simplification, we denote*

$$\mathit{start} = pT_K, \mathit{end} = \min\{(p+1)T_K, T\}.$$

If $M > L$, then we have

$$\mathbb{E}\left[F(\mathbf{x}_{\mathit{end}}) + c_{T_K}^{(K)} \cdot \|\mathbf{x}_{\mathit{end}} - \mathbf{x}_{\mathit{start}}\|_2^2 + \sum_{j=\mathit{start}}^{\mathit{end}-1} \frac{\|\nabla F(\mathbf{x}_j)\|_2^2}{100M} \middle| \mathcal{F}_{\mathit{start}}\right]$$
$$\le F(\mathbf{x}_{\mathit{start}}) + \frac{2}{M} \cdot \mathbb{E}\left[\|\nabla F(\mathbf{x}_{\mathit{start}}) - \mathbf{v}_{\mathit{start}}\|_2^2 \middle| \mathcal{F}_{\mathit{start}}\right] \cdot (\mathit{end} - \mathit{start}).$$

The following lemma provides an upper bound of $\mathbb{E}\left[\|\nabla F(\mathbf{x}_t^{(l)}) - \mathbf{v}_t^{(l)}\|_2^2\right]$, which plays an important role in our proof of Lemma 2.9.2.

Lemma 2.10.4. *Let t^l be as defined in* (2.4.1)*, then we have $\mathbf{x}_t^{(l)} = \mathbf{x}_{t^l}$, and*

$$\mathbb{E}\big[\big\|\nabla F(\mathbf{x}_t^{(l)}) - \mathbf{v}_t^{(l)}\big\|_2^2 \big| \mathcal{F}_{t^l}\big] \le \frac{L^2}{B_l}\big\|\mathbf{x}_t^{(l)} - \mathbf{x}_t^{(l-1)}\big\|_2^2 + \big\|\nabla F(\mathbf{x}_t^{(l-1)}) - \mathbf{v}_t^{(l-1)}\big\|_2^2.$$

Proof of Lemma 2.9.2. We use mathematical induction to prove that Lemma 2.9.2 holds for any $1 \le s \le K$. When $s = K$, we have the result hold because of Lemma 2.10.3. Suppose that for $s+1$, Lemma 2.9.2 holds for any p' which satisfies $p'\prod_{j=s+1}^K T_j < T$ and $q\prod_{j=1}^K T_j \le p'\prod_{j=s+1}^K T_j < (p'+1)\prod_{j=s+1}^K T_j \le (q+1)\prod_{j=1}^K T_j$. We need to prove Lemma 2.9.2 still holds for s and p, where p satisfies $p\prod_{j=s+1}^K T_j < T$ and $q\prod_{j=1}^K T_j \le p\prod_{j=s}^K T_j < (p+1)\prod_{j=s}^K T_j \le (q+1)\prod_{j=1}^K T_j$. We choose $p' = pT_s + u$ which satisfies that $p'\prod_{j=s+1}^K T_j < T$, and we set indices start_u and end_u as

$$\mathrm{start}_u = p'\prod_{j=s+1}^K T_j, \qquad \mathrm{end}_u = \min\Big\{\mathrm{start}_u + \prod_{j=s+1}^K T_j, T\Big\}.$$

Then we have

$$\begin{aligned}&\mathbb{E}\Bigg[\sum_{j=\mathrm{start}_u}^{\mathrm{end}_u-1}\frac{\|\nabla F(\mathbf{x}_j)\|_2^2}{100M} + F(\mathbf{x}_{\mathrm{end}_u}) + c_{T_{s+1}}^{(s+1)}\cdot\|\mathbf{x}_{\mathrm{end}_u} - \mathbf{x}_{\mathrm{start}_u}\|_2^2\big|\mathcal{F}_{\mathrm{start}_u}\Bigg]\\ &\le F(\mathbf{x}_{\mathrm{start}_u}) + \frac{2}{M}\cdot\mathbb{E}\big[\big\|\nabla F(\mathbf{x}_{\mathrm{start}_u}) - \mathbf{v}_{\mathrm{start}_u}\big\|_2^2\big|\mathcal{F}_{\mathrm{start}_u}\big]\cdot(\mathrm{end}_u - \mathrm{start}_u),\end{aligned} \tag{2.10.6}$$

where the last inequality holds because of the induction hypothesis that Lemma 2.9.2 holds for $s+1$ and p'. Note that we have $\mathbf{x}_{\mathrm{start}_u} = \mathbf{x}_{\mathrm{start}_u}^{(s)}$ from Proposition 2.10.1, which implies

$$\begin{aligned}\mathbb{E}\big[\big\|\nabla F(\mathbf{x}_{\mathrm{start}_u}) - \mathbf{v}_{\mathrm{start}_u}\big\|_2^2\big|\mathcal{F}_{\mathrm{start}_u}\big] &= \mathbb{E}\big[\big\|\nabla F(\mathbf{x}_{\mathrm{start}_u}^{(s)}) - \mathbf{v}_{\mathrm{start}_u}^{(s)}\big\|_2^2\big|\mathcal{F}_{\mathrm{start}_u}\big]\\ &\le \frac{L^2}{B_s}\big\|\mathbf{x}_{\mathrm{start}_u}^{(s)} - \mathbf{x}_{\mathrm{start}_u}^{(s-1)}\big\|_2^2 + \big\|\nabla F(\mathbf{x}_{\mathrm{start}_u}^{(s-1)}) - \mathbf{v}_{\mathrm{start}_u}^{(s-1)}\big\|_2^2 \qquad (2.10.7)\\ &= \frac{L^2}{B_s}\|\mathbf{x}_{\mathrm{start}_u} - \mathbf{x}_{\mathrm{start}}\|_2^2 + \big\|\nabla F(\mathbf{x}_{\mathrm{start}}) - \mathbf{v}_{\mathrm{start}}\big\|_2^2, \qquad (2.10.8)\end{aligned}$$

where (2.10.7) holds because of Lemma 2.10.4 and (2.10.8) holds due to Proposition 2.10.1. Plugging (2.10.8) into (2.10.6) and taking expectation $\mathbb{E}[\cdot|\mathcal{F}_{\mathrm{start}}]$ for (2.10.6) will yield

$$\mathbb{E}\Bigg[\sum_{j=\mathrm{start}_u}^{\mathrm{end}_u-1}\frac{\|\nabla F(\mathbf{x}_j)\|_2^2}{100M} + F(\mathbf{x}_{\mathrm{end}_u}) + c_{T_{s+1}}^{(s+1)}\|\mathbf{x}_{\mathrm{end}_u} - \mathbf{x}_{\mathrm{start}_u}\|_2^2\big|\mathcal{F}_{\mathrm{start}}\Bigg]$$

$$\le \mathbb{E}\bigg[F(\mathbf{x}_{\text{start}_u}) + (\text{end}_u - \text{start}_u)\frac{2L^2}{MB_s}\|\mathbf{x}_{\text{start}_u} - \mathbf{x}_{\text{start}}\|_2^2$$

$$+ \frac{2(\text{end}_u - \text{start}_u)}{M}\big\|\nabla F(\mathbf{x}_{\text{start}}) - \mathbf{v}_{\text{start}}\big\|_2^2\bigg|\mathcal{F}_{\text{start}}\bigg]$$

$$\le \mathbb{E}\bigg[F(\mathbf{x}_{\text{start}_u}) + \bigg(\prod_{j=s+1}^{K} T_j\bigg)\frac{2L^2}{MB_s}\|\mathbf{x}_{\text{start}_u} - \mathbf{x}_{\text{start}}\|_2^2$$

$$+ \frac{2(\text{end}_u - \text{start}_u)}{M}\big\|\nabla F(\mathbf{x}_{\text{start}}) - \mathbf{v}_{\text{start}}\big\|_2^2\bigg|\mathcal{F}_{\text{start}}\bigg]. \tag{2.10.9}$$

We now give a bound of $\|\mathbf{x}_{\text{end}_u} - \mathbf{x}_{\text{start}}\|_2^2$:

$$\|\mathbf{x}_{\text{end}_u} - \mathbf{x}_{\text{start}}\|_2^2$$

$$= \|\mathbf{x}_{\text{start}_u} - \mathbf{x}_{\text{start}}\|_2^2 + \|\mathbf{x}_{\text{end}_u} - \mathbf{x}_{\text{start}_u}\|_2^2 + 2\langle \mathbf{x}_{\text{end}_u} - \mathbf{x}_{\text{start}_u}, \mathbf{x}_{\text{start}_u} - \mathbf{x}_{\text{start}}\rangle$$

$$\le \|\mathbf{x}_{\text{start}_u} - \mathbf{x}_{\text{start}}\|_2^2 + \|\mathbf{x}_{\text{end}_u} - \mathbf{x}_{\text{start}_u}\|_2^2 + \frac{1}{T_s}\cdot\|\mathbf{x}_{\text{start}_u} - \mathbf{x}_{\text{start}}\|_2^2 + T_s\cdot\|\mathbf{x}_{\text{end}_u} - \mathbf{x}_{\text{start}_u}\|_2^2 \tag{2.10.10}$$

$$= \bigg(1 + \frac{1}{T_s}\bigg)\cdot\|\mathbf{x}_{\text{start}_u} - \mathbf{x}_{\text{start}}\|_2^2 + (1+T_s)\cdot\|\mathbf{x}_{\text{end}_u} - \mathbf{x}_{\text{start}_u}\|_2^2, \tag{2.10.11}$$

where (2.10.10) holds because of Young's inequality. Taking expectation $\mathbb{E}[\cdot|\mathcal{F}_{\text{start}}]$ over (2.10.11) and multiplying $c_{u+1}^{(s)}$ on both sides, we obtain

$$c_{u+1}^{(s)}\mathbb{E}\big[\|\mathbf{x}_{\text{end}_u} - \mathbf{x}_{\text{start}}\|_2^2\big|\mathcal{F}_{\text{start}}\big] \le c_{u+1}^{(s)}\bigg(1 + \frac{1}{T_s}\bigg)\mathbb{E}\big[\|\mathbf{x}_{\text{start}_u} - \mathbf{x}_{\text{start}}\|_2^2\big|\mathcal{F}_{\text{start}}\big]$$

$$+ c_{u+1}^{(s)}(1+T_s)\mathbb{E}\big[\|\mathbf{x}_{\text{end}_u} - \mathbf{x}_{\text{start}_u}\|_2^2\big|\mathcal{F}_{\text{start}}\big]. \tag{2.10.12}$$

Adding up inequalities(2.10.12) and (2.10.9) together, we have

$$\mathbb{E}\bigg[\sum_{j=\text{start}_u}^{\text{end}_u - 1}\frac{\|\nabla F(\mathbf{x}_j)\|_2^2}{100M} + F(\mathbf{x}_{\text{end}_u}) + c_{u+1}^{(s)}\|\mathbf{x}_{\text{end}_u} - \mathbf{x}_{\text{start}}\|_2^2 + c_{T_{s+1}}^{(s+1)}\|\mathbf{x}_{\text{end}_u} - \mathbf{x}_{\text{start}_u}\|_2^2\bigg|\mathcal{F}_{\text{start}}\bigg]$$

$$\le \mathbb{E}\bigg[F(\mathbf{x}_{\text{start}_u}) + \|\mathbf{x}_{\text{start}_u} - \mathbf{x}_{\text{start}}\|_2^2\bigg[c_{u+1}^{(s)}\bigg(1 + \frac{1}{T_s}\bigg) + \frac{3L^2}{B_s M}\prod_{j=s+1}^{K} T_j\bigg]\bigg|\mathcal{F}_{\text{start}}\bigg]$$

$$+ \frac{2}{M}\mathbb{E}\big[\big\|\nabla F(\mathbf{x}_{\text{start}}) - \mathbf{v}_{\text{start}}\big\|_2^2\big|\mathcal{F}_{\text{start}}\big](\text{end}_u - \text{start}_u)$$

$$+ c_{u+1}^{(s)}(1+T_s)\mathbb{E}\big[\|\mathbf{x}_{\text{end}_u} - \mathbf{x}_{\text{start}_u}\|_2^2\big|\mathcal{F}_{\text{start}}\big]$$

$$< \mathbb{E}\big[F(\mathbf{x}_{\text{start}_u}) + c_u^{(s)}\|\mathbf{x}_{\text{start}_u} - \mathbf{x}_{\text{start}}\|_2^2\big|\mathcal{F}_{\text{start}}\big]$$

$$+ \frac{2}{M}\mathbb{E}\big[\big\|\nabla F(\mathbf{x}_{\text{start}}) - \mathbf{v}_{\text{start}}\big\|_2^2\big|\mathcal{F}_{\text{start}}\big](\text{end}_u - \text{start}_u)$$

$$+ c_{T_{s+1}}^{(s+1)}\mathbb{E}\big[\|\mathbf{x}_{\mathrm{end}_u} - \mathbf{x}_{\mathrm{start}_u}\|_2^2\big|\mathcal{F}_{\mathrm{start}}\big], \tag{2.10.13}$$

where the last inequality holds due to the fact that $c_u^{(s)} = c_{u+1}^{(s)}(1 + 1/T_s) + 3L^2/(B_sM) \cdot \prod_{j=s+1}^K T_j$ by Definition 2.9.1 and $c_{u+1}^{(s)} \cdot (1 + T_s) < c_{T_{s+1}}^{(s+1)}$ by Lemma 2.10.2. Cancelling out the term $c_{T_{s+1}}^{(s+1)}\mathbb{E}\big[\|\mathbf{x}_{\mathrm{end}_u} - \mathbf{x}_{\mathrm{start}_u}\|_2^2\big|\mathcal{F}_{\mathrm{start}}\big]$ from both sides of (2.10.13), we get

$$\begin{aligned}&\sum_{j=\mathrm{start}_u}^{\mathrm{end}_u-1}\mathbb{E}\bigg[\frac{\|\nabla F(\mathbf{x}_j)\|_2^2}{100M}\bigg|\mathcal{F}_{\mathrm{start}}\bigg] + \mathbb{E}\big[F(\mathbf{x}_{\mathrm{end}_u}) + c_{u+1}^{(s)} \cdot \|\mathbf{x}_{\mathrm{end}_u} - \mathbf{x}_{\mathrm{start}}\|_2^2\big|\mathcal{F}_{\mathrm{start}}\big]\\ &\le \mathbb{E}\big[F(\mathbf{x}_{\mathrm{start}_u}) + c_u^{(s)}\|\mathbf{x}_{\mathrm{start}_u} - \mathbf{x}_{\mathrm{start}}\|_2^2\big|\mathcal{F}_{\mathrm{start}}\big]\\ &\qquad + \frac{2}{M}\mathbb{E}\big[\big\|\nabla F(\mathbf{x}_{\mathrm{start}}) - \mathbf{v}_{\mathrm{start}}\big\|_2^2\big|\mathcal{F}_{\mathrm{start}}\big](\mathrm{end}_u - \mathrm{start}_u).\end{aligned} \tag{2.10.14}$$

We now try to telescope the above inequality. We first suppose that $u^* = \max\{0 \le u < T_s : \mathrm{start}_u < T\}$. Next we telescope (2.10.14) for $u = 0$ to u^*. Since we have $\mathrm{start}_u = \mathrm{end}_{u-1}, \mathrm{start}_0 = \mathrm{start}$ for $0 \le u \le u^*$, then we get

$$\begin{aligned}&\mathbb{E}\bigg[\sum_{u=0}^{u^*}\sum_{j=\mathrm{start}_u}^{\mathrm{end}_u-1}\frac{\|\nabla F(\mathbf{x}_j)\|_2^2}{100M} + F(\mathbf{x}_{\mathrm{end}_{u^*}}) + c_{u^*}^{(s)} \cdot \|\mathbf{x}_{\mathrm{end}_{u^*}} - \mathbf{x}_{\mathrm{start}}\|_2^2\big|\mathcal{F}_{\mathrm{start}}\bigg]\\ &\le F(\mathbf{x}_{\mathrm{start}}) + \frac{2T_s}{M} \cdot \mathbb{E}\big[\big\|\nabla F(\mathbf{x}_{\mathrm{start}}) - \mathbf{v}_{\mathrm{start}}\big\|_2^2\big|\mathcal{F}_{\mathrm{start}}\big] \cdot \sum_{u=0}^{u^*}(\mathrm{end}_u - \mathrm{start}_u).\end{aligned}$$

Since for $0 \le u \le u^*$, we have $\mathrm{start}_u = \mathrm{end}_{u-1}, \mathrm{start}_0 = \mathrm{start}$, $\mathrm{end}_{u^*} = \mathrm{end}$, and $c_{u^*}^{(s)} > c_{T_s}^{(s)}$, thus we have that

$$\begin{aligned}&\mathbb{E}\bigg[\sum_{j=\mathrm{start}}^{\mathrm{end}-1}\frac{\|\nabla F(\mathbf{x}_j)\|_2^2}{100M} + F(\mathbf{x}_{\mathrm{end}}) + c_{T_s}^{(s)} \cdot \|\mathbf{x}_{\mathrm{end}} - \mathbf{x}_{\mathrm{start}}\|_2^2\big|\mathcal{F}_{\mathrm{start}}\bigg]\\ &\le F(\mathbf{x}_{\mathrm{start}}) + \frac{2}{M} \cdot \mathbb{E}\big[\big\|\nabla F(\mathbf{x}_{\mathrm{start}}) - \mathbf{v}_{\mathrm{start}}\big\|_2^2\big|\mathcal{F}_{\mathrm{start}}\big] \cdot (\mathrm{end} - \mathrm{start}).\end{aligned} \tag{2.10.15}$$

Therefore, we have proved that Lemma 2.9.2 still holds for s and p. Then by mathematical induction, we have for all $1 \le s \le K$ and p which satisfy $q\prod_{j=1}^K T_j \le p \cdot \prod_{j=s}^K T_j < (p+1) \cdot \prod_{j=s}^K T_j \le (q+1)\prod_{j=1}^K T_j$, Lemma 2.9.2 holds. □

2.10.2 Proof of Lemma 2.9.3

The following proof is adapted from that of Lemma A.1 in [LJCJ17]. We provide the proof here for the self-containedness of our work.

Proof of Lemma 2.9.3. We only consider the case when $m < N$. Let $W_j = \mathbb{1}(j \in \mathcal{J})$, then we have

$$\mathbb{E}W_j^2 = \mathbb{E}W_j = \frac{m}{N}, \mathbb{E}W_j W_{j'} = \frac{m(m-1)}{N(N-1)}.$$

Thus we can rewrite the sample mean as

$$\frac{1}{m}\sum_{j\in\mathcal{J}} \mathbf{a}_j = \frac{1}{m}\sum_{i=1}^{N} W_i \mathbf{a}_i,$$

which immediately implies

$$\begin{aligned}
\mathbb{E}\left\|\frac{1}{m}\sum_{j\in\mathcal{J}} \mathbf{a}_j\right\|^2 &= \frac{1}{m^2}\bigg(\sum_{j=1}^{N} \mathbb{E}W_j^2 \|\mathbf{a}_j\|_2^2 + \sum_{j\neq j'} \mathbb{E}W_j W_{j'} \langle \mathbf{a}_j, \mathbf{a}_{j'}\rangle\bigg) \\
&= \frac{1}{m^2}\bigg(\frac{m}{N}\sum_{j=1}^{N} \|\mathbf{a}_j\|_2^2 + \frac{m(m-1)}{N(N-1)}\sum_{j\neq j'} \langle \mathbf{a}_j, \mathbf{a}_{j'}\rangle\bigg) \\
&= \frac{1}{m^2}\Bigg(\bigg(\frac{m}{N} - \frac{m(m-1)}{N(N-1)}\bigg)\sum_{j=1}^{N} \|\mathbf{a}_j\|_2^2 + \frac{m(m-1)}{N(N-1)}\bigg\|\sum_{j=1}^{N} \mathbf{a}_j\bigg\|_2^2\Bigg) \\
&= \frac{1}{m^2}\bigg(\frac{m}{N} - \frac{m(m-1)}{N(N-1)}\bigg)\sum_{j=1}^{N} \|\mathbf{a}_j\|_2^2 \\
&\leq \frac{1}{m}\cdot\frac{1}{N}\sum_{j=1}^{N} \|\mathbf{a}_j\|_2^2.
\end{aligned}$$

This completes the proof. □

2.11 Proofs of Auxiliary Lemmas

In this section, we present the additional proofs of supporting lemmas used in Appendix 2.10. Let $M, \{T_l\}, \{B_l\}$ and B_0 be the parameters defined in Algorithm 1. Let $\{\mathbf{x}_t^{(l)}\}, \{\mathbf{g}_t^{(l)}\}$ be the reference points and reference gradients used in Algorithm 1. Finally, $\mathbf{v}_t^{(l)}, \mathcal{F}_t$ are the variables and filtration defined in Appendix 2.9 and $c_j^{(s)}$ are the constant series defined in Definition 2.9.1.

2.11.1 Proof of Proposition 2.10.1

Proof of Proposition 2.10.1. By the definition of reference point $\mathbf{x}_t^{(s)}$ in (2.4.1), we can easily verify that (2.10.1) holds trivially.

Next we prove (2.10.2). Note that by (2.10.1) we have $\mathbf{x}_t^{(s)} = \mathbf{x}_{t'}^{(s)}$. For any $0 \le s' \le s$, it is also true that $\mathbf{x}_t^{(s')} = \mathbf{x}_{t'}^{(s')}$ by (2.4.1), which means $\mathbf{x}_t$ and $\mathbf{x}_{t'}$ share the same first $s+1$ reference points. Then by the update rule of $\mathbf{g}_t^{(s')}$ in Algorithm 1, we will maintain $\mathbf{g}_t^{(s')}$ unchanged from time step t to t'. In other worlds, we have $\mathbf{g}_t^{(s')} = \mathbf{g}_{t'}^{(s')}$ for all $0 \le s' \le s$.

We now prove the last claim (2.10.3). Based on (2.9.1) and (2.10.2), we have $\mathbf{v}_t^{(s)} = \sum_{s'=0}^{s} \mathbf{g}_t^{(s')} = \sum_{s'=0}^{s} \mathbf{g}_{p\cdot\prod_{j=s+1}^K T_j}^{(s')} = \mathbf{v}_{p\cdot\prod_{j=s+1}^K T_j}^{(s)}$. Since for any $s \le s'' \le K$, we have the following equations by the update in Algorithm 1 (Line 14).

$$
\begin{aligned}
\mathbf{x}_{p\cdot\prod_{j=s+1}^K T_j}^{(s'')} &= \mathbf{x}_{\lfloor p\cdot\prod_{j=s+1}^K T_j/\prod_{j=s''+1}^K T_j\rfloor\cdot\prod_{j=s''+1}^K T_j} \\
&= \mathbf{x}_{p\cdot\prod_{j=s+1}^K T_j/\prod_{j=s''+1}^K T_j\cdot\prod_{j=s''+1}^K T_j} \\
&= \mathbf{x}_{p\cdot\prod_{j=s+1}^K T_j}^{(s)}.
\end{aligned}
$$

Then for any $s < s'' \le K$, we have

$$
\mathbf{g}_{p\cdot\prod_{j=s+1}^K T_j}^{(s'')} = \frac{1}{B_{s''}} \sum_{i\in I} \Big[\nabla f_i\Big(\mathbf{x}_{p\cdot\prod_{j=s+1}^K T_j}^{(s'')}\Big) - \nabla f_i\Big(\mathbf{x}_{p\cdot\prod_{j=s+1}^K T_j}^{(s''-1)}\Big)\Big] = 0. \tag{2.11.1}
$$

Thus, we have

$$
\mathbf{v}_{p\cdot\prod_{j=s+1}^K T_j} = \sum_{s''=0}^{K} \mathbf{g}_{p\cdot\prod_{j=s+1}^K T_j}^{(s'')} = \sum_{s''=0}^{s} \mathbf{g}_{p\cdot\prod_{j=s+1}^K T_j}^{(s'')} = \sum_{s''=0}^{s} \mathbf{g}_t^{(s'')} = \mathbf{v}_t^{(s)}, \tag{2.11.2}
$$

where the first equality holds because of the definition of $\mathbf{v}_{p\cdot\prod_{j=s+1}^K T_j}$, the second equality holds due to (2.11.1) , the third equality holds due to (2.10.2) and the last equality holds due to (2.9.1). This completes the proof of (2.10.3). $\square$

2.11.2 Proof of Lemma 2.10.2

Proof of Lemma 2.10.2. For any fixed s, it can be seen that from the definition in (2.9.3), $c_j^{(s)}$ is monotonically decreasing with j. In order to prove (2.10.4), we only need to compare

$(1+T_{s-1})\cdot c_0^{(s-1)}$ and $c_{T_s}^{(s)}$. Furthermore, by the definition of series $\{c_j^{(s)}\}$ in (2.9.3), it can be inducted that when $0 \le j \le T_{s-1}$,

$$c_j^{(s-1)} = \left(1+\frac{1}{T_{s-1}}\right)^{T_{s-1}-j} \cdot c_{T_{s-1}}^{(s-1)} + \frac{(1+1/T_{s-1})^{T_{s-1}-j}-1}{1/T_{s-1}} \cdot \frac{3L^2}{M} \cdot \frac{\prod_{l=s}^{K} T_l}{B_{s-1}}. \tag{2.11.3}$$

We take $j=0$ in (2.11.3) and obtain

$$\begin{aligned} c_0^{(s-1)} &= \left(1+\frac{1}{T_{s-1}}\right)^{T_{s-1}} \cdot c_{T_{s-1}}^{(s-1)} + \frac{(1+1/T_{s-1})^{T_{s-1}}-1}{1/T_{s-1}} \cdot \frac{3L^2}{M} \cdot \frac{\prod_{l=s}^{K} T_l}{B_{s-1}} \\ &< 2.8 \times c_{T_{s-1}}^{(s-1)} + \frac{6L^2}{M} \cdot \frac{\prod_{l=s-1}^{K} T_l}{B_{s-1}} && (2.11.4) \\ &\le \frac{2.8M + 6L^2/M}{6^{K-s+2} \cdot \prod_{l=s-1}^{K} T_l} && (2.11.5) \\ &< \frac{3M}{6^{K-s+2} \cdot \prod_{l=s-1}^{K} T_l}, && (2.11.6) \end{aligned}$$

where (2.11.4) holds because $(1+1/n)^n < 2.8$ for any $n \ge 1$, (2.11.5) holds due to the definition of $c_{T_{s-1}}^{(s-1)}$ in (2.9.2) and $B_{s-1} \ge 6^{K-s+2}(\prod_{l=s-1}^{K} T_l)^2$ and (2.11.6) holds because $M \ge 6L$. Recall that $c_j^{(s)}$ is monotonically decreasing with j and the inequality in (2.11.6). Thus for all $2 \le s \le K$ and $0 \le j \le T_{s-1}$, we have

$$\begin{aligned} (1+T_{s-1}) \cdot c_j^{(s-1)} &\le (1+T_{s-1}) \cdot c_0^{(s-1)} \\ &\le (1+T_{s-1}) \cdot \frac{3M}{6^{K-s+2} \cdot \prod_{l=s-1}^{K} T_l} \\ &< \frac{6M}{6^{K-s+2} \cdot \prod_{l=s}^{K} T_l} \\ &= c_{T_s}^{(s)}, && (2.11.7) \end{aligned}$$

where the third inequality holds because $(1+T_{s-1})/T_{s-1} \le 2$ when $T_{s-1} \ge 1$ and the last equation comes from the definition of $c_{T_s}^{s}$ in (2.9.2). This completes the proof of (2.10.4).

Using similar techniques, we can obtain the upper bound for c_0^K which is similar to inequality (2.11.6) with $s-1$ replaced by K. Therefore, we have

$$(1+T_K) \cdot c_j^{(K)} \le (1+T_K) \cdot c_0^{(K)} < \frac{6M}{6^{K-K+1} \cdot \prod_{l=K}^{K} T_l} \le M,$$

which completes the proof of (2.10.5). □

2.11.3 Proof of Lemma 2.10.3

Now we prove Lemma 2.10.3, which is a special case of Lemma 2.9.2 when we choose $s = K$.

Proof of Lemma 2.10.3. To simplify notations, we use $\mathbb{E}[\cdot]$ to denote the conditional expectation $\mathbb{E}[\cdot|\mathcal{F}_{p\cdot T_K}]$ in the rest of this proof. For $pT_K \le pT_K + j < \min\{(p+1)T_K, T\}$, we denote $\mathbf{h}_{p\cdot T_K+j} = -(10M)^{-1} \cdot \mathbf{v}_{p\cdot T_K+j}$. According to the update in Algorithm 1 (Line 9), we have

$$\mathbf{x}_{p\cdot T_K+j+1} = \mathbf{x}_{p\cdot T_K+j} + \mathbf{h}_{p\cdot T_K+j}, \tag{2.11.8}$$

which immediately implies

$$\begin{aligned}
&F(\mathbf{x}_{p\cdot T_K+j+1}) \\
&= F(\mathbf{x}_{p\cdot T_K+j} + \mathbf{h}_{p\cdot T_K+j}) \\
&\le F(\mathbf{x}_{p\cdot T_K+j}) + \langle \nabla F(\mathbf{x}_{p\cdot T_K+j}), \mathbf{h}_{p\cdot T_K+j}\rangle + \frac{L}{2}\|\mathbf{h}_{p\cdot T_K+j}\|_2^2 \qquad (2.11.9) \\
&= \big[\langle \mathbf{v}_{p\cdot T_K+j}, \mathbf{h}_{p\cdot T_K+j}\rangle + 5M\|\mathbf{h}_{p\cdot T_K+j}\|_2^2\big] + F(\mathbf{x}_{p\cdot T_K+j}) \\
&\qquad + \langle \nabla F(\mathbf{x}_{p\cdot T_K+j}) - \mathbf{v}_{p\cdot T_K+j}, \mathbf{h}_{p\cdot T_K+j}\rangle + \bigg(\frac{L}{2} - 5M\bigg)\|\mathbf{h}_{p\cdot T_K+j}\|_2^2 \\
&\le F(\mathbf{x}_{p\cdot T_K+j}) + \langle \nabla F(\mathbf{x}_{p\cdot T_K+j}) - \mathbf{v}_{p\cdot T_K+j}, \mathbf{h}_{p\cdot T_K+j}\rangle + (L - 5M)\|\mathbf{h}_{p\cdot T_K+j}\|_2^2, \qquad (2.11.10)
\end{aligned}$$

where (2.11.9) is due to the L-smoothness of F and (2.11.10) holds because $\langle \mathbf{v}_{p\cdot T_K+j}, \mathbf{h}_{p\cdot T_K+j}\rangle + 5M\|\mathbf{h}_{p\cdot T_K+j}\|_2^2 = -5M\|\mathbf{h}_{p\cdot T_K+j}\|_2^2 \le 0$. Further by Young's inequality, we obtain

$$\begin{aligned}
F(\mathbf{x}_{p\cdot T_K+j+1}) &\le F(\mathbf{x}_{p\cdot T_K+j}) + \frac{1}{2M}\|\nabla F(\mathbf{x}_{p\cdot T_K+j}) - \mathbf{v}_{p\cdot T_K+j}\|_2^2 + \bigg(\frac{M}{2} + L - 5M\bigg)\|\mathbf{h}_{p\cdot T_K+j}\|_2^2 \\
&\le F(\mathbf{x}_{p\cdot T_K+j}) + \frac{1}{M}\|\nabla F(\mathbf{x}_{p\cdot T_K+j}) - \mathbf{v}_{p\cdot T_K+j}\|_2^2 - 3M\|\mathbf{h}_{p\cdot T_K+j}\|_2^2, \qquad (2.11.11)
\end{aligned}$$

where the second inequality holds because $M > L$. Now we bound the term $c_{j+1}^{(K)}\|\mathbf{x}_{p\cdot T_K+j+1} - \mathbf{x}_{p\cdot T_K}\|_2^2$. By (2.11.8) we have

$$\begin{aligned}
&c_{j+1}^{(K)}\|\mathbf{x}_{p\cdot T_K+j+1} - \mathbf{x}_{p\cdot T_K}\|_2^2 \\
&= c_{j+1}^{(K)}\|\mathbf{x}_{p\cdot T_K+j} - \mathbf{x}_{p\cdot T_K} + \mathbf{h}_{p\cdot T_K+j}\|_2^2 \\
&= c_{j+1}^{(K)}\big[\|\mathbf{x}_{p\cdot T_K+j} - \mathbf{x}_{p\cdot T_K}\|_2^2 + \|\mathbf{h}_{p\cdot T_K+j}\|_2^2 + 2\langle \mathbf{x}_{p\cdot T_K+j} - \mathbf{x}_{p\cdot T_K}, \mathbf{h}_{p\cdot T_K+j}\rangle\big].
\end{aligned}$$

Applying Young's inequality yields

$$\begin{aligned}
&c_{j+1}^{(K)}\|\mathbf{x}_{p\cdot T_K+j+1}-\mathbf{x}_{p\cdot T_K}\|_2^2\\
&\le c_{j+1}^{(K)}\bigg[\|\mathbf{x}_{p\cdot T_K+j}-\mathbf{x}_{p\cdot T_K}\|_2^2+\|\mathbf{h}_{p\cdot T_K+j}\|_2^2\\
&\qquad+\frac{1}{T_K}\|\mathbf{x}_{p\cdot T_K+j}-\mathbf{x}_{p\cdot T_K}\|_2^2+T_K\|\mathbf{h}_{p\cdot T_K+j}\|_2^2\bigg]\\
&=c_{j+1}^{(K)}\bigg[\bigg(1+\frac{1}{T_K}\bigg)\|\mathbf{x}_{p\cdot T_K+j}-\mathbf{x}_{p\cdot T_K}\|_2^2+(1+T_K)\|\mathbf{h}_{p\cdot T_K+j}\|_2^2\bigg], \qquad (2.11.12)
\end{aligned}$$

Adding up inequalities (2.11.12) and (2.11.11), we get

$$\begin{aligned}
&F(\mathbf{x}_{p\cdot T_K+j+1})+c_{j+1}^{(K)}\|\mathbf{x}_{p\cdot T_K+j+1}-\mathbf{x}_{p\cdot T_K}\|_2^2\\
&\le F(\mathbf{x}_{p\cdot T_K+j})+\frac{1}{M}\|\nabla F(\mathbf{x}_{p\cdot T_K+j})-\mathbf{v}_{p\cdot T_K+j}\|_2^2-\big[3M-c_{j+1}^{(K)}(1+T_K)\big]\|\mathbf{h}_{p\cdot T_K+j}\|_2^2\\
&\qquad+c_{j+1}^{(K)}\bigg(1+\frac{1}{T_K}\bigg)\|\mathbf{x}_{p\cdot T_K+j}-\mathbf{x}_{p\cdot T_K}\|_2^2\\
&\le F(\mathbf{x}_{p\cdot T_K+j})+\frac{1}{M}\|\nabla F(\mathbf{x}_{p\cdot T_K+j})-\mathbf{v}_{p\cdot T_K+j}\|_2^2-2M\|\mathbf{h}_{p\cdot T_K+j}\|_2^2\\
&\qquad+c_{j+1}^{(K)}\bigg(1+\frac{1}{T_K}\bigg)\|\mathbf{x}_{p\cdot T_K+j}-\mathbf{x}_{p\cdot T_K}\|_2^2, \qquad (2.11.13)
\end{aligned}$$

where the last inequality holds due to the fact that $c_{j+1}^{(K)}(1+T_K)<M$ by Lemma 2.10.2. Next we bound $\|\nabla F(\mathbf{x}_{p\cdot T_K+j})\|_2^2$ with $\|\mathbf{h}_{p\cdot T_K+j}\|_2^2$. Note that by (2.11.8)

$$\begin{aligned}
\|\nabla F(\mathbf{x}_{p\cdot T_K+j})\|_2^2&=\big\|\big[\nabla F(\mathbf{x}_{p\cdot T_K+j})-\mathbf{v}_{p\cdot T_K+j}\big]-10M\mathbf{h}_{p\cdot T_K+j}\big\|_2^2\\
&\le 2\big(\|\nabla F(\mathbf{x}_{p\cdot T_K+j})-\mathbf{v}_{p\cdot T_K+j}\|_2^2+100M^2\|\mathbf{h}_{p\cdot T_K+j}\|_2^2\big),
\end{aligned}$$

which immediately implies

$$-2M\|\mathbf{h}_{p\cdot T_K+j}\|_2^2\le\frac{2}{100M}\big(\|\nabla F(\mathbf{x}_{p\cdot T_K+j})-\mathbf{v}_{p\cdot T_K+j}\|_2^2-\frac{1}{100M}\|\nabla F(\mathbf{x}_{p\cdot T_K+j})\|_2^2. \qquad (2.11.14)$$

Plugging (2.11.14) into (2.11.13), we have

$$\begin{aligned}
&F(\mathbf{x}_{p\cdot T_K+j+1})+c_{j+1}^{(K)}\|\mathbf{x}_{p\cdot T_K+j+1}-\mathbf{x}_{p\cdot T_K}\|_2^2\\
&\le F(\mathbf{x}_{p\cdot T_K+j})+\frac{1}{M}\|\nabla F(\mathbf{x}_{p\cdot T_K+j})-\mathbf{v}_{p\cdot T_K+j}\|_2^2+\frac{1}{50M}\cdot\|\nabla F(\mathbf{x}_{p\cdot T_K+j})-\mathbf{v}_{p\cdot T_K+j}\|_2^2\\
&\qquad-\frac{1}{100M}\|\nabla F(\mathbf{x}_{p\cdot T_K+j})\|_2^2+c_{j+1}^{(K)}\bigg(1+\frac{1}{T_K}\bigg)\|\mathbf{x}_{p\cdot T_K+j}-\mathbf{x}_{p\cdot T_K}\|_2^2
\end{aligned}$$

$$\leq F(\mathbf{x}_{p\cdot T_K+j}) + \frac{2}{M}\|\nabla F(\mathbf{x}_{p\cdot T_K+j}) - \mathbf{v}_{p\cdot T_K+j}\|_2^2 - \frac{1}{100M}\|\nabla F(\mathbf{x}_{p\cdot T_K+j})\|_2^2 + c_{j+1}^{(K)}\bigg(1+\frac{1}{T_K}\bigg)\|\mathbf{x}_{p\cdot T_K+j} - \mathbf{x}_{p\cdot T_K}\|_2^2. \tag{2.11.15}$$

Next we bound $\|\nabla F(\mathbf{x}_{p\cdot T_K+j}) - \mathbf{v}_{p\cdot T_K+j}\|_2^2$. First, by Lemma 2.10.4 we have

$$\mathbb{E}\Big\|\nabla F(\mathbf{x}^{(K)}_{p\cdot T_K+j}) - \mathbf{v}^{(K)}_{p\cdot T_K+j}\Big\|_2^2 \leq \frac{L^2}{B_K}\mathbb{E}\Big\|\mathbf{x}^{(K)}_{p\cdot T_K+j} - \mathbf{x}^{(K-1)}_{p\cdot T_K+j}\Big\|_2^2 + \mathbb{E}\Big\|\nabla F(\mathbf{x}^{(K-1)}_{p\cdot T_K+j}) - \mathbf{v}^{(K-1)}_{p\cdot T_K+j}\Big\|_2^2.$$

Since $\mathbf{x}^{(K)}_{p\cdot T_K+j} = \mathbf{x}_{p\cdot T_K+j}, \mathbf{v}^{(K)}_{p\cdot T_K+j} = \mathbf{v}_{p\cdot T_K+j}$, $\mathbf{x}^{(K-1)}_{p\cdot T_K+j} = \mathbf{x}_{p\cdot T_K}$ and $\mathbf{v}^{(K-1)}_{p\cdot T_K+j} = \mathbf{v}_{p\cdot T_K}$, we have

$$\mathbb{E}\|\nabla F(\mathbf{x}_{p\cdot T_K+j}) - \mathbf{v}_{p\cdot T_K+j}\|_2^2 \leq \frac{L^2}{B_K}\mathbb{E}\|\mathbf{x}_{p\cdot T_K+j} - \mathbf{x}_{p\cdot T_K}\|_2^2 + \mathbb{E}\|\nabla F(\mathbf{x}_{p\cdot T_K}) - \mathbf{v}_{p\cdot T_K}\|_2^2. \tag{2.11.16}$$

Taking expectation $\mathbb{E}[\cdot]$ with (2.11.15) and plugging (2.11.16) into (2.11.15) , we obtain

$$\begin{aligned}
&\mathbb{E}\bigg[F(\mathbf{x}_{p\cdot T_K+j+1}) + c_{j+1}^{(K)}\|\mathbf{x}_{p\cdot T_K+j+1} - \mathbf{x}_{p\cdot T_K}\|_2^2 + \frac{1}{100M}\|\nabla F(\mathbf{x}_{p\cdot T_K+j})\|_2^2\bigg] \\
&\leq \mathbb{E}\bigg[F(\mathbf{x}_{p\cdot T_K+j}) + \bigg(c_{j+1}^{(K)}\bigg(1+\frac{1}{T_K}\bigg) + \frac{3L^2}{B_K M}\bigg)\|\mathbf{x}_{p\cdot T_K+j} - \mathbf{x}_{p\cdot T_K}\|_2^2 \\
&\quad + \frac{2}{M}\|\nabla F(\mathbf{x}_{p\cdot T_K}) - \mathbf{v}_{p\cdot T_K}\|_2^2\bigg] \\
&= \mathbb{E}\bigg[F(\mathbf{x}_{p\cdot T_K+j}) + c_j^{(K)}\|\mathbf{x}_{p\cdot T_K+j} - \mathbf{x}_{p\cdot T_K}\|_2^2 + \frac{2}{M}\cdot\|\nabla F(\mathbf{x}_{p\cdot T_K}) - \mathbf{v}_{p\cdot T_K}\|_2^2\bigg],
\end{aligned} \tag{2.11.17}$$

where (2.11.17) holds because we have $c_j^{(K)} = c_{j+1}^{(K)}(1+1/T_K) + 3L^2/(B_K M)$ by Definition 2.9.1. Telescoping (2.11.17) for $j = 0$ to end − start − 1, we have

$$\begin{aligned}
&\mathbb{E}\big[F\big(\mathbf{x}_{\text{end}}\big) + c_{T_K}^{(K)}\cdot\|\mathbf{x}_{\text{end}} - \mathbf{x}_{\text{start}}\|_2^2\big] + \frac{1}{100M}\sum_{j=\text{start}}^{\text{end}-1}\mathbb{E}\|\nabla F(\mathbf{x}_j)\|_2^2 \\
&\leq \mathbb{E}\big[F\big(\mathbf{x}_{\text{end}}\big) + c_{\text{end}-\text{start}}^{(K)}\cdot\|\mathbf{x}_{\text{end}} - \mathbf{x}_{\text{start}}\|_2^2\big] + \frac{1}{100M}\sum_{j=\text{start}}^{\text{end}-1}\mathbb{E}\|\nabla F(\mathbf{x}_j)\|_2^2 \\
&\leq F(\mathbf{x}_{\text{start}}) + \frac{2(\text{end}-\text{start})}{M}\cdot\mathbb{E}\|\nabla F(\mathbf{x}_{\text{start}}) - \mathbf{v}_{\text{start}}\|_2^2,
\end{aligned}$$

which completes the proof. □

2.11.4 Proof of Lemma 2.10.4

Proof of Lemma 2.10.4. If $t^l = t^{l-1}$, we have $\mathbf{x}_t^{(l)} = \mathbf{x}_t^{(l-1)}$ and $\mathbf{v}_t^{(l)} = \mathbf{v}_t^{(l-1)}$. In this case the statement in Lemma 2.10.4 holds trivially. Therefore, we assume $t^l \neq t^{l-1}$ in the following proof. Note that

$$
\begin{aligned}
&\mathbb{E}\big[\big\|\nabla F(\mathbf{x}_t^{(l)}) - \mathbf{v}_t^{(l)}\big\|_2^2|\mathcal{F}_{t^l}\big] \\
&= \mathbb{E}\big[\big\|\nabla F(\mathbf{x}_t^{(l)}) - \mathbf{v}_t^{(l)} - \mathbb{E}\big[\nabla F(\mathbf{x}_t^{(l)}) - \mathbf{v}_t^{(l)}\big]\big\|_2^2|\mathcal{F}_{t^l}\big] + \big\|\mathbb{E}\big[\nabla F(\mathbf{x}_t^{(l)}) - \mathbf{v}_t^{(l)}|\mathcal{F}_{t^l}\big]\big\|_2^2 \\
&= \underbrace{\mathbb{E}\Bigg[\Bigg\|\nabla F(\mathbf{x}_t^{(l)}) - \sum_{j=0}^{l}\mathbf{g}_t^{(j)} - \mathbb{E}\Bigg[\nabla F(\mathbf{x}_t^{(l)}) - \sum_{j=0}^{l}\mathbf{g}_t^{(j)}\Bigg]\Bigg\|_2^2\Bigg|\mathcal{F}_{t^l}\Bigg]}_{J_1} \\
&\qquad + \underbrace{\Bigg\|\mathbb{E}\Bigg[\nabla F(\mathbf{x}_t^{(l)}) - \sum_{j=0}^{l}\mathbf{g}_t^{(j)}\Bigg|\mathcal{F}_{t^l}\Bigg]\Bigg\|_2^2}_{J_2}, \qquad (2.11.18)
\end{aligned}
$$

where in the second equation we used the definition $\mathbf{v}_t^{(l)} = \sum_{i=0}^{l}\mathbf{g}_t^{(i)}$ in (2.9.1). We first upper bound term J_1. According to the update rule in Algorithm 1 (Line 20-23), when $j < l$, $\mathbf{g}_t^{(j)}$ will not be updated at the t^l-th iteration. Thus we have $\mathbb{E}[\mathbf{g}_t^{(j)}|\mathcal{F}_{t^l}] = \mathbf{g}_t^{(j)}$ for all $j < l$. In addition, by the definition of $\mathcal{F}_{t^l}$, we have $\mathbb{E}[\nabla F(\mathbf{x}_t^{(l)})|\mathcal{F}_{t^l}] = \nabla F(\mathbf{x}_t^{(l)})$. Then we have the following equation

$$
J_1 = \mathbb{E}\big[\big\|\mathbf{g}_t^{(l)} - \mathbb{E}\big[\mathbf{g}_t^{(l)}|\mathcal{F}_{t^l}\big]\big\|_2^2|\mathcal{F}_{t^l}\big]. \qquad (2.11.19)
$$

We further have

$$
\mathbf{g}_t^{(l)} = \frac{1}{B_l}\sum_{i\in I}\big[\nabla f_i(\mathbf{x}_t^{(l)}) - \nabla f_i(\mathbf{x}_t^{(l-1)})\big], \quad \mathbb{E}\big[\mathbf{g}_t^{(l)}|\mathcal{F}_{t^l}\big] = \nabla F(\mathbf{x}_t^{(l)}) - \nabla F(\mathbf{x}_t^{(l-1)}).
$$

Therefore, we can apply Lemma 2.9.3 to (2.11.19) and obtain

$$
\begin{aligned}
J_1 &\le \frac{1}{B_l}\cdot\frac{1}{n}\sum_{i=1}^{n}\big\|\nabla f_i(\mathbf{x}_t^{(l)}) - \nabla f_i(\mathbf{x}_t^{(l-1)}) - \big[\nabla F(\mathbf{x}_t^{(l)}) - \nabla F(\mathbf{x}_t^{(l-1)})\big]\big\|_2^2 \\
&\le \frac{1}{B_l n}\sum_{i=1}^{n}\big\|\nabla f_i(\mathbf{x}_t^{(l)}) - \nabla f_i(\mathbf{x}_t^{(l-1)})\big\|_2^2 \\
&\le \frac{L^2}{B_l}\big\|\mathbf{x}_t^{(l)} - \mathbf{x}_t^{(l-1)}\big\|_2^2, \qquad (2.11.20)
\end{aligned}
$$

where the second inequality is due to the fact that $\mathbb{E}[\|\boldsymbol{X} - \mathbb{E}[\boldsymbol{X}]\|_2^2] \le \mathbb{E}\|\boldsymbol{X}\|_2^2$ for any random vector $\boldsymbol{X}$ and the last inequality holds due to the fact that F has averaged L-Lipschitz gradient.

Next we turn to bound term J_2. Note that

$$\mathbb{E}\big[\mathbf{g}_t^{(l)}\big|\mathcal{F}_{t^l}\big] = \mathbb{E}\bigg[\frac{1}{B_l}\sum_{i\in I}\big[\nabla f_i(\mathbf{x}_t^{(l)}) - \nabla f_i(\mathbf{x}_t^{(l-1)})\big]\bigg|\mathcal{F}_{t^l}\bigg] = \nabla F(\mathbf{x}_t^{(l)}) - \nabla F(\mathbf{x}_t^{(l-1)}),$$

which immediately implies

$$\begin{aligned}\mathbb{E}\bigg[\nabla F(\mathbf{x}_t^{(l)}) - \sum_{j=0}^{l}\mathbf{g}_t^{(j)}\bigg|\mathcal{F}_{t^l}\bigg] &= \mathbb{E}\bigg[\nabla F(\mathbf{x}_t^{(l)}) - \nabla F(\mathbf{x}_t^{(l)}) + \nabla F(\mathbf{x}_t^{(l-1)}) - \sum_{j=0}^{l-1}\mathbf{g}_t^{(j)}\bigg|\mathcal{F}_{t^l}\bigg]\\ &= \mathbb{E}\big[\nabla F(\mathbf{x}_t^{(l-1)}) - \mathbf{v}_t^{(l-1)}\big|\mathcal{F}_{t^l}\big]\\ &= \nabla F(\mathbf{x}_t^{(l-1)}) - \mathbf{v}_t^{(l-1)},\end{aligned}$$

where the last equation is due to the definition of $\mathcal{F}_t$. Plugging J_1 and J_2 into (2.11.18) yields the following result:

$$\mathbb{E}\big[\big\|\nabla F(\mathbf{x}_t^{(l)}) - \mathbf{v}_t^{(l)}\big\|_2^2\big|\mathcal{F}_{t^l}\big] \le \frac{L^2}{B_l}\big\|\mathbf{x}_t^{(l)} - \mathbf{x}_t^{(l-1)}\big\|_2^2 + \big\|\nabla F(\mathbf{x}_t^{(l-1)}) - \mathbf{v}_t^{(l-1)}\big\|_2^2,$$

which completes the proof. □

CHAPTER 3

Global Convergence of Langevin Dynamics Based Algorithms

3.1 Introduction

In this chapter, we aim to find the global optimum of the finite-sum nonconvex optimization problem defined in (1.1.1). Recent studies by [Dal17b, Dal17a] showed that sampling from a distribution which concentrates around the global minimum of $F(\mathbf{x})$ is a similar task as minimizing F via certain optimization algorithms. This justifies the use of Langevin dynamics based algorithms for optimization. In detail, the first order Langevin dynamics is defined by the following stochastic differential equation (SDE)

$$d\boldsymbol{X}(t) = -\nabla F(\boldsymbol{X}(t))dt + \sqrt{2\beta^{-1}}d\boldsymbol{B}(t), \tag{3.1.1}$$

where $\beta > 0$ is the inverse temperature parameter that is treated as a constant throughout the analysis of this chapter, and $\{\boldsymbol{B}(t)\}_{t\geq 0}$ is the standard Brownian motion in $\mathbb{R}^d$. Under certain assumptions on the drift coefficient ∇F, it was showed that the distribution of diffusion $\boldsymbol{X}(t)$ in (3.1.1) converges to its stationary distribution [CHS87], a.k.a., the Gibbs measure $\pi(d\mathbf{x}) \propto \exp(-\beta F(\mathbf{x}))$, which concentrates on the global minimum of F [Hwa80, GM91, RT96]. Note that the above convergence result holds even when $F(\mathbf{x})$ is nonconvex. This motivates the use of Langevin dynamics based algorithms for nonconvex optimization [RRT17, ZLC17, TLR18, SYN$^+$18]. However, unlike first order optimization algorithms [Nes13, GL13, RHS$^+$16, AZH16], which have been extensively studied, the non-asymptotic theoretical guarantee of applying Langevin dynamics based algorithms for nonconvex optimization, is still under studied. In a seminal work, [RRT17] provided a non-asymptotic analysis of stochastic gradient Langevin dynamics (SGLD) [WT11] for nonconvex optimization, which is a stochastic gradient based discretization of (3.1.1). They proved that SGLD converges to an almost minimizer[1] up to $d^2/(\sigma^{1/4}\lambda^*)\log(1/\epsilon)$ within $\tilde{O}(d/(\lambda^*\epsilon^4))$ iterations, where σ^2 is the variance of stochastic gradient and λ^* is called the *uniform spectral gap* of Langevin diffusion (3.1.1), and it is in the order of $e^{-\tilde{O}(d)}$. In a concurrent work, [ZLC17] analyzed the hitting time of SGLD and proved its convergence to an approximate local minimum. More recently, [TLR18] studied the local optimality and generalization performance of Langevin algorithm for nonconvex functions through the lens of metastability and [SYN$^+$18] developed an asynchronous-parallel stochastic L-BFGS algorithm for nonconvex optimization based on variants of SGLD. [EMS18] further developed non-asymptotic analysis of global optimization based on a broader class of diffusions.

In this chapter, we establish the global convergence for a family of Langevin dynamics based algorithms, including Gradient Langevin Dynamics (GLD) [Dal17b, DM15, Dal17a], Stochastic Gradient Langevin Dynamics (SGLD) [WT11] and Stochastic Variance Reduced Gradient Langevin Dynamics (SVRG-LD) [DRW$^+$16] for solving the finite sum nonconvex optimization problem in (1.1.1). Our analysis is built upon the direct analysis of the discrete-time Markov chain rather than the continuous-time Langevin diffusion, and therefore avoid the discretization error.

3.1.1 Our Contributions

The major contributions of our work are summarized as follows:

- We provide a unified analysis for a family of Langevin dynamics based algorithms by a new decomposition scheme of the optimization error, under which we directly analyze the ergodicity of numerical approximations for Langevin dynamics (see Figure 3.1).
- Under our unified framework, we establish the global convergence of GLD for solving

[1]Following [RRT17], an almost minimizer is defined to be a point which is within the ball of the global minimizer with radius $O(d\log(\beta+1)/\beta)$, where d is the problem dimension and β is the inverse temperature parameter.

(1.1.1). In detail, GLD requires $\tilde{O}\big(d/(\lambda\epsilon)\big)$ iterations to converge to the almost minimizer of (1.1.1) up to precision ϵ, where λ is the spectral gap of the discrete-time Markov chain generated by GLD and is in the order of $e^{-\tilde{O}(d)}$. This improves the $\tilde{O}\big(d/(\lambda^*\epsilon^4)\big)$ iteration complexity of GLD implied by [RRT17], where $\lambda^* = e^{-\tilde{O}(d)}$ is the spectral gap of Langevin diffusion (3.1.1).

- We establish a faster convergence of SGLD to the almost minimizer of (1.1.1). In detail, it converges to the almost minimizer up to ϵ precision within $\tilde{O}\big(d^7/(\lambda^5\epsilon^5)\big)$ stochastic gradient evaluations. This also improves the $\tilde{O}\big(d^{17}/(\lambda^{*8}\epsilon^8)\big)$ gradient complexity proved in [RRT17].

- We also analyze the SVRG-LD algorithm and investigate its global convergence property. We show that SVRG-LD is guaranteed to converge to the almost minimizer of (1.1.1) within $\tilde{O}\big(\sqrt{n}d^5/(\lambda^4\epsilon^{5/2})\big)$ stochastic gradient evaluations. It outperforms the gradient complexities of both GLD and SGLD when $1/\epsilon^3 \le n \le 1/\epsilon^5$. To the best of our knowledge, this is the first global convergence guarantee of SVRG-LD for nonconvex optimization, while the original paper [DRW$^+$16] only analyzed the posterior sampling property of SVRG-LD.

3.1.2 Additional Related Work

Stochastic gradient Langevin dynamics (SGLD) [WT11] and its extensions [AKW12, MCF15, DRW$^+$16] have been widely used in Bayesian learning. A large body of work has focused on analyzing the mean square error of Langevin dynamics based algorithms. In particular, [VZT16] analyzed the non-asymptotic bias and variance of the SGLD algorithm by using Poisson equations. [CDC15] showed the non-asymptotic bias and variance of MCMC algorithms with high order integrators. [DRW$^+$16] proposed variance-reduced algorithms based on stochastic gradient Langevin dynamics, namely SVRG-LD and SAGA-LD, for Bayesian posterior inference, and proved that their method improves the mean square error upon SGLD. [LZL18] further improved the mean square error by applying the variance reduction tricks on Hamiltonian Monte Carlo, which is also called the underdamped Langevin dynamics.

Another line of research [Dal17b, DM16, Dal17a, DK17, DCWY18, ZXG18c] focused on characterizing the distance between distributions generated by Langevin dynamics based algorithms and (strongly) log-concave target distributions. In detail, [Dal17b] proved that the distribution of the last step in GLD converges to the stationary distribution in $\tilde{O}(d/\epsilon^2)$ iterations in terms of total variation distance and Wasserstein distance respectively with a warm start and showed the similarities between posterior sampling and optimization. Later [DM15] improved the results by showing this result holds for any starting point and established similar bounds for the Wasserstein distance. [Dal17a] further improved the existing results in terms of the Wasserstein distance and provide further insights on the close relation between approximate sampling and gradient descent. [CCBJ18] improved existing 2-Wasserstein results by reducing the discretization error using underdamped Langevin dynamics. To improve the convergence rates in noisy gradient settings, [CFM$^+$18, ZXG18d] presented convergence guarantees in 2-Wasserstein distance for SAGA-LD and SVRG-LD using variance reduction techniques. [ZXG18c] proposed the variance reduced Hamilton Monte Carlo to accelerate the convergence of Langevin dynamics based sampling algorithms. As to sampling from distribution with compact support, [BEL18] analyzed sampling from log-concave distributions via projected Langevin Monte Carlo, and [BDMP17] proposed a proximal Langevin Monte Carlo algorithm. This line of research is orthogonal to our work since their analyses are regarding to the convergence of the distribution of the iterates to the stationary distribution of Langevin diffusion in total variation distance or 2-Wasserstein distance instead of expected function value gap.

On the other hand, many attempts have been made to escape from saddle points in nonconvex optimization, such as cubic regularization [NP06, ZXG18b], trust region Newton method [CRS14], Hessian-vector product based methods [AAZB$^+$17, CD16, CDHS16], noisy gradient descent [GHJY15, JGN$^+$17, JNJ18] and normalized gradient [Lev16]. Yet all these algorithms are only guaranteed to converge to an approximate local minimum rather than a global minimum. The global convergence for nonconvex optimization remains understudied.

3.1.3 Preliminaries

In this section, we present some preliminaries for SDE. Note that throughout this chapter, we use lower case bold symbol $\mathbf{x}$ to denote deterministic vector, and use upper case italicized bold symbol $\boldsymbol{X}$ to denote random vector.

Kolmogorov Operator and Infinitesimal Generator

Suppose $\boldsymbol{X}(t)$ is the solution to the diffusion process represented by the stochastic differential equation (3.1.1). For such a continuous time Markov process, let $P = \{P_t\}_{t>0}$ be the corresponding Markov semi-group [BGL13], and we define the Kolmogorov operator [BGL13] P_s as follows

$$P_s g(\boldsymbol{X}(t)) = \mathbb{E}[g(\boldsymbol{X}(s+t))|\boldsymbol{X}(t)],$$

where g is a smooth test function. We have $P_{s+t} = P_s \circ P_t$ by Markov property. Further we define the infinitesimal generator [BGL13] of the semi-group $\mathcal{L}$ to describe the the movement of the process in an infinitesimal time interval:

$$\begin{aligned}\mathcal{L}g(\boldsymbol{X}(t)) :&= \lim_{h\to 0^+} \frac{\mathbb{E}[g(\boldsymbol{X}(t+h))|\boldsymbol{X}(t)] - g(\boldsymbol{X}(t))}{h} \\ &= \big(- \nabla F(\boldsymbol{X}(t)) \cdot \nabla + \beta^{-1}\nabla^2\big) g(\boldsymbol{X}(t)),\end{aligned}$$

where β is the inverse temperature parameter.

Poisson Equation and the Time Average

Poisson equations are widely used in the study of homogenization and ergodic theory to prove the desired limit of a time-average. Let $\mathcal{L}$ be the infinitesimal generator and let ψ be defined as follows

$$\mathcal{L}\psi = g - \bar{g}, \tag{3.1.2}$$

where g is a smooth test function and $\bar{g}$ is the expectation of g over the Gibbs measure, i.e., $\bar{g} := \int g(\mathbf{x})\pi(d\mathbf{x})$. Smooth function ψ is called the solution of Poisson equation (3.1.2). Importantly, it has been shown [EMS18] that the first and second order derivatives of the solution ψ of Poisson equation for Langevin diffusion can be bounded by polynomial growth functions.

3.2 Review of Langevin Dynamics Based Algorithms

Now we briefly review three popular Langevin dynamics based algorithms.

In practice, numerical methods (a.k.a., numerical integrators) are used to approximate the Langevin diffusion in (3.1.1). For example, by Euler-Maruyama scheme [KP92], (3.1.1) can be discretized as follows:

$$\boldsymbol{X}_{k+1} = \boldsymbol{X}_k - \eta \nabla F(\boldsymbol{X}_k) + \sqrt{2\eta\beta^{-1}} \cdot \boldsymbol{\epsilon}_k, \tag{3.2.1}$$

where $\boldsymbol{\epsilon}_k \in \mathbb{R}^d$ is standard Gaussian noise and $\eta > 0$ is the step size. The update in (3.2.1) resembles gradient descent update except for an additional injected Gaussian noise. The magnitude of the Gaussian noise is controlled by the inverse temperature parameter β. In our work, we refer this update as Gradient Langevin Dynamics (GLD) [Dal17b, DM15, Dal17a]. The details of GLD are shown in Algorithm 4.

In the case that n is large, the above Euler approximation can be infeasible due to the high computational cost of the full gradient $\nabla F(\boldsymbol{X}_k)$ at each iteration. A natural idea is to use stochastic gradient to approximate the full gradient, which gives rise to Stochastic Gradient Langevin Dynamics (SGLD) [WT11] and its variants [AKW12, MCF15, CDC15]. However, the high variance brought by the stochastic gradient can make the convergence of SGLD slow. To reduce the variance of the stochastic gradient and accelerate the convergence of SGLD, we use a mini-batch of stochastic gradients in the following update form:

$$\boldsymbol{Y}_{k+1} = \boldsymbol{Y}_k - \frac{\eta}{B} \sum_{i \in I_k} \nabla f_i(\boldsymbol{Y}_k) + \sqrt{2\eta\beta^{-1}} \cdot \boldsymbol{\epsilon}_k, \tag{3.2.2}$$

where $1/B \sum_{i \in I_k} \nabla f_i(\boldsymbol{Y}_k)$ is the stochastic gradient, which is an unbiased estimator for $\nabla F(\boldsymbol{Y}_k)$ and I_k is a subset of $\{1, \ldots, n\}$ with $|I_k| = B$. Algorithm 5 displays the details of SGLD.

Motivated by recent advances in stochastic optimization, in particular, the variance reduction based techniques [JZ13, RHS$^+$16, AZH16], [DRW$^+$16] proposed the Stochastic Variance Reduced Gradient Langevin Dynamics (SVRG-LD) for posterior sampling. The key idea is to use semi-stochastic gradient to reduce the variance of the stochastic gradient. SVRG-LD takes the following update form:

$$\boldsymbol{Z}_{k+1} = \boldsymbol{Z}_k - \eta\tilde{\nabla}_k + \sqrt{2\eta\beta^{-1}} \cdot \boldsymbol{\epsilon}_k, \tag{3.2.3}$$

where $\tilde{\nabla}_k = 1/B \sum_{i_k \in I_k} \big(\nabla f_{i_k}(\boldsymbol{Z}_k) - \nabla f_{i_k}(\tilde{\boldsymbol{Z}}^{(s)}) + \nabla F(\tilde{\boldsymbol{Z}}^{(s)})\big)$ is the semi-stochastic gradient, $\tilde{\boldsymbol{Z}}^{(s)}$ is a snapshot of $\boldsymbol{Z}_k$ at every L iteration such that $k = sL+\ell$ for some $\ell = 0, 1, \ldots, L-1$, and I_k is a subset of $\{1, \ldots, n\}$ with $|I_k| = B$. SVRG-LD is summarized in Algorithm 6.

Note that although all the three algorithms are originally proposed for posterior sampling or more generally, Bayesian learning, they can be applied for nonconvex optimization, as demonstrated in many previous studies [AKW12, RRT17, ZLC17].

Algorithm 4 Gradient Langevin Dynamics (GLD)

```
**input:** step size $\eta > 0$; inverse temperature parameter $\beta > 0$; $\boldsymbol{X}_0 = \mathbf{0}$
**for** $k = 0, 1, \ldots, K-1$ **do**
    randomly draw $\boldsymbol{\epsilon}_k \sim N(\mathbf{0}, \mathbf{I}_{d\times d})$
    $\boldsymbol{X}_{k+1} = \boldsymbol{X}_k - \eta\nabla F(\boldsymbol{X}_k) + \sqrt{2\eta/\beta}\boldsymbol{\epsilon}_k$
**end for**
```

Algorithm 5 Stochastic Gradient Langevin Dynamics (SGLD)

```
**input:** step size $\eta > 0$; batch size $B$; inverse temperature parameter $\beta > 0$; $\mathbf{Y}_0 = \mathbf{0}$
**for** $k = 0, 1, \ldots, K-1$ **do**
    randomly pick a subset $I_k$ from $\{1, \ldots, n\}$ of size $|I_k| = B$; randomly draw $\boldsymbol{\epsilon}_k \sim N(\mathbf{0}, \mathbf{I}_{d\times d})$
    $\mathbf{Y}_{k+1} = \mathbf{Y}_k - \eta/B \sum_{i\in I_k} \nabla f_i(\mathbf{Y}_k) + \sqrt{2\eta/\beta}\boldsymbol{\epsilon}_k$
**end for**
```

3.3 Main Theory

We first lay out the following assumption on the loss function.

Assumption 3.3.1 (Smoothness). *The function $f_i(\mathbf{x})$ is M-smooth for $M > 0$, $i = 1, \ldots, n$, where the smoothness condition is defined in Definition 2.3.1.*

Algorithm 6 Stochastic Variance Reduced Gradient Langevin Dynamics (SVRG-LD)

```
input: step size $\eta > 0$; batch size $B$; epoch length $L$; inverse temperature parameter $\beta > 0$
initialization: $\boldsymbol{Z}_0 = \mathbf{0}$, $\tilde{\boldsymbol{Z}}^{(0)} = \boldsymbol{Z}_0$
for $s = 0, 1, \ldots, (K/L) - 1$ do
  $\tilde{\boldsymbol{W}} = \nabla F(\tilde{\boldsymbol{Z}}^{(s)})$
  for $\ell = 0, \ldots, L-1$ do
    $k = sL + \ell$
    randomly pick a subset $I_k$ from $\{1, \ldots, n\}$ of size $|I_k| = B$; draw $\boldsymbol{\epsilon}_k \sim N(\mathbf{0}, \mathbf{I}_{d\times d})$
    $\tilde{\nabla}_k = 1/B \sum_{i_k \in I_k} \big(\nabla f_{i_k}(\boldsymbol{Z}_k) - \nabla f_{i_k}(\tilde{\boldsymbol{Z}}^{(s)}) + \tilde{\boldsymbol{W}}\big)$
    $\boldsymbol{Z}_{k+1} = \boldsymbol{Z}_k - \eta \tilde{\nabla}_k + \sqrt{2\eta/\beta}\boldsymbol{\epsilon}_k$
  end for
  $\tilde{\boldsymbol{Z}}^{(s)} = \boldsymbol{Z}_{(s+1)L}$
end for
```

Assumption 3.3.2 (Dissipative)**.** *There exist constants $m, b > 0$, such that we have*

$$\langle \nabla F(\mathbf{x}), \mathbf{x} \rangle \geq m\|\mathbf{x}\|_2^2 - b, \quad \textit{for all } \mathbf{x} \in \mathbb{R}^d.$$

Assumption 3.3.2 is a typical assumption for the convergence analysis of an SDE and diffusion approximation [MSH02, RRT17, ZLC17], which can be satisfied by enforcing a weight decay regularization [RRT17]. It says that starting from a position that is sufficiently far away from the origin, the Markov process defined by (3.1.1) moves towards the origin on average. It can also be noted that all critical points are within the ball of radius $O(\sqrt{b/m})$ centered at the origin under this assumption.

Let $\mathbf{x}^* = \operatorname{argmin}_{\mathbf{x}\in\mathbb{R}^d} F(\mathbf{x})$ be the global minimizer of F. Our ultimate goal is to prove the convergence of the optimization error in expectation, i.e., $\mathbb{E}[F(\boldsymbol{X}_k)] - F(\mathbf{x}^*)$. In the sequel, we decompose the optimization error into two parts: (1) $\mathbb{E}[F(\boldsymbol{X}_k)] - \mathbb{E}[F(\boldsymbol{X}^\pi)]$, which characterizes the gap between the expected function value at the k-th iterate $\boldsymbol{X}_k$ and the expected function value at $\boldsymbol{X}^\pi$, where $\boldsymbol{X}^\pi$ follows the stationary distribution $\pi(d\mathbf{x})$ of Markov process $\{\boldsymbol{X}(t)\}_{t\geq 0}$, and (2) $\mathbb{E}[F(\boldsymbol{X}^\pi)] - F(\mathbf{x}^*)$. Note that the error in part (1) is algorithm dependent, while that in part (2) only depends on the diffusion itself and hence is identical for all Langevin dynamics based algorithms.

Now we are ready to present our main results regarding to the optimization error of each algorithm reviewed in Section 3.2. We first show the optimization error bound of GLD (Algorithm 4).

Theorem 3.3.3 (GLD). *Under Assumptions 3.3.1 and 3.3.2, consider $\boldsymbol{X}_K$ generated by Algorithm 4 with initial point $\boldsymbol{X}_0 = \mathbf{0}$. The optimization error is bounded by*

$$\mathbb{E}[F(\boldsymbol{X}_K)] - F(\mathbf{x}^*) \leq \Theta e^{-\lambda K \eta} + \frac{C_\psi \eta}{\beta} + \underbrace{\frac{d}{2\beta} \log\left(\frac{eM(b\beta/d+1)}{m}\right)}_{\mathcal{R}_M}, \tag{3.3.1}$$

where problem-dependent parameters Θ and λ are defined as

$$\Theta = \frac{C_0 M(b\beta + m\beta + d)(m + e^{m\eta} M(b\beta + m\beta + d))}{m^2 \rho_\beta^{d/2}}, \quad \lambda = \frac{2m\rho_\beta^d}{\log(2M(b\beta + m\beta + d)/m)},$$

$C_0, C_\psi > 0$ *are constants, and* $\rho_\beta \in (0,1)$ *is a contraction parameter depending on the inverse temperature of the Langevin dynamics.*

In the optimization error of GLD (3.3.1), we denote the upper bound of the error term $\mathbb{E}[F(\boldsymbol{X}^\pi)] - F(\mathbf{x}^*)$ by $\mathcal{R}_M$, which characterizes the distance between the expected function value at $\boldsymbol{X}^\pi$ and the global minimum of F. The stationary distribution of Langevin diffusion $\pi \propto e^{-\beta F(\mathbf{x})}$ is a Gibbs distribution, which concentrates around the minimizer $\mathbf{x}^*$ of F. Thus a random vector $\boldsymbol{X}^\pi$ following the law of π is called an *almost minimizer* of F within a neighborhood of $\mathbf{x}^*$ with radius $\mathcal{R}_M$ [RRT17].

It is worth noting that the first term in (3.3.1) vanishes at a exponential rate due to the ergodicity of Markov chain $\{\boldsymbol{X}_k\}_{k=0,1,\ldots}$. Moreover, the exponential convergence rate is controlled by λ, the spectral gap of the discrete-time Markov chain generated by GLD, which is in the order of $e^{-\tilde{O}(d)}$.

By setting $\mathbb{E}[F(\boldsymbol{X}_K)] - \mathbb{E}[F(\boldsymbol{X}^\pi)]$ to be less than a precision ϵ, and solving for K, we have the following corollary on the iteration complexity for GLD to converge to the almost minimizer $\boldsymbol{X}^\pi$.

Corollary 3.3.4 (GLD). *Under the same conditions as in Theorem 3.3.3, provided that $\eta \lesssim \epsilon$, GLD achieves $\mathbb{E}[F(\boldsymbol{X}_K)] - \mathbb{E}[F(\boldsymbol{X}^\pi)] \leq \epsilon$ after*

$$K = O\bigg(\frac{d}{\epsilon\lambda} \cdot \log\bigg(\frac{1}{\epsilon}\bigg)\bigg)$$

iterations.

Remark 3.3.5. *In a seminal work by [RRT17], they provided a non-asymptotic analysis of SGLD for nonconvex optimization. By setting the variance of stochastic gradient to* 0*, their result immediately suggests an $O(d/(\epsilon^4\lambda^*)\log^5((1/\epsilon)))$ iteration complexity for GLD to converge to the almost minimizer up to precision ϵ. Here the quantity λ^* is the so-called* ***uniform spectral gap*** *for continuous-time Markov process $\{\boldsymbol{X}_t\}_{t\geq 0}$ generated by Langevin dynamics. They further proved that $\lambda^* = e^{-\tilde{O}(d)}$, which is in the same order of our spectral gap λ for the discrete-time Markov chain $\{\boldsymbol{X}_k\}_{k=0,1\ldots}$ generated by GLD. Both of them match the lower bound for metastable exit times of SDE for nonconvex functions that have multiple local minima and saddle points [BEGK04]. Although for some specific function F, the spectral gap may be reduced to polynomial in d [GLR17], in general, the spectral gap for continuous-time Markov processes is in the same order as the spectral gap for discrete-time Markov chains. Thus, the iteration complexity of GLD suggested by Corollary 3.3.4 is better than that suggested by [RRT17] by a factor of $O(1/\epsilon^3)$.*

We now present the following theorem, which states the optimization error of SGLD (Algorithm 5).

Theorem 3.3.6 (SGLD). *Under Assumptions 3.3.1 and 3.3.2, consider $\mathbf{Y}_K$ generated by Algorithm 5 with initial point $\mathbf{Y}_0 = \mathbf{0}$, the optimization error is bounded by*

$$\mathbb{E}[F(\mathbf{Y}_K)] - F(\mathbf{x}^*) \leq C_1\Gamma K\eta\bigg[\frac{\beta(n-B)(M\sqrt{\Gamma}+G)^2}{B(n-1)}\bigg]^{1/2} + \Theta e^{-\lambda K\eta} + \frac{C_\psi\eta}{\beta} + \mathcal{R}_M, \quad (3.3.2)$$

where C_1 is an absolute constant, C_ψ, λ, Θ and $\mathcal{R}_M$ are the same as in Theorem 3.3.3, B is the mini-batch size, $G = \max_{i=1,\ldots,n}\{\|\nabla f_i(\mathbf{x}^*)\|_2\} + bM/m$ and $\Gamma = 2(1+1/m)(b+2G^2+d/\beta)$.

Similar to Corollary 3.3.4, by setting $\mathbb{E}[F(\mathbf{Y}_k)] - \mathbb{E}[F(\boldsymbol{X}^\pi)] \leq \epsilon$, we obtain the following corollary.

Corollary 3.3.7 (SGLD). *Under the same conditions as in Theorem 3.3.6, if $\eta \lesssim \epsilon$, SGLD achieves*

$$\mathbb{E}[F(\boldsymbol{Y}_K)] - \mathbb{E}[F(\boldsymbol{X}^\pi)] = O\bigg(\frac{d^{3/2}}{B^{1/4}\lambda} \cdot \log\left(\frac{1}{\epsilon}\right) + \epsilon\bigg), \tag{3.3.3}$$

after

$$K = O\bigg(\frac{d}{\epsilon\lambda} \cdot \log\left(\frac{1}{\epsilon}\right)\bigg)$$

iterations, where B is the mini-batch size of Algorithm 5.

Remark 3.3.8. *Corollary 3.3.7 suggests that if the mini-batch size B is chosen to be large enough to offset the divergent term $\log(1/\epsilon)$, SGLD is able to converge to the almost minimizer in terms of expected function value gap. This is also suggested by the result in [RRT17]. More specifically, the result in [RRT17] implies that SGLD achieves*

$$\mathbb{E}[F(\boldsymbol{Y}_K)] - \mathbb{E}[F(\boldsymbol{X}^\pi)] = O\bigg(\frac{d^2}{\lambda^*}\left(\sigma^{-1/4}\log\left(\frac{1}{\epsilon}\right) + \epsilon\right)\bigg)$$

with $K = O(d/(\lambda^*\epsilon^4)\cdot\log^5(1/\epsilon))$, where σ^2 is the upper bound of stochastic variance in SGLD, which can be reduced with larger batch size B. Recall that the spectral gap λ^* in their work scales as $O(e^{-\tilde{O}(d)})$, which is in the same order as λ in Corollary 3.3.7. In comparison, our result in Corollary 3.3.7 indicates that SGLD can actually achieve the same order of error for $\mathbb{E}[F(\boldsymbol{Y}_K)] - \mathbb{E}[F(\boldsymbol{X}^\pi)]$ with substantially fewer number of iterations, i.e., $O(d/(\lambda\epsilon)\log(1/\epsilon))$.

Remark 3.3.9. *To ensure SGLD converges in Corollary 3.3.7, one may set a sufficiently large batch size B to offset the divergent term. For example, if we choose*

$$B \gtrsim \frac{d^6}{\lambda^4\epsilon^4}\log^4\left(\frac{1}{\epsilon}\right),$$

SGLD achieves $\mathbb{E}[F(\boldsymbol{Y}_K)] - \mathbb{E}[F(\boldsymbol{X}^\pi)] \le \epsilon$ within $K = O(d/(\lambda\epsilon)\log(1/\epsilon))$ stochastic gradient evaluations.

In what follows, we proceed to present our result on the optimization error bound of SVRG-LD.

Theorem 3.3.10 (SVRG-LD). *Under Assumptions 3.3.1 and 3.3.2, consider $\boldsymbol{Z}_K$ generated by Algorithm 6 with initial point $\boldsymbol{Z}_0 = \mathbf{0}$. The optimization error is bounded by*

$$\begin{aligned}&\mathbb{E}[F(\boldsymbol{Z}_K)] - F(\mathbf{x}^*)\\ &\le C_1\Gamma K^{3/4}\eta\bigg[\frac{L\beta M^2(n-B)}{B(n-1)}\bigg(9\eta L(M^2\Gamma + G^2) + \frac{d}{\beta}\bigg)\bigg]^{1/4} + \Theta e^{-\lambda K\eta} + \frac{C_\psi \eta}{\beta} + \mathcal{R}_M, \quad (3.3.4)\end{aligned}$$

where constants $C_1, C_\psi, \lambda, \Theta, \Gamma, G$ and $\mathcal{R}_M$ are the same as in Theorem 3.3.6, B is the mini-batch size and L is the length of inner loop of Algorithm 6.

Similar to Corollaries 3.3.4 and 3.3.7, we have the following iteration complexity for SVRG-LD.

Corollary 3.3.11 (SVRG-LD). *Under the same conditions as in Theorem 3.3.10, if $\eta \lesssim \epsilon$, SVRG-LD achieves $\mathbb{E}[F(\boldsymbol{Z}_K)] - \mathbb{E}[F(\boldsymbol{X}^\pi)] \le \epsilon$ after*

$$K = O\bigg(\frac{Ld^5}{B\lambda^4\epsilon^4} \cdot \log^4\bigg(\frac{1}{\epsilon}\bigg) + \frac{1}{\epsilon}\bigg)$$

iterations. In addition, if we choose $B = \sqrt{n}\epsilon^{-3/2}$, $L = \sqrt{n}\epsilon^{3/2}$, the number of stochastic gradient evaluations needed for SVRG-LD to achieve ϵ precision is

$$\tilde{O}\bigg(\frac{\sqrt{n}}{\epsilon^{5/2}}\bigg) \cdot e^{\tilde{O}(d)}.$$

Remark 3.3.12. *In Theorem 3.3.10 and Corollary 3.3.11, we establish the global convergence guarantee for SVRG-LD to an almost minimizer of a nonconvex function F. To the best of our knowledge, this is the first iteration/gradient complexity guarantee for SVRG-LD in nonconvex finite-sum optimization. [DRW+16] first proposed the SVRG-LD algorithm for posterior sampling, but only proved that the mean square error between averaged sample pass and the stationary distribution converges to ϵ within $\tilde{O}(1/\epsilon^{3/2})$ iterations, which has no implication for nonconvex optimization.*

In large scale machine learning problems, the evaluation of full gradient can be quite expensive, in which case the iteration complexity is no longer appropriate to reflect the efficiency of different algorithms. To perform a comprehensive comparison among the three

Table 3.1: Gradient complexities of GLD, SGLD and SVRG-LD to converge to the almost minimizer.

Work	GLD	SGLD[2]	SVRG-LD
[RRT17]	$\tilde{O}\big(\frac{n}{\epsilon^4}\big)\cdot e^{\tilde{O}(d)}$	$\tilde{O}\big(\frac{1}{\epsilon^8}\big)\cdot e^{\tilde{O}(d)}$	N/A
this chapter	$\tilde{O}\big(\frac{n}{\epsilon}\big)\cdot e^{\tilde{O}(d)}$	$\tilde{O}\big(\frac{1}{\epsilon^5}\big)\cdot e^{\tilde{O}(d)}$	$\tilde{O}\Big(\frac{\sqrt{n}}{\epsilon^{5/2}}\Big)\cdot e^{\tilde{O}(d)}$

algorithms, we present their gradient complexities for converging to the almost minimizer $\boldsymbol{X}^\pi$ with ϵ precision in Table 3.1. Recall that gradient complexity is defined as the total number of stochastic gradient evaluations needed to achieve ϵ precision. It can be seen from Table 3.1 that the gradient complexity for GLD has worse dependence on the number of component functions n and SVRG-LD has worse dependence on the optimization precision ϵ. More specifically, when the number of component functions satisfies $n \leq 1/\epsilon^5$, SVRG-LD achieves better gradient complexity than SGLD. Additionally, if $n \geq 1/\epsilon^3$, SVRG-LD is better than both GLD and SGLD, therefore is more favorable.

3.4 Proof Sketch of the Main Results

In this section, we highlight our high level idea in the analysis of GLD, SGLD and SVRG-LD.

3.4.1 Roadmap of the Proof

Recall the problem in (1.1.1) and denote the global minimizer as $\mathbf{x}^* = \operatorname{argmin}_{\mathbf{x}} F(\mathbf{x})$. $\{\boldsymbol{X}(t)\}_{t\geq 0}$ and $\{\boldsymbol{X}_k\}_{k=0,\ldots,K}$ are the continuous-time and discrete-time Markov processes generated by Langevin diffusion (3.1.1) and GLD respectively. We propose to decompose the optimization error as follows:

$$\mathbb{E}[F(\boldsymbol{X}_k)] - F(\mathbf{x}^*)$$

[2]For SGLD in [RRT17], the result in the table is obtained by choosing the exact batch size suggested by the authors that could make the stochastic variance small enough to cancel out the divergent term in their paper.

$$= \underbrace{\mathbb{E}[F(\boldsymbol{X}_k)] - \mathbb{E}[F(\boldsymbol{X}^\mu)]}_{I_1} + \underbrace{\mathbb{E}[F(\boldsymbol{X}^\mu)] - \mathbb{E}[F(\boldsymbol{X}^\pi)]}_{I_2} + \underbrace{\mathbb{E}[F(\boldsymbol{X}^\pi)] - F(\mathbf{x}^*)}_{I_3}, \tag{3.4.1}$$

where $\boldsymbol{X}^\mu$ follows the stationary distribution $\mu(d\mathbf{x})$ of Markov process $\{\boldsymbol{X}_k\}_{k=0,1,\dots,K}$, and $\boldsymbol{X}^\pi$ follows the stationary distribution $\pi(d\mathbf{x})$ of Markov process $\{\boldsymbol{X}(t)\}_{t\geq 0}$, a.k.a., the Gibbs distribution. Following existing literature [MSH02, MST10, CDC15], here we assume the existence of stationary distributions, i.e., the ergodicity, of Langevin diffusion (3.1.1) and its numerical approximation (3.2.2). Note that the ergodicity property of an SDE is not trivially guaranteed in general and establishing the existence of the stationary distribution is beyond the scope of this work. Yet we will discuss the circumstances when geometric ergodicity holds in the Appendix.

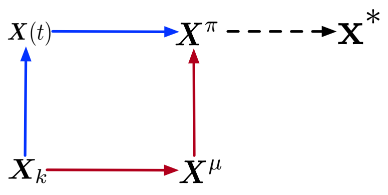

$$\begin{tikzcd} \boldsymbol{X}(t) \arrow[r] & \boldsymbol{X}^\pi \arrow[r, dashed] & \mathbf{X}^* \\ \boldsymbol{X}_k \arrow[u] \arrow[r] & \boldsymbol{X}^\mu \arrow[u] & \end{tikzcd}$$

Figure 3.1: Illustration of the analysis framework in our work.

We illustrate the decomposition (3.4.1) in Figure 3.1. Unlike existing optimization analysis of SGLD such as [RRT17], which measure the approximation error between $\boldsymbol{X}_k$ and $\boldsymbol{X}(t)$ (blue arrows in the chart), we directly analyze the geometric convergence of discretized Markov chain $\boldsymbol{X}_k$ to its stationary distribution (red arrows in the chart). Since the distance between $\boldsymbol{X}_k$ and $\boldsymbol{X}(t)$ is a slow-convergence term in [RRT17], and the distance between $\boldsymbol{X}(t)$ and $\boldsymbol{X}^\pi$ depends on the uniform spectral gap, our new roadmap of proof will bypass both of these two terms, hence leads to a faster convergence rate.

Bounding I_1: Geometric Ergodicity of GLD

To bound the first term in (3.4.1), we need to analyze the convergence of the Markov chain generated by Algorithm 4 to its stationary distribution, namely, the ergodic property of the numerical approximation of Langevin dynamics. In probability theory, ergodicity describes the long time behavior of Markov processes. For a finite-state Markov Chain, this is also closely related to the mixing time and has been thoroughly studied in the literature of Markov processes [HM08, LPW09, BGL13]. Note that [DM16] studied the convergence of the Euler-Maruyama discretization (also referred to as the unadjusted Langevin algorithm) towards its stationary distribution in total variation. Nevertheless, they only focus on strongly convex functions which are less challenging than our nonconvex setting.

The following lemma ensures the geometric ergodicity of gradient Langevin dynamics.

Lemma 3.4.1. *Under Assumptions 3.3.1 and 3.3.2, the gradient Langevin dynamics (GLD) in Algorithm 4 has a unique invariant measure μ on $\mathbb{R}^d$. It holds that*

$$|\mathbb{E}[F(\boldsymbol{X}_k)] - \mathbb{E}[F(\boldsymbol{X}^{\mu})]| \leq C\kappa\rho_\beta^{-d/2}(1+\kappa e^{m\eta}) \exp\bigg(-\frac{2mk\eta\rho_\beta^d}{\log(\kappa)}\bigg),$$

where $C > 0$ is an absolute constant, $\rho_\beta \in (0,1)$ is a contraction parameter depending on the inverse temperature parameter of Langevin dynamics (3.1.1)*, and $\kappa = 2M(b\beta + m\beta + d)/b$.*

Lemma 3.4.1 establishes the exponential decay of function gap between $F(\boldsymbol{X}_k)$ and $F(\boldsymbol{X}^{\pi})$ using coupling techniques. Note that the exponential dependence on dimension d is consistent with the result from [RRT17] using entropy methods. The contraction parameter ρ_β is a lower bound of the minorization condition for the Markov chain $\boldsymbol{X}_k$, which is established in [MSH02]. Nonetheless, we would like to point out that the exact computation of ρ_β requires additionally nontrivial efforts, which is beyond the scope of this work.

Bounding I_2: Convergence to Stationary Distribution of Langevin Diffusion

Now we are going to bound the distance between two invariant measures μ and π in terms of their expectations over the objective function F. Our proof is inspired by that of [VZT16, CDC15]. The key insight here is that after establishing the geometric ergodicity of GLD, by the stationarity of μ, we have

$$\int F(\mathbf{x})\mu(d\mathbf{x}) = \int \mathbb{E}[F(\boldsymbol{X}_k)|\boldsymbol{X}_0 = \mathbf{x}] \cdot \mu(d\mathbf{x}).$$

This property says that after reaching the stationary distribution, any further transition (GLD update) will not change the distribution. Thus we can bound the difference between two invariant measures.

Lemma 3.4.2. *Under Assumptions 3.3.1 and 3.3.2, the invariant measures μ and π satisfy*

$$\big|\mathbb{E}[F(\boldsymbol{X}^{\mu})] - \mathbb{E}[F(\boldsymbol{X}^{\pi})]\big| \leq \frac{C_{\psi}\eta}{\beta},$$

where $C_{\psi} > 0$ is a constant that dominates $\mathbb{E}[\|\nabla^p \psi(\boldsymbol{X}_k)\|]$ $(p = 0, 1, 2)$ and ψ is the solution of Poisson equation (3.1.2)*.*

Lemma 3.4.2 suggests that the bound on the difference between the two invariant measures depends on the numerical approximation step size η, the inverse temperature parameter β and the upper bound C_{ψ}. We emphasize that the dependence on β is reasonable since different β results in different diffusion, and further leads to different stationary distributions of the SDE and its numerical approximations.

Bounding I_3: Gap between Langevin Diffusion and Global Minimum

Most existing studies [WT11, SN14, CDC15] on Langevin dynamics based algorithms focus on the convergence of the averaged sample path to the stationary distribution. The property of Langevin diffusion asymptotically concentrating on the global minimum of F is well understood [CHS87, GM91] , which makes the convergence to a global minimum possible, even when the function F is nonconvex.

We give an explicit bound between the stationary distribution of Langevin diffusion and the global minimizer of F, i.e., the last term $\mathbb{E}[F(\boldsymbol{X}^{\pi})] - F(\mathbf{x}^*)$ in (3.4.1). For nonconvex objective function, this has been proved in [RRT17] using the concept of differential entropy and smoothness of F. We formally summarize it as the following lemma:

Lemma 3.4.3. *[RRT17] Under Assumptions 3.3.1 and 3.3.2, the model error I_3 in* (3.4.1) *can be bounded by*

$$\mathbb{E}[F(\boldsymbol{X}^{\pi})] - F(\mathbf{x}^*) \leq \frac{d}{2\beta}\log\left(\frac{eM(m\beta/d + 1)}{m}\right),$$

where $\boldsymbol{X}^{\pi}$ is a random vector following the stationary distribution of Langevin diffusion (3.1.1)*.*

Lemma 3.4.3 suggests that Gibbs density concentrates on the global minimizer of objective function. Therefore, the random vector $\boldsymbol{X}^{\pi}$ following the Gibbs distribution π is also referred to as an *almost minimizer* of the nonconvex function F in [RRT17].

3.4.2 Proof of Theorems 3.3.3, 3.3.6 and 3.3.10

Now we integrate the previous lemmas to prove our main theorems in Section 3.3. First, submitting the results in Lemmas 3.4.1, 3.4.2 and 3.4.3 into (3.4.1), we immediately obtain the optimization error bound in (3.3.1) for GLD, which proves Theorem 3.3.3. Second, consider the optimization error of SGLD (Algorithm 5), we only need to bound the error between $\mathbb{E}[F(\boldsymbol{Y}_K)]$ and $\mathbb{E}[F(\boldsymbol{X}_K)]$ and then apply the results for GLD, which is given by the following lemma.

Lemma 3.4.4. *Under Assumptions 3.3.1 and 3.3.2, by choosing mini-batch of size B, the output of SGLD in Algorithm 5 ($\boldsymbol{Y}_K$) and the output of GLD in Algorithm 4 ($\boldsymbol{X}_K$) satisfies*

$$|\mathbb{E}[F(\boldsymbol{Y}_K)] - \mathbb{E}[F(\boldsymbol{X}_K)]| \leq C_1\sqrt{\beta}\Gamma(M\sqrt{\Gamma}+G)K\eta\left[\frac{n-B}{B(n-1)}\right]^{1/4}, \tag{3.4.2}$$

where C_1 is an absolute constant and $\Gamma = 2(1+1/m)(b+2G^2+d/\beta)$.

Combining Lemmas 3.4.1, 3.4.2, 3.4.3 and 3.4.4 yields the desired result in (3.3.6) for SGLD, which completes the proof of Theorem 3.3.6. Third, similar to the proof of SGLD, we require an additional bound between $F(\boldsymbol{Z}_K)$ and $F(\boldsymbol{X}_K)$ for the proof of SVRG-LD, which is stated by the following lemma.

Lemma 3.4.5. *Under Assumptions 3.3.1 and 3.3.2, by choosing mini-batch of size B, the output of SVRG-LD in Algorithm 6 ($\boldsymbol{Z}_K$) and the output of GLD in Algorithm 4 ($\boldsymbol{X}_K$) satisfies*

$$\big|\mathbb{E}[F(\boldsymbol{Z}_K)] - \mathbb{E}[F(\boldsymbol{X}_K)]\big| \leq C_1\Gamma K^{3/4}\eta\left[\frac{LM^2(n-B)(3L\eta\beta(M^2\Gamma+G^2)+d/2)}{B(n-1)}\right]^{1/4},$$

where $\Gamma = 2(1+1/m)(b+2G^2+d/\beta)$, C_1 is an absolute constant and L is the number of inner loops in SVRG-LD.

The optimization error bound in (3.3.4) for SVRG-LD follows from Lemmas 3.4.1, 3.4.2, 3.4.3 and 3.4.5.

3.5 Fokker-Planck Equation and Backward Kolmogorov Equation

In this section, we introduce the Fokker-Planck Equation and the Backward Kolmogorov equation. Fokker-Planck equation addresses the evolution of probability density $p(\mathbf{x})$ that associates with the SDE. We give the following specific definition.

Definition 3.5.1 (Fokker–Planck Equation)**.** *Let $p(\mathbf{x}, t)$ be the probability density at time t of the stochastic differential equation and denote $p_0(\mathbf{x})$ the initial probability density. Then*

$$\partial_t p(\mathbf{x}, t) = \mathcal{L}^* p(\mathbf{x}, t), \quad p(\mathbf{x}, 0) = p_0(\mathbf{x}),$$

where $\mathcal{L}^*$ is the formal adjoint of $\mathcal{L}$.

Fokker-Planck equation gives us a way to find whether there exists a stationary distribution for the SDE. It can be shown [IW14] that for the stochastic differential equation (3.1.1), its stationary distribution exists and satisfies

$$\pi(d\mathbf{x}) = \frac{1}{Q} e^{-\beta F(\mathbf{x})}, \quad Q = \int e^{-\beta F(\mathbf{x})} d\mathbf{x}. \tag{3.5.1}$$

This is also known as Gibbs measure.

Backward Kolmogorov equation describes the evolution of $\mathbb{E}[g(\boldsymbol{X}(t))|\boldsymbol{X}(0) = \mathbf{x}]$ with g being a smooth test function.

Definition 3.5.2 (Backward Kolmogorov Equation)**.** *Let $\boldsymbol{X}(t)$ solves the stochastic differential equation* (3.1.1)*. Let $u(\mathbf{x}, t) = \mathbb{E}[g(\boldsymbol{X}(t))|\boldsymbol{X}(0) = \mathbf{x}]$, we have*

$$\partial_t u(\mathbf{x}, t) = \mathcal{L} u(\mathbf{x}, t), \quad u(\mathbf{x}, 0) = g(\mathbf{x}).$$

Now consider doing first order Taylor expansion on $u(\mathbf{x}, t)$, we have

$$\begin{aligned} u(\mathbf{x}, t) &= u(\mathbf{x}, 0) + \frac{\partial}{\partial t} u(\mathbf{x}, t)|_{t=0} \cdot (t - 0) + O(t^2) \\ &= g(\mathbf{x}) + t\mathcal{L} g(\mathbf{x}) + O(t^2). \end{aligned} \tag{3.5.2}$$

3.6 Proof of Corollaries

In this section, we provide the proofs of corollaries for iteration complexity in our main theory section.

Proof of Corollary 3.3.4. To ensure the iterate error converge to ϵ precision, we need

$$\Theta e^{-\lambda K\eta} \leq \frac{\epsilon}{2}, \qquad \frac{C_\psi \eta}{\beta} \leq \frac{\epsilon}{2}.$$

The second inequality can be easily satisfied with $\eta = O(\epsilon)$ and the first inequality implies

$$K \geq \frac{1}{\lambda\eta} \log\left(\frac{2\Theta}{\epsilon}\right).$$

Combining with $\eta = O(\epsilon)$ and $\Theta = O(d^2/\rho^{d/2})$, we obtain the iteration complexity

$$K = O\bigg(\frac{d}{\epsilon\lambda} \cdot \log\left(\frac{1}{\epsilon}\right)\bigg),$$

which completes the proof. □

Proof of Corollary 3.3.7. To ensure the iterate error of SGLD converging to ϵ precision, we require the following inequalities to hold

$$C_1\sqrt{\beta}\Gamma(M\sqrt{\Gamma}+G)K\eta\left[\frac{n-B}{B(n-1)}\right]^{1/4} \leq \frac{\epsilon}{3}, \qquad \Theta e^{-\lambda K\eta} \leq \frac{\epsilon}{3}, \qquad \frac{C_\psi\eta}{\beta} \leq \frac{\epsilon}{3}.$$

The third inequality can be easily satisfied with $\eta = O(\epsilon)$. For the second inequality, similar as in the proof of Corollary 3.3.4, we have

$$K\eta \geq \frac{1}{\lambda} \log\left(\frac{3\Theta}{\epsilon}\right).$$

Since $\epsilon < 1$, we know that $\log(1/\epsilon)$ will not go to zero when ϵ goes to zero. In fact, if we set $\eta = O(\epsilon)$ and $K = O(d/(\lambda\epsilon)\log(1/\epsilon))$, the first term in (3.3.2) scales as

$$C_1\sqrt{\beta}\Gamma(M\sqrt{\Gamma}+G)K\eta\left[\frac{n-B}{B(n-1)}\right]^{1/4} = O\bigg(\frac{d^{3/2}K\eta}{B^{1/4}}\bigg) = O\bigg(\frac{d^{3/2}}{B^{1/4}\lambda}\log\left(\frac{1}{\epsilon}\right)\bigg).$$

Therefore, within $K = O(d/(\epsilon\lambda)\cdot\log(1/\epsilon))$ iterations, the iterate error of SGLD scales as

$$O\bigg(\frac{d^{3/2}}{B^{1/4}\lambda}\log\left(\frac{1}{\epsilon}\right) + \epsilon\bigg).$$

This completes the proof. □

Proof of Corollary 3.3.11. Similar to previous proofs, in order to achieve an ϵ-precision iterate error for SVRG-LD, we require

$$C_1 \Gamma K^{3/4} \eta \bigg[\frac{L\beta M^2(n-B)}{B(n-1)}\bigg(9\eta(M^2\Gamma + G^2) + \frac{d}{\beta}\bigg)\bigg]^{1/4} \le \frac{\epsilon}{3}, \quad \Theta e^{-\lambda K \eta} \le \frac{\epsilon}{3}, \quad \frac{C_\psi \eta}{\beta} \le \frac{\epsilon}{3}.$$

By previous proofs we know that the second and third inequalities imply $\eta = O(\epsilon)$ and $K\eta = O(1/\lambda \log(3\Theta/\epsilon))$ respectively. Combining with the first inequality, we have

$$\eta^{1/4} = O\bigg(\frac{B^{1/4}\epsilon}{(K\eta)^{3/4} d^{5/4} L^{1/4}}\bigg).$$

Combining with the first inequality, we have

$$\eta = O\bigg(\min\bigg\{\frac{B\epsilon^4}{(K\eta)^3 d^5 L}, \epsilon\bigg\}\bigg).$$

Combining the above requirements yields

$$K = O\bigg(\frac{Ld^5}{B\lambda^4\epsilon^4}\log^4\bigg(\frac{1}{\epsilon}\bigg) + \frac{1}{\epsilon}\bigg). \tag{3.6.1}$$

For gradient complexity, note that for each iteration we need B stochastic gradient evaluations and we also need in total K/L full gradient calculations. Therefore, the gradient complexity for SVRG-LD is

$$O(K\cdot B + K/L \cdot n) = \tilde{O}\bigg(\bigg(\frac{n}{B} + L\bigg)\frac{1}{\epsilon^4} + \bigg(\frac{n}{L} + B\bigg)\frac{1}{\epsilon}\bigg)\cdot e^{\tilde{O}(d)}.$$

If we solve for the best B and L, we obtain $B = \sqrt{n}\epsilon^{-3/2}$, $L = \sqrt{n}\epsilon^{3/2}$. Therefore, we have the optimal gradient complexity for SVRG-LD as

$$\tilde{O}\bigg(\frac{\sqrt{n}}{\epsilon^{5/2}}\bigg)\cdot e^{\tilde{O}(d)},$$

which completes the proof. □

3.7 Proof of Technical Lemmas

In this section, we provide proofs of the technical lemmas used in the proof of our main theory.

3.7.1 Proof of Lemma 3.4.1

Geometric ergodicity of dynamical systems has been studied a lot in the literature [RT96, MSH02]. In particular, [RT96] proved that even when the diffusion converges exponentially fast to its stationary distribution, the Euler-Maruyama discretization in (3.2.2) may still lose the convergence properties and examples for Langevin diffusion can be found therein. To further address this problem, [MSH02] built their analysis of ergodicity for SDEs on a *minorization* condition and the existence of a Lyapunov function. In time discretization of dynamics systems, they studied how time-discretization affects the minorization condition and the Lyapunov structure. For the self-containedness of our analysis, we present the minorization condition on a compact set $\mathcal{C}$ as follows.

Proposition 3.7.1. *There exist $t_0 \in \mathbb{R}$ and $\xi > 0$ such that the Markov process $\{\boldsymbol{X}(t)\}$ satisfies*

$$\mathbb{P}(\boldsymbol{X}(t_0) \in A | \boldsymbol{X}(0) = \mathbf{x}) \geq \xi \nu(A),$$

for any $A \in \mathcal{B}(\mathbb{R}^d)$, some fixed compact set $\mathcal{C} \in \mathcal{B}(\mathbb{R}^d)$, and $\mathbf{x} \in \mathcal{C}$, where $\mathcal{B}(\mathbb{R}^d)$ denotes the Borel σ-algebra on $\mathbb{R}^d$ and ν is a probability measure with $\nu(\mathcal{C}^c) = 0$ and $\nu(\mathcal{C}) = 1$.

Proposition 3.7.1 does not always hold for a Markov process generated by an arbitrary SDE. However, for Langevin diffusion (3.1.1) studied in this chapter, [MSH02] proved that this minorization condition actually holds under the dissipative and smooth assumptions (see Corollary 7.4 in [MSH02]). For more explanation on the existence and robustness of the minorization condition under discretization approximations for Langevin diffusion, we refer interested readers to Corollary 7.5 and the proof of Theorem 6.2 in [MSH02]. Now we are going to prove Lemma 3.4.1, which requires the following useful lemmas:

Lemma 3.7.2. *Let $V(\mathbf{x}) = C + \|\mathbf{x}\|_2^2$ be a function on $\mathbb{R}^d$, where $C > 0$ is a constant. Denote the expectation with Markov process $\{\boldsymbol{X}(t)\}$ starting at $\mathbf{x}$ by $\mathbb{E}^{\mathbf{x}}[\cdot] = \mathbb{E}[\cdot | \boldsymbol{X}(0) = \mathbf{x}]$. Under Assumption 3.3.2, we have*

$$\mathbb{E}^{\mathbf{x}}[V(\boldsymbol{X}(t))] \leq e^{-2mt} V(\mathbf{x}) + \frac{b + m + d/\beta}{m}(1 - e^{-2mt}),$$

for all $\mathbf{x} \in \mathbb{R}^d$.

Proof. Applying Itô's Lemma yields

$$dV(\boldsymbol{X}(t)) = -2\langle \boldsymbol{X}(t), \nabla F(\boldsymbol{X}(t))\rangle dt + \frac{2d}{\beta}dt + 2\sqrt{\frac{2}{\beta}}\langle \boldsymbol{X}(t), d\boldsymbol{B}(t)\rangle. \tag{3.7.1}$$

Multiplying e^{2mt} to both sides of the above equation, where $m > 0$ is the dissipative constant, we obtain

$$\begin{aligned} 2me^{2mt}V(\boldsymbol{X}(t))dt + e^{2mt}dV(\boldsymbol{X}(t)) &= 2me^{2mt}V(\boldsymbol{X}(t))dt - 2e^{2mt}\langle \boldsymbol{X}(t), \nabla F(\boldsymbol{X}(t))\rangle dt \\ &\quad + \frac{2d}{\beta}e^{2mt}dt + \sqrt{\frac{8}{\beta}}e^{2mt}\langle \boldsymbol{X}(t), d\boldsymbol{B}(t)\rangle. \end{aligned}$$

We integrate the above equation from time 0 to t and have

$$\begin{aligned} V(\boldsymbol{X}(t)) &= e^{-2mt}V(\boldsymbol{X}_0) + 2m\int_0^t e^{2m(s-t)}V(\boldsymbol{X}(s))ds - 2\int_0^t e^{2m(s-t)}\langle \boldsymbol{X}(s), \nabla F(\boldsymbol{X}(s))\rangle ds \\ &\quad + \frac{2d}{\beta}\int_0^t e^{2m(s-t)}ds + 2\sqrt{\frac{2}{\beta}}\int_0^t e^{2m(s-t)}\langle \boldsymbol{X}(s), d\boldsymbol{B}(s)\rangle. \end{aligned} \tag{3.7.2}$$

Note that by Assumption 3.3.2, we have

$$\begin{aligned} -2\int_0^t e^{2m(s-t)}\langle \boldsymbol{X}(s), \nabla F(\boldsymbol{X}(s))\rangle ds &\le -2\int_0^t e^{2m(s-t)}\big(m\|\boldsymbol{X}(s)\|_2^2 - b\big)ds \\ &= -2m\int_0^t e^{2m(s-t)}V(\boldsymbol{X}(s))ds + \frac{b+m}{m}(1 - e^{-2mt}). \end{aligned} \tag{3.7.3}$$

Combining (3.7.2) and (3.7.3), and taking expectation over $\boldsymbol{X}(t)$ with initial point $\mathbf{x}$, we get

$$\begin{aligned} \mathbb{E}^{\mathbf{x}}[V(\boldsymbol{X}(t))] &\le e^{-2mt}V(\mathbf{x}) + \frac{b+m}{m}(1 - e^{-2mt}) + \frac{d}{m\beta}(1 - e^{-2mt}) \\ &= e^{-2mt}V(\mathbf{x}) + \frac{b + m + d/\beta}{m}(1 - e^{-2mt}), \end{aligned}$$

where we employed the fact that $d\boldsymbol{B}(s)$ follows Gaussian distribution with zero mean and is independent with $\boldsymbol{X}(s)$. □

Lemma 3.7.3. *(Theorem 7.3 in [MSH02]) Under Assumptions 3.3.1 and 3.3.2, let* $V(\mathbf{x}) = C_0 + M/2\|\mathbf{x}\|_2^2$ *be an essential quadratic function. The numerical approximation* (3.2.1) *(GLD) of Langevin diffusion* (3.1.1) *has a unique invariant measure* μ *and for all test function* g *such that* $|g| \le V$*, we have*

$$\big|\mathbb{E}[g(\boldsymbol{X}_k)] - \mathbb{E}[g(\boldsymbol{X}^{\mu})]\big| \le C\kappa\rho_\beta^{-d/2}(1 + \kappa e^{m\eta})\exp\bigg(-\frac{2mk\eta\rho_\beta^d}{\log(\kappa)}\bigg),$$

where $C > 0$ is an absolute constant, $\rho_\beta \in (0,1)$ is some contraction parameter depending on the inverse temperature β of Langevin dynamics, and $\kappa = 2M(b+m+d)/m$.

Here we provide a sketch of proof to refine the parameters in the results by [MSH02]. For detailed proof, we refer interested readers to Theorem 7.3 in [MSH02].

Proof. Denote $\kappa = 2M(b+m+d)/m$ according to Lemma 3.7.2 where b, m are the dissipative parameters. Following the result in [MSH02], we define $\phi = \rho_\beta^d$ with some parameter $0 < \rho_\beta < 1$ that can depend on the inverse temperature parameter β of the dynamics (3.1.1). ϕ is some lower bound of the minorization condition similar to that in Proposition 3.7.1 but is established for the Markov chain for the discretized algorithm. Note that we actually have $\beta = 1$ in [MSH02]. For the ease of presentation, we will omit the subscript β when no confusion arises. Let $\{\boldsymbol{X}_{l\tau}\}_{l=0,1,\ldots}$ be a sub-sampled chain from $\{\boldsymbol{X}_k\}_{k=0,1,\ldots}$ at sample rate $\tau > 0$. By the proof of Theorem 2.5 in [MSH02], we obtain the following result

$$\big|\mathbb{E}[g(\boldsymbol{X}_{l\tau})] - \mathbb{E}[g(\boldsymbol{X}^\mu)]\big| \le \kappa[\bar{V}+1](1-\phi)^{\alpha l\tau} + \sqrt{2}V(\mathbf{x}_0)\delta^{l\tau}\kappa^{\alpha l\tau/2}\frac{1}{\sqrt{\phi}}, \tag{3.7.4}$$

where $\boldsymbol{X}^\mu$ follows the invariant distribution of Markov process $\{\boldsymbol{X}_k\}_{k=0,1,\ldots}$, $\bar{V} = 2\sup_{\mathbf{x}\in\mathcal{C}} V(\mathbf{x})$ is a bounded constant, $\delta \in (e^{-2m\eta}, 1)$ is a constant, and $\alpha \in (0,1)$ is chosen small enough such that $\delta\kappa^{\alpha/2} \le 1$. In particular, we choose $\alpha \in (0,1)$ such that $\delta\kappa^{\alpha/2} \le (1-\phi)^\alpha$, which yields

$$\alpha \le \frac{\log(1/\delta)}{\log(\sqrt{\kappa}/(1-\phi))} \le \frac{\log(1/\delta)}{\log(\sqrt{\kappa})},$$

where the last inequality is due to $1-\phi < 1$. Submitting the choice of α into (3.7.4) we have

$$\begin{aligned}\big|\mathbb{E}[g(\boldsymbol{X}_{l\tau})] - \mathbb{E}[g(\boldsymbol{X}^\mu)]\big| &\le \frac{2\sqrt{2}\kappa}{\sqrt{\phi}}[\bar{V}+1]V(\mathbf{x}_0)(1-\phi)^{l\tau\log(1/\delta)/\log(\sqrt{\kappa})} \\ &= \frac{2\sqrt{2}\kappa}{\sqrt{\phi}}[\bar{V}+1]V(\mathbf{x}_0)e^{l\tau\log(r)},\end{aligned} \tag{3.7.5}$$

where $r = (1-\phi)^{\log(1/\delta)/\log(\sqrt{\kappa})}$ is defined as the contraction parameter. Note that by Taylor's expansion we have

$$\log r = \log(1-(1-r)) = -(1-r) - \frac{(1-r)^2}{2} - \frac{(1-r)^3}{3} - \ldots \le -(1-r), \tag{3.7.6}$$

when $|1-r| \le 1$. By definition $r = (1-\phi)^{\log(1/\delta)/\log(\sqrt{\kappa})}$ and $\phi = \rho^d$ where $\rho \in (0,1)$ is a constant. Since it is more interesting to deal with the situation where dimension parameter d is large enough and not negligible, we can always assume that $|\phi| = \rho^d$ is sufficiently small such that for any $0 < \zeta < 1$

$$(1-\phi)^\zeta = 1 - \zeta\phi + \zeta(\zeta-1)/2\phi^2 + \ldots + \binom{\zeta}{n}(-\phi)^n + \ldots \le 1 - \zeta\phi \tag{3.7.7}$$

by Taylor's expansion. Submitting (3.7.6) and (3.7.7) into (3.7.5) yields

$$\big|\mathbb{E}[g(\boldsymbol{X}_{l\tau})] - \mathbb{E}[g(\boldsymbol{X}^\mu)]\big| \le \frac{2\sqrt{2}\kappa}{\sqrt{\phi}}[\bar{V}+1]V(\mathbf{x}_0)\exp\bigg(-\frac{2ml\tau\eta\rho^d}{\log(\kappa)}\bigg), \tag{3.7.8}$$

where we chose $\delta = e^{-m\eta}$. Next we need to prove that the unsampled chain is also exponential ergodic. Let $k = l\tau + j$ with $j = 0, 1, \ldots, \tau - 1$. We immediately get

$$\big|\mathbb{E}[g(\boldsymbol{X}_{l\tau+j})] - \mathbb{E}[g(\boldsymbol{X}^\mu)]\big| \le \frac{2\sqrt{2}\kappa}{\sqrt{\phi}}[\bar{V}+1]\mathbb{E}[V(\boldsymbol{X}_j)]\exp\bigg(-\frac{2ml\tau\eta\rho^d}{\log(\kappa)}\bigg).$$

Since the GLD approximation (3.2.1) of Langevin is ergodic when sampled at rate $\tau = 1$, we have $k = l\tau = l$ and $j = 0$. Note that by Lemma A.2 in [MSH02], we have $\mathcal{C} = \{\mathbf{x} : V(\mathbf{x}) \le \kappa/e^{-m\eta}\}$, which implies that $\bar{V} = \kappa e^{m\eta}$. Thus we obtain

$$\big|\mathbb{E}[g(\boldsymbol{X}_k)] - \mathbb{E}[g(\boldsymbol{X}^\mu)]\big| \le C\kappa\rho^{-d/2}(\kappa e^{m\eta}+1)\exp\bigg(-\frac{2mk\eta\rho^d}{\log(\kappa)}\bigg),$$

where we used the fact that $\mathbf{x}_0 = \mathbf{0}$ and $C > 0$ is an absolute constant. □

Proof of Lemma 3.4.1. The proof is majorly adapted from that of Theorem 7.3 and Corollary 7.5 in [MSH02]. By Assumption 3.3.1, F is M-smooth. Thus we have

$$F(\mathbf{x}) \le F(\mathbf{y}) + \langle \nabla F(\mathbf{y}), \mathbf{x} - \mathbf{y}\rangle + \frac{M}{2}\|\mathbf{x}-\mathbf{y}\|_2^2,$$

for all $\mathbf{x}, \mathbf{y} \in \mathbb{R}^d$. By Lemma 3.7.5 and choosing $\mathbf{y} = \mathbf{0}$, this immediately implies that $F(\mathbf{x})$ can always be bounded by a quadratic function $V(\mathbf{x})$, i.e.,

$$F(\mathbf{x}) \le \frac{M}{2}V(\mathbf{x}) = \frac{M}{2}(C_0 + \|\mathbf{x}\|_2^2).$$

Therefore $V(\mathbf{x})$ is an essentially quadratic Lyapunov function such that $|F(\mathbf{x})| \le MV(\mathbf{x})/2$ for $\mathbf{x} \in \mathbb{R}^d$. By Lemma 3.7.2 the Lyapunov function satisfies

$$\mathbb{E}^{\mathbf{x}_0}[V(\boldsymbol{X}(t))] \le e^{-2mt}V(\mathbf{x}_0) + \frac{b + m + d/\beta}{m}(1 - e^{-2mt}).$$

According to Corollary 7.5 in [MSH02], the Markov chain $\{\boldsymbol{X}_k\}_{k=1,2,\ldots,K}$ satisfies

$$\mathbb{E}^{\mathbf{x}_0}[MV(\boldsymbol{X}_1)/2] \le e^{-2m\eta}[MV(\mathbf{x}_0)/2] + \frac{M(b+m+d/\beta)}{2m}. \tag{3.7.9}$$

Recall the GLD update formula defined in (3.2.1)

$$\boldsymbol{X}_{k+1} = \boldsymbol{X}_k - \eta\nabla F(\boldsymbol{X}_k) + \sqrt{2\eta\beta^{-1}} \cdot \boldsymbol{\epsilon}_k.$$

Define $F'(\boldsymbol{X}_k) = \beta F(\boldsymbol{X}_k)$ and $\eta' = \eta/\beta$, we have

$$\boldsymbol{X}_{k+1} = \boldsymbol{X}_k - \eta'\nabla F'(\boldsymbol{X}_k) + \sqrt{2\eta'} \cdot \boldsymbol{\epsilon}_k. \tag{3.7.10}$$

This suggests that the result for $\beta \neq 1$ is equivalent to rescaling η to η/β and $F(\cdot)$ to $\beta F(\cdot)$. Therefore, in the following proof, we will assume that $\beta = 1$ and then rescale η, $F(\cdot)$ at last. Similar tricks are used in [RRT17, ZLC17]. Under Assumptions 3.3.1 and 3.3.2, it is proved that Euler-Maruyama approximation of Langevin dynamics (3.1.1) has a unique invariant measure μ on $\mathbb{R}^d$. Denote $\boldsymbol{X}^\mu$ as a random vector which is sampled from measure μ. By Lemma 3.7.3, for all test function g such that $|g| \le V$, it holds that

$$\big|\mathbb{E}[g(\boldsymbol{X}_k)] - \mathbb{E}[g(\boldsymbol{X}^\mu)]\big| \le C\kappa'\rho_\beta^{-d/2}(1+\kappa' e^{m\eta}) \exp\bigg(-\frac{2mk\eta\rho_\beta^d}{\log(\kappa')}\bigg),$$

where $C > 0$ is an absolute constant, $\rho_\beta \in (0,1)$ is some contraction parameter depending on the inverse temperature β of Langevin dynamics, and $\kappa' = 2M(b+m+d)/m$. Take F as the test function and $\boldsymbol{X}_0 = \mathbf{0}$, and by rescaling η and $F(\cdot)$ (dissipative and smoothness parameters), we have

$$\big|\mathbb{E}[F(\boldsymbol{X}_k)] - \mathbb{E}[F(\boldsymbol{X}^\mu)]\big| \le C\kappa\rho_\beta^{-d/2}(1+\kappa e^{m\eta}) \exp\bigg(-\frac{2mk\eta\rho_\beta^d}{\log(\kappa)}\bigg),$$

where $\kappa = 2M(b\beta + m\beta + d)/m$. □

3.7.2 Proof of Lemma 3.4.2

To prove Lemma 3.4.2, we lay down the following supporting lemma, of which the derivation is inspired and adapted from [CDC15].

Lemma 3.7.4. *Under Assumptions 3.3.1 and 3.3.2, the Markov chain $\{\boldsymbol{X}_k\}_{k=1}^K$ generated by Algorithm 4 satisfies*

$$\left|\frac{1}{K}\sum_{k=0}^{K-1}\mathbb{E}[F(\boldsymbol{X}_k)|\boldsymbol{X}_0=\mathbf{x}]-\bar{F}\right|\le C_\psi\left(\frac{\beta}{\eta K}+\frac{\eta}{\beta}\right),$$

where $\bar{F}=\int F(\mathbf{x})\pi(d\mathbf{x})$ with π being the Gibbs measure for the Langevin diffusion (3.1.1)*.*

To prove Lemma 3.7.4, we choose the test function in Poisson equation (3.1.2) as $g = F$. Given the Poisson equation, suppose we choose g as F, the distance between the time average of the GLD process and the expectation of F over the Gibbs measure can be expressed by

$$\frac{1}{K}\sum_{k=1}^{K}F(\boldsymbol{X}_k)-\bar{F}=\frac{1}{K}\sum_{k=1}^{K}\mathcal{L}\psi(\boldsymbol{X}_k). \tag{3.7.11}$$

Note that by [MST10, VZT16], we know the Poisson equation (3.1.2) defined by the generator of Langevin dynamics has a unique solution ψ under Assumptions 3.3.1 and 3.3.2. According to Theorem 3.2 in [EMS18], the p-th order derivatives of ψ can be bounded by some polynomial growth function with sophisticated coefficients ($p = 0, 1, 2$). To simplify the presentation, we hence follow the convention in the line of literature [CDC15, VZT16] and assume that $\mathbb{E}\big[\|\nabla^p\psi(\boldsymbol{X}_k)\|\big]$ can be further upper bounded by a constant C_ψ for all $\{\boldsymbol{X}_k\}_{k\ge 0}$ and $p = (0, 1, 2)$, which is determined by the Langevin diffusion and its Poisson equation. In fact, [EMS18] showed that the upper bound of derivatives (up to fourth order) of ψ only requires the dissipative and smooth assumptions. We refer interested readers to [EMS18] for more details on deriving the C_ψ for Langevin diffusion. We show that the case $p = 0$ can be easily verified as follows. By Assumption 3.3.1, using a similar argument as in the proof of Lemma 3.4.1, we bound $F(\mathbf{x})$ by a quadratic function $V(\mathbf{x})$

$$F(\mathbf{x})\le\frac{M}{2}V(\mathbf{x})=\frac{M}{2}(C_0+\|\mathbf{x}\|_2^2).$$

Applying Assumption 3.3.2 and Theorem 13 in [VZT16] we have

$$|\psi(\mathbf{x})|\le C_1(1+\|\mathbf{x}\|_2^2)\le C_2V(\mathbf{x}). \tag{3.7.12}$$

Note that by Assumptions 3.3.1 and 3.3.2 we can verify that a quadratic $V(\mathbf{x})$ and $p^*=2$ satisfy Assumption 12 in [VZT16] and therefore we obtain that for all $p\le p^*$, we have

$$\sup_k\mathbb{E}V^p(\boldsymbol{X}_k)\le\infty. \tag{3.7.13}$$

Combining (3.7.12) and (3.7.13) we show that $\psi(\boldsymbol{X}_k)$ is bounded in expectation.

Proof. For the simplicity of notation, we first assume that $\beta = 1$ and then show the result for arbitrary β by a scaling technique. Note that for the continuous-time Markov process $\{\boldsymbol{D}(t)\}_{t\geq 0}$ defined in (3.7.33), the distribution of random vector $(\boldsymbol{X}_1, \ldots, \boldsymbol{X}_K)$ is equivalent to that of $(\boldsymbol{D}(\eta), \ldots, \boldsymbol{D}(\eta K))$. Let ψ be the solution of Poisson equation $\mathcal{L}\psi = g - \int g(\mathbf{x})\pi(d\mathbf{x})$. Since we have $\mathbb{E}[\psi(\boldsymbol{X}_k)|\boldsymbol{X}_0 = \mathbf{x}] = \mathbb{E}[\psi(\boldsymbol{D}(\eta k))|\boldsymbol{D}_0 = \mathbf{x}]$. We denote $\mathbb{E}[\psi(\boldsymbol{D}(\eta k))|\boldsymbol{D}_0 = \mathbf{x}]$ by $\mathbb{E}^{\mathbf{x}}[\psi(\boldsymbol{D}(\eta k))]$. By applying (3.5.2), we compute the Taylor expansion of $\mathbb{E}^{\mathbf{x}}[\psi(\boldsymbol{D}(\eta k))]$ at $\boldsymbol{D}(\eta(k-1))$:

$$\mathbb{E}^{\mathbf{x}}[\psi(\boldsymbol{D}(\eta k))] = \mathbb{E}^{\mathbf{x}}[\psi(\boldsymbol{D}(\eta(k-1)))] + \eta\mathbb{E}^{\mathbf{x}}[\mathcal{L}\psi(\boldsymbol{D}(\eta(k-1)))] + O(\eta^2).$$

Note that the remainder also depends on the second order derivative of the Poisson equation and are bounded by constant C_ψ. Take average over $k = 1, \ldots, K$ and rearrange the equation we have

$$\frac{1}{\eta K}\big(\mathbb{E}^{\mathbf{x}}[\psi(\boldsymbol{D}(\eta K))] - \psi(\mathbf{x})\big) + O(\eta) = \frac{1}{K}\sum_{k=1}^{K}\mathbb{E}^{\mathbf{x}}[\mathcal{L}\psi(\boldsymbol{D}(\eta(k-1)))]. \tag{3.7.14}$$

Submit the Poisson equation (3.7.11) into the above equation (3.7.14) we have

$$\begin{aligned}\frac{1}{K}\sum_{k=0}^{K-1}\mathbb{E}^{\mathbf{x}}[F(\boldsymbol{X}_k)] - \bar{F} = \frac{1}{K}\sum_{k=1}^{K}\mathbb{E}^{\mathbf{x}}[\mathcal{L}\psi(\boldsymbol{X}_{k-1})] &= \frac{1}{K}\sum_{k=1}^{K}\mathbb{E}^{\mathbf{x}}[\mathcal{L}\psi(\boldsymbol{D}(\eta(k-1)))]\\ &= \frac{1}{\eta K}\big(\mathbb{E}^{\mathbf{x}}[\psi(\boldsymbol{D}(\eta K))] - \psi(\mathbf{x})\big) + O(\eta)\\ &= \frac{1}{\eta K}\big(\mathbb{E}^{\mathbf{x}}[\psi(\boldsymbol{X}_K)] - \psi(\mathbf{x})\big) + O(\eta),\end{aligned}$$

where the second and the fourth equation hold due to the fact that the distribution of $\{\boldsymbol{X}_k\}$ is the same as the distribution of $\{\boldsymbol{D}(\eta k)\}$. We have assumed that $\psi(\boldsymbol{X}_k)$ and its first and second order derivatives are bounded by constant C_ψ in expectation over the randomness of $\boldsymbol{X}_k$. Therefore, we are able to obtain the following conclusion

$$\left|\frac{1}{K}\sum_{k=0}^{K-1}\mathbb{E}^{\mathbf{x}}[F(\boldsymbol{X}_k)] - \bar{F}\right| \leq C_\psi\left(\frac{1}{\eta K} + \eta\right).$$

This completes the proof for the case $\beta = 1$. In order to apply our analysis to the case where β can take any arbitrary constant value, we conduct the same scaling argument as in

(3.7.10).

$$\left|\frac{1}{K}\sum_{k=0}^{K-1}\mathbb{E}^{\mathbf{x}}[F(\boldsymbol{X}_k)]-\bar{F}\right|\leq C_\psi\left(\frac{1}{\eta' K}+\eta'\right)=C_\psi\left(\frac{\beta}{\eta K}+\frac{\eta}{\beta}\right).$$

This completes the proof. □

Proof of Lemma 3.4.2. By definition we have

$$\big|\mathbb{E}[F(\boldsymbol{X}^\mu)]-\mathbb{E}[F(\boldsymbol{X}^\pi)]\big|=\left|\int F(\mathbf{x})\mu(d\mathbf{x})-\int F(\mathbf{x})\pi(d\mathbf{x})\right|. \tag{3.7.15}$$

For simplicity, we denote the average $\int F(\mathbf{x})\pi(d\mathbf{x})$ as $\bar{F}$. Since μ is the ergodic limit of the Markov chain generated by the GLD process, for a given test function F, we have

$$\int F(\mathbf{x})\mu(d\mathbf{x})=\int\mathbb{E}[F(\boldsymbol{X}_k)|\boldsymbol{X}_0=\mathbf{x}]\cdot\mu(d\mathbf{x}).$$

Since μ and π are two invariant measures, we consider the case where $K\to\infty$. Take average over K steps $\{\boldsymbol{X}_k\}_{k=0}^{K-1}$ we have

$$\int F(\mathbf{x})\mu(d\mathbf{x})=\lim_{K\to\infty}\int\frac{1}{K}\sum_{k=0}^{K-1}\mathbb{E}[F(\boldsymbol{X}_k)|\boldsymbol{X}_0=\mathbf{x}]\cdot\mu(d\mathbf{x}). \tag{3.7.16}$$

Submitting (3.7.16) back into (3.7.15) yields

$$\begin{aligned}\big|\mathbb{E}[F(\boldsymbol{X}^\mu)]-\mathbb{E}[F(\boldsymbol{X}^\pi)]\big|&=\lim_{K\to\infty}\left|\int\left[\frac{1}{K}\sum_{k=0}^{K-1}\mathbb{E}[F(\boldsymbol{X}_k)|\boldsymbol{X}_0=\mathbf{x}]-\bar{F}\right]\cdot\mu(d\mathbf{x})\right|\\&\leq\lim_{K\to\infty}\int\left|\frac{1}{K}\sum_{k=0}^{K-1}\mathbb{E}[F(\boldsymbol{X}_k)|\boldsymbol{X}_0=\mathbf{x}]-\bar{F}\right|\cdot\mu(d\mathbf{x}). \end{aligned}\tag{3.7.17}$$

Apply Lemma 3.7.4 with g chosen as F we further bound (3.7.17) by

$$\begin{aligned}\big|\mathbb{E}[F(\boldsymbol{X}^\mu)]-\mathbb{E}[F(\boldsymbol{X}^\pi)]\big|&\leq C_\psi\cdot\lim_{K\to\infty}\int\left(\frac{\beta}{\eta K}+\frac{\eta}{\beta}\right)\cdot\mu(d\mathbf{x})\\&=C_\psi\cdot\lim_{K\to\infty}\left(\frac{\beta}{\eta K}+\frac{\eta}{\beta}\right)\\&=\frac{C_\psi\eta}{\beta}.\end{aligned}$$

This completes the proof. □

3.7.3 Proof of Lemma 3.4.4

Lemma 3.4.4 gives the upper bound of function value gap between the GLD iterates and the SGLD iterates. To bound the difference between $F(\boldsymbol{X}_K)$ and $F(\boldsymbol{Y}_K)$, we need the following lemmas.

We first lay down the following lemma on the bounds of gradient of f_i.

Lemma 3.7.5. *For any $\mathbf{x} \in \mathbb{R}^d$, it holds that*

$$\|\nabla f_i(\mathbf{x})\|_2 \le M\|\mathbf{x}\|_2 + G$$

for constant $G = \max_{i=1,\ldots,n}\{\|\nabla f_i(\mathbf{x}^*)\|_2\} + bM/m$.

Proof. By Assumption 3.3.2 we obtain

$$\langle \mathbf{x}^*, \nabla F(\mathbf{x}^*)\rangle \ge m\|\mathbf{x}^*\|_2^2 - b.$$

Note that $\mathbf{x}^*$ is the minimizer for F, which implies that $\nabla F(\mathbf{x}^*) = \mathbf{0}$ and therefore $\|\mathbf{x}^*\|_2 \le b/m$. By Assumption 3.3.1 we further have

$$\|\nabla f_i(\mathbf{x})\|_2 \le \|\nabla f_i(\mathbf{x}^*)\|_2 + M\|\mathbf{x} - \mathbf{x}^*\|_2 \le \|\nabla f_i(\mathbf{x}^*)\|_2 + \frac{bM}{m} + M\|\mathbf{x}\|_2.$$

The proof is completed by setting $G = \max_{i=1,\ldots,n}\{\|\nabla f_i(\mathbf{x}^*)\|_2\} + bM/m$. □

Lemma 3.7.6. *Under Assumptions 3.3.1 and 3.3.2, for any $\mathbf{x} \in \mathbb{R}^d$, it holds that*

$$\mathbb{E}\bigg\|\nabla F(\mathbf{x}) - \frac{1}{B}\sum_{i\in I_k}\nabla f_i(\mathbf{x})\bigg\|_2^2 \le \frac{4(n-B)(M\|\mathbf{x}\|_2 + G)^2}{B(n-1)},$$

where $B = |I_k|$ is the mini-batch size and $G = \max_{i=1,\ldots,n}\{\|\nabla f_i(\mathbf{x}^*)\|_2\} + bM/m$.

Proof of Lemma 3.7.6. Let $\mathbf{u}_i(\mathbf{x}) = \nabla F(\mathbf{x}) - \nabla f_i(\mathbf{x})$, consider

$$\begin{aligned}\mathbb{E}\bigg\|\frac{1}{B}\sum_{i\in I_k}\mathbf{u}_i(\mathbf{x})\bigg\|_2^2 &= \frac{1}{B^2}\mathbb{E}\sum_{i\ne i'\in I_k}\mathbf{u}_i(\mathbf{x})^\top\mathbf{u}_{i'}(\mathbf{x}) + \frac{1}{B}\mathbb{E}\|\mathbf{u}_i(\mathbf{x})\|_2^2 \\ &= \frac{B-1}{Bn(n-1)}\sum_{i\ne i'}\mathbf{u}_i(\mathbf{x})^\top\mathbf{u}_{i'}(\mathbf{x}) + \frac{1}{B}\mathbb{E}\|\mathbf{u}_i(\mathbf{x})\|_2^2\end{aligned}$$

$$= \frac{B-1}{Bn(n-1)} \sum_{i,i'} \mathbf{u}_i(\mathbf{x})^\top \mathbf{u}_{i'}(\mathbf{x}) - \frac{B-1}{B(n-1)} \mathbb{E}\|\mathbf{u}_i(\mathbf{x})\|_2^2 + \frac{1}{B}\mathbb{E}\|\mathbf{u}_i(\mathbf{x})\|_2^2$$
$$= \frac{n-B}{B(n-1)} \mathbb{E}\|\mathbf{u}_i(\mathbf{x})\|_2^2, \tag{3.7.18}$$

where the last equality is due to the fact that $1/n\sum_{i=1}^n \mathbf{u}_i(\mathbf{x}) = 0$. By Lemma 3.7.5 we have $\|\nabla f_i(\mathbf{x})\|_2 \le M\|\mathbf{x}\|_2 + G$, therefore we have $\|\nabla F(\mathbf{x})\|_2 \le M\|\mathbf{x}\|_2 + G$ and consequently, $\|\mathbf{u}_i(\mathbf{x})\|_2 \le 2(M\|\mathbf{x}\|_2 + G)$. Thus (3.7.18) can be further bounded as:

$$\mathbb{E}\bigg\|\frac{1}{B}\sum_{i\in I_k} \mathbf{u}_i(\mathbf{x})\bigg\|_2^2 \le \frac{n-B}{B(n-1)} 4(M\|\mathbf{x}\|_2 + G)^2.$$

This completes the proof. □

The following lemma describes the L^2 bound for discrete processes $\boldsymbol{X}_k$ (GLD), $\boldsymbol{Y}_k$ (SGLD) and $\mathbf{Z}_k$ (SVRG-LD). Note that for SGLD, similar result is also presented in [RRT17].

Lemma 3.7.7. *Under Assumptions 3.3.1 and 3.3.2, for sufficiently small step size η, suppose the initial points of Algorithms 4, 5 and 6 are chosen at $\mathbf{0}$, then the L^2 bound of the GLD process* (3.2.1)*, SGLD process* (3.2.2) *and SVRG-LD process* (3.2.3) *can be uniformly bounded by*

$$\max\{\mathbb{E}[\|\boldsymbol{X}_k\|_2^2], \mathbb{E}[\|\boldsymbol{Y}_k\|_2^2], \mathbb{E}[\|\boldsymbol{Z}_k\|_2^2]\} \le \Gamma, \quad \textit{where } \Gamma := 2\bigg(1+\frac{1}{m}\bigg)\bigg(b + 2G^2 + \frac{d}{\beta}\bigg),$$

for any $k = 0, 1, \ldots, K$, where $G = \max_{i=1,\ldots,n}\{\|\nabla f_i(\mathbf{x}^*)\|_2\} + bM/m$.

We provide the proof of L^2 bound of GLD and SVRG-LD iterates $\boldsymbol{X}_k$ and $\boldsymbol{Z}_k$. Note that a similar result of SGLD has been proved by [RRT17] and thus we omit the corresponding proof for the simplicity of presentation.

Proof of Lemma 3.7.7. **Part I**: We first prove the the upper bound for GLD. By the definition in (3.2.1), we have

$$\mathbb{E}[\|\boldsymbol{X}_{k+1}\|_2^2] = \mathbb{E}[\|\boldsymbol{X}_k - \eta\nabla F(\boldsymbol{X}_k)\|_2^2] + \sqrt{\frac{8\eta}{\beta}}\mathbb{E}[\langle \boldsymbol{X}_k - \eta\nabla F(\boldsymbol{X}_k), \boldsymbol{\epsilon}_k\rangle] + \frac{2\eta}{\beta}\mathbb{E}[\|\boldsymbol{\epsilon}_k\|_2^2]$$
$$= \mathbb{E}[\|\boldsymbol{X}_k - \eta\nabla F(\boldsymbol{X}_k)\|_2^2] + \frac{2\eta d}{\beta},$$

where the second equality follows from that $\boldsymbol{\epsilon}_k$ is independent on $\boldsymbol{X}_k$. Now we bound the first term

$$\begin{aligned}\mathbb{E}[\|\boldsymbol{X}_k - \eta\nabla F(\boldsymbol{X}_k)\|_2^2] &= \mathbb{E}[\|\boldsymbol{X}_k\|_2^2] - 2\eta\mathbb{E}[\langle \boldsymbol{X}_k, \nabla F(\boldsymbol{X}_k)\rangle] + \eta^2\mathbb{E}[\|\nabla F(\boldsymbol{X}_k)\|_2^2] \\ &\le \mathbb{E}[\|\boldsymbol{X}_k\|_2^2] + 2\eta(b - m\mathbb{E}[\|\boldsymbol{X}_k\|_2^2]) + 2\eta^2(M^2\mathbb{E}[\|\boldsymbol{X}_k\|_2^2] + G^2) \\ &= (1 - 2\eta m + 2\eta^2 M^2)\mathbb{E}[\|\boldsymbol{X}_k\|_2^2] + 2\eta b + 2\eta^2 G^2,\end{aligned}$$

where the inequality follows from Assumption 3.3.2, Lemma 3.7.5 and triangle inequality. Substitute the above bound back and we will have

$$\mathbb{E}[\|\boldsymbol{X}_{k+1}\|_2^2] \le (1 - 2\eta m + 2\eta^2 M^2)\mathbb{E}[\|\boldsymbol{X}_k\|_2^2] + 2\eta b + 2\eta^2 G^2 + \frac{2\eta d}{\beta}. \tag{3.7.19}$$

For sufficient small η that satisfies $\eta \le \min\{1, m/(2M^2)\}$, there are only two cases we need to take into account:
If $1 - 2\eta m + 2\eta^2 M^2 \le 0$, then from (3.7.19) we have

$$\mathbb{E}[\|\boldsymbol{X}_{k+1}\|_2^2] \le 2\eta b + 2\eta^2 G^2 + \frac{2\eta d}{\beta} \le \|\boldsymbol{X}_0\|_2^2 + 2\bigg(b + G^2 + \frac{d}{\beta}\bigg). \tag{3.7.20}$$

If $0 < 1 - 2\eta m + 2\eta^2 M^2 \le 1$, then iterate (3.7.19) and we have

$$\mathbb{E}[\|\boldsymbol{X}_k\|_2^2] \le (1 - 2\eta m + 2\eta^2 M^2)^k \|\boldsymbol{X}_0\|_2^2 + \frac{\eta b + \eta^2 G^2 + \frac{\eta d}{\beta}}{\eta m - \eta^2 M^2} \le \|\boldsymbol{X}_0\|_2^2 + \frac{2}{m}\bigg(b + G^2 + \frac{d}{\beta}\bigg). \tag{3.7.21}$$

Combine (3.7.20) and (3.7.21) and we have

$$\mathbb{E}[\|\boldsymbol{X}_k\|_2^2] \le \|\boldsymbol{X}_0\|_2^2 + \bigg(2 + \frac{2}{m}\bigg)\bigg(b + G^2 + \frac{d}{\beta}\bigg) = 2\bigg(1 + \frac{1}{m}\bigg)\bigg(b + G^2 + \frac{d}{\beta}\bigg),$$

where the equation holds by choosing $\boldsymbol{X}_0 = \mathbf{0}$.

Part II: Now we prove the L^2 bound for SVRG-LD, i.e., $\mathbb{E}[\|\boldsymbol{Z}_k\|_2^2]$, by mathematical induction. Since $\tilde{\nabla}_k = 1/B\sum_{i_k\in I_k}\big(\nabla f_{i_k}(\boldsymbol{Z}_k) - \nabla f_{i_k}(\tilde{\boldsymbol{Z}}^{(s)}) + \nabla F(\tilde{\boldsymbol{Z}}^{(s)})\big)$, we have

$$\begin{aligned}\mathbb{E}[\|\boldsymbol{Z}_{k+1}\|_2^2] &= \mathbb{E}[\|\boldsymbol{Z}_k - \eta\tilde{\nabla}_k\|_2^2] + \sqrt{\frac{8\eta}{\beta}}\mathbb{E}[\langle \boldsymbol{Z}_k - \eta\tilde{\nabla}_k, \boldsymbol{\epsilon}_k\rangle] + \frac{2\eta}{\beta}\mathbb{E}[\|\boldsymbol{\epsilon}_k\|_2^2] \\ &= \mathbb{E}[\|\boldsymbol{Z}_k - \eta\tilde{\nabla}_k\|_2^2] + \frac{2\eta d}{\beta},\end{aligned} \tag{3.7.22}$$

where the second equality follows from the fact that $\boldsymbol{\epsilon}_k$ is independent of $\boldsymbol{Z}_k$ and standard Gaussian. We prove it by induction. First, consider the case when $k = 1$. Since we choose the initial point at $\boldsymbol{Z}_0 = \mathbf{0}$, we immediately have

$$\begin{aligned}\mathbb{E}[\|\boldsymbol{Z}_1\|_2^2] &= \mathbb{E}[\|\boldsymbol{Z}_0 - \eta\tilde{\nabla}_0\|_2^2] + \sqrt{\frac{8\eta}{\beta}}\mathbb{E}[\langle \boldsymbol{Z}_0 - \eta\tilde{\nabla}_0, \boldsymbol{\epsilon}_0\rangle] + \frac{2\eta}{\beta}\mathbb{E}[\|\boldsymbol{\epsilon}_0\|_2^2] \\ &= \eta^2\mathbb{E}[\|\nabla F(\boldsymbol{Z}_0)\|_2^2] + \frac{2\eta d}{\beta} \\ &\le \eta^2 G^2 + \frac{2\eta d}{\beta},\end{aligned}$$

where the second equality holds due to the fact that $\tilde{\nabla}_0 = \nabla F(\boldsymbol{Z}_0)$ and the inequality follows from Lemma 3.7.5. For sufficiently small η we can see that the conclusion of Lemma 3.7.7 holds for $\mathbb{E}[\|\boldsymbol{Z}_1\|_2^2]$, i.e., $\mathbb{E}[\|\boldsymbol{Z}_1\|_2^2] \le \Gamma$, where $\Gamma = 2(1 + 1/m)(b + 2G^2 + d/\beta)$. Now assume that the conclusion holds for all iteration from 1 to k, then for the $(k+1)$-th iteration, by (3.7.22) we have,

$$\mathbb{E}[\|\boldsymbol{Z}_{k+1}\|_2^2] = \mathbb{E}[\|\boldsymbol{Z}_k - \eta\tilde{\nabla}_k\|_2^2] + \frac{2\eta d}{\beta}, \tag{3.7.23}$$

For the first term on the R.H.S of (3.7.23) we have

$$\begin{aligned}\mathbb{E}[\|\boldsymbol{Z}_k - \eta\tilde{\nabla}_k\|_2^2] &= \mathbb{E}[\|\boldsymbol{Z}_k - \eta\nabla F(\boldsymbol{Z}_k)\|_2^2] + 2\eta\mathbb{E}\langle \boldsymbol{Z}_k - \eta\nabla F(\boldsymbol{Z}_k), \nabla F(\boldsymbol{Z}_k) - \tilde{\nabla}_k\rangle \\ &\quad + \eta^2\mathbb{E}[\|\nabla F(\boldsymbol{Z}_k) - \tilde{\nabla}_k\|_2^2] \\ &= \underbrace{\mathbb{E}[\|\boldsymbol{Z}_k - \eta\nabla F(\boldsymbol{Z}_k)\|_2^2]}_{T_1} + \underbrace{\eta^2\mathbb{E}[\|\nabla F(\boldsymbol{Z}_k) - \tilde{\nabla}_k\|_2^2]}_{T_2},\end{aligned} \tag{3.7.24}$$

where the second equality holds due to the fact that $\mathbb{E}[\tilde{\nabla}_k] = \nabla F(\boldsymbol{Z}_k)$. For term T_1, we can further bound it by

$$\begin{aligned}\mathbb{E}[\|\boldsymbol{Z}_k - \eta\nabla F(\boldsymbol{Z}_k)\|_2^2] &= \mathbb{E}[\|\boldsymbol{Z}_k\|_2^2] - 2\eta\mathbb{E}[\langle \boldsymbol{Z}_k, \nabla F(\boldsymbol{Z}_k)\rangle] + \eta^2\mathbb{E}[\|\nabla F(\boldsymbol{Z}_k)\|_2^2] \\ &\le \mathbb{E}[\|\boldsymbol{Z}_k\|_2^2] + 2\eta(b - m\mathbb{E}[\|\boldsymbol{Z}_k\|_2^2]) + 2\eta^2(M^2\mathbb{E}[\|\boldsymbol{Z}_k\|_2^2] + G^2) \\ &= (1 - 2\eta m + 2\eta^2 M^2)\mathbb{E}[\|\boldsymbol{Z}_k\|_2^2] + 2\eta b + 2\eta^2 G^2,\end{aligned}$$

where the inequality follows from Lemma 3.7.5 and triangle inequality. For term T_2, by Lemma 3.7.11 we have

$$\mathbb{E}\|\nabla F(\boldsymbol{Z}_k) - \tilde{\nabla}_k\|_2^2 \le \frac{M^2(n-B)}{B(n-1)}\mathbb{E}\big\|\boldsymbol{Z}_k - \tilde{\boldsymbol{Z}}^{(s)}\big\|_2^2 \le \frac{2M^2(n-B)}{B(n-1)}\Big(\mathbb{E}\big\|\boldsymbol{Z}_k\big\|_2^2 + \mathbb{E}\big\|\tilde{\boldsymbol{Z}}^{(s)}\big\|_2^2\Big).$$

Submit the above bound back into (3.7.22) we have

$$\mathbb{E}[\|\boldsymbol{Z}_{k+1}\|_2^2] \le \bigg(1 - 2\eta m + 2\eta^2 M^2\Big(1 + \frac{n-B}{B(n-1)}\Big)\bigg)\mathbb{E}[\|\boldsymbol{Z}_k\|_2^2] \\ + \frac{2\eta^2 M^2(n-B)}{B(n-1)}\mathbb{E}\big\|\tilde{\boldsymbol{Z}}^{(s)}\big\|_2^2 + 2\eta b + 2\eta^2 G^2 + \frac{2\eta d}{\beta}. \tag{3.7.25}$$

Note that by assumption we have $\mathbb{E}\big\|\boldsymbol{Z}_j\big\|_2^2 \le \Gamma$ for all $j = 1, \dots, k$ where $\Gamma = 2\big(1+1/m\big)\big(b+2G^2 + d/\beta\big)$, thus (3.7.25) can be further bounded as:

$$\mathbb{E}[\|\boldsymbol{Z}_{k+1}\|_2^2] \le \underbrace{\bigg(1 - 2\eta m + 2\eta^2 M^2\Big(1 + \frac{2(n-B)}{B(n-1)}\Big)\bigg)}_{C_\lambda}\Gamma + 2\eta b + 2\eta^2 G^2 + \frac{2\eta d}{\beta}. \tag{3.7.26}$$

For sufficient small η that satisfies

$$\eta \le \min\bigg(1, \frac{m}{2M^2\big(1 + 2(n-B)/(B(n-1))\big)}\bigg),$$

there are only two cases we need to take into account:

If $C_\lambda \le 0$, then from (3.7.26) we have

$$\mathbb{E}[\|\boldsymbol{Z}_{k+1}\|_2^2] \le 2\eta b + 2\eta^2 G^2 + \frac{2\eta d}{\beta} \le 2\bigg(b + G^2 + \frac{d}{\beta}\bigg). \tag{3.7.27}$$

If $0 < C_\lambda \le 1$, then iterate (3.7.26) and we have

$$\mathbb{E}[\|\boldsymbol{Z}_{k+1}\|_2^2] \le C_\lambda^{k+1}\|\boldsymbol{Z}_0\|_2^2 + \frac{\eta b + \eta^2 G^2 + \frac{\eta d}{\beta}}{\eta m - \eta^2 M^2\Big(1 + \frac{2(n-B)}{B(n-1)}\Big)} \le \frac{2}{m}\bigg(b + G^2 + \frac{d}{\beta}\bigg). \tag{3.7.28}$$

Combining (3.7.27) and (3.7.28), we have

$$\mathbb{E}[\|\boldsymbol{Z}_{k+1}\|_2^2] \le 2\bigg(1 + \frac{1}{m}\bigg)\bigg(b + 2G^2 + \frac{d}{\beta}\bigg).$$

Thus we show that when $\mathbb{E}[\|\boldsymbol{Z}_j\|_2^2], j = 1, \dots, k$ are bounded, $\mathbb{E}[\|\boldsymbol{Z}_{k+1}\|_2^2]$ is also bounded. By mathematical induction we complete the proof. □

The following lemma gives out the upper bound for the exponential L^2 bound of $\boldsymbol{X}_k$.

Lemma 3.7.8. *Under Assumptions 3.3.1 and 3.3.2, for sufficiently small step size $\eta < 1$ and the inverse temperature satisfying $\beta \ge \max\{2/(m - M^2\eta), 4\eta\}$, it holds that*

$$\log \mathbb{E}[\exp(\|\boldsymbol{X}_k\|_2^2)] \le \|\boldsymbol{X}_0\|_2^2 + \frac{2\beta(b + G^2) + 2d}{\beta - 4\eta}k\eta.$$

Proof. We have the following equation according to the update of GLD in (3.2.1),

$$\begin{aligned}\mathbb{E}[\exp\big(\|\boldsymbol{X}_{k+1}\|_2^2\big)] &= \mathbb{E}\exp\left(\left\|\boldsymbol{X}_k - \eta\nabla F(\boldsymbol{X}_k) + \sqrt{\frac{2\eta}{\beta}}\boldsymbol{\epsilon}_k\right\|_2^2\right)\\ &= \mathbb{E}\exp\left(\|\boldsymbol{X}_k - \eta\nabla F(\boldsymbol{X}_k)\|_2^2 + \sqrt{\frac{8\eta}{\beta}}\langle \boldsymbol{X}_k - \eta\nabla F(\boldsymbol{X}_k), \boldsymbol{\epsilon}_k\rangle + \frac{2\eta}{\beta}\|\boldsymbol{\epsilon}_k\|_2^2\right).\end{aligned} \tag{3.7.29}$$

Let $H(\mathbf{x}) = \exp(\|\mathbf{x}\|_2^2)$, we have $\mathbb{E}[H(\boldsymbol{X}_{k+1})] = \mathbb{E}_{\boldsymbol{X}_k}[\mathbb{E}[H(\boldsymbol{X}_{k+1})|\boldsymbol{X}_k]]$. Thus we can first compute the conditional expectation on the R.H.S of (3.7.29) given $\boldsymbol{X}_k$, then compute the expectation with respect to $\boldsymbol{X}_k$. Note that $\boldsymbol{\epsilon}_k$ follows standard multivariate normal distribution, i.e., $\boldsymbol{\epsilon}_k \sim N(\mathbf{0}, \mathbf{I}_{d\times d})$. Then it can be shown that

$$\begin{aligned}&\mathbb{E}\left[\exp\left(\sqrt{\frac{8\eta}{\beta}}\langle \boldsymbol{X}_k - \eta\nabla F(\boldsymbol{X}_k), \boldsymbol{\epsilon}_k\rangle + \frac{2\eta}{\beta}\|\boldsymbol{\epsilon}_k\|_2^2\right)\middle|\boldsymbol{X}_k\right]\\ &= \frac{1}{\big(1-4\eta/\beta\big)^{d/2}}\exp\left(\frac{4\eta}{\beta-4\eta}\|\boldsymbol{X}_k - \eta\nabla F(\boldsymbol{X}_k)\|_2^2\right)\end{aligned}$$

holds as long as $\beta > 4\eta$. Plugging the above equation into (3.7.29), we have

$$\mathbb{E}[H(\boldsymbol{X}_{k+1})] = \frac{1}{\big(1-4\eta/\beta\big)^{d/2}}\mathbb{E}_{\boldsymbol{X}_k}\left[\exp\left(\frac{\beta}{\beta-4\eta}\|\boldsymbol{X}_k - \eta\nabla F(\boldsymbol{X}_k)\|_2^2\right)\right]. \tag{3.7.30}$$

Note that by Assumption 3.3.2 and Lemma 3.7.5 we have

$$\begin{aligned}&\mathbb{E}_{\boldsymbol{X}_k}\exp\left(\frac{\beta}{\beta-4\eta}\|\boldsymbol{X}_k - \eta\nabla F(\boldsymbol{X})\|_2^2\right)\\ &= \mathbb{E}_{\boldsymbol{X}_k}\exp\left(\frac{\beta}{\beta-4\eta}\big(\|\boldsymbol{X}_k\|_2^2 - 2\eta\langle\boldsymbol{X}_k, \nabla F(\boldsymbol{X}_k)\rangle + \eta^2\|\nabla F(\boldsymbol{X}_k)\|_2^2\big)\right)\\ &\le \mathbb{E}_{\boldsymbol{X}_k}\exp\left(\frac{\beta}{\beta-4\eta}\big(\|\boldsymbol{X}_k\|_2^2 - 2\eta(m\|\boldsymbol{X}_k\|_2^2 - b) + 2\eta^2(M^2\|\boldsymbol{X}_k\|_2^2 + G^2)\big)\right)\\ &= \mathbb{E}_{\boldsymbol{X}_k}\exp\left(\frac{\beta}{\beta-4\eta}\big((1-2\eta m + 2\eta^2M^2)\|\boldsymbol{X}_k\|_2^2 + 2b\eta + 2\eta^2G^2\big)\right).\end{aligned}$$

Consider sufficiently small η such that $\eta < m/M^2$. Then for β satisfying $\beta \ge \max\{2/(m - M^2\eta), 4\eta\}$, we have $\beta(1-2\eta m+2\eta^2M^2)/(\beta-4\eta) \le 1$. Therefore, the above expectation can be upper bounded by

$$\mathbb{E}_{\boldsymbol{X}_k}\exp\left(\frac{\beta}{\beta-4\eta}\|\boldsymbol{X}_k - \eta\nabla F(\boldsymbol{X})\|_2^2\right) \le \exp\left(\frac{2\eta\beta(b+\eta G^2)}{\beta-4\eta}\right)\mathbb{E}[H(\boldsymbol{X}_k)].$$

Substituting the above inequality into (3.7.30), it follows that

$$\begin{aligned}\mathbb{E}[H(\boldsymbol{X}_{k+1})] &\le \frac{1}{(1-4\eta/\beta)^{d/2}}\exp\bigg(\frac{2\eta\beta(b+\eta G^2)}{\beta-4\eta}\bigg)\mathbb{E}[H(\boldsymbol{X}_k)]\\ &\le \exp\bigg(\frac{2\eta(\beta b+\eta\beta G^2+d)}{\beta-4\eta}\bigg)\mathbb{E}[H(\boldsymbol{X}_k)],\end{aligned}$$

where we used the fact that $\log(1/(1-x)) \le x/(1-x)$ for $0<x<1$ and that

$$\log\bigg(\frac{1}{(1-4\eta/\beta)^{d/2}}\bigg) = \frac{d}{2}\log\bigg(\frac{1}{1-4\eta/\beta}\bigg) \le \frac{2d\eta/\beta}{1-4\eta/\beta} = \frac{2\eta d}{\beta-4\eta}.$$

Then we are able to show by induction that

$$\mathbb{E}[H(\boldsymbol{X}_k)] \le \exp\bigg(\frac{2k\eta(\beta b+\eta\beta G^2+d)}{\beta-4\eta}\bigg)\mathbb{E}[H(\|\boldsymbol{X}_0\|_2)],$$

which immediately implies that

$$\log\mathbb{E}[\exp(\|\boldsymbol{X}_k\|_2^2)] \le \|\boldsymbol{X}_0\|_2^2 + \frac{2\beta(b+G^2)+2d}{\beta-4\eta}k\eta,$$

where we assume that $\eta\le 1$ and $\beta > 4\eta$. $\square$

Lemma 3.7.9. *[PW16, RRT17] For any two probability density functions μ,ν with bounded second moments, let $g:\mathbb{R}^d\to\mathbb{R}$ be a C^1 function such that*

$$\|\nabla g(\mathbf{x})\|_2 \le C_1\|\mathbf{x}\|_2 + C_2, \forall \mathbf{x}\in\mathbb{R}^d$$

for some constants $C_1, C_2\ge 0$. Then

$$\bigg|\int_{\mathbb{R}^d} g(\mathbf{x})d\mu - \int_{\mathbb{R}^d} g(\mathbf{x})d\nu\bigg| \le (C_1\sigma + C_2)\mathcal{W}_2(\mu,\nu),$$

where $\mathcal{W}_2$ is the 2-Wasserstein distance and $\sigma^2 = \max\big\{\int_{\mathbb{R}^d}\|\mathbf{x}\|_2^2\mu(d\mathbf{x}), \int_{\mathbb{R}^d}\|\mathbf{x}\|_2^2\nu(d\mathbf{x})\big\}$.

Lemma 3.7.10. *(Corollary 2.3 in [BV05]) Let ν be a probability measure on $\mathbb{R}^d$. Assume that there exist $\mathbf{x}_0$ and a constant $\alpha>0$ such that $\int\exp(\alpha\|\mathbf{x}-\mathbf{x}_0\|_2^2)d\nu(\mathbf{x})<\infty$. Then for any probability measure μ on $\mathbb{R}^d$, it satisfies*

$$\mathcal{W}_2(\mu,\nu) \le C_\nu\Big(\sqrt{D_{KL}(\mu||\nu)} + \big(D_{KL}(\mu||\nu)/2\big)^{1/4}\Big),$$

where C_ν is defined as

$$C_\nu = \inf_{\mathbf{x}_0\in\mathbb{R}^d,\alpha>0}\sqrt{\frac{1}{\alpha}\bigg(\frac{3}{2}+\log\int\exp(\alpha\|\mathbf{x}-\mathbf{x}_0\|_2^2)d\nu(\mathbf{x})\bigg)}.$$

Proof of Lemma 3.4.4. Let P_K, Q_K denote the probability measures for GLD iterate $\boldsymbol{X}_K$ and SGLD iterate $\boldsymbol{Y}_K$ respectively. Applying Lemma 3.7.9 to probability measures P_K and Q_K yields

$$\big|\mathbb{E}[F(\boldsymbol{Y}_K)] - \mathbb{E}[F(\boldsymbol{X}_K)]\big| \leq (C_1\sqrt{\Gamma} + C_2)\mathcal{W}_2(Q_K, P_K), \tag{3.7.31}$$

where $C_1, C_2 > 0$ are absolute constants and $\Gamma = 2(1 + 1/m)(b + 2G^2 + d/\beta)$ is the upper bound for both $\mathbb{E}[\|\boldsymbol{X}_k\|_2^2]$ and $\mathbb{E}[\|\boldsymbol{Y}_k\|_2^2]$ according to Lemma 3.7.7. We further bound the $\mathcal{W}_2$ distance via the KL-divergence by Lemma 3.7.10 as follows

$$\mathcal{W}_2(Q_K, P_K) \leq \Lambda(\sqrt{D_{\mathrm{KL}}(Q_K||P_K)} + \sqrt[4]{D_{\mathrm{KL}}(Q_K||P_K)}), \tag{3.7.32}$$

where $\Lambda = \sqrt{3/2 + \log \mathbb{E}_{P_K}[\exp(\|\boldsymbol{X}_K\|_2^2)]}$. With Lemma 3.7.8 we have $\Lambda = \sqrt{(6 + 2\Gamma)K\eta}$. Therefore, we only need to bound the KL-divergence between density functions P_K and Q_K. To this end, we introduce a continuous-time Markov process $\{\boldsymbol{D}(t)\}_{t\geq 0}$ to bridge the gap between diffusion $\{\boldsymbol{X}(t)\}_{t\geq 0}$ and its numerical approximation $\{\boldsymbol{X}_k\}_{k=0,1,\ldots,K}$. Define

$$d\boldsymbol{D}(t) = b(\boldsymbol{D}(t))dt + \sqrt{2\beta^{-1}}d\boldsymbol{B}(t), \tag{3.7.33}$$

where $b(\boldsymbol{D}(t)) = -\sum_{k=0}^{\infty} \nabla F(\boldsymbol{X}(\eta k))\mathbb{1}\{t \in \big[\eta k, \eta(k+1)\big)\}$. Integrating (3.7.33) on interval $\big[\eta k, \eta(k+1)\big)$ yields

$$\boldsymbol{D}(\eta(k+1)) = \boldsymbol{D}(\eta k) - \eta\nabla F(\boldsymbol{D}(\eta k)) + \sqrt{2\eta\beta^{-1}} \cdot \boldsymbol{\epsilon}_k,$$

where $\boldsymbol{\epsilon}_k \sim N(\mathbf{0}, \mathbf{I}_{d\times d})$. This implies that the distribution of random vector $(\boldsymbol{X}_1, \ldots, \boldsymbol{X}_K)$ is equivalent to that of $(\boldsymbol{D}(\eta), \ldots, \boldsymbol{D}(\eta K))$. Similarly, for $\boldsymbol{Y}_k$ we define

$$d\tilde{\boldsymbol{M}}(t) = c(\tilde{\boldsymbol{M}}(t))dt + \sqrt{2\beta^{-1}}d\boldsymbol{B}(t),$$

where the drift coefficient is defined as $c(\tilde{\boldsymbol{M}}(t)) = -\sum_{k=0}^{\infty} g_k(\tilde{\boldsymbol{M}}(\eta k))\mathbb{1}\{t \in [\eta k, \eta(k+1))\}$ and $g_k(\mathbf{x}) = 1/B\sum_{i\in I_k} \nabla f_i(\mathbf{x})$ is a mini-batch of the full gradient with I_k being a random subset of $\{1, 2, \ldots, n\}$ of size B. Now we have that the distribution of random vector $(\boldsymbol{Y}_1, \ldots, \boldsymbol{Y}_K)$ is equivalent to that of $(\tilde{\boldsymbol{M}}(\eta), \ldots, \tilde{\boldsymbol{M}}(\eta K))$. However, the process $\tilde{\boldsymbol{M}}(t)$ is not Markov due to the randomness of the stochastic gradient g_k. Therefore, we define the following Markov process which has the same one-time marginals [Gyo86] as

$$d\boldsymbol{M}(t) = h(\boldsymbol{M}(t))dt + \sqrt{2\beta^{-1}}d\boldsymbol{B}(t), \tag{3.7.34}$$

where $h(\cdot) = -\mathbb{E}[g_k(\tilde{\boldsymbol{M}}(\eta k))\,\mathbb{1}\{t \in [\eta k, \eta(k+1))\}|\tilde{\boldsymbol{M}}(t) = \cdot]$ is the conditional expectation of the left end point of the interval which $\tilde{\boldsymbol{M}}(t)$ lies in. Let $\mathbb{P}_t$ denote the distribution of $\boldsymbol{D}(t)$ and $\mathbb{Q}_t$ denote the distribution of $\boldsymbol{M}(t)$. By (3.7.33) and (3.7.34), the Radon-Nikodym derivative of $\mathbb{P}_t$ with respective to $\mathbb{Q}_t$ is given by the following Girsanov formula [LS13]

$$\begin{aligned}\frac{d\mathbb{P}_t}{d\mathbb{Q}_t}(\boldsymbol{M}) = \exp\bigg\{\sqrt{\frac{\beta}{2}}\int_0^t (h(\boldsymbol{M}(s)) - b(\boldsymbol{M}(s)))^\top(d\boldsymbol{M}(s) - h(\boldsymbol{M}(s))ds)\\ - \frac{\beta}{4}\int_0^t \|h(\boldsymbol{M}(s)) - b(\boldsymbol{M}(s))\|_2^2 ds\bigg\}.\end{aligned}$$

Since Markov processes $\{\boldsymbol{D}(t)\}_{t\geq 0}$ and $\{\boldsymbol{M}(t)\}_{t\geq 0}$ are constructed based on Markov chains $\boldsymbol{X}_k$ and $\boldsymbol{Y}_k$, by data-processing inequality the K-L divergence between P_K and Q_K can be bounded by

$$\begin{aligned}D_{KL}(Q_K||P_K) &\leq D_{KL}(\mathbb{Q}_{\eta K}||\mathbb{P}_{\eta K})\\ &= -\mathbb{E}\bigg[\log\bigg(\frac{d\mathbb{P}_{\eta K}}{d\mathbb{Q}_{\eta K}}(\boldsymbol{M})\bigg)\bigg]\\ &= \frac{\beta}{4}\int_0^{\eta K} \mathbb{E}\big[\|h(\boldsymbol{M}(r)) - b(\boldsymbol{M}(r))\|_2^2\big]dr,\end{aligned} \tag{3.7.35}$$

where in the last equality we used the fact that $d\boldsymbol{B}(t)$ follows Gaussian distribution independently for any $t \geq 0$. By definition, we know that both $h(\boldsymbol{M}(r))$ and $b(\boldsymbol{M}(r))$ are step functions when $r \in [\eta k, \eta(k+1))$ for any k. This observation directly yields

$$\begin{aligned}\int_0^{\eta K} \mathbb{E}\big[\|h(\boldsymbol{M}(r)) - b(\boldsymbol{M}(r))\|_2^2\big]dr &\leq \sum_{k=0}^{K-1}\int_{\eta k}^{\eta(k+1)} \mathbb{E}\big[\|g_k(\tilde{\boldsymbol{M}}(\eta k)) - \nabla F(\tilde{\boldsymbol{M}}(\eta k))\|_2^2\big]dr\\ &= \eta\sum_{k=0}^{K-1}\mathbb{E}\big[\|g_k(\boldsymbol{Y}_k) - \nabla F(\boldsymbol{Y}_k)\|_2^2\big],\end{aligned}$$

where the first inequality is due to Jensen's inequality and the convexity of function $\|\cdot\|^2$, and the last equality is due to the equivalence in distribution. By Lemmas 3.7.6 and 3.7.7, we further have

$$\int_0^{\eta K} \mathbb{E}\big[\|h(\boldsymbol{M}(r)) - b(\boldsymbol{M}(r))\|_2^2\big]dr \leq \frac{4\eta K(n-B)(M\Gamma+G)^2}{B(n-1)}. \tag{3.7.36}$$

Submitting (3.7.35) and (3.7.36) into (3.7.32), we have

$$\mathcal{W}_2(Q_K, P_K) \leq \Lambda\Bigg(\sqrt{\frac{\beta\eta K(n-B)(M\Gamma+G)^2}{B(n-1)}} + \sqrt[4]{\frac{\beta\eta K(n-B)(M\Gamma+G)^2}{B(n-1)}}\Bigg)$$

$$\leq \Lambda\sqrt{\frac{\beta\eta K\sqrt{n-B}(M\Gamma+G)^2}{\sqrt{B(n-1)}}}. \tag{3.7.37}$$

Combining (3.7.31) with (3.7.37), we obtain the expected function value gap between SGLD and GLD:

$$|\mathbb{E}[F(\boldsymbol{Y}_k)]-\mathbb{E}[F(\boldsymbol{X}_k)]| \leq C_1\Gamma\sqrt{K\eta}\bigg[\frac{\beta\eta K\sqrt{n-B}(M\sqrt{\Gamma}+G)^2}{\sqrt{B(n-1)}}\bigg]^{1/2},$$

where we adopt the fact that $K\eta > 1$ and assume that $C_1 \geq C_2$. □

3.7.4 Proof of Lemma 3.4.5

Similar to the proof of Lemma 3.4.4, to bound the difference between $F(\boldsymbol{X}_K)$ and $F(\boldsymbol{Z}_K)$, we need the following lemmas.

Lemma 3.7.11. *Under Assumptions 3.3.1 and 3.3.2, for each iteration $k = sL + \ell$ in Algorithm 6, it holds that*

$$\mathbb{E}\|\tilde{\nabla}_k - \nabla F(\boldsymbol{Z}_k)\|_2^2 \leq \frac{M^2(n-B)}{B(n-1)}\mathbb{E}\big\|\boldsymbol{Z}_k - \tilde{\boldsymbol{Z}}^{(s)}\big\|_2^2,$$

where $\tilde{\nabla}_k = 1/B\sum_{i_k\in I_k}\big(\nabla f_{i_k}(\boldsymbol{Z}_k) - \nabla f_{i_k}(\tilde{\boldsymbol{Z}}^{(s)}) + \nabla F(\tilde{\boldsymbol{Z}}^{(s)}\big)$ and $B = |I_k|$ is the mini-batch size.

Proof. Since by Algorithm 6 we have $\tilde{\nabla}_k = (1/B)\sum_{i_k\in I_k}\big(\nabla f_{i_k}(\boldsymbol{Z}_k) - \nabla f_{i_k}(\tilde{\boldsymbol{Z}}^{(s)}) + \nabla F(\tilde{\boldsymbol{Z}}^{(s)})\big)$, therefore,

$$\mathbb{E}[\|\tilde{\nabla}_k - \nabla F(\boldsymbol{Z}_k)\|_2^2] = \mathbb{E}\bigg\|\frac{1}{B}\sum_{i_k\in I_k}\big(\nabla f_{i_k}(\boldsymbol{Z}_k) - \nabla f_{i_k}(\tilde{\boldsymbol{Z}}^{(s)}) + \nabla F(\tilde{\boldsymbol{Z}}^{(s)}) - \nabla F(\boldsymbol{Z}_k)\big)\bigg\|_2^2.$$

Let $\mathbf{u}_i = \nabla F(\boldsymbol{Z}_k) - \nabla F(\tilde{\boldsymbol{Z}}^{(s)}) - \big(\nabla f_{i_k}(\boldsymbol{Z}_k) - \nabla f_{i_k}(\tilde{\boldsymbol{Z}}^{(s)})\big)$.

$$\begin{aligned}
\mathbb{E}\bigg\|\frac{1}{B}\sum_{i\in I_k}\mathbf{u}_i(\mathbf{x})\bigg\|_2^2 &= \frac{1}{B^2}\mathbb{E}\sum_{i\neq i'\in I_k}\mathbf{u}_i(\mathbf{x})^\top\mathbf{u}_{i'}(\mathbf{x}) + \frac{1}{B}\mathbb{E}\|\mathbf{u}_i(\mathbf{x})\|_2^2\\
&= \frac{B-1}{Bn(n-1)}\sum_{i\neq i'}\mathbf{u}_i(\mathbf{x})^\top\mathbf{u}_{i'}(\mathbf{x}) + \frac{1}{B}\mathbb{E}\|\mathbf{u}_i(\mathbf{x})\|_2^2\\
&= \frac{B-1}{Bn(n-1)}\sum_{i,i'}\mathbf{u}_i(\mathbf{x})^\top\mathbf{u}_{i'}(\mathbf{x}) - \frac{B-1}{B(n-1)}\mathbb{E}\|\mathbf{u}_i(\mathbf{x})\|_2^2 + \frac{1}{B}\mathbb{E}\|\mathbf{u}_i(\mathbf{x})\|_2^2
\end{aligned}$$

$$= \frac{n-B}{B(n-1)}\mathbb{E}\|\mathbf{u}_i(\mathbf{x})\|_2^2, \tag{3.7.38}$$

where the last equality is due to the fact that $1/n\sum_{i=1}^n \mathbf{u}_i(\mathbf{x}) = 0$. Therefore, we have

$$\begin{aligned}
\mathbb{E}[\|\tilde{\nabla}_k - \nabla F(\boldsymbol{Z}_k)\|_2^2] &\le \frac{n-B}{B(n-1)}\mathbb{E}\|\mathbf{u}_i\|_2^2 \\
&= \frac{n-B}{B(n-1)}\mathbb{E}\|\nabla f_{i_k}(\boldsymbol{Z}_k) - \nabla f_{i_k}(\tilde{\boldsymbol{Z}}) - \mathbb{E}[\nabla f_{i_k}(\boldsymbol{Z}_k) - \nabla f_{i_k}(\tilde{\boldsymbol{Z}})]\|_2^2 \\
&\le \frac{n-B}{B(n-1)}\mathbb{E}\|\nabla f_{i_k}(\boldsymbol{Z}_k) - \nabla f_{i_k}(\tilde{\boldsymbol{Z}})\|_2^2 \\
&\le \frac{M^2(n-B)}{B(n-1)}\mathbb{E}\|\boldsymbol{Z}_k - \tilde{\boldsymbol{Z}}\|_2^2, \qquad (3.7.39)
\end{aligned}$$

where the second inequality holds due to the fact that $\mathbb{E}[\|\mathbf{x}-\mathbb{E}[\mathbf{x}]\|_2^2] \le \mathbb{E}[\|\mathbf{x}\|_2^2]$ and the last inequality follows from Assumption 3.3.1. This completes the proof. □

Proof of Lemma 3.4.5. Denote Q_K^Z as the probability density functions for $\boldsymbol{Z}_K$. For the simplicity of notation, we omit the index Z in the remaining part of this proof when no confusion arises. Similar as in the proof of Lemma 3.4.4, we first apply Lemma 3.7.9 to probability measures P_K for $\boldsymbol{X}_K$ and Q_K^Z for $\boldsymbol{Z}_K$, and obtain the following upper bound of function value gap

$$|\mathbb{E}[F(\boldsymbol{Z}_K)] - \mathbb{E}[F(\boldsymbol{X}_K)]| \le (C_1\sqrt{\Gamma} + C_2)\mathcal{W}_2(Q_K^Z, P_K), \tag{3.7.40}$$

where $C_1, C_2 > 0$ are absolute constants and $\Gamma = 2(1+1/m)(b+2G^2+d/\beta)$ is the upper bound for both $\mathbb{E}[\|\boldsymbol{X}_k\|_2^2]$ and $\mathbb{E}[\|\boldsymbol{Z}_k\|_2^2]$ according to Lemma 3.7.7. Further by Lemma 3.7.10, the $\mathcal{W}_2$ distance can be bounded by

$$\mathcal{W}_2(Q_K^Z, P_K) \le \Lambda(\sqrt{D_{\mathrm{KL}}(Q_K^Z||P_K)} + \sqrt[4]{D_{\mathrm{KL}}(Q_K^Z||P_K)}), \tag{3.7.41}$$

where $\Lambda = \sqrt{3/2 + \log\mathbb{E}_{P_K}[e^{\|\boldsymbol{X}_K\|_2^2}]}$. Applying Lemma 3.7.8 we obtain $\Lambda = \sqrt{(6+2\Gamma)K\eta}$. Therefore, we need to bound the KL-divergence between density functions P_K and Q_K^Z. Similar to the proof of Lemma 3.4.4, we define a continuous-time Markov process associated with $\boldsymbol{Z}_k$ as follows $d\tilde{\boldsymbol{N}}(t) = p(\tilde{\boldsymbol{N}}(t))dt + \sqrt{2\beta^{-1}}d\boldsymbol{B}(t)$, where $p(\tilde{\boldsymbol{N}}(t)) = -\sum_{k=0}^\infty \tilde{\nabla}_k \mathbb{1}\{t \in [\eta k, \eta(k+1))\}$ and $\tilde{\nabla}_k$ is the semi-stochastic gradient at k-th iteration of SVRG-LD. We have that the distribution of random vector $(\boldsymbol{Z}_1, \dots, \boldsymbol{Z}_K)$ is equivalent to that of $(\tilde{\boldsymbol{N}}(\eta), \dots, \tilde{\boldsymbol{N}}(\eta K))$.

However, $\tilde{\boldsymbol{N}}(t)$ is not Markov due to the randomness of $\tilde{\nabla}_k$. We define the following Markov process which has the same one-time marginals as $\tilde{\boldsymbol{N}}(t)$

$$d\boldsymbol{N}(t) = q(\boldsymbol{N}(t))dt + \sqrt{2\beta^{-1}}d\boldsymbol{B}(t), \tag{3.7.42}$$

where $q(\cdot) = -\mathbb{E}[\tilde{\nabla}_k \mathbb{1}\{t \in [\eta k, \eta(k+1))\}|p(\tilde{\boldsymbol{N}}(t)) = \cdot]$. Let $\mathbb{Q}_t^Z$ denote the distribution of $\boldsymbol{N}(t)$. By (3.7.33) and (3.7.42), the Radon-Nikodym derivative of $\mathbb{P}_t$ with respective to $\mathbb{Q}_t^Z$ is given by the Girsanov formula [LS13]

$$\begin{aligned}\frac{d\mathbb{P}_t}{d\mathbb{Q}_t^Z}(\boldsymbol{N}) = \exp\bigg\{\sqrt{\frac{\beta}{2}}\int_0^t (q(\boldsymbol{N}(r)) - b(\boldsymbol{N}(r)))^\top(d\boldsymbol{N}(r) - h(\boldsymbol{N}(r))dr)\\ - \frac{\beta}{4}\int_0^t \|q(\boldsymbol{N}(r)) - b(\boldsymbol{N}(r))\|_2^2 dr\bigg\}.\end{aligned}$$

Since Markov processes $\{\boldsymbol{D}(t)\}_{t\geq 0}$ and $\{\boldsymbol{N}(t)\}_{t\geq 0}$ are constructed based on $\boldsymbol{X}_k$ and $\boldsymbol{Z}_k$, by data-processing inequality the K-L divergence between P_K and Q_K^Z in (3.7.41) can be bounded by

$$\begin{aligned}D_{\mathrm{KL}}(Q_K^Z||P_K) &\leq D_{\mathrm{KL}}(\mathbb{Q}_{\eta K}^Z||\mathbb{P}_{\eta K})\\ &= -\mathbb{E}\bigg[\log\bigg(\frac{d\mathbb{P}_{\eta K}}{d\mathbb{Q}_{\eta K}^Z}(\boldsymbol{N})\bigg)\bigg]\\ &= \frac{\beta}{4}\int_0^{\eta K}\mathbb{E}\big[\|q(\boldsymbol{N}(r)) - b(\boldsymbol{N}(r))\|_2^2\big]dr. \end{aligned}\tag{3.7.43}$$

where in the last equality we used the fact that $d\boldsymbol{B}(t)$ follows Gaussian distribution independently for any $t \geq 0$. By definition, we know that both $q(\boldsymbol{N}(r))$ and $b(\boldsymbol{N}(r))$ are step functions when $r \in [\eta k, \eta(k+1))$ for any k. This observation directly yields

$$\begin{aligned}\int_0^{\eta K}\mathbb{E}\big[\|q(\boldsymbol{N}(r)) - b(\boldsymbol{N}(r))\|_2^2\big]dr &\leq \sum_{k=0}^{K-1}\int_{\eta k}^{\eta(k+1)}\mathbb{E}\big[\tilde{\nabla}_k(\tilde{\boldsymbol{N}}(\eta k)) - \nabla F(\tilde{\boldsymbol{N}}(\eta k))\|_2^2\big]dr\\ &= \eta\sum_{k=0}^{K-1}\mathbb{E}\big[\|\tilde{\nabla}_k(\boldsymbol{Z}_k) - \nabla F(\boldsymbol{Z}_k)\|_2^2\big],\end{aligned}$$

where the first inequality is due to Jensen's inequality and the convexity of function $\|\cdot\|_2^2$, and the last equality is due to the equivalence in distribution. Combine the above results we obtain

$$D_{\mathrm{KL}}(Q_K^Z||P_K) \leq \frac{\beta\eta}{4}\sum_{k=0}^{K-1}\mathbb{E}[\|\tilde{\nabla}_k - \nabla F(\boldsymbol{Z}_k)\|_2^2]$$

$$\leq \frac{\beta\eta}{4}\sum_{s=0}^{K/L}\sum_{\ell=0}^{L-1}\mathbb{E}[\|\tilde{\nabla}_{sL+\ell} - \nabla F(\boldsymbol{Z}_{sL+\ell})\|_2^2], \tag{3.7.44}$$

where the second inequality follows the fact that $k = sL + \ell \leq (s+1)L$ for some $\ell = 0, 1, \ldots, L-1$. Applying Lemma 3.7.11, the inner summation in (3.7.44) yields

$$\sum_{\ell=0}^{L-1}\mathbb{E}[\|\tilde{\nabla}_{sL+\ell} - \nabla F(\boldsymbol{Z}_{sL+\ell})\|_2^2] \leq \sum_{\ell=0}^{L-1}\frac{M^2(n-B)}{B(n-1)}\mathbb{E}\big\|\boldsymbol{Z}_{sL+\ell} - \tilde{\boldsymbol{Z}}^{(s)}\big\|_2^2. \tag{3.7.45}$$

Note that we have

$$\begin{aligned}
&\mathbb{E}\big\|\boldsymbol{Z}_{sL+\ell} - \tilde{\boldsymbol{Z}}^{(s)}\big\|_2^2 \\
&= \mathbb{E}\bigg\|\sum_{u=0}^{\ell-1}\eta\big(\nabla f_{i_{sL+u}}(\boldsymbol{Z}_{sL+u}) - \nabla f_{i_{sL+u}}(\tilde{\boldsymbol{Z}}^{(s)}) + \nabla F(\tilde{\boldsymbol{Z}}^{(s)})\big) - \sum_{u=0}^{\ell-1}\sqrt{\frac{2\eta}{\beta}}\epsilon_{sL+\ell}\bigg\|_2^2 \\
&\leq \ell\sum_{u=0}^{\ell-1}\mathbb{E}\big[2\eta^2\big\|\nabla f_{i_{sL+u}}(\boldsymbol{Z}_{sL+u}) - \nabla f_{i_{sL+u}}(\tilde{\boldsymbol{Z}}^{(s)}) + \nabla F(\tilde{\boldsymbol{Z}}^{(s)})\big\|_2^2\big] + \sum_{u=0}^{\ell-1}\frac{4\eta d}{\beta} \\
&\leq 4\ell\eta\bigg(9\ell\eta(M^2\Gamma^2 + G^2) + \frac{d}{\beta}\bigg),
\end{aligned} \tag{3.7.46}$$

where the first inequality holds due to the triangle inequality for the first summation term, the second one follows from Lemma 3.7.5 and Lemma 3.7.7. Submit (3.7.46) back into (3.7.45) we have

$$\begin{aligned}
\sum_{\ell=0}^{L-1}\mathbb{E}[\|\tilde{\nabla}_{sL+\ell} - \nabla F(\boldsymbol{Z}_{sL+\ell})\|_2^2] &\leq \frac{4\eta M^2(n-B)}{B(n-1)}\sum_{\ell=0}^{L-1}\bigg(9\ell^2\eta(M^2\Gamma^2 + G^2) + \frac{\ell d}{\beta}\bigg) \\
&\leq \frac{4\eta M^2(n-B)}{B(n-1)}\bigg(3L^3\eta(M^2\Gamma + G^2) + \frac{dL^2}{2\beta}\bigg),
\end{aligned} \tag{3.7.47}$$

Since (3.7.47) does not depend on the outer loop index i, submitting it into (3.7.44) yields

$$\frac{\beta\eta}{4}\sum_{k=0}^{K-1}\mathbb{E}[\|\tilde{\nabla}_k - \nabla F(\boldsymbol{Z}_k)\|_2^2] \leq \frac{\eta^2 KLM^2(n-B)(3L\eta\beta(M^2\Gamma + G^2) + d/2)}{B(n-1)}. \tag{3.7.48}$$

Combining (3.7.40), (3.7.41) (3.7.44) and (3.7.48), we obtain

$$\big|\mathbb{E}[F(\boldsymbol{Z}_K)] - \mathbb{E}[F(\boldsymbol{X}_K)]\big| \leq C_1\Gamma\sqrt{K\eta}\bigg[\frac{\eta^2 KLM^2(n-B)(3L\eta\beta(M^2\Gamma + G^2) + d/2)}{B(n-1)}\bigg]^{1/4}.$$

where we use the fact that $K\eta > 1$, $\eta < 1$ and assume that $C_1 \geq C_2$. □

Part II

Efficient Nonconvex Optimization for Reinforcement Learning

CHAPTER 4

Sample-Efficient Policy Optimization Methods with Variance Reduction

4.1 Introduction

In the previous part, we have discussed nonconvex optimization algorithms for general optimization problems. Now, we will generalize the sample-efficient optimization algorithms for general nonconvex optimization problems to reinforcement learning. In this chapter, we will focus on policy gradient methods, which are the most popular approach to optimize the agent's policy in the high dimensional continuous action space. Policy gradient method [SMSM00] parameterizes the policy by an unknown parameter $\boldsymbol{\theta} \in \mathbb{R}^d$ and directly optimizes the policy by finding the optimal $\boldsymbol{\theta}$. The objective function $J(\boldsymbol{\theta})$ is chosen to be the performance function, which is the expected return under a specific policy and is usually non-concave. Our goal is to maximize the value of $J(\boldsymbol{\theta})$ by finding a stationary point $\boldsymbol{\theta}^*$ such that $\|\nabla J(\boldsymbol{\theta}^*)\|_2 = 0$ using gradient based algorithms.

Due to the expectation in the definition of $J(\boldsymbol{\theta})$, it is usually infeasible to compute the gradient exactly. In practice, one often uses stochastic gradient estimators such as REINFORCE [Wil92], PGT [SMSM00] and GPOMDP [BB01] to approximate the gradient of the expected return based on a batch of sampled trajectories. However, this approximation will introduce additional variance and slow down the convergence of policy gradient, which thus requires a huge amount of trajectories to find a good policy. Theoretically, these stochastic gradient (SG) based algorithms require $O(1/\epsilon^2)$ trajectories [RM51] to find an ϵ-approximate stationary point such that $\mathbb{E}[\|\nabla J(\boldsymbol{\theta})\|_2^2] \leq \epsilon$. In order to reduce the variance of policy gradient algorithms, [PBC$^+$18] proposed a stochastic variance-reduced policy gradient (SVRPG) algorithm by borrowing the idea from the stochastic variance reduced gradient (SVRG) [JZ13, AZH16, RHS$^+$16] in stochastic optimization. The key idea is to use a so-called semi-stochastic gradient to replace the stochastic gradient used in SG methods. The semi-stochastic gradient combines the stochastic gradient in the current iterate with a snapshot of stochastic gradient stored in an early iterate which is called a reference iterate. In practice, SVRPG saves computation on trajectories and improves the performance of SG based policy gradient methods. [PBC$^+$18] also proved that SVRPG converges to an ϵ-approximate stationary point $\boldsymbol{\theta}$ of the nonconcave performance function $J(\boldsymbol{\theta})$ with $\mathbb{E}[\|\nabla J(\boldsymbol{\theta})\|_2^2] \leq \epsilon$ after $O(1/\epsilon^2)$ trajectories, which seems to have the same sample complexity as SG based methods. Recently, the sample complexity of SVRPG has been improved to $O(1/\epsilon^{5/3})$ by a refined analysis [XGG19], which theoretically justifies the advantage of SVRPG over SG based methods.

Table 4.1: Comparison on sample complexities of different algorithms to achieve a first-order stationary point, i.e., $\|\nabla J(\boldsymbol{\theta})\|_2^2 \leq \epsilon$.

Algorithms	Sample complexity
REINFORCE [Wil92]	$O(1/\epsilon^2)$
PGT [SMSM00]	$O(1/\epsilon^2)$
GPOMDP [BB01]	$O(1/\epsilon^2)$
SVRPG [PBC$^+$18]	$O(1/\epsilon^2)$
SVRPG [XGG19]	$O(1/\epsilon^{5/3})$
SRVR-PG (This work)	$O(1/\epsilon^{3/2})$

This work continues on this line of research. We propose a Stochastic Recursive Variance Reduced Policy Gradient algorithm (SRVR-PG), which provably improves the sample complexity of SVRPG. At the core of our proposed algorithm is a recursive semi-stochastic policy gradient inspired from the stochastic path-integrated differential estimator [FLLZ18], which accumulates all the stochastic gradients from different iterates to reduce the variance.

We prove that SRVR-PG only takes $O(1/\epsilon^{3/2})$ trajectories to converge to an ϵ-approximate stationary point $\boldsymbol{\theta}$ of the performance function, i.e., $\mathbb{E}[\|\nabla J(\boldsymbol{\theta})\|_2^2] \leq \epsilon$. We summarize the comparison of SRVR-PG with existing policy gradient methods in terms of sample complexity in Table 4.1. Evidently, the sample complexity of SRVR-PG is lower than that of REINFORCE, PGT and GPOMDP by a factor of $O(1/\epsilon^{1/2})$, and is lower than that of SVRPG [XGG19] by a factor of $O(1/\epsilon^{1/6})$.

In addition, we integrate our algorithm with parameter-based exploration (PGPE) method [SOR+08, SOR+10], and propose a SRVR-PG-PE algorithm which directly optimizes the prior probability distribution of the policy parameter $\boldsymbol{\theta}$ instead of finding the best value. The proposed SRVR-PG-PE enjoys the same trajectory complexity as SRVR-PG and performs even better in some applications due to its additional exploration over the parameter space. Our experimental results on classical control tasks in reinforcement learning demonstrate the superior performance of the proposed SRVR-PG and SRVR-PG-PE algorithms and verify our theoretical analysis.

4.1.1 Additional Related Work

We briefly review additional relevant work to ours with a focus on policy gradient based methods. For other RL methods such as value based [WD92, MKS+15] and actor-critic [KT00, PS08a, SLH+14] methods, we refer the reader to [PS08b, KBP13, SB18] for a complete review.

To reduce the variance of policy gradient methods, early works have introduced unbiased baseline functions [BB01, GBB04, PS08b] to reduce the variance, which can be constant, time-dependent or state-dependent. [SML+15] proposed the generalized advantage estimation (GAE) to explore the trade-off between bias and variance of policy gradient. Recently, action-dependent baselines are also used in [TBG+18, WRD+18] which introduces bias but reduces variance at the same time. [SOR+08, SOR+10] proposed policy gradient with parameter-based exploration (PGPE) that explores in the parameter space. It has been shown that PGPE enjoys a much smaller variance [ZHNS11]. The Stein variational policy gradient method is proposed in [LRLP17]. See [PS08b, DNPo13, Li17] for a more detailed survey on policy gradient.

Stochastic variance reduced gradient techniques such as SVRG [JZ13, XZ14], batching SVRG [HAV$^+$15], SAGA [DBLJ14] and SARAH [NLST17a] were first developed in stochastic convex optimization. When the objective function is nonconvex (or nonconcave for maximization problems), nonconvex SVRG [AZH16, RHS$^+$16] and SCSG [LJCJ17, LL18] were proposed and proved to converge to a first-order stationary point faster than vanilla SGD [RM51] with no variance reduction. The state-of-the-art stochastic variance reduced gradient methods for nonconvex functions are the SNVRG [ZXG18a] and SPIDER [FLLZ18] algorithms, which have been proved to achieve near optimal convergence rate for smooth functions.

There are yet not many papers studying variance reduced gradient techniques in RL. [DCL$^+$17] first applied SVRG in policy evaluation for a fixed policy. [XLP17] introduced SVRG into trust region policy optimization for model-free policy gradient and showed that the resulting algorithm SVRPO is more sample efficient than TRPO. [YLTZ19] further applied the techniques in SARAH [NLST17a] and SPIDER [FLLZ18] to TRPO [SLA$^+$15]. However, no analysis on sample complexity (i.e., number of trajectories required) was provided in the aforementioned papers [XLP17, YLTZ19]. We note that a recent work by [SRH$^+$19] proposed a Hessian aided policy gradient (HAPG) algorithm that converges to the stationary point of the performance function within $O(H^2/\epsilon^{3/2})$ trajectories, which is worse than our result by a factor of $O(H^2)$ where H is the horizon length of the environment. Moreover, they need additional samples to approximate the Hessian vector product, and cannot handle the policy in a constrained parameter space. Another related work pointed out by the anonymous reviewer is [YZ19], which extended the stochastic mirror descent algorithm [GLZ16] in the optimization field to policy gradient methods and achieved $O(H^2/\epsilon^2)$ sample complexity. After the ICLR conference submission deadline, [YZ19] revised their paper by adding a new variance reduction algorithm that achieves $O(H^2/\epsilon^{3/2})$ sample complexity, which is also worse than our result by a factor of $O(H^2)$.

Apart from the convergence analysis of the general nonconcave performance functions, there has emerged a line of work [CYLW19, LCYW19, YCHW19, WCYW20] that studies the global convergence of (proximal/trust-region) policy optimization with neural network function approximation, which applies the theory of overparameterized neural networks [DZPS19, DLL+19, AZLS19, ZCZG19, CG19b] to reinforcement learning.

4.2 Backgrounds on Policy Gradient

Markov Decision Process: A discrete-time Markov Decision Process (MDP) is a tuple $\mathcal{M} = \{\mathcal{S}, \mathcal{A}, \mathcal{P}, r, \gamma, \rho\}$. $\mathcal{S}$ and $\mathcal{A}$ are the state and action spaces respectively. $\mathcal{P}(s'|s,a)$ is the transition probability of transiting to state s' after taking action a at state s. Function $r(s,a) : \mathcal{S} \times \mathcal{A} \to [-R, R]$ emits a bounded reward after the agent takes action a at state s, where $R > 0$ is a constant. $\gamma \in (0,1)$ is the discount factor. ρ is the distribution of the starting state. A policy at state s is a probability function $\pi(a|s)$ over action space $\mathcal{A}$. In episodic tasks, following any stationary policy, the agent can observe and collect a sequence of state-action pairs $\tau = \{s_0, a_0, s_1, a_1, \ldots, s_{H-1}, a_{H-1}, s_H\}$, which is called a trajectory or episode. H is called the trajectory horizon or episode length. In practice, we can set H to be the maximum value among all the actual trajectory horizons we have collected. The sample return over one trajectory τ is defined as the discounted cumulative reward $\mathcal{R}(\tau) = \sum_{h=0}^{H-1} \gamma^h r(s_h, a_h)$.

Policy Gradient: Suppose the policy, denoted by $\pi_{\boldsymbol{\theta}}$, is parameterized by an unknown parameter $\boldsymbol{\theta} \in \mathbb{R}^d$. We denote the trajectory distribution induced by $\pi_{\boldsymbol{\theta}}$ as $p(\tau|\boldsymbol{\theta})$. Then

$$p(\tau|\boldsymbol{\theta}) = \rho(s_0) \prod_{h=0}^{H-1} \pi_{\boldsymbol{\theta}}(a_h|s_h) P(s_{h+1}|s_h, a_h). \tag{4.2.1}$$

We define the expected return under policy $\pi_{\boldsymbol{\theta}}$ as $J(\boldsymbol{\theta}) = \mathbb{E}_{\tau \sim p(\cdot|\boldsymbol{\theta})}[\mathcal{R}(\tau)|\mathcal{M}]$, which is also called the performance function. To maximize the performance function, we can update the policy parameter $\boldsymbol{\theta}$ by iteratively running gradient ascent based algorithms, i.e., $\boldsymbol{\theta}_{k+1} = \boldsymbol{\theta}_k + \eta \nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta}_k)$, where $\eta > 0$ is the step size and the gradient $\nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta})$ is derived as follows:

$$\begin{aligned}\nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta}) &= \int_\tau \mathcal{R}(\tau) \nabla_{\boldsymbol{\theta}} p(\tau|\boldsymbol{\theta}) \mathrm{d}\tau = \int_\tau \mathcal{R}(\tau)(\nabla_{\boldsymbol{\theta}} p(\tau|\boldsymbol{\theta})/p(\tau|\boldsymbol{\theta})) p(\tau|\boldsymbol{\theta}) \mathrm{d}\tau \\ &= \mathbb{E}_{\tau \sim p(\cdot|\boldsymbol{\theta})}[\nabla_{\boldsymbol{\theta}} \log p(\tau|\boldsymbol{\theta}) \mathcal{R}(\tau)|\mathcal{M}]. \end{aligned} \tag{4.2.2}$$

However, it is intractable to calculate the exact gradient in (4.2.2) since the trajectory distribution $p(\tau|\boldsymbol{\theta})$ is unknown. In practice, policy gradient algorithm samples a batch of trajectories $\{\tau_i\}_{i=1}^N$ to approximate the exact gradient based on the sample average over all sampled trajectories:

$$\hat{\nabla}_{\boldsymbol{\theta}} J(\boldsymbol{\theta}) = 1/N \textstyle\sum_{i=1}^N \nabla_{\boldsymbol{\theta}} \log p(\tau_i|\boldsymbol{\theta}) \mathcal{R}(\tau_i). \tag{4.2.3}$$

At the k-th iteration, the policy is then updated by $\boldsymbol{\theta}_{k+1} = \boldsymbol{\theta}_k + \eta \hat{\nabla}_{\boldsymbol{\theta}} J(\boldsymbol{\theta}_k)$. According to (4.2.1), we know that $\nabla_{\boldsymbol{\theta}} \log p(\tau_i|\boldsymbol{\theta})$ is independent of the transition probability matrix P. Recall the definition of $\mathcal{R}(\tau)$, we can rewrite the approximate gradient as follows

$$\begin{aligned}\hat{\nabla}_{\boldsymbol{\theta}} J(\boldsymbol{\theta}) &= 1/N \textstyle\sum_{i=1}^N \big(\sum_{h=0}^{H-1} \nabla_{\boldsymbol{\theta}} \log \pi_{\boldsymbol{\theta}}(a_h^i|s_h^i)\big)\big(\sum_{h=0}^{H-1} \gamma^h r(s_h^i, a_h^i)\big) \\ &\overset{\text{def}}{=} 1/N \textstyle\sum_{i=1}^N g(\tau_i|\boldsymbol{\theta}),\end{aligned} \tag{4.2.4}$$

where $\tau_i = \{s_0^i, a_0^i, s_1^i, a_1^i, \ldots, s_{H-1}^i, a_{H-1}^i, s_H^i\}$ for all $i = 1, \ldots, N$ and $g(\tau_i|\boldsymbol{\theta})$ is an unbiased gradient estimator computed based on the i-th trajectory τ_i. The gradient estimator in (4.2.4) is based on the likelihood ratio methods and is often referred to as the REINFORCE gradient estimator [Wil92]. Since $\mathbb{E}[\nabla_{\boldsymbol{\theta}} \log \pi_{\boldsymbol{\theta}}(a|s)] = 0$, we can add any constant baseline b_t to the reward that is independent of the current action and the gradient estimator still remains unbiased. With the observation that future actions do not depend on past rewards, another famous policy gradient theorem (PGT) estimator [SMSM00] removes the rewards from previous states:

$$g(\tau_i|\boldsymbol{\theta}) = \textstyle\sum_{h=0}^{H-1} \nabla_{\boldsymbol{\theta}} \log \pi_{\boldsymbol{\theta}}(a_h^i|s_h^i)\big(\sum_{t=h}^{H-1} \gamma^t r(s_t^i, a_t^i) - b_t\big), \tag{4.2.5}$$

where b_t is a constant baseline. It has been shown [PS08b] that the PGT estimator is equivalent to the commonly used GPOMDP estimator [BB01] defined as follows:

$$g(\tau_i|\boldsymbol{\theta}) = \textstyle\sum_{h=0}^{H-1} \big(\sum_{t=0}^{h} \nabla_{\boldsymbol{\theta}} \log \pi_{\boldsymbol{\theta}}(a_t^i|s_t^i)\big)\big(\gamma^h r(s_h^i, a_h^i) - b_h\big). \tag{4.2.6}$$

All the three gradient estimators mentioned above are unbiased [PS08b]. It has been proved that the variance of the PGT/GPOMDP estimator is independent of horizon H while the variance of REINFORCE depends on H polynomially [ZHNS11, PRB13]. Therefore, we will focus on the PGT/GPOMDP estimator in this chapter and refer to them interchangeably due to their equivalence.

4.3 The Proposed Algorithm

The approximation in (4.2.3) using a batch of trajectories often causes a high variance in practice. In this section, we propose a novel variance reduced policy gradient algorithm called stochastic recursive variance reduced policy gradient (SRVR-PG), which is displayed in Algorithm 7. Our SRVR-PG algorithm consists of S epochs. In the initialization, we set the parameter of a reference policy to be $\tilde{\boldsymbol{\theta}}^0 = \boldsymbol{\theta}_0$. At the beginning of the s-th epoch, where $s = 0, \ldots, S-1$, we set the initial policy parameter $\boldsymbol{\theta}_0^{s+1}$ to be the same as that of the reference policy $\tilde{\boldsymbol{\theta}}^s$. The algorithm then samples N episodes $\{\tau_i\}_{i=1}^N$ from the reference policy $\pi_{\tilde{\boldsymbol{\theta}}^s}$ to compute a gradient estimator $\mathbf{v}_0^s = 1/N \sum_{i=1}^N g(\tau_i|\tilde{\boldsymbol{\theta}}^s)$, where $g(\tau_i|\tilde{\boldsymbol{\theta}}^s)$ is the PGT/GPOMDP estimator. Then the policy is immediately update as in Line 6 of Algorithm 7.

Within the epoch, at the t-th iteration, SRVR-PG samples B episodes $\{\tau_j\}_{j=1}^B$ based on the current policy $\pi_{\boldsymbol{\theta}_t^{s+1}}$. We define the following recursive semi-stochastic gradient estimator:

$$\mathbf{v}_t^{s+1} = 1/B \sum_{j=1}^B g(\tau_j|\boldsymbol{\theta}_t^{s+1}) - 1/B \sum_{j=1}^B g_\omega(\tau_j|\boldsymbol{\theta}_{t-1}^{s+1}) + \mathbf{v}_{t-1}^{s+1}, \tag{4.3.1}$$

where the first term is a stochastic gradient based on B episodes sampled from the current policy, and the second term is a stochastic gradient defined based on the step-wise important weight between the current policy $\pi_{\boldsymbol{\theta}_t^{s+1}}$ and the reference policy $\pi_{\tilde{\boldsymbol{\theta}}^s}$. Take the GPOMDP estimator for example, the step-wise importance weighted estimator is defined as follows

$$g_\omega(\tau_j|\boldsymbol{\theta}) = \sum_{h=0}^{H-1} \omega_{0:h}(\tau|\boldsymbol{\theta}_1, \boldsymbol{\theta}_2)\big(\sum_{t=0}^h \nabla_{\boldsymbol{\theta}} \log \pi_{\boldsymbol{\theta}}(a_t^j|s_t^j)\big)\gamma^h r(s_h^j, a_h^j), \tag{4.3.2}$$

where $\omega_{0:h}(\tau|\boldsymbol{\theta}_1, \boldsymbol{\theta}_2) = \prod_{h'=0}^h \pi_{\boldsymbol{\theta}_1}(a_h|s_h)/\pi_{\boldsymbol{\theta}_2}(a_h|s_h)$ is the importance weight from $p(\tau_h|\boldsymbol{\theta}_t^{s+1})$ to $p(\tau_h|\boldsymbol{\theta}_{t-1}^{s+1})$ and τ_h is a truncated trajectory $\{(a_t, s_t)\}_{t=0}^h$ from the full trajectory τ. The difference between the last two terms in (4.3.1) can be viewed as a control variate to reduce the variance of the stochastic gradient. In many practical applications, the policy parameter space is a subset of $\mathbb{R}^d$, i.e., $\boldsymbol{\theta} \in \boldsymbol{\Theta}$ with $\boldsymbol{\Theta} \subseteq \mathbb{R}^d$ being a convex set. In this case, we need to project the updated policy parameter onto the constraint set. Base on the semi-stochastic gradient (4.3.1), we can update the policy parameter using projected gradient ascent along the direction of $\mathbf{v}_t^{s+1}$: $\boldsymbol{\theta}_{t+1}^{s+1} = \mathcal{P}_{\boldsymbol{\Theta}}(\boldsymbol{\theta}_t^{s+1} + \eta\mathbf{v}_t^{s+1})$, where $\eta > 0$ is the step size and the projection operator associated with $\mathbf{\Theta}$ is defined as

$$\mathcal{P}_{\mathbf{\Theta}}(\boldsymbol{\theta}) = \operatorname{argmin}_{\mathbf{u}\in\mathbf{\Theta}} \|\boldsymbol{\theta} - \mathbf{u}\|_2^2 = \operatorname{argmin}_{\mathbf{u}\in\mathbb{R}^d}\{\mathbb{1}_{\mathbf{\Theta}}(\mathbf{u}) + 1/(2\eta)\|\boldsymbol{\theta} - \mathbf{u}\|_2^2\}, \tag{4.3.3}$$

where $\mathbb{1}_{\mathbf{\Theta}}(\mathbf{u})$ is the set indicator function on $\mathbf{\Theta}$, i.e., $\mathbb{1}_{\mathbf{\Theta}}(\mathbf{u}) = 0$ if $\mathbf{u} \in \mathbf{\Theta}$ and $\mathbb{1}_{\mathbf{\Theta}}(\mathbf{u}) = +\infty$ otherwise. $\eta > 0$ is any finite real value and is chosen as the step size in this work. It is easy to see that $\mathbb{1}_{\mathbf{\Theta}}(\cdot)$ is nonsmooth. At the end of the s-th epoch, we update the reference policy as $\tilde{\theta}^{s+1} = \theta_m^{s+1}$, where θ_m^{s+1} is the last iterate of this epoch.

The goal of our algorithm is to find a point $\boldsymbol{\theta} \in \mathbf{\Theta}$ that maximizes the performance function $J(\boldsymbol{\theta})$ subject to the constraint, namely, $\max_{\boldsymbol{\theta}\in\mathbf{\Theta}} J(\boldsymbol{\theta}) = \max_{\boldsymbol{\theta}\in\mathbb{R}^d}\{J(\boldsymbol{\theta}) - \mathbb{1}_{\mathbf{\Theta}}(\boldsymbol{\theta})\}$. The gradient norm $\|\nabla J(\boldsymbol{\theta})\|_2$ is not sufficient to characterize the convergence of the algorithm due to additional the constraint. Following the literature on nonsmooth optimization [RSPS16, GLZ16, NLST17a, LL18, WJZ+19], we use the generalized first-order stationary condition: $\mathcal{G}_\eta(\boldsymbol{\theta}) = \mathbf{0}$, where the *gradient mapping* $\mathcal{G}_\eta$ is defined as follows

$$\mathcal{G}_\eta(\boldsymbol{\theta}) = 1/\eta(\mathcal{P}_{\mathbf{\Theta}}(\boldsymbol{\theta} + \eta\nabla J(\boldsymbol{\theta})) - \boldsymbol{\theta}). \tag{4.3.4}$$

We can view $\mathcal{G}_\eta$ as a generalized projected gradient at $\boldsymbol{\theta}$. By definition if $\mathbf{\Theta} = \mathbb{R}^d$, we have $\mathcal{G}_\eta(\boldsymbol{\theta}) \equiv \nabla J(\boldsymbol{\theta})$. Therefore, the policy is update is displayed in Line 10 in Algorithm 7, where prox is the proximal operator defined in (4.3.3). Similar recursive semi-stochastic gradients to (4.3.1) were first proposed in stochastic optimization for finite-sum problems, leading to the stochastic recursive gradient algorithm (SARAH) [NLST17a, NvDP+19] and the stochastic path-integrated differential estimator (SPIDER) [FLLZ18, WJZ+19]. However, our gradient estimator in (4.3.1) is noticeably different from that in [NLST17a, FLLZ18, WJZ+19, NvDP+19] due to the gradient estimator $g_\omega(\tau_j|\boldsymbol{\theta}_{t-1}^{s+1})$ defined in (4.3.2) that is equipped with step-wise importance weights. This term is essential to deal with the non-stationarity of the distribution of the trajectory τ. Specifically, $\{\tau_j\}_{j=1}^B$ are sampled from policy $\pi_{\boldsymbol{\theta}_t^{s+1}}$ while the PGT/GPOMDP estimator $g(\cdot|\boldsymbol{\theta}_{t-1}^{s+1})$ is defined based on policy $\pi_{\boldsymbol{\theta}_{t-1}^{s+1}}$ according to (4.2.6). This inconsistency introduces extra challenges in the convergence analysis of SRVR-PG. Using importance weighting, we can obtain $\mathbb{E}_{\tau\sim p(\tau|\boldsymbol{\theta}_t^{s+1})}[g_\omega(\tau|\boldsymbol{\theta}_{t-1}^{s+1})] = \mathbb{E}_{\tau\sim p(\tau|\boldsymbol{\theta}_{t-1}^{s+1})}[g(\tau|\boldsymbol{\theta}_{t-1}^{s+1})]$, which eliminates the inconsistency caused by the varying trajectory distribution.

Algorithm 7 Stochastic Recursive Variance Reduced Policy Gradient (SRVR-PG)

```
1: Input: number of epochs $S$, epoch size $m$, step size $\eta$, batch size $N$, mini-batch size $B$, gradient estimator $g$, initial parameter $\tilde{\boldsymbol{\theta}}^0 = \boldsymbol{\theta}_0 \in \boldsymbol{\Theta}$
2: for $s = 0, \ldots, S-1$ do
3:     $\boldsymbol{\theta}_0^{s+1} = \tilde{\boldsymbol{\theta}}^s$
4:     Sample $N$ trajectories $\{\tau_i\}$ from $p(\cdot|\tilde{\boldsymbol{\theta}}^s)$
5:     $\mathbf{v}_0^{s+1} = \hat{\nabla}_{\boldsymbol{\theta}} J(\tilde{\boldsymbol{\theta}}^s) := 1/N \sum_{i=1}^N g(\tau_i|\tilde{\boldsymbol{\theta}}^s)$
6:     $\boldsymbol{\theta}_1^{s+1} = \mathcal{P}_{\boldsymbol{\Theta}}(\boldsymbol{\theta}_0^{s+1} + \eta \mathbf{v}_0^{s+1})$
7:     for $t = 1, \ldots, m-1$ do
8:         Sample $B$ trajectories $\{\tau_j\}$ from $p(\cdot|\boldsymbol{\theta}_t^{s+1})$
9:         $\mathbf{v}_t^{s+1} = \mathbf{v}_{t-1}^{s+1} + \frac{1}{B}\sum_{j=1}^B \big(g\big(\tau_j|\boldsymbol{\theta}_t^{s+1}\big) - g_\omega\big(\tau_j|\boldsymbol{\theta}_{t-1}^{s+1}\big)\big)$
10:        $\boldsymbol{\theta}_{t+1}^{s+1} = \mathcal{P}_{\boldsymbol{\Theta}}(\boldsymbol{\theta}_t^{s+1} + \eta \mathbf{v}_t^{s+1})$
11:    end for
12:    $\tilde{\boldsymbol{\theta}}^{s+1} = \boldsymbol{\theta}_m^{s+1}$
13: end for
14: return $\boldsymbol{\theta}_{\text{out}}$, which is uniformly picked from $\{\boldsymbol{\theta}_t^s\}_{t=0,\ldots,m-1;s=0,\ldots,S}$
```

It is worth noting that the semi-stochastic gradient in (4.3.1) also differs from the one used in SVRPG [PBC+18] because we recursively update $\mathbf{v}_t^{s+1}$ using $\mathbf{v}_{t-1}^{s+1}$ from the previous iteration, while SVRPG uses a reference gradient that is only updated at the beginning of each epoch. Moreover, SVRPG wastes N trajectories without updating the policy at the beginning of each epoch, while Algorithm 7 updates the policy immediately after this sampling process (Line 6), which saves computation in practice.

We notice that very recently another algorithm called SARAPO [YLTZ19] is proposed which also uses a recursive gradient update in trust region policy optimization [SLA+15]. Our Algorithm 7 differs from their algorithm at least in the following ways: (1) our recursive gradient $\mathbf{v}_t^s$ defined in (4.3.1) has an importance weight from the snapshot gradient while SARAPO does not; (2) we are optimizing the expected return while [YLTZ19] optimizes the total advantage over state visitation distribution and actions under Kullback–Leibler divergence constraint; and most importantly (3) there is no convergence or sample complexity analysis for SARAPO.

4.4 Main Theory

In this section, we present the theoretical analysis of Algorithm 7. We first introduce some common assumptions used in the convergence analysis of policy gradient methods.

Assumption 4.4.1. *Let $\pi_{\boldsymbol{\theta}}(a|s)$ be the policy parameterized by $\boldsymbol{\theta}$. There exist constants $G, M > 0$ such that the gradient and Hessian matrix of $\log \pi_{\boldsymbol{\theta}}(a|s)$ with respect to $\boldsymbol{\theta}$ satisfy $\|\nabla_{\boldsymbol{\theta}} \log \pi_{\boldsymbol{\theta}}(a|s)\| \leq G$ and $\left\|\nabla_{\boldsymbol{\theta}}^2 \log \pi_{\boldsymbol{\theta}}(a|s)\right\|_2 \leq M$, for all $a \in \mathcal{A}$ and $s \in \mathcal{S}$.*

The above boundedness assumption is reasonable since we usually require the policy function to be twice differentiable and easy to optimize in practice. Similarly, in [PBC+18], the authors assume that $\frac{\partial}{\partial \theta_i} \log \pi_{\boldsymbol{\theta}}(a|s)$ and $\frac{\partial^2}{\partial \theta_i \partial \theta_j} \log \pi_{\boldsymbol{\theta}}(a|s)$ are upper bounded elementwisely, which is actually stronger than our Assumption 4.4.1.

In the following proposition, we show that Assumption4.4.1 directly implies that the Hessian matrix of the performance function $\nabla^2 J(\boldsymbol{\theta})$ is bounded, which is often referred to as the smoothness assumption and is crucial in analyzing the convergence of nonconvex optimization [RHS+16, AZH16].

Proposition 4.4.2. *Let $g(\tau|\boldsymbol{\theta})$ be the PGT estimator defined in (4.2.5). Assumption 4.4.1 implies: (1) $\|g(\tau|\boldsymbol{\theta}_1) - g(\tau|\boldsymbol{\theta}_2)\|_2 \leq L\|\boldsymbol{\theta}_1 - \boldsymbol{\theta}_2\|_2$, $\forall \boldsymbol{\theta}_1, \boldsymbol{\theta}_2 \in \mathbb{R}^d$, with $L = MR/(1-\gamma)^2$; (2) $J(\boldsymbol{\theta})$ is L-smooth, namely $\|\nabla_{\boldsymbol{\theta}}^2 J(\boldsymbol{\theta})\|_2 \leq L$; and (3) $\|g(\tau|\boldsymbol{\theta})\|_2 \leq C_g$ for all $\boldsymbol{\theta} \in \mathbb{R}^d$, with $C_g = GR/(1-\gamma)^2$.*

Similar properties are also proved in [XGG19]. However, in contrast to their results, the smoothness parameter L and the bound on the gradient norm here do not rely on horizon H and hence are tighter. The next assumption requires the variance of the gradient estimator is bounded.

Assumption 4.4.3. *There exists a constant $\xi > 0$ such that $Var\big(g(\tau|\boldsymbol{\theta})\big) \leq \xi^2$, for all policy $\pi_{\boldsymbol{\theta}}$.*

In Algorithm 7, we have used importance sampling to connect the trajectories between two different iterations. The following assumption ensures that the variance of the importance weight is bounded, which is also made in [PBC+18, XGG19].

Assumption 4.4.4. *Let $\omega(\cdot|\boldsymbol{\theta}_1, \boldsymbol{\theta}_2) = p(\cdot|\boldsymbol{\theta}_1)/p(\cdot|\boldsymbol{\theta}_2)$. There is a constant $W < \infty$ such that for each policy pairs encountered in Algorithm 7, $Var(\omega(\tau|\boldsymbol{\theta}_1, \boldsymbol{\theta}_2)) \le W, \quad \forall \boldsymbol{\theta}_1, \boldsymbol{\theta}_2 \in \mathbb{R}^d, \tau \sim p(\cdot|\boldsymbol{\theta}_2)$*

4.4.1 Convergence Rate and Sample Complexity of SRVR-PG

Now we are ready to present the convergence result of SRVR-PG to a stationary point:

Theorem 4.4.5. *Suppose that Assumptions 4.4.1, 4.4.3 and 4.4.4 hold. In Algorithm 7, we choose the step size $\eta \le 1/(4L)$ and epoch size m and mini-batch size B such that*

$$B \ge 72m\eta G^2(2G^2/M + 1)(W + 1)\gamma/(1 - \gamma)^3.$$

Then the generalized projected gradient of the output of Algorithm 7 satisfies

$$\mathbb{E}\big[\big\|\mathcal{G}_\eta\big(\boldsymbol{\theta}_{out}\big)\big\|_2^2\big] \le 8[J(\boldsymbol{\theta}^*) - J(\boldsymbol{\theta}_0) - \mathbb{1}_{\boldsymbol{\Theta}}(\boldsymbol{\theta}^*) + \mathbb{1}_{\boldsymbol{\Theta}}(\boldsymbol{\theta}_0)]/(\eta Sm) + 6\xi^2/N,$$

where $\boldsymbol{\theta}^* = \operatorname{argmax}_{\boldsymbol{\theta}\in\boldsymbol{\Theta}} J(\boldsymbol{\theta})$.

Remark 4.4.6. *Theorem 4.4.5 states that under a proper choice of step size, batch size and epoch length, the expected squared gradient norm of the performance function at the output of SRVR-PG is in the order of*

$$O\bigg(\frac{1}{Sm} + \frac{1}{N}\bigg).$$

Recall that S is the number of epochs and m is the epoch length of SRVR-PG, so Sm is the total number of iterations of SRVR-PG. Thus the first term $O(1/(Sm))$ characterizes the convergence rate of SRVR-PG. The second term $O(1/N)$ comes from the variance of the stochastic gradient used in the outer loop, where N is the batch size used in the snapshot gradient $\mathbf{v}_0^{s+1}$ in Line 5 of SRVR-PG. Compared with the $O(1/(Sm) + 1/N + 1/B)$ convergence rate in [PBC+18], our analysis avoids the additional term $O(1/B)$ that depends on the mini-batch size within each epoch.

Compared with [XGG19], our mini-batch size B is independent of the horizon length H. This enables us to choose a smaller mini-batch size B while maintaining the same convergence rate. As we will show in the next corollary, this improvement leads to a lower sample complexity.

Corollary 4.4.7. *Suppose the same conditions as in Theorem 4.4.5 hold. Set step size as $\eta = 1/(4L)$, the batch size parameters as $N = O(1/\epsilon)$ and $B = O(1/\epsilon^{1/2})$ respectively, epoch length as $m = O(1/\epsilon^{1/2})$ and the number of epochs as $S = O(1/\epsilon^{1/2})$. Then Algorithm 7 outputs a point $\boldsymbol{\theta}_{out}$ that satisfies $\mathbb{E}[\|\mathcal{G}_\eta(\boldsymbol{\theta}_{out})\|_2^2] \leq \epsilon$ within $O(1/\epsilon^{3/2})$ trajectories in total.*

Note that the results in [PBC+18, XGG19] are for $\|\nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta})\|_2^2 \leq \epsilon$, while our result in Corollary 4.4.7 is more general. In particular, when the policy parameter $\boldsymbol{\theta}$ is defined on the whole space $\mathbb{R}^d$ instead of $\boldsymbol{\Theta}$, our result reduces to the case for $\|\nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta})\|_2^2 \leq \epsilon$ since $\boldsymbol{\Theta} = \mathbb{R}^d$ and $\mathcal{G}_\eta(\boldsymbol{\theta}) = \nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta})$. In [XGG19], the authors improved the sample complexity of SVRPG [PBC+18] from $O(1/\epsilon^2)$ to $O(1/\epsilon^{5/3})$ by a sharper analysis. According to Corollary 4.4.7, SRVR-PG only needs $O(1/\epsilon^{3/2})$ number of trajectories to achieve $\|\nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta})\|_2^2 \leq \epsilon$, which is lower than the sample complexity of SVRPG by a factor of $O(1/\epsilon^{1/6})$. This improvement is more pronounced when the required precision ϵ is very small.

4.4.2 Implication for Gaussian Policy

Now, we consider the Gaussian policy model and present the sample complexity of SRVR-PG in this setting. For bounded action space $\mathcal{A} \subset \mathbb{R}$, a Gaussian policy parameterized by $\boldsymbol{\theta}$ is defined as

$$\pi_{\boldsymbol{\theta}}(a|s) = 1/\sqrt{2\pi} \exp\big(- (\boldsymbol{\theta}^\top \phi(s) - a)^2/(2\sigma^2)\big), \tag{4.4.1}$$

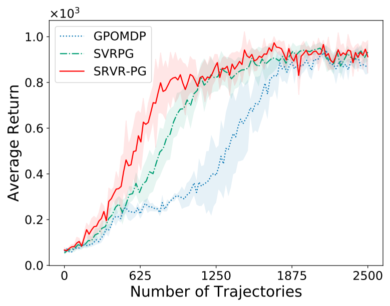
where σ^2 is a fixed standard deviation parameter and $\phi : \mathcal{S} \mapsto \mathbb{R}^d$ is a mapping from the state space to the feature space. For Gaussian policy, under the mild condition that the actions and the state feature vectors are bounded, we can verify that Assumptions 4.4.1 and 4.4.3 hold, which can be found in Appendix 4.9. It is worth noting that Assumption 4.4.4 does not hold trivially for all Gaussian distributions. In particular, [CMM10] showed that for two

Gaussian distributions $\pi_{\boldsymbol{\theta}_1}(a|s) \sim N(\mu_1, \sigma_1^2)$ and $\pi_{\boldsymbol{\theta}_2}(a|s) \sim N(\mu_2, \sigma_2^2)$, if $\sigma_2 > \sqrt{2}/2\sigma_1$, then the variance of $\omega(\tau|\boldsymbol{\theta}_1, \boldsymbol{\theta}_2)$ is bounded. For our Gaussian policy defined in (4.4.1) where the standard deviation σ^2 is fixed, we have $\sigma > \sqrt{2}/2\sigma$ trivially hold, and therefore Assumption 4.4.4 holds for some finite constant $W > 0$ according to (4.2.1).

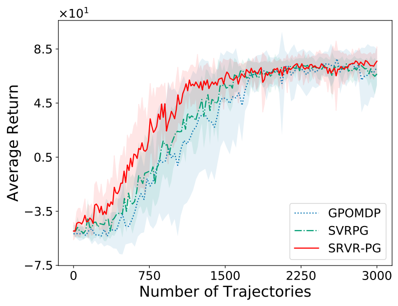
Recall that Theorem 4.4.5 holds for any general models under Assumptions 4.4.1, 4.4.3 and 4.4.4. Based on the above arguments, we know that the convergence analysis in Theorem 4.4.5 applies to Gaussian policy. In the following corollary, we present the sample complexity of Algorithm 7 for Gaussian policy with detailed dependency on precision parameter ϵ, horizon size H and the discount factor γ.

Corollary 4.4.8. *Given the Gaussian policy defined in* (4.4.1)*, suppose Assumption 4.4.4 holds and we have* $|a| \le C_a$ *for all* $a \in \mathcal{A}$ *and* $\|\phi(s)\|_2 \le M_\phi$ *for all* $s \in \mathcal{S}$*, where* $C_a, M_\phi > 0$ *are constants. If we set step size as* $\eta = O((1-\gamma)^2)$*, the mini-batch sizes and epoch length as* $N = O((1-\gamma)^{-3}\epsilon^{-1})$*,* $B = O((1-\gamma)^{-2}\epsilon^{-1/2})$ *and* $m = O((1-\gamma)^{-1}\epsilon^{-1/2})$*, then the output of Algorithm 7 satisfies* $\mathbb{E}[\|\mathcal{G}_\eta(\boldsymbol{\theta}_{out})\|_2^2] \le \epsilon$ *after* $O(1/((1-\gamma)^4\epsilon^{3/2}))$ *trajectories in total.*

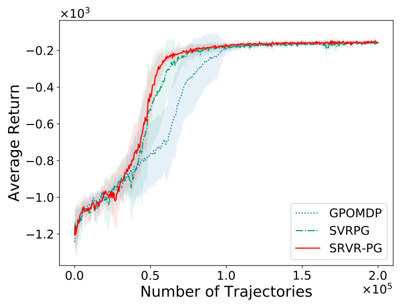
Remark 4.4.9. *For Gaussian policy, the number of trajectories Algorithm 7 needs to find an* ϵ*-approximate stationary point, i.e.,* $\mathbb{E}[\|\mathcal{G}_\eta(\boldsymbol{\theta}_{out})\|_2^2] \le \epsilon$*, is also in the order of* $O(\epsilon^{-3/2})$*, which is faster than PGT and SVRPG. Additionally, we explicitly show that the sample complexity does not depend on the horizon* H*, which is in sharp contrast with the results in [PBC+18, XGG19]. The dependence on* $1/(1-\gamma)$ *comes from the variance of PGT estimator.*



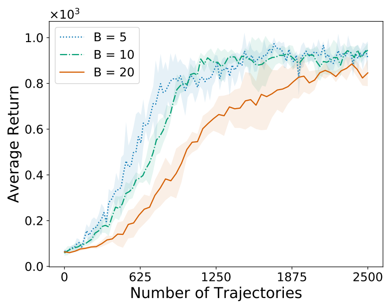
(a) Cartpole



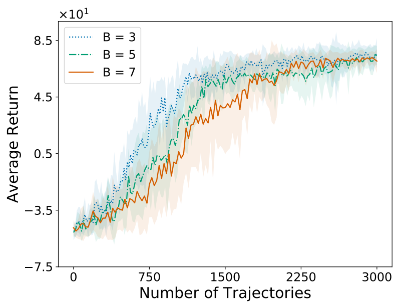
(b) Mountain Car



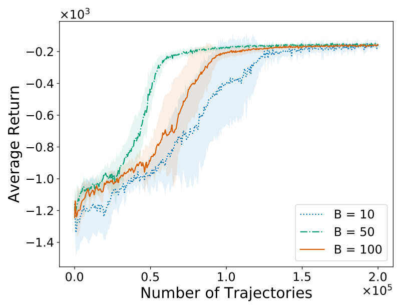
(c) Pendulum



(d) Cartpole



(e) Mountain Car



(f) Pendulum

Figure 4.1: (a)-(c): Comparison of different algorithms. Experimental results are averaged over 10 repetitions. (d)-(f): Comparison of different batch size B on the performance of SRVR-PG.

4.5 Experiments

In this section, we provide experiment results of the proposed algorithm on benchmark reinforcement learning environments including the Cartpole, Mountain Car and Pendulum problems. In all the experiments, we use the Gaussian policy defined in (4.4.1). In addition, we found that the proposed algorithm works well without the extra projection step. Therefore, we did not use projection in our experiments. For baselines, we compare the proposed SRVR-PG algorithm with the most relevant methods: GPOMDP [BB01] and SVRPG [PBC+18]. For the learning rates η in all of our experiments, we use grid search to directly tune η. For instance, we searched η for the Cartpole problem by evenly dividing the interval $[10^{-5}, 10^{-1}]$ into 20 points in the log-space. For the batch size parameters N and B and the epoch length m, according to Corollary 4.7, we choose $N = O(1/\epsilon)$, $B = O(1/\epsilon^{1/2})$ and thus $m = O(1/\epsilon^{1/2})$, where $\epsilon > 0$ is a user-defined precision parameter. In our experiments, we set $N = C_0/\epsilon$, $B = C_1/\epsilon^{1/2}$ and $m = C_2/\epsilon^{1/2}$ and tune the constant parameters C_0, C_1, C_2 using grid search. The detailed parameters used in the experiments are presented in Appendix 4.10.

We evaluate the performance of different algorithms in terms of the total number of trajectories they require to achieve a certain threshold of cumulative rewards. We run each experiment repeatedly for 10 times and plot the averaged returns with standard deviation. For a given environment, all experiments are initialized from the same random initialization. Figures 4.1(a), 4.1(b) and 4.1(c) show the results on the comparison of GPOMDP, SVRPG, and our proposed SRVR-PG algorithm across three different RL environments. It is evident that, for all environments, GPOMDP is overshadowed by the variance reduced algorithms SVRPG and SRVR-PG significantly. Furthermore, SRVR-PG outperforms SVRPG in all experiments, which is consistent with the comparison on the sample complexity of GPOMDP, SRVRPG and SRVR-PG in Table 4.1.

Corollaries 4.4.7 and 4.4.8 suggest that when the mini-batch size B is in the order of $O(\sqrt{N})$, SRVR-PG achieves the best performance. Here N is the number of episodes sampled in the outer loop of Algorithm 7 and B is the number of episodes sampled at each inner loop iteration. To validate our theoretical result, we conduct a sensitivity study to demonstrate the effectiveness of different batch sizes within each epoch of SRVR-PG on its performance. The results on different environments are displayed in Figures 4.1(d), 4.1(e) and 4.1(f) respectively. To interpret these results, we take the Pendulum problem as an example. In this setting, we choose outer loop batch size N of Algorithm 7 to be $N = 250$. By Corollary 4.4.8, the optimal choice of batch size in the inner loop of Algorithm 7 is $B = C\sqrt{N}$, where $C > 1$ is a constant depending on horizon H and discount factor γ. Figure 4.1(f) shows that $B = 50 \approx 3\sqrt{N}$ yields the best convergence results for SRVR-PG on Pendulum, which validates our theoretical analysis and implies that a larger batch size B does not necessarily result in an improvement in sample complexity, as each update requires more trajectories, but a smaller batch size B pushes SRVR-PG to behave more similar to

GPOMDP. Moreover, by comparing with the outer loop batch size N presented in Table 4.3 for SRVR-PG in Cartpole and Mountain Car environments, we found that the results in Figures 4.1(d) and 4.1(e) are again in alignment with our theory. Due to the space limit, additional experiment results are included in Appendix 4.10.

4.6 Extension to Parameter-based Exploration

Although SRVR-PG is proposed for action-based policy gradient, it can be easily extended to the policy gradient algorithm with parameter-based exploration (PGPE) [SOR$^+$08]. Unlike action-based policy gradient in previous sections, PGPE does not directly optimize the policy parameter $\boldsymbol{\theta}$ but instead assumes that it follows a prior distribution with hyper-parameter $\boldsymbol{\rho}$: $\boldsymbol{\theta} \sim p(\boldsymbol{\theta}|\boldsymbol{\rho})$. The expected return under the policy induced by the hyper-parameter $\boldsymbol{\rho}$ is formulated as follows[1]

$$J(\boldsymbol{\rho}) = \int\int p(\boldsymbol{\theta}|\boldsymbol{\rho})p(\tau|\boldsymbol{\theta})\mathcal{R}(\tau)\mathrm{d}\tau\mathrm{d}\boldsymbol{\theta}. \tag{4.6.1}$$

PGPE aims to find the hyper-parameter $\boldsymbol{\rho}^*$ that maximizes the performance function $J(\boldsymbol{\rho})$. Since $p(\boldsymbol{\theta}|\boldsymbol{\rho})$ is stochastic and can provide sufficient exploration, we can choose $\pi_{\boldsymbol{\theta}}(a|s) = \delta(a - \mu_{\boldsymbol{\theta}}(s))$ to be a deterministic policy, where δ is the Dirac delta function and $\mu_{\boldsymbol{\theta}}(\cdot)$ is a deterministic function. For instance, a linear deterministic policy is defined as $\pi_{\boldsymbol{\theta}}(a|s) = \delta(a - \boldsymbol{\theta}^\top s)$ [ZHNS11, MPFR18]. Given the policy parameter $\boldsymbol{\theta}$, a trajectory τ is only decided by the initial state distribution and the transition probability. Therefore, PGPE is called a parameter-based exploration approach. Similar to the action-based policy gradient methods, we can apply gradient ascent to find $\boldsymbol{\rho}^*$. In the k-th iteration, we update $\boldsymbol{\rho}_k$ by $\boldsymbol{\rho}_{k+1} = \boldsymbol{\rho}_k + \eta\nabla_{\boldsymbol{\rho}}J(\boldsymbol{\rho})$. The exact gradient of $J(\boldsymbol{\rho})$ with respect to $\boldsymbol{\rho}$ is given by

$$\nabla_{\boldsymbol{\rho}}J(\boldsymbol{\rho}) = \int\int p(\boldsymbol{\theta}|\boldsymbol{\rho})p(\tau|\boldsymbol{\theta})\nabla_{\boldsymbol{\rho}}\log p(\boldsymbol{\theta}|\boldsymbol{\rho})\mathcal{R}(\tau)\mathrm{d}\tau\mathrm{d}\boldsymbol{\theta}.$$

To approximate $\nabla_{\boldsymbol{\rho}}J(\boldsymbol{\rho})$, we first sample N policy parameters $\{\boldsymbol{\theta}_i\}$ from $p(\boldsymbol{\theta}|\boldsymbol{\rho})$. Then we sample one trajectory τ_i for each $\boldsymbol{\theta}_i$ and use the following empirical average to approximate

[1]We slightly abuse the notation by overloading J as the performance function defined on the hyper-parameter $\boldsymbol{\rho}$.

$\nabla_{\boldsymbol{\rho}} J(\boldsymbol{\rho})$

$$\hat{\nabla}_{\boldsymbol{\rho}} J(\boldsymbol{\rho}) = \frac{1}{N}\sum_{i=1}^{N} \nabla_{\boldsymbol{\rho}} \log p(\boldsymbol{\theta}_i|\boldsymbol{\rho}) \sum_{h=0}^{H} \gamma^h r(s_h^i, a_h^i) := \frac{1}{N}\sum_{i=1}^{N} g(\tau_i|\boldsymbol{\rho}), \tag{4.6.2}$$

where $\gamma \in [0,1)$ is the discount factor. Compared with the PGT/GPOMDP estimator in Section 4.2, the likelihood term $\nabla_{\boldsymbol{\rho}} \log p(\boldsymbol{\theta}_i|\boldsymbol{\rho})$ in (4.6.2) for PGPE is independent of horizon H.

Algorithm 7 can be directly applied to the PGPE setting, where we replace the policy parameter $\boldsymbol{\theta}$ with the hyper-parameter $\boldsymbol{\rho}$. When we need to sample N trajectories, we first sample N policy parameters $\{\boldsymbol{\theta}_i\}$ from $p(\boldsymbol{\theta}|\boldsymbol{\rho})$. Since the policy is deterministic with given $\boldsymbol{\theta}_i$, we sample one trajectory τ_i from each policy $p(\tau|\boldsymbol{\theta}_i)$. The recursive semi-stochastic gradient is given by

$$\mathbf{v}_t^{s+1} = \frac{1}{B}\sum_{j=1}^{B} g(\tau_j|\boldsymbol{\rho}_t^{s+1}) - \frac{1}{B}\sum_{j=1}^{B} g_\omega(\tau_j|\boldsymbol{\rho}_{t-1}^{s+1}) + \mathbf{v}_{t-1}^{s+1}, \tag{4.6.3}$$

where $g_\omega(\tau_j|\boldsymbol{\rho}_{t-1}^{s+1})$ is the gradient estimator with step-wise importance weight defined in the way as in (4.3.2). We call this variance reduced parameter-based algorithm SRVR-PG-PE, which is displayed in Algorithm 8.

Under similar assumptions on the parameter distribution $p(\boldsymbol{\theta}|\boldsymbol{\rho})$, as Assumptions 4.4.1, 4.4.3 and 4.4.4, we can easily prove that SRVR-PG-PE converges to a stationary point of $J(\boldsymbol{\rho})$ with $O(1/\epsilon^{3/2})$ sample complexity. In particular, we assume the policy parameter $\boldsymbol{\theta}$ follows the distribution $p(\boldsymbol{\theta}|\boldsymbol{\rho})$ and we update our estimation of $\boldsymbol{\rho}$ based on the semi-stochastic gradient in (4.6.3). Recall the gradient $\hat{\nabla}_{\boldsymbol{\rho}} J(\boldsymbol{\rho})$ derived in (4.6.2). Since the policy in SRVR-PG-PE is deterministic, we only need to make the boundedness assumption on $p(\boldsymbol{\theta}|\boldsymbol{\rho})$. In particular, we assume that

1. $\|\nabla_{\boldsymbol{\rho}} \log p(\boldsymbol{\theta}|\boldsymbol{\rho})\|_2$ and $\|\nabla_{\boldsymbol{\rho}}^2 \log p(\boldsymbol{\theta}|\boldsymbol{\rho})\|_2$ are bounded by constants in a similar way to Assumption 4.4.1;

2. the gradient estimator $g(\tau|\boldsymbol{\rho}) = \nabla_{\boldsymbol{\rho}} \log p(\boldsymbol{\theta}|\boldsymbol{\rho}) \sum_{h=0}^{H} \gamma^h r(s_h, a_h)$ has bounded variance;

3. and the importance weight $\omega(\tau_j|\boldsymbol{\rho}_{t-1}^{s+1}, \boldsymbol{\rho}_t^{s+1}) = p(\boldsymbol{\theta}_j|\boldsymbol{\rho}_{t-1}^{s+1})/p(\boldsymbol{\theta}_j|\boldsymbol{\rho}_t^{s+1})$ has bounded variance in a similar way to Assumption 4.4.4.

Then the same gradient complexity $O(1/\epsilon^{3/2})$ for SRVR-PG-PE can be proved in the same way as the proof of Theorem 4.4.5 and Corollary 4.4.7. Since the analysis is almost the same as that of SRVR-PG, we omit the proof of the convergence of SRVR-PG-PE. In fact, according to the analysis in [ZHNS11, MPFR18], all the three assumptions listed above can be easily verified under a Gaussian prior for $\boldsymbol{\theta}$ and a linear deterministic policy.

Algorithm 8 Stochastic Recursive Variance Reduced Policy Gradient with Parameter-based Exploration (SRVR-PG-PE)

```
 1: Input: number of epochs $S$, epoch size $m$, step size $\eta$, batch size $N$, mini-batch size $B$,
    gradient estimator $g$, initial parameter $\boldsymbol{\rho}_m^0 := \tilde{\boldsymbol{\rho}}^0 := \boldsymbol{\rho}_0$
 2: for $s = 0, \dots, S-1$ do
 3:   $\boldsymbol{\rho}_0^{s+1} = \boldsymbol{\rho}^s$
 4:   Sample $N$ policy parameters $\{\boldsymbol{\theta}_i\}$ from $p(\cdot|\boldsymbol{\rho}^s)$
 5:   Sample one trajectory $\tau_i$ from each policy $\pi_{\boldsymbol{\theta}_i}$
 6:   $\mathbf{v}_0^{s+1} = \hat{\nabla}_{\boldsymbol{\rho}} J(\boldsymbol{\rho}^s) := \frac{1}{N}\sum_{i=1}^N g(\tau_i|\tilde{\boldsymbol{\rho}}^s)$
 7:   $\boldsymbol{\rho}_1^{s+1} = \boldsymbol{\rho}_0^{s+1} + \eta\mathbf{v}_0^{s+1}$
 8:   for $t = 1, \dots, m-1$ do
 9:     Sample $B$ policy parameters $\{\boldsymbol{\theta}_j\}$ from $p(\cdot|\boldsymbol{\rho}_t^{s+1})$
10:     Sample one trajectory $\tau_j$ from each policy $\pi_{\boldsymbol{\theta}_j}$
11:     $\mathbf{v}_t^{s+1} = \mathbf{v}_{t-1}^{s+1} + \frac{1}{B}\sum_{j=1}^B \big(g(\tau_j|\boldsymbol{\rho}_t^{s+1}) - g_\omega(\tau_j|\boldsymbol{\rho}_{t-1}^{s+1})\big)$
12:     $\boldsymbol{\rho}_{t+1}^{s+1} = \boldsymbol{\rho}_t^{s+1} + \eta\mathbf{v}_t^{s+1}$
13:   end for
14: end for
15: return $\boldsymbol{\rho}_{\mathrm{out}}$, which is uniformly picked from $\{\boldsymbol{\rho}_t^s\}_{t=0,\dots,m;s=0,\dots,S}$
```

4.7 Proof of the Main Theory

In this section, we provide the proofs of the theoretical results for SRVR-PG (Algorithm 7). Before we start the proof of Theorem 4.4.5, we first lay down the following key lemma that controls the variance of the importance sampling weight ω.

Lemma 4.7.1. *For any $\boldsymbol{\theta}_1, \boldsymbol{\theta}_2 \in \mathbb{R}^d$, let $\omega_{0:h}\big(\tau|\boldsymbol{\theta}_1, \boldsymbol{\theta}_2\big) = p(\tau_h|\boldsymbol{\theta}_1)/p(\tau_h|\boldsymbol{\theta}_2)$, where τ_h is a truncated trajectory of τ up to step h. Under Assumptions 4.4.1 and 4.4.4, it holds that*

$$Var\big(\omega_{0:h}\big(\tau|\boldsymbol{\theta}_1, \boldsymbol{\theta}_2\big)\big) \leq C_\omega \|\boldsymbol{\theta}_1 - \boldsymbol{\theta}_2\|_2^2,$$

where $C_\omega = h(2hG^2 + M)(W + 1)$.

Recall that in Assumption 4.4.4 we assume the variance of the importance weight is upper bounded by a constant W. Based on this assumption, Lemma 4.7.1 further bounds the variance of the importance weight via the distance between the behavioral and the target policies. As the algorithm converges, these two policies will be very close and the bound in Lemma 4.7.1 could be much tighter than the constant bound.

Proof of Theorem 4.4.5. By plugging the definition of the projection operator in (4.3.3) into the update rule $\boldsymbol{\theta}_{t+1}^{s+1} = \mathcal{P}_{\boldsymbol{\Theta}}\big(\boldsymbol{\theta}_t^{s+1} + \eta \mathbf{v}_t^{s+1}\big)$, we have

$$\boldsymbol{\theta}_{t+1}^{s+1} = \operatorname*{argmin}_{\mathbf{u} \in \mathbb{R}^d} \mathbb{1}_{\boldsymbol{\Theta}}(\mathbf{u}) + 1/(2\eta)\big\|\mathbf{u} - \boldsymbol{\theta}_t^{s+1}\big\|_2^2 - \langle \mathbf{v}_t^{s+1}, \mathbf{u}\rangle. \tag{4.7.1}$$

Similar to the generalized projected gradient $\mathcal{G}_\eta(\boldsymbol{\theta})$ defined in (4.3.4), we define $\tilde{\mathcal{G}}_t^{s+1}$ to be a (stochastic) gradient mapping based on the recursive gradient estimator $\mathbf{v}_t^{s+1}$:

$$\tilde{\mathcal{G}}_t^{s+1} = \frac{1}{\eta}\big(\boldsymbol{\theta}_{t+1}^{s+1} - \boldsymbol{\theta}_t^{s+1}\big) = \frac{1}{\eta}\big(\mathcal{P}_{\boldsymbol{\Theta}}\big(\boldsymbol{\theta}_t^{s+1} + \eta \mathbf{v}_t^{s+1}\big) - \boldsymbol{\theta}_t^{s+1}\big). \tag{4.7.2}$$

The definition of $\tilde{\mathcal{G}}_t^{s+1}$ differs from $\mathcal{G}_\eta(\boldsymbol{\theta}_t^{s+1})$ only in the semi-stochastic gradient term $\mathbf{v}_t^{s+1}$, while the latter one uses the full gradient $\nabla J(\boldsymbol{\theta}_t^{s+1})$. Note that $\mathbb{1}_{\boldsymbol{\Theta}}(\cdot)$ is convex but not smooth. We assume that $\mathbf{p} \in \partial \mathbb{1}_{\boldsymbol{\Theta}}(\boldsymbol{\theta}_{t+1}^{s+1})$ is a sub-gradient of $\mathbb{1}_{\boldsymbol{\Theta}}(\cdot)$. According to the optimality condition of (4.7.1), we have $\mathbf{p} + 1/\eta(\boldsymbol{\theta}_{t+1}^{s+1} - \boldsymbol{\theta}_t^{s+1}) - \mathbf{v}_t^{s+1} = \mathbf{0}$. Further by the convexity of $\mathbb{1}_{\boldsymbol{\Theta}}(\cdot)$, we have

$$\begin{aligned}\mathbb{1}_{\boldsymbol{\Theta}}(\boldsymbol{\theta}_{t+1}^{s+1}) &\leq \mathbb{1}_{\boldsymbol{\Theta}}(\boldsymbol{\theta}_t^{s+1}) + \langle \mathbf{p}, \boldsymbol{\theta}_{t+1}^{s+1} - \boldsymbol{\theta}_t^{s+1}\rangle \\ &= \mathbb{1}_{\boldsymbol{\Theta}}(\boldsymbol{\theta}_t^{s+1}) - \langle 1/\eta(\boldsymbol{\theta}_{t+1}^{s+1} - \boldsymbol{\theta}_t^{s+1}) - \mathbf{v}_t^{s+1}, \boldsymbol{\theta}_{t+1}^{s+1} - \boldsymbol{\theta}_t^{s+1}\rangle. \end{aligned} \tag{4.7.3}$$

By Proposition 4.4.2, $J(\boldsymbol{\theta})$ is L-smooth, which by definition directly implies

$$J\big(\boldsymbol{\theta}_{t+1}^{s+1}\big) \geq J\big(\boldsymbol{\theta}_t^{s+1}\big) + \big\langle \nabla J\big(\boldsymbol{\theta}_t^{s+1}\big), \boldsymbol{\theta}_{t+1}^{s+1} - \boldsymbol{\theta}_t^{s+1}\big\rangle - \frac{L}{2}\big\|\boldsymbol{\theta}_{t+1}^{s+1} - \boldsymbol{\theta}_t^{s+1}\big\|_2^2.$$

For the simplification of presentation, let us define the notation $\Phi(\boldsymbol{\theta}) = J(\boldsymbol{\theta}) - \mathbb{1}_{\boldsymbol{\Theta}}(\boldsymbol{\theta})$. Then according to the definition of $\mathbb{1}_{\boldsymbol{\Theta}}$ we have $\operatorname{argmax}_{\boldsymbol{\theta}\in\mathbb{R}^d} \Phi(\boldsymbol{\theta}) = \operatorname{argmax}_{\boldsymbol{\theta}\in\boldsymbol{\Theta}} J(\boldsymbol{\theta}) := \boldsymbol{\theta}^*$. Combining the above inequality with (4.7.3), we have

$$\begin{aligned}
\Phi\big(\boldsymbol{\theta}_{t+1}^{s+1}\big) &\geq \Phi\big(\boldsymbol{\theta}_t^{s+1}\big) + \big\langle \nabla J\big(\boldsymbol{\theta}_t^{s+1}\big) - \mathbf{v}_t^{s+1}, \boldsymbol{\theta}_{t+1}^{s+1} - \boldsymbol{\theta}_t^{s+1}\big\rangle + \bigg(\frac{1}{\eta} - \frac{L}{2}\bigg)\big\|\boldsymbol{\theta}_{t+1}^{s+1} - \boldsymbol{\theta}_t^{s+1}\big\|_2^2 \\
&= \Phi\big(\boldsymbol{\theta}_t^{s+1}\big) + \big\langle \nabla J\big(\boldsymbol{\theta}_t^{s+1}\big) - \mathbf{v}_t^{s+1}, \eta\tilde{\mathcal{G}}_t^{s+1}\big\rangle + \eta\big\|\tilde{\mathcal{G}}_t^{s+1}\big\|_2^2 - \frac{L}{2}\big\|\boldsymbol{\theta}_{t+1}^{s+1} - \boldsymbol{\theta}_t^{s+1}\big\|_2^2 \\
&\geq \Phi\big(\boldsymbol{\theta}_t^{s+1}\big) - \frac{\eta}{2}\big\|\nabla J\big(\boldsymbol{\theta}_t^{s+1}\big) - \mathbf{v}_t^{s+1}\big\|_2^2 + \frac{\eta}{2}\big\|\tilde{\mathcal{G}}_t^{s+1}\big\|_2^2 - \frac{L}{2}\big\|\boldsymbol{\theta}_{t+1}^{s+1} - \boldsymbol{\theta}_t^{s+1}\big\|_2^2 \\
&= \Phi\big(\boldsymbol{\theta}_t^{s+1}\big) - \frac{\eta}{2}\big\|\nabla J\big(\boldsymbol{\theta}_t^{s+1}\big) - \mathbf{v}_t^{s+1}\big\|_2^2 + \frac{\eta}{4}\big\|\tilde{\mathcal{G}}_t^{s+1}\big\|_2^2 + \bigg(\frac{1}{4\eta} - \frac{L}{2}\bigg)\big\|\boldsymbol{\theta}_{t+1}^{s+1} - \boldsymbol{\theta}_t^{s+1}\big\|_2^2 \\
&\geq \Phi\big(\boldsymbol{\theta}_t^{s+1}\big) - \frac{\eta}{2}\big\|\nabla J\big(\boldsymbol{\theta}_t^{s+1}\big) - \mathbf{v}_t^{s+1}\big\|_2^2 + \frac{\eta}{8}\big\|\mathcal{G}_\eta\big(\boldsymbol{\theta}_t^{s+1}\big)\big\|_2^2 \\
&\qquad - \frac{\eta}{4}\|\mathcal{G}_\eta(\boldsymbol{\theta}_t^{s+1}) - \tilde{\mathcal{G}}_t^{s+1}\|_2^2 + \bigg(\frac{1}{4\eta} - \frac{L}{2}\bigg)\big\|\boldsymbol{\theta}_{t+1}^{s+1} - \boldsymbol{\theta}_t^{s+1}\big\|_2^2,
\end{aligned} \tag{4.7.4}$$

where the second inequality holds due to Young's inequality and the third inequality holds due to the fact that $\|\mathcal{G}_\eta(\boldsymbol{\theta}_t^{s+1})\|_2^2 \leq 2\|\tilde{\mathcal{G}}_t^{s+1}\|_2^2 + 2\|\mathcal{G}_\eta(\boldsymbol{\theta}_t^{s+1}) - \tilde{\mathcal{G}}_t^{s+1}\|_2^2$. Denote $\bar{\boldsymbol{\theta}}_{t+1}^{s+1} = \operatorname{prox}_{\eta\mathbb{1}_{\boldsymbol{\Theta}}}(\boldsymbol{\theta}_t^{s+1} + \eta\nabla J(\boldsymbol{\theta}_t^{s+1}))$. By similar argument in (4.7.3) we have

$$\begin{aligned}
\mathbb{1}_{\boldsymbol{\Theta}}(\boldsymbol{\theta}_{t+1}^{s+1}) &\leq \mathbb{1}_{\boldsymbol{\Theta}}(\bar{\boldsymbol{\theta}}_{t+1}^{s+1}) - \langle 1/\eta(\boldsymbol{\theta}_{t+1}^{s+1} - \boldsymbol{\theta}_t^{s+1}) - \mathbf{v}_t^{s+1}, \boldsymbol{\theta}_{t+1}^{s+1} - \bar{\boldsymbol{\theta}}_{t+1}^{s+1}\rangle, \\
\mathbb{1}_{\boldsymbol{\Theta}}(\bar{\boldsymbol{\theta}}_{t+1}^{s+1}) &\leq \mathbb{1}_{\boldsymbol{\Theta}}(\boldsymbol{\theta}_{t+1}^{s+1}) - \langle 1/\eta(\bar{\boldsymbol{\theta}}_{t+1}^{s+1} - \boldsymbol{\theta}_t^{s+1}) - \nabla J\big(\boldsymbol{\theta}_t^{s+1}\big), \bar{\boldsymbol{\theta}}_{t+1}^{s+1} - \boldsymbol{\theta}_{t+1}^{s+1}\rangle.
\end{aligned}$$

Adding the above two inequalities immediately yields $\|\bar{\boldsymbol{\theta}}_{t+1}^{s+1} - \boldsymbol{\theta}_{t+1}^{s+1}\|_2 \leq \eta\|\nabla J(\boldsymbol{\theta}_t^{s+1}) - \mathbf{v}_t^{s+1}\|_2$, which further implies $\|\mathcal{G}_\eta(\boldsymbol{\theta}_t^{s+1}) - \tilde{\mathcal{G}}_t^{s+1}\|_2 \leq \|\nabla J(\boldsymbol{\theta}_t^{s+1}) - \mathbf{v}_t^{s+1}\|_2$. Submitting this result into (4.7.4), we obtain

$$\begin{aligned}
\Phi\big(\boldsymbol{\theta}_{t+1}^{s+1}\big) \geq \Phi\big(\boldsymbol{\theta}_t^{s+1}\big) - \frac{3\eta}{4}\big\|\nabla J\big(\boldsymbol{\theta}_t^{s+1}\big) - \mathbf{v}_t^{s+1}\big\|_2^2 + \frac{\eta}{8}\big\|\mathcal{G}_\eta\big(\boldsymbol{\theta}_t^{s+1}\big)\big\|_2^2 \\
+ \bigg(\frac{1}{4\eta} - \frac{L}{2}\bigg)\big\|\boldsymbol{\theta}_{t+1}^{s+1} - \boldsymbol{\theta}_t^{s+1}\big\|_2^2.
\end{aligned} \tag{4.7.5}$$

We denote the index set of $\{\tau_j\}_{j=1}^B$ in the t-th inner iteration by $\mathcal{B}_t$. Note that

$$\begin{aligned}
&\big\|\nabla J\big(\boldsymbol{\theta}_t^{s+1}\big) - \mathbf{v}_t^{s+1}\big\|_2^2 \\
&= \bigg\|\nabla J\big(\boldsymbol{\theta}_t^{s+1}\big) - \mathbf{v}_{t-1}^{s+1} + \frac{1}{B}\sum_{j\in\mathcal{B}_t}\big(g_\omega\big(\tau_j|\boldsymbol{\theta}_{t-1}^{s+1}\big) - g\big(\tau_j|\boldsymbol{\theta}_t^{s+1}\big)\big)\bigg\|_2^2 \\
&= \bigg\|\nabla J\big(\boldsymbol{\theta}_t^{s+1}\big) - \nabla J(\boldsymbol{\theta}_{t-1}^{s+1}) + \frac{1}{B}\sum_{j\in\mathcal{B}_t}\big(g_\omega\big(\tau_j|\boldsymbol{\theta}_{t-1}^{s+1}\big) - g\big(\tau_j|\boldsymbol{\theta}_t^{s+1}\big)\big) + \nabla J(\boldsymbol{\theta}_{t-1}^{s+1}) - \mathbf{v}_{t-1}^{s+1}\bigg\|_2^2
\end{aligned}$$

$$
\begin{aligned}
&= \bigg\|\nabla J\big(\boldsymbol{\theta}_t^{s+1}\big) - \nabla J(\boldsymbol{\theta}_{t-1}^{s+1}) + \frac{1}{B}\sum_{j\in\mathcal{B}_t}\big(g_\omega\big(\tau_j|\boldsymbol{\theta}_{t-1}^{s+1}\big) - g\big(\tau_j|\boldsymbol{\theta}_t^{s+1}\big)\big)\bigg\|^2 \\
&\quad + \frac{2}{B}\sum_{j\in\mathcal{B}_t}\big\langle\nabla J\big(\boldsymbol{\theta}_t^{s+1}\big) - \nabla J(\boldsymbol{\theta}_{t-1}^{s+1}) + g_\omega\big(\tau_j|\boldsymbol{\theta}_{t-1}^{s+1}\big) - g\big(\tau_j|\boldsymbol{\theta}_t^{s+1}\big), \nabla J(\boldsymbol{\theta}_{t-1}^{s+1}) - \mathbf{v}_{t-1}^{s+1}\big\rangle \\
&\quad + \big\|\nabla J(\boldsymbol{\theta}_{t-1}^{s+1}) - \mathbf{v}_{t-1}^{s+1}\big\|_2^2.
\end{aligned}
\tag{4.7.6}
$$

Conditional on $\boldsymbol{\theta}_t^{s+1}$, taking the expectation over $\mathcal{B}_t$ yields

$$
\mathbb{E}\big[\big\langle\nabla J\big(\boldsymbol{\theta}_t^{s+1}\big) - g\big(\tau_j|\boldsymbol{\theta}_t^{s+1}\big), \nabla J(\boldsymbol{\theta}_{t-1}^{s+1}) - \mathbf{v}_{t-1}^{s+1}\big\rangle\big] = 0.
$$

Similarly, taking the expectation over $\boldsymbol{\theta}_t^{s+1}$ and the choice of $\mathcal{B}_t$ yields

$$
\mathbb{E}\big[\big\langle\nabla J(\boldsymbol{\theta}_{t-1}^{s+1}) - g_\omega\big(\tau_j|\boldsymbol{\theta}_{t-1}^{s+1}\big), \nabla J(\boldsymbol{\theta}_{t-1}^{s+1}) - \mathbf{v}_{t-1}^{s+1}\big\rangle\big] = 0.
$$

Combining the above equations with (4.7.6), we obtain

$$
\begin{aligned}
&\mathbb{E}\big[\big\|\nabla J\big(\boldsymbol{\theta}_t^{s+1}\big) - \mathbf{v}_t^{s+1}\big\|_2^2\big] \\
&= \mathbb{E}\bigg\|\nabla J\big(\boldsymbol{\theta}_t^{s+1}\big) - \nabla J(\boldsymbol{\theta}_{t-1}^{s+1}) + \frac{1}{B}\sum_{j\in\mathcal{B}_t}\big(g_\omega\big(\tau_j|\boldsymbol{\theta}_{t-1}^{s+1}\big) - g\big(\tau_j|\boldsymbol{\theta}_t^{s+1}\big)\big)\bigg\|_2^2 \\
&\qquad + \mathbb{E}\big\|\nabla J(\boldsymbol{\theta}_{t-1}^{s+1}) - \mathbf{v}_{t-1}^{s+1}\big\|_2^2 \\
&= \frac{1}{B^2}\sum_{j\in\mathcal{B}_t}\mathbb{E}\big\|\nabla J\big(\boldsymbol{\theta}_t^{s+1}\big) - \nabla J(\boldsymbol{\theta}_{t-1}^{s+1}) + g_\omega\big(\tau_j|\boldsymbol{\theta}_{t-1}^{s+1}\big) - g\big(\tau_j|\boldsymbol{\theta}_t^{s+1}\big)\big\|_2^2 \\
&\qquad + \mathbb{E}\big\|\nabla J(\boldsymbol{\theta}_{t-1}^{s+1}) - \mathbf{v}_{t-1}^{s+1}\big\|_2^2, && (4.7.7) \\
&\le \frac{1}{B^2}\sum_{j\in\mathcal{B}_t}\mathbb{E}\big\|g_\omega\big(\tau_j|\boldsymbol{\theta}_{t-1}^{s+1}\big) - g\big(\tau_j|\boldsymbol{\theta}_t^{s+1}\big)\big\|_2^2 + \big\|\nabla J(\boldsymbol{\theta}_{t-1}^{s+1}) - \mathbf{v}_{t-1}^{s+1}\big\|_2^2, && (4.7.8)
\end{aligned}
$$

where (4.7.7) is due to the fact that $\mathbb{E}\|\mathbf{x}_1+\ldots+\mathbf{x}_n\|_2^2 = \mathbb{E}\|\mathbf{x}_1\|_2+\ldots+\mathbb{E}\|\mathbf{x}_n\|_2$ for independent zero-mean random variables, and (4.7.8) holds due to the fact that $\mathbf{x}_1,\ldots,\mathbf{x}_n$ is due to $\mathbb{E}\|\mathbf{x}-\mathbb{E}\mathbf{x}\|_2^2 \le \mathbb{E}\|\mathbf{x}\|_2^2$. For the first term, we have

$$
\begin{aligned}
\mathbb{E}\big[\big\|g_\omega\big(\tau_j|\boldsymbol{\theta}_{t-1}^{s+1}\big) - g\big(\tau_j|\boldsymbol{\theta}_t^{s+1}\big)\big\|_2^2\big] &= \mathbb{E}\Bigg[\bigg\|\sum_{h=0}^{H-1}(\omega_{0:h}-1)\bigg[\sum_{t=0}^{h}\nabla_{\boldsymbol{\theta}}\log\pi_{\boldsymbol{\theta}}(a_t^i|s_t^i)\bigg]\gamma^h r(s_h^i,a_h^i)\bigg\|_2^2\Bigg] \\
&= \sum_{h=0}^{H-1}\mathbb{E}\Bigg[\bigg\|(\omega_{0:h}-1)\bigg[\sum_{t=0}^{h}\nabla_{\boldsymbol{\theta}}\log\pi_{\boldsymbol{\theta}}(a_t^i|s_t^i)\bigg]\gamma^h r(s_h^i,a_h^i)\bigg\|_2^2\Bigg] \\
&\le \sum_{h=0}^{H-1} h^2(2G^2+M)(W+1)\big\|\boldsymbol{\theta}_{t-1}^{s+1} - \boldsymbol{\theta}_t^{s+1}\big\|_2^2 \cdot h^2G^2\gamma^h R
\end{aligned}
$$

$$\leq \frac{24RG^2(2G^2+M)(W+1)\gamma}{(1-\gamma)^5}\big\|\boldsymbol{\theta}_{t-1}^{s+1}-\boldsymbol{\theta}_t^{s+1}\big\|_2^2, \tag{4.7.9}$$

where in the second equality we used the fact that $\mathbb{E}[\nabla\log\pi_{\boldsymbol{\theta}}(a|s)]=\mathbf{0}$, the first inequality is due to Lemma 4.7.1 and in the last inequality we use the fact that $\sum_{h=0}^{\infty}h^4\gamma^h=\gamma(\gamma^3+11\gamma^2+11\gamma+1)/(1-\gamma)^5$ for $|\gamma|<1$. Combining the results in (4.7.8) and (4.7.9), we get

$$\begin{aligned}\mathbb{E}\big\|\nabla J\big(\boldsymbol{\theta}_t^{s+1}\big)-\mathbf{v}_t^{s+1}\big\|_2^2 &\leq \frac{C_\gamma}{B}\big\|\boldsymbol{\theta}_t^{s+1}-\boldsymbol{\theta}_{t-1}^{s+1}\big\|_2^2+\big\|\nabla J(\boldsymbol{\theta}_{t-1}^{s+1})-\mathbf{v}_{t-1}^{s+1}\big\|_2^2\\ &\leq \frac{C_\gamma}{B}\sum_{l=1}^{t}\big\|\boldsymbol{\theta}_l^{s+1}-\boldsymbol{\theta}_{l-1}^{s+1}\big\|_2^2+\big\|\nabla J(\boldsymbol{\theta}_0^{s+1})-\mathbf{v}_0^{s+1}\big\|_2^2, \end{aligned}\tag{4.7.10}$$

which holds for $t=1,\ldots,m-1$, where $C_\gamma=24RG^2(2G^2+M)(W+1)\gamma/(1-\gamma)^5$. According to Algorithm 7 and Assumption 4.4.3, we have

$$\mathbb{E}\big\|\nabla J\big(\boldsymbol{\theta}_0^{s+1}\big)-\mathbf{v}_0^{s+1}\big\|_2^2\leq\frac{\xi^2}{N}. \tag{4.7.11}$$

Submitting the above result into (4.7.5) yields

$$\begin{aligned}\mathbb{E}_{N,B}\big[\Phi\big(\boldsymbol{\theta}_{t+1}^{s+1}\big)\big] &\geq \mathbb{E}_{N,B}\big[\Phi\big(\boldsymbol{\theta}_t^{s+1}\big)\big]+\frac{\eta}{8}\big\|\mathcal{G}_\eta\big(\boldsymbol{\theta}_t^{s+1}\big)\big\|_2^2+\bigg(\frac{1}{4\eta}-\frac{L}{2}\bigg)\big\|\boldsymbol{\theta}_{t+1}^{s+1}-\boldsymbol{\theta}_t^{s+1}\big\|_2^2\\ &\quad-\frac{3\eta C_\gamma}{4B}\mathbb{E}_{N,B}\bigg[\sum_{l=1}^{t}\big\|\boldsymbol{\theta}_l^{s+1}-\boldsymbol{\theta}_{l-1}^{s+1}\big\|_2^2\bigg]-\frac{3\eta\xi^2}{4N}, \end{aligned}\tag{4.7.12}$$

for $t=1,\ldots,m-1$.Recall Line 6 in Algorithm 7, where we update $\boldsymbol{\theta}_1^{t+1}$ with the average of a mini-batch of gradients $\mathbf{v}_0^s=1/N\sum_{i=1}^{N}g(\tau_i|\tilde{\boldsymbol{\theta}}^s)$. Similar to (4.7.5), by smoothness of $J(\boldsymbol{\theta})$, we have

$$\begin{aligned}\Phi\big(\boldsymbol{\theta}_1^{s+1}\big)\geq \Phi\big(\boldsymbol{\theta}_0^{s+1}\big)-\frac{3\eta}{4}\big\|\nabla J\big(\boldsymbol{\theta}_0^{s+1}\big)-\mathbf{v}_0^{s+1}\big\|_2^2+\frac{\eta}{8}\big\|\mathcal{G}_\eta\big(\boldsymbol{\theta}_0^{s+1}\big)\big\|_2^2\\ +\bigg(\frac{1}{4\eta}-\frac{L}{2}\bigg)\big\|\boldsymbol{\theta}_1^{s+1}-\boldsymbol{\theta}_0^{s+1}\big\|_2^2.\end{aligned}$$

Further by (4.7.11), it holds that

$$\mathbb{E}\big[\Phi\big(\boldsymbol{\theta}_1^{s+1}\big)\big]\geq\mathbb{E}\big[\Phi\big(\boldsymbol{\theta}_0^{s+1}\big)\big]-\frac{3\eta\xi^2}{4N}+\frac{\eta}{8}\big\|\mathcal{G}_\eta\big(\boldsymbol{\theta}_0^{s+1}\big)\big\|_2^2+\bigg(\frac{1}{4\eta}-\frac{L}{2}\bigg)\big\|\boldsymbol{\theta}_1^{s+1}-\boldsymbol{\theta}_0^{s+1}\big\|_2^2. \tag{4.7.13}$$

Telescoping inequality (4.7.12) from $t=1$ to $m-1$ and combining the result with (4.7.13), we obtain

$$\mathbb{E}_{N,B}\big[\Phi\big(\boldsymbol{\theta}_m^{s+1}\big)\big]\geq\mathbb{E}_{N,B}\big[\Phi\big(\boldsymbol{\theta}_0^{s+1}\big)\big]+\frac{\eta}{8}\sum_{t=0}^{m-1}\mathbb{E}_N\big[\big\|\mathcal{G}_\eta\big(\boldsymbol{\theta}_t^{s+1}\big)\big\|_2^2\big]-\frac{3m\eta\xi^2}{4N}$$

$$
\begin{aligned}
&+\left(\frac{1}{4\eta}-\frac{L}{2}\right)\sum_{t=0}^{m-1}\big\|\boldsymbol{\theta}_{t+1}^{s+1}-\boldsymbol{\theta}_{t}^{s+1}\big\|_2^2\\
&-\frac{3\eta C_\gamma}{2B}\mathbb{E}_{N,B}\left[\sum_{t=0}^{m-1}\sum_{l=1}^{t}\big\|\boldsymbol{\theta}_{l}^{s+1}-\boldsymbol{\theta}_{l-1}^{s+1}\big\|_2^2\right]\\
\geq{}& \mathbb{E}_{N,B}\big[\Phi\big(\boldsymbol{\theta}_0^{s+1}\big)\big]+\frac{\eta}{8}\sum_{t=0}^{m-1}\mathbb{E}_N\big[\big\|\mathcal{G}_\eta\big(\boldsymbol{\theta}_t^{s+1}\big)\big\|_2^2\big]-\frac{3m\eta\xi^2}{4N}\\
&+\left(\frac{1}{4\eta}-\frac{L}{2}-\frac{3m\eta C_\gamma}{2B}\right)\sum_{t=0}^{m-1}\big\|\boldsymbol{\theta}_{t+1}^{s+1}-\boldsymbol{\theta}_{t}^{s+1}\big\|_2^2. \qquad (4.7.14)
\end{aligned}
$$

If we choose step size η and the epoch length B such that

$$
\eta\leq\frac{1}{4L},\qquad \frac{B}{m}\geq\frac{3\eta C_\gamma}{L}=\frac{72\eta G^2(2G^2+M)(W+1)\gamma}{M(1-\gamma)^3}, \qquad (4.7.15)
$$

and note that $\boldsymbol{\theta}_0^{s+1}=\tilde{\boldsymbol{\theta}}^s$, $\boldsymbol{\theta}_m^{s+1}=\tilde{\boldsymbol{\theta}}^{s+1}$, then (4.7.14) leads to

$$
\mathbb{E}_N\big[\Phi\big(\tilde{\boldsymbol{\theta}}^{s+1}\big)\big]\geq\mathbb{E}_N\big[\Phi\big(\tilde{\boldsymbol{\theta}}^{s}\big)\big]+\frac{\eta}{8}\sum_{t=0}^{m-1}\mathbb{E}_N\big[\big\|\mathcal{G}_\eta\big(\boldsymbol{\theta}_t^{s+1}\big)\big\|_2^2\big]-\frac{3m\eta\xi^2}{4N}. \qquad (4.7.16)
$$

Summing up the above inequality over $s=0,\ldots,S-1$ yields

$$
\frac{\eta}{8}\sum_{s=0}^{S-1}\sum_{t=0}^{m-1}\mathbb{E}\big[\big\|\mathcal{G}_\eta\big(\boldsymbol{\theta}_t^{s+1}\big)\big\|_2^2\big]\leq\mathbb{E}\big[\Phi\big(\tilde{\boldsymbol{\theta}}^{S}\big)\big]-\mathbb{E}\big[\Phi\big(\tilde{\boldsymbol{\theta}}^{0}\big)\big]+\frac{3Sm\eta\xi^2}{4N},
$$

which immediately implies

$$
\mathbb{E}\big[\big\|\mathcal{G}_\eta\big(\boldsymbol{\theta}_{\text{out}}\big)\big\|_2^2\big]\leq\frac{8\big(\mathbb{E}\big[\Phi\big(\tilde{\boldsymbol{\theta}}^{S}\big)\big]-\mathbb{E}\big[\Phi\big(\tilde{\boldsymbol{\theta}}^{0}\big)\big]\big)}{\eta Sm}+\frac{6\xi^2}{N}\leq\frac{8(\Phi(\boldsymbol{\theta}^*)-\Phi(\boldsymbol{\theta}_0))}{\eta Sm}+\frac{6\xi^2}{N}.
$$

This completes the proof. □

Proof of Corollary 4.4.7. Based on the convergence results in Theorem 4.4.5, in order to ensure $\mathbb{E}\big[\big\|\nabla J\big(\boldsymbol{\theta}_{\text{out}}\big)\big\|_2^2\big]\leq\epsilon$, we can choose S,m and N such that

$$
\frac{8(J(\boldsymbol{\theta}^*)-J(\boldsymbol{\theta}_0))}{\eta Sm}=\frac{\epsilon}{2},\quad \frac{6\xi^2}{N}=\frac{\epsilon}{2},
$$

which implies $Sm=O(1/\epsilon)$ and $N=O(1/\epsilon)$. Note that we have set $m=O(B)$. The total number of stochastic gradient evaluations $\mathcal{T}_g$ we need is

$$
\mathcal{T}_g=SN+SmB=O\bigg(\frac{N}{B\epsilon}+\frac{B}{\epsilon}\bigg)=O\bigg(\frac{1}{\epsilon^{3/2}}\bigg),
$$

where we set $B=1/\epsilon^{1/2}$. □

4.8 Proof of Technical Lemmas

In this section, we provide the proofs of the technical lemmas. We first prove the smoothness of the performance function $J(\boldsymbol{\theta})$.

Proof of Proposition 4.4.2. Recall the definition of PGT in (4.2.5). We first show the Lipschitzness of $g(\tau|\boldsymbol{\theta})$ with baseline $b = 0$ as follows:

$$\begin{aligned}
\|\nabla g(\tau|\boldsymbol{\theta})\|_2 &= \left\| \sum_{h=0}^{H-1} \nabla_{\boldsymbol{\theta}}^2 \log \pi_{\boldsymbol{\theta}}(a_h|s_h) \left(\sum_{t=h}^{H-1} \gamma^t r(s_t, a_t) \right) \right\|_2 \\
&\le \left(\sum_{t=0}^{H-1} \gamma^h \left\| \nabla_{\boldsymbol{\theta}}^2 \log \pi_{\boldsymbol{\theta}}(a_t|s_t) \right\|_2 \right) \frac{R}{1-\gamma} \\
&\le \frac{MR}{(1-\gamma)^2},
\end{aligned}$$

where we used the fact that $0 < \gamma < 1$. When we have a nonzero baseline b_h, we can simply scale it with γ^h and the above result still holds up to a constant multiplier.

Since the PGT estimator is an unbiased estimator of the policy gradient $\nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta})$, we have $\nabla_{\boldsymbol{\theta}} J(\boldsymbol{\theta}) = \mathbb{E}_\tau[g(\tau|\boldsymbol{\theta})]$ and $\nabla_{\boldsymbol{\theta}}^2 J(\boldsymbol{\theta}) = \mathbb{E}_\tau[\nabla_{\boldsymbol{\theta}} g(\tau|\boldsymbol{\theta})]$. Therefore, the smoothness of $J(\boldsymbol{\theta})$ can be directly implied from the Lipschitzness of $g(\tau|\boldsymbol{\theta})$:

$$\left\| \nabla_{\boldsymbol{\theta}}^2 J(\boldsymbol{\theta}) \right\|_2 = \|\mathbb{E}_\tau[\nabla_{\boldsymbol{\theta}} g(\tau|\boldsymbol{\theta})]\|_2 \le \|\nabla_{\boldsymbol{\theta}} g(\tau|\boldsymbol{\theta})\|_2 \le \frac{MR}{(1-\gamma)^2},$$

which implies that $J(\boldsymbol{\theta})$ is L-smooth with $L = MR/(1-\gamma)^2$.

Similarly, we can bound the norm of gradient estimator as follows

$$\|g(\tau|\boldsymbol{\theta})\|_2 \le \left\| \sum_{h=0}^{H-1} \nabla_{\boldsymbol{\theta}} \log \pi_{\boldsymbol{\theta}}(a_h|s_h) \frac{\gamma^h R(1-\gamma^{H-h})}{1-\gamma} \right\|_2 \le \frac{GR}{(1-\gamma)^2},$$

which completes the proof. $\square$

Lemma 4.8.1 (Lemma 1 in [CMM10])**.** *Let $\omega(x) = P(x)/Q(x)$ be the importance weight for distributions P and Q. Then $\mathbb{E}[\omega] = 1, \mathbb{E}[\omega^2] = d_2(P||Q)$, where $d_2(P||Q) = 2^{D_2(P||Q)}$ and $D_2(P||Q)$ is the Rényi divergence between distributions P and Q. Note that this immediately implies $Var(\omega) = d_2(P||Q) - 1$.*

Proof of Lemma 4.7.1. According to the property of importance weight in Lemma 4.8.1, we know

$$\mathrm{Var}\big(\omega_{0:h}\big(\tau|\tilde{\boldsymbol{\theta}}^s,\boldsymbol{\theta}_t^{s+1}\big)\big)=d_2\big(p(\tau_h|\tilde{\boldsymbol{\theta}}^s)||p(\tau_h|\boldsymbol{\theta}_t^{s+1})\big)-1.$$

To simplify the presentation, we denote $\boldsymbol{\theta}_1=\tilde{\boldsymbol{\theta}}^s$ and $\boldsymbol{\theta}_2=\boldsymbol{\theta}_t^{s+1}$ in the rest of this proof. By definition, we have

$$d_2(p(\tau_h|\boldsymbol{\theta}_1)||p(\tau_h|\boldsymbol{\theta}_2))=\int_\tau p(\tau_h|\boldsymbol{\theta}_1)\frac{p(\tau_h|\boldsymbol{\theta}_1)}{p(\tau_h|\boldsymbol{\theta}_2)}\mathrm{d}\tau=\int_\tau p(\tau_h|\boldsymbol{\theta}_1)^2p(\tau_h|\boldsymbol{\theta}_2)^{-1}\mathrm{d}\tau.$$

Taking the gradient of $d_2(p(\tau_h|\boldsymbol{\theta}_1)||p(\tau_h|\boldsymbol{\theta}_2))$ with respect to $\boldsymbol{\theta}_1$, we have

$$\nabla_{\boldsymbol{\theta}_1}d_2(p(\tau_h|\boldsymbol{\theta}_1)||p(\tau_h|\boldsymbol{\theta}_2))=2\int_\tau p(\tau_h|\boldsymbol{\theta}_1)\nabla_{\boldsymbol{\theta}_1}p(\tau_h|\boldsymbol{\theta}_1)p(\tau_h|\boldsymbol{\theta}_2)^{-1}\mathrm{d}\tau.$$

In particular, if we set the value of $\boldsymbol{\theta}_1$ to be $\boldsymbol{\theta}_1=\boldsymbol{\theta}_2$ in the above formula of the gradient, we get

$$\nabla_{\boldsymbol{\theta}_1}d_2(p(\tau_h|\boldsymbol{\theta}_1)||p(\tau_h|\boldsymbol{\theta}_2))\big|_{\boldsymbol{\theta}_1=\boldsymbol{\theta}_2}=2\int_\tau\nabla_{\boldsymbol{\theta}_1}p(\tau_h|\boldsymbol{\theta}_1)\mathrm{d}\tau\big|_{\boldsymbol{\theta}_1=\boldsymbol{\theta}_2}=0.$$

Applying mean value theorem with respect to the variable $\boldsymbol{\theta}_1$, we have

$$d_2(p(\tau_h|\boldsymbol{\theta}_1)||p(\tau_h|\boldsymbol{\theta}_2))=1+1/2(\boldsymbol{\theta}_1-\boldsymbol{\theta}_2)^\top\nabla^2_{\boldsymbol{\theta}}d_2(p(\tau_h|\boldsymbol{\theta})||p(\tau_h|\boldsymbol{\theta}_2))(\boldsymbol{\theta}_1-\boldsymbol{\theta}_2),\qquad(4.8.1)$$

where $\boldsymbol{\theta}=t\boldsymbol{\theta}_1+(1-t)\boldsymbol{\theta}_2$ for some $t\in[0,1]$ and we used the fact that $d_2(p(\tau_h|\boldsymbol{\theta}_2)||p(\tau_h|\boldsymbol{\theta}_2))=1$. To bound the above exponentiated Rényi divergence, we need to compute the Hessian matrix. Taking the derivative of $\nabla_{\boldsymbol{\theta}_1}d_2(p(\tau_h|\boldsymbol{\theta}_1)||p(\tau_h|\boldsymbol{\theta}_2))$ with respect to $\boldsymbol{\theta}_1$ further yields

$$\begin{aligned}\nabla^2_{\boldsymbol{\theta}}d_2(p(\tau_h|\boldsymbol{\theta})||p(\tau_h|\boldsymbol{\theta}_2))&=2\int_\tau\nabla_{\boldsymbol{\theta}}\log p(\tau_h|\boldsymbol{\theta})\nabla_{\boldsymbol{\theta}}\log p(\tau_h|\boldsymbol{\theta})^\top\frac{p(\tau_h|\boldsymbol{\theta})^2}{p(\tau_h|\boldsymbol{\theta}_2)}\mathrm{d}\tau\\&\quad+2\int_\tau\nabla^2_{\boldsymbol{\theta}}p(\tau_h|\boldsymbol{\theta})p(\tau_h|\boldsymbol{\theta})p(\tau_h|\boldsymbol{\theta}_2)^{-1}\mathrm{d}\tau.\end{aligned}\qquad(4.8.2)$$

Thus we need to compute the Hessian matrix of the trajectory distribution function, i.e., $\nabla^2_{\boldsymbol{\theta}}p(\tau_h|\boldsymbol{\theta})$, which can further be derived from the Hessian matrix of the log-density function.

$$\nabla^2_{\boldsymbol{\theta}}\log p(\tau_h|\boldsymbol{\theta})=-p(\tau_h|\boldsymbol{\theta})^{-2}\nabla_{\boldsymbol{\theta}}p(\tau_h|\boldsymbol{\theta})\nabla_{\boldsymbol{\theta}}p(\tau_h|\boldsymbol{\theta})^\top+p(\tau_h|\boldsymbol{\theta})^{-1}\nabla^2_{\boldsymbol{\theta}}p(\tau_h|\boldsymbol{\theta}).\qquad(4.8.3)$$

Submitting (4.8.3) into (4.8.2) yields

$$\|\nabla^2_{\boldsymbol{\theta}}d_2(p(\tau_h|\boldsymbol{\theta})||p(\tau_h|\boldsymbol{\theta}_2))\|_2=\bigg\|4\int_\tau\nabla_{\boldsymbol{\theta}}\log p(\tau_h|\boldsymbol{\theta})\nabla_{\boldsymbol{\theta}}\log p(\tau_h|\boldsymbol{\theta})^\top\frac{p(\tau_h|\boldsymbol{\theta})^2}{p(\tau_h|\boldsymbol{\theta}_2)}\mathrm{d}\tau$$

$$
\begin{aligned}
&\quad + 2\int_{\tau}\nabla^2_{\boldsymbol{\theta}}\log p(\tau_h|\boldsymbol{\theta})\frac{p(\tau_h|\boldsymbol{\theta})^2}{p(\tau_h|\boldsymbol{\theta}_2)}\mathrm{d}\tau\Bigg\|_2\\
&\le \int_{\tau}\frac{p(\tau_h|\boldsymbol{\theta})^2}{p(\tau_h|\boldsymbol{\theta}_2)}\big(4\|\nabla_{\boldsymbol{\theta}}\log p(\tau_h|\boldsymbol{\theta})\|_2^2 + 2\|\nabla^2_{\boldsymbol{\theta}}\log p(\tau_h|\boldsymbol{\theta})\|_2\big)\mathrm{d}\tau\\
&\le (4h^2G^2 + 2hM)\mathbb{E}[\omega(\tau|\boldsymbol{\theta},\boldsymbol{\theta}_2)^2]\\
&\le 2h(2hG^2 + M)(W+1),
\end{aligned}
$$

where the second inequality comes from Assumption 4.4.1 and the last inequality is due to Assumption 4.4.4 and Lemma 4.8.1. Combining the above result with (4.8.1), we have

$$
\mathrm{Var}\big(\omega_{0:h}\big(\tau|\tilde{\boldsymbol{\theta}}^s,\boldsymbol{\theta}_t^{s+1}\big)\big) = d_2\big(p(\tau_h|\tilde{\boldsymbol{\theta}}^s)||p(\tau_h|\boldsymbol{\theta}_t^{s+1})\big) - 1 \le C_\omega\|\tilde{\boldsymbol{\theta}}^s - \boldsymbol{\theta}_t^{s+1}\|_2^2,
$$

where $C_\omega = h(2hG^2 + M)(W+1)$. □

4.9 Proof of Theoretical Results for Gaussian Policy

In this section, we prove the sample complexity for Gaussian policy. According to (4.4.1), we can calculate the gradient and Hessian matrix of the logarithm of the policy.

$$
\nabla\log\pi_{\boldsymbol{\theta}}(a|s) = \frac{(a-\boldsymbol{\theta}^\top\phi(s))\phi(s)}{\sigma^2},\quad \nabla^2\log\pi_{\boldsymbol{\theta}}(a|s) = -\frac{\phi(s)\phi(s)^\top}{\sigma^2}. \tag{4.9.1}
$$

It is easy to see that Assumption 4.4.1 holds with $G = C_aM_\phi/\sigma^2$ and $M = M_\phi^2/\sigma^2$. Based on this observation, Proposition 4.4.2 also holds for Gaussian policy with parameters defined as follows

$$
L = \frac{RM_\phi^2}{\sigma^2(1-\gamma)^2},\quad \text{and}\quad C_g = \frac{RC_aM_\phi}{\sigma^2(1-\gamma)^2}. \tag{4.9.2}
$$

The following lemma gives the variance ξ^2 of the PGT estimator, which verifies Assumption 4.4.3.

Lemma 4.9.1 (Lemma 5.5 in [PRB13]). *Given a Gaussian policy $\pi_{\boldsymbol{\theta}}(a|s)\sim N(\boldsymbol{\theta}^\top\phi(s),\sigma^2)$, if the $|r(s,a)|\le R$ and $\|\phi(s)\|_2\le M_\phi$ for all $s\in\mathcal{S}, a\in\mathcal{A}$ and $R, M_\phi > 0$ are constants, then the variance of PGT estimator defined in* (4.2.5) *can be bounded as follows:*

$$
Var(g(\tau|\boldsymbol{\theta})) \le \xi^2 = \frac{R^2M_\phi^2}{(1-\gamma)^2\sigma^2}\bigg(\frac{1-\gamma^{2H}}{1-\gamma^2} - H\gamma^{2H} - 2\gamma^H\frac{1-\gamma^H}{1-\gamma}\bigg).
$$

Proof of Corollary 4.4.8. The proof will be similar to that of Corollary 4.4.7. By Theorem 4.4.5, to ensure that $\mathbb{E}[\|\nabla J(\boldsymbol{\theta}_{\text{out}})\|_2^2] \le \epsilon$, we can set

$$\frac{8(J(\boldsymbol{\theta}^*) - J(\boldsymbol{\theta}_0))}{\eta S m} = \frac{\epsilon}{2}, \quad \frac{6\xi^2}{N} = \frac{\epsilon}{2}.$$

Plugging the value of ξ^2 in Lemma 4.9.1 into the second equation above yields $N = O(\epsilon^{-1}(1-\gamma)^{-3})$. For the first equation, we have $S = O(1/(\eta m \epsilon))$. Therefore, the total number of stochastic gradient evaluations $\mathcal{T}_g$ required by Algorithm 7 is

$$\mathcal{T}_g = SN + SmB = O\bigg(\frac{N}{\eta m \epsilon} + \frac{B}{\eta \epsilon}\bigg).$$

So a good choice of batch size B and epoch length m will lead to $Bm = N$. Combining this with the requirement of B in Theorem 4.4.5, we can set

$$m = \sqrt{\frac{LN}{\eta C_\gamma}}, \quad \text{and } B = \sqrt{\frac{N \eta C_\gamma}{L}}.$$

Note that $C_\gamma = 24RG^2(2G^2 + M)(W+1)\gamma/(1-\gamma)^5$. Plugging the values of G, N and L into the above equations yields

$$m = O\bigg(\frac{1}{(1-\gamma)\sqrt{\epsilon}}\bigg), \quad B = O\bigg(\frac{1}{(1-\gamma)^2\sqrt{\epsilon}}\bigg).$$

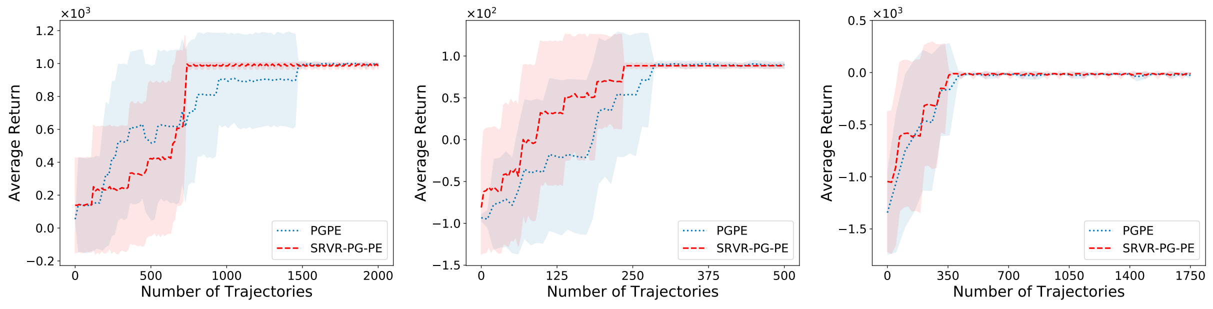
The corresponding sample complexity is

$$\mathcal{T}_g = O\bigg(\frac{1}{(1-\gamma)^4 \epsilon^{3/2}}\bigg).$$

This completes the proof for Gaussian policy. □

4.10 Additional Details on Experiments

Now, we provide more details of our experiments presented in Section 4.5. We first present the parameters for all algorithms we used in all our experiments in Tables 4.2 and 4.3. Among the parameters, the neural network structure and the RL environment parameters are shared across all the algorithms. As mentioned in Section 4.5, the order of the batch size parameters of our algorithm are chosen according to Corollary 4.7 and we multiply them by



(a) Cartpole (b) Mountain Car (c) Pendulum

Figure 4.2: Performance of SRVR-PG-PE compared with PGPE. Experiment results are averaged over 10 runs.

a tuning constant via grid search. Similarly, the orders of batch size parameters of SVRPG and GPOMDP are chosen based on the theoretical results suggested by [PBC+18, XGG19]. Moerover, the learning rates for different methods are tuned by grid search.

We then present the results of PGPE and SRVR-PG-PE on Cartpole, Mountain Car and Pendulum in Figure 4.2. In all three environments, our SRVR-PG-PE algorithm shows improvement over PGPE [SOR+10] in terms of number of trajectories. It is worth noting that in all these environments both PGPE and SRVR-PG-PE seem to solve the problem very quickly, which is consistent with the results reported in [ZHNS11, ZHT+13, MPFR18]. Our primary goal in this experiment is to show that our proposed variance reduced policy gradient algorithm can be easily extended to the PGPE framework. To avoid distracting the audience's attention from the variance reduction algorithm on the sample complexity, we do not thoroughly compare the performance of the parameter based policy gradient methods such as PGPE and SRVR-PG-PE with the action based policy gradient methods. We refer interested readers to the valuable empirical studies of PGPE based algorithms presented in [ZHNS11, ZHT+13, MPFR18].

Table 4.2: Parameters used in the SRVR-PG-PE experiments.

Parameters	Cartpole	Mountain Car	Pendulum
NN size	-	64	8×8
NN activation function	Tanh	Tanh	Tanh
Task horizon	100	1000	200
Total trajectories	2000	500	1750
Discount factor γ	0.99	0.999	0.99
Learning rate η	0.01	0.0075	0.01
Batch size N	10	5	50
Batch size B	5	3	10
Epoch size m	2	1	2

Table 4.3: Parameters used in the SRVR-PG experiments.

Parameters	Algorithm	Cartpole	Mountain Car	Pendulum
NN size	-	64	64	8×8
NN activation function	-	Tanh	Tanh	Tanh
Task horizon	-	100	1000	200
Total trajectories	-	2500	3000	2×10^5
	GPOMDP	0.99	0.999	0.99
Discount factor γ	SVRPG	0.999	0.999	0.995
	SRVR-PG	0.995	0.999	0.995
	GPOMDP	0.005	0.005	0.01
Learning rate η	SVRPG	0.0075	0.0025	0.01
	SRVR-PG	0.005	0.0025	0.01
	GPOMDP	10	10	250
Batch size N	SVRPG	25	10	250
	SRVR-PG	25	10	250
	GPOMDP	-	-	-
Batch size B	SVRPG	10	5	50
	SRVR-PG	5	3	50
	GPOMDP	-	-	-
Epoch size m	SVRPG	3	2	1
	SRVR-PG	3	2	1

CHAPTER 5

Finite-Time Analysis for Policy Optimization with Linear Value Function Approximation

5.1 Introduction

In the previous chapter, we discussed about optimization techniques in policy gradient methods. The gradient estimators for REINFORCE defined in (4.2.4) and for SRVR-PG in (4.3.2) are all based on the discounted average of the rewards obtained from a batch of trajectories. In this chapter, we will study the Actor-Critic (AC) method [BSA83, KT00], which use value function approximation to learn the gradient estimator used in policy gradient methods. The policy parameter is called the actor, and the value function approximator is called the critic. Specifically, actor-only methods, such as policy gradient [SMSM00] and trust region policy optimization [SLA$^+$15], utilize a parameterized policy function class and improve the policy by optimizing the parameters of some performance function using gradient ascent, whose exact form is characterized by the Policy Gradient Theorem [SMSM00]. Actor-only methods can be naturally applied to continuous setting but suffer from high variance when estimating the policy gradient. On the other hand, critic-only methods, such as temporal difference learning [Sut88] and Q-learning [WD92], focus on learning a value function (expected cumulative rewards), and determine the policy based on the value function, which is recursively approximated based on the Bellman equation. Although the critic-only methods can efficiently learn a satisfying policy under tabular setting [JAZBJ18], they can diverge with function approximation under continuous setting [Wie04]. Therefore, it is natural to combine actor and critic based methods to achieve the best of both worlds. The principal idea behind actor-critic methods is simple: the critic tries to learn the value function, given the policy from the actor, while the actor can estimate the policy gradient based on the approximate value function provided by the critic.

If the actor is fixed, the policy remains unchanged throughout the updates of the critic. Thus one can use policy evaluation algorithm such as temporal difference (TD) learning [SB18] to estimate the value function (critic). After many steps of the critic update, one can expect a good estimation of the value function, which in turn enables an accurate estimation of the policy gradient for the actor. A more favorable implementation is the so-called two time-scale actor-critic algorithm, where the actor and the critic are updated simultaneously at each iteration except that the actor changes more slowly (with a small step size) than the critic (with a large step size). In this way, one can hope the critic will be well approximated even after one step of update. From the theoretical perspective, the asymptotic analysis of two time-scale actor-critic methods has been established in [BK97, KT00]. In specific, under the assumption that the ratio of the two time-scales goes to infinity (i.e. $\lim_{t\to\infty} \beta_t/\alpha_t = \infty$), the asymptotic convergence is guaranteed through the lens of the two time-scale ordinary differential equations(ODE), where the slower component is fixed and the faster component converges to its stationary point. This type of analysis was also applied in the context of generic two time-scale stochastic approximation [Bor97].

However, finite-time analysis (non-asymptotic analysis) of two-time scale actor-critic is still largely missing in the literature, which is important because it can address the questions that how many samples are needed for two time-scale actor-critic to converge, and how to appropriately choose the different learning rates for the actor and the critic. Some recent work has attempted to provide the finite-time analysis for the "decoupled" actor-critic methods [KKR19, QYYW19]. The term "decoupled" means that before updating the actor at the t-th iteration, the critic starts from scratch to estimate the state-value (or Q-value) function. At each iteration, the "decoupled" setting requires the critic to perform multiple sampling and updating (often from another new sample trajectory). As we will see in the later comparison, this setting is sample-inefficient or even impractical. Besides, their analyses are based on either the i.i.d. assumption [KKR19] or the partially i.i.d. assumption [QYYW19] (the actor receives i.i.d. samples), which is unrealistic in practice. In this chapter, we present the first finite-time analysis on the convergence of the two time-scale actor-critic algorithm. We summarize our contributions as follows:

- We prove that, the actor in the two time-scale actor critic algorithm converges to an ϵ-approximate stationary point of the non-concave performance function J after accessing at most $\tilde{O}(\epsilon^{-2.5})$ samples. Compared with existing finite-time analysis of actor-critic methods [KKR19, QYYW19], the algorithm we analyzed is based on two time-scale update and therefore more practical and efficient than the "decoupled" version. Moreover, we do not need any i.i.d. data assumptions in the convergence analysis as required by [KKR19, QYYW19], which do not hold in real applications.
- From the technical viewpoint, we also present a new proof framework that can tightly characterize the estimation error in two time-scale algorithms. Compared with the proof technique used in [XZL19], we remove the extra artificial factor $O(t^{\xi})$ in the convergence rate introduced by their "iterative refinement" technique. Therefore, our new proof technique may be of independent interest for analyzing the convergence of other two time-scale algorithms to get sharper rates.

5.2 Related Work

In this section, we briefly review and discuss existing work, which is mostly related to ours.

Stochastic Bias Characterization The main difficulty in analyzing reinforcement learning algorithms under non-i.i.d. data assumptions is that the samples and the trainable parameters are correlated, which makes the noise term biased. [BRS18] used information-theoretical techniques to bound the Markovian bias and provide a simple and explicit analysis for the temporal difference learning. Similar techniques were also established in [SY19] through the lens of stochastic approximation methods. [GSY19, XZL19] applied such methods to deriving the non-asymptotic convergence of two time-scale temporal difference learning algorithms (TDC). [ZXL19, CZD$^+$19, XG20] further applied these analysis methods to on-policy learning algorithms including SARSA and Q-learning. In addition, [HS19] formulated a family of TD learning algorithms as Markov jump linear systems and analyzed the evolution of the mean and covariance matrix of the estimation error. [CYJW19] studied TD learning with neural network approximation, and proved its global convergence.

Two Time-Scale Reinforcement Learning The two time-scale stochastic approximation can be seen as a general framework for analyzing reinforcement learning [Bor97, TM03, KTo04]. Recently, the finite-time analysis of two time-scale stochastic approximation has gained much interest. [DSTM17] proved convergence rate for the two time-scale linear stochastic approximation under i.i.d. assumption. [GSY19] also provided finite-time analysis for the two time-scale linear stochastic approximation algorithms. Both can be applied to analyze two time-scale TD methods like GTD, GTD2 and TDC. [XZL19] proved convergence rate and sample complexity for the TDC algorithm over Markovian samples. [KMN$^+$20] further improved the convergence rate of two time-scale linear stochastic approximation and removed the projection step. However, since the update rule for the actor is generally not linear, we cannot apply these results to the actor-critic algorithms.

Analysis for Actor-Critic Methods The asymptotic analysis of actor-critic methods has been well established. [KT00] proposed the actor-critic algorithm, and established the asymptotic convergence for the two time-scale actor-critic, with TD(λ) learning-based critic. [BSGL09] proved the convergence result for the original actor-critic and natural actor-critic methods. [CM10] proposed a single time-scale actor-critic algorithm and proved its convergence. Recently, [ZLYW19] proved convergence of two time-scale off-policy actor-critic with function approximation. Recently, there has emerged some works concerning the finite-time behavior of actor-critic methods. [YCHW19] studied the global convergence of actor-critic algorithms under the Linear Quadratic Regulator. [YZHB18] analyzed the finite-sample performance of batched actor-critic, where all samples are assumed i.i.d. and the critic performs several empirical risk minimization (ERM) steps. [QYYW19] treated the actor-critic algorithms as a bilevel optimization problem and established a finite sample analysis under the "average-reward" setting, assuming that the actor has access to independent samples. Similar result has also been established by [KKR19], where they considered the sample complexity for the "decoupled" actor-critic methods under i.i.d. assumption. [WCYW20] also proved the global convergence of actor-critic algorithms with both actor and critic being approximated by overparameterized neural networks.

When we were preparing this work, we noticed that there is a concurrent and independent work [XWL20] which also analyzes the non-asymptotic convergence of two time-scale actor-critic algorithms and achieves the same sample complexity, i.e., $\tilde{\mathcal{O}}(\epsilon^{-2.5})$. However, there are two key differences between their work and ours. First, the two time-scale algorithms analyzed in both papers are very different. We analyze the classical two time-scale algorithm described in [SB18], where both actor and critic take one step update in each iteration. It is very easy to implement and has been widely used in practice, while the update rule in [XWL20] for the critic needs to call a sub-algorithm, which involves generating a fresh episode to estimate the Q-function. Second, the analysis in [XWL20] relies on the compatible function approximation [SMSM00], which requires the critic to be a specific linear function class, while our analysis does not require such specific approximation, and therefore is more general. This makes our analysis potentially extendable to non-linear function approximation such as neural networks [CYJW19].

5.3 Preliminaries

In this section, we present the background of the two time-scale actor-critic algorithm.

5.3.1 Markov Decision Processes

Reinforcement learning tasks can be modeled as a discrete-time Markov Decision Process (MDP) $\mathcal{M} = \{\mathcal{S}, \mathcal{A}, \mathcal{P}, r\}$, where $\mathcal{S}$ and $\mathcal{A}$ are the state and action spaces respectively. In this work we consider the finite action space $|\mathcal{A}| < \infty$. $\mathcal{P}(s'|s, a)$ is the transition probability that the agent transits to state s' after taking action a at state s. Function $r : \mathcal{S} \times \mathcal{A} \to [-U_r, U_r]$ emits a bounded reward after the agent takes action a at state s, where $U_r > 0$ is a constant. A policy parameterized by $\boldsymbol{\theta}$ at state s is a probability function $\pi_{\boldsymbol{\theta}}(a|s)$ over action space $\mathcal{A}$. $\mu_{\boldsymbol{\theta}}$ denotes the stationary distribution induced by the policy $\pi_{\boldsymbol{\theta}}$.

In this work we consider the "average reward" setting [SMSM00], where under the ergodicity assumption, the average reward over time eventually converges to the expected reward under the stationary distribution:

$$r(\boldsymbol{\theta}) := \lim_{N\to\infty} \frac{\sum_{t=0}^{N} r(s_t, a_t)}{N} = \mathbb{E}_{s\sim\mu_{\boldsymbol{\theta}}, a\sim\pi_{\boldsymbol{\theta}}}\big[r(s,a)\big].$$

To evaluate the overall rewards given a starting state s_0 and the behavior policy $\pi_{\boldsymbol{\theta}}$, we define the state-value function as

$$V^{\pi_{\boldsymbol{\theta}}}(\cdot) := \mathbb{E}\bigg[\sum_{t=0}^{\infty}\big(r(s_t,a_t) - r(\boldsymbol{\theta})\big)|s_0 = \cdot\bigg],$$

where the action follows the policy $a_t \sim \pi_{\boldsymbol{\theta}}(\cdot|s_t)$ and the next state follows the transition probability $s_{t+1} \sim \mathcal{P}(\cdot|s_t, a_t)$. Another frequently used function is the state-action value function, also called Q-value function:

$$Q^{\pi_{\boldsymbol{\theta}}}(s,a) : = \mathbb{E}\bigg[\sum_{t=0}^{\infty}\big(r(s_t,a_t) - r(\boldsymbol{\theta})\big)|s_0 = s, a_0 = a\bigg] = r(s,a) - r(\boldsymbol{\theta}) + \mathbb{E}\big[V^{\pi_{\boldsymbol{\theta}}}(s')\big],$$

where the expectation is taken over $s' \sim \mathcal{P}(\cdot|s,a)$.

Throughout this chapter, we use O to denote the tuple $O = (s, a, s')$, some variants are like $O_t = (s_t, a_t, s_{t+1})$ and $\tilde{O}_t = (\tilde{s}_t, \tilde{a}_t, \tilde{s}_{t+1})$.

5.3.2 Policy Gradient Theorem

We define the performance function associated with policy $\pi_{\boldsymbol{\theta}}$ naturally as the expected reward under the stationary distribution $\mu_{\boldsymbol{\theta}}$ induced by $\pi_{\boldsymbol{\theta}}$, which takes the form

$$J(\boldsymbol{\theta}) : = r(\boldsymbol{\theta}). \tag{5.3.1}$$

To maximize the performance function with respect to the policy parameters, [SMSM00] proved the following policy gradient theorem.

Lemma 5.3.1 (Policy Gradient). *Consider the performance function defined in* (5.3.1)*, its gradient takes the form*

$$\nabla J(\boldsymbol{\theta}) = \mathbb{E}_{s\sim\mu_{\boldsymbol{\theta}}(\cdot)}\bigg[\sum_{a\in\mathcal{A}} Q^{\pi_{\boldsymbol{\theta}}}(s,a)\nabla\pi(a|s)\bigg].$$

The policy gradient also admits a neat form in expectation:

$$\nabla J(\boldsymbol{\theta}) = \mathbb{E}_{s\sim\mu_{\boldsymbol{\theta}}(\cdot),a\sim\pi_{\boldsymbol{\theta}}(\cdot|s)}\big[Q^{\pi_{\boldsymbol{\theta}}}(s,a)\nabla\log\pi_{\boldsymbol{\theta}}(a|s)\big].$$

A typical way to estimate the policy gradient $\nabla J(\boldsymbol{\theta})$ is by Monte Carlo method, namely using the summed return along the trajectory as the estimated Q-value, which is known as the "REINFORCE" method [Wil92].

Remark 5.3.2. *The problem formulation in this chapter is what [SMSM00] had defined as "average-reward" formulation. An alternative formulation is the "start-state" formulation, which avoids estimating the average reward, but gives a more complicated form for the policy-gradient algorithm and the AC algorithm.*

5.3.3 REINFORCE with a Baseline

Note that for any function $b(s)$ depending only on the state, which is usually called "baseline" function, we have

$$\sum_{a\in\mathcal{A}} b(s)\nabla\pi_{\boldsymbol{\theta}}(a|s) = b(s)\nabla\bigg(\sum_{a\in\mathcal{A}}\pi_{\boldsymbol{\theta}}(a|s)\bigg) = 0.$$

So we also have

$$\nabla J(\boldsymbol{\theta}) = \mathbb{E}\bigg[\sum_{a\in\mathcal{A}}\big(Q^{\pi_{\boldsymbol{\theta}}}(s,a) - b(s)\big)\nabla\pi_{\boldsymbol{\theta}}(a|s)\bigg].$$

A popular choice of $b(s)$ is $b(s) = V^{\pi_{\boldsymbol{\theta}}}(s)$ and $\Delta^{\pi_{\boldsymbol{\theta}}}(s,a) = Q^{\pi_{\boldsymbol{\theta}}}(s,a) - V^{\pi_{\boldsymbol{\theta}}}(s)$ is viewed as the advantage of taking a specific action a, compared with the expected reward at state s. Also note that the expectation form still holds:

$$\nabla J(\boldsymbol{\theta}) = \mathbb{E}_{s,a}\big[\Delta^{\pi_{\boldsymbol{\theta}}}(s,a)\nabla\log\pi_{\boldsymbol{\theta}}(a|s)\big].$$

Based on this fact, [Wil92] also proposed a corresponding policy gradient algorithm named "REINFORCE with a baseline" which performs better due to the reduced variance.

In practice the policy gradient method could suffer from high variance. An alternative approach is to introduce another trainable model to approximate the state-value function, which is called the actor-critic methods.

5.3.4 The Two Time-Scale Actor-Critic Algorithm

In previous subsection, we have seen how the policy gradient theorem appears in the form of the advantage value instead of the Q-value. Assume the critic uses linear function approximation $\hat{V}(\cdot;\boldsymbol{\omega}) = \boldsymbol{\phi}^\top(\cdot)\boldsymbol{\omega}$, and is updated by TD(0) algorithm, then this gives rise to Algorithm 9 that we are going to analyze.

Algorithm 9 has been proposed in many literature, and is clearly introduced in [SB18] as a classic on-line one-step actor-critic algorithm. It uses the advantage (namely temporal difference error) to update the critic and the actor simultaneously. Based on its on-line nature, this algorithm can be implemented both under episodic and continuing setting. In practice, the asynchronous variant of this algorithm, called Asynchronous Advantage Actor-Critic(A3C), is an empirically very successful parallel actor-critic algorithm.

Sometimes, Algorithm 9 is also called Advantage Actor-Critic (A2C) because it is the synchronous version of A3C and the name indicates its use of advantage instead of Q-value [MBM+16].

Algorithm 9 Two Time-Scale Actor-Critic

```
1: Input: initial actor parameter $\boldsymbol{\theta}_0$, initial critic parameter $\boldsymbol{\omega}_0$, initial average reward
   estimator $\eta_0$, step size $\alpha_t$ for actor, $\beta_t$ for critic and $\gamma_t$ for the average reward estimator.
2: Draw $s_0$ from some initial distribution
3: for $t = 0, 1, 2, \dots$ do
4:     Take the action $a_t \sim \pi_{\boldsymbol{\theta}_t}(\cdot|s_t)$
5:     Observe next state $s_{t+1} \sim \mathcal{P}(\cdot|s_t, a_t)$ and the reward $r_t = r(s_t, a_t)$
6:     $\delta_t = r_t - \eta_t + \boldsymbol{\phi}(s_{t+1})^\top \boldsymbol{\omega}_t - \boldsymbol{\phi}(s_t)^\top \boldsymbol{\omega}_t$
7:     $\eta_{t+1} = \eta_t + \gamma_t(r_t - \eta_t)$
8:     $\boldsymbol{\omega}_{t+1} = \Pi_{R_\omega}\big(\boldsymbol{\omega}_t + \beta_t \delta_t \boldsymbol{\phi}(s_t)\big)$
9:     $\boldsymbol{\theta}_{t+1} = \boldsymbol{\theta}_t + \alpha_t \delta_t \nabla_{\boldsymbol{\theta}} \log \pi_{\boldsymbol{\theta}_t}(a_t|s_t)$
10: end for
```

In Line 6 of Algorithm 9, the temporal difference error δ_t can be calculated based on the critic's estimation of the value function $\boldsymbol{\phi}(\cdot)^\top \boldsymbol{\omega}_t$, where $\boldsymbol{\omega}_t \in \mathbb{R}^d$ and $\phi(\cdot) : \mathcal{S} \to \mathbb{R}^d$ is a known feature mapping. Then the critic will be updated using the semi-gradient from TD(0) method. Line 8 in Algorithm 9 also contains a projection operator. This is required to control the algorithm's convergence which also appears in some other literature [BRS18, XZL19]. The actor uses the advantage δ_t (estimated by critic) and the samples to get an estimation of the policy gradient.

Algorithm 9 is more general and practical than the algorithms analyzed in many previous work [QYYW19, KKR19]. In our algorithm, there is no need for independent samples or samples from the stationary distribution. There is only one naturally generated sample path. Also, the critic inherits from last iteration and continuously updates its parameter, without requiring a restarted sample path (or a new episode).

5.4 Main Theory

In this section, we first discuss on some standard assumptions used in the literature for deriving the convergence of reinforcement learning algorithms and then present our theoretical results for two time-scale actor-critic methods.

5.4.1 Assumptions and Propositions

We consider the setting where the critic uses TD [SB18] with linear function approximation to estimate the state-value function, namely $\hat{V}(\cdot;\boldsymbol{\omega}) = \boldsymbol{\phi}^\top(\cdot)\boldsymbol{\omega}$. We assume that the feature mapping has bounded norm $\|\boldsymbol{\phi}(\cdot)\| \leq 1$. Denote by $\boldsymbol{\omega}^*(\boldsymbol{\theta})$ the limiting point of TD(0) algorithms under the behavior policy $\pi_{\boldsymbol{\theta}}$, and define $\mathbf{A}$ and $\mathbf{b}$ as:

$$\mathbf{A} := \mathbb{E}_{s,a,s'}\big[\boldsymbol{\phi}(s)\big(\boldsymbol{\phi}(s') - \boldsymbol{\phi}(s)\big)^\top\big],$$
$$\mathbf{b} := \mathbb{E}_{s,a,s'}[(r(s,a) - r(\boldsymbol{\theta}))\boldsymbol{\phi}(s)],$$

where $s \sim \mu_{\boldsymbol{\theta}}(\cdot), a \sim \pi_{\boldsymbol{\theta}}(\cdot|s), s' \sim \mathcal{P}(\cdot|s,a)$. It is known that the TD limiting point satisfies:

$$\mathbf{A}\boldsymbol{\omega}^*(\boldsymbol{\theta}) + \mathbf{b} = \mathbf{0}.$$

In the sequel, when there is no confusion, we will use a shorthand notation $\boldsymbol{\omega}^*$ to denote $\boldsymbol{\omega}^*(\boldsymbol{\theta})$. Based on the complexity of the feature mapping, the approximation error of this function class can vary. The approximation error of the linear function class is defined as follows:

$$\epsilon_{\text{app}}(\boldsymbol{\theta}) := \sqrt{\mathbb{E}_{s\sim\mu_{\boldsymbol{\theta}}}\big(\boldsymbol{\phi}(s)^\top\boldsymbol{\omega}^*(\boldsymbol{\theta}) - V^{\pi_{\boldsymbol{\theta}}}(s)\big)^2}.$$

Throughout this chapter, we assume the approximation error for all potential policies is uniformly bounded,

$$\forall\boldsymbol{\theta}, \epsilon_{\text{app}}(\boldsymbol{\theta}) \leq \epsilon_{\text{app}},$$

for some constant $\epsilon_{\text{app}} \geq 0$.

In the analysis of TD learning, the following assumption is often made to ensure the uniqueness of the limiting point of TD and the problem's solvability.

Assumption 5.4.1. *For all potential policy parameters $\boldsymbol{\theta}$, the matrix $\mathbf{A}$ defined above is negative definite and has the maximum eigenvalues as $-\lambda$.*

Assumption 5.4.1 is often made to guarantee the problem's solvability [BRS18, ZXL19, XZL19]. Note that Algorithm 9 contains a projection step at Line 8. To guarantee convergence it is required all $\boldsymbol{\omega}^*$ lie within this projection radius $R_{\boldsymbol{\omega}}$. Assumption 5.4.1 indicates that a sufficient condition is to set $R_\omega = 2U_r/\lambda$ because $\|\mathbf{b}\| \leq 2U_r$ and $\|\mathbf{A}^{-1}\| \leq \lambda^{-1}$.

The next assumption, first adopted by [BRS18] in TD learning, addresses the issue of Markovian noise.

Assumption 5.4.2 (Uniform ergodicity)**.** *For a fixed $\boldsymbol{\theta}$, denote $\mu_{\boldsymbol{\theta}}(\cdot)$ as the stationary distribution induced by the policy $\pi_{\boldsymbol{\theta}}(\cdot|s)$ and the transition probability measure $\mathcal{P}(\cdot|s,a)$. Consider a Markov chain generated by the rule $a_t \sim \pi_{\boldsymbol{\theta}}(\cdot|s_t), s_{t+1} \sim \mathcal{P}(\cdot|s_t, a_t)$. Then there exists $m > 0$ and $\rho \in (0,1)$ such that:*

$$d_{TV}\big(\mathbb{P}(s_\tau \in \cdot|s_0 = s), \mu_{\boldsymbol{\theta}}(\cdot)\big) \leq m\rho^\tau, \forall\tau \geq 0, \forall s \in \mathcal{S}.$$

We also need some regularity assumptions on the policy.

Assumption 5.4.3. *Let $\pi_{\boldsymbol{\theta}}(a|s)$ be a policy parameterized by $\boldsymbol{\theta}$. There exist constants $L, B, L_l > 0$ such that for all given state s and action a it holds*

(a) $\left\|\nabla \log \pi_{\boldsymbol{\theta}}(a|s)\right\| \le B$, $\forall \boldsymbol{\theta} \in \mathbb{R}^d$,

(b) $\left\|\nabla \log \pi_{\boldsymbol{\theta}_1}(a|s) - \nabla \log \pi_{\boldsymbol{\theta}_2}(a|s)\right\| \le L_l \|\boldsymbol{\theta}_1 - \boldsymbol{\theta}_2\|$, $\forall \boldsymbol{\theta}_1, \boldsymbol{\theta}_2 \in \mathbb{R}^d$,

(c) $\left|\pi_{\boldsymbol{\theta}_1}(a|s) - \pi_{\boldsymbol{\theta}_2}(a|s)\right| \le L \|\boldsymbol{\theta}_1 - \boldsymbol{\theta}_2\|$, $\forall \boldsymbol{\theta}_1, \boldsymbol{\theta}_2 \in \mathbb{R}^d$.

The first two inequalities are regularity conditions to guarantee actor's convergence in the literature of policy gradient [PBC$^+$18, ZKZB19, KKR19, XGG19, XGG20]. The last inequality in Assumption 5.4.3 is also adopted by [ZXL19] when analyzing SARSA.

An important fact arises from our assumptions is that the limiting point $\boldsymbol{\omega}^*$ of TD(0) , which can be viewed as a mapping of the policy's parameter $\boldsymbol{\theta}$, is Lipschitz.

Proposition 5.4.4. *Under Assumptions 5.4.1 and 5.4.2, there exists a constant $L_* > 0$ such that*

$$\left\|\boldsymbol{\omega}^*(\boldsymbol{\theta}_1) - \boldsymbol{\omega}^*(\boldsymbol{\theta}_2)\right\| \le L_* \|\boldsymbol{\theta}_1 - \boldsymbol{\theta}_2\|, \forall \boldsymbol{\theta}_1, \boldsymbol{\theta}_2 \in \mathbb{R}^d.$$

Proposition 5.4.4 states that the target point $\boldsymbol{\omega}^*$ moves slowly compared with the actor's update on $\boldsymbol{\theta}$. This is an observation pivotal to the two time-scale analysis. Specifically, the two time-scale analysis can be informally described as "the actor moves slowly while the critic chases the slowly moving target determined by the actor".

Now we are ready to present the convergence result of two time-scale actor-critic methods. We first define an integer that depends on the learning rates α_t and β_t.

$$\tau_t := \min\left\{i \ge 0 | m\rho^{i-1} \le \min\{\alpha_t, \beta_t\}\right\}, \tag{5.4.1}$$

where m, ρ are defined as in Assumption 5.4.2. By definition, τ_t is a mixing time of an ergodic Markov chain. We will use τ_t to control the Markovian noise encountered in the training process.

5.4.2 Convergence of the Actor

At the k-th iteration of the actor's update, $\boldsymbol{\omega}_k$ is the critic parameter estimated by Line 7 of Algorithm 9 and $\boldsymbol{\omega}_k^*$ is the unknown parameter of value function $V^{\pi_{\boldsymbol{\theta}_k}}(\cdot)$ defined in Assumption 5.4.1. The following theorem gives the convergence rate of the actor when the averaged mean squared error between $\boldsymbol{\omega}_k$ and $\boldsymbol{\omega}_k^*$ and the error between η_k and $r(\boldsymbol{\theta}_k)$ from $k = \tau_t$ to $k = t$ are small.

Theorem 5.4.5. *Suppose Assumptions 5.4.1-5.4.3 hold and we choose $\alpha_t = c_\alpha/(1+t)^\sigma$ in Algorithm 9, where $\sigma \in (0,1)$ and $c_\alpha > 0$ are constants. If we assume at the t-th iteration, the critic satisfies*

$$\frac{8}{t}\sum_{k=1}^{t}\mathbb{E}\|\boldsymbol{\omega}_k - \boldsymbol{\omega}_k^*\|^2 + \frac{2}{t}\sum_{k=1}^{t}\mathbb{E}\big(\eta_k - r(\boldsymbol{\theta}_k)\big)^2 = \mathcal{E}(t), \tag{5.4.2}$$

where $\mathcal{E}(t)$ is a bounded sequence, then we have

$$\min_{0\le k\le t}\mathbb{E}\big\|\nabla J(\boldsymbol{\theta}_k)\big\|^2 = O(\epsilon_{app}) + O\bigg(\frac{1}{t^{1-\sigma}}\bigg) + O\bigg(\frac{\log^2 t}{t^\sigma}\bigg) + O\big(\mathcal{E}(t)\big),$$

where $O(\cdot)$ hides constants, whose exact forms can be found in the detailed proof in Appendix 5.7.1.

Note that $\mathcal{E}(t)$ in Theorem 5.4.5 is the averaged estimation error made by the critic throughout the learning process, which will be bounded in the next Theorem 5.4.7.

Remark 5.4.6. *Theorem 5.4.5 recovers the results for the decoupled case [QYYW19, KKR19] by setting $\sigma = 1/2$. Nevertheless, we are considering a much more practical and challenging case where the actor and critic are simultaneously updated under Markovian noises. It is worth noting that the non-i.i.d. data assumption leads to an additional logarithm term, which is also observed in [BRS18, ZXL19, SY19, CZD+19].*

5.4.3 Convergence of the Critic

The condition in (5.4.2) is guaranteed by the following theorem that characterizes the convergence of the critic.

Theorem 5.4.7. *Suppose Assumptions 5.4.1-5.4.3 hold and we choose* $\alpha_t = c_\alpha/(1+t)^\sigma$ *and* $\beta_t = c_\beta/(1+t)^\nu$ *in Algorithm 9, where* $0 < \nu < \sigma < 1$, c_α *and* $c_\beta \le \lambda^{-1}$ *are positive constants. Then we have*

$$\frac{1}{1+t-\tau_t}\sum_{k=\tau_t}^{t}\mathbb{E}\|\boldsymbol{\omega}_k - \boldsymbol{\omega}_k^*\|^2 = O\bigg(\frac{1}{t^{1-\nu}}\bigg) + O\bigg(\frac{\log t}{t^\nu}\bigg) + O\bigg(\frac{1}{t^{2(\sigma-\nu)}}\bigg), \tag{5.4.3}$$

$$\frac{1}{1+t-\tau_t}\sum_{k=\tau_t}^{t}\mathbb{E}\big(\eta_k - r(\boldsymbol{\theta}_k)\big)^2 = O\bigg(\frac{1}{t^{1-\nu}}\bigg) + O\bigg(\frac{\log t}{t^\nu}\bigg) + O\bigg(\frac{1}{t^{2(\sigma-\nu)}}\bigg), \tag{5.4.4}$$

where $O(\cdot)$ *hides constants, whose exact forms can be found in the detailed proof in Appendix 5.7.2 and 5.7.3.*

Remark 5.4.8. *The first term* $O(t^{\nu-1})$ *on the right hand side of* (5.4.3) *and* (5.4.4) *comes from loosely bounding the error's norm, and can be removed by applying the "iterative refinement" technique used in [XZL19]. Using this technique, we can obtain a bound (also holds for* η_t*)* $\mathbb{E}\|\boldsymbol{\omega}_t - \boldsymbol{\omega}_t^*\|^2 = O(\log t/t^\nu) + O(1/t^{2(\sigma-\nu)-\xi})$, *where* $\xi > 0$ *is an arbitrarily small constant. The constant* ξ *is an artifact due to the the "iterative refinement" technique. Similar simplification can be done for* (5.4.4). *Nevertheless, if we plug* (5.4.3) *and* (5.4.4) *(after some transformation) into the result of Theorem 5.4.5, it is easy to see that the term* $O(1/t^{1-\nu})$ *is actually dominated by the term* $O(1/t^{1-\sigma})$. *Thus this term makes no difference in the total sample complexity of Algorithm 9 and we choose not to complicate the proof or introduce the extra artificial parameter* ξ *in the result of Theorem 5.4.7.*

The second term in both (5.4.3) *and* (5.4.4) *comes from the Markovian noise and the variance of the semi-gradient. The third term in these two equations comes from the slow drift of the actor. These two terms together can be interpreted as follows: if the actor moves much slower than the critic (i.e.,* $\sigma - \nu \gg \nu$*), then the error is dominated by the Markovian noise and gradient variance; if the actor moves not too slowly compared with the critic (i.e.* $\sigma - \nu \ll \nu$*), then the critic's error is dominated by the slowly drifting effect of the actor.*

5.4.4 Convergence Rate and Sample Complexity

Combining Theorems 5.4.5 and 5.4.7 leads to the following convergence rate and sample complexity for Algorithm 9. The detailed proof is in Appendix 5.7.4.

Corollary 5.4.9. *Under the same assumptions of Theorems 5.4.5 and 5.4.7, we have*

$$\min_{0\le k\le t}\mathbb{E}\|\nabla J(\boldsymbol{\theta}_k)\|^2 = O(\epsilon_{app}) + O\bigg(\frac{1}{t^{1-\sigma}}\bigg) + O\bigg(\frac{\log t}{t^{\nu}}\bigg) + O\bigg(\frac{1}{t^{2(\sigma-\nu)}}\bigg).$$

If we set $\sigma = 3/5, \nu = 2/5$, leading to the actor step size $\alpha_t = O(1/t^{3/5})$ and the critic step size $\beta_t = O(1/t^{2/5})$, Algorithm 9 can find an ϵ-approximate stationary point of $J(\cdot)$ within T steps, namely,

$$\min_{0\le k\le T}\mathbb{E}\big\|\nabla J(\boldsymbol{\theta}_k)\big\|^2 \le O(\epsilon_{app}) + \epsilon,$$

where $T = \tilde{O}(\epsilon^{-2.5})$ is the total iteration number.

Corollary 5.4.9 combines the results of Theorems 5.4.5 and 5.4.7 and shows that the convergence rate of Algorithm 9 is $\tilde{O}(t^{-2/5})$. Since the per iteration sample is 1, the sample complexity of two time-scale actor-critic is $\tilde{O}(\epsilon^{-2.5})$.

Remark 5.4.10. *We compare our results with existing results on the sample complexity of actor-critic methods in the literature. [KKR19] provided a general result that after $T = O(\epsilon^{-2})$ updates for the actor, the algorithm can achieve $\min_{0\le k\le T}\mathbb{E}\|\nabla J(\boldsymbol{\theta}_k)\|^2 \le \epsilon$, as long as the estimation error of the critic can be bounded by $O(t^{-1/2})$ at the t-th actor's update. However, to ensure such a condition on the critic, they need to draw t samples to estimate the critic at the t-th actor's update. Therefore, the total number of samples drawn from the whole training process by the actor-critic algorithm in [KKR19] is $O(T^2)$, yielding a $O(\epsilon^{-4})$ sample complexity. Under the similar setting, [QYYW19] proved the same sample complexity $\tilde{O}(\epsilon^{-4})$ when TD(0) is used for estimating the critic. Thus Corollary 5.4.9 suggests that the sample complexity of Algorithm 9 is significantly better than the sample complexity presented in [KKR19, QYYW19] by a factor of $O(\epsilon^{-1.5})$.*

Remark 5.4.11. *The gap between the "decoupled" actor-critic and the two time-scale actor-critic seems huge. Intuitively, this is due to the inefficient usage of the samples. At each iteration, the critic in the "decoupled" algorithm starts over to evaluate the policy's value function and discards the history information, regardless of the fact that the policy might only changed slightly. The two time-scale actor-critic keeps the critic's parameter and thus takes full advantage of each samples in the trajectory.*

Remark 5.4.12. *According to [PBC+18], the sample complexity of policy gradient methods such as REINFORCE is $O(\epsilon^{-2})$. As a comparison, if the critic converges faster than $O(t^{-1/2})$, namely $\mathcal{E}(t) = O(t^{-1/2})$, then Theorem 5.4.5 combined with Corollary 5.4.9 implies that the complexity of two time-scale actor-critic is $\tilde{O}(\epsilon^{-2})$, which matches the result of policy gradient methods [PBC+18] up to logarithmic factors. Nevertheless, as we have discussed in the previous remarks, a smaller estimation error for critic often comes at the cost of more samples needed for the critic update [QYYW19, KKR19], which eventually increases the total sample complexity. Therefore, the $\tilde{O}(\epsilon^{-2.5})$ sample complexity in Corollary 5.4.9 is indeed the lowest we can achieve so far for classic two time-scale actor-critic methods. However, it is possible to further improve the sample complexity by using policy evaluation algorithms better than vanilla TD(0), such as GTD and TDC methods.*

5.5 Proof Sketch

In this section, we provide the proof roadmap of the main theory. Detailed proofs can be found in Appendix 5.7.

5.5.1 Proof Sketch of Theorem 5.4.5

The following lemma is important in that it enables the analysis of policy gradient method:

Lemma 5.5.1 ([ZKZB19]). *For the performance function defined in* (5.3.1)*, there exists a constant $L_J > 0$ such that for all $\boldsymbol{\theta}_1, \boldsymbol{\theta}_2 \in \mathbb{R}^d$, it holds that*

$$\big\|\nabla J(\boldsymbol{\theta}_1) - \nabla J(\boldsymbol{\theta}_2)\big\| \le L_J \|\boldsymbol{\theta}_1 - \boldsymbol{\theta}_2\|,$$

which by the definition of smoothness [Nes18] is also equivalent to

$$J(\boldsymbol{\theta}_2) \ge J(\boldsymbol{\theta}_1) + \big\langle \nabla J(\boldsymbol{\theta}_1), \boldsymbol{\theta}_2 - \boldsymbol{\theta}_1 \big\rangle - \frac{L_J}{2}\|\boldsymbol{\theta}_1 - \boldsymbol{\theta}_2\|^2.$$

This lemma enables us to perform a gradient ascent style analysis on the non-concave function $J(\boldsymbol{\theta})$:

$$J(\boldsymbol{\theta}_{t+1}) \ge J(\boldsymbol{\theta}_t) + \alpha_t \big\langle \nabla J(\boldsymbol{\theta}_t), \delta_t \nabla \log \pi_{\boldsymbol{\theta}_t}(a_t|s_t) \big\rangle - L_J \alpha_t^2 \big\|\delta_t \nabla \log \pi_{\boldsymbol{\theta}_t}(a_t|s_t)\big\|^2$$

$$
\begin{aligned}
&= J(\boldsymbol{\theta}_t) + \alpha_t \big\langle \nabla J(\boldsymbol{\theta}_t), \Delta h(O_t, \eta_t, \boldsymbol{\omega}_t, \boldsymbol{\theta}_t) \big\rangle + \alpha_t \big\langle \nabla J(\boldsymbol{\theta}_t), \Delta h'(O_t, \boldsymbol{\theta}_t) \big\rangle \\
&\quad + \alpha_t \Gamma(O_t, \boldsymbol{\theta}_t) + \alpha_t \big\| \nabla J(\boldsymbol{\theta}_t) \big\|^2 - L_J \alpha_t^2 \big\| \delta_t \nabla \log \pi_{\boldsymbol{\theta}_t}(a_t|s_t) \big\|^2, \qquad (5.5.1)
\end{aligned}
$$

where $O_t = (s_t, a_t, s_{t+1})$ is a tuple of observations. The second term $\Delta h(O_t, \boldsymbol{\omega}_t, \boldsymbol{\theta}_t)$ on the right hand side of (5.5.1) is the bias introduced by the critic. The third term $\Delta h'(O_t, \boldsymbol{\theta}_t)$ is from the linear approximation error. The fourth term $\Gamma(O_t, \boldsymbol{\theta}_t)$ is due to the Markovian noise. The last term can be viewed as the variance of the stochastic gradient update. Please refer to (5.7.1) for the definition of each notation.

Now we bound each term's expectation in (5.5.1) respectively.

First, we have

$$
\mathbb{E}\big\langle \nabla J(\boldsymbol{\theta}_t), \Delta h(O_t, \eta_t, \boldsymbol{\omega}_t, \boldsymbol{\theta}_t) \big\rangle \ge -B \sqrt{\mathbb{E}\big\| \nabla J(\boldsymbol{\theta}_t) \big\|^2} \sqrt{8\mathbb{E}\|\mathbf{z}_t\|^2 + 2\mathbb{E}[y_t^2]},
$$

where $\mathbf{z}_t := \boldsymbol{\omega}_t - \boldsymbol{\omega}_t^*$ and $y_t := \eta_t - \eta_t^*$, and the inequality is due to Cauchy inequality and Lemma 5.7.2.

Second, taking expectation over the approximation error term containing $\Delta h'$, we have

$$
\begin{aligned}
\mathbb{E}\big\langle \nabla J(\boldsymbol{\theta}_t), \Delta h'(O_t, \boldsymbol{\theta}_t) \big\rangle &\ge -G_{\boldsymbol{\theta}} \sqrt{\mathbb{E}\big\| \Delta h'(O_t, \boldsymbol{\theta}_t) \big\|^2} \\
&\ge -G_{\boldsymbol{\theta}} \cdot 2B \sqrt{\mathbb{E}\big(\phi(s)^\top \boldsymbol{\omega}_t^* - V^{\pi_{\boldsymbol{\theta}_t}}(s)\big)^2} \\
&\ge -2BG_{\boldsymbol{\theta}} \epsilon_{\text{app}},
\end{aligned}
$$

Third, we have

$$
\begin{aligned}
\mathbb{E}[\Gamma(O_t, \boldsymbol{\theta}_t)] &\ge -G_{\boldsymbol{\theta}} \bigg(D_1(\tau+1) \sum_{k=t-\tau+1}^{t} \mathbb{E}\|\boldsymbol{\theta}_k - \boldsymbol{\theta}_{k-1}\| + D_2 m \rho^{\tau-1} \bigg), \\
&\ge -G_{\boldsymbol{\theta}} \bigg(D_1(\tau+1) G_{\boldsymbol{\theta}} \sum_{k=t-\tau+1}^{t-1} \alpha_k + D_2 m \rho^{\tau-1} \bigg),
\end{aligned}
$$

where the first inequality is due to Lemma 5.7.3, and the second inequality is due to $\big\| \delta_t \nabla \log \pi_{\boldsymbol{\theta}_t}(a_t|s_t) \big\| \le G_{\boldsymbol{\theta}}$ by Lemma 5.7.3. Taking the expectation of (5.7.2), plugging the above terms back into it and rearranging give

$$
\mathbb{E}\big\| \nabla J(\boldsymbol{\theta}_t) \big\|^2 \le \alpha_t^{-1} \big(\mathbb{E}[J(\boldsymbol{\theta}_{t+1})] - \mathbb{E}[J(\boldsymbol{\theta}_t)] \big) + B \sqrt{\mathbb{E}\big\| \nabla J(\boldsymbol{\theta}_t) \big\|^2} \sqrt{8\mathbb{E}\|\mathbf{z}_t\|^2 + 2\mathbb{E}[y_t^2]}
$$

$$+ D_1 G_{\boldsymbol{\theta}}^2(\tau+1)\sum_{k=t-\tau}^{t-1}\alpha_k + D_2 G_{\boldsymbol{\theta}} m\rho^{\tau-1} + L_J G_{\boldsymbol{\theta}}^2 \alpha_t.$$

Setting $\tau = \tau_t$ and summing over each term, and further dividing $(1 + t - \tau_t)$ at both sides and assuming $t > 2\tau_t - 1$, we can express the result as

$$\frac{1}{1+t-\tau_t}\sum_{k=\tau_t}^{t}\mathbb{E}\big\|\nabla J(\boldsymbol{\theta}_t)\big\|^2 \le O\bigg(\frac{1}{t^{1-\sigma}}\bigg) + O\bigg(\frac{(\log t)^2}{t^{\sigma}}\bigg) + O(\epsilon_{\text{app}})$$
$$+ \frac{2B}{1+t-\tau_t}\sum_{k=\tau_t}^{t}\sqrt{\mathbb{E}\big\|\nabla J(\boldsymbol{\theta}_t)\big\|^2}\sqrt{8\mathbb{E}\|\mathbf{z}_t\|^2 + 2\mathbb{E}[y_t^2]}. \quad (5.5.2)$$

By Cauchy-Schwartz inequality, we have

$$\frac{1}{1+t-\tau_t}\sum_{k=\tau_t}^{t}\sqrt{\mathbb{E}\big\|\nabla J(\boldsymbol{\theta}_t)\big\|^2}\sqrt{8\mathbb{E}\|\mathbf{z}_t\|^2 + 2\mathbb{E}[y_t^2]}$$
$$\le \bigg(\frac{1}{1+t-\tau_t}\sum_{k=\tau_t}^{t}\mathbb{E}\big\|\nabla J(\boldsymbol{\theta}_t)\big\|^2\bigg)^{\frac{1}{2}}\bigg(\frac{1}{1+t-\tau_t}\sum_{k=\tau_t}^{t}\big(8\mathbb{E}\|\mathbf{z}_t\|^2 + 2\mathbb{E}[y_t^2]\big)\bigg)^{\frac{1}{2}}.$$

Now, denote $F(t) := 1/(1+t-\tau_t)\sum_{k=\tau_t}^{t}\mathbb{E}\|\nabla J(\boldsymbol{\theta}_k)\|^2$ and $Z(t) := 1/(1+t-\tau_t)\sum_{k=\tau_t}^{t}\big(8\mathbb{E}\|\mathbf{z}_t\|^2 + 2\mathbb{E}[y_t^2]\big)$, and putting them back to (5.5.2) (O-notation for simplicity):

$$F(t) \le O\bigg(\frac{1}{t^{1-\sigma}}\bigg) + O\bigg(\frac{(\log t)^2}{t^{\sigma}}\bigg) + O(\epsilon_{\text{app}}) + 2B\sqrt{F(t)}\cdot\sqrt{Z(t)},$$

which further gives

$$\big(\sqrt{F(t)} - B\sqrt{Z(t)}\big)^2 \le O\bigg(\frac{1}{t^{1-\sigma}}\bigg) + O\bigg(\frac{(\log t)^2}{t^{\sigma}}\bigg) + O(\epsilon_{\text{app}}) + B^2 Z(t).$$

Note that for a general function $H(t) = A(t) + B(t)$(with each positive), we have

$$H^2(t) = O\big(A^2(t)\big) + O\big(B^2(t)\big), \quad \sqrt{H(t)} = O\big(\sqrt{A(t)}\big) + O\big(\sqrt{B(t)}\big).$$

This means

$$\min_{0\le k\le t}\mathbb{E}\big\|\nabla J(\boldsymbol{\theta}_k)\big\|^2 \le \frac{1}{1+t-\tau_t}\sum_{k=\tau_t}^{t}\mathbb{E}\big\|\nabla J(\boldsymbol{\theta}_k)\big\|^2$$
$$= O\bigg(\frac{1}{t^{1-\sigma}}\bigg) + O\bigg(\frac{1}{t^{\sigma}}\bigg) + O(\epsilon_{\text{app}}) + O\big(\mathcal{E}(t)\big).$$

5.5.2 Proof Sketch of Theorem 5.4.7

The proof of Theorem 5.4.7 can be divided into the following two parts.

5.5.2.1 Estimating the Average Reward η_k

We denote $y_k := \eta_k - r(\boldsymbol{\theta}_k)$. First, we shall mention that many components in this step is uses the same framework and partial result as the proof regarding $\boldsymbol{\omega}_t$ in the next part. Also, part of the proof is intriguingly similar with the proof of Theorem 5.4.5. For simplicity, here we only present the final result regarding η_k. Please refer to Section 5.7.2 for the detailed proof. By setting $\gamma_k = (1+t)^{-\nu}$, we have that

$$\sum_{k=\tau_t}^{t} \mathbb{E}[y_k^2] = O(t^{\nu}) + O(\log t \cdot t^{1-\nu}) + O(t^{1-2(\sigma-\nu)}).$$

5.5.2.2 Approximating the TD Fixed Point

Step 1: decomposition of the estimation error. For simplicity, we denote $\mathbf{z}_t := \boldsymbol{\omega}_t - \boldsymbol{\omega}_t^*$, where the $\boldsymbol{\omega}_t^*$ denotes the exact parameter under policy $\pi_{\boldsymbol{\theta}_t}$. By the critic update in Line 7 of Algorithm 9, we have

$$\begin{aligned}\|\mathbf{z}_{t+1}\|^2 = \|\mathbf{z}_t\|^2 &+ 2\beta_t\big\langle \mathbf{z}_t, \bar{g}(\boldsymbol{\omega}_t, \boldsymbol{\theta}_t)\big\rangle + 2\beta_t\Lambda(O_t, \boldsymbol{\omega}_t, \boldsymbol{\theta}_t) + 2\beta_t\big\langle \mathbf{z}_t, \Delta g(O_t, \eta_t, \boldsymbol{\theta}_t)\big\rangle \\ &+ 2\langle \mathbf{z}_t, \boldsymbol{\omega}_t^* - \boldsymbol{\omega}_{t+1}^*\rangle + \big\|\beta_t(g(O_t, \boldsymbol{\omega}_t, \boldsymbol{\theta}_t) + \Delta g(O_t, \eta_t, \boldsymbol{\theta}_t)) + (\boldsymbol{\omega}_t^* - \boldsymbol{\omega}_{t+1}^*)\big\|^2.\end{aligned} \tag{5.5.3}$$

where $O_t := (s_t, a_t, s_{t+1})$ is a tuple of observations, $g(O_t, \boldsymbol{\omega}_t)$ and $\bar{g}(\boldsymbol{\theta}_t, \boldsymbol{\omega}_t)$ are the estimated gradient and the true gradient respectively. $\Lambda(O_t, \boldsymbol{\omega}_t, \boldsymbol{\theta}_t) := \langle \boldsymbol{\omega}_t - \boldsymbol{\omega}_t^*, g(O_t, \boldsymbol{\omega}_t) - \bar{g}(\boldsymbol{\theta}_t, \boldsymbol{\omega}_t)\rangle$ can be seen as the error induced by the Markovian noise. Please refer to (5.7.7) for formal definition of each notation.

The second term on the right hand side of (5.5.3) can be bounded by $-2\lambda\beta_t\|\mathbf{z}_t\|^2$ due to Assumption 5.4.1. The third term is a bias term caused by the Markovian noise. The fourth term $\Delta g(O_t, \eta_t, \boldsymbol{\theta}_t)$ is another bias term caused by inaccurate average reward estimator η_t. The fifth term is caused by the slowly drifting policy parameter $\boldsymbol{\theta}_t$. And the last term can be considered as the variance term.

Rewriting (5.5.3) and telescoping from $\tau = \tau_t$ to t, we have

$$
\begin{aligned}
2\lambda \sum_{k=\tau_t}^{t} \mathbb{E}\|\mathbf{z}_k\|^2 \leq &\underbrace{\sum_{k=\tau_t}^{t} \frac{1}{\beta_k}\left(\mathbb{E}\|\mathbf{z}_k\|^2 - \mathbb{E}\|\mathbf{z}_{k+1}\|^2\right)}_{I_1} + 2\underbrace{\sum_{k=\tau_t}^{t} \mathbb{E}\Lambda(\boldsymbol{\theta}_k, \boldsymbol{\omega}_k, O_k)}_{I_2} \\
&+ 2L_* G_{\boldsymbol{\theta}} \underbrace{\sum_{k=\tau_t}^{t} \frac{\alpha_k}{\beta_k}\sqrt{\mathbb{E}\|\mathbf{z}_k\|}}_{I_3} + \underbrace{\sum_{k=\tau_t}^{t} \sqrt{\mathbb{E}[y_k^2]} \cdot \sqrt{\mathbb{E}\|\mathbf{z}_k\|}}_{I_4} + C_q \underbrace{\sum_{k=\tau_t}^{t} \beta_k}_{I_5}. \qquad (5.5.4)
\end{aligned}
$$

We will see that the Markovian noise I_2, the "slowly drifting policy" term I_3 and the estimation bias I_4 from η_t are significant, and bounding the Markovian term is another challenge.

Step 2: bounding the Markovian bias. We first decompose $\Lambda(\boldsymbol{\theta}_t, \boldsymbol{\omega}_t, O_t)$ as follows.

$$
\begin{aligned}
\Lambda(\boldsymbol{\theta}_t, \boldsymbol{\omega}_t, O_t) = &\left(\Lambda(\boldsymbol{\theta}_t, \boldsymbol{\omega}_t, O_t) - \Lambda(\boldsymbol{\theta}_{t-\tau}, \boldsymbol{\omega}_t, O_t)\right) + \left(\Lambda(\boldsymbol{\theta}_{t-\tau}, \boldsymbol{\omega}_t, O_t) - \Lambda(\boldsymbol{\theta}_{t-\tau}, \boldsymbol{\omega}_{t-\tau}, O_t)\right) \\
&+ \left(\Lambda(\boldsymbol{\theta}_{t-\tau}, \boldsymbol{\omega}_{t-\tau}, O_t) - \Lambda(\boldsymbol{\theta}_{t-\tau}, \boldsymbol{\omega}_{t-\tau}, \tilde{O}_t)\right) + \Lambda(\boldsymbol{\theta}_{t-\tau}, \boldsymbol{\omega}_{t-\tau}, \tilde{O}_t). \qquad (5.5.5)
\end{aligned}
$$

The motivation is to employ the uniform ergodicity defined by Assumption 5.4.2. This technique was first introduced by [BRS18] to address the Markovian noise in policy evaluation. [ZXL19] extended to the Q-learning setting where the parameter itself both keeps updated and determines the behavior policy. In this work we take one step further to consider that the policy parameter $\boldsymbol{\theta}_t$ is changing, and the evaluation parameter $\boldsymbol{\omega}_t$ is updated. The analysis relies on the auxiliary Markov chain constructed by [ZXL19], which is obtained by repeatedly applying policy $\pi_{\boldsymbol{\theta}_{t-\tau}}$:

$$
s_{t-\tau} \xrightarrow{\boldsymbol{\theta}_{t-\tau}} a_{t-\tau} \xrightarrow{\mathcal{P}} s_{t-\tau+1} \xrightarrow{\boldsymbol{\theta}_{t-\tau}} \tilde{a}_{t-\tau+1} \xrightarrow{\mathcal{P}} \tilde{s}_{t-\tau+2} \xrightarrow{\boldsymbol{\theta}_{t-\tau}} \tilde{a}_{t-\tau+2} \xrightarrow{\mathcal{P}} \cdots \xrightarrow{\mathcal{P}} \tilde{s}_t \xrightarrow{\boldsymbol{\theta}_{t-\tau}} \tilde{a}_t \xrightarrow{\mathcal{P}} \tilde{s}_{t+1}.
$$

For reference, recall that the original Markov chain is given by:

$$
s_{t-\tau} \xrightarrow{\boldsymbol{\theta}_{t-\tau}} a_{t-\tau} \xrightarrow{\mathcal{P}} s_{t-\tau+1} \xrightarrow{\boldsymbol{\theta}_{t-\tau+1}} a_{t-\tau+1} \xrightarrow{\mathcal{P}} s_{t-\tau+2} \xrightarrow{\boldsymbol{\theta}_{t-\tau+2}} a_{t-\tau+2} \xrightarrow{\mathcal{P}} \cdots \xrightarrow{\mathcal{P}} s_t \xrightarrow{\boldsymbol{\theta}_t} a_t \xrightarrow{\mathcal{P}} s_{t+1}.
$$

By Lipschitz conditions, we can bound the first two terms in (5.5.5). The third term will be bounded by the total variation between s_k and $\tilde{s}_k$, which is achieved by recursively bounding total variation between s_{k-1} and $\tilde{s}_{k-1}$.

In fact, the Markovian noise $\Gamma(O_t, \boldsymbol{\theta}_t)$ in Section 5.7.1 is obtained in a similar way. Due to the space limit, we only present how to bound the more complicated $\Lambda(\boldsymbol{\theta}_t, \boldsymbol{\omega}_t, O_t)$.

We have the final form as:

$$\Lambda(\boldsymbol{\theta}_t, \boldsymbol{\omega}_t, O_t) \leq C_1(\tau+1)\|\boldsymbol{\theta}_t - \boldsymbol{\theta}_{t-\tau}\| + C_2 m\rho^{\tau-1} + C_3\|\boldsymbol{\omega}_t - \boldsymbol{\omega}_{t-\tau}\|, \tag{5.5.6}$$

where $C_1 = 2U_\delta^2|\mathcal{A}|L(1+\lceil\log_\rho m^{-1}\rceil + 1/(1-\rho)) + 2U_\delta L_*, C_2 = 2U_\delta^2, C_3 = 4U_\delta$ are constants.

Step 3: integrating the results. By some calculation, terms I_1, I_2 and I_4 can be respectively bounded as follows (set $\tau = \tau_t$ defined in (5.4.1)). The detailed derivation can be found in Appendix 5.7.3,

$$\begin{aligned}
I_1 &= 4R_{\boldsymbol{\omega}}^2 \frac{1}{\beta_t} = O(t^\nu),\\
I_2 &\leq C_1 G_{\boldsymbol{\theta}}(\tau_t+1)^2 \sum_{k=0}^{t-\tau_t}\alpha_k + C_2(t-\tau_t+1)\alpha_t + C_3 U_\delta \tau_t \sum_{k=0}^{t-\tau_t}\beta_k\\
&= O\big((\log t)^2 t^{1-\sigma}\big) + O(t^{1-\sigma}) + O\big((\log t)t^{1-\nu}\big)\\
&= O\big((\log t)t^{1-\nu}\big),\\
I_5 &= \sum_{k=0}^{t-\tau_t}\beta_k = O(t^{1-\nu}).
\end{aligned}$$

The $\log t$ comes from $\tau_t = O(\log t)$. Performing the same technique on I_3 as in Step 3 in the proof sketch of Theorem 5.4.5, we have

$$\begin{aligned}
I_3 &\leq \bigg(\sum_{k=0}^{t-\tau_t}\frac{\alpha_k^2}{\beta_k^2}\bigg)^{\frac{1}{2}}\bigg(\sum_{k=\tau_t}^{t}\mathbb{E}\|\mathbf{z}_k\|^2\bigg)^{\frac{1}{2}},\\
I_4 &\leq \bigg(\sum_{k=\tau_t}^{t}\mathbb{E}[y_k^2]\bigg)^{\frac{1}{2}}\bigg(\sum_{k=\tau_t}^{t}\mathbb{E}\|\mathbf{z}_k\|^2\bigg)^{\frac{1}{2}}.
\end{aligned}$$

After plugging each term into (5.5.4), we have that

$$\begin{aligned}
2\lambda\sum_{k=\tau_t}^{t}\mathbb{E}\|\mathbf{z}_k\|^2 &\leq O(t^\nu) + O\big((\log t)t^{1-\nu}\big)\\
&\quad + 2L_* G_{\boldsymbol{\theta}}\bigg(\sum_{k=0}^{t-\tau_t}\frac{\alpha_k^2}{\beta_k^2}\bigg)^{\frac{1}{2}}\bigg(\sum_{k=\tau_t}^{t}\mathbb{E}\|\mathbf{z}_k\|^2\bigg)^{\frac{1}{2}} + \bigg(\sum_{k=0}^{t-\tau_t}\mathbb{E}[y_k^2]\bigg)^{\frac{1}{2}}\bigg(\sum_{k=\tau_t}^{t}\mathbb{E}\|\mathbf{z}_k\|^2\bigg)^{\frac{1}{2}}.
\end{aligned}$$

This inequality actually resembles (5.5.2). Following the same procedure as the proof of Theorem 5.4.5, starting from (5.5.2), we can finally get

$$\frac{1}{1+t-\tau_t}\sum_{k=\tau_t}^{t}\mathbb{E}\|\mathbf{z}_k\|^2 = O\bigg(\frac{1}{t^{1-\nu}}\bigg) + O\bigg(\frac{\log t}{t^\nu}\bigg) + O\bigg(\frac{1}{t^{2(\sigma-\nu)}}\bigg).$$

Note that this requires the step sizes γ_t and β_t should be of the same order $O(t^{-\nu})$.

5.6 Preliminary Lemmas

These useful lemmas are frequently applied throughout the proof.

5.6.1 Probabilistic Lemmas

The first two statements in the following lemma come from [ZXL19].

Lemma 5.6.1. *For any $\boldsymbol{\theta}_1$ and $\boldsymbol{\theta}_2$, it holds that*

$$\begin{aligned} d_{TV}(\mu_{\boldsymbol{\theta}_1}, \mu_{\boldsymbol{\theta}_2}) &\leq |\mathcal{A}|L\left(\lceil \log_\rho m^{-1} \rceil + \frac{1}{1-\rho}\right)\|\boldsymbol{\theta}_1 - \boldsymbol{\theta}_2\|, \\ d_{TV}(\mu_{\boldsymbol{\theta}_1} \otimes \pi_{\boldsymbol{\theta}_1}, \mu_{\boldsymbol{\theta}_2} \otimes \pi_{\boldsymbol{\theta}_2}) &\leq |\mathcal{A}|L\left(1 + \lceil \log_\rho m^{-1} \rceil + \frac{1}{1-\rho}\right)\|\boldsymbol{\theta}_1 - \boldsymbol{\theta}_2\|, \\ d_{TV}(\mu_{\boldsymbol{\theta}_1} \otimes \pi_{\boldsymbol{\theta}_1} \otimes \mathcal{P}, \mu_{\boldsymbol{\theta}_2} \otimes \pi_{\boldsymbol{\theta}_2} \otimes \mathcal{P}) &\leq |\mathcal{A}|L\left(1 + \lceil \log_\rho m^{-1} \rceil + \frac{1}{1-\rho}\right)\|\boldsymbol{\theta}_1 - \boldsymbol{\theta}_2\|. \end{aligned}$$

Proof. The proof of the first two inequality is exactly the same as Lemma A.3 in [ZXL19], which mainly depends on Theorem 3.1 in [Mit05]. Here we provide the proof of the third inequality. Note that

$$\begin{aligned} & d_{TV}(\mu_{\boldsymbol{\theta}_1} \otimes \pi_{\boldsymbol{\theta}_1} \otimes \mathcal{P}, \mu_{\boldsymbol{\theta}_2} \otimes \pi_{\boldsymbol{\theta}_2} \otimes \mathcal{P}) \\ &= \frac{1}{2}\int_{\mathcal{S}}\sum_{\mathcal{A}}\int_{\mathcal{S}} \left|\mu_{\boldsymbol{\theta}_1}(ds)\pi_{\boldsymbol{\theta}_1}(a|s)\mathcal{P}(ds'|s,a) - \mu_{\boldsymbol{\theta}_2}(ds)\pi_{\boldsymbol{\theta}_2}(a|s)\mathcal{P}(ds'|s,a)\right| \\ &= \frac{1}{2}\int_{\mathcal{S}}\sum_{\mathcal{A}}\int_{\mathcal{S}} \mathcal{P}(ds'|s,a)\left|\mu_{\boldsymbol{\theta}_1}(ds)\pi_{\boldsymbol{\theta}_1}(a|s) - \mu_{\boldsymbol{\theta}_2}(ds)\pi_{\boldsymbol{\theta}_2}(a|s)\right| \\ &= \frac{1}{2}\int_{\mathcal{S}}\sum_{\mathcal{A}} \left|\mu_{\boldsymbol{\theta}_1}(ds)\pi_{\boldsymbol{\theta}_1}(a|s) - \mu_{\boldsymbol{\theta}_2}(ds)\pi_{\boldsymbol{\theta}_2}(a|s)\right| \\ &= d_{TV}(\mu_{\boldsymbol{\theta}_1} \otimes \pi_{\boldsymbol{\theta}_1}, \mu_{\boldsymbol{\theta}_2} \otimes \pi_{\boldsymbol{\theta}_2}), \end{aligned} \tag{5.6.1}$$

so it has the same upper bound as the second inequality. □

Lemma 5.6.2. *Given time indexes t and τ such that $t \geq \tau > 0$, consider the auxiliary Markov chain starting from $s_{t-\tau}$. Conditioning on $s_{t-\tau+1}$ and $\boldsymbol{\theta}_{t-\tau}$, the Markov chain is obtained by repeatedly applying policy $\pi_{\boldsymbol{\theta}_{t-\tau}}$.*

$$s_{t-\tau} \xrightarrow{\boldsymbol{\theta}_{t-\tau}} a_{t-\tau} \xrightarrow{\mathcal{P}} s_{t-\tau+1} \xrightarrow{\boldsymbol{\theta}_{t-\tau}} \tilde{a}_{t-\tau+1} \xrightarrow{\mathcal{P}} \tilde{s}_{t-\tau+2} \xrightarrow{\boldsymbol{\theta}_{t-\tau}} \tilde{a}_{t-\tau+2} \xrightarrow{\mathcal{P}} \cdots \xrightarrow{\mathcal{P}} \tilde{s}_t \xrightarrow{\boldsymbol{\theta}_{t-\tau}} \tilde{a}_t \xrightarrow{\mathcal{P}} \tilde{s}_{t+1}.$$

For reference, recall that the original Markov chain is given as:

$$s_{t-\tau} \xrightarrow{\boldsymbol{\theta}_{t-\tau}} a_{t-\tau} \xrightarrow{\mathcal{P}} s_{t-\tau+1} \xrightarrow{\boldsymbol{\theta}_{t-\tau+1}} a_{t-\tau+1} \xrightarrow{\mathcal{P}} s_{t-\tau+2} \xrightarrow{\boldsymbol{\theta}_{t-\tau+2}} a_{t-\tau+2} \xrightarrow{\mathcal{P}} \cdots \xrightarrow{\mathcal{P}} s_t \xrightarrow{\boldsymbol{\theta}_t} a_t \xrightarrow{\mathcal{P}} s_{t+1}.$$

Throughout this lemma, we always condition the expectation on $s_{t-\tau+1}$ and $\boldsymbol{\theta}_{t-\tau}$ and omit this in order to simplify the presentation. Under the setting introduced above, we have:

$$d_{TV}\big(\mathbb{P}(s_{t+1} \in \cdot), \mathbb{P}(\tilde{s}_{t+1} \in \cdot)\big) \le d_{TV}\big(\mathbb{P}(O_t \in \cdot), \mathbb{P}(\tilde{O}_t \in \cdot)\big), \tag{5.6.2}$$

$$d_{TV}\big(\mathbb{P}(O_t \in \cdot), \mathbb{P}(\tilde{O}_t \in \cdot)\big) = d_{TV}\big(\mathbb{P}((s_t, a_t) \in \cdot), \mathbb{P}((\tilde{s}_t, \tilde{a}_t) \in \cdot)\big), \tag{5.6.3}$$

$$d_{TV}\big(\mathbb{P}((s_t, a_t) \in \cdot), \mathbb{P}((\tilde{s}_t, \tilde{a}_t) \in \cdot)\big) \le d_{TV}\big(\mathbb{P}(s_t \in \cdot), \mathbb{P}((\tilde{s}_t \in \cdot)\big) + \frac{1}{2}|\mathcal{A}|L\mathbb{E}\big[\|\boldsymbol{\theta}_t - \boldsymbol{\theta}_{t-\tau}\|\big]. \tag{5.6.4}$$

Proof of (5.6.2). By the Law of Total Probability,

$$\mathbb{P}(s_{t+1} \in \cdot) = \int_{\mathcal{S}} \sum_{\mathcal{A}} \mathbb{P}(s_t = ds, a_t = a, s_{t+1} \in \cdot),$$

and a similar argument also holds for $\tilde{O}_t$. Then we have

$$\begin{aligned}
&2d_{TV}\big(\mathbb{P}(s_{t+1} \in \cdot), \mathbb{P}(\tilde{s}_{t+1} \in \cdot)\big) \\
&= \int_{\mathcal{S}} \left| \int_{\mathcal{S}} \sum_{\mathcal{A}} \mathbb{P}(s_t = ds, a_t = a, s_{t+1} = ds') - \int_{\mathcal{S}} \sum_{\mathcal{A}} \mathbb{P}(s_t = ds, a_t = a, s_{t+1} = ds') \right| \\
&\le \int_{\mathcal{S}} \int_{\mathcal{S}} \sum_{\mathcal{A}} \big|\mathbb{P}(s_t = ds, a_t = a, s_{t+1} = ds') - \mathbb{P}(s_t = ds, a_t = a, s_{t+1} = ds')\big| \\
&= \int_{\mathcal{S}} \int_{\mathcal{S}} \sum_{\mathcal{A}} \big|\mathbb{P}(O_t = (ds, a, ds')) - \mathbb{P}(\tilde{O}_t = (ds, a, ds'))\big| \\
&= 2d_{TV}\big(\mathbb{P}(O_t \in \cdot), \mathbb{P}(\tilde{O}_t \in \cdot)\big).
\end{aligned}$$

The last equality requires exchange of integral, which should be guaranteed by the regularity.

□

Proof of (5.6.3).

$$\begin{aligned}
&2d_{TV}\big(\mathbb{P}(O_t \in \cdot), \mathbb{P}(\tilde{O}_t \in \cdot)\big) \\
&= \int_{\mathcal{S}} \sum_{\mathcal{A}} \int_{\mathcal{S}} \big|\mathbb{P}(O_t = (ds, a, ds')) - \mathbb{P}(\tilde{O}_t = (ds, a, ds'))\big|
\end{aligned}$$

$$
\begin{aligned}
&= \int_{\mathcal{S}} \sum_{\mathcal{A}} \int_{\mathcal{S}} \big|\mathcal{P}(ds'|s,a)\mathbb{P}((s_t,a_t)=(ds,a)) - \mathcal{P}(ds'|s,a)\mathbb{P}((\tilde{s}_t,\tilde{a}_t)=(ds,a))\big| \\
&= \int_{\mathcal{S}} \sum_{\mathcal{A}} \int_{\mathcal{S}} \mathcal{P}(ds'|s,a)\big|\mathbb{P}((s_t,a_t)=(ds,a)) - \mathbb{P}((\tilde{s}_t,\tilde{a}_t)=(ds,a))\big| \\
&= \int_{\mathcal{S}} \sum_{\mathcal{A}} \big|\mathbb{P}((s_t,a_t)=(ds,a)) - \mathbb{P}((\tilde{s}_t,\tilde{a}_t)=(ds,a))\big| \\
&= 2d_{TV}\big(\mathbb{P}((s_t,a_t)\in\cdot),\mathbb{P}((\tilde{s}_t,\tilde{a}_t)\in\cdot)\big).
\end{aligned}
$$

□

Proof of (5.6.4). Because $\boldsymbol{\theta}_t$ is also dependent on s_t, we make it clear here that

$$
\begin{aligned}
\mathbb{P}\big((s_t,a_t)=(ds,a)\big) &= \int_{\boldsymbol{\theta}\in\mathbb{R}^d} \mathbb{P}(s_t=ds)\mathbb{P}(\boldsymbol{\theta}_t=d\boldsymbol{\theta}|s_t=ds)\mathbb{P}(a_t=a|s_t=ds,\boldsymbol{\theta}_t=d\boldsymbol{\theta}) \\
&= \int_{\boldsymbol{\theta}\in\mathbb{R}^d} \mathbb{P}(s_t=ds)\mathbb{P}(\boldsymbol{\theta}_t=d\boldsymbol{\theta}|s_t=ds)\pi_{\boldsymbol{\theta}_t}(a|ds) \\
&= \mathbb{P}(s_t=ds)\int_{\boldsymbol{\theta}\in\mathbb{R}^d} \mathbb{P}(\boldsymbol{\theta}_t=d\boldsymbol{\theta}|s_t=ds)\pi_{\boldsymbol{\theta}_t}(a|ds) \\
&= \mathbb{P}(s_t=ds)\mathbb{E}\big[\pi_{\boldsymbol{\theta}_t}(a|ds)|s_t=ds\big].
\end{aligned}
$$

Therefore, the total variance can be bounded as

$$
\begin{aligned}
&2d_{TV}\big(\mathbb{P}((s_t,a_t)\in\cdot),\mathbb{P}((\tilde{s}_t,\tilde{a}_t)\in\cdot)\big) \\
&= \int_{\mathcal{S}} \sum_{\mathcal{A}} \big|\mathbb{P}(s_t=ds)\mathbb{E}[\pi_{\boldsymbol{\theta}_t}(a|ds)|s_t=ds] - \mathbb{P}(\tilde{s}_t=ds)\pi_{\boldsymbol{\theta}_{t-\tau}}(a|ds)\big| \\
&= \int_{\mathcal{S}} \sum_{\mathcal{A}} \big|\mathbb{P}(s_t=ds)\mathbb{E}[\pi_{\boldsymbol{\theta}_t}(a|ds)|s_t=ds] - \mathbb{P}(s_t=ds)\pi_{\boldsymbol{\theta}_{t-\tau}}(a|ds)\big| \\
&\quad + \int_{\mathcal{S}} \sum_{\mathcal{A}} \big|\mathbb{P}(s_t=ds)\pi_{\boldsymbol{\theta}_{t-\tau}}(a|ds) - \mathbb{P}(\tilde{s}_t=ds)\pi_{\boldsymbol{\theta}_{t-\tau}}(a|ds)\big| \\
&= \int_{\mathcal{S}} \mathbb{P}(s_t=ds)\sum_{\mathcal{A}} \big|\mathbb{E}[\pi_{\boldsymbol{\theta}_t}(a|ds)|s_t=ds] - \pi_{\boldsymbol{\theta}_{t-\tau}}(a|ds)\big| \\
&\quad + 2d_{TV}\big(\mathbb{P}(s_t\in\cdot),\mathbb{P}((\tilde{s}_t\in\cdot)\big) \\
&\le |\mathcal{A}|L\mathbb{E}\big[\|\boldsymbol{\theta}_t-\boldsymbol{\theta}_{t-\tau}\|\big] + 2d_{TV}\big(\mathbb{P}(s_t\in\cdot),\mathbb{P}((\tilde{s}_t\in\cdot)\big),
\end{aligned}
$$

where the inequality holds due to the Lipschitz continuity of the policy as in Assumption 5.4.3. □

5.6.2 Lipschitzness of the Optimal Parameter

This section is used to present the proof of Proposition 5.4.4.

Proof of Proposition 5.4.4. [SB18] has proved in Chapter 9 the fact that the linear TD(0) will converge to the optimal point (w.r.t. Mean Square Projected Bellman Error) which satisfies $\mathbf{A}_i\boldsymbol{\omega}^*(\boldsymbol{\theta}_i) = \mathbf{b}_i$, where $\mathbf{A}_i := \mathbb{E}[\boldsymbol{\phi}(s)(\boldsymbol{\phi}(s) - \boldsymbol{\phi}(s'))^\top]$ and $\mathbf{b}_i := \mathbb{E}[(r(s,a) - r(\boldsymbol{\theta}_i))\boldsymbol{\phi}(s)]$. The expectation is taken over the stationary distribution $s \sim \mu_{\boldsymbol{\theta}_i}$, the action $a \sim \pi_{\boldsymbol{\theta}_i}(\cdot|s)$ and the transition probability matrix $s' \sim \mathcal{P}(\cdot|s,a)$. Now we denote $\boldsymbol{\omega}_1^*, \boldsymbol{\omega}_2^*, \hat{\boldsymbol{\omega}}_1$ as the unique solutions of the following equations respectively:

$$\mathbf{A}_1\boldsymbol{\omega}_1^* = \mathbf{b}_1, \quad \mathbf{A}_2\hat{\boldsymbol{\omega}}_1 = \mathbf{b}_1, \quad \mathbf{A}_2\boldsymbol{\omega}_2^* = \mathbf{b}_2.$$

First we bound $\|\boldsymbol{\omega}_1^* - \hat{\boldsymbol{\omega}}_1\|$. By definition, we have

$$\|\boldsymbol{\omega}_1^* - \hat{\boldsymbol{\omega}}_1\| \le \|\mathbf{A}_1^{-1} - \mathbf{A}_2^{-1}\|\|\mathbf{b}_1\|.$$

It can be easily shown that

$$\mathbf{A}_1^{-1} - \mathbf{A}_2^{-1} = \mathbf{A}_1^{-1}(\mathbf{A}_2 - \mathbf{A}_1)\mathbf{A}_2^{-1},$$

which further gives

$$\|\boldsymbol{\omega}_1^* - \hat{\boldsymbol{\omega}}_1\| \le \|\mathbf{A}_1^{-1}\|\|\mathbf{A}_1 - \mathbf{A}_2\|\|\mathbf{A}_2^{-1}\|\|\mathbf{b}_1\|.$$

Then we bound $\|\hat{\boldsymbol{\omega}}_1 - \boldsymbol{\omega}_2^*\|$ as $\|\hat{\boldsymbol{\omega}}_1 - \boldsymbol{\omega}_2^*\| \le \|\mathbf{A}_2^{-1}\|\|\mathbf{b}_1 - \mathbf{b}_2\|$. By Assumption 5.4.1, the eigenvalues of $\mathbf{A}_i$ are bounded from below by $\lambda > 0$, therefore $\|\mathbf{A}_i^{-1}\| \le \lambda^{-1}$. Also $\|\mathbf{b}_1\| \le U_r$ due to the assumption that $|r(s,a)| \le U_r$ and $\|\boldsymbol{\phi}(s)\| \le 1$. To bound $\|\mathbf{A}_1 - \mathbf{A}_2\|$ and $\|\mathbf{b}_1 - \mathbf{b}_2\|$, we first note that

$$\begin{aligned}
\|\mathbf{A}_1 - \mathbf{A}_2\|_2 &\le \sup_{s,s'\in\mathcal{S}} \big\|\boldsymbol{\phi}(s)(\boldsymbol{\phi}(s) - \boldsymbol{\phi}(s'))^\top\big\|_2 \cdot 2d_{TV}\big(\mathbb{P}(O^1 \in \cdot), \mathbb{P}(O^2 \in \cdot)\big), \\
&\le 4d_{TV}\big(\mathbb{P}(O^1 \in \cdot), \mathbb{P}(O^2 \in \cdot)\big) \\
\|\mathbf{b}_1 - \mathbf{b}_2\| &\le \big\|\mathbb{E}[r(s^1,a^1)\boldsymbol{\phi}(s^1)] - \mathbb{E}[r(s^2,a^2)\boldsymbol{\phi}(s^2)]\big\| + \big\|r(\boldsymbol{\theta}_1)\mathbb{E}[\boldsymbol{\phi}(s^1)] - r(\boldsymbol{\theta}_2)\mathbb{E}[\boldsymbol{\phi}(s^2)]\big\| \\
&\le 6U_r d_{TV}\big(\mathbb{P}(O^1 \in \cdot), \mathbb{P}(O^2 \in \cdot)\big),
\end{aligned}$$

where O^i is the tuple obtained by $s^i \sim \mu_{\boldsymbol{\theta}_i}(\cdot)$, $a^i \sim \pi_{\boldsymbol{\theta}_i}(\cdot|s^i)$ and $(s')^i \sim \mathcal{P}(\cdot|s^i, a^i)$. And the total variation norm can be bounded by Lemma 5.6.1 as:

$$d_{TV}\big(\mathbb{P}(O^1 \in \cdot), \mathbb{P}(O^2 \in \cdot)\big) \leq |\mathcal{A}|L\bigg(1 + \lceil \log_\rho m^{-1} \rceil + \frac{1}{1-\rho}\bigg)\|\boldsymbol{\theta}_1 - \boldsymbol{\theta}_2\|.$$

Collecting the results above gives

$$\begin{aligned}\|\boldsymbol{\omega}_1^* - \boldsymbol{\omega}_2^*\| &\leq \|\boldsymbol{\omega}_1^* - \hat{\boldsymbol{\omega}}_1\| + \|\hat{\boldsymbol{\omega}}_1 - \boldsymbol{\omega}_2^*\| \\ &\leq (2\lambda^{-2}U_r + 3\lambda^{-1}U_r)|\mathcal{A}|L\bigg(1 + \lceil \log_\rho m^{-1} \rceil + \frac{1}{1-\rho}\bigg)\|\boldsymbol{\theta}_1 - \boldsymbol{\theta}_2\|,\end{aligned}$$

and we set $L_* := (2\lambda^{-2}U_r + 3\lambda^{-1}U_r)|\mathcal{A}|L(1 + \lceil \log_\rho m^{-1} \rceil + 1/(1-\rho))$ to obtain the final result. □

5.6.3 Asymptotic Equivalence

Lemma 5.6.3. *Suppose $\{a_i\}$ is a non-negative, bounded sequence, $\tau := C_1 + C_2 \log t (C_2 > 0)$, then for any large enough t such that $t \geq \tau > 0$, we have:*

$$\begin{aligned}\frac{1}{1+t-\tau}\sum_{k=\tau}^{t} a_i &= O\bigg(\frac{1}{t}\sum_{k=1}^{t} a_i\bigg), \\ \frac{1}{t}\sum_{k=1}^{t} a_i &= O\bigg(\frac{\log t}{t}\bigg) + O\bigg(\frac{1}{1+t-\tau}\sum_{k=\tau}^{t} a_i\bigg).\end{aligned}$$

Proof. We know that $\tau = O(\log t)$ and the sequence is bounded: $0 < a_i < B$. For the first equation, we have

$$\frac{1}{1+t-\tau}\sum_{k=\tau}^{t} a_i \leq \frac{1}{1+t-\tau}\sum_{k=1}^{t} a_i \leq \frac{t}{1+t-\tau} \cdot \frac{1}{t}\sum_{k=1}^{t} a_i \leq O\bigg(\frac{1}{t}\sum_{k=1}^{t} a_i\bigg),$$

and further assuming $t \geq 2\tau - 2$ gives a constant 2. For the second equation, we have

$$\frac{1}{t}\sum_{k=1}^{t} a_i \leq \frac{1}{t}\bigg((\tau-1)B + \sum_{k=\tau}^{t} a_i\bigg) = \frac{\tau-1}{t}B + \frac{1}{t}\sum_{k=\tau}^{t} a_i = O\bigg(\frac{\log t}{t}\bigg) + O\bigg(\frac{1}{1+t-\tau}\sum_{k=\tau}^{t} a_i\bigg).$$

□

5.7 Proof of Main Theorems and Propositions

5.7.1 Proof of Theorem 5.4.5

We first define several notations to clarify the dependence:

$$
\begin{aligned}
O_t &:= (s_t, a_t, s_{t+1}),\\
\eta^* &:= \eta(\boldsymbol{\theta}) = \mathbb{E}_{s\sim\mu_{\boldsymbol{\theta}}, a\sim\pi_{\boldsymbol{\theta}}(\cdot|s)}[r(s,a)]\\
\Delta h(O, \eta, \boldsymbol{\omega}, \boldsymbol{\theta}) &:= \Big(\eta(\boldsymbol{\theta}) - \eta + \big(\boldsymbol{\phi}(s') - \boldsymbol{\phi}(s)\big)^\top (\boldsymbol{\omega} - \boldsymbol{\omega}^*)\Big)\nabla \log \pi_{\boldsymbol{\theta}}(a|s),\\
\Delta h'(O, \boldsymbol{\theta}) &:= \Big(\big(\boldsymbol{\phi}(s')^\top \boldsymbol{\omega}^* - V^{\pi_{\boldsymbol{\theta}}}(s')\big) - \big(\boldsymbol{\phi}(s)^\top \boldsymbol{\omega}^* - V^{\pi_{\boldsymbol{\theta}}}(s)\big)\Big)\nabla \log \pi_{\boldsymbol{\theta}}(a|s),\\
h(O, \boldsymbol{\theta}) &:= \big(r(s,a) - \eta(\boldsymbol{\theta}) + \boldsymbol{\phi}(s')^\top \boldsymbol{\omega}^* - \boldsymbol{\phi}(s)^\top \boldsymbol{\omega}^*\big)\nabla \log \pi_{\boldsymbol{\theta}}(a|s),\\
\Gamma(O, \boldsymbol{\theta}) &:= \big\langle \nabla J(\boldsymbol{\theta}), h(O, \boldsymbol{\theta}) - \mathbb{E}_{s\sim\mu_{\boldsymbol{\theta}}, a\sim\pi_{\boldsymbol{\theta}}, s'\sim\mathbb{P}}\big[h(O', \boldsymbol{\theta})\big]\big\rangle. \qquad (5.7.1)
\end{aligned}
$$

Note that Δh, $\Delta h'$ and $h - \Delta h'$ together gives a decomposition of the actual gradient we use in Algorithm 9. They each correspond to the error caused by the critic $\boldsymbol{\omega}_t$, the approximation error of the linear class, and the stochastic policy gradient.

$\Gamma(O, \boldsymbol{\theta})$ is the Markovian noise for $h(O, \boldsymbol{\theta})$. Here $O' = (s, a, s')$ is a shorthand for an independent sample from $s \sim \mu_{\boldsymbol{\theta}}, a \sim \pi_{\boldsymbol{\theta}}, s' \sim \mathbb{P}$. Using a more compact notation $\mathbb{E}_{O'}[\cdot]$, it is clear we have $\mathbb{E}_{O'}[h(O', \boldsymbol{\theta}) - \Delta h'(O', \boldsymbol{\theta})] = \nabla J(\boldsymbol{\theta})$ and $\mathbb{E}_{O'}\|\Delta h'(O, \boldsymbol{\theta})\|^2 \le 4B^2\epsilon_{\text{app}}^2$ because

$$
\begin{aligned}
\mathbb{E}_{O'}\|\Delta h'(O, \boldsymbol{\theta})\|^2 &= \mathbb{E}_{O'}\Big\|\Big(\big(\boldsymbol{\phi}(s')^\top \boldsymbol{\omega}^* - V^{\pi_{\boldsymbol{\theta}}}(s')\big) - \big(\boldsymbol{\phi}(s)^\top \boldsymbol{\omega}^* - V^{\pi_{\boldsymbol{\theta}}}(s)\big)\Big)\nabla \log \pi_{\boldsymbol{\theta}}(a|s)\Big\|^2\\
&\le \mathbb{E}_{O'}\Big[B^2\Big(\big(\boldsymbol{\phi}(s')^\top \boldsymbol{\omega}^* - V^{\pi_{\boldsymbol{\theta}}}(s')\big) - \big(\boldsymbol{\phi}(s)^\top \boldsymbol{\omega}^* - V^{\pi_{\boldsymbol{\theta}}}(s)\big)\Big)^2\Big]\\
&\le \mathbb{E}_{O'}\Big[2B^2\big(\boldsymbol{\phi}(s')^\top \boldsymbol{\omega}^* - V^{\pi_{\boldsymbol{\theta}}}(s')\big)^2 + \big(\boldsymbol{\phi}(s)^\top \boldsymbol{\omega}^* - V^{\pi_{\boldsymbol{\theta}}}(s)\big)^2\Big]\\
&= 4B^2\mathbb{E}_{O'}\Big[\big(\boldsymbol{\phi}(s)^\top \boldsymbol{\omega}^*(\boldsymbol{\theta}) - V^{\pi_{\boldsymbol{\theta}}}(s)\big)^2\Big]\\
&= 4B^2\epsilon_{\text{app}}^2.
\end{aligned}
$$

There are several lemmas that will be used in the proof.

Lemma 5.7.1. *For the performance function defined in* (5.3.1)*, there exists a constant* $L_J > 0$ *such that for all* $\boldsymbol{\theta}_1, \boldsymbol{\theta}_2 \in \mathbb{R}^d$*, it holds that*

$$\big\|\nabla J(\boldsymbol{\theta}_1) - \nabla J(\boldsymbol{\theta}_2)\big\| \le L_J\|\boldsymbol{\theta}_1 - \boldsymbol{\theta}_2\|,$$

which by the definition of smoothness [Nes18] implies

$$J(\boldsymbol{\theta}_2) \geq J(\boldsymbol{\theta}_1) + \big\langle \nabla J(\boldsymbol{\theta}_1), \boldsymbol{\theta}_2 - \boldsymbol{\theta}_1 \big\rangle - \frac{L_J}{2}\|\boldsymbol{\theta}_1 - \boldsymbol{\theta}_2\|^2.$$

The following two lemmas characterize the bias introduced by the critic's approximation and the Markovian noise.

Lemma 5.7.2. *For any $t \geq 0$,*

$$\big\|\Delta h(O_t, \eta_t, \boldsymbol{\omega}_t, \boldsymbol{\theta}_t)\big\|^2 \leq B^2\big(8\|\boldsymbol{\omega}_t - \boldsymbol{\omega}_t^*\|^2 + 2(\eta_t - \eta_t^*)^2\big).$$

Lemma 5.7.3. *For any $\boldsymbol{\theta} \in \mathbb{R}^d$, we have $\|\delta \nabla \log \pi_{\boldsymbol{\theta}}(a|s)\| \leq G_{\boldsymbol{\theta}} := U_\delta \cdot B$, where $U_\delta = 2U_r + 2R_{\boldsymbol{\omega}}$. Furthermore, for any $t \geq 0$, it holds that*

$$\mathbb{E}\big[\Gamma(O_t, \boldsymbol{\theta}_t)\big] \geq -G_{\boldsymbol{\theta}}\Big(D_1(\tau+1)\sum_{k=t-\tau+1}^{t}\mathbb{E}\|\boldsymbol{\theta}_k - \boldsymbol{\theta}_{k-1}\| + D_2 m\rho^{\tau-1}\Big),$$

where $D_1 = \max\{(U_\delta L_l + 2L_* B + 3L_J), 2U_\delta B|\mathcal{A}|L\}$ and $D_2 = 4U_\delta B$.

Proof of Theorem 5.4.5. Under the update rule of Algorithm 9, we have

$$\begin{aligned}
J(\boldsymbol{\theta}_{t+1}) &\geq J(\boldsymbol{\theta}_t) + \alpha_t\big\langle \nabla J(\boldsymbol{\theta}_t), \delta_t \nabla \log \pi_{\boldsymbol{\theta}_t}(a_t|s_t)\big\rangle - L_J\alpha_t^2\big\|\delta_t \nabla \log \pi_{\boldsymbol{\theta}_t}(a_t|s_t)\big\|^2 \\
&= J(\boldsymbol{\theta}_t) + \alpha_t\big\langle \nabla J(\boldsymbol{\theta}_t), \Delta h(O_t, \eta_t, \boldsymbol{\omega}_t, \boldsymbol{\theta}_t)\big\rangle \\
&\quad + \alpha_t\big\langle \nabla J(\boldsymbol{\theta}_t), h(O_t, \boldsymbol{\theta}_t)\big\rangle - L_J\alpha_t^2\big\|\delta_t \nabla \log \pi_{\boldsymbol{\theta}_t}(a_t|s_t)\big\|^2 \\
&= J(\boldsymbol{\theta}_t) + \alpha_t\big\langle \nabla J(\boldsymbol{\theta}_t), \Delta h(O_t, \eta_t, \boldsymbol{\omega}_t, \boldsymbol{\theta}_t)\big\rangle \\
&\quad + \alpha_t\big\langle \nabla J(\boldsymbol{\theta}_t), \mathbb{E}_{O'}[h(O', \boldsymbol{\theta}_t)]\big\rangle + \alpha_t\Gamma(O_t, \boldsymbol{\theta}_t) - L_J\alpha_t^2\big\|\delta_t \nabla \log \pi_{\boldsymbol{\theta}_t}(a_t|s_t)\big\|^2 \\
&= J(\boldsymbol{\theta}_t) + \alpha_t\big\langle \nabla J(\boldsymbol{\theta}_t), \Delta h(O_t, \eta_t, \boldsymbol{\omega}_t, \boldsymbol{\theta}_t)\big\rangle + \alpha_t\big\langle \nabla J(\boldsymbol{\theta}_t), \mathbb{E}_{O'}[\Delta h'(O', \boldsymbol{\theta}_t)]\big\rangle \\
&\quad + \alpha_t\big\|\nabla J(\boldsymbol{\theta}_t)\big\|^2 + \alpha_t\Gamma(O_t, \boldsymbol{\theta}_t) - L_J\alpha_t^2\big\|\delta_t \nabla \log \pi_{\boldsymbol{\theta}_t}(a_t|s_t)\big\|^2.
\end{aligned} \tag{5.7.2}$$

The first inequality is by Lemma 5.7.1. The first equality is by the definitions in (5.7.1); the second equality is by the definition of $\Lambda(O_t, \boldsymbol{\theta}_t)$ in (5.7.1). The last inequality is due to the remarks under (5.7.1).

We will bound the expectation of each term on the right hand side of (5.7.2) as follows. First, we have

$$\mathbb{E}\big\langle \nabla J(\boldsymbol{\theta}_t), \Delta h(O_t, \eta_t, \boldsymbol{\omega}_t, \boldsymbol{\theta}_t)\big\rangle \geq -B\sqrt{\mathbb{E}\big\|\nabla J(\boldsymbol{\theta}_t)\big\|^2}\sqrt{8\mathbb{E}\|\mathbf{z}_t\|^2 + 2\mathbb{E}[y_t^2]},$$

where $\mathbf{z}_t := \boldsymbol{\omega}_t - \boldsymbol{\omega}_t^*$ and $y_t := \eta_t - \eta_t^*$, and the inequality is due to Cauchy inequality and Lemma 5.7.2.

Second, we have

$$\begin{aligned}\mathbb{E}[\Gamma(O_t, \boldsymbol{\theta}_t)] &\geq -G_{\boldsymbol{\theta}}\bigg(D_1(\tau+1)\sum_{k=t-\tau+1}^{t}\mathbb{E}\|\boldsymbol{\theta}_k - \boldsymbol{\theta}_{k-1}\| + D_2 m\rho^{\tau-1}\bigg),\\ &\geq -G_{\boldsymbol{\theta}}\bigg(D_1(\tau+1)G_{\boldsymbol{\theta}}\sum_{k=t-\tau+1}^{t-1}\alpha_k + D_2 m\rho^{\tau-1}\bigg),\end{aligned}$$

where the first inequality is due to Lemma 5.7.3, and the second inequality is due to $\big\|\delta_t \nabla \log \pi_{\boldsymbol{\theta}_t}(a_t|s_t)\big\| \leq G_{\boldsymbol{\theta}}$ by Lemma 5.7.3.

Third, by the remarks under (5.7.1) regarding $\Delta h'$, we have

$$\begin{aligned}\big\langle \nabla J(\boldsymbol{\theta}_t), \mathbb{E}_{O'}[\Delta h'(O_t, \boldsymbol{\theta}_t)]\big\rangle &\geq -G_{\boldsymbol{\theta}}\sqrt{\big\|\mathbb{E}_{O'}[\Delta h'(O_t, \boldsymbol{\theta}_t)]\big\|^2}\\ &\geq -G_{\boldsymbol{\theta}}\sqrt{\mathbb{E}_{O'}\big\|\Delta h'(O_t, \boldsymbol{\theta}_t)\big\|^2}\\ &\geq -2BG_{\boldsymbol{\theta}}\epsilon_{\text{app}},\end{aligned}$$

Taking the expectation of (5.7.2) and plugging the above terms back into it gives

$$\begin{aligned}\mathbb{E}[J(\boldsymbol{\theta}_{t+1})] \geq \mathbb{E}[J(\boldsymbol{\theta}_t)] &- \alpha_t B\sqrt{\mathbb{E}\big\|\nabla J(\boldsymbol{\theta}_t)\big\|^2}\sqrt{8\mathbb{E}\|\mathbf{z}_t\|^2 + 2\mathbb{E}[y_t^2]} - 2BG_{\boldsymbol{\theta}}\epsilon_{\text{app}}\alpha_t\\ &- \alpha_t G_{\boldsymbol{\theta}}\bigg(D_1(\tau+1)G_{\boldsymbol{\theta}}\sum_{k=t-\tau}^{t-1}\alpha_k + D_2 m\rho^{\tau-1}\bigg) + \alpha_t\mathbb{E}\|\nabla J(\boldsymbol{\theta}_t)\|^2 - L_J G_{\boldsymbol{\theta}}^2\alpha_t^2.\end{aligned}$$

Rearranging the above inequality gives

$$\begin{aligned}\mathbb{E}\big\|\nabla J(\boldsymbol{\theta}_t)\big\|^2 \leq \frac{1}{\alpha_t}\big(\mathbb{E}[J(\boldsymbol{\theta}_{t+1})] - \mathbb{E}[J(\boldsymbol{\theta}_t)]\big) &+ B\sqrt{\mathbb{E}\big\|\nabla J(\boldsymbol{\theta}_t)\big\|^2}\sqrt{8\mathbb{E}\|\mathbf{z}_t\|^2 + 2\mathbb{E}[y_t^2]}\\ &+ D_1 G_{\boldsymbol{\theta}}^2(\tau+1)\sum_{k=t-\tau}^{t-1}\alpha_k + D_2 G_{\boldsymbol{\theta}} m\rho^{\tau-1} + L_J G_{\boldsymbol{\theta}}^2\alpha_t.\end{aligned}$$

By setting $\tau = \tau_t$, we get

$$\begin{aligned}\mathbb{E}\big\|\nabla J(\boldsymbol{\theta}_t)\big\|^2 \leq \frac{1}{\alpha_t}\big(\mathbb{E}\big[J(\boldsymbol{\theta}_{t+1})\big] - \mathbb{E}\big[J(\boldsymbol{\theta}_t)\big]\big) &+ B\sqrt{\mathbb{E}\big\|\nabla J(\boldsymbol{\theta}_t)\big\|^2}\sqrt{8\mathbb{E}\|\mathbf{z}_t\|^2 + 2\mathbb{E}[y_t^2]}\\ &+ 2BG_{\boldsymbol{\theta}}\epsilon_{\text{app}} + D_1 G_{\boldsymbol{\theta}}^2(\tau_t+1)^2\alpha_{t-\tau_t} + D_2 G_{\boldsymbol{\theta}}\alpha_t + L_J G_{\boldsymbol{\theta}}^2\alpha_t.\end{aligned}$$

Summing over k from τ_t to t gives

$$
\begin{aligned}
&\sum_{k=\tau_t}^{t} \mathbb{E}\big\|\nabla J(\boldsymbol{\theta}_t)\big\|^2 \\
&\le \underbrace{\sum_{k=\tau_t}^{t} \frac{1}{\alpha_k}\big(\mathbb{E}[J(\boldsymbol{\theta}_{k+1})] - \mathbb{E}[J(\boldsymbol{\theta}_k)]\big)}_{I_1} + B \sum_{k=\tau_t}^{t} \sqrt{\mathbb{E}\big\|\nabla J(\boldsymbol{\theta}_t)\big\|^2}\sqrt{8\mathbb{E}\|\mathbf{z}_t\|^2 + 2\mathbb{E}[y_t^2]} \\
&\quad + \underbrace{\sum_{k=\tau_t}^{t} D_1 G_{\boldsymbol{\theta}}^2(\tau_t+1)^2\alpha_{k-\tau_t} + \sum_{k=\tau_t}^{t}(D_2 G_{\boldsymbol{\theta}} + L_J G_{\boldsymbol{\theta}}^2)\alpha_k}_{I_2} + 2BG_{\boldsymbol{\theta}}\epsilon_{\text{app}}(t-\tau_t+1).
\end{aligned}
$$

For the term I_1, we have,

$$
\begin{aligned}
\sum_{k=\tau_t}^{t} \frac{1}{\alpha_k}\big(J(\boldsymbol{\theta}_{k+1}) - J(\boldsymbol{\theta}_k)\big) &= \sum_{k=\tau_t}^{t}\left(\frac{1}{\alpha_{k-1}} - \frac{1}{\alpha_k}\right)\mathbb{E}[J(\boldsymbol{\theta}_k)] - \frac{1}{\alpha_{\tau_t-1}}\mathbb{E}[J(\boldsymbol{\theta}_{\tau_t})] + \frac{1}{\alpha_t}\mathbb{E}[J(\boldsymbol{\theta}_{t+1})] \\
&\le \sum_{k=\tau_t}^{t}\left(\frac{1}{\alpha_k} - \frac{1}{\alpha_{k-1}}\right)U_r + \frac{1}{\alpha_{\tau_t-1}}U_r + \frac{1}{\alpha_t}U_r \\
&= U_r\left[\sum_{k=\tau_t}^{t}\left(\frac{1}{\alpha_k} - \frac{1}{\alpha_{k-1}}\right) + \frac{1}{\alpha_{\tau_t-1}} + \frac{1}{\alpha_t}\right] \\
&= 2U_r\alpha_t^{-1},
\end{aligned}
$$

where the inequality holds due to $|\mathbb{E}[J(\boldsymbol{\theta})]| \le U_r$.

For the term I_2, we have

$$
\begin{aligned}
\sum_{k=\tau_t}^{t} D_1 G_{\boldsymbol{\theta}}^2(\tau_t+1)^2\alpha_{k-\tau_t} &= D_1 G_{\boldsymbol{\theta}}^2(\tau_t+1)^2 \sum_{k=\tau_t}^{t}\alpha_{k-\tau_t} \\
&= D_1 G_{\boldsymbol{\theta}}^2(\tau_t+1)^2 \sum_{k=0}^{t-\tau_t}\alpha_k \\
&= D_1 G_{\boldsymbol{\theta}}^2(\tau_t+1)^2 c_\alpha \sum_{k=0}^{t-\tau_t}\frac{1}{(1+k)^\sigma},
\end{aligned}
$$

and

$$
\begin{aligned}
\sum_{k=\tau_t}^{t}(D_2 G_{\boldsymbol{\theta}} + L_J G_{\boldsymbol{\theta}}^2)\alpha_k &= (D_2 G_{\boldsymbol{\theta}} + L_J G_{\boldsymbol{\theta}}^2)\sum_{k=\tau_t}^{t}\alpha_k \\
&\le (D_2 G_{\boldsymbol{\theta}} + L_J G_{\boldsymbol{\theta}}^2)\sum_{k=0}^{t-\tau_t}\alpha_k
\end{aligned}
$$

$$= (D_2 G_{\boldsymbol{\theta}} + L_J G_{\boldsymbol{\theta}}^2) c_\alpha \sum_{k=0}^{t-\tau_t} \frac{1}{(1+k)^\sigma}.$$

Note that both upper bounds rely on the summation $\sum_{k=0}^{t-\tau_t} 1/(1+k)^\sigma \leq \int_0^{t-\tau_t+1} x^{-\sigma} dx = 1/(1-\sigma)(t-\tau_t+1)^{1-\sigma}$. Combining the results for terms I_1 and I_2, we have

$$\begin{aligned}\sum_{k=\tau_t}^{t} \mathbb{E}\big\|\nabla J(\boldsymbol{\theta}_t)\big\|^2 \leq \frac{2U_r}{c_\alpha}(1+t)^\sigma \\ &+ \big(D_1 G_{\boldsymbol{\theta}}^2 (\tau_t+1)^2 + D_2 G_{\boldsymbol{\theta}} + L_J G_{\boldsymbol{\theta}}^2\big) \frac{c_\alpha}{1-\sigma}(t-\tau_t+1)^{1-\sigma} \\ &+ B \sum_{k=\tau_t}^{t} \sqrt{\mathbb{E}\big\|\nabla J(\boldsymbol{\theta}_t)\big\|^2} \sqrt{8\mathbb{E}\|\mathbf{z}_t\|^2 + 2\mathbb{E}[y_t^2]} \\ &+ 2BG_{\boldsymbol{\theta}}\epsilon_{\text{app}}(t-\tau_t+1).\end{aligned}$$

Dividing $(1+t-\tau_t)$ at both sides and assuming $t > 2\tau_t - 1$, we can express the result as

$$\begin{aligned}\frac{1}{1+t-\tau_t} \sum_{k=\tau_t}^{t} \mathbb{E}\big\|\nabla J(\boldsymbol{\theta}_t)\big\|^2 &\leq \big(D_1 G_{\boldsymbol{\theta}}^2 (\tau_t+1)^2 + D_2 G_{\boldsymbol{\theta}} + L_J G_{\boldsymbol{\theta}}^2\big) \frac{c_\alpha}{1-\sigma} \frac{1}{(t-\tau_t+1)^\sigma} \\ &+ \frac{2B}{1+t-\tau_t} \sum_{k=\tau_t}^{t} \sqrt{\mathbb{E}\big\|\nabla J(\boldsymbol{\theta}_t)\big\|^2} \sqrt{8\mathbb{E}\|\mathbf{z}_t\|^2 + 2\mathbb{E}[y_t^2]} \\ &+ \frac{4U_r}{c_\alpha} \frac{1}{(t+1)^{1-\sigma}} + 2BG_{\boldsymbol{\theta}}\epsilon_{\text{app}}.\end{aligned} \tag{5.7.3}$$

By Cauchy-Schwartz inequality, we have

$$\begin{aligned}&\frac{1}{1+t-\tau_t} \sum_{k=\tau_t}^{t} \sqrt{\mathbb{E}\big\|\nabla J(\boldsymbol{\theta}_t)\big\|^2} \sqrt{8\mathbb{E}\|\mathbf{z}_t\|^2 + 2\mathbb{E}[y_t^2]} \\ &\quad \leq \bigg(\frac{1}{1+t-\tau_t} \sum_{k=\tau_t}^{t} \mathbb{E}\big\|\nabla J(\boldsymbol{\theta}_t)\big\|^2\bigg)^{\frac{1}{2}} \bigg(\frac{1}{1+t-\tau_t} \sum_{k=\tau_t}^{t} \big(8\mathbb{E}\|\mathbf{z}_t\|^2 + 2\mathbb{E}[y_t^2]\big)\bigg)^{\frac{1}{2}}.\end{aligned}$$

Denote $F(t) = (1+t-\tau_t)^{-1} \sum_{k=\tau_t}^{t} \mathbb{E}\|\nabla J(\boldsymbol{\theta}_k)\|^2$ and $Z(t) = (1+t-\tau_t)^{-1} \sum_{k=\tau_t}^{t} \big(8\mathbb{E}\|\mathbf{z}_t\|^2 + 2\mathbb{E}[y_t^2]\big)$, and putting them back to (5.7.3) (O-notation for simplicity):

$$F(t) \leq O\bigg(\frac{1}{t^{1-\sigma}}\bigg) + O\bigg(\frac{(\log t)^2}{t^\sigma}\bigg) + O(\epsilon_{\text{app}}) + 2B\sqrt{F(t)} \cdot \sqrt{Z(t)},$$

which further gives

$$\big(\sqrt{F(t)} - B\sqrt{Z(t)}\big)^2 \leq O\bigg(\frac{1}{t^{1-\sigma}}\bigg) + O\bigg(\frac{(\log t)^2}{t^\sigma}\bigg) + O(\epsilon_{\text{app}}) + B^2 Z(t). \tag{5.7.4}$$

Note that for a general function $H(t) \leq A(t) + B(t)$(with each positive), we have

$$H^2(t) \leq 2A^2(t) + 2B^2(t),$$
$$\sqrt{H(t)} \leq \sqrt{A(t)} + \sqrt{B(t)}.$$

This means (5.7.4) implies

$$\begin{aligned}\sqrt{F(t)} - B\sqrt{Z(t)} &\leq \sqrt{A(t)} + B\sqrt{Z(t)},\\ \sqrt{F(t)} &\leq \sqrt{A(t)} + 2B\sqrt{Z(t)},\\ F(t) &\leq 2A(t) + 8B^2 Z(t).\end{aligned}$$

By Lemma 5.6.3, assuming $t \geq 2\tau_t - 1$, it holds that

$$Z(t) = \frac{1}{1+t-\tau_t}\sum_{k=\tau_t}^{t} 8\mathbb{E}\|\mathbf{z}_k\|^2 + 2\mathbb{E}[y_t^2] \leq \frac{2}{t}\sum_{k=1}^{t} 8\mathbb{E}\|\mathbf{z}_k\|^2 + 2\mathbb{E}[y_t^2] = 2\mathcal{E}(t).$$

And finally, we have

$$\begin{aligned}\min_{0\leq k\leq t} \mathbb{E}\big\|\nabla J(\boldsymbol{\theta}_k)\big\|^2 &\leq \frac{1}{1+t-\tau_t}\sum_{k=\tau_t}^{t}\mathbb{E}\big\|\nabla J(\boldsymbol{\theta}_k)\big\|^2\\ &\leq \frac{8U_r}{c_\alpha}\frac{1}{(t+1)^{1-\sigma}}\\ &\quad + \big(D_1 G_{\boldsymbol{\theta}}^2(\tau_t+1)^2 + D_2 G_{\boldsymbol{\theta}} + L_J G_{\boldsymbol{\theta}}^2\big)\frac{2c_\alpha}{1-\sigma}\frac{1}{(t-\tau_t+1)^\sigma}\\ &\quad + 4BG_{\boldsymbol{\theta}}\epsilon_{\text{app}}\\ &\quad + 16B^2\mathcal{E}(t)\\ &= O\Big(\frac{1}{t^{1-\sigma}}\Big) + O\Big(\frac{1}{t^\sigma}\Big) + O(\epsilon_{\text{app}}) + O\big(\mathcal{E}(t)\big).\end{aligned}$$

□

5.7.2 Proof of Theorem 5.4.7: Estimating the Average Reward

The two time-scale analysis with Markovian noise and moving behavior policy can be complicated, so we define some useful notations here that could hopefully clarify the probabilistic dependency.

$$\begin{aligned} O_t &:= (s_t, a_t, s_{t+1}), \\ \eta_t^* &:= \eta^*(\boldsymbol{\theta}_t) = J(\boldsymbol{\theta}_t), \\ y_t &:= \eta_t - \eta_t^*, \\ \Xi(O, \eta, \boldsymbol{\theta}) &:= y_t(r_t - \eta_t^*). \end{aligned} \tag{5.7.5}$$

We also write $J(\boldsymbol{\theta}_t) = r(\boldsymbol{\theta}_t)$ sometimes in the proof.

Lemma 5.7.4. *For any $\boldsymbol{\theta}_1, \boldsymbol{\theta}_2$, we have $|J(\boldsymbol{\theta}_1) - J(\boldsymbol{\theta}_2)| \leq C_J \|\boldsymbol{\theta}_1 - \boldsymbol{\theta}_2\|$, where $C_J = 2U_r|\mathcal{A}|L(1 + \lceil \log_\rho m^{-1} \rceil + 1/(1-\rho))$.*

Lemma 5.7.5. *Given the definition of $\Xi(O_t, \eta_t, \boldsymbol{\theta}_t)$, for any $t > 0$, we have*

$$\begin{aligned} \mathbb{E}[\Xi(O_t, \eta_t, \boldsymbol{\theta}_t)] \leq 4U_r C_J \|\boldsymbol{\theta}_t - \boldsymbol{\theta}_{t-\tau}\| + 2U_r|\eta_t - \eta_{t-\tau}| \\ + 2U_r^2|\mathcal{A}|L \sum_{i=t-\tau}^{t} \mathbb{E}\|\boldsymbol{\theta}_i - \boldsymbol{\theta}_{t-\tau}\|. + 4U_r^2 m\rho^{\tau-1}. \end{aligned}$$

Proof. From the definition, η_t is the average reward estimator, $\eta_t^* = J(\boldsymbol{\theta}_t) = \mathbb{E}[r(s,a)]$ is the average reward under the stationary distribution $\mu_{\boldsymbol{\theta}_t} \otimes \pi_{\boldsymbol{\theta}_t}$, and $y_t = \eta_t - \eta_t^*$. From the algorithm we have the update rule as $\eta_{t+1} := \eta_t + \gamma_t\big(r(s_t, a_t) - \eta_t\big)$, where we leave the step size γ_t unspecified for now. Unrolling the recursive definition we have

$$\begin{aligned} y_{t+1}^2 &= \big(y_t + \eta_t^* - \eta_{t+1}^* + \gamma_t(r_t - \eta_t)\big)^2 \\ &\leq y_t^2 + 2\gamma_t y_t(r_t - \eta_t) + 2y_t(\eta_t^* - \eta_{t+1}^*) + 2(\eta_t^* - \eta_{t+1}^*)^2 + 2\gamma_t^2(r_t - \eta_t)^2 \\ &= (1 - 2\gamma_t)y_t^2 + 2\gamma_t y_t(r_t - \eta_t^*) + 2y_t(\eta_t^* - \eta_{t+1}^*) + 2(\eta_t^* - \eta_{t+1}^*)^2 + 2\gamma_t^2(r_t - \eta_t)^2 \\ &= (1 - 2\gamma_t)y_t^2 + 2\gamma_t \Xi(O_k, \eta_k, \boldsymbol{\theta}_k) + 2y_t(\eta_t^* - \eta_{t+1}^*) + 2(\eta_t^* - \eta_{t+1}^*)^2 + 2\gamma_t^2(r_t - \eta_t)^2. \end{aligned}$$

Rearranging and summing from τ_t to t, we have

$$\begin{aligned} \sum_{k=\tau_t}^{t} \mathbb{E}[y_k^2] \leq \underbrace{\sum_{k=\tau_t}^{t} \frac{1}{2\gamma_k} \mathbb{E}(y_k^2 - y_{k+1}^2)}_{I_1} + \underbrace{\sum_{k=\tau_t}^{t} \mathbb{E}[\Xi(O_k, \eta_k, \boldsymbol{\theta}_k)]}_{I_2} \\ + \underbrace{\sum_{k=\tau_t}^{t} \frac{1}{\gamma_k} \mathbb{E}[y_k(\eta_k^* - \eta_{k+1}^*)]}_{I_3} + \underbrace{\sum_{k=\tau_t}^{t} \frac{1}{\gamma_k} \mathbb{E}[(\eta_k^* - \eta_{k+1}^*)^2]}_{I_4} + \underbrace{\sum_{k=\tau_t}^{t} \gamma_k \mathbb{E}[(r_k - \eta_k)^2]}_{I_5}. \end{aligned}$$

For I_1, following the Abel summation formula, we have

$$\begin{aligned}
I_1 &= \sum_{k=\tau_t}^{t} \frac{1}{2\gamma_k}(y_k^2 - y_{k+1}^2) \\
&= \sum_{k=\tau_t}^{t} \left(\frac{1}{2\gamma_k} - \frac{1}{2\gamma_{k-1}} \right) y_k^2 + \frac{1}{2\gamma_{\tau_t - 1}} y_{\tau_t}^2 - \frac{1}{2\gamma_t} y_{t+1}^2 \\
&\le \frac{2U_r^2}{\gamma_t}.
\end{aligned}$$

For I_2, from Lemma 5.7.5, we have

$$\begin{aligned}
&\mathbb{E}[\Xi(O_t, \eta_t, \boldsymbol{\theta}_t)] \\
&\le 4U_r C_J \|\boldsymbol{\theta}_t - \boldsymbol{\theta}_{t-\tau}\| + 2U_r|\eta_t - \eta_{t-\tau}| + 2U_r^2|\mathcal{A}|L \sum_{i=t-\tau}^{t} \mathbb{E}\|\boldsymbol{\theta}_i - \boldsymbol{\theta}_{t-\tau}\|. + 4U_r^2 m\rho^{\tau-1} \\
&\le 4U_r C_J G_{\boldsymbol{\theta}} \tau\alpha_{t-\tau} + 4U_r^2 \tau\gamma_{t-\tau} + 2U_r^2|\mathcal{A}|L\tau(\tau+1)G_{\boldsymbol{\theta}}\alpha_{t-\tau} + 4U_r^2 m\rho^{\tau-1} \\
&\le C_1\tau^2\alpha_{t-\tau} + C_2\tau\gamma_{t-\tau} + C_3 m\rho^{\tau-1}.
\end{aligned}$$

By the choice of τ_t, we have

$$I_2 = \sum_{k=\tau_t}^{t} \mathbb{E}[\Xi(O_k, \eta_k, \boldsymbol{\theta}_k)] \le (C_1\tau_t^2 + C_3)\sum_{k=\tau_t}^{t}\alpha_k + C_2\tau_t\sum_{k=\tau_t}^{t}\gamma_k.$$

For I_3, we have

$$I_3 \le \left(\sum_{k=\tau_t}^{t} \mathbb{E}[y_k^2] \right)^{1/2} \left(C_J^2 G_{\boldsymbol{\theta}}^2 \sum_{k=\tau_t}^{t} \frac{\alpha_k^2}{\gamma_k^2} \right)^{1/2},$$

which is because by Lemma 5.7.4, $(\eta_k^* - \eta_{k+1}^*)$ can be linearly bounded by $\|\boldsymbol{\theta}_k - \boldsymbol{\theta}_{k+1}\| \le G_{\boldsymbol{\theta}} \cdot \alpha_k$.

For I_4, by the same argument it holds that

$$\begin{aligned}
I_4 &= \sum_{k=\tau_t}^{t} \frac{1}{\gamma_k} \mathbb{E}[(\eta_k^* - \eta_{k+1}^*)^2] \\
&= \sum_{k=\tau_t}^{t} \frac{1}{\gamma_k} \mathbb{E}\big[\big(J(\boldsymbol{\theta}_k) - J(\boldsymbol{\theta}_{k+1})\big)^2\big] \\
&\le \sum_{k=\tau_t}^{t} \frac{1}{\gamma_k} C_J^2 \|\boldsymbol{\theta}_k - \boldsymbol{\theta}_{k+1}\|^2 \\
&\le \sum_{k=\tau_t}^{t} \frac{1}{\gamma_k} C_J^2 G_{\boldsymbol{\theta}}^2 \alpha_k^2
\end{aligned}$$

$$= O\bigg(\sum_{k=\tau_t}^{t} \frac{\alpha_k^2}{\gamma_k} \bigg).$$

For I_5, we have

$$I_5 = \sum_{k=\tau_t}^{t} \gamma_k \mathbb{E}[(r_k - \eta_k)^2] \leq \sum_{k=\tau_t}^{t} 4U_r^2 \gamma_k = O\bigg(\sum_{k=\tau_t}^{t} \gamma_k \bigg),$$

by bounding the expectation uniformly.

Now, we set $\gamma_k = 1/(1+t)^\nu$ and combine all the terms together to get

$$\begin{aligned}
\sum_{k=\tau_t}^{t} \mathbb{E}[y_k^2] &\leq 2U_r^2(1+t)^\nu + (C_1\tau_t^2 + C_3)c_\alpha \sum_{k=\tau_t}^{t}(1+k)^{-\sigma} + C_2\tau_t \sum_{k=\tau_t}^{t}(1+k)^{-\nu} \\
&\quad + C_J G_{\boldsymbol{\theta}} c_\alpha \bigg(\sum_{k=\tau_t}^{t} \mathbb{E}[y_k^2] \bigg)^{1/2} \bigg(\sum_{k=\tau_t}^{t} (1+k)^{-2(\sigma-\nu)} \bigg)^{1/2} \\
&\quad + C_J^2 G_{\boldsymbol{\theta}}^2 c_\alpha^2 \sum_{k=\tau_t}^{t} (1+k)^{\nu-2\sigma} + 4U_r^2 \sum_{k=\tau_t}^{t}(1+k)^{-\nu} \\
&\leq 2U_r^2(1+t)^\nu + \big[(C_1\tau^2 + C_3)c_\alpha + C_2\tau_t + C_J^2 G_{\boldsymbol{\theta}}^2 c_\alpha^2 + 4U_r^2\big] \sum_{k=\tau_t}^{t}(1+k)^{-\nu} \\
&\quad + C_J G_{\boldsymbol{\theta}} c_\alpha \bigg(\sum_{k=\tau_t}^{t} \mathbb{E}[y_k^2] \bigg)^{1/2} \bigg(\sum_{k=\tau_t}^{t} (1+k)^{-2(\sigma-\nu)} \bigg)^{1/2} \\
&\leq 2U_r^2(1+t)^\nu + \big[(C_1\tau^2 + C_3)c_\alpha + C_2\tau_t + C_J^2 G_{\boldsymbol{\theta}}^2 c_\alpha^2 + 4U_r^2\big] \frac{(1+t-\tau_t)^{1-\nu}}{1-\nu} \\
&\quad + C_J G_{\boldsymbol{\theta}} c_\alpha \bigg(\sum_{k=\tau_t}^{t} \mathbb{E}[y_k^2] \bigg)^{1/2} \bigg(\frac{(1+t-\tau_t)^{1-2(\sigma-\nu)}}{1-2(\sigma-\nu)} \bigg)^{1/2}
\end{aligned}$$

By applying the squaring technique already stated in the proof of Theorem 5.4.5, we have

$$\begin{aligned}
\sum_{k=\tau_t}^{t} \mathbb{E}[y_k^2] &\leq 4U_r^2(1+t)^\nu + 2\big[(C_1\tau_t^2 + C_3)c_\alpha + C_2\tau_t + C_J^2 G_{\boldsymbol{\theta}}^2 c_\alpha^2 + 4U_r^2\big] \frac{(1+t-\tau_t)^{1-\nu}}{1-\nu} \\
&\quad + 8C_J^2 G_{\boldsymbol{\theta}}^2 c_\alpha^2 \frac{(1+t-\tau_t)^{1-2(\sigma-\nu)}}{1-2(\sigma-\nu)} \qquad (5.7.6) \\
&= O(t^\nu) + O(\log^2 t \cdot t^{1-\nu}) + O(t^{1-2(\sigma-\nu)}).
\end{aligned}$$

□

5.7.3 Proof of Theorem 5.4.7: Approximating the TD Fixed Point

Now we deal with the critic's parameter $\boldsymbol{\omega}_t$. The two time-scale analysis with Markovian noise and moving behavior policy can be complicated, so we define some useful notations here that could hopefully clarify the probabilistic dependency.

$$
\begin{aligned}
O_t &:= (s_t, a_t, s_{t+1}), \\
g(O, \boldsymbol{\omega}, \boldsymbol{\theta}) &:= [r(s,a) - J(\boldsymbol{\theta}) + (\boldsymbol{\phi}(s') - \boldsymbol{\phi}(s))^\top \boldsymbol{\omega}]\boldsymbol{\phi}(s), \\
\Delta g(O, \eta, \boldsymbol{\theta}) &:= [J(\boldsymbol{\theta}) - \eta]\boldsymbol{\phi}(s), \\
\bar{g}(\boldsymbol{\omega}, \boldsymbol{\theta}) &:= \mathbb{E}_{s\sim\mu_{\boldsymbol{\theta}}, a\sim\pi_{\boldsymbol{\theta}}, s'\sim\mathcal{P}}\Big[\big[r(s,a) - J(\boldsymbol{\theta}) + \big(\boldsymbol{\phi}(s') - \boldsymbol{\phi}(s)\big)^\top \boldsymbol{\omega}\big]\boldsymbol{\phi}(s)\Big], \\
\boldsymbol{\omega}_t^* &:= \boldsymbol{\omega}^*(\boldsymbol{\theta}_t), \\
\eta_t^* &:= \eta^*(\boldsymbol{\theta}_t) = J(\boldsymbol{\theta}_t) \\
\Lambda(O, \boldsymbol{\omega}, \boldsymbol{\theta}) &:= \big\langle \boldsymbol{\omega} - \boldsymbol{\omega}^*(\boldsymbol{\theta}), g(O, \boldsymbol{\omega}, \boldsymbol{\theta}) - \bar{g}(\boldsymbol{\omega}, \boldsymbol{\theta})\big\rangle, \\
\mathbf{z}_t &:= \boldsymbol{\omega}_t - \boldsymbol{\omega}_t^* \\
y_t &:= \eta_t - \eta_t^*.
\end{aligned} \tag{5.7.7}
$$

A bounded lemma is used frequently in this section.

Lemma 5.7.6. *Under Assumption 5.4.3, for any $\boldsymbol{\theta}$, $\boldsymbol{\omega}$, $O = (s, a, s')$ such that $\|\boldsymbol{\omega}\| \le R_{\boldsymbol{\omega}}$,*

$$\big\|g(O, \boldsymbol{\omega}, \boldsymbol{\theta})\big\| \le U_\delta := 2U_r + 2R_{\boldsymbol{\omega}}, \quad \big\|\Delta g(O, \eta, \boldsymbol{\theta})\big\| \le 2U_r, \quad \big|\Lambda(O, \boldsymbol{\omega}, \boldsymbol{\theta})\big| \le 2R_{\boldsymbol{\omega}} \cdot 2U_\delta \le 2U_\delta^2.$$

The following lemma is used to control the bias due to Markovian noise.

Lemma 5.7.7. *Given the definition of $\Lambda(\boldsymbol{\theta}_t, \boldsymbol{\omega}_t, O_t)$, for any $0 \le \tau \le t$, we have*

$$\mathbb{E}[\Lambda(O_t, \boldsymbol{\omega}_t, \boldsymbol{\theta}_t)] \le C_1(\tau + 1)\|\boldsymbol{\theta}_t - \boldsymbol{\theta}_{t-\tau}\| + C_2 m\rho^{\tau-1} + C_3\|\boldsymbol{\omega}_t - \boldsymbol{\omega}_{t-\tau}\|,$$

where $C_1 = 2U_\delta^2|\mathcal{A}|L(1 + \lceil\log_\rho m^{-1}\rceil + 1/(1-\rho)) + 2U_\delta L_*, C_2 = 2U_\delta^2, C_3 = 4U_\delta$ are constants.

Proof of Theorem 5.4.7. By the updating rule of $\boldsymbol{\omega}_t$ in Algorithm 9, unrolling and decomposing the squared error gives

$$\|\mathbf{z}_{t+1}\|^2 = \big\|\mathbf{z}_t + \beta_t(g(O_t, \boldsymbol{\omega}_t, \boldsymbol{\theta}_t) + \Delta g(O_t, \eta_t, \boldsymbol{\theta}_t)) + (\boldsymbol{\omega}_t^* - \boldsymbol{\omega}_{t+1}^*)\big\|^2$$

$$
\begin{aligned}
&= \|\mathbf{z}_t\|^2 + 2\beta_t\big\langle \mathbf{z}_t, g(O_t,\boldsymbol{\omega}_t,\boldsymbol{\theta}_t)\big\rangle + 2\beta_t\big\langle \mathbf{z}_t, \Delta g(O_t,\eta_t,\boldsymbol{\theta}_t)\big\rangle \\
&\qquad + 2\langle \mathbf{z}_t, \boldsymbol{\omega}_t^* - \boldsymbol{\omega}_{t+1}^*\rangle + \big\|\beta_t(g(O_t,\boldsymbol{\omega}_t,\boldsymbol{\theta}_t) + \Delta g(O_t,\eta_t,\boldsymbol{\theta}_t)) + (\boldsymbol{\omega}_t^* - \boldsymbol{\omega}_{t+1}^*)\big\|^2 \\
&= \|\mathbf{z}_t\|^2 + 2\beta_t\big\langle \mathbf{z}_t, \bar{g}(\boldsymbol{\omega}_t,\boldsymbol{\theta}_t)\big\rangle + 2\beta_t\Lambda(O_t,\boldsymbol{\omega}_t,\boldsymbol{\theta}_t) + 2\beta_t\big\langle \mathbf{z}_t, \Delta g(O_t,\eta_t,\boldsymbol{\theta}_t)\big\rangle \\
&\qquad + 2\langle \mathbf{z}_t, \boldsymbol{\omega}_t^* - \boldsymbol{\omega}_{t+1}^*\rangle + \big\|\beta_t(g(O_t,\boldsymbol{\omega}_t,\boldsymbol{\theta}_t) + \Delta g(O_t,\eta_t,\boldsymbol{\theta}_t)) + (\boldsymbol{\omega}_t^* - \boldsymbol{\omega}_{t+1}^*)\big\|^2 \\
&\le \|\mathbf{z}_t\|^2 + 2\beta_t\big\langle \mathbf{z}_t, \bar{g}(\boldsymbol{\omega}_t,\boldsymbol{\theta}_t)\big\rangle + 2\beta_t\Lambda(O_t,\boldsymbol{\omega}_t,\boldsymbol{\theta}_t) + 2\beta_t\big\langle \mathbf{z}_t, \Delta g(O_t,\eta_t,\boldsymbol{\theta}_t)\big\rangle \\
&\qquad + 2\langle \mathbf{z}_t, \boldsymbol{\omega}_t^* - \boldsymbol{\omega}_{t+1}^*\rangle + 2\beta_t^2\big\|g(O_t,\boldsymbol{\omega}_t,\boldsymbol{\theta}_t) + \Delta g(O_t,\eta_t,\boldsymbol{\theta}_t)\big\|^2 + 2\|\boldsymbol{\omega}_t^* - \boldsymbol{\omega}_{t+1}^*\|^2 \\
&\le \|\mathbf{z}_t\|^2 + 2\beta_t\big\langle \mathbf{z}_t, \bar{g}(\boldsymbol{\omega}_t,\boldsymbol{\theta}_t)\big\rangle + 2\beta_t\Lambda(O_t,\boldsymbol{\omega}_t,\boldsymbol{\theta}_t) + 2\beta_t\big\langle \mathbf{z}_t, \Delta g(O_t,\eta_t,\boldsymbol{\theta}_t)\big\rangle \\
&\qquad + 2\langle \mathbf{z}_t, \boldsymbol{\omega}_t^* - \boldsymbol{\omega}_{t+1}^*\rangle + 2U_\delta^2\beta_t^2 + 2\|\boldsymbol{\omega}_t^* - \boldsymbol{\omega}_{t+1}^*\|^2,
\end{aligned}
$$

where the first inequality is due to $\|\mathbf{x}+\mathbf{y}\|^2 \le 2\|\mathbf{x}\|^2 + 2\|\mathbf{y}\|^2$ and the second is due to $\|g(O_t,\boldsymbol{\omega}_t,\boldsymbol{\theta}_t) + \Delta g(O_t,\eta_t,\boldsymbol{\theta}_t)\| \le U_\delta$. First, note that due to Assumption 5.4.1, we have

$$
\begin{aligned}
\big\langle \mathbf{z}_t, \bar{g}(\boldsymbol{\omega}_t,\boldsymbol{\theta}_t)\big\rangle &= \big\langle \mathbf{z}_t, \bar{g}(\boldsymbol{\omega}_t,\boldsymbol{\theta}_t) - \bar{g}(\boldsymbol{\omega}_t^*,\boldsymbol{\theta}_t)\big\rangle \\
&= \Big\langle \mathbf{z}_t, \mathbb{E}\big[\big(\boldsymbol{\phi}(s') - \boldsymbol{\phi}(s)\big)^\top(\boldsymbol{\omega}_t - \boldsymbol{\omega}_t^*)\boldsymbol{\phi}(s)\big]\Big\rangle \\
&= \mathbf{z}_t^\top\mathbb{E}\big[\boldsymbol{\phi}(s)\big(\boldsymbol{\phi}(s') - \boldsymbol{\phi}(s)\big)^\top\big]\mathbf{z}_t \\
&= \mathbf{z}_t^\top\mathbf{A}\mathbf{z}_t \\
&\le -\lambda\|\mathbf{z}_t\|^2,
\end{aligned}
$$

where the first equation is due to the fact that $\bar{g}(\boldsymbol{\omega}^*,\boldsymbol{\theta}) = 0$ [SB18]. Taking expectation up to s_{t+1}, we have

$$
\begin{aligned}
\mathbb{E}\|\mathbf{z}_{t+1}\|^2 &\le \mathbb{E}\|\mathbf{z}_t\|^2 + 2\beta_t\mathbb{E}\big\langle \mathbf{z}_t, \bar{g}(\boldsymbol{\omega}_t,\boldsymbol{\theta}_t)\big\rangle + 2\beta_t\mathbb{E}\Lambda(O_t,\boldsymbol{\omega}_t,\boldsymbol{\theta}_t) + 2\beta_t\mathbb{E}\big\langle \mathbf{z}_t, \Delta g(O_t,\eta_t,\boldsymbol{\theta}_t)\big\rangle \\
&\qquad + 2\mathbb{E}\langle \mathbf{z}_t, \boldsymbol{\omega}_t^* - \boldsymbol{\omega}_{t+1}^*\rangle + 2U_\delta^2\beta_t^2 + 2\mathbb{E}\|\boldsymbol{\omega}_t^* - \boldsymbol{\omega}_{t+1}^*\|^2 \\
&\le (1 - 2\lambda\beta_t)\mathbb{E}\|\mathbf{z}_t\|^2 + 2\beta_t\mathbb{E}\Lambda(O_t,\boldsymbol{\omega}_t,\boldsymbol{\theta}_t) + 2\beta_t\mathbb{E}\big\langle \mathbf{z}_t, \Delta g(O_t,\eta_t,\boldsymbol{\theta}_t)\big\rangle \\
&\qquad + 2\mathbb{E}\langle \mathbf{z}_t, \boldsymbol{\omega}_t^* - \boldsymbol{\omega}_{t+1}^*\rangle + 2U_\delta^2\beta_t^2 + 2\mathbb{E}\|\boldsymbol{\omega}_t^* - \boldsymbol{\omega}_{t+1}^*\|^2.
\end{aligned}
$$

Based on the result above, we can further rewrite it as:

$$
\begin{aligned}
\mathbb{E}\|\mathbf{z}_{t+1}\|^2 &\le (1 - 2\lambda\beta_t)\mathbb{E}\|\mathbf{z}_t\|^2 + 2\beta_t\mathbb{E}\Lambda(O_t,\boldsymbol{\omega}_t,\boldsymbol{\theta}_t) + 2\beta_t\mathbb{E}\|\mathbf{z}_t\|\cdot|y_t| \\
&\qquad + 2L_*\mathbb{E}\|\mathbf{z}_t\|\cdot\|\boldsymbol{\theta}_t - \boldsymbol{\theta}_{t+1}\| + 2U_\delta^2\beta_t^2 + 2L_*^2\mathbb{E}\|\boldsymbol{\theta}_t - \boldsymbol{\theta}_{t+1}\|^2
\end{aligned}
$$

$$
\begin{aligned}
&\le (1-2\lambda\beta_t)\mathbb{E}\|\mathbf{z}_t\|^2 + 2\beta_t\mathbb{E}\Lambda(O_t,\boldsymbol{\omega}_t,\boldsymbol{\theta}_t) + 2\beta_t\mathbb{E}\|\mathbf{z}_t\|\cdot|y_t| \\
&\qquad + 2L_*G_{\boldsymbol{\theta}}\alpha_t\mathbb{E}\|\mathbf{z}_t\| + 2U_\delta^2\beta_t^2 + 2L_*^2G_{\boldsymbol{\theta}}^2\alpha_t^2 \\
&\le (1-2\lambda\beta_t)\mathbb{E}\|\mathbf{z}_t\|^2 + 2\beta_t\mathbb{E}\Lambda(O_t,\boldsymbol{\omega}_t,\boldsymbol{\theta}_t) + 2\beta_t\mathbb{E}\|\mathbf{z}_t\|\cdot|y_t| \\
&\qquad + 2L_*G_{\boldsymbol{\theta}}\alpha_t\mathbb{E}\|\mathbf{z}_t\| + \left(2U_\delta^2 + 2L_*^2G_{\boldsymbol{\theta}}^2\Big(\max_t\frac{\alpha_t}{\beta_t}\Big)^2\right)\beta_t^2 \\
&= (1-2\lambda\beta_t)\mathbb{E}\|\mathbf{z}_t\|^2 + 2\beta_t\mathbb{E}\Lambda(O_t,\boldsymbol{\omega}_t,\boldsymbol{\theta}_t) + 2\beta_t\mathbb{E}\|\mathbf{z}_t\|\cdot|y_t| + 2L_*G_{\boldsymbol{\theta}}\alpha_t\mathbb{E}\|\mathbf{z}_t\| + C_q\beta_t^2,
\end{aligned}
$$

where we denote the constant coefficient before the quadratic stepsize β_t^2 as C_q at the last step. The first inequality is due to Proposition 5.4.4 and Cauchy-Schwartz inequality. The second inequality is due to the update of $\boldsymbol{\theta}_t$ is bounded by $G_{\boldsymbol{\theta}}\alpha_t$. The third inequality is from employing the fact that $\sigma > \nu$ so α_t/β_t is bounded. Rearranging the inequality yields

$$
\begin{aligned}
2\lambda\mathbb{E}\|\mathbf{z}_t\|^2 &\le \frac{1}{\beta_t}\big(\mathbb{E}\|\mathbf{z}_t\|^2 - \mathbb{E}\|\mathbf{z}_{t+1}\|^2\big) + 2\mathbb{E}\Lambda(O_t,\boldsymbol{\omega}_t,\boldsymbol{\theta}_t) + \mathbb{E}\|\mathbf{z}_t\|\cdot|y_t| + 2L_*G_{\boldsymbol{\theta}}\frac{\alpha_t}{\beta_t}\mathbb{E}\|\mathbf{z}_t\| + C_q\beta_t \\
&\le \frac{1}{\beta_t}\big(\mathbb{E}\|\mathbf{z}_t\|^2 - \mathbb{E}\|\mathbf{z}_{t+1}\|^2\big) + 2\mathbb{E}\Lambda(O_t,\boldsymbol{\omega}_t,\boldsymbol{\theta}_t) + \sqrt{\mathbb{E}y_t^2}\cdot\sqrt{\mathbb{E}\|\mathbf{z}_t\|^2} \\
&\qquad + 2L_*G_{\boldsymbol{\theta}}\frac{\alpha_t}{\beta_t}\sqrt{\mathbb{E}\|\mathbf{z}_t\|^2} + C_q\beta_t,
\end{aligned}
$$

where the second inequality is due to the concavity of square root function. Telescoping from τ_t to t gives:

$$
\begin{aligned}
2\lambda\sum_{k=\tau_t}^t\mathbb{E}\|\mathbf{z}_k\|^2 \le{}& \underbrace{\sum_{k=\tau_t}^t\frac{1}{\beta_k}\big(\mathbb{E}\|\mathbf{z}_k\|^2 - \mathbb{E}\|\mathbf{z}_{k+1}\|^2\big)}_{I_1} + 2\underbrace{\sum_{k=\tau_t}^t\mathbb{E}\Lambda(\boldsymbol{\theta}_k,\boldsymbol{\omega}_k,O_k)}_{I_2} \\
&+ 2L_*G_{\boldsymbol{\theta}}\underbrace{\sum_{k=\tau_t}^t\frac{\alpha_k}{\beta_k}\sqrt{\mathbb{E}\|\mathbf{z}_k\|^2}}_{I_3} + \underbrace{\sum_{k=\tau_t}^t\sqrt{\mathbb{E}y_k^2}\cdot\sqrt{\mathbb{E}\|\mathbf{z}_k\|^2}}_{I_4} + C_q\underbrace{\sum_{k=\tau_t}^t\beta_k}_{I_5}. \qquad (5.7.8)
\end{aligned}
$$

From (5.7.8), we can see the proof of the critic again shares the same spirit with the proof of Theorem 5.4.5. For term I_1, we have

$$
\begin{aligned}
I_1 &:= \sum_{k=\tau_t}^t\frac{1}{\beta_k}(\mathbb{E}\|\mathbf{z}_k\|^2 - \mathbb{E}\|\mathbf{z}_{k+1}\|^2) \\
&= \sum_{k=\tau_t}^t\left(\frac{1}{\beta_k} - \frac{1}{\beta_{k-1}}\right)\mathbb{E}\|\mathbf{z}_k\|^2 + \frac{1}{\beta_{\tau_t-1}}\mathbb{E}\|\mathbf{z}_{\tau_t}\|^2 - \frac{1}{\beta_t}\mathbb{E}\|\mathbf{z}_{t+1}\|^2
\end{aligned}
$$

$$
\begin{aligned}
&\le \sum_{k=\tau_t}^{t}\left(\frac{1}{\beta_k}-\frac{1}{\beta_{k-1}}\right)\mathbb{E}\|\mathbf{z}_k\|^2+\frac{1}{\beta_{\tau_t-1}}\mathbb{E}\|\mathbf{z}_{\tau_t}\|^2\\
&\le 4R_{\boldsymbol{\omega}}^2\left(\sum_{k=\tau_t}^{t}\left(\frac{1}{\beta_k}-\frac{1}{\beta_{k-1}}\right)+\frac{1}{\beta_{\tau_t-1}}\right)\\
&= 4R_{\boldsymbol{\omega}}^2\frac{1}{\beta_t}\\
&= 4R_{\boldsymbol{\omega}}^2(1+t)^{\nu}=O(t^{\nu}),
\end{aligned}
$$

where the first inequality is due to discarding the last term, and the second inequality is due to $\mathbb{E}\|\mathbf{z}_k\|^2 \le (R_{\boldsymbol{\omega}}+R_{\boldsymbol{\omega}})^2$.

For term I_2, note that due to Lemma 5.7.7, we actually have

$$
\begin{aligned}
\Lambda(O_k,\boldsymbol{\omega}_k,\boldsymbol{\theta}_k) &\le C_1(\tau_t+1)\|\boldsymbol{\theta}_k-\boldsymbol{\theta}_{k-\tau_t}\|+C_2 m\rho^{\tau_t-1}+C_3\|\boldsymbol{\omega}_k-\boldsymbol{\omega}_{k-\tau_t}\|\\
&\le C_1(\tau_t+1)\sum_{i=k-\tau_t}^{k-1}G_{\boldsymbol{\theta}}\alpha_i+C_2 m\rho^{\tau_t-1}+C_3\sum_{i=k-\tau_t}^{k-1}U_\delta\beta_i\\
&\le C_1G_{\boldsymbol{\theta}}(\tau_t+1)^2\alpha_{k-\tau_t}+C_2\alpha_t+C_3U_\delta\tau_t\beta_k,
\end{aligned}
$$

and the summation is

$$
\begin{aligned}
I_2 &:= \sum_{k=\tau_t}^{t}\mathbb{E}\Lambda(O_k,\boldsymbol{\omega}_k,\boldsymbol{\theta}_k)\\
&\le C_1G_{\boldsymbol{\theta}}(\tau_t+1)^2\sum_{k=\tau_t}^{t}\alpha_{k-\tau_t}+C_2\sum_{k=\tau_t}^{t}\alpha_t+C_3U_\delta\tau_t\sum_{k=\tau_t}^{t}\beta_k\\
&\le C_1G_{\boldsymbol{\theta}}(\tau_t+1)^2\sum_{k=0}^{t-\tau_t}\alpha_k+C_2(t-\tau_t+1)\alpha_t+C_3U_\delta\tau_t\sum_{k=0}^{t-\tau_t}\beta_k\\
&\le C_1G_{\boldsymbol{\theta}}(\tau_t+1)^2c_\alpha\frac{(1+t-\tau_t)^{1-\sigma}}{1-\sigma}+C_2(t-\tau_t+1)c_\alpha(1+t)^{-\sigma}+C_3U_\delta\tau_t\frac{(1+t-\tau_t)^{1-\nu}}{1-\nu}\\
&\le \left[\frac{C_1G_{\boldsymbol{\theta}}(\tau_t+1)^2c_\alpha}{1-\sigma}+C_2c_\alpha+\frac{C_3U_\delta\tau_t}{1-\nu}\right](1+t)^{1-\nu}\\
&= O\big((\log t)^2t^{1-\nu}\big),
\end{aligned}
$$

where the second inequality is due to the monotonicity of α_k and β_k. The $O(\cdot)$ comes from that $\tau=O(\log t)$ and $\sum k^{-\nu}=O(t^{1-\nu})$.

For term I_3 and I_4, we will instead show it can be bounded in a different form. Using

Cauchy-Schwartz inequality we have

$$I_3 := \sum_{k=\tau_t}^{t} \frac{\alpha_k}{\beta_k}\sqrt{\mathbb{E}\|\mathbf{z}_k\|^2} \le \Big(\sum_{k=\tau_t}^{t}\frac{\alpha_k^2}{\beta_k^2}\Big)^{\frac{1}{2}}\Big(\sum_{k=\tau_t}^{t}\mathbb{E}\|\mathbf{z}_k\|^2\Big)^{\frac{1}{2}} \le \Big(\sum_{k=0}^{t-\tau_t}\frac{\alpha_k^2}{\beta_k^2}\Big)^{\frac{1}{2}}\Big(\sum_{k=\tau_t}^{t}\mathbb{E}\|\mathbf{z}_k\|^2\Big)^{\frac{1}{2}},$$

$$I_4 := \sum_{k=\tau_t}^{t} \sqrt{\mathbb{E}y_k^2}\cdot\sqrt{\mathbb{E}\|\mathbf{z}_k\|^2} \le \Big(\sum_{k=\tau_t}^{t}\mathbb{E}y_k^2\Big)^{\frac{1}{2}}\Big(\sum_{k=\tau_t}^{t}\mathbb{E}\|\mathbf{z}_k\|^2\Big)^{\frac{1}{2}} \le \Big(\sum_{k=0}^{t-\tau_t}\mathbb{E}y_k^2\Big)^{\frac{1}{2}}\Big(\sum_{k=\tau_t}^{t}\mathbb{E}\|\mathbf{z}_k\|^2\Big)^{\frac{1}{2}}.$$

For term I_5, simply bound it as $\sum_{k=0}^{t-\tau_t}\beta_k \le (1+t)^{1-\nu}/(1-\nu)$.

Collecting the upper bounds of the above five terms, and writing them using $O(\cdot)$ notation give

$$\begin{aligned}2\lambda\sum_{k=\tau_t}^{t}\mathbb{E}\|\mathbf{z}_k\|^2 \le 4R_{\boldsymbol{\omega}}^2(1+t)^{\nu} &+ 2\Big[\frac{C_1G_{\boldsymbol{\theta}}(\tau_t+1)^2c_\alpha}{1-\sigma} + C_2c_\alpha + \frac{C_3U_\delta\tau_t + C_q}{1-\nu}\Big](1+t)^{1-\nu}\\ &+ 2L_*G_{\boldsymbol{\theta}}\Big(\sum_{k=0}^{t-\tau_t}\frac{\alpha_k^2}{\beta_k^2}\Big)^{\frac{1}{2}}\Big(\sum_{k=\tau_t}^{t}\mathbb{E}\|\mathbf{z}_k\|^2\Big)^{\frac{1}{2}}\\ &+ \Big(\sum_{k=0}^{t-\tau_t}\mathbb{E}y_k^2\Big)^{\frac{1}{2}}\Big(\sum_{k=\tau_t}^{t}\mathbb{E}\|\mathbf{z}_k\|^2\Big)^{\frac{1}{2}}.\end{aligned} \tag{5.7.9}$$

Now, we first divide both sides by $(1+t-\tau_t)$, and denote

$$\begin{aligned}Z(t) &:= \frac{1}{1+t-\tau_t}\sum_{k=\tau_t}^{t}\mathbb{E}\|\mathbf{z}_k\|^2,\\ F(t) &:= \frac{1}{1+t-\tau_t}\sum_{k=0}^{t-\tau_t}\frac{\alpha_k^2}{\beta_k^2} \le \frac{t^{-2(\sigma-\nu)}}{1-2(\sigma-\nu)} = O(t^{-2(\sigma-\nu)}),\\ G(t) &:= \frac{1}{1+t-\tau_t}\sum_{k=0}^{t-\tau_t}\mathbb{E}[y_k^2] = O(t^{\nu-1}) + O(\log t\cdot t^{-\nu}) + O(t^{-2(\sigma-\nu)}),\end{aligned}$$

and the rest as $A(t) = O(t^{\nu}) + O(t^{1-\nu})$. $G(t)$'s constants appear at (5.7.6) in exact form. This simplification leads to

$$2\lambda\Big(\sqrt{Z(t)} - \frac{L_*G_{\boldsymbol{\theta}}}{2\lambda}\cdot\sqrt{F(t)} - \frac{1}{4\lambda}\sqrt{G(t)}\Big)^2 \le A(t) + 2\lambda\Big(\frac{L_*G_{\boldsymbol{\theta}}}{2\lambda}\sqrt{F(t)} + \frac{1}{4\lambda}\sqrt{G(t)}\Big)^2,$$

which further gives $Z(t) \le A(t)/\lambda + 16F(t) + 16G(t)$. This is again a similar reasoning as in the end of the proof of Theorem 5.4.5. We actually show that

$$\frac{1}{1+t-\tau_t}\sum_{k=\tau_t}^{t}\mathbb{E}\|\boldsymbol{\omega}_k - \boldsymbol{\omega}_k^*\|^2 = O\Big(\frac{1}{t^{1-\nu}}\Big) + O\Big(\frac{\log t}{t^{\nu}}\Big) + O\Big(\frac{1}{t^{2(\sigma-\nu)}}\Big).$$

This completes the proof. To obtain the exact constant, please refer to (5.7.6) and (5.7.9). □

5.7.4 Proof of Corollary 5.4.9

Proof of Corollary 5.4.9. By Theorem 5.4.7, we have

$$\frac{1}{1+t-\tau_t}\sum_{k=\tau_t}^{t}\mathbb{E}\|\boldsymbol{\omega}_k-\boldsymbol{\omega}_k^*\|^2=O\bigg(\frac{1}{t^{1-\nu}}\bigg)+O\bigg(\frac{\log t}{t^{\nu}}\bigg)+O\bigg(\frac{1}{t^{2(\sigma-\nu)}}\bigg).$$

By Lemma 5.6.3, $\mathcal{E}(t)$ in Theorem 5.4.5 is of the equivalent order:

$$\begin{aligned}
\mathcal{E}_1(t)&=\frac{1}{t}\sum_{k=1}^{t}\mathbb{E}\|\boldsymbol{\omega}_k-\boldsymbol{\omega}_k^*\|^2\\
&=O\bigg(\frac{1}{1+t-\tau_t}\sum_{k=\tau_t}^{t}\mathbb{E}\|\boldsymbol{\omega}_k-\boldsymbol{\omega}_k^*\|^2\bigg)+O\bigg(\frac{\log t}{t}\bigg)\\
&=O\bigg(\frac{1}{t^{1-\nu}}\bigg)+O\bigg(\frac{\log t}{t^{\nu}}\bigg)+O\bigg(\frac{1}{t^{2(\sigma-\nu)}}\bigg)+O\bigg(\frac{\log t}{t}\bigg)\\
&=O\bigg(\frac{1}{t^{1-\nu}}\bigg)+O\bigg(\frac{\log t}{t^{\nu}}\bigg)+O\bigg(\frac{1}{t^{2(\sigma-\nu)}}\bigg).
\end{aligned}$$

The same reasoning also applies to

$$\mathcal{E}_2(t)=\frac{1}{t}\sum_{k=1}^{t}\mathbb{E}(\eta_k-r(\boldsymbol{\theta}_k))^2=O\bigg(\frac{1}{t^{1-\nu}}\bigg)+O\bigg(\frac{\log t}{t^{\nu}}\bigg)+O\bigg(\frac{1}{t^{2(\sigma-\nu)}}\bigg).$$

Plugging the above results into Theorem 5.4.5, and optimizing over the choice of σ and ν (which gives $\sigma=3/5$ and $\nu=2/5$), we have

$$\begin{aligned}
\min_{0\le k\le t}\mathbb{E}\|\nabla J(\boldsymbol{\theta}_k)\|^2&=O\bigg(\frac{1}{t^{1-\sigma}}\bigg)+O\bigg(\frac{\log^2 t}{t^{\sigma}}\bigg)+O\bigg(\frac{1}{t^{1-\nu}}\bigg)+O\bigg(\frac{\log t}{t^{\nu}}\bigg)\\
&\quad+O\bigg(\frac{1}{t^{2(\sigma-\nu)}}\bigg)+O(\epsilon_{\text{app}})\\
&=O\bigg(\frac{1}{t^{1-\sigma}}\bigg)+O\bigg(\frac{\log t}{t^{\nu}}\bigg)+O\bigg(\frac{1}{t^{2(\sigma-\nu)}}\bigg)+O(\epsilon_{\text{app}})\\
&=O\bigg(\frac{\log t}{t^{2/5}}\bigg)+O(\epsilon_{\text{app}}).
\end{aligned}$$

Therefore, in order to obtain an ϵ-approximate(ignoring the approximation error) stationary point of J, namely,

$$\min_{0\le k\le T}\mathbb{E}\big\|\nabla J(\boldsymbol{\theta}_k)\big\|^2=O\bigg(\frac{\log T}{T^{2/5}}\bigg)+O(\epsilon_{\text{app}})\le O(\epsilon_{\text{app}})+\epsilon,$$

we need to set $T=\tilde{O}(\epsilon^{-2.5})$. □

5.8 Proof of Technical Lemmas

5.8.1 Proof of Lemma 5.7.1

Proof of Lemma 5.7.1. The first inequality comes from Lemma 3.2 in [ZKZB19]. The second inequality is well known as a partial result of $[-L, L]$-smoothness of non-convex functions. $\square$

5.8.2 Proof of Lemma 5.7.2

Proof of Lemma 5.7.2. Applying the definition of $\Delta h()$ and Cauchy-Schwartz inequality immediately yields the result. $\square$

5.8.3 Proof of Lemma 5.7.3

The proof of Lemma 5.7.3 will be built on the following supporting lemmas.

Lemma 5.8.1. *For any $t \geq 0$,*

$$\big|\Gamma(O_t, \boldsymbol{\theta}_t) - \Gamma(O_t, \boldsymbol{\theta}_{t-\tau})\big| \leq G_{\boldsymbol{\theta}}(U_\delta L_l + 2L_* B + 3L_J)\|\boldsymbol{\theta}_t - \boldsymbol{\theta}_{t-\tau}\|.$$

Proof of Lemma 5.8.1. Let $\delta(O_t, \boldsymbol{\theta}) := r(s_t, a_t) + (\boldsymbol{\phi}(s_{t+1}) - \boldsymbol{\phi}(s_t))^\top \boldsymbol{\omega}^* - r(\boldsymbol{\theta})$ and it can be shown that $\delta(O_t, \boldsymbol{\theta}_1) - \delta(O_t, \boldsymbol{\theta}_2) = (\boldsymbol{\phi}(s_{t+1}) - \boldsymbol{\phi}(s_t))^\top(\boldsymbol{\omega}_1^* - \boldsymbol{\omega}_2^*) - (r(\boldsymbol{\theta}_1) - r(\boldsymbol{\theta}_2))$.

$$\begin{aligned}
\big\|h(O_t, \boldsymbol{\theta}_t) - h(O_t, \boldsymbol{\theta}_{t-\tau})\big\| &= \big\|\delta(O_t, \boldsymbol{\theta}_t)\nabla \log \pi_{\boldsymbol{\theta}_t}(a_t|s_t) - \delta(O_t, \boldsymbol{\theta}_{t-\tau})\nabla \log \pi_{\boldsymbol{\theta}_{t-\tau}}(a_t|s_t)\big\| \\
&\leq \big\|\delta(O_t, \boldsymbol{\theta}_t)\nabla \log \pi_{\boldsymbol{\theta}_t}(a_t|s_t) - \delta(O_t, \boldsymbol{\theta}_t)\nabla \log \pi_{\boldsymbol{\theta}_{t-\tau}}(a_t|s_t)\big\| \\
&\quad + \big\|\delta(O_t, \boldsymbol{\theta}_t)\nabla \log \pi_{\boldsymbol{\theta}_{t-\tau}}(a_t|s_t) - \delta(O_t, \boldsymbol{\theta}_{t-\tau})\nabla \log \pi_{\boldsymbol{\theta}_{t-\tau}}(a_t|s_t)\big\| \\
&\leq U_\delta L_l\|\boldsymbol{\theta}_t - \boldsymbol{\theta}_{t-\tau}\| + 2L_* B\|\boldsymbol{\theta}_t - \boldsymbol{\theta}_{t-\tau}\|.
\end{aligned}$$

By triangle inequality, we have

$$\begin{aligned}
\big|\Gamma(O_t, \boldsymbol{\theta}_t) - \Gamma(O_t, \boldsymbol{\theta}_{t-\tau})\big| &\leq G_{\boldsymbol{\theta}}\big\|h(O_t, \boldsymbol{\theta}_t) - h(O_t, \boldsymbol{\theta}_{t-\tau})\big\| + 3G_{\boldsymbol{\theta}}\big\|\nabla J(\boldsymbol{\theta}_t) - \nabla J(\boldsymbol{\theta}_{t-\tau})\big\| \\
&\leq G_{\boldsymbol{\theta}}(U_\delta L_l + 2L_* B + 3L_J)\|\boldsymbol{\theta}_t - \boldsymbol{\theta}_{t-\tau}\|.
\end{aligned}$$

$\square$

Lemma 5.8.2. *For any $t \geq 0$,*

$$\big|\mathbb{E}[\Gamma(O_t, \boldsymbol{\theta}_{t-\tau}) - \Gamma(\tilde{O}_t, \boldsymbol{\theta}_{t-\tau})]\big| \leq 2U_\delta B G_{\boldsymbol{\theta}} |\mathcal{A}| L \sum_{i=t-\tau}^{t} \|\boldsymbol{\theta}_i - \boldsymbol{\theta}_{t-\tau}\|.$$

Proof of Lemma 5.8.2. By the definition of in (5.7.1),

$$\begin{aligned}
\mathbb{E}\big[\Gamma(O_t, \boldsymbol{\theta}_{t-\tau}) - \Gamma(\tilde{O}_t, \boldsymbol{\theta}_{t-\tau})\big] &= \mathbb{E}\big[\big\langle \nabla J(\boldsymbol{\theta}_{t-\tau}), h(O_t, \boldsymbol{\theta}_{t-\tau}) - h(\tilde{O}_t, \boldsymbol{\theta}_{t-\tau})\big\rangle\big] \\
&= \mathbb{E}\Big[\big\langle \nabla J(\boldsymbol{\theta}_{t-\tau}), h(O_t, \boldsymbol{\theta}_{t-\tau})\big\rangle - \big\langle \nabla J(\boldsymbol{\theta}_{t-\tau}), h(\tilde{O}_t, \boldsymbol{\theta}_{t-\tau})\big\rangle\Big] \\
&\leq 4U_\delta B G_{\boldsymbol{\theta}} d_{TV}\big(\mathbb{P}(O_t = \cdot | s_{t-\tau+1}, \boldsymbol{\theta}_{t-\tau}), \mathbb{P}(\tilde{O}_t = \cdot | s_{t-\tau+1}, \boldsymbol{\theta}_{t-\tau})\big),
\end{aligned} \tag{5.8.1}$$

where the inequality is by the definition of total variation. By Lemma 5.6.2 we have

$$\begin{aligned}
& d_{TV}\big(\mathbb{P}(O_t \in \cdot | s_{t-\tau+1}, \boldsymbol{\theta}_{t-\tau}), \mathbb{P}(\tilde{O}_t \in \cdot | s_{t-\tau+1}, \boldsymbol{\theta}_{t-\tau})\big) \\
&= d_{TV}\big(\mathbb{P}((s_t, a_t) \in \cdot | s_{t-\tau+1}, \boldsymbol{\theta}_{t-\tau}), \mathbb{P}((\tilde{s}_t, \tilde{a}_t) \in \cdot | s_{t-\tau+1}, \boldsymbol{\theta}_{t-\tau})\big) \\
&\leq d_{TV}\big(\mathbb{P}(s_t \in \cdot | s_{t-\tau+1}, \boldsymbol{\theta}_{t-\tau}), \mathbb{P}(\tilde{s}_t \in \cdot | s_{t-\tau+1}, \boldsymbol{\theta}_{t-\tau})\big) + \frac{1}{2}|\mathcal{A}| L \mathbb{E}\|\boldsymbol{\theta}_t - \boldsymbol{\theta}_{t-\tau}\| \\
&\leq d_{TV}\big(\mathbb{P}(O_{t-1} \in \cdot | s_{t-\tau+1}, \boldsymbol{\theta}_{t-\tau}), \mathbb{P}(\tilde{O}_{t-1} \in \cdot | s_{t-\tau+1}, \boldsymbol{\theta}_{t-\tau})\big) + \frac{1}{2}|\mathcal{A}| L \mathbb{E}\|\boldsymbol{\theta}_t - \boldsymbol{\theta}_{t-\tau}\|.
\end{aligned}$$

Repeat the inequality above over t to $t - \tau + 1$ we have

$$d_{TV}\big(\mathbb{P}(O_t \in \cdot | s_{t-\tau+1}, \boldsymbol{\theta}_{t-\tau}), \mathbb{P}(\tilde{O}_t \in \cdot | s_{t-\tau+1}, \boldsymbol{\theta}_{t-\tau})\big) \leq \frac{1}{2}|\mathcal{A}| L \sum_{i=t-\tau}^{t} \mathbb{E}\|\boldsymbol{\theta}_i - \boldsymbol{\theta}_{t-\tau}\|. \tag{5.8.2}$$

Plugging (5.8.2) into (5.8.1) we get

$$\mathbb{E}\big[\Gamma(O_t, \boldsymbol{\theta}_{t-\tau}) - \Gamma(\tilde{O}_t, \boldsymbol{\theta}_{t-\tau})\big] \leq 2U_\delta B G_{\boldsymbol{\theta}} |\mathcal{A}| L \sum_{i=t-\tau}^{t} \|\boldsymbol{\theta}_i - \boldsymbol{\theta}_{t-\tau}\|.$$

□

Lemma 5.8.3. *For any $t \geq 0$, it holds $\big|\mathbb{E}[\Gamma(\tilde{O}_t, \boldsymbol{\theta}_{t-\tau}) - \Gamma(O'_t, \boldsymbol{\theta}_{t-\tau})]\big| \leq 4U_\delta B G_{\boldsymbol{\theta}} m \rho^{\tau-1}$.*

Proof of Lemma 5.8.3.

$$\begin{aligned}
\mathbb{E}\big[\Gamma\big(\tilde{O}_t, \boldsymbol{\theta}_{t-\tau}\big) - \Gamma(O'_t, \boldsymbol{\theta}_{t-\tau})\big] &\leq 4U_\delta B G_{\boldsymbol{\theta}} d_{TV}\big(\mathbb{P}(\tilde{O}_t = \cdot | s_{t-\tau+1}, \boldsymbol{\theta}_{t-\tau}), \mu_{\boldsymbol{\theta}_{t-\tau}} \otimes \pi_{\boldsymbol{\theta}_{t-\tau}} \otimes \mathcal{P}\big) \\
&\leq 4U_\delta B G_{\boldsymbol{\theta}} m \rho^{\tau-1}.
\end{aligned}$$

The first inequality is by the definition of total variation norm and the second inequality is shown in Lemma 5.8.11. □

Proof of Lemma 5.7.3. First note that

$$\begin{aligned}\delta &= \big|r(s,a) - J(\boldsymbol{\theta}) + \boldsymbol{\phi}^\top(s')\boldsymbol{\omega} - \boldsymbol{\phi}^\top(s)\boldsymbol{\omega}\big| \\ &\le \big|r(s,a)\big| + \big|J(\boldsymbol{\theta})\big| + \big|\boldsymbol{\phi}^\top(s')\boldsymbol{\omega}\big| + \big|\boldsymbol{\phi}^\top(s)\boldsymbol{\omega}\big| \\ &= 2U_r + 2R_{\boldsymbol{\omega}} \\ &=: U_\delta,\end{aligned}$$

which immediately implies

$$\big\|\delta \nabla \log \pi_{\boldsymbol{\theta}}(a|s)\big\| \le |\delta| \cdot \big\|\nabla \log \pi_{\boldsymbol{\theta}}(a|s)\big\| \le U_\delta \cdot B,$$

where the last inequality is due to Assumption 5.4.3. We decompose the Markovian bias as

$$\begin{aligned}\mathbb{E}[\Gamma(O_t,\boldsymbol{\theta}_t)] = \mathbb{E}[\Gamma(O_t,\boldsymbol{\theta}_t) - \Gamma(O_t,\boldsymbol{\theta}_{t-\tau})] + \mathbb{E}[\Gamma(O_t,\boldsymbol{\theta}_{t-\tau}) - \Gamma(\tilde{O}_t,\boldsymbol{\theta}_{t-\tau})] \\ + \mathbb{E}[\Gamma(\tilde{O}_t,\boldsymbol{\theta}_{t-\tau}) - \Gamma(O_t',\boldsymbol{\theta}_{t-\tau})] + \mathbb{E}[\Gamma(O_t',\boldsymbol{\theta}_{t-\tau})],\end{aligned}$$

where $\tilde{O}_t$ is from the auxiliary Markovian chain and O_t' is from the stationary distribution which actually satisfy $\mathbb{E}[\Gamma(O_t',\boldsymbol{\theta}_{t-\tau})] = 0$. By collecting the corresponding bounds from Lemmas 5.8.1, 5.8.2 and 5.8.3, we have that

$$\begin{aligned}\mathbb{E}[\Gamma(O_t,\boldsymbol{\theta}_t)] &\ge -G_{\boldsymbol{\theta}}(U_\delta L_l + 2L_*B + 3L_J)\mathbb{E}\|\boldsymbol{\theta}_t - \boldsymbol{\theta}_{t-\tau}\| - 2U_\delta B G_{\boldsymbol{\theta}}|\mathcal{A}|L\sum_{i=t-\tau}^{t}\mathbb{E}\|\boldsymbol{\theta}_i - \boldsymbol{\theta}_{t-\tau}\| \\ &\quad - 4U_\delta B G_{\boldsymbol{\theta}} m\rho^{\tau-1} \\ &\ge -G_{\boldsymbol{\theta}}(U_\delta L_l + 2L_*B + 3L_J)\sum_{i=t-\tau+1}^{t}\mathbb{E}\|\boldsymbol{\theta}_i - \boldsymbol{\theta}_{i-1}\| \\ &\quad - 2U_\delta B G_{\boldsymbol{\theta}}|\mathcal{A}|L\sum_{i=t-\tau+1}^{t}\sum_{j=t-\tau+1}^{i}\mathbb{E}\|\boldsymbol{\theta}_j - \boldsymbol{\theta}_{j-1}\| - 4U_\delta B G_{\boldsymbol{\theta}} m\rho^{\tau-1} \\ &\ge -G_{\boldsymbol{\theta}}(U_\delta L_l + 2L_*B + 3L_J)\sum_{i=t-\tau+1}^{t}\mathbb{E}\|\boldsymbol{\theta}_i - \boldsymbol{\theta}_{i-1}\| \\ &\quad - 2U_\delta B G_{\boldsymbol{\theta}}|\mathcal{A}|L\tau\sum_{j=t-\tau+1}^{t}\mathbb{E}\|\boldsymbol{\theta}_j - \boldsymbol{\theta}_{j-1}\| - 4U_\delta B G_{\boldsymbol{\theta}} m\rho^{\tau-1} \\ &\ge -G_{\boldsymbol{\theta}}\bigg(D_1(\tau+1)\sum_{k=t-\tau+1}^{t}\mathbb{E}\|\boldsymbol{\theta}_k - \boldsymbol{\theta}_{k-1}\| + D_2 m\rho^{\tau-1}\bigg),\end{aligned}$$

where $D_1 := \max\{(U_\delta L_l + 2L_*B + 3L_J), 2U_\delta B|\mathcal{A}|L\}$ and $D_2 := 4U_\delta B$, which completes the proof. □

5.8.4 Proof of Lemma 5.7.4

Proof of Lemma 5.7.4. By definition, we have

$$J(\boldsymbol{\theta}_1) - J(\boldsymbol{\theta}_2) = \mathbb{E}[r(s^{(1)}, a^{(1)}) - r(s^{(2)}, a^{(2)})],$$

where $s^{(i)} \sim \mu_{\boldsymbol{\theta}_i}, a^{(i)} \sim \pi_{\boldsymbol{\theta}_i}$. Therefore, it holds that

$$\begin{aligned} J(\boldsymbol{\theta}_1) - J(\boldsymbol{\theta}_2) &= \mathbb{E}[r(s^{(1)}, a^{(1)}) - r(s^{(2)}, a^{(2)})] \\ &\le 2U_r d_{TV}(\mu_{\boldsymbol{\theta}_1} \otimes \pi_{\boldsymbol{\theta}_1}, \mu_{\boldsymbol{\theta}_2} \otimes \pi_{\boldsymbol{\theta}_2}) \\ &\le 2U_r |\mathcal{A}| L \bigg(1 + \lceil \log_\rho m^{-1} \rceil + \frac{1}{1-\rho}\bigg) \|\boldsymbol{\theta}_1 - \boldsymbol{\theta}_2\| \\ &= C_J \|\boldsymbol{\theta}_1 - \boldsymbol{\theta}_2\|. \end{aligned}$$

□

5.8.5 Proof of Lemma 5.7.5

The proof of this lemma depends on several auxiliary lemmas as follows.

Lemma 5.8.4. *For any $\boldsymbol{\theta}_1, \boldsymbol{\theta}_2, eta, O = (s, a, s')$, we have*

$$\big|\Xi(O, \eta, \boldsymbol{\theta}_1) - \Xi(O, \eta, \boldsymbol{\theta}_2)\big| \le 4U_r C_J \|\boldsymbol{\theta}_1 - \boldsymbol{\theta}_2\|.$$

Proof of Lemma 5.8.4. By the definition of $\Xi(O, \eta, \boldsymbol{\theta})$ in (5.7.5), we have

$$\begin{aligned} \big|\Xi(O, \eta, \boldsymbol{\theta}_1) - \Xi(O, \eta, \boldsymbol{\theta}_2)\big| &= \big|(\eta - \eta_1^*)(r - \eta_1^*) - (\eta - \eta_2^*)(r - \eta_2^*)\big| \\ &\le \big|(\eta - \eta_1^*)(r - \eta_1^*) - (\eta - \eta_1^*)(r - \eta_2^*)\big| \\ &\quad + \big|(\eta - \eta_1^*)(r - \eta_2^*) - (\eta - \eta_2^*)(r - \eta_2^*)\big| \\ &\le 4U_r |\eta_1^* - \eta_2^*| \\ &= 4U_r \big|J(\boldsymbol{\theta}_1) - J(\boldsymbol{\theta}_2)\big| \\ &\le 4U_r C_J \|\boldsymbol{\theta}_1 - \boldsymbol{\theta}_2\|. \end{aligned}$$

□

Lemma 5.8.5. *For any $\boldsymbol{\theta}, \eta_1, \eta_2, O$, we have*

$$\big|\Xi(O, \eta_1, \boldsymbol{\theta}) - \Xi(O, \eta_2, \boldsymbol{\theta})\big| \le 2U_r|\eta_1 - \eta_2|.$$

Proof of Lemma 5.8.5. By definition,

$$\begin{aligned}\big|\Xi(O, \eta_1, \boldsymbol{\theta}) - \Xi(O, \eta_2, \boldsymbol{\theta})\big| &= \big|(\eta_1 - \eta^*)(r - \eta^*) - (\eta_2 - \eta^*)(r - \eta^*)\big| \\ &\le 2U_r|\eta_1 - \eta_2|.\end{aligned}$$

□

Lemma 5.8.6. *Consider original tuples $O_t = (s_t, a_t, s_{t+1})$ and the auxiliary tuples $\tilde{O}_t = (\tilde{s}_t, \tilde{a}_t, \tilde{s}_{t+1})$. Conditioned on $s_{t-\tau+1}$ and $\boldsymbol{\theta}_{t-\tau}$, we have*

$$\big|\mathbb{E}[\Xi(O_t, \eta_{t-\tau}, \boldsymbol{\theta}_{t-\tau}) - \Xi(\tilde{O}_t, \eta_{t-\tau}, \boldsymbol{\theta}_{t-\tau})]\big| \le 2U_r^2|\mathcal{A}|L \sum_{i=t-\tau}^{t} \mathbb{E}\|\boldsymbol{\theta}_i - \boldsymbol{\theta}_{t-\tau}\|.$$

Proof of Lemma 5.8.6. By the Cauchy-Schwartz inequality and the definition of total variation norm, we have

$$\mathbb{E}\big[\Xi(O_t, \eta_{t-\tau}, \boldsymbol{\theta}_{t-\tau}) - \Xi(\tilde{O}_t, \eta_{t-\tau}, \boldsymbol{\theta}_{t-\tau})\big] = (\eta_{t-\tau} - \eta^*_{t-\tau})\mathbb{E}[r(s_t, a_t) - r(\tilde{s}_t, \tilde{a}_t)].$$

Since

$$\mathbb{E}[r(s_t, a_t) - r(\tilde{s}_t, \tilde{a}_t)] \le 2U_r d_{TV}\big(\mathbb{P}(O_t \in \cdot|s_{t-\tau+1}, \boldsymbol{\theta}_{t-\tau}), \mathbb{P}(\tilde{O}_t \in \cdot|s_{t-\tau+1}, \boldsymbol{\theta}_{t-\tau})\big),$$

the total variation between O_t and $\tilde{O}_t$ has appeared in (5.8.2), in the proof of Lemma 5.8.2, which is

$$d_{TV}\big(\mathbb{P}(O_t \in \cdot|s_{t-\tau+1}, \boldsymbol{\theta}_{t-\tau}), \mathbb{P}(\tilde{O}_t \in \cdot|s_{t-\tau+1}, \boldsymbol{\theta}_{t-\tau})\big) \le \frac{1}{2}|\mathcal{A}|L \sum_{i=t-\tau}^{t} \mathbb{E}\|\boldsymbol{\theta}_i - \boldsymbol{\theta}_{t-\tau}\|.$$

Plugging this bound, we have

$$\big|\mathbb{E}[\Xi(O_t, \eta_{t-\tau}, \boldsymbol{\theta}_{t-\tau}) - \Xi(\tilde{O}_t, \eta_{t-\tau}, \boldsymbol{\theta}_{t-\tau})]\big| \le 2U_r^2|\mathcal{A}|L \sum_{i=t-\tau}^{t} \mathbb{E}\|\boldsymbol{\theta}_i - \boldsymbol{\theta}_{t-\tau}\|.$$

□

Lemma 5.8.7. *Conditioned on $s_{t-\tau+1}$ and $\boldsymbol{\theta}_{t-\tau}$, we have $\mathbb{E}[\Xi(\tilde{O}_t, \eta_{t-\tau}, \boldsymbol{\theta}_{t-\tau})] \le 4U_r^2 m\rho^{\tau-1}$.*

Proof of Lemma 5.8.7. We first note that according to the definition,

$$\mathbb{E}[\eta(O'_t, \eta_{t-\tau}, \boldsymbol{\theta}_{t-\tau})|\boldsymbol{\theta}_{t-\tau}] = 0,$$

where $O'_t = (s'_t, a'_t, s'_{t+1})$ is the tuple generated by $s'_t \sim \mu_{\boldsymbol{\theta}_{t-\tau}}, a'_t \sim \pi_{\boldsymbol{\theta}_{t-\tau}}, s'_{t+1} \sim \mathcal{P}$. By the ergodicity in Assumption 5.4.2, it holds that $d_{TV}\big(\mathbb{P}(\tilde{s}_t = \cdot|s_{t-\tau+1}, \boldsymbol{\theta}_{t-\tau}), \mu_{\boldsymbol{\theta}_{t-\tau}}\big) \leq m\rho^{\tau-1}$. It can be shown that

$$\begin{aligned}
\mathbb{E}[\Xi(\tilde{O}_t, \eta_{t-\tau}, \boldsymbol{\theta}_{t-\tau})] &= \mathbb{E}\big[\Xi\big(\tilde{O}_t, \eta_{t-\tau}, \boldsymbol{\theta}_{t-\tau}\big) - \Xi(O'_t, \eta_{t-\tau}, \boldsymbol{\theta}_{t-\tau})\big] \\
&= \mathbb{E}\big[(\eta_{t-\tau} - \eta^*_{t-\tau})\big(r(\tilde{s}_t, \tilde{a}_t) - r(s', a')\big)\big] \\
&\leq 4U_r^2 d_{TV}\big(\mathbb{P}\big(\tilde{O}_t = \cdot|s_{t-\tau+1}, \boldsymbol{\theta}_{t-\tau}\big), \mu_{\boldsymbol{\theta}_{t-\tau}} \otimes \pi_{\boldsymbol{\theta}_{t-\tau}} \otimes \mathcal{P}\big) \\
&\leq 4U_r^2 m\rho^{\tau-1}.
\end{aligned}$$

The argument used here is the same as that in the proof of Lemma 5.8.11. □

Proof of Lemma 5.7.5. By the Lemma 5.8.4, 5.8.5, 5.8.6 and 5.8.7, we can collect the corresponding term and get the bound

$$\begin{aligned}
\mathbb{E}[\Xi(O_t, \eta_t, \boldsymbol{\theta}_t)] &= \mathbb{E}[\Xi(O_t, \eta_t, \boldsymbol{\theta}_t) - \Xi(O_t, \eta_t, \boldsymbol{\theta}_{t-\tau})] + \mathbb{E}[\Xi(O_t, \eta_t, \boldsymbol{\theta}_{t-\tau}) - \Xi(O_t, \eta_{t-\tau}, \boldsymbol{\theta}_{t-\tau})] \\
&\quad + \mathbb{E}[\Xi(O_t, \eta_{t-\tau}, \boldsymbol{\theta}_{t-\tau}) - \Xi(\tilde{O}_t, \eta_{t-\tau}, \boldsymbol{\theta}_{t-\tau})] + \mathbb{E}[\Xi(\tilde{O}_t, \eta_{t-\tau}, \boldsymbol{\theta}_{t-\tau})] \\
&\leq 4U_r C_J\|\boldsymbol{\theta}_t - \boldsymbol{\theta}_{t-\tau}\| + 2U_r|\eta_t - \eta_{t-\tau}| + 2U_r^2|\mathcal{A}|L\sum_{i=t-\tau}^{t}\mathbb{E}\|\boldsymbol{\theta}_i - \boldsymbol{\theta}_{t-\tau}\| \\
&\quad + 4U_r^2 m\rho^{\tau-1}.
\end{aligned}$$

□

5.8.6 Proof of Lemma 5.7.6

Proof of Lemma 5.7.6. For the first inequality, apply the property of norm and the Cauchy-Schwartz inequality:

$$\begin{aligned}
\big\|g(O, \boldsymbol{\omega}, \boldsymbol{\theta})\big\| &= \big\|(r(s,a) - J(\boldsymbol{\theta}) + \boldsymbol{\phi}^\top(s')\boldsymbol{\omega} - \boldsymbol{\phi}^\top(s)\boldsymbol{\omega})\boldsymbol{\phi}(s)\big\| \\
&\leq \big|r(s,a)\big| + \big\|J(\boldsymbol{\theta})\big\| + \big|\boldsymbol{\phi}^\top(s')\boldsymbol{\omega}\big| \cdot \big\|\boldsymbol{\phi}^\top(s)\big\| + \big|\boldsymbol{\phi}^\top(s)\boldsymbol{\omega}\big| \cdot \big\|\boldsymbol{\phi}^\top(s)\big\|
\end{aligned}$$

$$= U_r + U_r + R_{\boldsymbol{\omega}} + R_{\boldsymbol{\omega}} \le 2U_r + 2R_{\boldsymbol{\omega}}.$$

For the second inequality, we can directly apply Cauchy-Schwartz inequality and obtain the result. For the third inequality, apply Cauchy-Schwartz inequality as we have

$$\begin{aligned}
\big|\Lambda(O,\boldsymbol{\omega},\boldsymbol{\theta})\big| &= \Big|\big\langle \boldsymbol{\omega}-\boldsymbol{\omega}^*, g(O,\boldsymbol{\omega},\boldsymbol{\theta}) - \bar{g}(\boldsymbol{\omega},\boldsymbol{\theta})\big\rangle\Big| \\
&\le \|\boldsymbol{\omega}-\boldsymbol{\omega}^*\| \cdot \big\|g(O,\boldsymbol{\omega},\boldsymbol{\theta}) - \bar{g}(\boldsymbol{\omega},\boldsymbol{\theta})\big\| \\
&\le 2R_{\boldsymbol{\omega}} \cdot 2U_\delta \le 2U_\delta^2,
\end{aligned}$$

which completes the proof. □

5.8.7 Proof of Lemma 5.7.7

This Lemma is actually a combination of several auxiliary lemmas listed here:

Lemma 5.8.8. *For any $\boldsymbol{\theta}_1, \boldsymbol{\theta}_2$, $\boldsymbol{\omega}$ and tuple $O = (s, a, s')$,*

$$\big|\Lambda(O,\boldsymbol{\omega},\boldsymbol{\theta}_1) - \Lambda(O,\boldsymbol{\omega},\boldsymbol{\theta}_2)\big| \le K_1\|\boldsymbol{\theta}_1 - \boldsymbol{\theta}_2\|,$$

where $K_1 = 2U_\delta^2|\mathcal{A}|L(1 + \lceil \log_\rho m^{-1}\rceil + 1/(1-\rho)) + 2U_\delta L_*$.

Proof of Lemma 5.8.8.

$$\begin{aligned}
&\big|\Lambda(O,\boldsymbol{\omega},\boldsymbol{\theta}_1) - \Lambda(O,\boldsymbol{\omega},\boldsymbol{\theta}_2)\big| \\
&= \Big|\big\langle \boldsymbol{\omega}-\boldsymbol{\omega}_1^*, g(O,\boldsymbol{\omega}) - \bar{g}(\boldsymbol{\theta}_1,\boldsymbol{\omega})\big\rangle - \big\langle \boldsymbol{\omega}-\boldsymbol{\omega}_2^*, g(O,\boldsymbol{\omega}) - \bar{g}(\boldsymbol{\theta}_2,\boldsymbol{\omega})\big\rangle\Big| \\
&\le \underbrace{\Big|\big\langle \boldsymbol{\omega}-\boldsymbol{\omega}_1^*, g(O,\boldsymbol{\omega}) - \bar{g}(\boldsymbol{\theta}_1,\boldsymbol{\omega})\big\rangle - \big\langle \boldsymbol{\omega}-\boldsymbol{\omega}_1^*, g(O,\boldsymbol{\omega}) - \bar{g}(\boldsymbol{\theta}_2,\boldsymbol{\omega})\big\rangle\Big|}_{I_1} \\
&\quad + \underbrace{\Big|\big\langle \boldsymbol{\omega}-\boldsymbol{\omega}_1^*, g(O,\boldsymbol{\omega}) - \bar{g}(\boldsymbol{\theta}_2,\boldsymbol{\omega})\big\rangle - \big\langle \boldsymbol{\omega}-\boldsymbol{\omega}_2^*, g(O,\boldsymbol{\omega}) - \bar{g}(\boldsymbol{\theta}_2,\boldsymbol{\omega})\big\rangle\Big|}_{I_2}.
\end{aligned}$$

For the term I_2, we simply use the Cauchy-Schwartz inequality to get $2U_\delta\|\boldsymbol{\omega}_1^* - \boldsymbol{\omega}_2^*\|$. For the term I_1, it can be bounded as:

$$\Big|\big\langle \boldsymbol{\omega}-\boldsymbol{\omega}_1^*, g(O,\boldsymbol{\omega}) - \bar{g}(\boldsymbol{\theta}_1,\boldsymbol{\omega})\big\rangle - \big\langle \boldsymbol{\omega}-\boldsymbol{\omega}_1^*, g(O,\boldsymbol{\omega}) - \bar{g}(\boldsymbol{\theta}_2,\boldsymbol{\omega})\big\rangle\Big|$$

$$
\begin{aligned}
&= \Big|\big\langle \boldsymbol{\omega} - \boldsymbol{\omega}_1^*, \bar{g}(\boldsymbol{\theta}_1, \boldsymbol{\omega}) - \bar{g}(\boldsymbol{\theta}_2, \boldsymbol{\omega})\big\rangle\Big| \\
&\le 2R_{\boldsymbol{\omega}} \big\|\bar{g}(\boldsymbol{\theta}_1, \boldsymbol{\omega}) - \bar{g}(\boldsymbol{\theta}_2, \boldsymbol{\omega})\big\| \\
&\le 2R_{\boldsymbol{\omega}} \cdot 2U_\delta \cdot d_{TV}(\mu_{\boldsymbol{\theta}_1} \otimes \pi_{\boldsymbol{\theta}_1} \otimes \mathcal{P}, \mu_{\boldsymbol{\theta}_2} \otimes \pi_{\boldsymbol{\theta}_2} \otimes \mathcal{P}) \\
&\le 2U_\delta^2 d_{TV}(\mu_{\boldsymbol{\theta}_1} \otimes \pi_{\boldsymbol{\theta}_1} \otimes \mathcal{P}, \mu_{\boldsymbol{\theta}_2} \otimes \pi_{\boldsymbol{\theta}_2} \otimes \mathcal{P}),
\end{aligned}
$$

where the first inequality is due to Cauchy-Schwartz; the second inequality is by the definition of total variation norm; the third inequality is due to the fact $U_\delta \ge 2R_{\boldsymbol{\omega}}$. Therefore, we have

$$
\begin{aligned}
&\big|\Lambda(\boldsymbol{\theta}_1, \boldsymbol{\omega}, O) - \Lambda(\boldsymbol{\theta}_2, \boldsymbol{\omega}, O)\big| \\
&\le 2U_\delta^2 d_{TV}(\mu_{\boldsymbol{\theta}_1} \otimes \pi_{\boldsymbol{\theta}_1} \otimes \mathcal{P}, \mu_{\boldsymbol{\theta}_2} \otimes \pi_{\boldsymbol{\theta}_2} \otimes \mathcal{P}) + 2U_\delta \|\boldsymbol{\omega}_1^* - \boldsymbol{\omega}_2^*\| \\
&\le 2U_\delta^2 |\mathcal{A}| L \bigg(1 + \lceil \log_\rho m^{-1} \rceil + \frac{1}{1-\rho}\bigg) \|\boldsymbol{\theta}_1 - \boldsymbol{\theta}_2\| + 2U_\delta L_* \|\boldsymbol{\theta}_1 - \boldsymbol{\theta}_2\| \\
&= K_1 \|\boldsymbol{\theta}_1 - \boldsymbol{\theta}_2\|,
\end{aligned}
$$

where the second inequality is due to Lemma 5.6.1 and Proposition 5.4.4. □

Lemma 5.8.9. *For any $\boldsymbol{\theta}$, $\boldsymbol{\omega}_1, \boldsymbol{\omega}_2$ and tuple $O = (s, a, s')$,*

$$
\big|\Lambda(O, \boldsymbol{\omega}_1, \boldsymbol{\theta}) - \Lambda(O, \boldsymbol{\omega}_2, \boldsymbol{\theta})\big| \le 6U_\delta \|\boldsymbol{\omega}_1 - \boldsymbol{\omega}_2\|.
$$

Proof of Lemma 5.8.9. By definition,

$$
\begin{aligned}
&\big|\Lambda(O, \boldsymbol{\omega}_1, \boldsymbol{\theta}) - \Lambda(O, \boldsymbol{\omega}_2, \boldsymbol{\theta})\big| \\
&= \Big|\big\langle \boldsymbol{\omega}_1 - \boldsymbol{\omega}^*, g(O, \boldsymbol{\omega}_1) - \bar{g}(\boldsymbol{\omega}_1, \boldsymbol{\theta})\big\rangle - \big\langle \boldsymbol{\omega}_2 - \boldsymbol{\omega}^*, g(O, \boldsymbol{\omega}_2) - \bar{g}(\boldsymbol{\omega}_2, \boldsymbol{\theta})\big\rangle\Big| \\
&\le \Big|\big\langle \boldsymbol{\omega}_1 - \boldsymbol{\omega}^*, g(O, \boldsymbol{\omega}_1) - \bar{g}(\boldsymbol{\omega}_1, \boldsymbol{\theta})\big\rangle - \big\langle \boldsymbol{\omega}_1 - \boldsymbol{\omega}^*, g(O, \boldsymbol{\omega}_2) - \bar{g}(\boldsymbol{\omega}_2, \boldsymbol{\theta})\big\rangle\Big| \\
&\qquad + \Big|\big\langle \boldsymbol{\omega}_1 - \boldsymbol{\omega}^*, g(O, \boldsymbol{\omega}_2) - \bar{g}(\boldsymbol{\omega}_2, \boldsymbol{\theta})\big\rangle - \big\langle \boldsymbol{\omega}_2 - \boldsymbol{\omega}^*, g(O, \boldsymbol{\omega}_2) - \bar{g}(\boldsymbol{\omega}_2, \boldsymbol{\theta})\big\rangle\Big| \\
&\le 2R_{\boldsymbol{\omega}} \Big\|\big(g(O, \boldsymbol{\omega}_1) - g(O, \boldsymbol{\omega}_2)\big) - \big(\bar{g}(\boldsymbol{\omega}_1, \boldsymbol{\theta}) - \bar{g}(\boldsymbol{\omega}_2, \boldsymbol{\theta})\big)\Big\| + 2U_\delta \|\boldsymbol{\omega}_1 - \boldsymbol{\omega}_2\|.
\end{aligned}
$$

Note that we have $\|g(O, \boldsymbol{\omega}_1, \boldsymbol{\theta}) - g(O, \boldsymbol{\omega}_2, \boldsymbol{\theta})\| = |(\boldsymbol{\phi}(s') - \boldsymbol{\phi}(s))^\top (\boldsymbol{\omega}_1 - \boldsymbol{\omega}_2)| \le 2\|\boldsymbol{\omega}_1 - \boldsymbol{\omega}_2\|$ and similarly $\|\bar{g}(\boldsymbol{\omega}_1, \boldsymbol{\theta}) - \bar{g}(\boldsymbol{\omega}_2, \boldsymbol{\theta})\| \le |\mathbb{E}\big[(\boldsymbol{\phi}(s') - \boldsymbol{\phi}(s))^\top (\boldsymbol{\omega}_1 - \boldsymbol{\omega}_2)\big]| \le 2\|\boldsymbol{\omega}_1 - \boldsymbol{\omega}_2\|$. Therefore,

$$
\big|\Lambda(O, \boldsymbol{\omega}_1, \boldsymbol{\theta}) - \Lambda(O, \boldsymbol{\omega}_2, \boldsymbol{\theta})\big|
$$

$$\leq 2R_{\boldsymbol{\omega}}\Big\|\big(g(O,\boldsymbol{\omega}_1)-g(O,\boldsymbol{\omega}_2)\big)-\big(\bar{g}(\boldsymbol{\omega}_1,\boldsymbol{\theta})-\bar{g}(\boldsymbol{\omega}_2,\boldsymbol{\theta})\big)\Big\|+2U_\delta\|\boldsymbol{\omega}_1-\boldsymbol{\omega}_2\|$$
$$\leq 2R_{\boldsymbol{\omega}}\cdot 4\|\boldsymbol{\omega}_1-\boldsymbol{\omega}_2\|+2U_\delta\|\boldsymbol{\omega}_1-\boldsymbol{\omega}_2\|$$
$$\leq 6U_\delta\|\boldsymbol{\omega}_1-\boldsymbol{\omega}_2\|.$$

□

Lemma 5.8.10. *Consider original tuples $O_t=(s_t,a_t,s_{t+1})$ and the auxiliary tuples $\tilde{O}_t=(\tilde{s}_t,\tilde{a}_t,\tilde{s}_{t+1})$. Conditioned on $s_{t-\tau+1}$ and $\boldsymbol{\theta}_{t-\tau}$, we have*

$$\mathbb{E}[\Lambda(O_t,\boldsymbol{\omega}_{t-\tau},\boldsymbol{\theta}_{t-\tau})-\Lambda(\tilde{O}_t,\boldsymbol{\omega}_{t-\tau},\boldsymbol{\theta}_{t-\tau})]\leq U_\delta^2|\mathcal{A}|L\sum_{i=t-\tau}^{t}\mathbb{E}\|\boldsymbol{\theta}_i-\boldsymbol{\theta}_{t-\tau}\| \tag{5.8.3}$$

Proof of Lemma 5.8.10. By the Cauchy-Schwartz inequality and the definition of total variation norm, we have

$$\begin{aligned}&\mathbb{E}[\Lambda(O_t,\boldsymbol{\omega}_{t-\tau},\boldsymbol{\theta}_{t-\tau})-\Lambda(\tilde{O}_t,\boldsymbol{\omega}_{t-\tau},\boldsymbol{\theta}_{t-\tau})] &(5.8.4)\\ &=\mathbb{E}\big[\big\langle\boldsymbol{\omega}_{t-\tau}-\boldsymbol{\omega}^*_{t-\tau},g(O_t,\boldsymbol{\omega}_{t-\tau})-g(\tilde{O}_t,\boldsymbol{\omega}_{t-\tau})\big\rangle\big]\\ &\leq 2U_\delta^2 d_{TV}\big(\mathbb{P}(O_t\in\cdot|s_{t-\tau+1},\boldsymbol{\theta}_{t-\tau}),\mathbb{P}(\tilde{O}_t\in\cdot|s_{t-\tau+1},\boldsymbol{\theta}_{t-\tau})\big). &(5.8.5)\end{aligned}$$

The total variation between O_t and $\tilde{O}_t$ has appeared in (5.8.2), in the proof of Lemma 5.8.2, which is

$$d_{TV}\big(\mathbb{P}(O_t\in\cdot|s_{t-\tau+1},\boldsymbol{\theta}_{t-\tau}),\mathbb{P}(\tilde{O}_t\in\cdot|s_{t-\tau+1},\boldsymbol{\theta}_{t-\tau})\big)\leq\frac{1}{2}|\mathcal{A}|L\sum_{i=t-\tau}^{t}\mathbb{E}\|\boldsymbol{\theta}_i-\boldsymbol{\theta}_{t-\tau}\|.$$

Plugging this bound into (5.8.5), we have

$$\mathbb{E}\big|\Lambda(O_t,\boldsymbol{\omega}_{t-\tau},\boldsymbol{\theta}_{t-\tau})-\Lambda(\tilde{O}_t,\boldsymbol{\omega}_{t-\tau},\boldsymbol{\theta}_{t-\tau})\big|\leq U_\delta^2|\mathcal{A}|L\sum_{i=t-\tau}^{t}\mathbb{E}\|\boldsymbol{\theta}_i-\boldsymbol{\theta}_{t-\tau}\|.$$

□

Lemma 5.8.11. *Conditioned on $s_{t-\tau+1}$ and $\boldsymbol{\theta}_{t-\tau}$, we have $\mathbb{E}[\Lambda(\tilde{O}_t,\boldsymbol{\omega}_{t-\tau},\boldsymbol{\theta}_{t-\tau})]\leq 2U_\delta^2 m\rho^{\tau-1}$.*

Proof of Lemma 5.8.11. We first note that according to the definition in Section 5.7.3,

$$\mathbb{E}[\Lambda(O'_t,\boldsymbol{\omega}_{t-\tau},\boldsymbol{\theta}_{t-\tau})|s_{t-\tau+1},\boldsymbol{\theta}_{t-\tau}]=0,$$

where $O'_t = (s'_t, a'_t, s'_{t+1})$ is the tuple generated by $s'_t \sim \mu_{\boldsymbol{\theta}_{t-\tau}}, a'_t \sim \pi_{\boldsymbol{\theta}_{t-\tau}}, s'_{t+1} \sim \mathcal{P}$. By the ergodicity in Assumption 5.4.2, it holds that $d_{TV}\big(\mathbb{P}(\tilde{s}_t = \cdot|s_{t-\tau+1}, \boldsymbol{\theta}_{t-\tau}), \mu_{\boldsymbol{\theta}_{t-\tau}}\big) \le m\rho^{\tau-1}$. It can be shown that

$$\begin{aligned}
\mathbb{E}[\Lambda(\tilde{O}_t, \boldsymbol{\omega}_{t-\tau}, \boldsymbol{\theta}_{t-\tau})] &= \mathbb{E}[\Lambda(\tilde{O}_t, \boldsymbol{\omega}_{t-\tau}, \boldsymbol{\theta}_{t-\tau}) - \Lambda(O'_t, \boldsymbol{\omega}_{t-\tau}, \boldsymbol{\theta}_{t-\tau})] \\
&= \mathbb{E}\big\langle \boldsymbol{\omega}_{t-\tau} - \boldsymbol{\omega}^*_{t-\tau}, g(\tilde{O}_t, \boldsymbol{\omega}_{t-\tau}) - g(O'_t, \boldsymbol{\omega}_{t-\tau})\big\rangle \\
&\le 4R_{\boldsymbol{\omega}} U_\delta d_{TV}\big(\mathbb{P}(\tilde{O}_t = \cdot|s_{t-\tau+1}, \boldsymbol{\theta}_{t-\tau}), \mu_{\boldsymbol{\theta}_{t-\tau}} \otimes \pi_{\boldsymbol{\theta}_{t-\tau}} \otimes \mathcal{P}\big) \\
&\le 2U_\delta^2 d_{TV}\big(\mathbb{P}(\tilde{s}_t = \cdot|s_{t-\tau+1}, \boldsymbol{\theta}_{t-\tau}), \mu_{\boldsymbol{\theta}_{t-\tau}}\big) \\
&\le 2U_\delta^2 m\rho^{\tau-1}.
\end{aligned}$$

The third inequality holds because $2R_{\boldsymbol{\omega}} < U_\delta$ and

$$\begin{aligned}
&d_{TV}\big(\mathbb{P}(\tilde{O}_t = \cdot|s_{t-\tau+1}, \boldsymbol{\theta}_{t-\tau}), \mu_{\boldsymbol{\theta}_{t-\tau}} \otimes \pi_{\boldsymbol{\theta}_{t-\tau}} \otimes \mathcal{P}\big) \\
&= d_{TV}\big(\mathbb{P}((\tilde{s}_t, \tilde{a}_t) = \cdot|s_{t-\tau+1}, \boldsymbol{\theta}_{t-\tau}), \mu_{\boldsymbol{\theta}_{t-\tau}} \otimes \pi_{\boldsymbol{\theta}_{t-\tau}}\big) \\
&= d_{TV}\big(\mathbb{P}(\tilde{s}_t = \cdot|s_{t-\tau+1}, \boldsymbol{\theta}_{t-\tau}), \mu_{\boldsymbol{\theta}_{t-\tau}}\big).
\end{aligned}$$

This can be shown following the same procedure in (5.6.1), because $\mathbb{P}(\tilde{O}_t = \cdot|s_{t-\tau+1}, \boldsymbol{\theta}_{t-\tau}) = \mathbb{P}(\tilde{s}_t = \cdot|s_{t-\tau+1}, \boldsymbol{\theta}_{t-\tau}) \otimes \pi_{\boldsymbol{\theta}_{t-\tau}} \otimes \mathcal{P}$. □

Proof of Lemma 5.7.7. By the Lemma 5.8.8, 5.8.9, 5.8.10 and 5.8.11, we can collect the corresponding term and get the bound

$$\begin{aligned}
\mathbb{E}[\Lambda(O_t, \boldsymbol{\omega}_t, \boldsymbol{\theta}_t)] &= \mathbb{E}[\Lambda(O_t, \boldsymbol{\omega}_t, \boldsymbol{\theta}_t) - \Lambda(O_t, \boldsymbol{\omega}_t, \boldsymbol{\theta}_{t-\tau})] + \mathbb{E}[\Lambda(O_t, \boldsymbol{\omega}_t, \boldsymbol{\theta}_{t-\tau}) - \Lambda(O_t, \boldsymbol{\omega}_{t-\tau}, \boldsymbol{\theta}_{t-\tau})] \\
&\quad + \mathbb{E}[\Lambda(O_t, \boldsymbol{\omega}_{t-\tau}, \boldsymbol{\theta}_{t-\tau}) - \Lambda(\tilde{O}_t, \boldsymbol{\omega}_{t-\tau}, \boldsymbol{\theta}_{t-\tau})] + \mathbb{E}[\Lambda(\tilde{O}_t, \boldsymbol{\omega}_{t-\tau}, \boldsymbol{\theta}_{t-\tau})] \\
&\le C_1(\tau+1)\|\boldsymbol{\theta}_t - \boldsymbol{\theta}_{t-\tau}\| + C_2 m\rho^{\tau-1} + C_3\|\boldsymbol{\omega}_t - \boldsymbol{\omega}_{t-\tau}\|,
\end{aligned}$$

where $C_1 = 2U_\delta^2|\mathcal{A}|L(1 + \lceil \log_\rho m^{-1} \rceil + 1/(1-\rho)) + 2U_\delta L_*, C_2 = 2U_\delta^2, C_3 = 4U_\delta$. □

CHAPTER 6

Q-Learning with Deep Neural Network Function Approximation

6.1 Introduction

Apart from policy gradient methods and actor-critic methods, Q-learning has been another important and effective learning strategies in Reinforcement Learning (RL) over the past decades [WD92, Sch15, SB18]. Different from policy gradient methods in Chapter 4 and actor-critic methods in Chapter 5 which directly optimizes the parameterized policy, Q-learning estimates an action-value function (a.k.a., Q-value function) for all action-state pairs and the agent takes an action based on the Q-value of actions at the current state. Recent advance in deep learning has also enabled the application of Q-learning algorithms to large-scale decision problems such as mastering Go [SHM$^+$16, SSS$^+$17], robotic motion control [LWA15, KIP$^+$18] and autonomous driving [SSSS16, SAMR18]. In particular, the seminal work by [MKS$^+$15] introduced the Deep Q-Network (DQN) to approximate the action-value function and achieved a superior performance versus a human expert in playing Atari games, which triggers a line of research on deep reinforcement learning such as Double Deep Q-Learning [VHGS16] and Dueling DQN [WSH$^+$16].

Apart from its widespread empirical success in numerous applications, the convergence of Q-learning and temporal difference (TD) learning algorithms has also been extensively studied in the literature [JJS94, Bai95, TVR97, PP02, MMR08, MM09, LLG$^+$15, BRS18, LS18, ZXL19]. However, the convergence guarantee of deep Q-learning algorithms remains a largely open problem. The only exceptions are [YXW19] which studied the fitted Q-iteration (FQI) algorithm [Rie05, MS08] with action-value function approximation based on a sparse

ReLU network, and [CYLW19] which studied the global convergence of Q-learning algorithm with an i.i.d. observation model and action-value function approximation based on a two-layer neural network. The main limitation of the aforementioned work is the unrealistic assumption that all the data used in the Q-learning algorithm are sampled i.i.d. from a fixed stationary distribution, which fails to capture the practical setting of neural Q-learning.

In this chapter, in order to bridge the gap between the empirical success of neural Q-learning and the theory of conventional Q-learning (i.e., tabular Q-learning, and Q-learning with linear function approximation), we study the non-asymptotic convergence of a neural Q-learning algorithm under non-i.i.d. observations. In particular, we use a deep neural network with the ReLU activation function to approximate the action-value function. In each iteration of the neural Q-learning algorithm, it updates the network weight parameters using the temporal difference (TD) error and the gradient of the neural network function. Our work extends existing finite-time analyses for TD learning [BRS18] and Q-learning [ZXL19], from linear function approximation to deep neural network based function approximation. Compared with the very recent theoretical work for neural Q-learning [YXW19, CYLW19], our analysis relaxes the non-realistic i.i.d. data assumption and applies to neural network approximation with arbitrary number of layers. Our main contributions are summarized as follows

- We establish the first finite-time analysis of Q-learning with deep neural network function approximation when the data are generated from a Markov decision process (MDP). We show that, when the network is sufficiently wide, neural Q-learning converges to the optimal action-value function up to the approximation error of the neural network function class.

- We establish an $O(1/\sqrt{T})$ convergence rate of neural Q-learning to the optimal Q-value function up to the approximation error, where T is the number of iterations. This convergence rate matches the one for TD-learning with linear function approximation and constant stepsize [BRS18]. Although we study a more challenging setting where the data are non-i.i.d. and the neural network approximator has multiple layers, our convergence

Table 6.1: Comparison with existing finite-time analyses of Q-learning.

Work	Non-i.i.d.	Neural Approximation	Multiple Layers	Rate
[BRS18]	✓	✗	✗	$O(1/T)$
[ZXL19]	✓	✗	✗	$O(1/T)$
[CYLW19]	✗	✓	✗	$O(1/\sqrt{T})$
This work	✓	✓	✓	$O(1/\sqrt{T})$

rate also matches the $O(1/\sqrt{T})$ rate proved in [CYLW19] with i.i.d. data and a two-layer neural network approximator.

To sum up, we present a comprehensive comparison between our work and the most relevant work in terms of their respective settings and convergence rates in Table 6.1.

6.2 Related Work

We review the most relevant work here in this section.

Asymptotic analysis The asymptotic convergence of TD learning and Q-learning algorithms has been well established in the literature [JJS94, TVR97, KT00, BM00, OS02, MMR08, DM17]. In particular, [TVR97] specified the precise conditions for TD learning with linear function approximation to converge and gave counterexamples that diverge. [MMR08] proved the asymptotic convergence of Q-learning with linear function approximation from standard ODE analysis, and identified a critic condition on the relationship between the learning policy and the greedy policy that ensures the almost sure convergence.

Finite-time analysis The finite-time analysis of the convergence rate for Q-learning algorithms has been largely unexplored until recently. In specific, [DSTM18, LS18] studied the convergence of TD(0) algorithm with linear function approximation under i.i.d. data assumptions and constant step sizes. Concurrently, a seminal work by [BRS18] provided a unified framework of analysis for TD learning under both i.i.d. and Markovian noise assumptions with an extra projection step. The analysis has been extended by [ZXL19] to SARSA and

Q-learning algorithms with linear function approximation. More recently, [SY19] established the finite-time convergence for TD learning algorithms with linear function approximation and a constant step-size without the extra projection step under non-i.i.d. data assumptions through carefully choosing the Lyapunov function for the associated ordinary differential equation of TD update. A similar analysis was also extended to Q-learning with linear function approximation [CZD+19]. [HS19] further provided a unified analysis for a class of TD learning algorithms using Markov jump linear system.

Neural function approximation Despite the empirical success of DQN, the theoretical convergence of Q-learning with deep neural network approximation is still missing in the literature. Following the recent advances in the theory of deep learning for overparameterized networks [JGH18, CB18, DZPS19, DLL+19, AZLS19, AZLL19, ZCZG19, ADH+19, CG19b, ZG19b, CGH+19], two recent work by [YXW19] and [CYLW19] proved the convergence rates of fitted Q-iteration and Q-learning with a sparse multi-layer ReLU network and two-layer neural network approximation respectively, under i.i.d. observations.

6.3 Preliminaries

A discrete-time Markov Decision Process (MDP) is denoted by a tuple $\mathcal{M} = (\mathcal{S}, \mathcal{A}, \mathcal{P}, r, \gamma)$. $\mathcal{S}$ and $\mathcal{A}$ are the sets of all states and actions respectively. $\mathcal{P} : \mathcal{S} \times \mathcal{A} \to \mathcal{P}(\mathcal{S})$ is the transition kernel such that $\mathcal{P}(s'|s, a)$ gives the probability of transiting to state s' after taking action a at state s. $r : \mathcal{S} \times \mathcal{A} \to [-1, 1]$ is a deterministic reward function. $\gamma \in (0, 1)$ is the discounted factor. A policy $\pi : \mathcal{S} \to \mathcal{P}(\mathcal{A})$ is a function mapping a state $s \in \mathcal{S}$ to a probability distribution $\pi(\cdot|s)$ over the action space. Let s_t and a_t denote the state and action at time step t. Then the transition kernel $\mathcal{P}$ and the policy π determine a Markov chain $\{s_t\}_{t=0,1,\ldots}$ For any fixed policy π, its associated value function $V^\pi : \mathcal{S} \to \mathbb{R}$ is defined as the expected total discounted reward:

$$V^\pi(s) = \mathbb{E}[\textstyle\sum_{t=0}^\infty \gamma^t r(s_t, a_t)|s_0 = s], \quad \forall s \in \mathcal{S}.$$

The corresponding action-value function $Q^\pi : \mathcal{S} \times \mathcal{A} \to \mathbb{R}$ is defined as

$$Q^\pi(s,a) = \mathbb{E}[\textstyle\sum_{t=0}^\infty \gamma^t r(s_t,a_t)|s_0 = s, a_0 = a] = r(s,a) + \gamma \int_{\mathcal{S}} V^\pi(s')\mathcal{P}(s'|s,a)\mathrm{d}s',$$

for all $s \in \mathcal{S}, a \in \mathcal{A}$. The optimal action-value function Q^* is defined as $Q^*(s,a) = \sup_\pi Q^\pi(s,a)$ for all $(s,a) \in \mathcal{S} \times \mathcal{A}$. Based on Q^*, the optimal policy π^* can be derived by following the greedy algorithm such that $\pi^*(a|s) = 1$ if $Q(s,a) = \max_{b\in\mathcal{A}} Q^*(s,b)$ and $\pi^*(a|s) = 0$ otherwise. We define the optimal Bellman operator $\mathcal{T}$ as follows

$$\mathcal{T}Q(s,a) = r(s,a) + \gamma \cdot \mathbb{E}\big[\max\nolimits_{b\in\mathcal{A}} Q(s',b)|s' \sim \mathcal{P}(\cdot|s,a)\big]. \tag{6.3.1}$$

It is worth noting that the optimal Bellman operator $\mathcal{T}$ is γ-contractive in the sup-norm and Q^* is the unique fixed point of $\mathcal{T}$ [Ber95].

6.4 The Neural Q-Learning Algorithm

In this section, we start with a brief review of Q-learning with linear function approximation. Then we will present the neural Q-learning algorithm.

6.4.1 Q-Learning with Linear Function Approximation

In many reinforcement learning algorithms, the goal is to estimate the action-value function $Q(\cdot,\cdot)$, which can be formulated as minimizing the mean-squared Bellman error (MSBE) [SB18]:

$$\min_{Q(\cdot,\cdot)} \mathbb{E}_{\mu,\pi,\mathcal{P}}\big[(\mathcal{T}Q(s,a) - Q(s,a))^2\big], \tag{6.4.1}$$

where state s is generated from the initial state distribution μ and action a is chosen based on a fixed learning policy π. To optimize (6.4.1), Q-learning iteratively updates the action-value function using the Bellman operator in (6.3.1), i.e., $Q_{t+1}(s,a) = \mathcal{T}Q_t(s,a)$ for all $(s,a) \in \mathcal{S} \times \mathcal{A}$. However, due to the large state and action spaces, whose cardinalities, i.e., $|\mathcal{S}|$ and $|\mathcal{A}|$, can be infinite for continuous problems in many applications, the aforementioned update is impractical. To address this issue, a linear function approximator is often used

[Sze10, SB18], where the action-value function is assumed to be parameterized by a linear function, i.e., $Q(s,a;\boldsymbol{\theta}) = \phi(s,a)^\top\boldsymbol{\theta}$ for any $(s,a) \in \mathcal{S}\times\mathcal{A}$, where $\phi : \mathcal{S}\times\mathcal{A} \to \mathbb{R}^d$ maps the state-action pair to a d-dimensional vector, and $\boldsymbol{\theta} \in \boldsymbol{\Theta} \subseteq \mathbb{R}^d$ is an unknown weight vector. The minimization problem in (6.4.1) then turns to minimizing the MSBE over the parameter space $\boldsymbol{\Theta}$.

6.4.2 Neural Q-Learning

Analogous to Q-learning with linear function approximation, the action-value function can also be approximated by a deep neural network to increase the representation power of the approximator. Specifically, we define a L-hidden-layer neural network as follows

$$f(\boldsymbol{\theta};\mathbf{x}) = \sqrt{m}\mathbf{W}_L\sigma_L(\mathbf{W}_{L-1}\cdots\sigma(\mathbf{W}_1\mathbf{x})\cdots), \tag{6.4.2}$$

where $\mathbf{x} \in \mathbb{R}^d$ is the input data, $\mathbf{W}_1 \in \mathbb{R}^{m\times d}$, $\mathbf{W}_L \in \mathbb{R}^{1\times m}$ and $\mathbf{W}_l \in \mathbb{R}^{m\times m}$ for $l = 2,\ldots,L-1$, $\boldsymbol{\theta} = (\mathrm{vec}(\mathbf{W}_1)^\top,\ldots,\mathrm{vec}(\mathbf{W}_L)^\top)^\top$ is the concatenation of the vectorization of all parameter matrices, and $\sigma(x) = \max\{0,x\}$ is the ReLU activation function. Then, we can parameterize $Q(s,a)$ using a deep neural network as $Q(s,a;\boldsymbol{\theta}) = f(\boldsymbol{\theta};\phi(s,a))$, where $\boldsymbol{\theta} \in \boldsymbol{\Theta}$ and $\phi : \mathcal{S}\times\mathcal{A} \to \mathbb{R}^d$ is a feature mapping. Without loss of generality, we assume that $\|\phi(s,a)\|_2 \le 1$ in this chapter. Let π be an arbitrarily stationary policy. The MSBE minimization problem in (6.4.1) can be rewritten in the following form

$$\min_{\boldsymbol{\theta}\in\boldsymbol{\Theta}} \mathbb{E}_{\mu,\pi,\mathcal{P}}\big[(Q(s,a;\boldsymbol{\theta}) - \mathcal{T}Q(s,a;\boldsymbol{\theta}))^2\big]. \tag{6.4.3}$$

Recall that the optimal action-value function Q^* is the fixed point of Bellman optimality operator $\mathcal{T}$ which is γ-contractive. Therefore Q^* is the unique global minimizer of (6.4.3).

The nonlinear parameterization of $Q(\cdot,\cdot)$ turns the MSBE in (6.4.3) to be highly nonconvex, which imposes difficulty in finding the global optimum $\boldsymbol{\theta}^*$. To mitigate this issue, we will approximate the solution of (6.4.3) by project the Q-value function into some function class parameterized by $\boldsymbol{\theta}$, which leads to minimizing the mean square projected Bellman error (MSPBE):

$$\min_{\boldsymbol{\theta}\in\boldsymbol{\Theta}} \mathbb{E}_{\mu,\pi,\mathcal{P}}\big[(Q(s,a;\boldsymbol{\theta}) - \Pi_{\mathcal{F}}\mathcal{T}Q(s,a;\boldsymbol{\theta}))^2\big], \tag{6.4.4}$$

Algorithm 10 Neural Q-Learning with Gaussian Initialization

```
1: Input: learning policy $\pi$, learning rate $\{\eta_t\}_{t=0,1,\dots}$, discount factor $\gamma$, constraint set $\boldsymbol{\Theta}$,
   Randomly generate the entries of $\mathbf{W}_l^{(0)}$ from $N(0, 1/m)$, $l = 1, \dots, m$
2: Initialization: $\boldsymbol{\theta}_0 = (\mathbf{W}_0^{(1)\top}, \dots, \mathbf{W}_0^{(L)\top})^\top$
3: for $t = 0, \dots, T-1$ do
4:     Sample data $(s_t, a_t, r_t, s_{t+1})$ from policy $\pi$
5:     $\Delta_t = f(\boldsymbol{\theta}_t; \phi(s_t, a_t)) - (r_t + \gamma \max_{b \in \mathcal{A}} f(\boldsymbol{\theta}_t; \phi(s_{t+1}, b)))$
6:     $\mathbf{g}_t(\boldsymbol{\theta}_t) = \nabla_{\boldsymbol{\theta}} f(\boldsymbol{\theta}_t; \phi(s_t, a_t)) \Delta_t$
7:     $\boldsymbol{\theta}_{t+1} = \Pi_{\boldsymbol{\Theta}}(\boldsymbol{\theta}_t - \eta_t \mathbf{g}_t(\boldsymbol{\theta}_t))$
8: end for
```

where $\mathcal{F} = \{Q(\cdot, \cdot; \boldsymbol{\theta}) : \boldsymbol{\theta} \in \boldsymbol{\Theta}\}$ is some function class parameterized by $\boldsymbol{\theta} \in \boldsymbol{\Theta}$, and $\Pi_{\mathcal{F}}$ is a projection operator. Then the neural Q-learning algorithm updates the weight parameter $\boldsymbol{\theta}$ using the following projected descent step: $\boldsymbol{\theta}_{t+1} = \Pi_{\boldsymbol{\Theta}}(\boldsymbol{\theta}_t - \eta_t \mathbf{g}_t(\boldsymbol{\theta}_t))$, where the gradient term $\mathbf{g}_t(\boldsymbol{\theta}_t)$ is defined as

$$\begin{aligned}\mathbf{g}_t(\boldsymbol{\theta}_t) &= \nabla_{\boldsymbol{\theta}} f(\boldsymbol{\theta}_t; \phi(s_t, a_t)) \big(f(\boldsymbol{\theta}_t; \phi(s_t, a_t)) - r_t - \gamma \max_{b \in \mathcal{A}} f(\boldsymbol{\theta}_t; \phi(s_{t+1}, b)) \big) \\ &\stackrel{\text{def}}{=} \Delta_t(s_t, a_t, s_{t+1}; \boldsymbol{\theta}_t) \nabla_{\boldsymbol{\theta}} f(\boldsymbol{\theta}_t; \phi(s_t, a_t)), \end{aligned} \tag{6.4.5}$$

and Δ_t is the temporal difference (TD) error. It should be noted that $\mathbf{g}_t$ is not the gradient of the MSPBE nor an unbiased estimator for it. The details of the neural Q-learning algorithm are displayed in Algorithm 10, where $\boldsymbol{\theta}_0$ is randomly initialized, and the constraint set is chosen to be $\boldsymbol{\Theta} = \mathbb{B}(\boldsymbol{\theta}_0, \omega)$, which is defined as follows

$$\mathbb{B}(\boldsymbol{\theta}_0, \omega) \stackrel{\text{def}}{=} \{\boldsymbol{\theta} = (\text{vec}(\mathbf{W}_1)^\top, \dots, \text{vec}(\mathbf{W}_L)^\top)^\top : \|\mathbf{W}_l - \mathbf{W}_l^{(0)}\|_F \le \omega, l = 1, \dots, L\} \tag{6.4.6}$$

for some tunable parameter ω. It is easy to verify that $\|\boldsymbol{\theta} - \boldsymbol{\theta}'\|_2^2 = \sum_{l=1}^L \|\mathbf{W}_l - \mathbf{W}_l'\|_F^2$.

6.5 Convergence Analysis of Neural Q-Learning

In this section, we provide a finite-sample analysis of neural Q-learning. Throughout this chapter, we reserve the notations $\{C_i\}_{i=0,1,\dots}$ to represent universal positive constants that are independent of problem parameters. The specific value of $\{C_i\}_{i=1,2,\ldots}$ can be different line by line. Note that the optimization problem in (6.4.4) is nonconvex. We focus on finding a surrogate action-value function in the network function class that well approximates Q^*.

6.5.1 Approximate Stationary Point in the Constrained Space

To ease the presentation, we abbreviate $f(\boldsymbol{\theta}; \phi(s,a))$ as $f(\boldsymbol{\theta})$ when no confusion arises. We define the function class $\mathcal{F}_{\boldsymbol{\Theta},m}$ as a collection of all local linearization of $f(\boldsymbol{\theta})$ at the initial point $\boldsymbol{\theta}_0$

$$\mathcal{F}_{\boldsymbol{\Theta},m} = \{f(\boldsymbol{\theta}_0) + \langle \nabla_{\boldsymbol{\theta}} f(\boldsymbol{\theta}_0), \boldsymbol{\theta} - \boldsymbol{\theta}_0 \rangle : \boldsymbol{\theta} \in \boldsymbol{\Theta}\}. \tag{6.5.1}$$

Following to the local linearization analysis in [CYLW19], we define the approximate stationary point of Algorithm 10 as follows.

Definition 6.5.1 ([CYLW19])**.** *A point $\boldsymbol{\theta}^* \in \boldsymbol{\Theta}$ is said to be the approximate stationary point of Algorithm 10 if for all $\boldsymbol{\theta} \in \boldsymbol{\Theta}$ it holds that*

$$\mathbb{E}_{\mu,\pi,\mathcal{P}}\big[\hat{\Delta}(s,a,s';\boldsymbol{\theta}^*)\langle \nabla_{\boldsymbol{\theta}} \hat{f}(\boldsymbol{\theta}^*; \phi(s,a)), \boldsymbol{\theta} - \boldsymbol{\theta}^* \rangle\big] \geq 0, \tag{6.5.2}$$

where $\hat{f}(\boldsymbol{\theta}; \phi(s,a)) := \hat{f}(\boldsymbol{\theta}) \in \mathcal{F}_{\boldsymbol{\Theta},m}$ and the temporal difference error $\hat{\Delta}$ is

$$\hat{\Delta}(s,a,s';\boldsymbol{\theta}) = \hat{f}(\boldsymbol{\theta}; \phi(s,a)) - \big(r(s,a) + \gamma \max\nolimits_{b \in \mathcal{A}} \hat{f}(\boldsymbol{\theta}; \phi(s',b))\big). \tag{6.5.3}$$

For any $\hat{f} \in \mathcal{F}_{\boldsymbol{\Theta},m}$, it holds that $\langle \nabla_{\boldsymbol{\theta}} \hat{f}(\boldsymbol{\theta}^*), \boldsymbol{\theta} - \boldsymbol{\theta}^* \rangle = \langle \nabla_{\boldsymbol{\theta}} f(\boldsymbol{\theta}_0), \boldsymbol{\theta} - \boldsymbol{\theta}^* \rangle = \hat{f}(\boldsymbol{\theta}) - \hat{f}(\boldsymbol{\theta}^*)$. Definition 6.5.1 immediately implies

$$\mathbb{E}_{\mu,\pi,\mathcal{P}}\big[\big(\hat{f}(\boldsymbol{\theta}^*) - \mathcal{T}\hat{f}(\boldsymbol{\theta}^*)\big)\big(\hat{f}(\boldsymbol{\theta}) - \hat{f}(\boldsymbol{\theta}^*)\big)\big] \geq 0, \qquad \forall \boldsymbol{\theta} \in \boldsymbol{\Theta}. \tag{6.5.4}$$

According to Proposition 4.2 in [CYLW19], this further indicates $\hat{f}(\boldsymbol{\theta}^*) = \Pi_{\mathcal{F}_{\boldsymbol{\Theta},m}} \mathcal{T}\hat{f}(\boldsymbol{\theta}^*)$. In other words, $\hat{f}(\boldsymbol{\theta}^*)$ is the unique fixed point of the MSPBE in (6.4.4). Therefore, we can show the convergence of neural Q-learning to the optimal action-value function Q^* by first connecting it to the minimizer $\hat{f}(\boldsymbol{\theta}^*)$ and then adding the approximation error of $\mathcal{F}_{\boldsymbol{\Theta},m}$.

6.5.2 The Main Theory

Before we present the convergence of Algorithm 10, let us lay down the assumptions used throughout this chapter. The first assumption controls the bias caused by the Markovian noise in the observations through assuming the uniform ergodicity of the Markov chain generated by the learning policy π.

Assumption 6.5.2. *The learning policy π and the transition kernel $\mathcal{P}$ induce a Markov chain $\{s_t\}_{t=0,1,\ldots}$ such that there exist constants $\lambda > 0$ and $\rho \in (0,1)$ satisfying*

$$\sup_{s\in\mathcal{S}} d_{TV}(\mathbb{P}(s_t \in \cdot | s_0 = s), \pi) \le \lambda\rho^t, \quad \textit{for all } t = 0, 1, \ldots$$

Assumption 6.5.2 also appears in [BRS18, ZXL19], which is essential for the analysis of the Markov decision process. The uniform ergodicity can be established via the minorization condition for irreducible Markov chains [MT12, LP17].

For the purpose of exploration, we also need to assume that the learning policy π satisfies some regularity condition. Denote $b_{\max}(\boldsymbol{\theta}) = \mathrm{argmax}_{b\in\mathcal{A}} |\langle \nabla_{\boldsymbol{\theta}} f(\boldsymbol{\theta}_0; s, b), \boldsymbol{\theta}\rangle|$ for any $\boldsymbol{\theta} \in \boldsymbol{\Theta}$. Similar to [MMR08, ZXL19, CZD+19], we define

$$\boldsymbol{\Sigma}_\pi = 1/m \mathbb{E}_{\mu,\pi}\big[\nabla_{\boldsymbol{\theta}} f(\boldsymbol{\theta}_0; s, a) \nabla_{\boldsymbol{\theta}} f(\boldsymbol{\theta}_0; s, a)^\top\big], \tag{6.5.5}$$

$$\boldsymbol{\Sigma}_\pi^*(\boldsymbol{\theta}) = 1/m \mathbb{E}_{\mu,\pi}\big[\nabla_{\boldsymbol{\theta}} f(\boldsymbol{\theta}_0; s, b_{\max}(\boldsymbol{\theta})) \nabla_{\boldsymbol{\theta}} f(\boldsymbol{\theta}_0; s, b_{\max}(\boldsymbol{\theta}))^\top\big]. \tag{6.5.6}$$

Note that $\boldsymbol{\Sigma}_\pi$ is independent of $\boldsymbol{\theta}$ and only depends on the policy π and the initial point $\boldsymbol{\theta}_0$ in the definition of $\hat{f}$. In contrast, $\boldsymbol{\Sigma}_\pi^*(\boldsymbol{\theta})$ is defined based on the greedy action under the policy associated with $\boldsymbol{\theta}$. The scaling parameter $1/m$ is used to ensure that the operator norm of $\boldsymbol{\Sigma}_\pi$ to be in the order of $O(1)$. It is worth noting that $\boldsymbol{\Sigma}_\pi$ is different from the neural tangent kernel (NTK) or the Gram matrix in [JGH18, DLL+19, ADH+19], which are $n \times n$ matrices defined based on a finite set of data points $\{(s_i, a_i)\}_{i=1,\ldots,n}$. When f is linear, $\boldsymbol{\Sigma}_\pi$ reduces to the covariance matrix of the feature vector.

Assumption 6.5.3. *There exists a constant $\alpha > 1$ such that $\boldsymbol{\Sigma}_\pi - \alpha\gamma^2\boldsymbol{\Sigma}_\pi^*(\boldsymbol{\theta}) \succ \mathbf{0}$ for all $\boldsymbol{\theta}$ and $\boldsymbol{\theta}_0$.*

Assumption 6.5.3 is also made for Q-learning with linear function approximation in [MMR08, ZXL19, CZD$^+$19]. Moreover, [CZD$^+$19] presented numerical simulations to verify the validity of Assumption 6.5.3. [CYLW19] imposed a slightly different assumption but with the same idea that the learning policy π should be not too far away from the greedy policy. The regularity assumption on the learning policy is directly imposed on the action value function in [CYLW19], which can be implied by Assumption 6.5.3 and thus is slightly weaker. We note that Assumption 6.5.3 can be relaxed to the one made in [CYLW19] without changing any of our analysis. Nevertheless, we choose to present the current version which is more consistent with existing work on Q-learning with linear function approximation [MMR08, CZD$^+$19].

Theorem 6.5.4. *Suppose Assumptions 6.5.2 and 6.5.3 hold. The constraint set* $\boldsymbol{\Theta}$ *is defined as in* (6.4.6). *We set the radius as* $\omega = C_0 m^{-1/2} L^{-9/4}$*, the step size in Algorithm 10 as* $\eta = 1/(2(1-\alpha^{-1/2})mT)$*, and the width of the neural network as*

$$m \geq C_1 \max\{dL^2 \log(m/\delta), \omega^{-4/3} L^{-8/3} \log(m/(\omega\delta))\},$$

where $\delta \in (0,1)$. *Then with probability at least* $1 - 2\delta - L^2 \exp(-C_2 m^{2/3} L)$ *over the randomness of the Gaussian initialization* $\boldsymbol{\theta}_0$ *, it holds that*

$$\frac{1}{T}\sum_{t=0}^{T-1} \mathbb{E}\big[\big(\hat{f}(\boldsymbol{\theta}_t) - \hat{f}(\boldsymbol{\theta}^*)\big)^2 \big| \boldsymbol{\theta}_0\big] \leq \frac{1}{\sqrt{T}} + \frac{C_2 \tau^* \log(T/\delta) \log T}{\beta^2 \sqrt{T}} + \frac{C_3 \log m \log(T/\delta)}{\beta m^{1/6}},$$

where $\beta = 1 - \alpha^{-1/2} \in (0,1)$ *is a constant,* $\tau^* = \min\{t = 0,1,2,\ldots | \lambda\rho^t \leq \eta_T\}$ *is the mixing time of the Markov chain* $\{s_t, a_t\}_{t=0,1,\ldots}$*, and* $\{C_i\}_{i=0,\ldots,5}$ *are universal constants independent of problem parameters.*

Remark 6.5.5. *Theorem 6.5.4 characterizes the distance between the output of Algorithm 10 to the approximate stationary point defined in function class* $\mathcal{F}_{\boldsymbol{\Theta},m}$. *From* (6.5.4)*, we know that* $\hat{f}(\boldsymbol{\theta}^*)$ *is the minimizer of the MSPBE* (6.4.4). *Note that* τ^* *is in the order of* $O(\log(mT/\log T))$. *Theorem 6.5.4 suggests that neural Q-learning converges to the minimizer of MSPBE with a rate in the order of* $O((\log(mT))^3/\sqrt{T} + \log m \log T/m^{1/6})$*, which reduces to* $\tilde{O}(1/\sqrt{T})$ *when the width* m *of the neural network is sufficiently large.*

In the following theorem, we show that neural Q-learning converges to the optimal action-value function within finite time if the neural network is overparameterized.

Theorem 6.5.6. *Under the same conditions as in Theorem 6.5.4, with probability at least $1 - 3\delta - L^2 \exp(-C_0 m^{2/3} L)$ over the randomness of $\boldsymbol{\theta}_0$, it holds that*

$$\frac{1}{T}\sum_{t=0}^{T-1}\mathbb{E}\big[(Q(s,a;\boldsymbol{\theta}_t) - Q^*(s,a))^2\big] \le \frac{3\mathbb{E}\big[\big(\Pi_{\mathcal{F}_{\boldsymbol{\Theta},m}}Q^*(s,a) - Q^*(s,a)\big)^2\big]}{(1-\gamma)^2} + \frac{1}{\sqrt{T}} + \frac{C_1\tau^*\log(T/\delta)\log T}{\beta^2\sqrt{T}} + \frac{C_2\log(T/\delta)\log m}{\beta m^{1/6}},$$

where all the expectations are taken conditional on $\boldsymbol{\theta}_0$, Q^* is the optimal action-value function, $\delta \in (0,1)$ and $\{C_i\}_{i=0,\dots,2}$ are universal constants.

The optimal policy π^* can be obtained by the greedy algorithm derived based on Q^*.

Remark 6.5.7. *The convergence rate in Theorem 6.5.6 can be simplifies as follows*

$$\frac{1}{T}\sum_{t=0}^{T-1}\mathbb{E}[(Q(s,a;\boldsymbol{\theta}_t) - Q^*(s,a))^2|\boldsymbol{\theta}_0] = \tilde{O}\bigg(\mathbb{E}\big[\big(\Pi_{\mathcal{F}_{\boldsymbol{\Theta},m}}Q^*(s,a) - Q^*(s,a)\big)^2\big] + \frac{1}{m^{1/6}} + \frac{1}{\sqrt{T}}\bigg).$$

The first term is the projection error of the optimal Q-value function on to the function class $\mathcal{F}_{\boldsymbol{\Theta},m}$, which decreases to zero as the representation power of $\mathcal{F}_{\boldsymbol{\Theta},m}$ increases. In fact, when the width m of the DNN is sufficiently large, recent studies [CG19b, CG19a] show that $f(\boldsymbol{\theta})$ is almost linear around the initialization and the approximate stationary point $\hat{f}(\boldsymbol{\theta}^*)$ becomes the fixed solution of the MSBE [CYLW19]. Moreover, this term diminishes when the Q function is approximated by linear functions when the underlying parameter has a bounded norm [BRS18, ZXL19]. As m goes to infinity, we obtain the convergence of neural Q-learning to the optimal Q-value function with an $O(1/\sqrt{T})$ rate.

6.6 Proof of Main Results

In this section, we provide the detailed proof of the convergence of Algorithm 10. To simplify the presentation, we write $f(\boldsymbol{\theta};\phi(s,a))$ as $f(\boldsymbol{\theta};s,a)$ throughout the proof when no confusion arises.

We first define some notations that will simplify the presentation of the proof. Recall the definition of $\mathbf{g}_t(\cdot)$ in (6.4.5). For any $\boldsymbol{\theta} \in \boldsymbol{\Theta}$, we define the following vector-value map $\overline{\mathbf{g}}$ that is independent of the data point.

$$\overline{\mathbf{g}}(\boldsymbol{\theta}) = \mathbb{E}_{\mu,\pi,\mathcal{P}}[\nabla_{\boldsymbol{\theta}} f(\boldsymbol{\theta}; s, a)(f(\boldsymbol{\theta}; s, a) - r(s, a) - \gamma \mathrm{max}_{b \in \mathcal{A}} f(\boldsymbol{\theta}; s', b))], \tag{6.6.1}$$

where s follows the initial state distribution μ, a is chosen based on the policy $\pi(\cdot|s)$ and s' follows the transition probability $\mathcal{P}(\cdot|s, a)$. Similarly, for all $\boldsymbol{\theta} \in \boldsymbol{\Theta}$, we define the following gradient terms based on the linearized function $\hat{f} \in \mathcal{F}_{\boldsymbol{\Theta},m}$

$$\mathbf{m}_t(\boldsymbol{\theta}) = \hat{\Delta}(s_t, a_t, s_{t+1}; \boldsymbol{\theta}) \nabla_{\boldsymbol{\theta}} \hat{f}(\boldsymbol{\theta}), \quad \overline{\mathbf{m}}(\boldsymbol{\theta}) = \mathbb{E}_{\mu,\pi,\mathbf{P}}\big[\hat{\Delta}(s, a, s'; \boldsymbol{\theta}) \nabla_{\boldsymbol{\theta}} \hat{f}(\boldsymbol{\theta})\big], \tag{6.6.2}$$

where $\hat{\Delta}$ is defined in (6.5.3), and a population version based on the linearized function.

Now we present the technical lemmas that are useful in our proof of Theorem 6.5.4. For the gradients $\mathbf{g}_t(\cdot)$ defined in (6.4.5) and $\mathbf{m}_t(\cdot)$ defined in (6.6.2), we have the following lemma that characterizes the difference between the gradient of the neural network function f and the gradient of the linearized function $\hat{f}$.

Lemma 6.6.1. *The gradient of neural network function is close to the linearized gradient. Specifically, if $\boldsymbol{\theta}_t \in \mathbb{B}(\boldsymbol{\Theta}, \omega)$ and m and ω satisfy*

$$\begin{aligned} & m \geq C_0 \max\{dL^2 \log(m/\delta), \omega^{-4/3} L^{-8/3} \log(m/(\omega\delta))\}, \\ & \textit{and} \quad C_1 d^{3/2} L^{-1} m^{-3/4} \leq \omega \leq C_2 L^{-6} (\log m)^{-3}, \end{aligned} \tag{6.6.3}$$

then it holds that

$$\begin{aligned} |\langle \mathbf{g}_t(\boldsymbol{\theta}_t) - \mathbf{m}_t(\boldsymbol{\theta}_t), \boldsymbol{\theta}_t - \boldsymbol{\theta}^* \rangle| \leq {} & C_3 (2 + \gamma) \omega^{1/3} L^3 \sqrt{m \log m \log(T/\delta)} \|\boldsymbol{\theta}_t - \boldsymbol{\theta}^*\|_2 \\ & + \big(C_4 \omega^{4/3} L^{11/3} m \sqrt{\log m} + C_5 \omega^2 L^4 m\big) \|\boldsymbol{\theta}_t - \boldsymbol{\theta}^*\|_2, \end{aligned}$$

with probability at least $1 - 2\delta - 3L^2 \exp(-C_6 m \omega^{2/3} L)$ over the randomness of the initial point, and $\|\mathbf{g}_t(\boldsymbol{\theta}_t)\|_2 \leq (2 + \gamma) C_7 \sqrt{m \log(T/\delta)}$ holds with probability at least $1 - \delta - L^2 \exp(-C_6 m \omega^{2/3} L)$. where $\{C_i > 0\}_{i=0,\ldots,7}$ are universal constants.

The next lemma upper bounds the bias of the non-i.i.d. data for the linearized gradient map.

Lemma 6.6.2. *Suppose the step size sequence $\{\eta_0, \eta_1, \ldots, \eta_T\}$ is nonincreasing. Then it holds that*

$$\mathbb{E}[\langle \mathbf{m}_t(\boldsymbol{\theta}_t) - \overline{\mathbf{m}}(\boldsymbol{\theta}_t), \boldsymbol{\theta}_t - \boldsymbol{\theta}^* \rangle | \boldsymbol{\theta}_0] \leq C_0(m \log(T/\delta) + m^2\omega^2)\tau^* \eta_{\max\{0, t-\tau^*\}},$$

for any fixed $t \leq T$, where $C_0 > 0$ is an universal constant and $\tau^* = \min\{t = 0, 1, 2, \ldots | \lambda\rho^t \leq \eta_T\}$ is the mixing time of the Markov chain $\{s_t, a_t\}_{t=0,1,\ldots}$.

Since $\hat{f}$ is a linear function approximator of the neural network function f, we can show that the gradient of $\hat{f}$ satisfies the following nice property in the constrained set $\boldsymbol{\Theta}$.

Lemma 6.6.3. *Under Assumption 6.5.3, $\overline{\mathbf{m}}(\cdot)$ defined in* (6.6.2) *satisfies*

$$\langle \overline{\mathbf{m}}(\boldsymbol{\theta}) - \overline{\mathbf{m}}(\boldsymbol{\theta}^*), \boldsymbol{\theta} - \boldsymbol{\theta}^* \rangle \geq (1 - \alpha^{-1/2})\mathbb{E}\big[\big(\hat{f}(\boldsymbol{\theta}) - \hat{f}(\boldsymbol{\theta}^*)\big)^2 \big| \boldsymbol{\theta}_0\big], \quad \forall \boldsymbol{\theta} \in \boldsymbol{\Theta}.$$

Now we can integrate the above results and obtain proof of Theorem 6.5.4.

Proof of Theorem 6.5.4. By Algorithm 10 and the non-expansiveness of projection $\Pi_{\boldsymbol{\Theta}}$, we have

$$\begin{aligned}
\|\boldsymbol{\theta}_{t+1} - \boldsymbol{\theta}^*\|_2^2 &= \|\Pi_{\boldsymbol{\Theta}}\big(\boldsymbol{\theta}_t - \eta_t \mathbf{g}_t\big) - \boldsymbol{\theta}^*\|_2^2 \\
&\leq \|\boldsymbol{\theta}_t - \eta_t \mathbf{g}_t - \boldsymbol{\theta}^*\|_2^2 \\
&= \|\boldsymbol{\theta}_t - \boldsymbol{\theta}^*\|_2^2 + \eta_t^2 \|\mathbf{g}_t\|_2^2 - 2\eta_t \langle \mathbf{g}_t, \boldsymbol{\theta}_t - \boldsymbol{\theta}^* \rangle. \qquad (6.6.4)
\end{aligned}$$

We need to find an upper bound for the gradient norm and a lower bound for the inner product. According to Definition 6.5.1, the approximate stationary point $\boldsymbol{\theta}^*$ of Algorithm 10 satisfies $\langle \overline{\mathbf{m}}(\boldsymbol{\theta}^*), \boldsymbol{\theta} - \boldsymbol{\theta}^* \rangle \geq 0$ for all $\boldsymbol{\theta} \in \boldsymbol{\Theta}$. The inner product in (6.6.4) can be decomposed into

$$\begin{aligned}
\langle \mathbf{g}_t, \boldsymbol{\theta}_t - \boldsymbol{\theta}^* \rangle &= \langle \mathbf{g}_t - \mathbf{m}_t(\boldsymbol{\theta}_t), \boldsymbol{\theta}_t - \boldsymbol{\theta}^* \rangle + \langle \mathbf{m}_t(\boldsymbol{\theta}_t) - \overline{\mathbf{m}}(\boldsymbol{\theta}_t), \boldsymbol{\theta}_t - \boldsymbol{\theta}^* \rangle + \langle \overline{\mathbf{m}}(\boldsymbol{\theta}_t), \boldsymbol{\theta}_t - \boldsymbol{\theta}^* \rangle \\
&\geq \langle \mathbf{g}_t - \mathbf{m}_t(\boldsymbol{\theta}_t), \boldsymbol{\theta}_t - \boldsymbol{\theta}^* \rangle + \langle \mathbf{m}_t(\boldsymbol{\theta}_t) - \overline{\mathbf{m}}(\boldsymbol{\theta}_t), \boldsymbol{\theta}_t - \boldsymbol{\theta}^* \rangle \\
&\quad + \langle \overline{\mathbf{m}}(\boldsymbol{\theta}_t) - \overline{\mathbf{m}}(\boldsymbol{\theta}^*), \boldsymbol{\theta}_t - \boldsymbol{\theta}^* \rangle. \qquad (6.6.5)
\end{aligned}$$

Combining results from (6.6.4)and (6.6.5), we have

$$\|\boldsymbol{\theta}_{t+1}-\boldsymbol{\theta}^*\|_2^2 \le \|\boldsymbol{\theta}_t-\boldsymbol{\theta}^*\|_2^2+\eta_t^2\|\mathbf{g}_t\|_2^2-2\eta_t\underbrace{\langle \mathbf{g}_t-\mathbf{m}_t(\boldsymbol{\theta}_t),\boldsymbol{\theta}_t-\boldsymbol{\theta}^*\rangle}_{I_1}$$
$$-2\eta_t\underbrace{\langle \mathbf{m}_t(\boldsymbol{\theta}_t)-\overline{\mathbf{m}}(\boldsymbol{\theta}_t),\boldsymbol{\theta}_t-\boldsymbol{\theta}^*\rangle}_{I_2}-2\eta_t\underbrace{\langle \overline{\mathbf{m}}(\boldsymbol{\theta}_t)-\overline{\mathbf{m}}(\boldsymbol{\theta}^*),\boldsymbol{\theta}_t-\boldsymbol{\theta}^*\rangle}_{I_3}. \quad (6.6.6)$$

Recall constraint set defined in (6.4.6). We have $\boldsymbol{\Theta}=\mathbb{B}(\boldsymbol{\theta}_0,\omega)=\{\boldsymbol{\theta}:\|\mathbf{W}_l-\mathbf{W}_l^{(0)}\|_F\le \omega,\forall l=1,\ldots,L\}$ and that m and ω satisfy the condition in (6.6.3).

Term I_1 is the error of the local linearization of $f(\boldsymbol{\theta})$ at $\boldsymbol{\theta}_0$. By Lemma 6.6.1, with probability at least $1-2\delta-3L^2\exp(-C_1m\omega^{2/3}L)$ over the randomness of the initial point $\boldsymbol{\theta}_0$, we have

$$|\langle \mathbf{g}_t-\mathbf{m}_t(\boldsymbol{\theta}_t),\boldsymbol{\theta}_t-\boldsymbol{\theta}^*\rangle|\le C_2(2+\gamma)m^{-1/6}\sqrt{\log m\log(T/\delta)} \quad (6.6.7)$$

holds uniformly for all $\boldsymbol{\theta}_t,\boldsymbol{\theta}^*\in\boldsymbol{\Theta}$, where we used the fact that $\omega=C_0m^{-1/2}L^{-9/4}$.

Term I_2 is the bias of caused by the non-i.i.d. data (s_t,a_t,s_{t+1}) used in the update of Algorithm 10. Conditional on the initialization, by Lemma 6.6.2, we have

$$\mathbb{E}[\langle \mathbf{m}_t(\boldsymbol{\theta}_t)-\overline{\mathbf{m}}(\boldsymbol{\theta}_t),\boldsymbol{\theta}_t-\boldsymbol{\theta}^*\rangle|\boldsymbol{\theta}_0]\le C_3(m\log(T/\delta)+m^2\omega^2)\tau^*\eta_{\max\{0,t-\tau^*\}}, \quad (6.6.8)$$

where $\tau^*=\min\{t=0,1,2,\ldots|\lambda\rho^t\le\eta_T\}$ is the mixing time of the Markov chain $\{s_t,a_t\}_{t=0,1,\ldots}$.

Term I_3 is the estimation error for the linear function approximation. By Lemma 6.6.3, we have

$$\langle \overline{\mathbf{m}}(\boldsymbol{\theta}_t)-\overline{\mathbf{m}}(\boldsymbol{\theta}^*),\boldsymbol{\theta}_t-\boldsymbol{\theta}^*\rangle\ge\beta\mathbb{E}\big[\big(\hat{f}(\boldsymbol{\theta}_t)-\hat{f}(\boldsymbol{\theta}^*)\big)^2\big|\boldsymbol{\theta}_0\big], \quad (6.6.9)$$

where $\beta=(1-\alpha^{-1/2})\in(0,1)$ is a constant. Substituting (6.6.7), (6.6.8) and (6.6.9) into (6.6.6), we have it holds that

$$\begin{aligned}\|\boldsymbol{\theta}_{t+1}-\boldsymbol{\theta}^*\|_2^2 &\le \|\boldsymbol{\theta}_t-\boldsymbol{\theta}^*\|_2^2+\eta_t^2C_4^2(2+\gamma)^2m\log(T/\delta)\\ &\quad-2\eta_tC_2(2+\gamma)m^{-1/6}\sqrt{\log m\log(T/\delta)}-2\eta_t\beta\mathbb{E}\big[\big(\hat{f}(\boldsymbol{\theta}_t)-\hat{f}(\boldsymbol{\theta}^*)\big)^2\big|\boldsymbol{\theta}_0\big]\\ &\quad-2\eta_tC_3(m\log(T/\delta)+m^2\omega^2)\tau^*\eta_{\max\{0,t-\tau^*\}}, \quad (6.6.10)\end{aligned}$$

with probability at least $1-2\delta-3L^2\exp(-C_1m\omega^{2/3}L)$ over the randomness of the initial point $\boldsymbol{\theta}_0$, where we used the fact that $\|\mathbf{g}_t\|_F\le C_4(2+\gamma)\sqrt{m\log(T/\delta)}$ from Lemma 6.6.1.

Rearranging the above inequality yields

$$\begin{aligned}
&\mathbb{E}\big[\big(\hat{f}(\boldsymbol{\theta}_t)-\hat{f}(\boldsymbol{\theta}^*)\big)^2\big|\boldsymbol{\theta}_0\big] \\
&\leq \frac{\|\boldsymbol{\theta}_t-\boldsymbol{\theta}^*\|_2^2-\|\boldsymbol{\theta}_{t+1}-\boldsymbol{\theta}^*\|_2^2}{2\beta\eta_t}+\frac{C_2(2+\gamma)m^{-1/6}\log m\log(T/\delta)}{\beta} \\
&\quad +\frac{C_4(2+\gamma)^2m\log(T/\delta)\eta_t}{\beta}+\frac{C_3m(\log(T/\delta)+m\omega^2)\tau^*\eta_{\max\{0,t-\tau^*\}}}{\beta},
\end{aligned}$$

with probability at least $1-2\delta-3L^2\exp(-C_1m\omega^{2/3}L)$ over the randomness of the initial point $\boldsymbol{\theta}_0$. Recall the choices of the step sizes $\eta_0=\ldots=\eta_T=1/(2\beta m\sqrt{T})$ and the radius $\omega=C_0m^{-1/2}L^{-9/4}$. Dividing the above inequality by T and telescoping it from $t=0$ to T yields

$$\begin{aligned}
\frac{1}{T}\sum_{t=0}^{T-1}\mathbb{E}\big[\big(\hat{f}(\boldsymbol{\theta}_t)-\hat{f}(\boldsymbol{\theta}^*)\big)^2\big|\boldsymbol{\theta}_0\big] &\leq \frac{m\|\boldsymbol{\theta}_0-\boldsymbol{\theta}^*\|_2^2}{\sqrt{T}}+\frac{C_2(2+\gamma)m^{-1/6}\log m\log(T/\delta)}{\beta} \\
&\quad +\frac{C_4(2+\gamma)^2\log(T/\delta)\log T}{\beta^2\sqrt{T}}+\frac{C_3(\log(T/\delta)+1)\tau^*\log T}{\beta\sqrt{T}}.
\end{aligned}$$

For $\boldsymbol{\theta}_0,\boldsymbol{\theta}^*\in\boldsymbol{\Theta}$, again by $\omega=Cm^{-1/2}L^{-9/4}$, we have $\|\boldsymbol{\theta}_0-\boldsymbol{\theta}^*\|_2^2\leq 1/m$. Since $\hat{f}(\cdot)\in\mathcal{F}_{\boldsymbol{\Theta},m}$, by Lemma 6.6.1, it holds with probability at least $1-2\delta-3L^2\exp(-C_0m^{2/3}L)$ over the randomness of the initial point $\boldsymbol{\theta}_0$ that

$$\frac{1}{T}\sum_{t=0}^{T-1}\mathbb{E}\big[\big(\hat{f}(\boldsymbol{\theta}_t)-\hat{f}(\boldsymbol{\theta}^*)\big)^2\big|\boldsymbol{\theta}_0\big]\leq\frac{1}{\sqrt{T}}+\frac{C_1\tau^*\log(T/\delta)\log T}{\beta^2\sqrt{T}}+\frac{C_2\log m\log(T/\delta)}{\beta m^{1/6}},$$

where we used the fact that $\gamma<1$. This completes the proof. □

6.7 Proof of Theorem 6.5.6

Before we prove the global convergence of Algorithm 10, we present the following lemma that shows that near the initialization point $\boldsymbol{\theta}_0$, the neural network function $f(\boldsymbol{\theta};\mathbf{x})$ is almost linear in $\boldsymbol{\theta}$ for all unit input vectors.

Lemma 6.7.1 (Theorems 5.3 and 5.4 in [CG19b]). *Let* $\boldsymbol{\theta}_0=(\mathbf{W}_0^{(1)\top},\ldots,\mathbf{W}_0^{(L)\top})^\top$ *be the initial point and* $\boldsymbol{\theta}=(\mathbf{W}^{(1)\top},\ldots,\mathbf{W}^{(L)\top})^\top\in\mathbb{B}(\boldsymbol{\theta}_0,\omega)$ *be a point in the neighborhood of* $\boldsymbol{\theta}_0$*. If*

$$m\geq C_1\max\{dL^2\log(m/\delta),\omega^{-4/3}L^{-8/3}\log(m/(\omega\delta))\},\quad \textit{and } \omega\leq C_2L^{-5}(\log m)^{-3/2},$$

then for all $\mathbf{x} \in S^{d-1}$*, with probability at least* $1-\delta$ *it holds that*

$$|f(\boldsymbol{\theta};\mathbf{x}) - \hat{f}(\boldsymbol{\theta};\mathbf{x})| \le \omega^{1/3}L^{8/3}\sqrt{m\log m}\sum_{l=1}^{L}\big\|\mathbf{W}^{(l)} - \mathbf{W}_0^{(l)}\big\|_2 + C_3L^3\sqrt{m}\sum_{l=1}^{L}\big\|\mathbf{W}^{(l)} - \mathbf{W}_0^{(l)}\big\|_2^2.$$

Under the same conditions on m *and* ω*, if* $\boldsymbol{\theta}_t \in \mathbb{B}(\boldsymbol{\theta}_0, \omega)$ *for all* $t = 1, \ldots, T$*, then with probability at least* $1-\delta$*, we have* $|f(\boldsymbol{\theta}_t;\phi(s_t,a_t))| \le C_4\sqrt{\log(T/\delta)}$ *for all* $t \in [T]$*.*

Proof of Theorem 6.5.6. By triangle inequality, it holds that

$$\begin{aligned} Q(s,a;\boldsymbol{\theta}_T) - Q^*(s,a) \le f(\boldsymbol{\theta}_T;s,a) - \hat{f}(\boldsymbol{\theta}_T;s,a) + \hat{f}(\boldsymbol{\theta}_T;s,a) - \hat{f}(\boldsymbol{\theta}^*;s,a) \\ + \hat{f}(\boldsymbol{\theta}^*;s,a) - Q^*(s,a). \end{aligned} \tag{6.7.1}$$

Recall that $\hat{f}(\boldsymbol{\theta}^*;\cdot,\cdot)$ is the fixed point of $\Pi_{\mathcal{F}}\mathcal{T}$ and $Q^*(\cdot,\cdot)$ is the fixed point of $\mathcal{T}$. Then we have

$$\begin{aligned} \big|\hat{f}(\boldsymbol{\theta}^*;s,a) - Q^*(s,a)\big| &= \big|\hat{f}(\boldsymbol{\theta}^*;s,a) - \Pi_{\mathcal{F}_{\boldsymbol{\Theta},m}}Q^*(s,a) + \Pi_{\mathcal{F}_{\boldsymbol{\Theta},m}}Q^*(s,a) - Q^*(s,a)\big| \\ &= \big|\Pi_{\mathcal{F}_{\boldsymbol{\Theta},m}}\mathcal{T}\hat{f}(\boldsymbol{\theta}^*;s,a) - \Pi_{\mathcal{F}_{\boldsymbol{\Theta},m}}\mathcal{T}Q^*(s,a) + \Pi_{\mathcal{F}_{\boldsymbol{\Theta},m}}Q^*(s,a) - Q^*(s,a)\big| \\ &\le \big|\Pi_{\mathcal{F}_{\boldsymbol{\Theta},m}}\mathcal{T}\hat{f}(\boldsymbol{\theta}^*;s,a) - \Pi_{\mathcal{F}_{\boldsymbol{\Theta},m}}\mathcal{T}Q^*(s,a)\big| + \big|\Pi_{\mathcal{F}_{\boldsymbol{\Theta},m}}Q^*(s,a) - Q^*(s,a)\big| \\ &\le \gamma|\hat{f}(\boldsymbol{\theta}^*;s,a) - Q^*(s,a)| + \big|\Pi_{\mathcal{F}_{\boldsymbol{\Theta},m}}Q^*(s,a) - Q^*(s,a)\big|, \end{aligned}$$

where the first inequality follows the triangle inequality and in the second inequality we used the fact that $\Pi_{\mathcal{F}_{\boldsymbol{\Theta},m}}\mathcal{T}$ is γ-contractive. This further leads to

$$(1-\gamma)|\hat{f}(\boldsymbol{\theta}^*;s,a) - Q^*(s,a)| \le |\Pi_{\mathcal{F}_{\boldsymbol{\Theta},m}}Q^*(s,a) - Q^*(s,a)|.$$

To simplify the notation, we abbreviate $\mathbb{E}[\cdot|\boldsymbol{\theta}_0]$ as $\mathbb{E}[\cdot]$ in the rest of this proof. Therefore, we have

$$\begin{aligned} &\mathbb{E}\big[(Q(s,a;\boldsymbol{\theta}_T) - Q^*(s,a))^2\big] \\ &\le 3\mathbb{E}\big[\big(f(\boldsymbol{\theta}_T;s,a) - \hat{f}(\boldsymbol{\theta}_T;s,a)\big)^2\big] + 3\mathbb{E}\big[\big(\hat{f}(\boldsymbol{\theta}_T;s,a) - \hat{f}(\boldsymbol{\theta}^*;s,a)\big)^2\big] \\ &\qquad + 3\mathbb{E}\big[\big(\hat{f}(\boldsymbol{\theta}^*;s,a) - Q^*(s,a)\big)^2\big] \\ &\le 3\mathbb{E}\big[\big(f(\boldsymbol{\theta}_T;s,a) - \hat{f}(\boldsymbol{\theta}_T;s,a)\big)^2\big] + 3\mathbb{E}\big[\big(\hat{f}(\boldsymbol{\theta}_T;s,a) - \hat{f}(\boldsymbol{\theta}^*;s,a)\big)^2\big] \\ &\qquad + 3(1-\gamma)^{-2}\mathbb{E}\big[\big(\Pi_{\mathcal{F}_{\boldsymbol{\Theta},m}}Q^*(s,a) - Q^*(s,a)\big)^2\big]. \end{aligned}$$

By Lemma 6.7.1 and the parameter choice that $\omega = C_1/(\sqrt{m}L^{9/4})$, we have

$$\mathbb{E}[(f(\boldsymbol{\theta}_T; s, a) - \hat{f}(\boldsymbol{\theta}_T; s, a))^2] \leq C_2(\omega^{4/3}L^4\sqrt{m\log m})^2 \leq C_1^{4/3}C_2m^{-1/3}\log m$$

with probability at least $1-\delta$. Combining the above result with Theorem 6.5.4, we have

$$\begin{aligned}\mathbb{E}\big[(Q(s,a;\boldsymbol{\theta}_T) - Q^*(s,a))^2\big] \leq \frac{3\mathbb{E}\big[\big(\Pi_{\mathcal{F}_{\boldsymbol{\Theta},m}}Q^*(s,a) - Q^*(s,a)\big)^2\big]}{(1-\gamma)^2} + \frac{1}{\sqrt{T}} \\ + \frac{C_2\tau^*\log(T/\delta)\log T}{\beta^2\sqrt{T}} + \frac{C_3\log(T/\delta)\log m}{\beta m^{1/6}},\end{aligned}$$

with probability at least $1 - 3\delta - L^2\exp(-C_6m^{2/3}L)$, which completes the proof. $\square$

6.8 Proof of Supporting Lemmas

6.8.1 Proof of Lemma 6.6.1

Before we prove the error bound for the local linearization, we first present some useful lemmas from recent studies of overparameterized deep neural networks. Note that in the following lemmas, $\{C_i\}_{i=1,\ldots}$ are universal constants that are independent of problem parameters such as $d, \boldsymbol{\theta}, m, L$ and their values can be different in different contexts. The first lemma states the uniform upper bound for the gradient of the deep neural network. Note that by definition, our parameter $\boldsymbol{\theta}$ is a long vector containing the concatenation of the vectorization of all the weight matrices. Correspondingly, the gradient $\nabla_{\boldsymbol{\theta}}f(\boldsymbol{\theta};\mathbf{x})$ is also a long vector.

Lemma 6.8.1 (Lemma B.3 in [CG19a])**.** *Let $\boldsymbol{\theta} \in \mathbb{B}(\boldsymbol{\theta}_0, \omega)$ where ω is the radius satisfying $C_1d^{3/2}L^{-1}m^{-3/2} \leq \omega \leq C_2L^{-6}(\log m)^{-3/2}$. Then for all unit vectors in $\mathbb{R}^d$, i.e., $\mathbf{x} \in S^{d-1}$, the gradient of the neural network f defined in* (6.4.2) *is bounded as $\|\nabla_{\boldsymbol{\theta}}f(\boldsymbol{\theta};\mathbf{x})\|_2 \leq C_3\sqrt{m}$ with probability at least $1 - L^2\exp(-C_4m\omega^{2/3}L)$.*

The second lemma provides the perturbation bound for the gradient of the neural network function. Note that the original theorem holds for any fixed d dimensional unit vector $\mathbf{x}$. However, due to the choice of ω and its dependency on m and d, it is easy to modify the results to hold for all $\mathbf{x} \in S^{d-1}$.

Lemma 6.8.2 (Theorem 5 in [AZLS19]). *Let $\boldsymbol{\theta} \in \mathbb{B}(\boldsymbol{\theta}_0, \omega)$ with the radius satisfying*

$$C_1 d^{3/2} L^{-3/2} m^{-3/2} (\log m)^{-3/2} \le \omega \le C_2 L^{-9/2} (\log m)^{-3}.$$

Then for all $\mathbf{x} \in S^{d-1}$, with probability at least $1 - \exp(-C_3 m\omega^{2/3} L)$ over the randomness of $\boldsymbol{\theta}_0$, it holds that

$$\|\nabla_{\boldsymbol{\theta}} f(\boldsymbol{\theta}; \mathbf{x}) - \nabla_{\boldsymbol{\theta}} f(\boldsymbol{\theta}_0; \mathbf{x})\|_2 \le C_4 \omega^{1/3} L^3 \sqrt{\log m} \|\nabla_{\boldsymbol{\theta}} f(\boldsymbol{\theta}_0; \mathbf{x})\|_2.$$

Now we are ready to bound the linearization error.

Proof of Lemma 6.6.1. Recall the definition of $\mathbf{g}_t(\boldsymbol{\theta}_t)$ and $\mathbf{m}_t(\boldsymbol{\theta}_t)$ in (6.4.5) and (6.6.2) respectively. We have

$$\begin{aligned}
\|\mathbf{g}_t(\boldsymbol{\theta}_t) - \mathbf{m}_t(\boldsymbol{\theta}_t)\|_2 &= \big\|\nabla_{\boldsymbol{\theta}} f(\boldsymbol{\theta}_t; s_t, a_t)\Delta(s_t, a_t, s_{t+1}; \boldsymbol{\theta}_t) - \nabla_{\boldsymbol{\theta}} \hat{f}(\boldsymbol{\theta}_t; s_t, a_t)\hat{\Delta}(s_t, a_t, s_{t+1}; \boldsymbol{\theta}_t)\big\|_2 \\
&\le \big\|(\nabla_{\boldsymbol{\theta}} f(\boldsymbol{\theta}_t; s_t, a_t) - \nabla_{\boldsymbol{\theta}} \hat{f}(\boldsymbol{\theta}_t; s_t, a_t))\Delta(s_t, a_t, s_{t+1}; \boldsymbol{\theta}_t)\big\|_2 \\
&\quad + \big\|\nabla_{\boldsymbol{\theta}} \hat{f}(\boldsymbol{\theta}_t; s_t, a_t)\big(\Delta(s_t, a_t, s_{t+1}; \boldsymbol{\theta}_t) - \hat{\Delta}(s_t, a_t, s_{t+1}; \boldsymbol{\theta}_t)\big)\big\|_2. \quad (6.8.1)
\end{aligned}$$

Since $\hat{f}(\boldsymbol{\theta}) \in \mathcal{F}_{\boldsymbol{\Theta},m}$, we have $\hat{f}(\boldsymbol{\theta}) = f(\boldsymbol{\theta}_0) + \langle \nabla_{\boldsymbol{\theta}} f(\boldsymbol{\theta}_0), \boldsymbol{\theta} - \boldsymbol{\theta}_0\rangle$ and $\nabla_{\boldsymbol{\theta}} \hat{f}(\boldsymbol{\theta}) = \nabla_{\boldsymbol{\theta}} f(\boldsymbol{\theta}_0)$. Then with probability at least $1 - 2L^2 \exp(-C_1 m\omega^{2/3} L)$, we have

$$\begin{aligned}
&\big\|(\nabla_{\boldsymbol{\theta}} f(\boldsymbol{\theta}_t; s_t, a_t) - \nabla_{\boldsymbol{\theta}} \hat{f}(\boldsymbol{\theta}_t; s_t, a_t))\Delta(s_t, a_t, s_{t+1}; \boldsymbol{\theta}_t)\big\|_2 \\
&= |\Delta(s_t, a_t, s_{t+1}; \boldsymbol{\theta}_t)| \cdot \big\|(\nabla_{\boldsymbol{\theta}} f(\boldsymbol{\theta}_t; s_t, a_t) - \nabla_{\boldsymbol{\theta}} f(\boldsymbol{\theta}_0; s_t, a_t))\big\|_2 \\
&\le C_2 \omega^{1/3} L^3 \sqrt{m \log m} |\Delta(s_t, a_t, s_{t+1}; \boldsymbol{\theta}_t)|,
\end{aligned}$$

where the inequality comes from Lemmas 6.8.1 and 6.8.2. By Lemma 6.7.1, with probability at least $1 - \delta$, it holds that

$$|\Delta(s_t, a_t, s_{t+1}; \boldsymbol{\theta}_t)| = \Big|f(\boldsymbol{\theta}_t; s_t, a_t) - r_t - \gamma \max_{b \in \mathcal{A}} f(\boldsymbol{\theta}_t; s_{t+1}, b)\Big| \le (2+\gamma) C_3 \sqrt{\log(T/\delta)},$$

which further implies that with probability at least $1 - \delta - 2L^2 \exp(-C_1 m\omega^{2/3} L)$, we have

$$\begin{aligned}
&\big\|(\nabla_{\boldsymbol{\theta}} f(\boldsymbol{\theta}_t; s_t, a_t) - \nabla_{\boldsymbol{\theta}} \hat{f}(\boldsymbol{\theta}_t; s_t, a_t))\Delta(s_t, a_t, s_{t+1}; \boldsymbol{\theta}_t)\big\|_2 \\
&\le (2+\gamma) C_2 C_3 \omega^{1/3} L^3 \sqrt{m \log m \log(T/\delta)}.
\end{aligned}$$

For the second term in (6.8.1), we have

$$\begin{aligned}
&\big\|\nabla_{\boldsymbol{\theta}} \hat{f}(\boldsymbol{\theta}_t; s_t, a_t)\big(\Delta(s_t, a_t, s_{t+1}; \boldsymbol{\theta}_t) - \hat{\Delta}(s_t, a_t, s_{t+1}; \boldsymbol{\theta}_t)\big)\big\|_2 \\
&\le \big\|\nabla_{\boldsymbol{\theta}} \hat{f}(\boldsymbol{\theta}_t; s_t, a_t)\big(f(\boldsymbol{\theta}_t; s_t, a_t) - \hat{f}(\boldsymbol{\theta}_t; s_t, a_t)\big)\big\|_2 \\
&\qquad + \Big\|\nabla_{\boldsymbol{\theta}} \hat{f}(\boldsymbol{\theta}_t; s_t, a_t)\Big(\max_{b\in\mathcal{A}} f(\boldsymbol{\theta}_t; s_{t+1}, b) - \max_{b\in\mathcal{A}} \hat{f}(\boldsymbol{\theta}_t; s_{t+1}, b)\Big)\Big\|_2 \\
&\le \big\|\nabla_{\boldsymbol{\theta}} \hat{f}(\boldsymbol{\theta}_t; s_t, a_t)\big\|_2 \cdot \big|f(\boldsymbol{\theta}_t; s_t, a_t) - \hat{f}(\boldsymbol{\theta}_t; s_t, a_t)\big| \\
&\qquad + \|\nabla_{\boldsymbol{\theta}} \hat{f}(\boldsymbol{\theta}_t; s_t, a_t)\|_2 \max_{b\in\mathcal{A}} \big|f(\boldsymbol{\theta}_t; s_{t+1}, b) - \hat{f}(\boldsymbol{\theta}; s_{t+1}, b)\big|. \qquad (6.8.2)
\end{aligned}$$

By Lemma 6.7.1, with probability at least $1-\delta$ we have

$$|f(\boldsymbol{\theta}_t; s_t, a_t) - \hat{f}(\boldsymbol{\theta}_t; s_t, a_t)| \le \omega^{4/3} L^{11/3} \sqrt{m \log m} + C_4 \omega^2 L^4 \sqrt{m},$$

for all $(s_t, a_t) \in \mathcal{S} \times \mathcal{A}$ such that $\|\phi(s_t, a_t)\|_2 = 1$. Substituting the above result into (6.8.2) and applying the gradient bound in Lemma 6.8.1, we obtain with probability at least $1 - \delta - L^2 \exp(-C_1 m \omega^{2/3} L)$ that

$$\begin{aligned}
&\big\|\nabla_{\boldsymbol{\theta}} \hat{f}(\boldsymbol{\theta}_t; s_t, a_t)\big(\Delta(s_t, a_t, s_{t+1}; \boldsymbol{\theta}_t) - \hat{\Delta}(s_t, a_t, s_{t+1}; \boldsymbol{\theta}_t)\big)\big\|_2 \\
&\le C_5 \omega^{4/3} L^{11/3} m \sqrt{\log m} + C_6 \omega^2 L^4 m.
\end{aligned}$$

Note that the above results require that the choice of ω should satisfy all the constraints in Lemmas 6.8.1, 6.7.1 and 6.8.2, of which the intersection is

$$C_7 d^{3/2} L^{-1} m^{-3/4} \le \omega \le C_8 L^{-6} (\log m)^{-3}.$$

Therefore, the error of the local linearization of $\mathbf{g}_t(\boldsymbol{\theta}_t)$ can be upper bounded by

$$\begin{aligned}
|\langle \mathbf{g}(\boldsymbol{\theta}_t) - \mathbf{m}(\boldsymbol{\theta}_t), \boldsymbol{\theta}_t - \boldsymbol{\theta}^* \rangle| &\le (2+\gamma) C_2 C_3 \omega^{1/3} L^3 \sqrt{m \log m \log(T/\delta)} \|\boldsymbol{\theta}_t - \boldsymbol{\theta}^*\|_2 \\
&\qquad + \big(C_5 \omega^{4/3} L^{11/3} m \sqrt{\log m} + C_6 \omega^2 L^4 m\big) \|\boldsymbol{\theta}_t - \boldsymbol{\theta}^*\|_2,
\end{aligned}$$

which holds with probability at least $1 - 2\delta - 3L^2 \exp(-C_1 m \omega^{2/3} L)$ over the randomness of the initial point. For the upper bound of the norm of $\mathbf{g}_t$, by Lemmas 6.8.1 and 6.7.1, we have

$$\|\mathbf{g}_t\|_2 = \Big\|\nabla_{\boldsymbol{\theta}} f(\boldsymbol{\theta}_t; s_t, a_t)\Big(f(\boldsymbol{\theta}_t; s_t, a_t) - r_t - \gamma \max_{b\in\mathcal{A}} f(\boldsymbol{\theta}_t; s_{t+1}, b)\Big)\Big\|_2 \le (2+\gamma) C_9 \sqrt{m \log(T/\delta)}$$

holds with probability at least $1 - \delta - L^2 \exp(-C_1 m \omega^{2/3} L)$. □

6.8.2 Proof of Lemma 6.6.2

Let us define $\zeta_t(\boldsymbol{\theta}) = \langle \mathbf{m}_t(\boldsymbol{\theta}) - \overline{\mathbf{m}}(\boldsymbol{\theta}), \boldsymbol{\theta} - \boldsymbol{\theta}^* \rangle$, which characterizes the bias of the data. Different from the similar quantity ζ_t in [BRS18], our definition is based on the local linearization of f, which is essential to the analysis in our proof. It is easy to verify that $\mathbb{E}[\mathbf{m}_t(\boldsymbol{\theta})] = \overline{\mathbf{m}}(\boldsymbol{\theta})$ for any fixed and deterministic $\boldsymbol{\theta}$. However, it should be noted that $\mathbb{E}[\mathbf{m}_t(\boldsymbol{\theta}_t)|\boldsymbol{\theta}_t = \boldsymbol{\theta}] \neq \overline{\mathbf{m}}(\boldsymbol{\theta})$ because $\boldsymbol{\theta}_t$ depends on all historical states and actions $\{s_t, a_t, s_{t-1}, a_{t-1}, \ldots\}$ and $\mathbf{m}_t(\cdot)$ depends on the current observation $\{s_t, a_t, s_{t+1}\}$ and thus also depends on $\{s_{t-1}, a_{t-1}, s_{t-2}, a_{t-2}, \ldots\}$. Therefore, we need a careful analysis of Markov chains to decouple the dependency between $\boldsymbol{\theta}_t$ and $\mathbf{m}_t(\cdot)$.

The following lemma uses data processing inequality to provide an information theoretic control of coupling.

Lemma 6.8.3 (Control of coupling, [BRS18]). *Consider two random variables X and Y that form the following Markov chain:*

$$X \rightarrow s_t \rightarrow s_{t+\tau} \rightarrow Y,$$

where $t \in \{0, 1, 2, \ldots\}$ and $\tau > 0$. Suppose Assumption 6.5.2 holds. Let X' and Y' be independent copies drawn from the marginal distributions of X and Y respectively, i.e., $\mathbb{P}(X' = \cdot, Y' = \cdot) = \mathbb{P}(X = \cdot) \otimes \mathbb{P}(Y = \cdot)$. Then for any bounded function $h : \mathcal{S} \times \mathcal{S} \rightarrow \mathbb{R}$, it holds that

$$|\mathbb{E}[h(X, Y)] - \mathbb{E}[h(X', Y')]| \leq 2 \sup_{s,s'} |h(s, s')| \lambda \rho^{\tau}.$$

Proof of Lemma 6.6.2. The proof of this lemma is adapted from [BRS18], where the result was originally proved for linear function approximation of temporal difference learning. We first show that $\zeta_t(\boldsymbol{\theta})$ is Lipschitz. For any $\boldsymbol{\theta}, \boldsymbol{\theta}' \in \mathbb{B}(\boldsymbol{\theta}_0, \omega)$, we have

$$\begin{aligned}
\zeta_t(\boldsymbol{\theta}) - \zeta_t(\boldsymbol{\theta}') &= \langle \mathbf{m}_t(\boldsymbol{\theta}) - \overline{\mathbf{m}}(\boldsymbol{\theta}), \boldsymbol{\theta} - \boldsymbol{\theta}^* \rangle - \langle \mathbf{m}_t(\boldsymbol{\theta}') - \overline{\mathbf{m}}(\boldsymbol{\theta}'), \boldsymbol{\theta}' - \boldsymbol{\theta}^* \rangle \\
&= \langle \mathbf{m}_t(\boldsymbol{\theta}) - \overline{\mathbf{m}}(\boldsymbol{\theta}) - (\mathbf{m}_t(\boldsymbol{\theta}') - \overline{\mathbf{m}}(\boldsymbol{\theta}')), \boldsymbol{\theta} - \boldsymbol{\theta}^* \rangle \\
&\quad + \langle \mathbf{m}_t(\boldsymbol{\theta}') - \overline{\mathbf{m}}(\boldsymbol{\theta}'), \boldsymbol{\theta} - \boldsymbol{\theta}' \rangle,
\end{aligned}$$

which directly implies

$$|\zeta_t(\boldsymbol{\theta}) - \zeta_t(\boldsymbol{\theta}')| \le \|\mathbf{m}_t(\boldsymbol{\theta}) - \mathbf{m}_t(\boldsymbol{\theta}')\|_2 \cdot \|\boldsymbol{\theta} - \boldsymbol{\theta}^*\|_2 + \|\overline{\mathbf{m}}(\boldsymbol{\theta}) - \overline{\mathbf{m}}(\boldsymbol{\theta}')\|_2 \cdot \|\boldsymbol{\theta} - \boldsymbol{\theta}^*\|_2$$
$$+ \|\mathbf{m}_t(\boldsymbol{\theta}') - \overline{\mathbf{m}}(\boldsymbol{\theta}')\|_2 \cdot \|\boldsymbol{\theta} - \boldsymbol{\theta}'\|_2.$$

By the definition of $\mathbf{m}_t$, we have

$$\begin{aligned}
&\|\mathbf{m}_t(\boldsymbol{\theta}) - \mathbf{m}_t(\boldsymbol{\theta}')\|_2 \\
&= \Big\|\nabla_{\boldsymbol{\theta}} f(\boldsymbol{\theta}_0)\Big(\big(f(\boldsymbol{\theta}; s, a) - f(\boldsymbol{\theta}'; s, a)\big) - \gamma\Big(\max_{b\in\mathcal{A}} f(\boldsymbol{\theta}; s', b) - \max_{b\in\mathcal{A}} f(\boldsymbol{\theta}'; s', b)\Big)\Big)\Big\|_2 \\
&\le (1+\gamma)C_3^2 m\|\boldsymbol{\theta} - \boldsymbol{\theta}'\|_2,
\end{aligned}$$

which holds with probability at least $1 - L^2\exp(-C_4 m\omega^{2/3}L)$, where we used the fact that the neural network function is Lipschitz with parameter $C_3\sqrt{m}$ by Lemma 6.8.1. Similar bound can also be established for $\|\overline{\mathbf{m}}_t(\boldsymbol{\theta}) - \overline{\mathbf{m}}_t(\boldsymbol{\theta}')\|$ in the same way. Note that for $\boldsymbol{\theta} \in \mathbb{B}(\boldsymbol{\theta}_0, \omega)$ with ω and m satisfying the conditions in Lemma 6.6.1, we have by the definition in (6.6.2) that

$$\begin{aligned}
\|\mathbf{m}_t(\boldsymbol{\theta})\|_2 &\le \Big(|\hat{f}(\boldsymbol{\theta}; s, a)| + r(s,a) + \gamma\big|\max_b \hat{f}(\boldsymbol{\theta}; s', b)\big|\Big)\|\nabla_{\boldsymbol{\theta}}\hat{f}(\boldsymbol{\theta})\|_2 \\
&\le 2(2+\gamma)(|f(\boldsymbol{\theta}_0)| + \|\nabla_{\boldsymbol{\theta}} f(\boldsymbol{\theta}_0)\|_2 \cdot \|\boldsymbol{\theta} - \boldsymbol{\theta}_0\|_2)\|\nabla_{\boldsymbol{\theta}} f(\boldsymbol{\theta}_0)\|_2 \\
&\le 2(2+\gamma)C_3(C_8\sqrt{m}\sqrt{\log(T/\delta)} + C_3 m\omega).
\end{aligned}$$

The same bound can be established for $\|\bar{\mathbf{m}}_t\|$ in a similar way. Therefore, we have $|\zeta_t(\boldsymbol{\theta}) - \zeta_t(\boldsymbol{\theta}')| \le \ell_{m,L}\|\boldsymbol{\theta} - \boldsymbol{\theta}'\|_2$, where $\ell_{m,L}$ is defined as

$$\ell_{m,L} = 2(1+\gamma)C_3^2 m\omega + 2(2+\gamma)C_3(C_8\sqrt{m}\sqrt{\log(T/\delta)} + C_3 m\omega).$$

Applying the above inequality recursively, for all $\tau = 0, \ldots, t$, we have

$$\begin{aligned}
\zeta_t(\boldsymbol{\theta}_t) &\le \zeta_t(\boldsymbol{\theta}_{t-\tau}) + \ell_{m,L}\sum_{i=t-\tau}^{t-1}\|\boldsymbol{\theta}_{i+1} - \boldsymbol{\theta}_i\|_2 \\
&\le \zeta_t(\boldsymbol{\theta}_{t-\tau}) + 2(2+\gamma)C_3(C_8\sqrt{m}\sqrt{\log(T/\delta)} + C_3 m\omega)\ell_{m,L}\sum_{i=t-\tau}^{t-1}\eta_i. \qquad (6.8.3)
\end{aligned}$$

Next, we need to bound $\zeta_t(\boldsymbol{\theta}_{t-\tau})$. Define the observed tuple $O_t = (s_t, a_t, s_{t+1})$ as the collection of the current state and action and the next state. Note that $\boldsymbol{\theta}_{t-\tau} \to s_{t-\tau} \to s_t \to O_t$ forms a

Markov chain induced by the target policy π. Recall that $\mathbf{m}_t(\cdot)$ depends on the observation O_t. Let's rewrite $\mathbf{m}(\boldsymbol{\theta}, O_t) = \mathbf{m}_t(\boldsymbol{\theta})$. Similarly, we can rewrite $\zeta_t(\boldsymbol{\theta})$ as $\zeta(\boldsymbol{\theta}, O_t)$. Let $\boldsymbol{\theta}'_t$ and O'_t be independently drawn from the marginal distributions of $\boldsymbol{\theta}_t$ and O_t respectively. Applying Lemma 6.8.3 yields

$$\mathbb{E}[\zeta(\boldsymbol{\theta}_{t-\tau}, O_t)] - \mathbb{E}[\zeta(\boldsymbol{\theta}'_{t-\tau}, O'_t)] \leq 2 \sup_{\boldsymbol{\theta}, O} |\zeta(\boldsymbol{\theta}, O)| \lambda \rho^{\tau},$$

where we used the uniform mixing result in Assumption 6.5.2. By definition $\boldsymbol{\theta}'_{t-\tau}$ and O'_t are independent, which implies $\mathbb{E}[\mathbf{m}(\boldsymbol{\theta}'_t, O'_t)|\boldsymbol{\theta}'_t] = \overline{\mathbf{m}}(\boldsymbol{\theta}'_t)$ and

$$\mathbb{E}[\zeta(\boldsymbol{\theta}'_{t-\tau}, O'_t)] = \mathbb{E}[\mathbb{E}[\langle \mathbf{m}(\boldsymbol{\theta}'_t, O'_t) - \overline{\mathbf{m}}(\boldsymbol{\theta}'_t), \boldsymbol{\theta}'_t - \boldsymbol{\theta}^* \rangle] | \boldsymbol{\theta}'_t] = 0.$$

Therefore, for any $\tau = 0, \ldots, t$, we have

$$\begin{aligned} \mathbb{E}[\zeta_t(\boldsymbol{\theta}_t)] &\leq \mathbb{E}\zeta_t(\boldsymbol{\theta}_{t-\tau}) + 2(2+\gamma) C_3 (C_8 \sqrt{m} \sqrt{\log(T/\delta)} + C_3 m \omega) \ell_{m,L} \sum_{i=t-\tau}^{t-1} \eta_i \\ &\leq 2 \sup \lambda \rho^{\tau} + 2(2+\gamma) C_3 (C_8 \sqrt{m} \sqrt{\log(T/\delta)} + C_3 m \omega) \ell_{m,L} \tau \eta_{t-\tau}. \end{aligned} \tag{6.8.4}$$

Define τ^* as the mixing time of the Markov chain that satisfies

$$\tau^* = \min\{t = 0, 1, 2, \ldots | \lambda \rho^t \leq \eta_T\}.$$

When $t \leq \tau^*$, we choose $\tau = t$ in (6.8.4) and obtain

$$\begin{aligned} \mathbb{E}[\zeta_t(\boldsymbol{\theta}_t)] &\leq \mathbb{E}[\zeta_t(\boldsymbol{\theta}_0)] + 2(2+\gamma) C_3 (C_8 \sqrt{m} \sqrt{\log(T/\delta)} + C_3 m \omega) \ell_{m,L} \tau^* \eta_0 \\ &= 2(2+\gamma) C_3 (C_8 \sqrt{m} \sqrt{\log(T/\delta)} + C_3 m \omega) \ell_{m,L} \tau^* \eta_0, \end{aligned}$$

where we used the fact that the initial point $\boldsymbol{\theta}_0$ is independent of $\{s_t, a_t, s_{t-1}, a_{t-1}, \ldots, s_0, a_0\}$ and thus independent of $\zeta_t(\cdot)$. When $t > \tau^*$, we can choose $\tau = \tau^*$ in (6.8.4) and obtain

$$\begin{aligned} \mathbb{E}[\zeta_t(\boldsymbol{\theta}_t)] &\leq 2\eta_T + 2(2+\gamma) C_3 (C_8 \sqrt{m} \sqrt{\log(T/\delta)} + C_3 m \omega) \ell_{m,L} \tau^* \eta_{t-\tau^*} \\ &\leq \tilde{C}(m \log(T/\delta) + m^2 \omega^2) \tau^* \eta_{t-\tau^*}, \end{aligned}$$

where $\tilde{C} > 0$ is a universal constant, which completes the proof. $\square$

6.8.3 Proof of Lemma 6.6.3

Proof of Lemma 6.6.3. To simplify the notation, we use $\mathbb{E}_\pi$ to denote $\mathbb{E}_{\mu,\pi,\mathcal{P}}$, namely, the expectation over $s \in \mu, a \sim \pi(\cdot|s)$ and $s' \sim \mathcal{P}(\cdot|s,a)$, in the rest of the proof. By the definition of $\overline{\mathbf{m}}$ in (6.6.2), we have

$$\begin{aligned}
&\langle \overline{\mathbf{m}}(\boldsymbol{\theta}) - \overline{\mathbf{m}}(\boldsymbol{\theta}^*), \boldsymbol{\theta} - \boldsymbol{\theta}^* \rangle \\
&= \mathbb{E}_\pi\big[\big(\hat{\Delta}(s,a,s';\boldsymbol{\theta}) - \hat{\Delta}(s,a,s';\boldsymbol{\theta}^*)\big)\langle \nabla_{\boldsymbol{\theta}} f(\boldsymbol{\theta}_0;s,a), \boldsymbol{\theta} - \boldsymbol{\theta}^* \rangle\big] \\
&= \mathbb{E}_\pi\big[\big(\hat{f}(\boldsymbol{\theta};s,a) - \hat{f}(\boldsymbol{\theta}^*;s,a)\big)\langle \nabla_{\boldsymbol{\theta}} f(\boldsymbol{\theta}_0;s,a), \boldsymbol{\theta} - \boldsymbol{\theta}^* \rangle\big] \\
&\quad - \gamma \mathbb{E}_\pi\Big[\Big(\max_{b\in\mathcal{A}} \hat{f}(\boldsymbol{\theta};s',b) - \max_{b\in\mathcal{A}} \hat{f}(\boldsymbol{\theta}^*;s',b)\Big)\langle \nabla_{\boldsymbol{\theta}} f(\boldsymbol{\theta}_0;s,a), \boldsymbol{\theta} - \boldsymbol{\theta}^* \rangle\Big],
\end{aligned}$$

where in the first equation we used the fact that $\nabla_{\boldsymbol{\theta}} \hat{f}(\boldsymbol{\theta}) = \nabla_{\boldsymbol{\theta}} f(\boldsymbol{\theta}_0)$ for all $\boldsymbol{\theta} \in \boldsymbol{\Theta}$ and $\hat{f} \in \mathcal{F}_{\boldsymbol{\Theta},m}$. Further by the property of the local linearization of f at $\boldsymbol{\theta}_0$, we have

$$\hat{f}(\boldsymbol{\theta};s,a) - \hat{f}(\boldsymbol{\theta}^*;s,a) = \langle \nabla_{\boldsymbol{\theta}} f(\boldsymbol{\theta}_0;s,a), \boldsymbol{\theta} - \boldsymbol{\theta}^* \rangle, \tag{6.8.5}$$

which further implies

$$\begin{aligned}
&\mathbb{E}\big[\big(\hat{f}(\boldsymbol{\theta};s,a) - \hat{f}(\boldsymbol{\theta}^*;s,a)\big)\langle \nabla_{\boldsymbol{\theta}} f(\boldsymbol{\theta}_0;s,a), \boldsymbol{\theta} - \boldsymbol{\theta}^* \rangle | \boldsymbol{\theta}_0\big] \\
&= (\boldsymbol{\theta} - \boldsymbol{\theta}^*)^\top \mathbb{E}\big[\nabla_{\boldsymbol{\theta}} f(\boldsymbol{\theta}_0;s,a) \nabla_{\boldsymbol{\theta}} f(\boldsymbol{\theta}_0;s,a)^\top | \boldsymbol{\theta}_0\big](\boldsymbol{\theta} - \boldsymbol{\theta}^*) \\
&= m\|\boldsymbol{\theta} - \boldsymbol{\theta}^*\|_{\boldsymbol{\Sigma}_\pi}^2.
\end{aligned}$$

where $\boldsymbol{\Sigma}_\pi$ is defined in Assumption 6.5.3. Let us define $b_{\max}(\boldsymbol{\theta}) = \mathrm{argmax}_{b\in\mathcal{A}} \hat{f}(\boldsymbol{\theta};s',b)$ and $b_{\max}(\boldsymbol{\theta}^*) = \mathrm{argmax}_{b\in\mathcal{A}} \hat{f}(\boldsymbol{\theta}^*;s',b)$. Then for the remaining term, we have

$$\begin{aligned}
&\mathbb{E}_\pi\Big[\Big(\max_{b\in\mathcal{A}} \hat{f}(\boldsymbol{\theta};s',b) - \max_{b\in\mathcal{A}} \hat{f}(\boldsymbol{\theta}^*;s',b)\Big)\langle \nabla_{\boldsymbol{\theta}} f(\boldsymbol{\theta}_0;s,a), \boldsymbol{\theta} - \boldsymbol{\theta}^* \rangle\Big] \\
&= \mathbb{E}_\pi\big[\big(\hat{f}(\boldsymbol{\theta};s',b_{\max}) - \hat{f}(\boldsymbol{\theta}^*;s',b_{\max}^*)\big)\langle \nabla_{\boldsymbol{\theta}} f(\boldsymbol{\theta}_0;s,a), \boldsymbol{\theta} - \boldsymbol{\theta}^* \rangle\big].
\end{aligned} \tag{6.8.6}$$

For all (s,a,s'), when $\langle \nabla_{\boldsymbol{\theta}} f(\boldsymbol{\theta}_0;s,a), \boldsymbol{\theta} - \boldsymbol{\theta}^* \rangle \geq 0$, (6.8.6) can be upper bounded by

$$\begin{aligned}
&\big(\hat{f}(\boldsymbol{\theta};s',b_{\max}) - \hat{f}(\boldsymbol{\theta}^*;s',b_{\max}^*)\big)\langle \nabla_{\boldsymbol{\theta}} f(\boldsymbol{\theta}_0;s,a), \boldsymbol{\theta} - \boldsymbol{\theta}^* \rangle \\
&= \big(\hat{f}(\boldsymbol{\theta};s',b_{\max}) - \hat{f}(\boldsymbol{\theta}^*;s',b_{\max}) + \hat{f}(\boldsymbol{\theta}^*;s',b_{\max}) - \hat{f}(\boldsymbol{\theta}^*;s',b_{\max}^*)\big)\langle \nabla_{\boldsymbol{\theta}} f(\boldsymbol{\theta}_0;s,a), \boldsymbol{\theta} - \boldsymbol{\theta}^* \rangle \\
&\leq \big(\hat{f}(\boldsymbol{\theta};s',b_{\max}) - \hat{f}(\boldsymbol{\theta}^*;s',b_{\max})\big)\langle \nabla_{\boldsymbol{\theta}} f(\boldsymbol{\theta}_0;s,a), \boldsymbol{\theta} - \boldsymbol{\theta}^* \rangle
\end{aligned}$$

$$
\begin{aligned}
&= (\boldsymbol{\theta} - \boldsymbol{\theta}^*)^\top \nabla_{\boldsymbol{\theta}} f(\boldsymbol{\theta}_0; s', b_{\max}) \nabla_{\boldsymbol{\theta}} f(\boldsymbol{\theta}_0; s, a)^\top (\boldsymbol{\theta} - \boldsymbol{\theta}^*) \\
&\le |(\boldsymbol{\theta} - \boldsymbol{\theta}^*)^\top \nabla_{\boldsymbol{\theta}} f(\boldsymbol{\theta}_0; s', b_{\max})| \cdot |\nabla_{\boldsymbol{\theta}} f(\boldsymbol{\theta}_0; s, a)^\top (\boldsymbol{\theta} - \boldsymbol{\theta}^*)|,
\end{aligned}
$$

where the inequality comes from the optimality of $b^*_{\max}$ and the last equality follows the fact that $\hat{f}(\boldsymbol{\theta}; \cdot, \cdot)$ is linear. When $\langle \nabla_{\boldsymbol{\theta}} f(\boldsymbol{\theta}_0; s, a), \boldsymbol{\theta} - \boldsymbol{\theta}^* \rangle < 0$, using the same argument, we can upper bound (6.8.6) as follows

$$
\begin{aligned}
&\big(\hat{f}(\boldsymbol{\theta}; s', b_{\max}) - \hat{f}(\boldsymbol{\theta}^*; s', b^*_{\max})\big) \langle \nabla_{\boldsymbol{\theta}} f(\boldsymbol{\theta}_0; s, a), \boldsymbol{\theta} - \boldsymbol{\theta}^* \rangle \\
&\ = \big(\hat{f}(\boldsymbol{\theta}; s', b_{\max}) - \hat{f}(\boldsymbol{\theta}; s', b^*_{\max}) + \hat{f}(\boldsymbol{\theta}; s', b^*_{\max}) - \hat{f}(\boldsymbol{\theta}^*; s', b^*_{\max})\big) \langle \nabla_{\boldsymbol{\theta}} f(\boldsymbol{\theta}_0; s, a), \boldsymbol{\theta} - \boldsymbol{\theta}^* \rangle \\
&\ \le \big(\hat{f}(\boldsymbol{\theta}; s', b^*_{\max}) - \hat{f}(\boldsymbol{\theta}^*; s', b^*_{\max})\big) \langle \nabla_{\boldsymbol{\theta}} f(\boldsymbol{\theta}_0; s, a), \boldsymbol{\theta} - \boldsymbol{\theta}^* \rangle \\
&\ \le |(\boldsymbol{\theta} - \boldsymbol{\theta}^*)^\top \nabla_{\boldsymbol{\theta}} f(\boldsymbol{\theta}_0; s', b^*_{\max})| \cdot |\nabla_{\boldsymbol{\theta}} f(\boldsymbol{\theta}_0; s, a)^\top (\boldsymbol{\theta} - \boldsymbol{\theta}^*)|.
\end{aligned}
$$

Combining the above result, we have for all tuples (s, a, s') it holds that

$$
\begin{aligned}
&\big(\hat{f}(\boldsymbol{\theta}; s', b_{\max}) - \hat{f}(\boldsymbol{\theta}^*; s', b^*_{\max})\big) \langle \nabla_{\boldsymbol{\theta}} f(\boldsymbol{\theta}_0; s, a), \boldsymbol{\theta} - \boldsymbol{\theta}^* \rangle \\
&\le |(\boldsymbol{\theta} - \boldsymbol{\theta}^*)^\top \nabla_{\boldsymbol{\theta}} f(\boldsymbol{\theta}_0; s', b_{\max})| \cdot |\nabla_{\boldsymbol{\theta}} f(\boldsymbol{\theta}_0; s, a)^\top (\boldsymbol{\theta} - \boldsymbol{\theta}^*)| \mathbb{1}^+ \\
&\qquad + |(\boldsymbol{\theta} - \boldsymbol{\theta}^*)^\top \nabla_{\boldsymbol{\theta}} f(\boldsymbol{\theta}_0; s', b^*_{\max})| \cdot |\nabla_{\boldsymbol{\theta}} f(\boldsymbol{\theta}_0; s, a)^\top (\boldsymbol{\theta} - \boldsymbol{\theta}^*)| \mathbb{1}^-,
\end{aligned}
$$

where we denote $\mathbb{1}^+ = \mathbb{1}\{\langle \nabla_{\boldsymbol{\theta}} f(\boldsymbol{\theta}_0; s, a), \boldsymbol{\theta} - \boldsymbol{\theta}^* \rangle \ge 0\}$ and $\mathbb{1}^- = \mathbb{1}\{\langle \nabla_{\boldsymbol{\theta}} f(\boldsymbol{\theta}_0; s, a), \boldsymbol{\theta} - \boldsymbol{\theta}^* \rangle < 0\}$. Taking expectation over the above inequality and applying Cauchy-Schwarz inequality, we have

$$
\begin{aligned}
&\mathbb{E}_{\mu,\pi,\mathcal{P}}\big[\big(\hat{f}(\boldsymbol{\theta}; s', b_{\max}) - \hat{f}(\boldsymbol{\theta}^*; s', b^*_{\max})\big) \langle \nabla_{\boldsymbol{\theta}} f(\boldsymbol{\theta}_0; s, a), \boldsymbol{\theta} - \boldsymbol{\theta}^* \rangle\big] \\
&\le \sqrt{\mathbb{E}_\pi\big[\big(\max_b |(\boldsymbol{\theta} - \boldsymbol{\theta}^*)^\top \nabla_{\boldsymbol{\theta}} f(\boldsymbol{\theta}_0; s', b)|\big)^2\big]} \sqrt{\mathbb{E}_\pi\big[\big(\nabla_{\boldsymbol{\theta}} f(\boldsymbol{\theta}_0; s, a)^\top (\boldsymbol{\theta} - \boldsymbol{\theta}^*)\big)^2\big]} \\
&= m\|\boldsymbol{\theta} - \boldsymbol{\theta}^*\|_{\boldsymbol{\Sigma}^*_\pi(\boldsymbol{\theta} - \boldsymbol{\theta}^*)} \|\boldsymbol{\theta} - \boldsymbol{\theta}^*\|_{\boldsymbol{\Sigma}_\pi},
\end{aligned}
$$

where we used the fact that $\boldsymbol{\Sigma}^*_\pi(\boldsymbol{\theta} - \boldsymbol{\theta}^*) = 1/m \mathbb{E}_\mu[\nabla_{\boldsymbol{\theta}} f(\boldsymbol{\theta}_0; s, \tilde{b}_{\max}) \nabla_{\boldsymbol{\theta}} f(\boldsymbol{\theta}_0; s, \tilde{b}_{\max})^\top]$ and $\tilde{b}_{\max} = \operatorname{argmax}_{b \in \mathcal{A}} |\langle \nabla_{\boldsymbol{\theta}} f(\boldsymbol{\theta}_0; s, b), \boldsymbol{\theta} - \boldsymbol{\theta}^* \rangle|$ according to (6.5.6). Substituting the above results into (6.8.6), we obtain

$$
\begin{aligned}
&\mathbb{E}_\pi\big[\big(\max_{b \in \mathcal{A}} \hat{f}(\boldsymbol{\theta}; s', b) - \max_{b \in \mathcal{A}} \hat{f}(\boldsymbol{\theta}^*; s', b)\big) \langle \nabla_{\boldsymbol{\theta}} f(\boldsymbol{\theta}_0; s, a), \boldsymbol{\theta} - \boldsymbol{\theta}^* \rangle\big] \\
&\le m\|\boldsymbol{\theta} - \boldsymbol{\theta}^*\|_{\boldsymbol{\Sigma}^*_\pi(\boldsymbol{\theta} - \boldsymbol{\theta}^*)} \|\boldsymbol{\theta} - \boldsymbol{\theta}^*\|_{\boldsymbol{\Sigma}_\pi},
\end{aligned}
$$

which immediately implies

$$
\begin{aligned}
\langle \overline{\mathbf{m}}(\boldsymbol{\theta}) - \overline{\mathbf{m}}(\boldsymbol{\theta}^*), \boldsymbol{\theta} - \boldsymbol{\theta}^* \rangle &\geq m\|\boldsymbol{\theta} - \boldsymbol{\theta}^*\|_{\boldsymbol{\Sigma}_\pi} \cdot \Big(\|\boldsymbol{\theta} - \boldsymbol{\theta}^*\|_{\boldsymbol{\Sigma}_\pi} - \|\boldsymbol{\theta} - \boldsymbol{\theta}^*\|_{\boldsymbol{\Sigma}_\pi^*(\boldsymbol{\theta} - \boldsymbol{\theta}^*)} \Big) \\
&= m\|\boldsymbol{\theta} - \boldsymbol{\theta}^*\|_{\boldsymbol{\Sigma}_\pi} \cdot \frac{\|\boldsymbol{\theta} - \boldsymbol{\theta}^*\|_{\boldsymbol{\Sigma}_\pi}^2 - \gamma^2 \|\boldsymbol{\theta} - \boldsymbol{\theta}^*\|_{\boldsymbol{\Sigma}_\pi^*(\boldsymbol{\theta} - \boldsymbol{\theta}^*)}^2}{\|\boldsymbol{\theta} - \boldsymbol{\theta}^*\|_{\boldsymbol{\Sigma}_\pi} + \gamma \|\boldsymbol{\theta} - \boldsymbol{\theta}^*\|_{\boldsymbol{\Sigma}_\pi^*(\boldsymbol{\theta} - \boldsymbol{\theta}^*)}} \\
&\geq m(1 - \alpha^{-1/2})\|\boldsymbol{\theta} - \boldsymbol{\theta}^*\|_{\boldsymbol{\Sigma}_\pi}^2 \\
&= (1 - \alpha^{-1/2}) \mathbb{E}\big[\big(\hat{f}(\boldsymbol{\theta}) - \hat{f}(\boldsymbol{\theta}^*)\big)^2 | \boldsymbol{\theta}_0 \big],
\end{aligned}
$$

where the second inequality is due to Assumption 6.5.3 and the last equation is due to (6.8.5) and the definition of $\boldsymbol{\Sigma}_\pi$ in (6.5.5). □

CHAPTER 7

Conclusion

This thesis discussed how to find the first-order stationary point, the local optimum, and the global optimum in finite-sum nonconvex optimization problems, where the objective function is an average of loss functions over a finite or infinite dataset. We proposed sample-efficient optimization algorithms for these different settings and provided asymptotic analyses of their convergence rates and sample complexities, matching or outperforming previous state-of-the-art methods. We also generalized these sample-efficient optimization algorithms and their analyses to reinforcement learning problems, including policy gradient, actor-critic, and Q-learning methods. The works presented in this thesis are an incomplete collection of the recent advances in sample-efficient nonconvex optimization methods in both machine learning and reinforcement learning.

[AAB$^+$17] Naman Agarwal, Zeyuan Allenzhu, Brian Bullins, Elad Hazan, and Tengyu Ma. Finding approximate local minima for nonconvex optimization in linear time. 2017.

[AAZB$^+$17] Naman Agarwal, Zeyuan Allen-Zhu, Brian Bullins, Elad Hazan, and Tengyu Ma. Finding approximate local minima faster than gradient descent. In *Proceedings of the 49th annual ACM SIGACT symposium on theory of computing*, pages 1195–1199, 2017. tex.organization: ACM.

[ADH$^+$19] Sanjeev Arora, Simon Du, Wei Hu, Zhiyuan Li, and Ruosong Wang. Fine-grained analysis of optimization and generalization for overparameterized two-layer neural networks. In *International conference on machine learning*, pages 322–332, 2019.

[AG16] Animashree Anandkumar and Rong Ge. Efficient approaches for escaping higher order saddle points in non-convex optimization. In *Conference on learning theory*, pages 81–102, 2016.

[AKW12] Sungjin Ahn, Anoop Korattikara, and Max Welling. Bayesian posterior sampling via stochastic gradient fisher scoring. In *Proceedings of the 29th international conference on machine learning*, pages 1771–1778, 2012.

[AZ18] Zeyuan Allen-Zhu. Natasha 2: Faster non-convex optimization than sgd. In *Advances in neural information processing systems*, pages 2676–2687, 2018.

[AZH16] Zeyuan Allen-Zhu and Elad Hazan. Variance reduction for faster non-convex optimization. In *International conference on machine learning*, pages 699–707, 2016.

[AZL18] Zeyuan Allen-Zhu and Yuanzhi Li. Neon2: Finding local minima via first-order oracles. In *Advances in neural information processing systems*, pages 3720–3730, 2018.

[AZLL19] Zeyuan Allen-Zhu, Yuanzhi Li, and Yingyu Liang. Learning and generalization in overparameterized neural networks, going beyond two layers. In *Advances in* *neural information processing systems*, 2019.

[AZLS19] Zeyuan Allen-Zhu, Yuanzhi Li, and Zhao Song. A convergence theory for deep learning via over-parameterization. In *International conference on machine learning*, pages 242–252, 2019.

[Bai95] Leemon Baird. Residual algorithms: Reinforcement learning with function approximation. In *Machine learning proceedings 1995*, pages 30–37. Elsevier, 1995.

[BB01] Jonathan Baxter and Peter L Bartlett. Infinite-horizon policy-gradient estimation. *Journal of Artificial Intelligence Research*, 15:319–350, 2001.

[BDMP17] Nicolas Brosse, Alain Durmus, Eric Moulines, and Marcelo Pereyra. Sampling from a log-concave distribution with compact support with proximal Langevin Monte Carlo. In *Conference on learning theory*, pages 319–342, 2017.

[BEGK04] Anton Bovier, Michael Eckhoff, Véronique Gayrard, and Markus Klein. Metastability in reversible diffusion processes I: Sharp asymptotics for capacities and exit times. *Journal of the European Mathematical Society*, 6(4):399–424, 2004.

[BEL18] Sébastien Bubeck, Ronen Eldan, and Joseph Lehec. Sampling from a log-concave distribution with Projected Langevin Monte Carlo. *Discrete & Computational Geometry*, 59(4):757–783, 2018. Publisher: Springer.

[Ber95] Dimitri P Bertsekas. *Dynamic programming and optimal control*, volume 1. Athena scientific Belmont, MA, 1995.

[BGL13] Dominique Bakry, Ivan Gentil, and Michel Ledoux. *Analysis and geometry of Markov diffusion operators*, volume 348. Springer Science & Business Media, 2013.

[BK97] Vivek S Borkar and Vijaymohan R Konda. The actor-critic algorithm as multi-time-scale stochastic approximation. *Sadhana. Academy Proceedings in Engineering Sciences*, 22(4):525–543, 1997. Publisher: Springer.

[BM00] Vivek S Borkar and Sean P Meyn. The ODE method for convergence of stochastic approximation and reinforcement learning. *SIAM Journal on Control and Optimization*, 38(2):447–469, 2000. Publisher: SIAM.

[BNS16] Srinadh Bhojanapalli, Behnam Neyshabur, and Nati Srebro. Global optimality of local search for low rank matrix recovery. In *Advances in neural information processing systems*, pages 3873–3881, 2016.

[Bor97] Vivek S Borkar. Stochastic approximation with two time scales. *Systems & Control Letters*, 29(5):291–294, 1997. Publisher: Elsevier.

[BRS18] Jalaj Bhandari, Daniel Russo, and Raghav Singal. A finite time analysis of temporal difference learning with linear function approximation. In *Conference on learning theory*, pages 1691–1692, 2018.

[BSA83] A. G. Barto, R. S. Sutton, and C. W. Anderson. Neuronlike adaptive elements that can solve difficult learning control problems. *IEEE Transactions on Systems, Man, and Cybernetics*, SMC-13(5):834–846, 1983.

[BSGL09] Shalabh Bhatnagar, Richard S Sutton, Mohammad Ghavamzadeh, and Mark Lee. Natural actor–critic algorithms. *Automatica*, 45(11):2471–2482, 2009. Publisher: Elsevier.

[BV05] Francois Bolley and Cedric Villani. Weighted csiszár-kullback-pinsker inequalities and applications to transportation inequalities. *Annales de la Faculté des Sciences de Toulouse. Série VI. Mathématiques*, 14, January 2005.

[CB18] Lenaic Chizat and Francis Bach. On the global convergence of gradient descent for over-parameterized models using optimal transport. In *Advances in neural information processing systems*, pages 3036–3046, 2018.

[CCBJ18] Xiang Cheng, Niladri S. Chatterji, Peter L. Bartlett, and Michael I. Jordan. Underdamped Langevin MCMC: A non-asymptotic analysis. In *Proceedings of the 31st conference on learning theory*, volume 75, pages 300–323, 2018.

[CD16] Yair Carmon and John C Duchi. Gradient descent efficiently finds the Cubic-Regularized non-convex newton step. *arXiv preprint arXiv:1612.00547*, 2016.

[CDC15] Changyou Chen, Nan Ding, and Lawrence Carin. On the convergence of stochastic gradient MCMC algorithms with high-order integrators. In *Advances in neural information processing systems*, pages 2278–2286, 2015.

[CDHS16] Yair Carmon, John C. Duchi, Oliver Hinder, and Aaron Sidford. Accelerated methods for non-convex optimization. 2016.

[CDHS17] Yair Carmon, John C Duchi, Oliver Hinder, and Aaron Sidford. "Convex until proven guilty": Dimension-free acceleration of gradient descent on non-convex functions. In *International conference on machine learning*, pages 654–663, 2017.

[CFM$^+$18] Niladri Chatterji, Nicolas Flammarion, Yian Ma, Peter Bartlett, and Michael Jordan. On the theory of variance reduction for stochastic gradient Monte Carlo. In *Proceedings of the 35th international conference on machine learning*, pages 764–773, 2018.

[CG19a] Yuan Cao and Quanquan Gu. Generalization bounds of stochastic gradient descent for wide and deep neural networks. In *Advances in neural information processing systems*, 2019.

[CG19b] Yuan Cao and Quanquan Gu. A generalization theory of gradient descent for learning over-parameterized deep relu networks. *arXiv preprint arXiv:1902.01384*, 2019.

[CGH$^+$19] Tianle Cai, Ruiqi Gao, Jikai Hou, Siyu Chen, Dong Wang, Di He, Zhihua Zhang, and Liwei Wang. A gram-gauss-newton method learning overparameterized deep neural networks for regression problems. *arXiv preprint arXiv:1905.11675*, 2019.

[CHM$^+$15] Anna Choromanska, Mikael Henaff, Michael Mathieu, Gérard Ben Arous, and Yann LeCun. The loss surfaces of multilayer networks. In *Artificial intelligence and statistics*, pages 192–204, 2015.

[CHS87] Tzuu-Shuh Chiang, Chii-Ruey Hwang, and Shuenn Jyi Sheu. Diffusion for global optimization in Rˆn. *SIAM Journal on Control and Optimization*, 25(3):737–753, 1987. Publisher: SIAM.

[CM10] Dotan Di Castro and Ron Meir. A convergent online single time scale actor critic algorithm. *Journal of Machine Learning Research*, 11(Jan):367–410, 2010.

[CMM10] Corinna Cortes, Yishay Mansour, and Mehryar Mohri. Learning bounds for importance weighting. In *Advances in neural information processing systems*, pages 442–450, 2010.

[CRS14] Frank E Curtis, Daniel P Robinson, and Mohammadreza Samadi. A trust region algorithm with a worst-case iteration complexity of O(\epsilon$^{-3/2}$) for nonconvex optimization. *Mathematical Programming*, pages 1–32, 2014. Publisher: Springer.

[CRS17] Frank E Curtis, Daniel P Robinson, and Mohammadreza Samadi. A trust region algorithm with a worst-case iteration complexity of O(\epsilon$^{-3/2}$) for nonconvex optimization. *Mathematical Programming*, 162(1-2):1–32, 2017. Publisher: Springer.

[CYJW19] Qi Cai, Zhuoran Yang, Chi Jin, and Zhaoran Wang. Provably efficient exploration in policy optimization. *arXiv preprint arXiv:1912.05830*, 2019.

[CYLW19] Qi Cai, Zhuoran Yang, Jason D Lee, and Zhaoran Wang. Neural temporal-difference learning converges to global optima. In *Advances in neural information processing systems*, 2019.

[CZD$^+$19] Zaiwei Chen, Sheng Zhang, Thinh T. Doan, Siva Theja Maguluri, and John-Paul Clarke. Performance of q-learning with linear function approximation: Stability and finite-time analysis. *arXiv preprint arXiv:1905.11425*, 2019.

[Dal17a] Arnak Dalalyan. Further and stronger analogy between sampling and optimization: Langevin Monte Carlo and gradient descent. In *Conference on learning theory*, pages 678–689, 2017.

[Dal17b] Arnak S Dalalyan. Theoretical guarantees for approximate sampling from smooth and log-concave densities. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 79(3):651–676, 2017. Publisher: Wiley Online Library.

[DBLJ14] Aaron Defazio, Francis Bach, and Simon Lacoste-Julien. Saga: A fast incremental gradient method with support for non-strongly convex composite objectives. In *Advances in neural information processing systems*, pages 1646–1654, 2014.

[DCL$^+$17] Simon S Du, Jianshu Chen, Lihong Li, Lin Xiao, and Dengyong Zhou. Stochastic variance reduction methods for policy evaluation. In *Proceedings of the 34th international conference on machine learning-volume 70*, pages 1049–1058, 2017.

tex.organization: JMLR. org.

[DCWY18] Raaz Dwivedi, Yuansi Chen, Martin J Wainwright, and Bin Yu. Log-concave sampling: Metropolis-Hastings algorithms are fast! In *Proceedings of the 31st conference on learning theory*, pages 793–797, 2018.

[DDo14] Aaron Defazio, Justin Domke, and others. Finito: A faster, permutable incremental gradient method for big data problems. In *International conference on machine learning*, pages 1125–1133, 2014.

[DK17] Arnak S Dalalyan and Avetik G Karagulyan. User-friendly guarantees for the Langevin Monte Carlo with inaccurate gradient. *arXiv preprint arXiv:1710.00095*, 2017.

[DKLH18] Hadi Daneshmand, Jonas Kohler, Aurelien Lucchi, and Thomas Hofmann. Escaping saddles with stochastic gradients. In *International Conference on Machine Learning*, pages 1155–1164. PMLR, 2018.

[DLL+19] Simon Du, Jason Lee, Haochuan Li, Liwei Wang, and Xiyu Zhai. Gradient descent finds global minima of deep neural networks. In *International conference on machine learning*, pages 1675–1685, 2019.

[DM15] Alain Durmus and Eric Moulines. Non-asymptotic convergence analysis for the Unadjusted Langevin Algorithm. *arXiv preprint arXiv:1507.05021*, 2015.

[DM16] Alain Durmus and Eric Moulines. High-dimensional bayesian inference via the unadjusted Langevin algorithm. *arXiv preprint arXiv:1605.01559*, 2016.

[DM17] Adithya M Devraj and Sean Meyn. Zap q-learning. In *Advances in neural information processing systems*, pages 2235–2244, 2017.

[DNPo13] Marc Peter Deisenroth, Gerhard Neumann, Jan Peters, and others. A survey on policy search for robotics. *Foundations and Trends® in Robotics*, 2(1–2):1–142, 2013. Publisher: Now Publishers, Inc.

[DPG+14] Yann Dauphin, Razvan Pascanu, Caglar Gulcehre, Kyunghyun Cho, Surya Ganguli, and Yoshua Bengio. Identifying and attacking the saddle point problem in high-dimensional non-convex optimization. *Mathematics*, 111(6 Pt 1):2475–2485, 2014.

[DRW$^+$16] Kumar Avinava Dubey, Sashank J Reddi, Sinead A Williamson, Barnabas Poczos, Alexander J Smola, and Eric P Xing. Variance reduction in stochastic gradient Langevin dynamics. In *Advances in neural information processing systems*, pages 1154–1162, 2016.

[DSTM17] Gal Dalal, Balazs Szorenyi, Gugan Thoppe, and Shie Mannor. Finite sample analysis of two-timescale stochastic approximation with applications to reinforcement learning. *arXiv preprint arXiv:1703.05376*, 2017.

[DSTM18] Gal Dalal, Balázs Szörényi, Gugan Thoppe, and Shie Mannor. Finite sample analyses for td (0) with function approximation. In *Thirty-second AAAI conference on artificial intelligence*, 2018.

[DZPS19] Simon S. Du, Xiyu Zhai, Barnabas Poczos, and Aarti Singh. Gradient descent provably optimizes over-parameterized neural networks. In *International conference on learning representations*, 2019.

[EMS18] Murat A Erdogdu, Lester Mackey, and Ohad Shamir. Global non-convex optimization with discretized diffusions. In *Advances in neural information processing systems*, pages 9693–9702, 2018.

[FLLZ18] Cong Fang, Chris Junchi Li, Zhouchen Lin, and Tong Zhang. SPIDER: Near-optimal non-convex optimization via stochastic path-integrated differential estimator. In *Advances in neural information processing systems*, pages 686–696, 2018.

[FLZ19] Cong Fang, Zhouchen Lin, and Tong Zhang. Sharp analysis for nonconvex sgd escaping from saddle points. In *Conference on Learning Theory*, pages 1192–1234. PMLR, 2019.

[GBB04] Evan Greensmith, Peter L Bartlett, and Jonathan Baxter. Variance reduction techniques for gradient estimates in reinforcement learning. *Journal of Machine Learning Research*, 5(Nov):1471–1530, 2004.

[GH15] Dan Garber and Elad Hazan. Fast and simple PCA via convex optimization. *arXiv preprint arXiv:1509.05647*, 2015.

[GHJY15] Rong Ge, Furong Huang, Chi Jin, and Yang Yuan. Escaping from saddle points—online stochastic gradient for tensor decomposition. In *Conference on learning theory*, pages 797–842, 2015.

[GL13] Saeed Ghadimi and Guanghui Lan. Stochastic first-and zeroth-order methods for nonconvex stochastic programming. *SIAM Journal on Optimization*, 23(4):2341–2368, 2013. Publisher: SIAM.

[GL16] Saeed Ghadimi and Guanghui Lan. Accelerated gradient methods for nonconvex nonlinear and stochastic programming. *Mathematical Programming*, 156(1-2):59–99, 2016. Publisher: Springer.

[GLM16] Rong Ge, Jason D Lee, and Tengyu Ma. Matrix completion has no spurious local minimum. In *Advances in neural information processing systems*, pages 2973–2981, 2016.

[GLR17] Rong Ge, Holden Lee, and Andrej Risteski. Beyond log-concavity: Provable guarantees for sampling multi-modal distributions using simulated tempering Langevin Monte Carlo. *arXiv preprint arXiv:1710.02736*, 2017.

[GLZ16] Saeed Ghadimi, Guanghui Lan, and Hongchao Zhang. Mini-batch stochastic approximation methods for nonconvex stochastic composite optimization. *Mathematical Programming*, 155(1-2):267–305, 2016. Publisher: Springer.

[GM91] Saul B Gelfand and Sanjoy K Mitter. Recursive stochastic algorithms for global optimization in \$R\^d\$. *SIAM Journal on Control and Optimization*, 29(5):999–1018, 1991. Publisher: SIAM.

[GSY19] Harsh Gupta, R Srikant, and Lei Ying. Finite-time performance bounds and adaptive learning rate selection for two time-scale reinforcement learning. In *Advances in neural information processing systems*, pages 4706–4715, 2019.

[Gyo86] Istvan Gyongy. Mimicking the one-dimensional marginal distributions of processes having an Itô differential. *Probability theory and related fields*, 71(4):501–516, 1986. Publisher: Springer.

[HAV$^+$15] Reza Harikandeh, Mohamed Osama Ahmed, Alim Virani, Mark Schmidt, Jakub Kone{\v{c}}n{\'y}, and Scott Sallinen. Stop wasting my gradients: Practical svrg. In *Advances in neural information processing systems*, pages 2251–2259, 2015.

[HL13] Christopher J Hillar and Lek-Heng Lim. Most tensor problems are NP-hard. *Journal of the ACM (JACM)*, 60(6):45, 2013. Publisher: ACM.

[HM08] Martin Hairer and Jonathan C Mattingly. Spectral gaps in Wasserstein distances and the 2D stochastic Navier-Stokes equations. *The Annals of Probability*, pages 2050–2091, 2008. Publisher: JSTOR.

[HS19] Bin Hu and Usman Ahmed Syed. Characterizing the exact behaviors of temporal difference learning algorithms using markov jump linear system theory. In *Advances in neural information processing systems*, 2019.

[Hwa80] Chii-Ruey Hwang. Laplace's method revisited: weak convergence of probability measures. *The Annals of Probability*, pages 1177–1182, 1980. Publisher: JSTOR.

[IW14] Nobuyuki Ikeda and Shinzo Watanabe. *Stochastic differential equations and diffusion processes*, volume 24. Elsevier, 2014.

[JAZBJ18] Chi Jin, Zeyuan Allen-Zhu, Sebastien Bubeck, and Michael I Jordan. Is q-learning provably efficient? In *Advances in neural information processing systems*, pages 4863–4873, 2018.

[JGH18] Arthur Jacot, Franck Gabriel, and Clément Hongler. Neural tangent kernel: Convergence and generalization in neural networks. In *Advances in neural information processing systems*, pages 8571–8580, 2018.

[JGN$^+$17] Chi Jin, Rong Ge, Praneeth Netrapalli, Sham M Kakade, and Michael I Jordan. How to escape saddle points efficiently. In *International conference on machine learning*, pages 1724–1732, 2017.

[JJS94] Tommi Jaakkola, Michael I Jordan, and Satinder P Singh. Convergence of stochastic iterative dynamic programming algorithms. In *Advances in neural information processing systems*, pages 703–710, 1994.

[JNJ18] Chi Jin, Praneeth Netrapalli, and Michael I. Jordan. Accelerated gradient descent escapes saddle points faster than gradient descent. In *Proceedings of the 31st conference on learning theory*, pages 1042–1085, 2018.

[JZ13] Rie Johnson and Tong Zhang. Accelerating stochastic gradient descent using predictive variance reduction. In *Advances in neural information processing systems*, pages 315–323, 2013.

[KB14] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *CoRR*, abs/1412.6980, 2014.

[KBP13] Jens Kober, J Andrew Bagnell, and Jan Peters. Reinforcement learning in robotics: A survey. *The International Journal of Robotics Research*, 32(11):1238–1274, 2013. Publisher: SAGE Publications Sage UK: London, England.

[KIP$^+$18] Dmitry Kalashnikov, Alex Irpan, Peter Pastor, Julian Ibarz, Alexander Herzog, Eric Jang, Deirdre Quillen, Ethan Holly, Mrinal Kalakrishnan, Vincent Vanhoucke, and others. Scalable deep reinforcement learning for vision-based robotic manipulation. In *Conference on robot learning*, pages 651–673, 2018.

[KKR19] Harshat Kumar, Alec Koppel, and Alejandro Ribeiro. On the sample complexity of actor-critic method for reinforcement learning with function approximation. *arXiv preprint arXiv:1910.08412*, 2019.

[KL17] Jonas Moritz Kohler and Aurelien Lucchi. Sub-sampled cubic regularization for non-convex optimization. In *Proceedings of the 34th international conference on machine learning*, volume 70, pages 1895–1904. PMLR, 2017.

[KMN$^+$20] Maxim Kaledin, Eric Moulines, Alexey Naumov, Vladislav Tadic, and Hoi-To Wai. Finite time analysis of linear two-timescale stochastic approximation with Markovian noise. *arXiv preprint arXiv:2002.01268*, 2020.

[KP92] Peter E Kloeden and Eckhard Platen. Higher-order implicit strong numerical schemes for stochastic differential equations. *Journal of statistical physics*, 66(1):283–314, 1992. Publisher: Springer.

[Kri09] Alex Krizhevsky. Learning multiple layers of features from tiny images. Technical report, Citeseer, 2009.

[KT00] Vijay R Konda and John N Tsitsiklis. Actor-critic algorithms. In *Advances in neural information processing systems*, pages 1008–1014, 2000.

[KTo04] Vijay R Konda, John N Tsitsiklis, and others. Convergence rate of linear two-time-scale stochastic approximation. *The Annals of Applied Probability*, 14(2):796–819, 2004. Publisher: Institute of Mathematical Statistics.

[LBBH98] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998. Publisher: IEEE.

[LCYW19] Boyi Liu, Qi Cai, Zhuoran Yang, and Zhaoran Wang. Neural Proximal/Trust region policy optimization attains globally optimal policy. In *Advances in neural information processing systems*, 2019.

[Lev16] Kfir Y Levy. The power of normalization: Faster evasion of saddle points. *arXiv preprint arXiv:1611.04831*, 2016.

[Li17] Yuxi Li. Deep reinforcement learning: An overview. *CoRR*, abs/1701.07274, 2017. arXiv: 1701.07274.

[LJCJ17] Lihua Lei, Cheng Ju, Jianbo Chen, and Michael I Jordan. Non-convex finite-sum optimization via scsg methods. In *Advances in neural information processing systems*, pages 2348–2358, 2017.

[LL18] Zhize Li and Jian Li. A simple proximal stochastic gradient method for nonsmooth nonconvex optimization. In *Advances in neural information processing systems*, pages 5569–5579, 2018.

[LLG$^+$15] Bo Liu, Ji Liu, Mohammad Ghavamzadeh, Sridhar Mahadevan, and Marek Petrik. Finite-sample analysis of proximal gradient TD algorithms. In *Proceedings of the thirty-first conference on uncertainty in artificial intelligence*, pages 504–513, 2015. tex.organization: AUAI Press.

[LP17] David A Levin and Yuval Peres. *Markov chains and mixing times*, volume 107. American Mathematical Soc., 2017.

[LPW09] David Asher Levin, Yuval Peres, and Elizabeth Lee Wilmer. *Markov chains and mixing times*. American Mathematical Soc., 2009.

[LRLP17] Yang Liu, Prajit Ramachandran, Qiang Liu, and Jian Peng. Stein variational policy gradient. 2017.

[LS13] Robert S Liptser and Albert N Shiryaev. *Statistics of random processes: I.*

general theory, volume 5. Springer Science & Business Media, 2013.

[LS18] Chandrashekar Lakshminarayanan and Csaba Szepesvari. Linear stochastic approximation: How far does constant step-size and iterate averaging go? In *International conference on artificial intelligence and statistics*, pages 1347–1355, 2018.

[LWA15] Sergey Levine, Nolan Wagener, and Pieter Abbeel. Learning contact-rich manipulation skills with guided policy search. In *2015 IEEE international conference on robotics and automation (ICRA)*, pages 156–163, 2015. tex.organization: IEEE.

[LZL18] Zhize Li, Tianyi Zhang, and Jian Li. Stochastic gradient hamiltonian Monte Carlo with variance reduction for bayesian inference. *arXiv preprint arXiv:1803.11159*, 2018.

[MBM$^+$16] Volodymyr Mnih, Adria Puigdomenech Badia, Mehdi Mirza, Alex Graves, Timothy Lillicrap, Tim Harley, David Silver, and Koray Kavukcuoglu. Asynchronous methods for deep reinforcement learning. In *International conference on machine learning*, pages 1928–1937, 2016.

[MCF15] Yi-An Ma, Tianqi Chen, and Emily Fox. A complete recipe for stochastic gradient MCMC. In *Advances in neural information processing systems*, pages 2917–2925, 2015.

[Mit05] A Yu Mitrophanov. Sensitivity and convergence of uniformly ergodic Markov chains. *Journal of Applied Probability*, 42(4):1003–1014, 2005. Publisher: Cambridge University Press.

[MKS$^+$15] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A Rusu, Joel Veness, Marc G Bellemare, Alex Graves, Martin Riedmiller, Andreas K Fidjeland, Georg Ostrovski, and others. Human-level control through deep reinforcement learning. *Nature*, 518(7540):529, 2015. Publisher: Nature Publishing Group.

[MM09] Prashant Mehta and Sean Meyn. Q-learning and pontryagin's minimum principle. In *Proceedings of the 48h IEEE Conference on Decision and Control (CDC) held jointly with 2009 28th Chinese Control Conference*, pages 3598–3605, 2009.

tex.organization: IEEE.

[MMR08] Francisco S Melo, Sean P Meyn, and M Isabel Ribeiro. An analysis of reinforcement learning with function approximation. In *Proceedings of the 25th international conference on machine learning*, pages 664–671, 2008. tex.organization: ACM.

[MPFR18] Alberto Maria Metelli, Matteo Papini, Francesco Faccio, and Marcello Restelli. Policy optimization via importance sampling. In *Advances in neural information processing systems*, pages 5447–5459, 2018.

[MS08] Rémi Munos and Csaba Szepesvári. Finite-time bounds for fitted value iteration. *Journal of Machine Learning Research*, 9(May):815–857, 2008.

[MSH02] Jonathan C Mattingly, Andrew M Stuart, and Desmond J Higham. Ergodicity for SDEs and approximations: locally Lipschitz vector fields and degenerate noise. *Stochastic processes and their applications*, 101(2):185–232, 2002. Publisher: Elsevier.

[MST10] Jonathan C Mattingly, Andrew M Stuart, and Michael V Tretyakov. Convergence of numerical time-averaging and stationary measures via Poisson equations. *SIAM Journal on Numerical Analysis*, 48(2):552–577, 2010. Publisher: SIAM.

[MT12] Sean P Meyn and Richard L Tweedie. *Markov chains and stochastic stability*. Springer Science & Business Media, 2012.

[Nes13] Yurii Nesterov. *Introductory lectures on convex optimization: A basic course*, volume 87. Springer Science & Business Media, 2013.

[Nes18] Yurii Nesterov. *Lectures on convex optimization*, volume 137. Springer, 2018.

[NLST17a] Lam M Nguyen, Jie Liu, Katya Scheinberg, and Martin Takáč. SARAH: A novel method for machine learning problems using stochastic recursive gradient. In *Proceedings of the 34th international conference on machine learning-volume 70*, pages 2613–2621, 2017. tex.organization: JMLR. org.

[NLST17b] Lam M Nguyen, Jie Liu, Katya Scheinberg, and Martin Takáč. Stochastic recursive gradient algorithm for nonconvex optimization. *arXiv preprint* *arXiv:1705.07261*, 2017.

[NP06] Yurii Nesterov and B. T. Polyak. Cubic regularization of Newton method and its global performance. *Mathematical Programming*, 108(1):177–205, 2006.

[NvDP$^+$19] Lam M Nguyen, Marten van Dijk, Dzung T Phan, Phuong Ha Nguyen, Tsui-Wei Weng, and Jayant R Kalagnanam. Optimal finite-sum smooth non-convex optimization with SARAH. *CoRR, abs/1901.07648*, 2019.

[NWC$^+$11] Yuval Netzer, Tao Wang, Adam Coates, Alessandro Bissacco, Bo Wu, and Andrew Y Ng. Reading digits in natural images with unsupervised feature learning. In *NIPS workshop on deep learning and unsupervised feature learning*, 2011.

[Oja82] Erkki Oja. Simplified neuron model as a principal component analyzer. *Journal of mathematical biology*, 15(3):267–273, 1982.

[OS02] Dirk Ormoneit and {\'S}aunak Sen. Kernel-based reinforcement learning. *Machine learning*, 49(2-3):161–178, 2002. Publisher: Springer.

[PBC$^+$18] Matteo Papini, Damiano Binaghi, Giuseppe Canonaco, Matteo Pirotta, and Marcello Restelli. Stochastic variance-reduced policy gradient. In *International conference on machine learning*, pages 4023–4032, 2018.

[PP02] Theodore J Perkins and Mark D Pendrith. On the existence of fixed points for q-learning and sarsa in partially observable domains. In *Proceedings of the nineteenth international conference on machine learning*, pages 490–497, 2002. tex.organization: Morgan Kaufmann Publishers Inc.

[PRB13] Matteo Pirotta, Marcello Restelli, and Luca Bascetta. Adaptive step-size for policy gradient methods. In *Advances in neural information processing systems*, pages 1394–1402, 2013.

[PS08a] Jan Peters and Stefan Schaal. Natural actor-critic. *Neurocomputing*, 71(7-9):1180–1190, 2008. Publisher: Elsevier.

[PS08b] Jan Peters and Stefan Schaal. Reinforcement learning of motor skills with policy gradients. *Neural Networks*, 21(4):682–697, 2008. Publisher: Elsevier.

[PW16] Yury Polyanskiy and Yihong Wu. Wasserstein continuity of entropy and outer bounds for interference channels. *IEEE Transactions on Information Theory*, 62(7):3992–4002, 2016. Publisher: IEEE.

[Qia99] Ning Qian. On the momentum term in gradient descent learning algorithms. *Neural networks*, 12(1):145–151, 1999.

[QYYW19] Shuang Qiu, Zhuoran Yang, Jieping Ye, and Zhaoran Wang. On the finite-time convergence of actor-critic algorithm. *NeurIPS 2019 Optimization Foundations of Reinforcement Learning Workshop*, 2019.

[RHS$^+$16] Sashank J Reddi, Ahmed Hefny, Suvrit Sra, Barnabas Poczos, and Alex Smola. Stochastic variance reduction for nonconvex optimization. In *International conference on machine learning*, pages 314–323, 2016.

[Rie05] Martin Riedmiller. Neural fitted Q iteration–first experiences with a data efficient neural reinforcement learning method. In *European conference on machine learning*, pages 317–328, 2005. tex.organization: Springer.

[RM51] Herbert Robbins and Sutton Monro. A stochastic approximation method. *The Annals of Mathematical Statistics*, pages 400–407, 1951. Publisher: JSTOR.

[Ro61] Alfréd Rényi and others. On measures of entropy and information. In *Proceedings of the fourth berkeley symposium on mathematical statistics and probability, volume 1: Contributions to the theory of statistics*, 1961. tex.organization: The Regents of the University of California.

[RRT17] Maxim Raginsky, Alexander Rakhlin, and Matus Telgarsky. Non-convex learning via stochastic gradient Langevin dynamics: a nonasymptotic analysis. In *Conference on learning theory*, pages 1674–1703, 2017.

[RSB12] Nicolas L Roux, Mark Schmidt, and Francis R Bach. A stochastic gradient method with an exponential convergence rate for finite training sets. In *Advances in neural information processing systems*, pages 2663–2671, 2012.

[RSPS16] Sashank J Reddi, Suvrit Sra, Barnabas Poczos, and Alexander J Smola. Proximal stochastic methods for nonsmooth nonconvex finite-sum optimization. In *Advances in neural information processing systems*, pages 1145–1153, 2016.

[RT96] Gareth O Roberts and Richard L Tweedie. Exponential convergence of Langevin distributions and their discrete approximations. *Bernoulli*, pages 341–363, 1996.

Publisher: JSTOR.

[RZS$^+$18] Sashank Reddi, Manzil Zaheer, Suvrit Sra, Barnabas Poczos, Francis Bach, Ruslan Salakhutdinov, and Alex Smola. A generic approach for escaping saddle points. In *International conference on artificial intelligence and statistics*, pages 1233–1242, 2018.

[SAMR18] Wilko Schwarting, Javier Alonso-Mora, and Daniela Rus. Planning and decision-making for autonomous vehicles. *Annual Review of Control, Robotics, and Autonomous Systems*, 2018. Publisher: Annual Reviews.

[SB18] Richard S Sutton and Andrew G Barto. *Reinforcement learning: An introduction.* MIT press, 2018.

[Sch15] Jürgen Schmidhuber. Deep learning in neural networks: An overview. *Neural Networks*, 61:85–117, 2015. Publisher: Elsevier.

[SHM$^+$16] David Silver, Aja Huang, Chris J. Maddison, Arthur Guez, Laurent Sifre, George van den Driessche, Julian Schrittwieser, Ioannis Antonoglou, Vedavyas Panneershelvam, Marc Lanctot, Sander Dieleman, Dominik Grewe, John Nham, Nal Kalchbrenner, Ilya Sutskever, Timothy P. Lillicrap, Madeleine Leach, Koray Kavukcuoglu, Thore Graepel, and Demis Hassabis. Mastering the game of Go with deep neural networks and tree search. *Nature*, 529:484–489, 2016.

[SLA$^+$15] John Schulman, Sergey Levine, Pieter Abbeel, Michael I Jordan, and Philipp Moritz. Trust region policy optimization. In *International conference on machine learning*, volume 37, pages 1889–1897, 2015.

[SLH$^+$14] David Silver, Guy Lever, Nicolas Heess, Thomas Degris, Daan Wierstra, and Martin Riedmiller. Deterministic policy gradient algorithms. In *International conference on machine learning*, 2014.

[SML$^+$15] John Schulman, Philipp Moritz, Sergey Levine, Michael Jordan, and Pieter Abbeel. High-dimensional continuous control using generalized advantage estimation. *CoRR, abs/1506.02438*, 2015.

[SMSM00] Richard S Sutton, David A McAllester, Satinder P Singh, and Yishay Mansour. Policy gradient methods for reinforcement learning with function approximation. In *Advances in neural information processing systems*, pages 1057–1063, 2000.

[SN14] Issei Sato and Hiroshi Nakagawa. Approximation analysis of stochastic gradient Langevin dynamics by using fokker-planck equation and ito process. In *Proceedings of the 31st international conference on machine learning (ICML-14)*, pages 982–990, 2014.

[SOR^{+}08] Frank Sehnke, Christian Osendorfer, Thomas R{\"u}ckstie{\ss}, Alex Graves, Jan Peters, and J{\"u}rgen Schmidhuber. Policy gradients with parameter-based exploration for control. In *International conference on artificial neural networks*, pages 387–396, 2008. tex.organization: Springer.

[SOR^{+}10] Frank Sehnke, Christian Osendorfer, Thomas R{\"u}ckstie{\ss}, Alex Graves, Jan Peters, and J{\"u}rgen Schmidhuber. Parameter-exploring policy gradients. *Neural Networks*, 23(4):551–559, 2010. Publisher: Elsevier.

[SRH^{+}19] Zebang Shen, Alejandro Ribeiro, Hamed Hassani, Hui Qian, and Chao Mi. Hessian aided policy gradient. In *International conference on machine learning*, pages 5729–5738, 2019.

[SS02] Bernhard Schölkopf and Alexander J Smola. *Learning with kernels: support vector machines, regularization, optimization, and beyond.* MIT press, 2002.

[SS16] Shai Shalev-Shwartz. Sdca without duality, regularization, and individual convexity. In *International conference on machine learning*, pages 747–754, 2016.

[SSS^{+}17] David Silver, Julian Schrittwieser, Karen Simonyan, Ioannis Antonoglou, Aja Huang, Arthur Guez, Thomas Hubert, Lucas Baker, Matthew Lai, Adrian Bolton, and others. Mastering the game of go without human knowledge. *Nature*, 550(7676):354, 2017. Publisher: Nature Publishing Group.

[SSSS16] Shai Shalev-Shwartz, Shaked Shammah, and Amnon Shashua. Safe, multi-agent, reinforcement learning for autonomous driving. *CoRR*, abs/1610.03295, 2016. arXiv: 1610.03295.

[SSZ13] Shai Shalev-Shwartz and Tong Zhang. Stochastic dual coordinate ascent methods for regularized loss minimization. *Journal of Machine Learning Research*, 14(Feb):567–599, 2013.

[Sut88] Richard S Sutton. Learning to predict by the methods of temporal differences. *Machine learning*, 3(1):9–44, 1988. Publisher: Springer.

[SY19] R Srikant and Lei Ying. Finite-time error bounds for linear stochastic approximation andTD learning. In *Conference on learning theory*, pages 2803–2830, 2019.

[SYN$^+$18] Umut Simsekli, Cagatay Yildiz, Than Huy Nguyen, Taylan Cemgil, and Gael Richard. Asynchronous stochastic quasi-Newton MCMC for non-convex optimization. In *Proceedings of the 35th international conference on machine learning*, pages 4674–4683, 2018.

[Sze10] Csaba Szepesvári. Algorithms for reinforcement learning. In *Algorithms for reinforcement learning*, Synthesis lectures on artificial intelligence and machine learning. Morgan & Claypool Publishers, 2010.

[TBG$^+$18] George Tucker, Surya Bhupatiraju, Shixiang Gu, Richard Turner, Zoubin Ghahramani, and Sergey Levine. The mirage of action-dependent baselines in reinforcement learning. In *International conference on machine learning*, pages 5022–5031, 2018.

[TLR18] Belinda Tzen, Tengyuan Liang, and Maxim Raginsky. Local optimality and generalization guarantees for the Langevin algorithm via empirical metastability. In *Proceedings of the 31st conference on learning theory*, pages 857–875, 2018.

[TM03] Vladislav B Tadic and Sean P Meyn. Asymptotic properties of two time-scale stochastic approximation algorithms with constant step sizes. In *Proceedings of the 2003 american control conference, 2003.*, volume 5, pages 4426–4431, 2003. tex.organization: IEEE.

[TVR97] John N Tsitsiklis and Benjamin Van Roy. Analysis of temporal-diffference learning with function approximation. In *Advances in neural information processing systems*, pages 1075–1081, 1997.

[Ver10] Roman Vershynin. Introduction to the non-asymptotic analysis of random matrices. *arXiv preprint arXiv:1011.3027*, 2010.

[VHGS16] Hado Van Hasselt, Arthur Guez, and David Silver. Deep reinforcement learning with double q-learning. In *Thirtieth AAAI conference on artificial intelligence*, 2016.

[VZT16] Sebastian J Vollmer, Konstantinos C Zygalakis, and Yee Whye Teh. Exploration of the (non-) asymptotic bias and variance of stochastic gradient Langevin dynamics. *Journal of Machine Learning Research*, 17(159):1–48, 2016.

[WCYW20] Lingxiao Wang, Qi Cai, Zhuoran Yang, and Zhaoran Wang. Neural policy gradient methods: Global optimality and rates of convergence. In *International conference on learning representations*, 2020.

[WD92] Christopher JCH Watkins and Peter Dayan. Q-learning. *Machine learning*, 8(3-4):279–292, 1992. Publisher: Springer.

[Wie04] Marco A Wiering. Convergence and divergence in standard and averaging reinforcement learning. In *European conference on machine learning*, pages 477–488, 2004. tex.organization: Springer.

[Wil92] Ronald J Williams. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine Learning*, 8(3-4):229–256, 1992. Publisher: Springer.

[WJZ$^+$19] Zhe Wang, Kaiyi Ji, Yi Zhou, Yingbin Liang, and Vahid Tarokh. SpiderBoost: A class of faster variance-reduced algorithms for nonconvex optimization. *Advances in Neural Information Processing Systems*, 32:2406–2416, 2019.

[WRD$^+$18] Cathy Wu, Aravind Rajeswaran, Yan Duan, Vikash Kumar, Alexandre M Bayen, Sham Kakade, Igor Mordatch, and Pieter Abbeel. Variance reduction for policy gradient with action-dependent factorized baselines. In *International conference on learning representations*, 2018.

[WSH$^+$16] Ziyu Wang, Tom Schaul, Matteo Hessel, Hado Hasselt, Marc Lanctot, and Nando Freitas. Dueling network architectures for deep reinforcement learning. In *International conference on machine learning*, pages 1995–2003, 2016.

[WT11] Max Welling and Yee W Teh. Bayesian learning via stochastic gradient Langevin dynamics. In *Proceedings of the 28th international conference on machine learn-*

ing, pages 681–688, 2011.

[WZXG20] Yue Frank Wu, Weitong ZHANG, Pan Xu, and Quanquan Gu. A finite-time analysis of two time-scale actor-critic methods. In *Advances in Neural Information Processing Systems*, volume 33, pages 17617–17628, 2020.

[XCZG18] Pan Xu, Jinghui Chen, Difan Zou, and Quanquan Gu. Global Convergence of Langevin Dynamics Based Algorithms for Nonconvex Optimization. *Advances in neural information processing systems*, 2018.

[XG20] Pan Xu and Quanquan Gu. A Finite-Time Analysis of Q-Learning with Neural Network Function Approximation. In *International Conference on Machine Learning*, pages 10555–10565. PMLR, November 2020. ISSN: 2640-3498.

[XGG19] Pan Xu, Felicia Gao, and Quanquan Gu. An improved convergence analysis of stochastic variance-reduced policy gradient. In *International conference on uncertainty in artificial intelligence*, 2019.

[XGG20] Pan Xu, Felicia Gao, and Quanquan Gu. Sample efficient policy gradient methods with recursive variance reduction. In *International conference on learning representations*, 2020.

[XLP17] Tianbing Xu, Qiang Liu, and Jian Peng. Stochastic variance reduction for policy gradient estimation. *CoRR*, abs/1710.06034, 2017. arXiv: 1710.06034.

[XRM20] Peng Xu, Fred Roosta, and Michael W Mahoney. Newton-type methods for non-convex optimization under inexact Hessian information. *Mathematical Programming*, 184(1):35–70, 2020.

[XRY18] Yi Xu, Jing Rong, and Tianbao Yang. First-order stochastic algorithms for escaping from saddle points in almost linear time. In *Advances in neural information processing systems*, pages 5531–5541, 2018.

[XWL20] Tengyu Xu, Zhe Wang, and Yingbin Liang. Non-asymptotic convergence analysis of two time-scale (natural) actor-critic algorithms. *arXiv preprint arXiv:2005.03557*, 2020.

[XZ14] Lin Xiao and Tong Zhang. A proximal stochastic gradient method with progressive variance reduction. *SIAM Journal on Optimization*, 24(4):2057–2075,

2014. Publisher: SIAM.

[XZL19] Tengyu Xu, Shaofeng Zou, and Yingbin Liang. Two time-scale off-policy TD learning: Non-asymptotic analysis over Markovian samples. In *Advances in neural information processing systems*, pages 10634–10644, 2019.

[YCHW19] Zhuoran Yang, Yongxin Chen, Mingyi Hong, and Zhaoran Wang. On the global convergence of actor-critic: A case for linear quadratic regulator with ergodic cost. In *Advances in neural information processing systems*, 2019.

[YLTZ19] Huizhuo Yuan, Chris Junchi Li, Yuhao Tang, and Yuren Zhou. Policy optimization via stochastic recursive gradient algorithm. 2019.

[YXG18] Yaodong Yu, Pan Xu, and Quanquan Gu. Third-order smoothness helps: Faster stochastic optimization algorithms for finding local minima. In *Advances in neural information processing systems*, pages 4526–4536, 2018.

[YXW19] Zhuoran Yang, Yuchen Xie, and Zhaoran Wang. A theoretical analysis of deep Q-learning. *arXiv preprint arXiv:1901.00137*, 2019.

[YZ19] Long Yang and Yu Zhang. Policy optimization with stochastic mirror descent. *arXiv preprint arXiv:1906.10462*, 2019.

[YZG17] Yaodong Yu, Difan Zou, and Quanquan Gu. Saving gradient and negative curvature computations: Finding local minima more efficiently. *arXiv preprint arXiv:1712.03950*, 2017.

[YZHB18] Zhuoran Yang, Kaiqing Zhang, Mingyi Hong, and Tamer Başar. A finite sample analysis of the actor-critic algorithm. In *2018 IEEE conference on decision and control (CDC)*, pages 2759–2764, 2018. tex.organization: IEEE.

[ZCZG19] Difan Zou, Yuan Cao, Dongruo Zhou, and Quanquan Gu. Stochastic gradient descent optimizes over-parameterized deep relu networks. *Machine Learning*, 2019.

[ZG19a] Dongruo Zhou and Quanquan Gu. Lower bounds for smooth nonconvex finite-sum optimization. In *International Conference on Machine Learning*, pages 7574–7583. PMLR, 2019.

[ZG19b] Difan Zou and Quanquan Gu. An improved analysis of training overparameterized deep neural networks. In *Advances in neural information processing systems*, 2019.

[ZHNS11] Tingting Zhao, Hirotaka Hachiya, Gang Niu, and Masashi Sugiyama. Analysis and improvement of policy gradient estimation. In *Advances in neural information processing systems*, pages 262–270, 2011.

[ZHT^{+}13] Tingting Zhao, Hirotaka Hachiya, Voot Tangkaratt, Jun Morimoto, and Masashi Sugiyama. Efficient sample reuse in policy gradients with parameter-based exploration. *Neural computation*, 25(6):1512–1547, 2013. Publisher: MIT Press.

[ZKZB19] Kaiqing Zhang, Alec Koppel, Hao Zhu, and Tamer Başar. Global convergence of policy gradient methods to (almost) locally optimal policies. *arXiv preprint arXiv:1906.08383*, 2019.

[ZLC17] Yuchen Zhang, Percy Liang, and Moses Charikar. A hitting time analysis of stochastic gradient Langevin dynamics. In *Conference on learning theory*, pages 1980–2022, 2017.

[ZLYW19] Shangtong Zhang, Bo Liu, Hengshuai Yao, and Shimon Whiteson. Provably convergent two-timescale off-policy actor-critic with function approximation. *arXiv*, pages arXiv–1911, 2019.

[ZWYG18] Xiao Zhang, Lingxiao Wang, Yaodong Yu, and Quanquan Gu. A primal-dual analysis of global optimality in nonconvex low-rank matrix recovery. In *International conference on machine learning*, 2018.

[ZXG18a] Dongruo Zhou, Pan Xu, and Quanquan Gu. Stochastic nested variance reduced gradient descent for nonconvex optimization. In *Advances in neural information processing systems*, pages 3922–3933, 2018.

[ZXG18b] Dongruo Zhou, Pan Xu, and Quanquan Gu. Stochastic variance-reduced cubic regularized Newton methods. In *Proceedings of the 35th international conference on machine learning*, pages 5990–5999, 2018.

[ZXG18c] Difan Zou, Pan Xu, and Quanquan Gu. Stochastic variance-reduced Hamilton Monte Carlo methods. In *Proceedings of the 35th international conference on machine learning*, pages 6028–6037, 2018.

[ZXG18d] Difan Zou, Pan Xu, and Quanquan Gu. Subsampled stochastic variance-reduced gradient Langevin dynamics. In *Proceedings of international conference on uncertainty in artificial intelligence*, 2018.

[ZXG20] Dongruo Zhou, Pan Xu, and Quanquan Gu. Stochastic nested variance reduction for nonconvex optimization. *Journal of machine learning research*, 2020.

[ZXL19] Shaofeng Zou, Tengyu Xu, and Yingbin Liang. Finite-sample analysis for sarsa with linear function approximation. In *Advances in neural information processing systems*, pages 8665–8675, 2019.