UNIVERSITY OF CALIFORNIA

Los Angeles

Learning Under Random Reshuffling and Distributed Features

A dissertation submitted in partial satisfaction

of the requirements for the degree

Doctor of Philosophy in Electrical and Computer Engineering

by

Bicheng Ying

2018

ABSTRACT OF THE DISSERTATION

Learning Under
Random Reshuffling and Distributed Features

by

Bicheng Ying
Doctor of Philosophy in Electrical and Computer Engineering
University of California, Los Angeles, 2018
Professor Ali H. Sayed, Chair

This dissertation focuses on stochastic gradient learning for problems involving large data sets or large feature spaces. One of the main advantages of the stochastic gradient technique is its simplicity and low computational complexity especially for large data sets. This is because it selects, at each iteration, one data point at random from the training set, updates the weight iterate using this data point, and continuously repeats the procedure until sufficient convergence is attained. Two popular mechanisms for sampling the training data is to sample with or without replacement. In the first case, some samples may be repeated during the learning process, while in the second case the training data is first reshuffled randomly and the algorithm is run over the reshuffled data one sample at a time.

It has been observed in the literature through experimentation that learning under random reshuffling leads to enhanced performance, not only for the traditional stochastic gradient algorithms but also for variance-reduced implementations. However, no theoretical analysis exists that explains the phenomenon well under constant step-size adaptation. The first part of this dissertation resolves this issue and establishes analytically that, under random reshuffling, convergence is guaranteed to a small neighborhood of the optimizer at a linear rate. The analysis further shows that random reshuffling outperforms uniform sampling by showing that the iterates approach a smaller neighborhood of size $O(\mu^2)$ around the minimizer as opposed to $O(\mu)$. An analytical expression for the steady-state mean-square-error under random reshuffling is derived, which helps clarify in greater detail the differences

between sampling with and without replacement. The dissertation also provides an explanation for the periodic behavior that is observed in random reshuffling implementations.

In addition, the dissertation examines the effect of random reshuffling on variance-reduced techniques, which are known to converge to the exact minimizers of empirical risks at linear convergence rates. The existing convergence results assume uniform data sampling with replacement and no proofs or guarantees of convergence exist under random reshuffling. The dissertation provides a theoretical guarantee of linear convergence under random reshuffling for the SAGA algorithm in the mean-square sense by using an argument that is also applicable to other variance-reduced algorithms. A new amortized variance-reduced gradient (AVRG) algorithm is also proposed, which has constant storage requirements compared to SAGA and balanced gradient computations compared to SVRG.

In the second part of the dissertation, we examine learning under large feature spaces, where the feature information is assumed to be spread across agents in a network. In this setting, each agent is assumed to observe part of the feature space. Through local cooperation, the agents are supposed to interact with each other to solve an inference problem and converge towards the global minimizer of the empirical risk. The dissertation proposes two solution methods: one operates in the dual domain and another operates in the primal domain. The dual domain solution builds on gradient boosting techniques, where each agent maintains a local dual variable. By sharing the dual variable with their immediate neighbors through a diffusion learning protocol, all agents are able to match the performance of centralized boosting solutions even when the individual agents only have access to partial information about the feature space. In comparison, the primal domain solution is achieved by combining a dynamic diffusion construction, a pipeline strategy, and variance-reduced techniques. One of the main advantages of the primal solution is that it does not require separate time scales and convergence towards the exact minimizer occurs at a linear rate.

The dissertation of Bicheng Ying is approved.

Wotao Yin

Vwani P. Roychowdhury

Lieven Vandenberghe

Ali H. Sayed, Committee Chair

University of California, Los Angeles

2018

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ACKNOWLEDGMENTS

First and foremost, I would like to express my gratitude to my advisor, Professor Ali H. Sayed, who has supported and guided my Master and Ph.D. studies for five years. I still keep the draft of my first attempt at writing a formal academic paper, which was based on a naive presentation, unorganized notations, and numerous typos. I do not forget how Professor Sayed polished that coarse draft into a formal journal paper by carefully reviewing the draft, constantly discussing the idea, and revamping the presentation. His high standards of research greatly enhanced the presentation and quality of my work. My five years of training as a Ph.D. student will always be the most precious treasure of my life not only because of the professional training and understanding in scientific research but also thanks to my learning experiences from his hard-working attitude and high standards in research.

I am also thankful to Professors Lieven Vandenberghe, Vwani Roychowdhury and Wotao Yin for taking the time to serve on my Ph.D. committee.

I am glad to have met many good friends at the Adaptive Systems Laboratory (ASL) at UCLA including: Kun Yuan, Stefan Vlaski, Hawraa Salami, Chung-Kai Yu, Lucas Cassano, and Sulaiman Alghunaim. I have learned a lot from them. I will always remember our discussions on the whiteboard and the group meetings, and the collaborations on papers. I also appreciate the opportunity to have met many friends from all over the world who have visited ASL and the friends I met when I visited EPFL: Steven Lee, Chengcheng Wang, Jianshu Chen, Prof. Wenling Li, Prof. Joao Y. Ishihara, Sara Al-Sayed, Saeed Ghazanfari Rad, Edeward Nguyen, Sergio Valcarcel Macua, Roula Nassif, Augusto Santos. I would also like to thank all my other friends whom I have met here at UCLA or while an intern at Google.

Furthermore, I want to thank my family for their selfless support of my studies over the past years. It is my first time to leave home for such a long time and at such a far away distance. Working towards a PhD is always challenging, not to mention studying abroad.

None of this would have been possible without your love and support over the years.

Finally, the work in this dissertation is based upon work partially supported by the National Science Foundation under grants CCF-1524250 and ECCS-1407712. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author and do not necessarily reflect the views of the National Science Foundation.

VITA

2013	B.S., Micro-Electronics Department, Shanghai Jiao Tong University (SJTU), Shanghai, China
2014	M.S., Department of Electrical Engineering, University of California, Los Angeles (UCLA), CA, USA
2014-2018	Research Assistant, Department of Electrical and Computer Engineering, University of California, Los Angeles (UCLA), CA, USA
2016	Intern, Google, Mountain View, CA, USA
2017	Intern, Google, Los Angeles, CA, USA
2018	Visiting Student, École Polytechnique Fédérale de Lausanne(EPFL), Switzerland

PUBLICATIONS

Selected Journal Papers:

B. Ying, K. Yuan, and A. H. Sayed, "Learning under distributed features," *submitted for publication*, also available at arXiv:1805.11384, May, 2018.

B. Ying, K. Yuan, S. Vlaski, and A. H. Sayed, "Stochastic learning under random reshuffling," to appear in *IEEE Trans. Signal Processing*, also available at arXiv:1803.07964, March, 2018.

B. Ying, K. Yuan, and A. H. Sayed, "Convergence of variance-reduced stochastic learning under random reshuffling," *submitted for publication*, also available at arXiv:1708.01383, August, 2017.

B. Ying and A. H. Sayed, "Information exchange and learning dynamics over weakly-connected adaptive networks," *IEEE Trans. Information Theory*, vol. 62, no. 3, pp. 1396-1414, March 2016.

B. Ying and A. H. Sayed, "Performance limits of stochastic sub-gradient learning, Part I: Single-agent case," *Signal Processing*, vol. 144, pp. 271-282, March 2018

Selected Conference Papers:

B. Ying, K. Yuan, and A. H. Sayed, "An exponentially-convergent algorithm for learning under distributed features," in *Proc. IEEE Data Science Workshop*, pp. 1-5, Lausanne, Switzerland, June 2018

B. Ying, K. Yuan, and A. H. Sayed, "Convergence of variance-reduced learning under random reshuffling," in *Proc. IEEE ICASSP*, pp. 1-5, Calgary, Canada, April 2018.

B. Ying, K. Yuan, S. Vlaski, and A. H. Sayed, "On the performance of random reshuffling in stochastic learning," *Proc. Information Theory and Applications Workshop* (ITA), pp. 1-5, San Diego, CA, February 2017.

B. Ying, K. Yuan, and A. H. Sayed, "Online dual coordinate ascent learning," *Proc. EUSIPCO*, pp. 1793-1797, Budapest, Hungary, Sep. 2016.

B. Ying and A. H. Sayed, "Diffusion gradient boosting for networked learning," *Proc. ICASSP*, pp. 2512-2516, New Orleans, LA, March 2017.

B. Ying and A. H. Sayed, "Performance limits of single-agent and multi-agent sub-gradient stochastic learning," in *Proc. IEEE ICASSP*, Shanghai, China, March 2016.

B. Ying and A. H. Sayed, "Learning by weakly-connected adaptive agents," in *Proc. IEEE ICASSP*, pp. 5788-5792, Brisbane, Australia, April 2015.

CHAPTER 1

Introduction

One of the main advantages of the stochastic gradient technique is its simplicity and low computational complexity, especially for large datasets. This is because it selects, at each iteration, one data point at random from the training set, updates the weight iterate using this data point, and continuously repeats the procedure until sufficient convergence is attained. Two popular mechanisms for sampling the training data is to sample with or without replacement. In the first case, some samples may be repeated during the learning process, while in the second case, the training data is first reshuffled randomly and the algorithm is run over the reshuffled data one sample at a time. It has been observed in the literature through experimentation that operation under random reshuffling leads to enhanced performance, not only for the traditional stochastic gradient algorithms but also for variance-reduced implementations. However, no theoretical analysis exists that explains the phenomenon well under constant step-size adaptation. The first part of this dissertation addresses this point and establishes analytically that, under random reshuffling, convergence is guaranteed to a small neighborhood of the optimizer at a linear rate. The dissertation further examines the effect of random reshuffling on variance-reduced techniques, which are known to converge to the exact minimizers of empirical risks at linear convergence rates. The results provide a theoretical guarantee of linear convergence under random reshuffling for the SAGA algorithm in the mean-square-error sense by using an argument that is also applicable to other variance-reduced algorithms.

In the second part of the dissertation, we examine learning under large feature spaces, where the feature information is assumed to be spread across agents in a network. In this setting, each agent is assumed to observe part of the feature space. Through local

cooperation, the agents are supposed to interact with each other to solve an inference problem and converge towards the global minimizer of the empirical risk. The dissertation proposes two solution methods: one operates in the dual domain and another operates in the primal domain. The first method builds on gradient boosting techniques, while the second method avoids the need for separate time scales and is shown to converge at a linear rate to the exact minimizer.

In this initial chapter, we provide a brief review of stochastic gradient learning and variance-reduced algorithms, and also explain the modes for learning from distributed features over networked agents.

1.1 Empirical Risk Minimization

Most supervised learning problems involve the minimization of empirical risk functions which, by ergodicity arguments, provide good approximations for stochastic risk functions. The latter are usually defined as the average over some convex loss functions, $Q(w;\boldsymbol{x})$, where $\boldsymbol{x}$ denotes the random data: [8–13].

$$w^o \triangleq \arg\min_{w\in\mathbb{R}^M} \; J_E(w) \triangleq \mathbb{E}_{\boldsymbol{x}} Q(w;\boldsymbol{x}) \tag{1.1}$$

In (1.1), the expectation operator $\mathbb{E}_x$ is over the probability distribution of the data. Since the distribution is generally unknown beforehand, and data measurements $\{x_n\}_{n=1}^N$ arising from the distribution are usually available, it is customary to replace (1.1) by an empirical risk minimization problem of the form:

$$w^\star \triangleq \arg\min_{w\in\mathbb{R}^M} \; J(w) \triangleq \frac{1}{N}\sum_{n=1}^{N} Q(w;x_n), \tag{1.2}$$

In many cases of interest, each data point x_n consists of a pair $(\gamma(n), h_n)$, where $\gamma(n)$ is a scalar label variable and $h_n \in \mathbb{R}^M$ is a feature vector. Initially, in this dissertation, we assume the loss function $Q(w;x_n)$ is differentiable and the empirical risk $J(w)$ is strongly-

convex. In later chapters, we examine the case of non-smooth loss functions. Some common choices for the loss function are listed in the Table 1.1, where $r(w)$ represents a regularization parameter.

Table 1.1: Examples of regularized empirical risks based on N data pairs $\{\gamma(n), h_n\}_{n=1}^N$ and where $r(w)$ denotes a regularization factor.

Name	**Regularized empirical risk, $J(w)$.**
least-squares	$r(w) + \dfrac{1}{N}\displaystyle\sum_{n=1}^{N}\big(\gamma(n) - h_n^{\mathsf{T}} w\big)^2$
logistic	$r(w) + \dfrac{1}{N}\displaystyle\sum_{n=1}^{N}\ln\Big(1 + e^{-\gamma(n)h_n^{\mathsf{T}} w}\Big)^2$
exponential	$r(w) + \dfrac{1}{N}\displaystyle\sum_{n=1}^{N} e^{-\gamma(n)h_n^{\mathsf{T}} w}$
perceptron	$r(w) + \dfrac{1}{N}\displaystyle\sum_{n=1}^{N}\max\big\{0,\, -\gamma(n)h_n^{\mathsf{T}} w\big\}$
Huber	$r(w) + \begin{cases} \dfrac{1}{N}\displaystyle\sum_{n=1}^{N}\dfrac{1}{\eta}\big(\gamma(n) - h_n^{\mathsf{T}} w\big)^2, & \text{if } \ \lvert\gamma(n) - h_n^{\mathsf{T}} w\rvert \le \eta \\ \dfrac{1}{N}\displaystyle\sum_{n=1}^{N}\big(\lvert\gamma(n) - h_n^{\mathsf{T}} w\rvert - \eta\big), & \text{otherwise} \end{cases}$
hinge	$r(w) + \dfrac{1}{N}\displaystyle\sum_{n=1}^{N}\max\big\{0,\, 1 - \gamma(n)h_n^{\mathsf{T}} w\big\}$

1.2 Stochastic Gradient Learning

When the size of the dataset N is large, it is impractical to solve (1.2) directly with classical gradient descent. One simple, yet powerful, approach to remedy this difficulty is to employ the stochastic gradient descent (SGD) method [14–21]. In this method, at every iteration i, rather than compute the full gradient $\nabla_w J(w)$ on the entire data set, the algorithm picks one index $\boldsymbol{n}_i$ at random, and employs $\nabla_w Q(w; x_{\boldsymbol{n}_i})$ to approximate $\nabla_w J(w)$. Specifically, at

iteration i, the update for estimating the minimizer is of the form:

$$\boldsymbol{w}_{i+1} = \boldsymbol{w}_i - \mu(i)\nabla_w Q(\boldsymbol{w}_i; x_{\boldsymbol{n}_i}), \qquad i \geq 0 \tag{1.3}$$

where $\mu(i)$ is the step-size parameter.

Stochastic gradient recursions of the form (1.3) have been studied extensively in the literature, primarily in the case when the step-size $\mu(i)$ is diminishing [16–20,22]. When $J(w)$ is strongly convex, these algorithms have been shown to converge to the minimizer $w^\star$ at the sublinear rate $O(1/i)$. In comparison, implementations with constant step-size $\mu(i) = \mu$ do not converge to the exact minimizer $w^\star$, but rather to a small region around the minimizer in the order of $O(\mu)$ [11, 12, 21, 23–25]. However, convergence to this region occurs at an exponentially rate $O(\lambda^n)$, for some scalar $\lambda \in (0, 1)$. Fast convergence to an approximate but close solution is very useful in the context of machine learning since, after all, empirical risks of the form (1.2) correspond to auxiliary problem formulations for the true, yet inaccessible problem of interest, i.e., (1.1). While (1.2) is used for training, the actual performance on unseen data is measured through (1.1). The intrinsic bias between $w^\star$ and w^o removes the need for exact convergence to $w^\star$ [18, 19]. This line of reasoning, along with the fast exponential convergence rate and robustness to initialization, has motivated a tremendous interest in constant step-size implementations with a focus on practical solutions [19, 26–28].

A fundamental question that arises when employing a constant step-size is how to choose μ in order to ensure a desired tolerance on the excess risk (ER) or mean-square-deviation (MSD) that persist after convergence. Non-asymptotic bounds have been given in [1, 12, 21, 25, 29, 30], which are useful in revealing worst-case performance guarantees, but do not predict exact performance. Recent advances in the field of online adaptation, on the other hand, have yielded insights into the related problem of learning from streaming data [11, 24, 28]. In particular, MSD and ER expressions, which are accurate to first order in the step-size, are derived in [11, 28] for a broad class of risk functions beyond the traditional quadratic measure.

1.3 Random Reshuffling

Unlike the case of the stochastic-gradient algorithm under uniform sampling, which has well-established performance results, the study of stochatic gradient learning under random reshuffling with constant step-sizes is less developed. In random reshuffling implementations, the data points are no longer picked independently and uniformly at random. Instead, the algorithm runs multiple times over the data where each run is indexed by $k \geq 1$ and is referred to as an epoch. For each epoch, the original data is first reshuffled and then passed over in order. In this manner, the i-th sample of epoch k is denoted by $\boldsymbol{\sigma}^k(i)$, where the symbol $\boldsymbol{\sigma}$ represents a uniform random permutation of the indices. We can then express the random reshuffling algorithm for the $k-$th epoch in the following manner:

$$\boldsymbol{w}_{i+1}^k = \boldsymbol{w}_i^k - \mu \nabla_w Q(\boldsymbol{w}_i^k; x_{\boldsymbol{\sigma}^k(i)}), \quad i = 0, \ldots, N-1 \tag{1.4}$$

with the boundary condition:

$$\boldsymbol{w}_0^k = \boldsymbol{w}_N^{k-1} \tag{1.5}$$

In other words, the initial condition for epoch k is the last iterate from epoch $k-1$. The boldface notation for the symbols $\boldsymbol{w}$ and $\boldsymbol{\sigma}$ in (1.4) emphasizes the random nature of these variables due to the randomness in the permutation operation. While the samples over one epoch are no longer picked independently from each other, the uniformity of the permutation function implies the following useful properties [31–33]:

$$\boldsymbol{\sigma}^k(i) \neq \boldsymbol{\sigma}^k(j), \;\; 1 \leq i \neq j \leq N \tag{1.6}$$

$$\mathbb{P}[\, \boldsymbol{\sigma}^k(i) = n \,] = \frac{1}{N}, \qquad 1 \leq n \leq N \tag{1.7}$$

$$\mathbb{P}[\boldsymbol{\sigma}^k(i+1) = n \,|\, \boldsymbol{\sigma}^k(1\colon i)] = \begin{cases} \dfrac{1}{N-i}, & n \notin \boldsymbol{\sigma}^k(1{:}i) \\ 0 \quad , & n \in \boldsymbol{\sigma}^k(1{:}i) \end{cases} \tag{1.8}$$

where $\boldsymbol{\sigma}^k(1{:}i)$ represents the collection of permuted indices for the samples numbered 1 through i.

Several recent works [34–36] have pursued justifications for the enhanced behavior of random reshuffling implementations over uniform sampling by examining the convergence rate of the learning process under diminishing step-sizes. It has been analytically shown that the convergence rate under random reshuffling can be improved from $O(1/i)$ to $O(1/i^2)$ for strongly-convex risks [35, 37], where i is the number of iterations. We also provide a short proof sketch later in Appendix 3.G. However, some of the justifications rely on loose bounds, or their conclusions are dependent on the sample size which is problematic for large datasets. Also, in the work [36], it only establishes that random reshuffling will not degrade performance relative to the stochastic gradient implementation.

In this dissertation, we focus on a different setting than [34–36] involving random reshuffling under *constant* rather than decaying step-sizes. In this case, convergence is only guaranteed to a small neighborhood of the optimizer albeit at a linear rate. The analysis will establish analytically that random reshuffling outperforms independent sampling (with replacement) by showing that the mean-square-error of the iterate at the end of each run in the random reshuffling strategy will be in the order of $O(\mu^2)$. This is a significant improvement over the performance of traditional stochastic gradient descent, which is $O(\mu)$ [13]. Furthermore, we derive an analytical expression for the steady-state mean-square-error performance of the algorithm, which helps clarify in greater detail the differences between sampling with and without replacement. We also explain the periodic behavior that is observed in random reshuffling implementations.

1.4 Variance-Reduction Algorithms

As mentioned in the previous section, the stochastic gradient algorithm under constant step size μ converges to a small $O(\mu)-$region around the minimizer instead of the exact minimizer. In recent years, several useful variance-reduced stochastic gradient algorithms have been proposed, including SVRG [38], SAGA [39], Finito [40], SDCA [41], and SAG [42], with the intent of reaching the exact minimizer of an empirical risk. Under constant step-sizes and strong-convexity assumptions on the loss functions, these methods have been shown to

attain linear convergence towards the exact minimizer when the data samples are uniformly sampled with replacement. It is sufficient for our purposes in this dissertation to focus on SAGA and SVRG. We list the SAGA recursions in Table 1.2.

Table 1.2: Listing of the SAGA algorithm under uniform sampling

SAGA under uniform sampling [39]

Initialization: $\boldsymbol{w}_0^0 = 0, \nabla Q(\boldsymbol{\phi}_{0,n}^0; x_n) = 0,\ n = 1, 2, \ldots, N.$
Repeat $i = 0, 1, \ldots N-1$ **(iteration)**:

$$\boldsymbol{n} = \mathcal{U}[1, N] \qquad \text{(uniformly sampled integer between 1 and } N\text{)} \tag{1.9}$$

$$\boldsymbol{w}_{i+1} = \boldsymbol{w}_i - \mu\Big[\nabla Q(\boldsymbol{w}_i; x_{\boldsymbol{n}}) - \nabla Q(\boldsymbol{\phi}_{i,\boldsymbol{n}}; x_{\boldsymbol{n}}) + \frac{1}{N}\sum_{n=1}^{N} \nabla Q(\boldsymbol{\phi}_{i,n}; x_n)\Big] \tag{1.10}$$

$$\boldsymbol{\phi}_{i+1,\boldsymbol{n}} = \boldsymbol{w}_{i+1}, \ \text{ and } \ \boldsymbol{\phi}_{i+1,n} = \boldsymbol{\phi}_{i,n}, \ \text{ for } n \neq \boldsymbol{n} \tag{1.11}$$

End

Comparing the SAGA update step (1.10) with the stochastic gradient update step (1.3), we find that SAGA has two extra gradients represented by $\nabla Q(\boldsymbol{\phi}_{i,\boldsymbol{n}}; x_{\boldsymbol{n}})$ and $\frac{1}{N}\sum_{n=1}^{N} \nabla Q(\boldsymbol{\phi}_{i,n}; x_n)$. These gradients are defined in terms of history variables $\{\boldsymbol{\phi}_{i,\boldsymbol{n}}\}$, which have size $M \times 1$ each and store previous values of the iterates $\boldsymbol{w}_i$. To facilitate the understanding of the algorithm, we associate a matrix $\boldsymbol{\Phi}_i$ with each run:

$$\boldsymbol{\Phi}_i \ \triangleq \ \Big[\ \boldsymbol{\phi}_{i,1} \ \Big|\ \boldsymbol{\phi}_{i,2} \ \Big|\ \ldots \ \Big|\ \boldsymbol{\phi}_{i,N} \ \Big] \tag{1.12}$$

At every iteration i, one random column of $\boldsymbol{\Phi}_i$ is populated by the iterate $\boldsymbol{w}_{i+1}$; the column location of this random cell is determined by the value of $\boldsymbol{n}_i$.

In [40], it is shown that under some mild conditions on the loss function and the step-size parameter, the stochastic gradient used in the SAGA implement has two crucial properties:

$$\mathbb{E}_{\boldsymbol{n}} \left[\nabla Q(\boldsymbol{w}_i; x_{\boldsymbol{n}}) - \nabla Q(\boldsymbol{\phi}_{i,\boldsymbol{n}}; x_{\boldsymbol{n}}) + \frac{1}{N}\sum_{n=1}^{N} \nabla Q(\boldsymbol{\phi}_{i,n}; x_n)\right] = \nabla J(\boldsymbol{w}_i) \tag{1.13}$$

$$\lim_{i\to\infty} \mathbb{E} \left\| \left[\nabla Q(\boldsymbol{w}_i; x_{\boldsymbol{n}}) - \nabla Q(\boldsymbol{\phi}_{i,\boldsymbol{n}}; x_{\boldsymbol{n}}) + \frac{1}{N}\sum_{n=1}^{N} \nabla Q(\boldsymbol{\phi}_{i,n}; x_n) \right] - \nabla J(\boldsymbol{w}_i) \right\|^2 = 0 \qquad (1.14)$$

The first property (1.13) implies the SAGA update step provides an unbiased estimate for the true gradient vector. The left hand side of the second expression (1.14) represents the variance of the gradient noise, which is seen to converge to zero asymptotically. These two properties combined imply that the SAGA iteration will be able to learn the true full gradient given sufficient iterations.

One inconvenience of the SAGA implementation is its high storage requirement, which refers to the need to track the history variables $\{\boldsymbol{\phi}_{i,n}\}$ or the gradients for use in (1.10). There is a need to store $O(N)$ variables. In big data applications, the size of N can be prohibitive. The same storage requirement applies to the variant with reshuffling proposed in [43]. An alternative method is the stochastic variance-reduced gradient (SVRG) algorithm [38], which is listed below (again under uniform sampling) for ease of reference.

Table 1.3: Listing of the SVRG algorithm under uniform sampling

SVRG under uniform sampling [38]

Initialization: $\boldsymbol{w}_0^0 = 0$.
Repeat $t = 0, 1, 2 \ldots, T$ **(epochs)**:

$$\nabla J(\boldsymbol{w}_0^t) \;=\; \frac{1}{N}\sum_{n=1}^{N} \nabla Q(\boldsymbol{w}_0^t; x_n)$$

Repeat $i = 0, 1, \ldots N-1$ **(iteration)**:

$$\boldsymbol{n} = \mathcal{U}[1, N] \qquad \text{(uniformly sampled integer between 1 and } N) \qquad (1.15)$$

$$\boldsymbol{w}_{i+1}^t = \boldsymbol{w}_i^t - \mu \left[\nabla Q(\boldsymbol{w}_i^t; x_{\boldsymbol{n}}) - \nabla Q(\boldsymbol{w}_0^t; x_{\boldsymbol{n}}) + \nabla J(\boldsymbol{w}_0^t) \right] \qquad (1.16)$$

End

$$\boldsymbol{w}_0^{t+1} = \boldsymbol{w}_N^t \qquad (1.17)$$

End

The SVRG algorithm replaces the history variables $\{\boldsymbol{\phi}_{i,\boldsymbol{n}}\}$ of SAGA by a fixed initial condition $\boldsymbol{w}_0^t$ for each epoch. This simplification greatly reduces the storage requirement.

However, each epoch in SVRG is preceded by an aggregation step to compute a gradient estimate, which is time-consuming for large datasets. It also causes the operation of SVRG to become *unbalanced*, with a larger time interval needed before each epoch, and shorter time intervals needed within the epoch. Similar to the gradient properties (1.13) and 1.14 in SAGA, the gradient step used in SVRG also has similar properties:

$$\mathbb{E}_{\boldsymbol{n}} \left[\nabla Q(\boldsymbol{w}_i^t; x_{\boldsymbol{n}}) - \nabla Q(\boldsymbol{w}_0^t; x_{\boldsymbol{n}}) + \nabla J(\boldsymbol{w}_0^t)\right] = \nabla J(\boldsymbol{w}_i^t) \tag{1.18}$$

$$\lim_{i\to\infty} \mathbb{E} \left\| \left[\left[\nabla Q(\boldsymbol{w}_i^t; x_{\boldsymbol{n}}) - \nabla Q(\boldsymbol{w}_0^t; x_{\boldsymbol{n}}) + \nabla J(\boldsymbol{w}_0^t)\right]\right] - \nabla J(\boldsymbol{w}_i^t) \right\|^2 = 0 \tag{1.19}$$

All previous results and the existing convergence results assume uniform data sampling with replacement and no proofs or guarantees of exact convergence exist for variance-reduced algorithms under random reshuffling. The dissertation provides a theoretical guarantee of linear convergence under random reshuffling for SAGA in the mean-square sense; the argument is also adaptable to other variance-reduced algorithms. Furthermore, a new amortized variance-reduced gradient (AVRG) algorithm is proposed, which has constant storage requirements compared to SAGA and balanced gradient computations compared to SVRG. AVRG is also shown analytically to converge linearly.

1.5 Distributed Features over Networks

Large-scale optimization problems are common in data-intensive machine learning problems [9, 44–46]. For applications, where both the size of the dataset and the dimension of the feature space are large, it is not uncommon for the dataset to be too large to be stored or even processed effectively at a single location or by a single agent. In this article, we examine the situation where the feature data is split across agents either due to privacy considerations or because they are already physically collected in a distributed manner by means of a networked architecture and aggregation of the data at a central location entails unreasonable costs. More specifically, the entries (blocks) of the feature vector are assumed to be distributed over a collection of K networked agents, as illustrated in Fig. 1.1. For

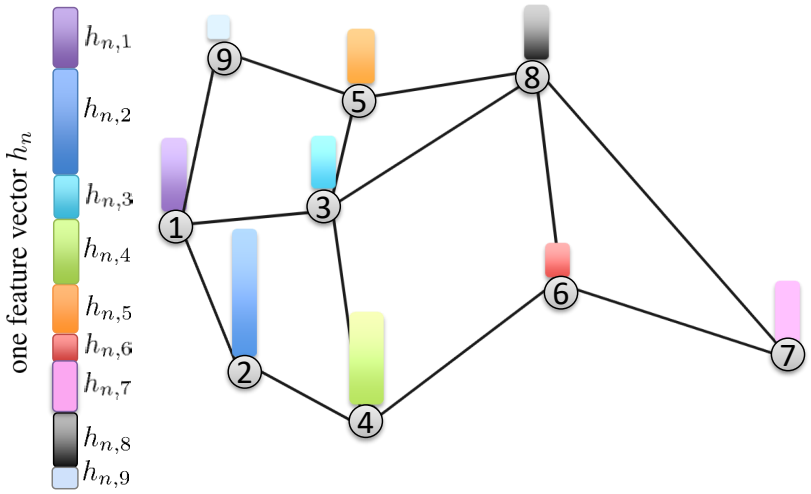

Figure 1.1: Distributing the feature across the networked agents.

instance, in sensor network applications, [47, 48], multiple sensors are normally employed to monitor an environment; the sensors are distributed over space and can be used to collect different measurements. Likewise, in multi-view learning problems [49, 50], the observed model is represented by multiple feature sets. Another example is the Cournot competition problem in networked markets [51–53], where individual factories have information about their local markets, which will not share with each other. Distributed dictionary learning problems [54] also fit into this scenario if viewing the dictionary as feature.

In this work, we focus on empirical risk minimization problems, which, by ergodicity arguments, provide a good approximation for average risks [11, 15, 48, 55–59]. Formally, we consider an empirical risk of the form:

$$J(w) = \frac{1}{N}\sum_{n=1}^{N} Q\Big(h_n^{\mathsf{T}} w; \gamma_n\Big) + r(w) \tag{1.20}$$

where the unknown parameter model (or separating hyperplane) is designated by $w \in \mathbb{R}^{M\times 1}$, while $h_n \in \mathbb{R}^{M\times 1}$ denotes the n-th feature vector and γ_n the corresponding scalar label. Moreover, the notation $Q(h^{\mathsf{T}}w; \gamma)$ refers to the loss function and is assumed to be a differentiable and convex function over w. In most problems of interest, the loss function is dependent on the inner product $h^{\mathsf{T}}w$ rather than the individual terms $\{h, w\}$. The factor $r(w)$ represents the regularization term. We denote the minimizer of $J(w)$ in (1.20) by $w^\star$. Although we are

assuming w and h_n to be column vectors, and γ_n to be a scalar, the analysis can be easily extended to matrix quantities $W \in \mathbb{R}^{M\times C}$ and to vector labels $\gamma \in \mathbb{R}^{C\times 1}$, which we will illustrate later in the simulations.

Since the entries of the feature vector are assumed distributed over K agents, we partition each h_n, and similarly the weight vector w, into K sub-vectors denoted by $\{h_{n,k}, w_k\}$, where $k = 1, 2, \ldots, K$:

$$h_n \triangleq \begin{bmatrix} h_{n,1} \\ \hline h_{n,2} \\ \hline \vdots \\ \hline h_{n,K} \end{bmatrix}, \quad w \triangleq \begin{bmatrix} w_1 \\ \hline w_2 \\ \hline \vdots \\ \hline w_K \end{bmatrix} \tag{1.21}$$

Each sub-feature vector $h_{n,k}$ and sub-vector w_k are assumed to be located at agent k. The dimensions of $\{h_{n,k}, w_k\}$ can vary over k, i.e., over agents. In this way, the empirical risk function can be rewritten in the form

$$J(w) = \frac{1}{N}\sum_{n=1}^{N} Q\left(\sum_{k=1}^{K} h_{n,k}^{\mathsf{T}} w_k; \gamma_n\right) + \sum_{k=1}^{K} r(w_k) \tag{1.22}$$

where we are also assuming that the regularization term satisfies an additive factorization of the form

$$r(w) = \sum_{k=1}^{K} r(w_k) \tag{1.23}$$

with regularization applied to each sub-vector w_k. This property holds for many popular regularization choices, such as ℓ_2, ℓ_1, KL-divergence, etc. Observe that in the form (1.22), the argument of the loss function is now a sum over the inner products $h_{n,k}^{\mathsf{T}} w_k$. That is, we have a "cost-of-sum" form similar to what was discussed in [54]. Our objective is to optimize (1.22) over the $\{w_k\}$ and to seek the optimal values in a distributed manner.

In the dissertation, we will first propose a solution in the dual domain that employs gradient boosting techniques. Next, we will propose a solution that operates directly in the *primal* domain, which will be shown to converge at a *linear* rate to the *exact* minimizer of

$J(w)$ without requiring separate time-scales.

1.6 Organization

The dissertation is organized as follows:

- **Chapter 2:** In this chapter, we draw from recent results in the field of online adaptation to derive new tight performance expressions for empirical implementations of stochastic gradient descent, mini-batch gradient descent, and importance sampling algorithms. The expressions are exact to first order in the step-size parameter and are tighter than existing bounds. We further quantify the performance gained from employing mini-batch solutions, and propose an optimal importance sampling algorithm to optimize performance.

- **Chapter 3:** This chapter focuses on random reshuffling algorithms with constant step-sizes. In this case, convergence is guaranteed to a small neighborhood of the optimizer albeit at a linear rate. The analysis establishes analytically that random reshuffling outperforms uniform sampling by showing that iterates approach a smaller neighborhood of size $O(\mu^2)$ around the minimizer rather than $O(\mu)$.

- **Chapter 4:** This chapter focuses on random reshuffling algorithm with the variance-reduced stochastic gradient algorithms. The existing convergence results assume uniform data sampling with replacement. However, it has been observed in the literature that random reshuffling can deliver superior performance over uniform sampling and, yet, no formal proofs or guarantees of exact convergence exist. This chapter provides a theoretical guarantee of linear convergence under random reshuffling for SAGA in the mean-square sense; the argument is also adaptable to other variance-reduced algorithms. Under random reshuffling, this chapter also proposes a new amortized variance-reduced gradient (AVRG) algorithm with constant storage requirements compared to SAGA and with balanced gradient computations compared to SVRG. AVRG is also shown to converge linearly.

- **Chapter 5:** Using duality arguments from optimization theory, this chapter develops an effective distributed gradient boosting strategy for inference and classification by networked clusters of learners. By sharing local dual variables with their immediate neighbors through a diffusion learning protocol, the clusters are able to match the performance of centralized boosting solutions even when the individual clusters only have access to partial information about the feature space. The performance of the resulting fully-distributed procedure is illustrated on two data sets with superior results in comparison to a centralized boosting solution.

- **Chapter 6:** This chapter continues to study the problem of learning under both large data and large feature space scenarios. The feature information is assumed to be spread across agents in a network, where each agent observes some of the features. Through local cooperation, the agents are supposed to interact with each other to solve the inference problem and converge towards the global minimizer of the empirical risk. We study this problem exclusively in the primal domain, and propose new and effective distributed solutions with guaranteed convergence to the minimizer. This is achieved by combining a dynamic diffusion construction, a pipeline strategy, and variance-reduced techniques. Simulation results illustrate the conclusions.

- **Chapter 7:** This chapter examines the performance of stochastic sub-gradient learning strategies under weaker conditions than usually considered in the literature. The conditions are shown to be automatically satisfied by several important cases of interest including the construction of Linear-SVM, LASSO, and Total-Variation denoising formulations. In comparison, these problems do not satisfy the traditional assumptions automatically and, therefore, conclusions derived based on these earlier assumptions are not directly applicable to these problems.

1.7 Notation

All vectors are column vectors. We use boldface letters to denote random quantities (such as $w_{k,i}$) and regular font to denote their realizations or deterministic variables (such as x_k). We use $\mathrm{diag}\{x_1, \cdots, x_N\}$ to denote a (block) diagonal matrix consisting of diagonal entries (blocks) $\{x_1, \cdots, x_N\}$, and use $\mathrm{col}\{x_1, \cdots, x_N\}$ to denote a column vector formed by stacking $\{x_1, \cdots, x_N\}$ on top of each other. We also use $\mathbb{E}_x$ to denote the expectation with respect to x, $(\cdot)^{\mathsf{T}}$ to denote transposition, and $\|\cdot\|$ for the 2-norm of a matrix or the Euclidean norm of a vector. The notation $\mathbb{1}_N = \mathrm{col}\{1, \ldots, 1\} \in \mathbb{R}^N$.

CHAPTER 2

Stochastic Gradient Descent with Finite Samples Sizes

In this chapter, we investigate the performance of stochastic gradient descent over finite data sets and under constant step-sizes.

We thus consider the problem of minimizing an empirical risk function $J(w)$, which is defined as the sample average over a possibly large, yet finite training set:

$$w^\star \triangleq \arg\min_{w\in\mathbb{R}^M} \quad J(w) \triangleq \frac{1}{N}\sum_{n=1}^{N} Q(w;x_n), \tag{2.1}$$

where the $\{x_n\}_{n=1}^N$ are training data samples. When the size of the dataset N is large, it is impractical to solve (2.1) directly with classical gradient descent. One simple, yet powerful, approach to remedy this difficulty is to employ the stochastic gradient method (SG) [15–21]. In this method, at every iteration, rather than compute the full gradient $\nabla_w J(w)$ on the entire data set, the algorithm picks one index $\boldsymbol{n}_i$ at random, and employs $\nabla_w Q(w;x_{\boldsymbol{n}_i})$ to approximate $\nabla_w J(w)$. Specifically, at iteration i, the update for estimating the minimizer is of the form:

$$\boldsymbol{w}_{i+1} = \boldsymbol{w}_i - \mu(i)\nabla_w Q(\boldsymbol{w}_i; x_{\boldsymbol{n}_i}), \tag{2.2}$$

where $\mu(i)$ is the step-size parameter. Although uncommon in the literature, in this chapter *alone* we refer to recursion (2.1) as the *empirical* stochastic gradient (E-SG) iteration, mainly because we will be contrasting it with an *online* stochastic gradient (O-SG) algorithm:

$$\boldsymbol{w}_{i+1} = \boldsymbol{w}_i - \mu(i)\nabla_w Q(\boldsymbol{w}_i; \boldsymbol{x}_i) \tag{2.3}$$

where $\boldsymbol{x}_i$ is the streaming data arriving at time i. In comparison, in the empirical (E-SG) implementation (2.2), the data $x_{\boldsymbol{n}_i}$ is indexed by a randomly selected index, $\boldsymbol{n}_i$, from the finite sample-size range $1 \leq n \leq N$.

In this chapter, we reveal an interesting connection between the two classes of empirical (E-SG) and online (O-SG) constructions. First, we show that the stochastic gradient descent algorithm for learning empirical risks (E-SG) is a special case of online stochastic gradient descent algorithms (O-SG) studied in [11, 28]. This connection helps establish a powerful unification for learning from finite datasets and learning from streaming data. Once this connection is established, we then leverage this insight to great effect to derive first-order expressions for both the MSD and ER of empirical (E-SG) implementations. The resulting expressions appear to be the tightest in comparison to available results in the literature, such as [1, 12, 21, 25] and other similar works. We further extend the analysis to include mini-batch gradient descent [60, 61] and importance sampling methods [1, 2], and also derive the corresponding MSD and ER expressions for both algorithms. In particular, we show that the MSD and ER of mini-batch are inversely proportional to the batch size.

Another important topic in this chapter is that we use the performance expressions to optimize the probability with which the data samples are selected during the empirical implementation. Different from previous works [1, 2], which assume knowledge of Lipschitz constants and use them to design the sampling probability, we start from the uniform distribution and devise a procedure that automatically learns the optimal sampling distribution and attains the optimal ER performance.

2.1 Empirical Stochastic Gradient Descent

In this section we derive the steady-state performance of E-SG implementations by showing how they can be viewed as special cases of O-SG implementations. The analysis will build on results from [11], which considered stochastic optimization problems of the form (1.1).

2.1.1 Relating Both Formulations

Given a finite number of data samples $\{x_1, x_2, \ldots, x_N\}$, we introduce a discrete random variable $\boldsymbol{x}_e$ having these samples as realizations and a uniform probability mass function (pmf) defined by

$$p(\boldsymbol{x}_e) = \begin{cases} \frac{1}{N}, & \text{if } \boldsymbol{x}_e = x_1, \\ \vdots & \vdots \\ \frac{1}{N}, & \text{if } \boldsymbol{x}_e = x_N. \end{cases} \tag{2.4}$$

As a result, the empirical problem (2.1) can be rewritten as

$$\min_{w \in \mathbb{R}^M} \quad J(w) = \mathbb{E}[Q(w; \boldsymbol{x}_e)] = \frac{1}{N} \sum_{n=1}^{N} Q(w; x_n), \tag{2.5}$$

which has the same form as (4) with the random data $\boldsymbol{x}$ replaced by $\boldsymbol{x}_e$. Therefore, we can apply the O-SG algorithm (2.3) to solve (2.5), namely,

$$\boldsymbol{w}_{i+1} = \boldsymbol{w}_i - \mu \nabla Q(\boldsymbol{w}_i; \boldsymbol{x}_{e,i}), \tag{2.6}$$

where the notation $\boldsymbol{x}_{e,i}$ represents the realization of $\boldsymbol{x}_e$ that streams in at iteration i. Since $\boldsymbol{x}_{e,i}$ is selected from $\{x_1, x_2, \cdots, x_N\}$ at iteration i according to the pmf (2.4), we can rewrite $\boldsymbol{x}_{e,i}$ as $x_{\boldsymbol{n}_i}$ and replace (2.6) by

$$\boldsymbol{w}_{i+1} = \boldsymbol{w}_i - \mu \nabla Q(\boldsymbol{w}_i; x_{\boldsymbol{n}_i}). \tag{2.7}$$

Here, the variable $\boldsymbol{n}_i$ is a uniform discrete random variable indicating the index of the sample that is picked at iteration i. Recursion (2.7) is the E-SG algorithm (2.2). We therefore conclude that the E-SG recursion is an O-SG recursion applied to the solution of the stochastic optimization problem (2.5). This interpretation is useful because we can now call upon results from [11] for O-SG and apply them to characterize the performance of

E-SG. This step is not as straightforward as it appears. This is because the results in [11], as is common in studies on stochastic optimization, rely on certain regularity conditions on the risk function and the gradient noise process. In order to be able to appeal to the earlier results from stochastic optimization theory, we need to verify first that problem (2.5) satisfies these regularity conditions. In preparation for the main results, we list two typical conditions on the empirical loss function.

Assumption 2.1 (Condition on loss function) *It is assumed that $Q(w;x_n)$ is differentiable and has a δ_n-Lipschitz continuous gradient, i.e., for every $n = 1,\ldots,N$ and any $w_1, w_2 \in \mathbb{R}^M$:*

$$\|\nabla_w Q(w_1;x_n) - \nabla_w Q(w_2;x_n)\| \leq \delta_n \|w_1 - w_2\|. \tag{2.8}$$

We also assume $J(w)$ is ν-strongly convex. ■

If we introduce $\delta = \max\{\delta_1, \delta_2, \cdots, \delta_N\}$, then each $\nabla_w Q(w;x_n)$ is also δ-Lipschitz continuous.

Assumption 2.2 (Smoothness condition) *It is assumed that $J(w)$ is twice differentiable and that the Hessian matrix of $J(w)$ is locally Lipschitz continuous in a small neighborhood around $w^\star$:*

$$\|\nabla_w^2 J(w^\star + \Delta w) - \nabla_w^2 J(w^\star)\| \leq \kappa_e \|\Delta w\|, \tag{2.9}$$

where $\|\Delta w\| \leq \epsilon$ and constant $\kappa_e \geq 0$. ■

2.1.2 Gradient Noise and its Moments

For the E-SG algorithm (2.7), the gradient noise is given by the approximate gradient at iteration i is $\nabla_w Q(\boldsymbol{w}_i; x_{\boldsymbol{n}_i})$ and the true gradient is

$$\nabla_w J(\boldsymbol{w}_i) = \frac{1}{N}\sum_{n=1}^{N} Q(\boldsymbol{w}_i; x_n) \tag{2.10}$$

Therefore, we introduce the gradient noise

$$\boldsymbol{s}_i(\boldsymbol{w}_i) \triangleq \nabla_w Q(\boldsymbol{w}_i; x_{\boldsymbol{n}_i}) - \nabla_w J(\boldsymbol{w}_i). \tag{2.11}$$

Let $\boldsymbol{\mathcal{F}}_i$ refer to the collection of all past iterates $\{\boldsymbol{w}_j, j \leq i\}$ and define

$$\begin{aligned} R_s &\triangleq \lim_{i\to\infty} \mathbb{E}[\boldsymbol{s}_i(w^\star)\boldsymbol{s}_i^{\mathsf{T}}(w^\star)|\boldsymbol{\mathcal{F}}_{i-1}] \\ &\overset{(a)}{=} \frac{1}{N}\sum_{n=1}^{N}[\nabla_w Q(w^\star; x_n)\nabla_w Q(w^\star; x_n)^{\mathsf{T}}], \end{aligned} \tag{2.12}$$

where (a) holds because $\nabla_w J(w^\star) = 0$. Based on this definition,

$$\mathrm{Tr}(R_s) = \frac{1}{N}\sum_{n=1}^{N} \|\nabla_w Q(w^\star; x_n)\|^2. \tag{2.13}$$

We now verify that the gradient noise process (2.11) has zero mean and its second-order moment improves as the iterate gets closer to the desired minimizer, $w^\star$. These are among the regularity conditions required in [11]. Here we show that this is not an assumption anymore for E-SG but that it does actually hold.

Lemma 2.1 (Gradient noise properties) *The first, second and fourth-order moments of the gradient noise $\boldsymbol{s}_i(\boldsymbol{w}_i)$ satisfy:*

$$\mathbb{E}[\boldsymbol{s}_i(\boldsymbol{w}_i)|\boldsymbol{\mathcal{F}}_i] = 0, \tag{2.14}$$

$$\mathbb{E}[\|\boldsymbol{s}_i(\boldsymbol{w}_i)\|^2|\boldsymbol{\mathcal{F}}_i] \leq \beta_e^2\|\widetilde{w}_i\|^2 + \sigma_e^2, \tag{2.15}$$

$$\mathbb{E}[\|\boldsymbol{s}_i(\boldsymbol{w}_i)\|^4|\boldsymbol{\mathcal{F}}_i] \leq \beta_{e4}^4\|\widetilde{w}_i\|^2 + \sigma_{e4}^4, \tag{2.16}$$

where $\widetilde{w}_i = w^\star - w_i$ and

$$\beta_e^2 \triangleq 2\delta^2, \quad \sigma_e^2 \triangleq \frac{2}{N}\sum_{n=1}^{N} \|\nabla_w Q(w^\star; x_n)\|^2. \tag{2.17}$$

$$\beta_{e4}^4 \triangleq 128\delta^4, \quad \sigma_{e4}^4 \triangleq \frac{8}{N}\sum_{n=1}^{N} \|\nabla_w Q(w^\star; x_n)\|^4. \tag{2.18}$$

Proof: We first prove (2.14). Since $\boldsymbol{w}_i \in \boldsymbol{\mathcal{F}}_i$ and $\boldsymbol{n}_i$ is selected uniformly, it holds that

$$\mathbb{E}[\boldsymbol{s}_i(\boldsymbol{w}_i)|\boldsymbol{\mathcal{F}}_i] = \frac{1}{N}\sum_{n=1}^{N} \nabla_w Q(w_i; x_n) - \nabla_w J(w_i) = 0. \tag{2.19}$$

Next we establish (2.15). Using Jensen's inequality:

$$\begin{aligned}
\mathbb{E}[\|\boldsymbol{s}_i(\boldsymbol{w}_{i-1})\|^2|\boldsymbol{\mathcal{F}}_i] =& \mathbb{E}[\|\nabla_w Q(\boldsymbol{w}_i; x_{\boldsymbol{n}_i}) - \nabla_w J(\boldsymbol{w}_{i-1})\|^2|\boldsymbol{\mathcal{F}}_i] \\
=& \mathbb{E}[\|\nabla_w Q(\boldsymbol{w}_i; x_{\boldsymbol{n}_i}) - \nabla_w Q(w^\star; x_{\boldsymbol{n}_i}) - \nabla_w J(\boldsymbol{w}_i) + \nabla_w Q(w^\star; x_{\boldsymbol{n}_i})\|^2|\boldsymbol{\mathcal{F}}_i] \\
\le& 2\mathbb{E}[\|\nabla_w Q(\boldsymbol{w}_{i-1}; x_{\boldsymbol{n}_i}) - \nabla_w Q(w^\star; x_{\boldsymbol{n}_i}) - \nabla_w J(\boldsymbol{w}_{i-1})\|^2|\boldsymbol{\mathcal{F}}_i] \\
&+ 2\mathbb{E}[\|\nabla_w Q(w^\star; x_{\boldsymbol{n}_i})\|^2|\boldsymbol{\mathcal{F}}_i].
\end{aligned} \tag{2.20}$$

where the last inequality holds because of Jensen's inequality. For the first term in the right-hand side of the above inequality, notice that

$$\mathbb{E}[\nabla_w Q(\boldsymbol{w}_i; x_{\boldsymbol{n}_i}) - \nabla_w Q(w^\star; x_{\boldsymbol{n}_i})|\boldsymbol{\mathcal{F}}_i] = \nabla_w J(\boldsymbol{w}_i). \tag{2.21}$$

Noting that for any random variable $\boldsymbol{x}$:

$$\mathbb{E}\|\boldsymbol{x} - \mathbb{E}\boldsymbol{x}\|^2 = \mathbb{E}\|\boldsymbol{x}\|^2 - \|\mathbb{E}\boldsymbol{x}\|^2 \le \mathbb{E}\|\boldsymbol{x}\|^2, \tag{2.22}$$

and using $\nabla_w J(w^\star) = 0$, we have

$$\begin{aligned}
&\mathbb{E}[\|\underbrace{\nabla_w Q(\boldsymbol{w}_i; x_{\boldsymbol{n}_i}) - \nabla_w Q(w^\star; x_{\boldsymbol{n}_i})}_{\boldsymbol{x}} - \underbrace{\nabla_w J(\boldsymbol{w}_i)}_{\mathbb{E}\boldsymbol{x}}\|^2|\boldsymbol{\mathcal{F}}_i] \\
&\le \mathbb{E}[\|\nabla_w Q(\boldsymbol{w}_i; x_{\boldsymbol{n}_i}) - \nabla_w Q(w^\star; x_{\boldsymbol{n}_i})\|^2|\boldsymbol{\mathcal{F}}_i] \\
&= \frac{1}{N}\sum_{n=1}^{N} \|\nabla_w Q(w_i; x_n) - \nabla_w Q(w^\star; x_n)\|^2 \\
&\overset{(a)}{\le} \delta^2\|w_i - w^\star\|^2 = \delta^2\|\widetilde{w}_i\|^2,
\end{aligned} \tag{2.23}$$

where (a) holds because of Assumption 2.1. Substituting (2.23) into (2.20), we obtain (2.15).

A similar argument can be used to establish the fourth-order moment property (2.16), which we omit for brevity. ■

2.1.3 Mean-square Stability and Performance of E-SG

Next we are ready to establish the mean-square stability of E-SG. Recall that the empirical problem (2.1) is the special case of the general stochastic optimization problem (1.1), and E-SG is the special case of O-SG. Meanwhile, we have also verified that E-SG gradient noise (2.11) satisfies all assumptions required in [11], now we can directly apply the convergence theorem of O-SG (Lemma 3.1 in [11]) to E-SG, and achieve its mean-square-error convergence property.

Theorem 2.1 (Mean-square-error stability of E-SG) *Under Assumption 2.1 and any step-size satisfying* $\mu < 2\nu/(\delta^2 + \beta_e^2) = 2\nu/3\delta^2$, *it holds that*

$$\mathbb{E}\|\widetilde{\boldsymbol{w}}_i\|^2 \leq \alpha^i \, \mathbb{E}\|w_0 - w^\star\|^2 + O(\mu). \tag{2.24}$$

where $\alpha = 1 - 2\nu\mu + 3\delta^2\mu^2 \in (0, 1)$. ■

Theorem 2.1 states that E-SG converges exponentially fast to a small neighborhood around $w^\star$ of size $\mathrm{O}(\mu)$. From (2.24), it is also observed that E-SG is not sensitive to the initial staring point w_0 because $\|w_0 - w^\star\|^2$ will diminish exponentially fast. Theorem 2.1 does not provide an accurate expression for the steady-state performance of E-SG. More is needed to arrive at this expression. Here we appeal to Theorem 4.7 from [11] to derive the expression for the steady-state performance of E-SG. By steady-state we mean the algorithm is applied repeatedly in random passes over the finite training data. We denote the Hessian of the empirical risk (2.1) at $w^\star$ by

$$H \;\triangleq\; \nabla_w^2 J(w^\star). \tag{2.25}$$

Theorem 2.2 (Steady-state performance) *Assume the conditions under Assumptions*

2.1 and 2.2 hold. When the step-size is sufficiently small, the MSD and ER metrics to first-order in μ for the E-SG algorithm (2.2) *are given by the following expressions:*

$$\text{MSD} \triangleq \limsup_{i\to\infty} \mathbb{E}[\|\boldsymbol{w}_i - \boldsymbol{w}^\star\|^2] = \frac{\mu}{2}\text{Tr}(H^{-1}R_s), \tag{2.26}$$

$$\text{ER} \triangleq \limsup_{i\to\infty} \mathbb{E}[J(\boldsymbol{w}_i) - J(w^\star)] = \frac{\mu}{4}\text{Tr}(R_s), \tag{2.27}$$

where H is defined in (2.25) *and R_s is defined in* (2.12). ■

The MSD expression (2.26) is tighter than the bound given in [1], which is written as

$$\limsup_{i\to\infty} \mathbb{E}\|\widetilde{\boldsymbol{w}}_i\|^2 \leq \frac{\mu\text{Tr}(R_s)}{\nu(1-\mu\max_n\{\delta_n\})} = \frac{\mu\text{Tr}(R_s)}{\nu(1-\mu\delta)}. \tag{2.28}$$

Since $J(w)$ is ν-strongly convex, we have $H \geq \nu I$. Therefore,

$$\frac{\mu}{2}\text{Tr}(H^{-1}R_s) \leq \frac{\mu}{2\nu}\text{Tr}(R_s) < \frac{\mu\text{Tr}(R_s)}{\nu(1-\mu\delta)}, \tag{2.29}$$

where the last inequality holds because $1-\mu\delta < 2$. Relation (2.29) shows that our MSD expression (2.26) is tighter. As a result, expressions (2.26) and (2.27) are more helpful to determine a proper step-size when a certain accuracy ϵ is required for the MSD or ER performance.

2.2 Mini-batch Stochastic Gradient Learning

In this section, we derive the MSD and ER performance expressions for mini-batch gradient descent. Following arguments similar to Section 2.1, we will also interpret mini-batch gradient descent as a special case of O-SG, and then refer to the theoretical results in [11].

Suppose we have B independent discrete random variables $\boldsymbol{x}_e^{(1)}, \boldsymbol{x}_e^{(2)}, \cdots, \boldsymbol{x}_e^{(B)}$, each with the same distribution as $\boldsymbol{x}_e$ that is defined from Section 2.1.1. Using these variables, we

define

$$\mathcal{Q}(w;\{\boldsymbol{x}_e^{(1)},\boldsymbol{x}_e^{(2)},\cdots,\boldsymbol{x}_e^{(B)}\}) \;\triangleq\; \frac{1}{B}\sum_{j=1}^{B} Q(w;\boldsymbol{x}_e^{(j)}), \tag{2.30}$$

and note that

$$\mathbb{E}[\mathcal{Q}(w;\{\boldsymbol{x}_e^{(1)},\boldsymbol{x}_e^{(2)},\cdots,\boldsymbol{x}_e^{(B)}\})] \overset{(2.30)}{=} \frac{1}{B}\sum_{j=1}^{B}\mathbb{E}[Q(w;\boldsymbol{x}_e^{(j)})] \overset{(2.5)}{=} \frac{1}{B}\sum_{j=1}^{B} J(w) = J(w). \tag{2.31}$$

It follows that the empirical problem (2.1) can also be rewritten as:

$$\min_{w\in\mathbb{R}^M}\; J(w) = \mathbb{E}[\mathcal{Q}(w;\{\boldsymbol{x}_e^{(1)},\boldsymbol{x}_e^{(2)},\cdots,\boldsymbol{x}_e^{(B)}\})], \tag{2.32}$$

which is a stochastic optimization problem. We can therefore apply the O-SG algorithm to seek its minimizer, which leads to the mini-batch gradient descent algorithm:

$$\begin{aligned}
\boldsymbol{w}_{i+1} &= \boldsymbol{w}_i - \mu\nabla_w \mathcal{Q}(\boldsymbol{w}_i;\{\boldsymbol{x}_{e,i}^{(1)},\boldsymbol{x}_{e,i}^{(2)},\cdots,\boldsymbol{x}_{e,i}^{(B)}\}) \\
&= \boldsymbol{w}_i - \frac{\mu}{B}\sum_{j=1}^{B}\nabla_w Q(\boldsymbol{w}_i;\boldsymbol{x}_{e,i}^{(j)}) \\
&\overset{(a)}{=} \boldsymbol{w}_i - \frac{\mu}{B}\sum_{j=1}^{B}\nabla_w Q(\boldsymbol{w}_i;x_{\boldsymbol{n}_i(j)}),
\end{aligned} \tag{2.33}$$

where $\boldsymbol{x}_{e,i}^{(j)}$ is the instantaneous realization of the random variable $\boldsymbol{x}_e^{(j)}$ at iteration i. Equality (a) holds because, similarly to (2.6) and (2.7), we redefine $\boldsymbol{x}_{e,i}^{(j)}$ as $x_{\boldsymbol{n}_i(j)}$ where the random variables $\{\boldsymbol{n}_i(j)\}_{j=1}^{B}$ are mutually independent with the same pmf as $\boldsymbol{n}_i$. During the implementation of recursion (2.33), we will sample B data with replacement at each iteration, and then compute their average.

2.2.1 Gradient Noise and its Moments

According to the mini-batch recursion (2.33), the gradient noise is

$$\boldsymbol{s}_i^b(\boldsymbol{w}_i)=\frac{1}{B}\sum_{j=1}^{B}\nabla_w Q(\boldsymbol{w}_i;x_{\boldsymbol{n}_i(j)})-\nabla_w J(\boldsymbol{w}_i). \tag{2.34}$$

The following result extends Lemma 2.1 to the mini-batch method.

Lemma 2.2 (Gradient noise properties) *The first and second-order moments of the gradient noise $\boldsymbol{s}_i^b(\boldsymbol{w}_i)$ defined in (2.34) satisfy:*

$$\mathbb{E}[\boldsymbol{s}_i^b(\boldsymbol{w}_i)|\boldsymbol{\mathcal{F}}_i]=0, \tag{2.35}$$

$$\mathbb{E}[\|\boldsymbol{s}_i^b(\boldsymbol{w}_i)\|^2|\boldsymbol{\mathcal{F}}_i]\leq\beta_b^2\|\widetilde{w}_i\|^2+\sigma_b^2, \tag{2.36}$$

$$\mathbb{E}[\|\boldsymbol{s}_i^b(\boldsymbol{w}_i)\|^4|\boldsymbol{\mathcal{F}}_i]\leq\beta_{b4}^4\|\widetilde{w}_i\|^2+\sigma_{b4}^4, \tag{2.37}$$

where $\widetilde{w}_i=w^\star-w_i$,

$$\beta_b^2\ \triangleq\ \beta_e^2/B,\qquad \sigma_b^2\ \triangleq\ \sigma_e^2/B, \tag{2.38}$$

$$\beta_{b4}^4\ \triangleq\ \beta_{e4}^4,\qquad \sigma_{b4}^4\ \triangleq\ \sigma_{e4}^4. \tag{2.39}$$

and β_e^2, σ_e^2, β_{b4}^4, and σ_{b4}^4 are defined in Lemma 2.1. ■

Proof: The argument for (2.35) is similar to (2.19). To prove (2.36), we start by noting that

$$\begin{aligned}\mathbb{E}[\|\boldsymbol{s}_i^b(\boldsymbol{w}_i)\|^2|\boldsymbol{\mathcal{F}}_i]&=\mathbb{E}[\|\frac{1}{B}\sum_{j=1}^{B}\nabla_w Q(\boldsymbol{w}_i;x_{\boldsymbol{n}_i(j)})-\nabla_w J_{\text{emp}}(\boldsymbol{w}_i)\|^2|\boldsymbol{\mathcal{F}}_i]\\&=\mathbb{E}\left[\left\|\frac{1}{B}\sum_{j=1}^{B}\Big(\nabla_w Q(\boldsymbol{w}_i;x_{\boldsymbol{n}_i(j)})-\nabla J(\boldsymbol{w}_i)\Big)\right\|^2\Bigg|\boldsymbol{\mathcal{F}}_i\right]\\&=\frac{1}{B^2}\mathbb{E}\left[\sum_{j=1}^{B}\left\|\nabla_w Q(\boldsymbol{w}_i;x_{\boldsymbol{n}_i(j)})-\nabla J(\boldsymbol{w}_i)\right\|^2\Bigg|\boldsymbol{\mathcal{F}}_i\right]\end{aligned}$$

$$
\begin{aligned}
&+\frac{1}{B^2}\mathbb{E}\left[\sum_{j=1}^{B}\sum_{k\neq j}\Big(\nabla_w Q(\boldsymbol{w}_i;x_{\boldsymbol{n}_i(j)})-\nabla J(\boldsymbol{w}_i)\Big)^{\mathsf{T}}\right.\\
&\qquad\left.\Big(\nabla_w Q(\boldsymbol{w}_i;x_{\boldsymbol{n}_i(k)})-\nabla J(\boldsymbol{w}_i)\Big)\,\Big|\,\boldsymbol{\mathcal{F}}_i\right]\\
&\overset{(a)}{=}\frac{1}{B^2}\sum_{j=1}^{B}\mathbb{E}\left[\Big\|\nabla_w Q(\boldsymbol{w}_i;x_{\boldsymbol{n}_i(j)})-\nabla J(\boldsymbol{w}_i)\Big\|^2\,\Big|\,\boldsymbol{\mathcal{F}}_i\right] \qquad (2.40)
\end{aligned}
$$

where (a) holds because when $k \neq j$, $\boldsymbol{n}_i(j)$ is independent of $\boldsymbol{n}_i(k)$. Now using property (2.15) from Lemma 2.1 in (2.40) gives (2.36). The derivation for the fourth-order moment result follows from Jensen's inequality and (2.16). We omit the proof for brevity. ■

One important observation is that, with the mini-batch technique, the magnitude of the second-order moment of the gradient noise is reduced to $1/B$ of its original magnitude (see (2.38)), which suggests that we should expect both the MSD and ER of mini-batch implementations to improve by a factor of B. The analysis in the next section confirms this conclusion.

2.2.2 Performance of Mini-batch Stochastic Gradient Learning

First, the limiting covariance matrix of the mini batch gradient noise process is given by

$$
R_s^b = \frac{1}{B^2}\sum_{j=1}^{B} R_s = \frac{1}{B}R_s. \qquad (2.41)
$$

Then, using Theorem 4.7 from [11] we deduce the following.

Theorem 2.3 (Steady-state performance) *Under Assumptions 2.1 and 2.2, for a sufficiently small step-size, the MSD and ER metrics for the mini-batch method* (2.33) *are given by:*

$$
\begin{aligned}
\mathrm{MSD}_b &= \frac{\mu}{2}\mathrm{Tr}(H^{-1}R_s^b) = \frac{\mu}{2B}\mathrm{Tr}(H^{-1}R_s), \qquad (2.42)\\
\mathrm{ER}_b &= \frac{\mu}{4}\mathrm{Tr}(R_s^b) = \frac{\mu}{4B}\mathrm{Tr}(R_s), \qquad (2.43)
\end{aligned}
$$

where H is defined in (2.25) *and R_s is defined in* (2.41)*. Moreover, the algorithm converges*

at an exponential rate:

$$\alpha_b = 1 - 2\nu\mu + (1 + 2/B)\delta^2\mu^2. \tag{2.44}$$

■

2.3 Optimal Importance Sampling

We have assumed so far that the data samples in an empirical SG implementation are selected uniformly at random, according to (2.4). However, we can consider other selection policies in order to enhance performance. The works [1, 2] proposed to measure the importance of each sample according to its Lipschitz constant δ_n in (6.52). Specifically, they suggest selecting the sampling probability according to

$$p(n) = \frac{\delta_n}{\sum_{m=1}^{N} \delta_m}, \tag{2.45}$$

where $p(n)$ is the sampling probability of data x_n. This scheme assumes knowledge of the Lipschitz constants, which is usually not available in advance or even known. Moreover, this importance sampling method is not optimal, as the ensuing discussion will show where we derive the optimal sampling algorithm.

Let us denote the new pmf for the random variable $\boldsymbol{n}$ that we wish to determine optimally by

$$p(\boldsymbol{n}) = \begin{cases} \alpha(1), & \text{if } \boldsymbol{n} = 1, \\ \alpha(2), & \text{if } \boldsymbol{n} = 2, \\ \vdots & \vdots \\ \alpha(N), & \text{if } \boldsymbol{n} = N, \end{cases} \tag{2.46}$$

where $\alpha(n)$ is the sampling probability for data x_n, and it holds that the $\{\alpha(n)\}$ add up to one. With this new pmf for $\boldsymbol{n}$, the empirical problem (2.1) can be interpreted as the

following stochastic optimization problem

$$\min_{w\in\mathbb{R}^M} \; J(w) = \sum_{n=1}^{N} \frac{\alpha(n)}{\alpha(n)N} Q(w; x_n) = \mathbb{E}_{\boldsymbol{n}}[Q^l(w; x_{\boldsymbol{n}})], \tag{2.47}$$

where we defined

$$Q^l(w; x_n) \;\triangleq\; \frac{1}{\alpha(n)N} Q(w; x_n). \tag{2.48}$$

Now if we apply O-SG to solve problem (2.47), we obtain the following importance sampling recursion:

$$\begin{aligned} \boldsymbol{w}_{i+1} &= \boldsymbol{w}_i - \mu \nabla_w Q^l(\boldsymbol{w}_i; x_{\boldsymbol{n}_i}) \\ &= \boldsymbol{w}_i - \frac{\mu}{p(\boldsymbol{n}_i)N} \nabla_w Q(\boldsymbol{w}_i; x_{\boldsymbol{n}_i}). \end{aligned} \tag{2.49}$$

Next we will explain how to choose $p(\boldsymbol{n}_i)$ such that the above recursion can reach optimal steady-state performance. First, the gradient noise of the importance sampling approach is given by:

$$\boldsymbol{s}_i^l(\boldsymbol{w}_i) = \nabla_w Q^l(\boldsymbol{w}_i; x_{\boldsymbol{n}_i}) - \nabla_w J(\boldsymbol{w}_i) \tag{2.50}$$

Lemma 2.3 (Gradient Noise Property) *The gradient noise process in (2.50) satisfies the following conditions:*

$$\mathbb{E}[\boldsymbol{s}_i^l(\boldsymbol{w}_i)|\boldsymbol{\mathcal{F}}_i] = 0, \tag{2.51}$$

$$\mathbb{E}[\|\boldsymbol{s}_i^l(\boldsymbol{w}_i)\|^2|\boldsymbol{\mathcal{F}}_i] \le \beta_l^2 \|\widetilde{w}_i\|^2 + \sigma_l^2, \tag{2.52}$$

where $\widetilde{w}_i = w^\star - w_i$ *and*

$$\beta_l^2 \;\triangleq\; 2\sum_{n=1}^{N} \frac{\delta^2}{\alpha(n)N^2}, \quad \sigma_l^2 \;\triangleq\; 2\sum_{n=1}^{N} \frac{1}{\alpha(n)N^2} \|\nabla_w Q(w^\star; x_n)\|^2$$

■

This Lemma can be established by following arguments similar to those used in Lemmas 2.1 and 2.2. Now, calling upon Theorem 4.7 from [11] we arrive at the following expression for the proposed importance sampling recursion (2.49):

$$\mathrm{E}R_l = \frac{\mu}{4}\mathrm{Tr}(R_s^l) = \frac{\mu}{4}\sum_{n=1}^{N}\frac{1}{\alpha(n)N^2}\|\nabla_w Q(w^\star; x_n)\|^2 \tag{2.53}$$

We can minimize this expression over the $\{\alpha(n)\}$ and solve:

$$\begin{aligned} \min_{\alpha} \quad & \sum_{n=1}^{N}\frac{1}{\alpha(n)}\|\nabla_w Q(w^\star; x_n)\|^2 \\ \text{s.}t. \quad & \sum_{n=1}^{N}\alpha(n) = 1, \;\; 0 \le \alpha(n) \le 1, \quad n = 1, 2, \cdots, N. \end{aligned} \tag{2.54}$$

Fortunately, this problem has a closed-form solution, which can be derived by the Lagrangian multiplier method:

$$\alpha^\star(n) = \frac{\|\nabla_w Q(w^\star; x_n)\|}{\sum_{m=1}^{N}\|\nabla_w Q(w^\star; x_m)\|} \tag{2.55}$$

Substituting into (2.53) yields the optimal ER value:

$$\mathrm{E}R_l^\star = \frac{\mu}{4}\left(\sum_{n=1}^{N}\frac{1}{N}\|\nabla_w Q(w^\star; x_n)\|\right)^2 \tag{2.56}$$

From Jensen's inequality, we can verify that $\mathrm{ER}_l^\star$ is always smaller than or equal to the ER of E-SG derived earlier in (2.27).

Although we determined the optimal pmf in (2.56), one practical problem is that the expression for $\alpha^\star(n)$ depends on the unknown $w^\star$. This problem can be overcome by replacing the minimizer by its estimate, which leads to an adaptive importance sampling method:

$$\alpha_i(n) = \frac{\|\nabla_w Q(w_i; x_n)\|}{\sum_{m=1}^{N}\|\nabla_w Q(w_i; x_m)\|}. \tag{2.57}$$

Table 2.1: Listing of optimal adaptive importance sampling for SG

Optimal adaptive importance sampling for SG
Initialization:
$\boldsymbol{\psi}_0$ is initialized to be some large positive vector;
θ_0 is initialized as the sum of the entires in $\boldsymbol{\psi}_0$;
p_0 is initialized as uniform distribution;
for $i = 1, 2, 3, \ldots$
Pick $\boldsymbol{n}_i$ according to sampling probability p_i;
Update $\boldsymbol{\psi}_i$ and θ_i according to (2.58) and (2.59) respectively;
Update sampling probability $p_i = \boldsymbol{\psi}_i/\theta_i$;
Update $\boldsymbol{w}_{i+1}$ according to (2.49)
end

Expression (2.57) is still inefficient to update because at each iteration we have to compute $\|\nabla_w Q(w_i; x_n)\|$ for all data samples and then calculate the average. To reach an efficient update, we introduce an auxiliary variable $\boldsymbol{\psi} \in \mathbb{R}^N$, with its nth entry updated as follows:

$$\boldsymbol{\psi}_i(n) = \begin{cases} \gamma \boldsymbol{\psi}_{i-1}(n) + (1-\gamma)\|\nabla_w Q(\boldsymbol{w}_i; x_n)\|, & \text{if } n = \boldsymbol{n}_i \\ \boldsymbol{\psi}_{i-1}(n), & \text{if } n \neq \boldsymbol{n}_i \end{cases} \tag{2.58}$$

where $\gamma \in (0, 1)$; in the simulations we selected $\gamma \in (0.1, 0.5)$. Note that each entry $\boldsymbol{\psi}_i(n)$ is an estimate of $\|\nabla_w Q(\boldsymbol{w}_i; x_n)\|$. Note further that at iteration i, only one entry of $\boldsymbol{\psi}_i$ is updated, and hence this update is cheap. We also update a scalar θ to maintain the sum of $\boldsymbol{\psi}$. Suppose $\boldsymbol{n}_i$ is picked up at iteration i, then

$$\begin{aligned} \theta_i &= \sum_{n=1}^{N} \boldsymbol{\psi}_i(n) = \sum_{n=1}^{N} \boldsymbol{\psi}_i(n) + \boldsymbol{\psi}_i(\boldsymbol{n}_i) - \boldsymbol{\psi}_{i-1}(\boldsymbol{n}_i) \\ &\overset{(2.58)}{=} \theta_i + (1-\gamma)\left(\|\nabla_w Q(\boldsymbol{w}_i; x_n)\| - \boldsymbol{\psi}_{i-1}(\boldsymbol{n}_i)\right). \end{aligned} \tag{2.59}$$

Note that each update of θ only requires $O(1)$ operations, which is also cheap. The algorithm is summarized in Table 2.1, where $p_i \in \mathbb{R}^N$ is the sampling probability vector with each entry $p_i(n)$ indicating the probability that data x_n is selected.

In the algorithm algorithm, we initialize $\boldsymbol{\psi}_0$ to large entries so that p_i is not too small for some indices. In this way, we can guarantee that all data samples are accessed with large enough probability during the initial stages. The key feature of this algorithm is that it does not depend on any pre-knowledge of each data sample (such as the Lipschitz constants needed in [1,2]), and can automatically learn the optimal sampling probability distribution. Moreover, the algorithm is very efficient in computational cost.

2.4 Numerical Experiments

We illustrate the results by considering the regularized logistic regression problem:

$$J(w) = \frac{\rho}{2}\|w\|^2 + \frac{1}{N}\sum_{n=1}^{N} \ln\left(1 + \exp(-\gamma(n)h_n^{\mathsf{T}}w)\right), \tag{2.60}$$

where $h_n \in \mathbb{R}^{10}$ is the feature and $\gamma(n) \in \{\pm 1\}$ is the label scalar. In the simulation, we generate a random data set $\{h_n, \gamma(n)\}$ with $N = 500$. We set $\rho = 0.01$ and $\mu = 0.01$. We run the empirical SG and mini-batch algorithms over 25 epochs. All simulation results shown below are averaged over 100 trials. From Fig 2.1, it is clear that our bound is significantly tighter than the bound from [1], which is also shown in (2.28). We also observe that the MSD performance is inversely proportionally to the size of the mini-batch, as predicted by Theorem 2.3. Moreover, the figure shows that our theoretical performance expressions match well with the simulated results.

Next, in Fig. 2.2 we illustrate the behavior of our optimal importance sampling algorithm with the same problem setting. All algorithms use a 10 mini-batch size. The red curve is the standard SG learning curve, which is used as reference; The blue curve is using the fixed optimal importance sampling probability, which is precalculated with the $w^\star$ information (2.55). The green curve is our proposed adaptive importance sampling method, which is seen to be as good as the optimal solution. We also compare against the resampling technique from [1, 2], which use the Lipschitz constants. The result is the black curve, which is only matching the performance of the standard SG implementation and is away from the optimal

performance.

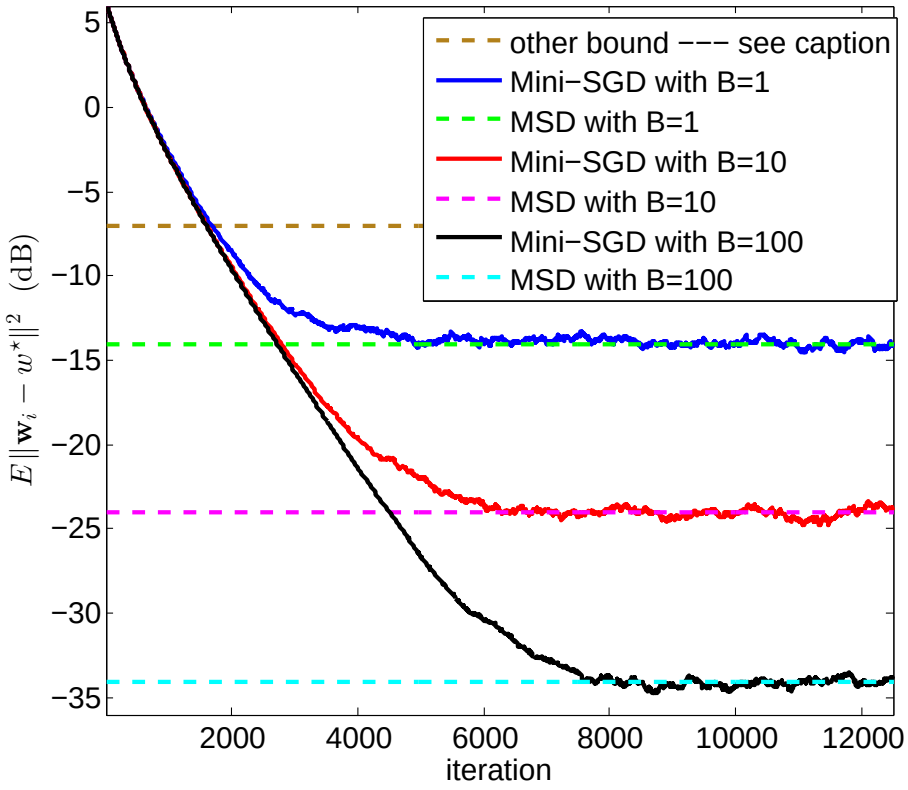

Figure 2.1: Convergence behavior of mini-batch SG for regularized logistic regression problem. B indicates the size of the mini-batch. The dotted MSD levels are calculated according (2.42) except for the brown line, which is calculated according to [1].

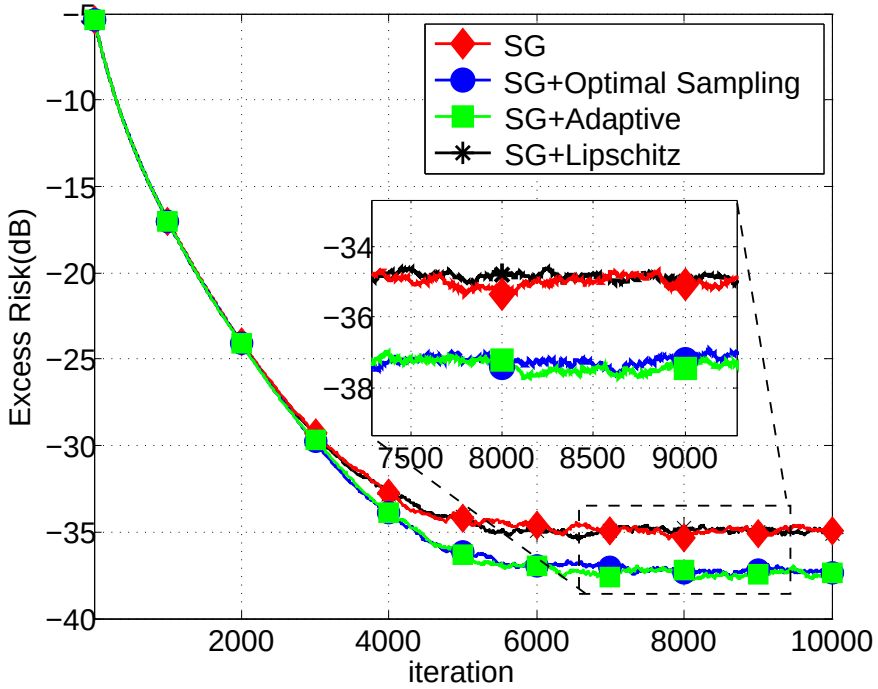

Figure 2.2: Optimal adaptive importance sampling algorithm for regularized logistic regression problem. In the figure legend, SG refers to iteration (2.2); SG+Optimal Sampling refers to (2.55); SG+Adaptive refers to (2.58)—(2.59); SG+Lipschitz is the algorithm proposed in [2].

2.5 Conclusion

This chapter establishes a useful connection between empirical stochastic gradient methods for learning from finite data samples, and online stochastic gradient methods for learning from streaming data. Using performance expressions for the excess risk (ER), an optimal sampling strategy is devised to attain the best ER performance. Simulation runs illustrate the results.

CHAPTER 3

Stochastic Learning under Random Reshuffling

This chapter focuses on random reshuffling algorithms with constant step-size case and strongly convex loss functions. In this case, convergence is guaranteed to a small neighborhood of the optimizer albeit at a linear rate. The analysis establishes analytically that random reshuffling outperforms uniform sampling by showing explicitly that iterates approach a smaller neighborhood of size $O(\mu^2)$ around the minimizer rather than $O(\mu)$. Furthermore, we derive an analytical expression for the steady-state mean-square-error performance of the algorithm, which is exact for quadratic risks and good approximation for general risks. This helps clarify in greater detail the differences between sampling with and without replacement that can be described through t$anh(\cdot)$ functions. We also explain the periodic behavior that is observed in random reshuffling implementations in this chapter.

3.1 Motivation

It has been noted in the literature [34, 35, 62, 63] that incorporating random reshuffling into the gradient descent implementation helps achieve better performance. More broadly than in the case of the pure SGD algorithm, it has also been observed that applying random reshuffling in variance-reduction algorithms, like SVRG [64], SAGA [39], can accelerate the convergence speed [31, 40, 43]. The reshuffling technique has also been applied in distributed system to reduce the communication and computation cost [65].

Several recent works [34–36] have pursued justifications for the enhanced behavior of random reshuffling implementations over independent sampling (with replacement). The work [35] examined the convergence rate of the learning process under diminishing step-sizes,

i.e., $\mu(i) = c/i$, where c is some positive constant. It analytically showed that, for strongly convex objective functions, the convergence rate under random reshuffling can be improved from $O(1/i)$ in vanilla SGD [37] to $O(1/i^2)$. The incremental gradient methods [3,66], which can be viewed as the deterministic version of random reshuffling, shares similar conclusions, i.e., random reshuffling helps accelerate the convergence rate from $O(1/i)$ to $O(1/i^2)$ under decaying step-sizes. Also, in the work [36], it establishes that random reshuffling will not degrade performance relative to the stochastic gradient descent implementation, provided the number of epochs is not too large. In this chapter, we focus on a different setting than [34–36] involving random reshuffling under *constant* rather than decaying step-sizes. In this case, convergence is only guaranteed to a small neighborhood of the optimizer albeit at a linear rate. The analysis will establish analytically that random reshuffling outperforms independent sampling (with replacement) by showing that the mean-square-error of the iterate at the end of each run in the random reshuffling strategy will be in the order of $O(\mu^2)$. This is a significant improvement over the performance of traditional stochastic gradient descent, which is $O(\mu)$ [13]. Furthermore, we derive an analytical expression for the steady-state mean-square-error performance of the algorithm, which is exact for quadratic risks and provides a good approximation for general risks. This helps clarify in greater detail the differences between sampling with and without replacement We also explain the periodic behavior that is observed in random reshuffling implementations.

3.2 Stability of SGD Under Random Reshuffling

3.2.1 Properties of the Gradient Approximation

We start by examining the properties of the stochastic gradient $\nabla_w Q(\boldsymbol{w}_i^k; x_{\boldsymbol{\sigma}^k(i)})$ under random reshuffling. One main source of difficulty that we shall encounter in the analysis of performance under random reshuffling is the fact that a single sample of the stochastic gradient $\nabla_w Q(\boldsymbol{w}_i^k; x_{\boldsymbol{\sigma}^k(i)})$ is now a *biased* estimate of the true gradient and, moreover, it is no longer independent of past selections, $\boldsymbol{\sigma}^k(1:i)$. This is in contrast to implementations where samples are picked independently at every iteration. Indeed, note that conditioned

on previously picked data and on the previous iterate, we have:

$$\begin{aligned}\mathbb{E}\left[\nabla_w Q_{\boldsymbol{\sigma}^k(i)}(\boldsymbol{w}_i^k)\,|\,\boldsymbol{w}_i^k,\boldsymbol{\sigma}^k(1:i-1)\right] &= \frac{1}{N-i+1}\sum_{n\notin\boldsymbol{\sigma}^k(1:i-1)}\nabla_w Q(\boldsymbol{w}_i^k)\\ &\neq \nabla J(\boldsymbol{w}_i^k)\end{aligned}\tag{3.1}$$

The difference (3.1) is generally nonzero in view of the definition (1.2). For the first iteration of every epoch however, it can be verified that the following holds:

$$\begin{aligned}\mathbb{E}\left[\nabla_w Q_{\boldsymbol{\sigma}^k(i)}(\boldsymbol{w}_0^k)\,\middle|\,\boldsymbol{w}_0^k\right] &\overset{(1.7)}{=} \frac{1}{N}\sum_{n=1}^{N} Q(\boldsymbol{w}_0^k;x_n)\\ &\overset{(1.2)}{=} \nabla J(\boldsymbol{w}_0^k)\end{aligned}\tag{3.2}$$

since at the beginning of one epoch, no data has been selected yet. Perhaps surprisingly, we will be showing that the biased construction of the stochastic gradient estimate not only does not hurt the performance of the algorithm, but instead significantly improves it. In large part, the analysis will revolve around considering the accuracy of the gradient approximation over an entire epoch, rather than focusing on single samples at a time. Recall that by construction in random reshuffling, every sample is picked once and only once over one epoch. This means that the *sample average* (rather than the true mean) of the gradient noise process is zero since

$$\frac{1}{N}\sum_{i=1}^{N}\nabla_w Q(\boldsymbol{w};x_{\boldsymbol{\sigma}^k(i)}) = \nabla J(\boldsymbol{w})\tag{3.3}$$

for any $\boldsymbol{w}$ and any reshuffling order $\boldsymbol{\sigma}^k$. This property will become key in the analysis.

3.2.2 Convergence Analysis

We can now establish a key convergence and performance property for the random reshuffling algorithm, which provides solid analytical justification for its observed improved performance in practice.

To begin with, we assume that the risk function satisfies the following conditions, which are automatically satisfied by many learning problems of interest, such as mean-square-error or logistic regression analysis and their regularized versions — see, e.g., [9–11, 28, 67].

Assumption 3.1 (Condition on loss function) *It is assumed that $Q(w; x_n)$ is differentiable and has a δ_n-Lipschitz continuous gradient, i.e., for every $n = 1, \ldots, N$ and any $w_1, w_2 \in \mathbb{R}^M$:*

$$\|\nabla_w Q(w_1; x_n) - \nabla_w Q(w_2; x_n)\| \leq \delta_n \|w_1 - w_2\| \tag{3.4}$$

where $\delta_n > 0$. We also assume $J(w)$ is ν-strongly convex:

$$\Big(\nabla_w J(w_1) - \nabla_w J(w_2)\Big)^{\mathsf{T}} (w_1 - w_2) \geq \nu \|w_1 - w_2\|^2 \tag{3.5}$$

■

If we introduce $\delta = \max\{\delta_1, \delta_2, \cdots, \delta_N\}$, then each $\nabla_w Q(w; x_n)$ and $\nabla_w J(w)$ are also δ-Lipschitz continuous.

The following theorem focuses on the convergence of the *starting* point of each epoch and establishes in (3.7) that it actually approaches a smaller neighborhood of size $O(\mu^2)$ around $w^\star$. Afterwards, using this result, we also show that the same $O(\mu^2)-$performance level holds for *all* iterates $\boldsymbol{w}_i^k$ and not just for the starting points of the epochs.

To simplify the notation, we introduce the constant $\mathcal{K}$, which is the gradient noise variance at optimal point $w^\star$:

$$\mathcal{K} \triangleq \frac{1}{N} \sum_{n=1}^{N} \|\nabla_w Q(w^\star; x_n)\|^2 \tag{3.6}$$

Theorem 3.1 (Stability of starting points) *Under assumption 3.1, the starting point of*

each run satisfies

$$\limsup_{k\to\infty} \mathbb{E}\,\|\boldsymbol{w}_0^k - w^\star\|^2 \leq \frac{4\mu^2\delta^2 N^2}{\nu^2}\mathcal{K} = O(\mu^2) \tag{3.7}$$

when the step-size is sufficiently small, namely, for $\mu \leq \frac{\nu}{3\delta^2 N}$.

Proof: Note first that

$$\begin{aligned}
\boldsymbol{w}_0^{k+1} &\overset{\Delta}{=} \boldsymbol{w}_N^k \\
&\overset{(1.4)}{=} \boldsymbol{w}_{N-1}^k - \mu\nabla_w Q(\boldsymbol{w}_{N-1}^k; x_{\boldsymbol{\sigma}^k(N)}) \\
&\;\;\vdots \\
&= \boldsymbol{w}_0^k - \mu\sum_{i=0}^{N-1}\nabla_w Q(\boldsymbol{w}_i^k; x_{\boldsymbol{\sigma}^k(i)}) \\
&\overset{(3.3)}{=} \boldsymbol{w}_0^k - \mu N\nabla_w J(\boldsymbol{w}_0^k) - \mu\sum_{i=0}^{N-1}\underbrace{\big(\nabla_w Q(\boldsymbol{w}_i^k; x_{\boldsymbol{\sigma}^k(i)}) - \nabla_w Q(\boldsymbol{w}_0^k; x_{\boldsymbol{\sigma}^k(i)})\big)}_{\overset{\Delta}{=} g_{\boldsymbol{\sigma}^k(i)}(\boldsymbol{w}_i^k)}
\end{aligned} \tag{3.8}$$

where we denote by $g_{\boldsymbol{\sigma}^k(i)}(\boldsymbol{w}_i^k)$ the incremental gradient noise which is the mismatch between the gradient approximations evaluated at $\boldsymbol{w}_0^k$ and $\boldsymbol{w}_i^k$. Next, we introduce the error vector:

$$\widetilde{\boldsymbol{w}}_0^k \;\overset{\Delta}{=}\; w^\star - \boldsymbol{w}_0^k \tag{3.9}$$

and let $0 < t < 1$ be any scalar that we will specify further below. Subtracting $w^\star$ from both sides of (3.8), squaring, and using Jensen's inequality in step (a) below we get:

$$\begin{aligned}
\|\widetilde{\boldsymbol{w}}_0^{k+1}\|^2 &= \left\|\widetilde{\boldsymbol{w}}_0^k + \mu N\nabla_w J(\boldsymbol{w}_0^k) + \mu\sum_{i=1}^{N} g_{\boldsymbol{\sigma}^k(i)}(\boldsymbol{w}_{i-1}^k)\right\|^2 \\
&\overset{(a)}{\leq} \frac{1}{t}\|\widetilde{\boldsymbol{w}}_0^k + \mu N\nabla_w J(\boldsymbol{w}_0^k)\|^2 + \frac{\mu^2}{1-t}\left\|\sum_{i=1}^{N} g_{\boldsymbol{\sigma}^k(i)}(\boldsymbol{w}_{i-1}^k)\right\|^2 \\
&\overset{(b)}{\leq} \frac{1}{t}\left\|\widetilde{\boldsymbol{w}}_0^k + \mu N\nabla_w J(\boldsymbol{w}_0^k)\right\|^2 + \frac{\mu^2 N}{1-t}\left(\sum_{i=1}^{N}\left\|g_{\boldsymbol{\sigma}^k(i)}(\boldsymbol{w}_{i-1}^k)\right\|^2\right)
\end{aligned} \tag{3.10}$$

where step (a) exploits the Jensen's inequality:

$$\|a+b\|^2 = \left\|\frac{t}{t}a + \frac{1-t}{1-t}b\right\|^2 \leq \frac{1}{t}\|a\| + \frac{1}{1-t}\|b\|^2 \tag{3.11}$$

and step (b) uses the fact that:

$$\left\|\sum_{i=1}^{N} x_i\right\|^2 = N^2\left\|\sum_{i=1}^{N}\frac{1}{N}x_i\right\|^2 \leq N\sum_{i=1}^{N}\|x_i\|^2 \tag{3.12}$$

We show in Appendix 3.A that the rightmost term in (3.10) can be bounded by:

$$\sum_{i=1}^{N}\left\|g_{\boldsymbol{\sigma}^k(i)}(\boldsymbol{w}_{i-1}^k)\right\|^2 \leq \frac{\mu^2\delta^2N^3}{1-2\mu^2\delta^2N^2}\left(2\delta^2\|\widetilde{\boldsymbol{w}}_0^k\|^2 + \mathcal{K}\right) \tag{3.13}$$

while for the first term in (3.10) we have

$$\begin{aligned}\left\|\widetilde{\boldsymbol{w}}_0^k + \mu N\nabla J(\boldsymbol{w}_0^k)\right\|^2 =& \|\widetilde{\boldsymbol{w}}_0^k\|^2 + \mu^2N^2\|\nabla J(\boldsymbol{w}_0^k)\|^2 + 2\mu N(\widetilde{\boldsymbol{w}}_0^k)^{\mathsf{T}}\nabla J(\boldsymbol{w}_0^k)\\ \leq& \left(1-2\mu N\frac{\nu\delta}{\delta+\nu}\right)\|\widetilde{\boldsymbol{w}}_0^k\|^2 + \mu N(\mu N - \frac{2}{\delta+\nu})\|\nabla J(\boldsymbol{w}_0^k)\|^2\end{aligned} \tag{3.14}$$

where in the first inequality we exploit the co-coercivity inequality [68] that

$$(\nabla J(x)-\nabla J(y))^{\mathsf{T}}(x-y) \geq \frac{\nu\delta}{\delta+\nu}\|x-y\|^2 + \frac{1}{\delta+\nu}\|\nabla J(x)-\nabla J(y)\|^2 \tag{3.15}$$

Next we require the step size to satisfy

$$\mu \leq \frac{2}{(\delta+\nu)N} \tag{3.16}$$

Then, the coefficient of the last term in (3.14) is negative. Combining with the strongly convexity property $\|\nabla J(\boldsymbol{w}_0^k) - \nabla J(w^\star)\| \geq \nu\|\widetilde{\boldsymbol{w}}_0^k\|$, we have

$$\left\|\widetilde{\boldsymbol{w}}_0^k + \mu N\nabla J(\boldsymbol{w}_0^k)\right\|^2 \leq \left(1-2\mu N\frac{\nu\delta}{\delta+\nu}\right)\|\widetilde{\boldsymbol{w}}_0^k\|^2 + \mu N\nu^2(\mu N - \frac{2}{\delta+\nu})\|\widetilde{\boldsymbol{w}}_0^k\|^2$$

$$=\Big(1-\mu\nu N)^2\|\widetilde{\boldsymbol{w}}_0^k\|^2 \tag{3.17}$$

Combining (3.13) and (4.71), we establish:

$$\|\widetilde{\boldsymbol{w}}_0^{k+1}\|^2 \leq \frac{1}{t}\left(1-\mu N\nu\right)^2\|\widetilde{\boldsymbol{w}}_0^k\|^2+\frac{\mu^2 N}{1-t}\frac{\mu^2\delta^2N^3}{1-2\mu^2\delta^2N^2}\left(2\delta^2\|\widetilde{\boldsymbol{w}}_0^k\|^2+\mathcal{K}\right) \tag{3.18}$$

We are free to choose $t\in(0,1)$. Thus, let $t=1-\mu N\nu$. Then, we conclude that

$$\begin{aligned}\|\widetilde{\boldsymbol{w}}_0^{k+1}\|^2 \leq&(1-\mu N\nu)\|\widetilde{\boldsymbol{w}}_0^k\|^2+\frac{\mu^3\delta^2N^3}{\nu(1-2\mu^2\delta^2N^2)}\left(2\delta^2\|\widetilde{\boldsymbol{w}}_0^k\|^2+\mathcal{K}\right)\\ =&\left(1-\mu N\nu+\frac{2\mu^3\delta^4N^3}{\nu(1-2\mu^2\delta^2N^2)}\right)\|\widetilde{\boldsymbol{w}}_0^k\|^2+\frac{\mu^3\delta^2N^3\mathcal{K}}{\nu(1-2\mu^2\delta^2N^2)}\end{aligned} \tag{3.19}$$

If we assume μ is sufficiently small such that

$$1-2\mu^2\delta^2N^2\geq\frac{1}{2}, \tag{3.20}$$

then inequality (3.19) becomes

$$\|\widetilde{\boldsymbol{w}}_0^{k+1}\|^2\leq\Big(1-\mu N\nu+\frac{4\mu^3\delta^4N^3}{\nu}\Big)\|\widetilde{\boldsymbol{w}}_0^k\|^2+\frac{2\mu^3\delta^2N^3}{\nu}\mathcal{K}. \tag{3.21}$$

If we further assume the step-size μ is sufficiently small such that

$$1-\mu N\nu+\frac{4\mu^3\delta^4N^3}{\nu}\leq 1-\frac{1}{2}\mu N\nu \tag{3.22}$$

then inequality (3.21) becomes

$$\begin{aligned}\|\widetilde{\boldsymbol{w}}_0^{k+1}\|^2\leq&\left(1-\frac{1}{2}\mu N\nu\right)\|\widetilde{\boldsymbol{w}}_0^k\|^2+\frac{2\mu^3\delta^2N^3}{\nu}\mathcal{K}\\ \leq&\left(1-\frac{1}{2}\mu N\nu\right)^k\|\widetilde{\boldsymbol{w}}_0^0\|^2+\left(\frac{2\mu^3\delta^2N^3}{\nu}\mathcal{K}\right)\sum_{j=1}^{k}\left(1-\frac{1}{2}\mu N\nu\right)^j\\ \leq&\left(1-\frac{1}{2}\mu N\nu\right)^k\|\widetilde{\boldsymbol{w}}_0^0\|^2+\frac{4\mu^2\delta^2N^2}{\nu^2}\mathcal{K}.\end{aligned} \tag{3.23}$$

By taking expectations with respect to the filtration, i.e. the collections of the past information, on both sides, we have

$$\mathbb{E}\|\widetilde{\boldsymbol{w}}_0^{k+1}\|^2 \leq \left(1-\frac{1}{2}\mu N\nu\right)^k \mathbb{E}\|\widetilde{\boldsymbol{w}}_0^0\|^2 + \frac{4\mu^2\delta^2N^2}{\nu^2}\mathcal{K}, \tag{3.24}$$

which implies that

$$\limsup_{k\to\infty} \mathbb{E}\|\widetilde{\boldsymbol{w}}_0^k\|^2 = O(\mu^2) \tag{3.25}$$

Finally we find a sufficient range for μ for stability. To satisfy (3.16), (3.20) and (3.22), it is enough to set μ as

$$\mu \leq \min\left\{\frac{2}{(\delta+\nu)N}, \frac{1}{2\delta N}, \frac{\nu}{\sqrt{8}\delta^2 N}\right\} < \frac{\nu}{3\delta^2 N}. \tag{3.26}$$

The argument in this derivation provides a self-contained proof for the convergence result (3.7), which generalizes the approach from [69]. There, the bound (3.7) was derived from an intermediate property (23) in [69], which does not always hold. Here, the same result is re-derived and shown to hold irrespective of this property. Consequently, we are now able to obtain Lemma 1 from [69] as a corollary to our current result, as shown next. ■

Having established the stability of the first point of every epoch, we can now establish the stability of every point.

Corollary 3.1 (Full Stability) *Under assumption 3.1, it holds that*

$$\limsup_{k\to\infty} \mathbb{E}\,\|\boldsymbol{w}_i^k - w^\star\|^2 = O(\mu^2) \tag{3.27}$$

for all i when the step-size is sufficiently small.

Proof: We have

$$\mathbb{E}\,\|\widetilde{\boldsymbol{w}}_i^k\|^2 \leq 2\mathbb{E}\,\|\boldsymbol{w}_i^k - \boldsymbol{w}_0^k\|^2 + 2\mathbb{E}\,\|\widetilde{\boldsymbol{w}}_0^k\|^2$$

$$
\begin{aligned}
&\leq 2\sum_{j=0}^{i-1} j\mathbb{E}\left\|\boldsymbol{w}_{j+1}^k - \boldsymbol{w}_j^k\right\|^2 + 2\mathbb{E}\left\|\widetilde{\boldsymbol{w}}_0^k\right\|^2 \\
&\leq 2\sum_{j=0}^{i-1} j\mathbb{E}\left\|\nabla_w Q(\boldsymbol{w}_j^k; x_{\boldsymbol{\sigma}^k(j)})\right\|^2 + 2\mathbb{E}\left\|\widetilde{\boldsymbol{w}}_0^k\right\|^2 \\
&\leq 2\mu^2\delta^2\sum_{j=0}^{i-1} j\mathbb{E}\left\|\widetilde{\boldsymbol{w}}_j^k\right\|^2 + 2\mathbb{E}\left\|\widetilde{\boldsymbol{w}}_0^k\right\|^2
\end{aligned} \tag{3.28}
$$

Summing over i;

$$
\begin{aligned}
\sum_{i=1}^{N-1}\mathbb{E}\left\|\widetilde{\boldsymbol{w}}_i^k\right\|^2 &\leq 2\mu^2\delta^2\sum_{i=1}^{N-1}\sum_{j=0}^{i-1} i\mathbb{E}\left\|\widetilde{\boldsymbol{w}}_j^k\right\|^2 + 2N\mathbb{E}\left\|\widetilde{\boldsymbol{w}}_0^k\right\|^2 \\
&= 2\mu^2\delta^2\sum_{j=0}^{N-1}\sum_{i=j+1}^{N-1} i\mathbb{E}\left\|\widetilde{\boldsymbol{w}}_j^k\right\|^2 + 2N\mathbb{E}\left\|\widetilde{\boldsymbol{w}}_0^k\right\|^2 \\
&\leq \mu^2\delta^2 N^2\sum_{j=0}^{N-1}\mathbb{E}\left\|\widetilde{\boldsymbol{w}}_j^k\right\|^2 + 2N\mathbb{E}\left\|\widetilde{\boldsymbol{w}}_0^k\right\|^2 \\
&= \mu^2\delta^2 N^2\sum_{j=1}^{N-1}\mathbb{E}\left\|\widetilde{\boldsymbol{w}}_j^k\right\|^2 + (2N+\mu^2\delta^2N^2)\mathbb{E}\left\|\widetilde{\boldsymbol{w}}_0^k\right\|^2
\end{aligned} \tag{3.29}
$$

Rearranging terms, we get

$$
\sum_{i=1}^{N-1}\mathbb{E}\left\|\widetilde{\boldsymbol{w}}_i^k\right\|^2 \leq \frac{2N+\mu^2\delta^2N^2}{1-\mu^2\delta^2N^2}\mathbb{E}\left\|\widetilde{\boldsymbol{w}}_0^k\right\|^2 \tag{3.30}
$$

Let $k\to\infty$, then

$$
\limsup_{k\to\infty}\sum_{i=1}^{N-1}\mathbb{E}\left\|\widetilde{\boldsymbol{w}}_i^k\right\|^2 = O(\mu^2) \tag{3.31}
$$

Noting that every term in the summation is non-negative, we conclude that for all j:

$$
\limsup_{k\to\infty}\mathbb{E}\left\|\widetilde{\boldsymbol{w}}_j^k\right\|^2 \leq \limsup_{k\to\infty}\sum_{i=1}^{N-1}\mathbb{E}\left\|\widetilde{\boldsymbol{w}}_i^k\right\|^2 = O(\mu^2) \tag{3.32}
$$

■

3.3 Illustrating Behavior and Periodicity

In this section we illustrate the theoretical findings so far by numerical simulations. We consider the following logistic regression problem:

$$\min_{w} \quad J(w) = \frac{1}{N}\sum_{n=1}^{N} Q(w; h_n, \gamma(n)), \tag{3.33}$$

where $h_n \in \mathbb{R}^M$ is the feature vector, $\gamma(n) \in \{\pm 1\}$ is the scalar label, and

$$Q(w; h_n, \gamma_n) \;\triangleq\; \rho\|w\|^2 + \ln\left(1 + \exp(-\gamma(n) h_n^{\mathsf{T}} w)\right). \tag{3.34}$$

The constant ρ is the regularization parameter. In the first simulation, we compare the performance of the standard stochastic gradient descent (SGD) algorithm (1.3) with replacement and the random reshuffling (RR) algorithm (1.4). We set $N = 1000$ and $M = 10$. Each h_n is generated from the normal distribution $\mathcal{N}(0; \Lambda_M)$, where Λ_M is a diagonal matrix with each diagonal entry generated from the uniform distribution $\mathcal{U}(1, 10)$. To generate $\gamma(n)$, we first generate an auxiliary random vector $w_0 \in \mathbb{R}^M$ with each entry following $\mathcal{N}(0, 1)$. Next, we generate $\boldsymbol{u}(n)$ from a uniform distribution $\mathcal{U}(0, 1)$. If $\boldsymbol{u}(n) \leq 1/(1 + \exp(-h_n^{\mathsf{T}} w_0))$ then $\gamma(n)$ is set as $+1$; otherwise $\gamma(n)$ is set as -1. We select $\rho = 0.1$ during all simulations. Figure 3.1 illustrates the MSD performance of the SGD and RR algorithms when $\mu = 0.003$. It is observed that the RR algorithm oscillates during the steady-state regime, and that the MSD at the $\boldsymbol{w}_0^k$ is the best among all iterates $\{\boldsymbol{w}_i^k\}_{i=1}^{N-1}$ during epoch k. Furthermore, it is also observed that RR has better MSD performance than SGD. Similar observations also occur in Fig. 3.2, where $\mu = 0.0003$. It is worth noting that the gap between SGD and RR is much larger in Fig. 3.2 than in Fig. 3.1.

Next, in the second simulation we verify the conclusion that the MSD for the starting point of each epoch for the random reshuffling algorithm, i.e., $\boldsymbol{w}_0^k$, can achieve $O(\mu^2)$ instead of $O(\mu)$. We still consider the regularized logistic regression problem (3.33) and (3.34), and

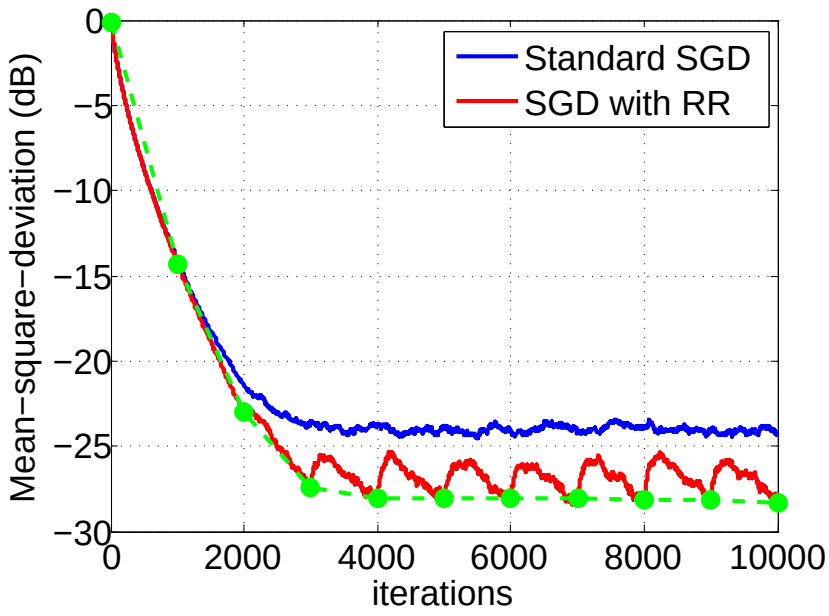

Figure 3.1: RR has better mean-square-deviation (MSD) performance than standard SGD when $\mu = 0.003$. The dotted blue curve is drawn by connecting the MSD performance at the starting points of the successive epochs.

the same experimental setting. Recall that in Theorem 3.1, we proved that

$$\limsup_{k\to\infty} \mathbb{E}\,\|\widetilde{\boldsymbol{w}}_0^k\|^2 \leq O(\mu^2), \tag{3.35}$$

which indicates that when μ is reduced a factor of 10, the MSD-performance $\mathbb{E}\,\|\widetilde{\boldsymbol{w}}_0^k\|^2$ should be improved by at least 20 dB. We observe a decay of about 20dB per decade in Fig. 3.3 for a logistic regression problem with $N = 25$ data points and 30dB per decade in Fig. 3.4 with $N = 1000$.

3.4 Introducing a Long-Term Model

We proved in the earlier sections that the mean-square error under random reshuffling approaches a small $O(\mu^2)-$neighborhood around the minimizer. Our objective now is to assess more accurately the size of the constant that multiplies μ^2 in the $O(\mu^2)$ result, and examine how this constant may depend on various parameters including the amount of data, N, and

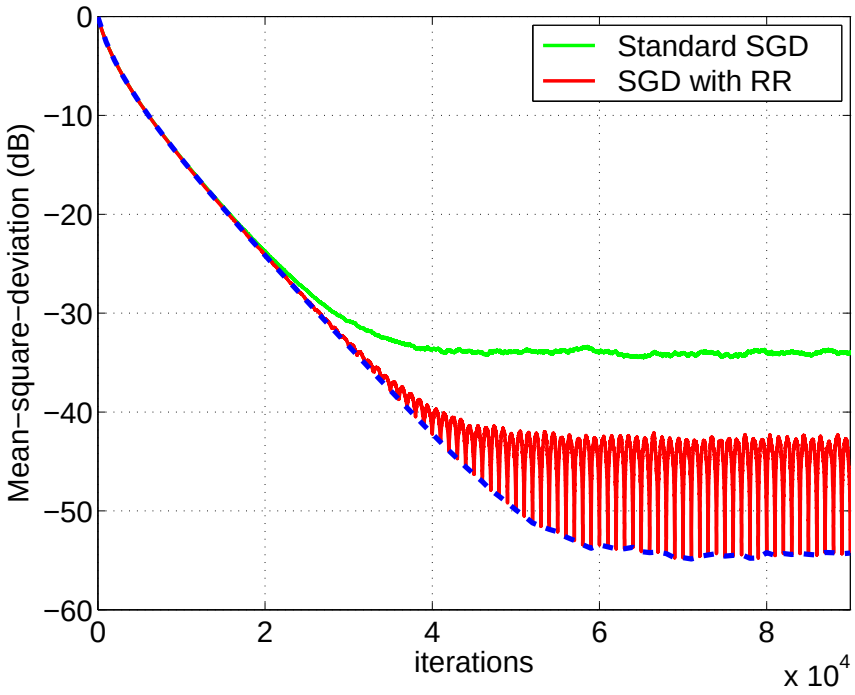

Figure 3.2: RR has much better MSD performance than standard SGD when $\mu = 0.0003$. The dotted blue curve is drawn by connecting the MSD performance at the starting points of the successive epochs.

the form of the loss function Q. To do that, we proceed in two steps. First, we introduce an auxiliary long-term model in (3.44) below and subsequently determine how far the performance of this model is from the original system described by (3.43) further ahead.

3.4.1 Error Dynamics

In order to quantify the performance of the random reshuffling implementation more accurately than the $O(\mu^2)-$figure obtained earlier, we will need to impose a condition on the smoothness of the Hessian matrix of the risk function.

Assumption 3.2 (Hessian is Lipschitz continuous) *The risk function $J(w)$ has a Lipschitz continuous Hessian matrix, i.e., there exists a constant $\kappa \geq 0$, such that*

$$\|\nabla_w^2 J(w_1) - \nabla_w^2 J(w_2)\| \leq \kappa \|w_1 - w_2\| \tag{3.36}$$

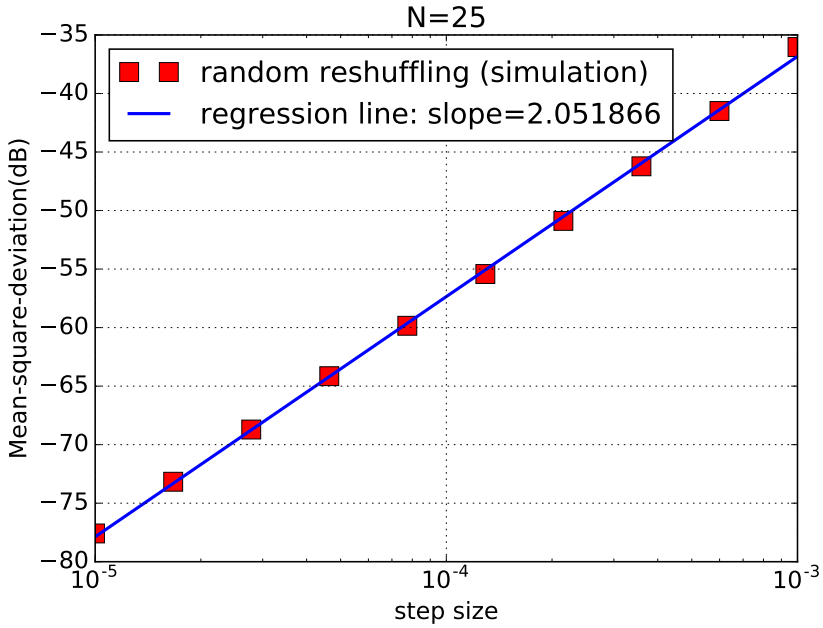

Figure 3.3: Mean-square-deviation performance at steady-state versus the step size for a logistic problem involving $N = 25$ data points. The slope is around 20 dB per decade.

■

Under this assumption, the gradient vector, $\nabla_w J(w)$, can be expressed in Taylor expansion in the form [11, p. 378]:

$$\nabla_w J(w) = \nabla_w^2 J(w^\star)(w - w^\star) + \xi(w), \quad \forall w \tag{3.37}$$

where the residual term satisfies:

$$\|\xi(w)\| \leq \frac{\kappa}{2}\|w - w^\star\|^2 \tag{3.38}$$

As such, we can rewrite algorithm (1.4) in the form:

$$\widetilde{\boldsymbol{w}}_{i+1}^k = \widetilde{\boldsymbol{w}}_i^k + \mu\nabla_w J(\boldsymbol{w}_i^k) + \mu\Big(\nabla_w Q(\boldsymbol{w}_i^k; x_{\boldsymbol{\sigma}^k(i)}) - \nabla_w J(\boldsymbol{w}_i^k)\Big)$$

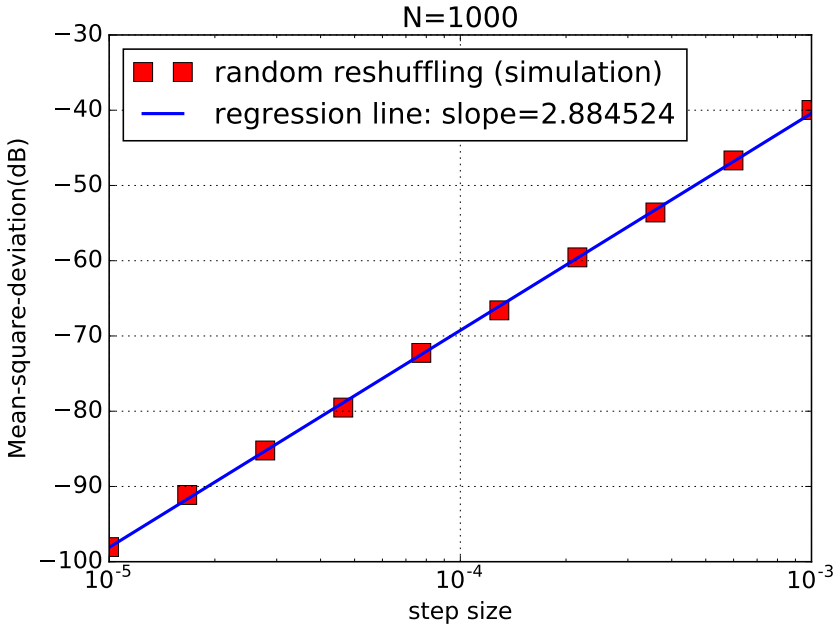

Figure 3.4: Mean-square-deviation performance at steady-state versus the step size for a logistic problem involving $N = 1000$ data points. The slope is around 30 dB per decade.

$$=\widetilde{\boldsymbol{w}}_i^k - \mu\nabla_w^2 J(\boldsymbol{w}^\star)\widetilde{\boldsymbol{w}}_i^k + \mu\xi(\boldsymbol{w}_i^k) + \mu\Big(\nabla_w Q(\boldsymbol{w}_i^k; x_{\boldsymbol{\sigma}^k(i)}) - \nabla_w J(\boldsymbol{w}_i^k)\Big) \tag{3.39}$$

To ease the notation, we introduce the Hessian matrix H and the gradient noise process:

$$\begin{aligned} H &\triangleq \nabla_w^2 J(w^\star) \\ s_{\boldsymbol{\sigma}^k(i)}(\boldsymbol{w}_i^k) &\triangleq \nabla_w Q(\boldsymbol{w}_i^k; x_{\boldsymbol{\sigma}^k(i)}) - \nabla_w J(\boldsymbol{w}_i^k) \end{aligned} \tag{3.40}$$

so that (3.39) is simplified as:

$$\widetilde{\boldsymbol{w}}_{i+1}^k = (I - \mu H)\widetilde{\boldsymbol{w}}_i^k + \mu\xi(\boldsymbol{w}_i^k) + \mu s_{\boldsymbol{\sigma}^k(i)}(\boldsymbol{w}_i^k) \tag{3.41}$$

Now property (3.2) motivates us to expand (3.41) into the following error recursion by adding and subtracting the same gradient noise term evaluated at $\boldsymbol{w}_0^k$:

$$\widetilde{\boldsymbol{w}}_{i+1}^k =(I - \mu H)\widetilde{\boldsymbol{w}}_i^k + \mu s_{\boldsymbol{\sigma}^k(i)}(\boldsymbol{w}_0^k)$$

$$+\underbrace{\mu\big(s_{\boldsymbol{\sigma}^k(i)}(\boldsymbol{w}_i^k)-s_{\boldsymbol{\sigma}^k(i)}(\boldsymbol{w}_0^k)\big)}_{\text{noise mismatch}}+\mu\xi(\boldsymbol{w}_i^k) \tag{3.42}$$

Iterating (3.42) and using (1.5) we can establish the following useful relation, which we call upon in the sequel:

$$\begin{aligned}\widetilde{\boldsymbol{w}}_0^{k+1} = \; & (I-\mu H)^N\widetilde{\boldsymbol{w}}_0^k+\mu\sum_{i=0}^{N-1}(I-\mu H)^{N-i-1}s_{\boldsymbol{\sigma}^k(i)}(\boldsymbol{w}_0^k)\\ & +\mu\sum_{i=0}^{N-1}(I-\mu H)^{N-i-1}\left(s_{\boldsymbol{\sigma}^k(i)}(\boldsymbol{w}_i^k)-s_{\boldsymbol{\sigma}^k(i)}(\boldsymbol{w}_0^k)\right)\\ & +\mu\sum_{i=0}^{N-1}(I-\mu H)^{N-i}\xi(\boldsymbol{w}_i^k)\end{aligned} \tag{3.43}$$

Note that recursion (3.43) relates $\widetilde{\boldsymbol{w}}_0^k$ to $\widetilde{\boldsymbol{w}}_0^{k+1}$, which are the starting points of two successive epochs. In this way, we have now transformed recursion (1.4), which runs from one sample to another within the same epoch, into a relation that runs from one starting point to another over two successive epochs.

To proceed, we will ignore the last two terms in (3.43) and consider the following approximate model, which we shall refer to as a *long-term* model.

$$\widetilde{\boldsymbol{w}}_0'^{k+1}=(I-\mu H)^N\widetilde{\boldsymbol{w}}_0'^{k}-\underbrace{\mu\sum_{i=0}^{N-1}(I-\mu H)^{N-i-1}s_{\boldsymbol{\sigma}^k(i)}(\boldsymbol{w}_0^k)}_{\triangleq\, s'(\boldsymbol{w}_0^k)} \tag{3.44}$$

Obviously, the state evolution will be different than (3.43) and is therefore denoted by the prime notation, $\widetilde{\boldsymbol{w}}_0'^k$. Observe, however, that in model (3.44) the gradient noise process is still being evaluated at the original state vector, $\boldsymbol{w}_0^k$, and not at the new state vector, $\boldsymbol{w}_0'^k$.

3.4.2 Performance of the Long-Term Model across Epochs

Note that the gradient noise $\boldsymbol{s}'(\boldsymbol{w}_0^k)$ in (3.44) has the form of a weighted sum over one epoch. This noise clearly satisfies the property:

$$\mathbb{E}\,[\,s'(\boldsymbol{w}_0^k)\,|\,\boldsymbol{w}_0^k\,] = 0 \tag{3.45}$$

We also know that $s'(\boldsymbol{w}_0^k)$ satisfies the Markov property, i.e., it is independent of all previous $\boldsymbol{w}_i^{k'}$ and $\boldsymbol{\sigma}^{k'}(\cdot)$, where $k' < k$, conditioned on $\boldsymbol{w}_0^k$. To motivate the next lemma consider the following auxiliary setting.

Assume we have a collection of N vectors $\{x_i\}$ in $\mathbb{R}^2$ whose sum is zero. We define a random walk over these vectors in the following manner. At each time instant, we select a random vector $x_{\boldsymbol{n}_i}$ uniformly and with replacement from this set and move from the current location along the vector $x_{\boldsymbol{n}_i}$ to the next location. If we keep repeating this construction, we obtain behavior that is represented by the right plot in Fig. 5. Assume instead that we repeat the same experiment except that now we assume the data $\{x_i\}$ is first reshuffled and then vectors $x_{\boldsymbol{\sigma}(i)}$ are selected uniformly without replacement. Because of the zero sum property, and because sampling is now performed without replacement, we find that in this second implementation we always return to the origin after N selections. This situation is illustrated in the left plot of the same Fig. 3.5. The next lemma considers this scenario and provides useful expressions that allow us to estimate the expected location after $1, 2$ or more (unitl $N-1$) movements. These results will be used in the sequel in our analysis of the performance of stochastic learning under RR.

Lemma 3.1 *Suppose we have a set of N vectors $X = \{x_i\}_{i=1}^N$ with the constraint $\sum_{i=1}^N x_i = 0$. Assume the elements of X are randomly reshuffled and then selected uniformly without replacement. Let β be any nonnegative constant, B be any symmetric positive semi-definite matrix, and introduce*

$$R_x \;\triangleq\; \frac{1}{N}\sum_{i=1}^{N} x_i x_i^{\mathsf{T}} \tag{3.46}$$

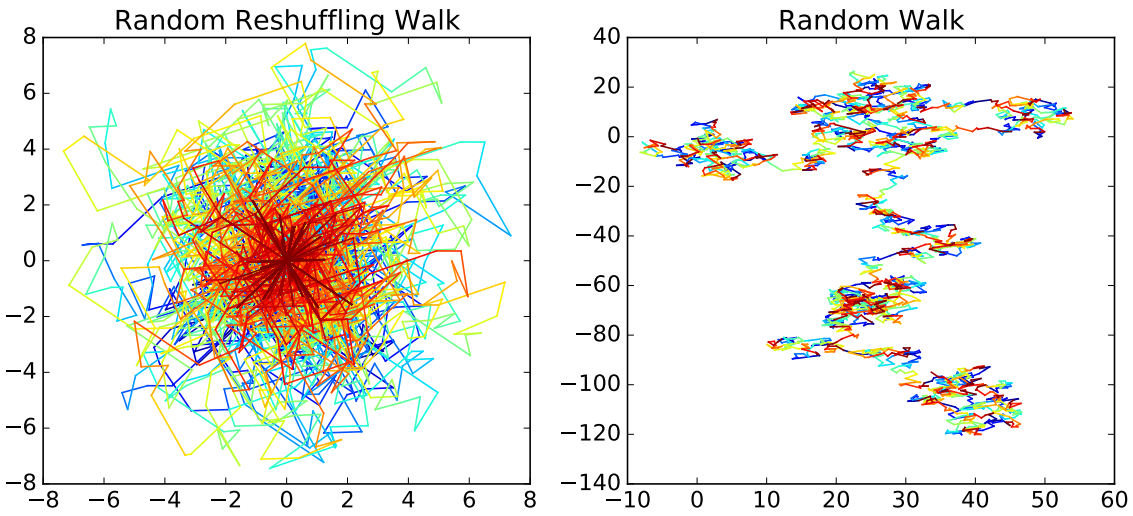

Figure 3.5: Random walk versus Random reshuffling walk. The lines with same color represent all i-th choices walk in different epochs.

$$Var(X) \triangleq \frac{1}{N}\sum_{i=1}^{N}\|x_i\|^2 = Tr(R_x) \tag{3.47}$$

Define the following functions for any $1 \leq n \leq N$*:*

$$f(n;X,\beta) \triangleq \mathbb{E}\left\|\sum_{j=1}^{n}\beta^{n-j}x_{\boldsymbol{\sigma}(j)}\right\|^2 \tag{3.48}$$

$$F(n;X,B) \triangleq \mathbb{E}\left[\sum_{j=1}^{n}B^{n-j}x_{\boldsymbol{\sigma}(j)}\right]\left[\sum_{j=1}^{n}x_{\boldsymbol{\sigma}(j)}^{\mathsf{T}}B^{n-j}\right] \tag{3.49}$$

It then holds that

$$f(n;X,\beta) = \frac{(\sum_{i=0}^{n-1}\beta^{2i})N - (\sum_{i=0}^{n-1}\beta^{i})^2}{N-1}\,Var(X) \tag{3.50}$$

$$F(n;X,B) = \frac{\left[\sum_{i=0}^{n-1}B^iR_xB^i\right]N - \left[\sum_{i=0}^{n-1}B^i\right]R_x\left[\sum_{i=0}^{n-1}B^i\right]}{N-1} \tag{3.51}$$

Proof: The proof is provided in Appendix 3.B. ■

We now return to the stochastic gradient implementation under random reshuffling. Re-

call from (3.3) that the stochastic gradient satisfies the zero sample mean property so that

$$\sum_{i=1}^{N} s_{\boldsymbol{\sigma}^k(i)}(\boldsymbol{w}) = 0 \tag{3.52}$$

at any given point $\boldsymbol{w}$. Applying Lemma 3.1, we readily conclude that

$$\begin{aligned}\mathbb{E}\,[s'(\boldsymbol{w}_0^k)s'(\boldsymbol{w}_0^k)^{\mathsf{T}} \,|\, \boldsymbol{w}_0^k] = &\frac{N\left(\sum_{i=0}^{N-1}(I-\mu H)^i R_s^k (I-\mu H)^i\right)}{N-1} \\ &- \frac{\left[\sum_{i=0}^{N-1}(I-\mu H)^i\right] R_s^k \left[\sum_{i=0}^{N-1}(I-\mu H)^i\right]}{N-1}\end{aligned} \tag{3.53}$$

where

$$R_s^k \;\triangleq\; \frac{1}{N}\sum_{n=1}^{N} \boldsymbol{s}_n(\boldsymbol{w}_0^k)\boldsymbol{s}_n(\boldsymbol{w}_0^k)^{\mathsf{T}} \tag{3.54}$$

Similarly, we conclude for the gradient noise at the optimal $w^\star$:

$$\begin{aligned}R_s'^\star \;&\triangleq\; \mathbb{E}\,[s'(\boldsymbol{w}^\star)s'(\boldsymbol{w}^\star)^{\mathsf{T}}] \\ &= \frac{N\left(\sum_{i=0}^{N-1}(I-\mu H)^i R_s^\star (I-\mu H)^i\right)}{N-1} - \frac{\left[\sum_{i=0}^{N-1}(I-\mu H)^i\right] R_s^\star \left[\sum_{i=0}^{N-1}(I-\mu H)^i\right]}{N-1}\end{aligned} \tag{3.55}$$

where

$$R_s^\star = \frac{1}{N}\sum_{i=0}^{N} \nabla Q(w^\star; x_i)\nabla Q(w^\star; x_i)^{\mathsf{T}} \tag{3.56}$$

Theorem 3.2 (Performance of Long-term Model) *Under assumptions 3.1 and 3.2, when the step size μ is small enough, the mean-square-deviation (MSD) of the long term model (3.44) is given by*

$$\begin{aligned}MSD_{RR}^{lt} \;&\triangleq\; \limsup_{k\to\infty} \|\boldsymbol{w}_0'^k - w^\star\|^2 \\ &= \mu^2\, Tr\left((I-(I-\mu H)^{2N})^{-1} R_s'^\star\right) + O(\mu^4)\end{aligned} \tag{3.57}$$

Proof: See Appendix 3.C. ■

The simulations in Fig. 3.6 show that the MSD expression (3.57) fits well the performance of the original random reshuffling algorithm. We will establish this fact analytically in the sequel. For now, the simulation is simply confirming that the performance of the long-term model is a good indication of the performance of the original stochastic gradient implementation under RR.

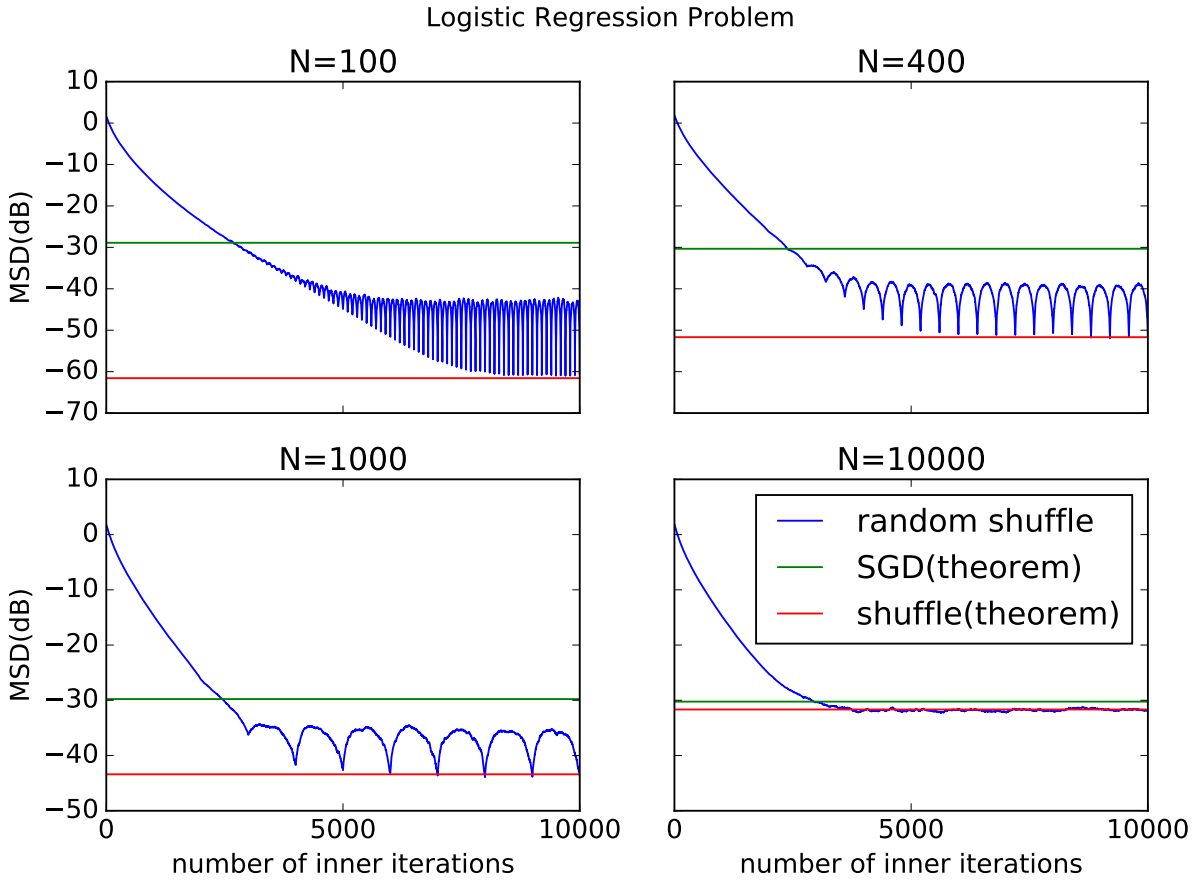

Figure 3.6: Mean-square-deviation perfromance of random reshuffling algorithm curve on least-mean-square cost function

3.4.3 Performance of the Long-Term Model over Iterations

In the previous section we examined the performance of the long-term model at the starting points of successive epochs. In this section, we examine the performance of the same model

at any iterate $\boldsymbol{w}_i^k$ as time approaches ∞. This analysis will help explain the oscillations that are observed in the learning curves in the simulations. First, similar to (3.55), we need to determine the covariance matrix $R'^{\star}_{s,i}$ for any i. From Lemma 3.1, we immediately get that

$$\begin{aligned} R'^{\star}_{s,i} &\triangleq \mathbb{E}\, s'_i(w^\star) s'_i(w^\star)^{\mathsf{T}} \\ &= \frac{N\left(\sum_{j=0}^{i-1}(I-\mu H)^j R_s^\star (I-\mu H)^j\right)}{N-1} \\ &\quad - \frac{\left[\sum_{j=0}^{i-1}(I-\mu H)^j\right] R_s^\star \left[\sum_{j=0}^{i-1}(I-\mu H)^j\right]}{N-1} \end{aligned} \tag{3.58}$$

Theorem 3.3 (Performance Upper-bound for Long- Term Model) *Under assumptions 3.1 and 3.2, when the step size μ satisfies $\mu \leq \frac{2}{\delta+\nu}$, the upper-bound of mean-square-deviation (MSD) of the long term model (3.44) at all iterations is given by*

$$\begin{aligned} &\lim_{k\to\infty} \mathbb{E}\,\|\widetilde{\boldsymbol{w}}_i'^k\|^2 \\ &\leq (1-\mu\nu)^{2i}\mu^2\, Tr\left(\left(I-(I-\mu H)^{2N}\right)^{-1} R'^{\star}_s\right) \\ &\quad + \left(1-(1-\mu\nu)^{2i}\right)\mu^2\, Tr\left(\left(I-(I-\mu\nu)^{2i}\right)^{-1} R'^{\star}_{s,i}\right) \qquad (3.59) \\ &\triangleq \eta_i MSD^{lt}_{RR} + (1-\eta_i) MSD^{lt}_{RR,i} \qquad (3.60) \end{aligned}$$

Proof:— See Appendix 3.D. ■

We need to point out unlike that (3.57), expression (3.131) is an upper-bound rather than an actual performance expression. Still, this bound can help provide useful insights on the periodic behavior that is observed in the simulations. The expression (3.59) on the right-hand side is a convex combination of two performance measures as defined in (3.60), where the second term is always larger than the first term but approaching it as i increases towards N. This behavior will become clearer later in the context of an example and the hyperbolic representation in section 3.5.2.

Before we continue, we would like to comment on the convergence curve under random

reshuffling. Unlike the convergence curve under uniform sampling, we observe periodic fluctuations under random reshuffling in Figures 3.2 and 3.6. The main reason for this behavior is the fact that the gradient noise is no longer i.i.d. in steady-state. Specifically, the noise variance is now a function of the iterate and it assumes its lowest value at the beginning and end of every epoch. In lemma 3.1, we show that the variance of the random walk process resulting from random reshuffling at each iteration n in Eq. (3.50). We plot the function for $N = 20$ and $\mathrm{V}ar(x) = 1$ in Fig. 3.7. Since the mean-square performance of the algorithm is related to the variance of the gradient noise, it is expected that this bell-shape behavior will be reflected in to the MSD curve as well, thus, resulting in better performance at the beginning and end of every epoch.

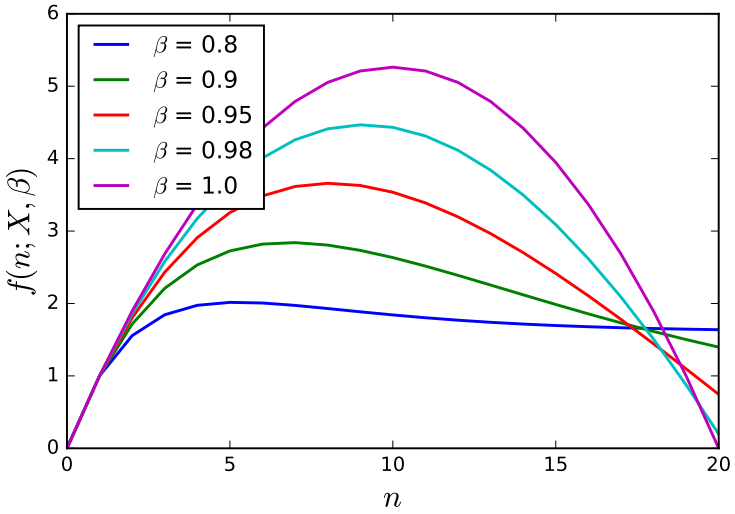

Figure 3.7: The variance function $f(n; X, \beta)$ at (3.50) versus n with different β value.

3.4.4 Mismatch Bound

Now we provide an upper bound on the mismatch between the long-term model (3.44) and the original algorithm (1.4).

Lemma 3.2 (Mismatch Bound) *After long enough iterations, i.e., $k \gg 1$, the difference between the long term model trajectory (3.44) and the original trajectory (1.4) is*

$$\limsup_{k\to\infty} \mathbb{E}\,\|\widetilde{\boldsymbol{w}}_0'^k - \widetilde{\boldsymbol{w}}_0^k\|^2 \leq \frac{4\mu^2\delta^2 N^2}{\nu^2(N-1)}\mathcal{K} + O(\mu^3) \tag{3.61}$$

Proof: See Appendix 3.F. ■

3.5 Quadratic Risks and Hyperbolic Representation

Lastly, we consider an example involving a quadratic (least-squares) risk to show that, in this case, the long-term model provides the exact MSD for the original algorithm. The analysis will also provide some insights into expression (3.57). It also motivates a hyperbolic representation for the MSD, which helps provides some more insights into the MSD behavior.

3.5.1 Quadratic Risks

Thus, consider the following quadratic risk function:

$$\min_{w} J(w) = \frac{1}{2N}\sum_{n=1}^{N} \|Aw - x_n\|^2 \tag{3.62}$$

where A has full column rank. We have:

$$\nabla_w J(w) = A^{\mathsf{T}}Aw - A^{\mathsf{T}}\underbrace{\left(\frac{1}{N}\sum_{n=1}^{N} x_n\right)}_{\triangleq\, \bar{x}} \tag{3.63}$$

$$\nabla_w Q(w; x_n) = A^{\mathsf{T}}Aw - A^{\mathsf{T}}x_n \tag{3.64}$$

$$\nabla^2 J(w_i^k) = A^{\mathsf{T}}A \tag{3.65}$$

$$s_n(w) = A^{\mathsf{T}}(x_n - \bar{x}) \tag{3.66}$$

Since the gradient noise $s_n(\boldsymbol{w})$ is independent of $\boldsymbol{w}$, we have

$$s_n(\boldsymbol{w}_i^k) - s_n(\boldsymbol{w}_0^k) \equiv 0 \tag{3.67}$$

Moreover, since the risk is quadratic, it also holds that

$$\xi(w) \equiv 0 \tag{3.68}$$

Table 3.1: Summary of the Results with Random Reshuffling versus Uniform Sampling with Replacement.

	Uniform sampling [13]	Random Reshuffling (Long-term or Quadratic)	Random Reshuffling
Steady-state	$O(\mu)$	$O(\mu^3)$ — Eq. (3.94)	$O(\mu^2)$ — Eq. (3.7)
$\text{MSD}^{\text{epoch}}$	$\frac{\mu}{2}\text{Tr}(H^{-1}R_s)$	$\mu^2\text{Tr}\Big((I-(I-\mu H)^{2N})^{-1}R_s'^{\star}\Big)$ — Eq. (3.57)	$(3.57)+O(\mu^2)$ — Eq.(3.61)
$\text{MSD}^{\text{epoch}}_{\text{(Hyperbolic)}}$	$\frac{\mu}{2}\text{Tr}\big(\Lambda^{-1}U^{\mathsf{T}}R_s^{\star}U\big)$	$\frac{\mu}{2}\text{Tr}\Bigg(M_{\text{R}R}\Lambda^{-1}U^{\mathsf{T}}R_s^{\star}U\Bigg)$ — Eq. (3.91)	$(3.91)+O(\mu^2)$—Eq.(3.61)
$\text{MSD}^{\text{iteraion}}$	$\frac{\mu}{2}\text{Tr}(H^{-1}R_s)$	$(1-\mu\nu)^{2i}\mu^2\text{Tr}\Big(\big(I-(I-\mu H)^{2N}\big)^{-1}R_s'^{\star}\Big)$ $+\Big(1-(1-\mu\nu)^{2i}\Big)\mu^2\text{Tr}\Big(\big(I-(I-\mu\nu)^{2i}\big)^{-1}R_{s,i}'^{\star}\Big)$ — Eq. (3.59)	$(3.59)+O(\mu^2)$ — Eq.(3.61)
$\text{MSD}^{\text{iteraion}}_{\text{(Hyperbolic)}}$	$\frac{\mu}{2}\text{Tr}\big(\Lambda^{-1}U^{\mathsf{T}}R_s^{\star}U\big)$	$e^{-2\mu i}\frac{\mu}{2}\text{Tr}\Big(M_{RR}\Lambda^{-1}U^{\mathsf{T}}R_s^{\star}U\Big)$ $+(1-e^{-2\mu i})\frac{\mu}{2}\text{Tr}\Big(M_{RR}(i)\Lambda^{-1}U^{\mathsf{T}}R_s^{\star}U\Big)$ — Eq. (3.95)	$(3.95)+O(\mu^2)$ — Eq.(3.61)
Infinite data	$\frac{\mu}{2}\text{Tr}\big(H^{-1}R_s\big)$	$\frac{\mu}{2}\text{Tr}(H^{-1}R_s)$— Eq. (3.100)	$\frac{\mu}{2}\text{Tr}(H^{-1}R_s)$
Periodic	No	Yes — Eq. (3.59)	Yes

Therefore, the long-term model is exactly the same as the original algorithm. For this example, we can calculate the following quantities:

$$w^\star = (A^\mathsf{T}A)^{-1}A^\mathsf{T}\bar{x} \tag{3.69}$$

$$R_s^\star = A^\mathsf{T}\frac{1}{N}\sum_{n=1}^{N}(x_n-\bar{x})(x_n-\bar{x})^\mathsf{T}A = A^\mathsf{T}R_{xx}A \tag{3.70}$$

$$\mathrm{V}ar(x) = \frac{1}{N}\sum_{n=1}^{N}\|x_n-\bar{x}\|^2 \tag{3.71}$$

$$I-\mu H = I-\mu A^\mathsf{T}A \tag{3.72}$$

In special case when the columns of A are orthogonal and normalized, i.e., $A^\mathsf{T}A = I$, we can simplify the MSD expression (3.57) by noting that

$$\begin{aligned} R_s'^\star &= \frac{1}{N-1}\left(N\sum_{i=0}^{N-1}(1-\mu)^{2i} - \Big(\sum_{i=0}^{N-1}(1-\mu)^i\Big)^2\right)A^\mathsf{T}R_{xx}A \\ &= \frac{1}{N-1}\left(\frac{N(1-(1-\mu)^{2N})}{1-(1-\mu)^2} - \left(\frac{1-(1-\mu)^N}{1-(1-\mu)}\right)^2\right)A^\mathsf{T}R_{xx}A \\ &= \frac{1}{N-1}\left(\frac{N(1-(1-\mu)^{2N})}{2\mu-\mu^2} - \frac{\big(1-(1-\mu)^N\big)^2}{\mu^2}\right)A^\mathsf{T}R_{xx}A \end{aligned} \tag{3.73}$$

and, hence,

$$\begin{aligned} \mathrm{M}SD_{\mathrm{R}R} &= \mu^2\mathrm{Tr}\left((1-(1-\mu)^{2N})^{-1}R_s'^\star\right) \\ &= \frac{\mu^2}{N-1}\left(\frac{N}{2\mu-\mu^2} - \frac{(1-(1-\mu)^N)^2}{\mu^2(1-(1-\mu)^{2N})}\right)\mathrm{V}ar(x) \\ &= \frac{\mu^2}{N-1}\left(\frac{N}{2\mu-\mu^2} - \frac{1-(1-\mu)^N}{\mu^2(1+(1-\mu)^N)}\right)\mathrm{V}ar(x) \end{aligned} \tag{3.74}$$

In order to provide further insights on this MSD expression, we simplify it under a small μ assumption. We could introduce the Taylor series:

$$(1-\mu)^N = 1-N\mu+O(N^2\mu^2) \tag{3.75}$$

However, this approximation can be bad if N is large, which is not uncommon in big data. Instead, we appeal to:

$$(1-\mu)^N = e^{N\ln(1-\mu)} = e^{-\mu N + O(\mu^2 N)} \approx e^{-\mu N} \tag{3.76}$$

Notice it is $O(\mu^2 N)$ instead of $O(\mu^2 N^2)$, and therefore (3.76) is a tighter approximation than (3.75) when N is large. Based on this, we further approximate:

$$\frac{1-(1-\mu)^N}{1+(1-\mu)^N} \approx \tanh(\mu N/2) \tag{3.77}$$

and arrive at the simplified expression:

$$\begin{aligned} \mathrm{MSD}_{\mathrm{R}R} &\approx \frac{\mu}{N-1}\left(\frac{N}{2} - \frac{\tanh(\frac{\mu N}{2})}{\mu}\right)\mathrm{V}ar(x) \\ &= \frac{\mu}{2}\frac{N}{N-1}\left(1 - \frac{2}{\mu N}\tanh\left(\frac{\mu N}{2}\right)\right)\mathrm{V}ar(x) \end{aligned} \tag{3.78}$$

For comparison purposes, we know that a simplified expression for MSD under uniform sampling has the following expression [13]:

$$\mathrm{MSD}_{\mathrm{u}s} = \frac{\mu}{2}\mathrm{V}ar(x) \tag{3.79}$$

Hence, the random reshuffling case has an extra multiplicative factor:

$$m_{\mathrm{R}R} \triangleq \frac{N}{N-1}\left(1 - \frac{2}{\mu N}\tanh\left(\frac{\mu N}{2}\right)\right) \tag{3.80}$$

We plot $m_{\mathrm{R}R}$ versus μN in the left plot of Fig. 3.8 where we ignore $\frac{N}{N-1}$. Now it is clear from the figure that the smaller the step size μ or the smaller sample size N are, the larger the improvement in performance is. In contrast, when μN goes to infinity, the term $m_{\mathrm{R}R}$ will converge to 1, i.e., the same performance as uniform sampling situation, which is consistent with the infinite-horizon case.

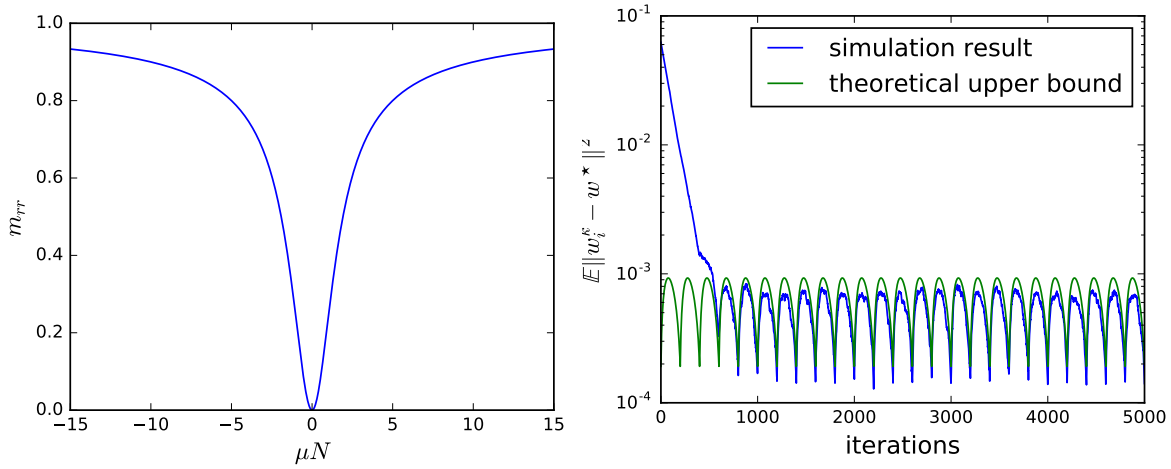

Figure 3.8: Left: The curve of m_{RR} versus μN. Right: Mean-square-deviation performance of random reshuffling for a quadratic risk.

Lastly, noting that

$$\begin{aligned} R_{s,i}'^{\star} &= \frac{1}{N-1}\left(N\sum_{j=0}^{i-1}(1-\mu)^{2j} - \Big(\sum_{j=0}^{i-1}(1-\mu)^{j}\Big)^2\right) A^{\mathsf{T}}R_{xx}A \\ &= \frac{1}{N-1}\left(\frac{N(1-(1-\mu)^{2i})}{1-(1-\mu)^2} - \left(\frac{1-(1-\mu)^i}{1-(1-\mu)}\right)^2\right) A^{\mathsf{T}}R_{xx}A \\ &= \frac{1}{N-1}\left(\frac{N(1-(1-\mu)^{2i})}{2\mu-\mu^2} - \frac{(1-(1-\mu)^i)^2}{\mu^2}\right) A^{\mathsf{T}}R_{xx}A \end{aligned} \tag{3.81}$$

and using the approximation (3.76):

$$R_{s,i}'^{\star} \approx \frac{N}{N-1}\left(\frac{1-e^{-2\mu i}}{2\mu} - \frac{(1-e^{-\mu i})^2}{\mu^2 N}\right) A^{\mathsf{T}}R_{xx}A \tag{3.82}$$

in (3.59) we get for $i \in [1, N]$:

$$\begin{aligned} \lim_{k\to\infty}\mathbb{E}\|\widetilde{\boldsymbol{w}}_i'^{k}\|^2 \approx\; & e^{-2\mu i}\frac{\mu}{2}\frac{N}{N-1}\left(1-\frac{2}{\mu N}\tanh\left(\frac{\mu N}{2}\right)\right)\mathrm{V}ar(x) \\ & + (1-e^{-2\mu i})\frac{\mu}{2}\underbrace{\frac{N}{N-1}\left(1-\frac{2}{\mu N}\tanh\left(\frac{\mu i}{2}\right)\right)}_{\triangleq\, m_{\mathrm{RR}}(i)}\mathrm{V}ar(x) \end{aligned} \tag{3.83}$$

Since $\tanh(\cdot)$ is monotonically increasing, $m_{\mathrm{RR}}(i) \geq m_{\mathrm{RR}}$. With i increasing, the convex

combination gives more weight to the second term, which is larger than the first term. This explains the increasing of MSD at the first half of the cycle. With i increasing further, $m_{\mathrm{R}R}(i)$ will decrease to the same level as $m_{\mathrm{R}R}$. Hence, MSD at the second half of the cycle will decrease again. The simulation result shows in the right plot of Fig. 3.8 fits with the theoretical analysis for quadratic risks rather well.

3.5.2 Hyperbolic Representation for the MSD

Motivated by the result for the quadratic risk case, we now derive a similar expression for the MSD more generally also in terms of a tanh function. First, we extend result (3.76) into a matrix version. Supposing Λ is a positive diagonal matrix and μ is sufficiently small such that $I-\mu\Lambda$ is a stable matrix, we have

$$(I-\mu\Lambda)^N \approx e^{-\mu N\Lambda} \tag{3.84}$$

and

$$\begin{aligned}\sum_{i=0}^{N-1}(I-\mu\Lambda)^i =& \frac{1}{\mu}(I-\mu\Lambda)^N\Lambda^{-1}\\ \approx& \frac{1}{\mu}e^{-\mu N\Lambda}\Lambda^{-1}\end{aligned} \tag{3.85}$$

It follows that

$$\begin{aligned}&\mathrm{Tr}\left((I-(I-\mu H)^{2N})^{-1}\left(\sum_{i=0}^{N-1}(I-\mu H)^i R_s^\star (I-\mu H)^i\right)\right)\\ \overset{(3.122)}{=}&\ \mathrm{Tr}\left(\sum_{i=0}^{N-1}\sum_{k=0}^{\infty}(I-\mu H)^i(I-\mu H)^{2kN}(I-\mu H)^i R_s^\star\right)\\ =&\ \mathrm{Tr}\left(\sum_{k=0}^{\infty}\sum_{i=0}^{N-1}(I-\mu H)^{2(kN+i)}R_s^\star\right)\\ \overset{(a)}{=}&\ \mathrm{Tr}\left(\sum_{j=0}^{\infty}(I-\mu H)^{2j}R_s^\star\right)\\ =&\ \mathrm{Tr}\left((I-(I-\mu H)^2)^{-1}R_s^\star\right)\end{aligned}$$

$$\approx \frac{1}{2\mu}\mathrm{Tr}(H^{-1}R_s^\star) = \frac{1}{2\mu}\mathrm{Tr}(\Lambda^{-1}U^\mathsf{T}R_s^\star U) \tag{3.86}$$

where in step (a) we used the fact that $kN+i$ is the N-modular representation of all integer numbers. To shorten the notation, we let:

$$\tau \triangleq \mu N \tag{3.87}$$

Next, for the second part of (3.57):

$$\begin{aligned}
&\mathrm{Tr}\left((I-(I-\mu H)^{2N})^{-1}\Big[\sum_{i=0}^{N-1}(I-\mu H)^i\Big]R_s^\star\Big[\sum_{i=0}^{N-1}(I-\mu H)^i\Big]\right)\\
&\overset{(a)}{=} \frac{1}{\mu^2}\mathrm{Tr}\Big((I-e^{-2\tau\Lambda})^{-1}(I-e^{-\tau\Lambda})\Lambda^{-1}U^\mathsf{T}R_s^\star U\Lambda^{-1}(I-e^{-\tau\Lambda})\Big)\\
&= \frac{1}{\mu^2}\mathrm{Tr}\Big(\Lambda^{-1}(I-e^{-\tau\Lambda})(I-\mu e^{-2\tau\Lambda})^{-1}(I-e^{-\tau\Lambda})\Lambda^{-1}U^\mathsf{T}R_s^\star U\Big)\\
&\overset{(b)}{=} \frac{1}{\mu^2}\mathrm{Tr}\Big(\Lambda^{-1}(I+e^{-\tau\Lambda})^{-1}(I-e^{-\tau\Lambda})\Lambda^{-1}U^\mathsf{T}R_s^\star U\Big)\\
&= \frac{1}{\mu^2}\mathrm{Tr}\Big(\Lambda^{-1}\tanh(\tau\Lambda/2)\Lambda^{-1}UR_s^\star U^\mathsf{T}\Big)\\
&= \frac{N}{2\mu}\mathrm{Tr}\Big(2\tau^{-1}\Lambda^{-1}\tanh(\tau\Lambda/2)\Lambda^{-1}UR_s^\star U^\mathsf{T}\Big)
\end{aligned} \tag{3.88}$$

where step (a) replaces H by its eigendecomposition and uses (3.85), while step (b) exploits the fact that

$$I-e^{-2\tau\Lambda} = (I+e^{-\tau\Lambda})(I-e^{-\tau\Lambda}) \tag{3.89}$$

Moreover, the tanh notation refers to

$$\tanh\Lambda = \mathrm{diag}\{\tanh(\Lambda_{1,1}),\cdots,\tanh(\Lambda_{M,M})\} \tag{3.90}$$

Combining the above two results gives

$$\mathrm{MSD}_{\mathrm{RR}}^{lt} = \frac{\mu}{2}\mathrm{Tr}\Bigg(\underbrace{\frac{N}{N-1}\Big[I - \frac{2}{\mu N}\Lambda^{-1}\tanh\Big(\frac{\mu N}{2}\Lambda\Big)\Big]}_{\triangleq M_{RR}}\Lambda^{-1}U^{\mathsf{T}}R_s^{\star}U\Bigg) \tag{3.91}$$

Compared with the uniform sampling case:

$$\mathrm{MSD}_{\mathrm{US}} = \frac{\mu}{2}\mathrm{Tr}\big(H^{-1}R_s^{\star}\big) = \frac{\mu}{2}\mathrm{Tr}\big(\Lambda^{-1}U^{\mathsf{T}}R_s^{\star}U\big) \tag{3.92}$$

Now, it is clear that the diagonal matrix factor M_{RR} serves the same purpose as m_{RR}. Each entry of this factor matrix captures the improvement of random reshuffling over uniform sampling. Lastly, we focus on the order of expression (3.92). We know from the Taylor's expansion that

$$1 - \frac{1}{x}\tanh(x) = O(x^2) \tag{3.93}$$

We conclude that

$$M_{\mathrm{RR}} = O(\mu^2 N^2) \implies \mathrm{MSD}_{\mathrm{RR}}^{lt} = O(\mu^3) \tag{3.94}$$

that confirms the observation of $O(\mu^3)$ in the Fig. 3.2.

Lastly, similar to the derivation for the quadratic case (3.81)–(3.83), we can establish the hyperbolic representation of MSD for general case at all iterations:

$$\begin{aligned}\lim_{k\to\infty}\mathbb{E}\,\|\widetilde{\boldsymbol{w}}_i'^{k}\|^2 \approx & e^{-2\mu i}\frac{\mu}{2}\mathrm{Tr}\Bigg(\frac{N}{N-1}\Big[I - \frac{2}{\mu N}\Lambda^{-1}\tanh\Big(\frac{\mu N}{2}\Lambda\Big)\Big]\Lambda^{-1}U^{\mathsf{T}}R_s^{\star}U\Bigg) \\ & + (1-e^{-2\mu i})\frac{\mu}{2}\mathrm{Tr}\Bigg(\underbrace{\frac{N}{N-1}\Big[I - \frac{2}{\mu N}\Lambda^{-1}\tanh\Big(\frac{\mu i}{2}\Lambda\Big)\Big]}_{\triangleq M_{RR}(i)}\Lambda^{-1}U^{\mathsf{T}}R_s^{\star}U\Bigg)\end{aligned} \tag{3.95}$$

3.5.3 Infinite-Horizon Case

In this work, we are mostly interested in the finite-data case, where the data size is N. The results so far are based on this assumption. However, it is inspiring though to see how the performance result would simplify if we allow N to grow to infinity. In that case, we get

$$\begin{aligned}\lim_{N\to\infty} \mathrm{MSD}_{\mathrm{R}R}^{\mathrm{lt}} &= \mu^2 \lim_{N\to\infty} \mathrm{Tr}\left((I-(I-\mu H)^{2N})^{-1} R_s'^{\star}\right) \\ &= \mu^2 \lim_{N\to\infty} \mathrm{Tr}\left(R_s'^{\star}\right) \end{aligned} \tag{3.96}$$

since for sufficiently small μ, the matrix $I-\mu H$ is stable. Moreover, observe further that:

$$\begin{aligned}\lim_{N\to\infty} \sum_{i=0}^{N-1} (I-\mu H)^{2i} &= \Big(I-(I-\mu H)^2\Big)^{-1} \\ &= \frac{1}{2\mu} H^{-1}\big(I-\mu H/2\big)^{-1} \\ &= \frac{1}{2\mu} H^{-1} + O(1) \end{aligned} \tag{3.97}$$

where $O(1)$ represents a matrix where all entries are $O(1)$. Hence,

$$\begin{aligned}\lim_{N\to\infty} \mathrm{Tr}\left(R_s'^{\star}\right) &= \mathrm{Tr}\left(\lim_{N\to\infty} R_s'^{\star}\right) \\ &= \mathrm{Tr}\left(\lim_{N\to\infty} \sum_{i=0}^{N} (I-\mu H)^i R_s^{\star} (I-\mu H)^i\right) \\ &= \mathrm{Tr}\left(\lim_{N\to\infty} \sum_{i=0}^{N} (I-\mu H)^{2i} R_s^{\star}\right) \qquad (3.98) \\ &= \frac{1}{2\mu}\mathrm{Tr}(H^{-1}R_s^{\star}) + O(1) \qquad (3.99) \end{aligned}$$

Substituting this result back into (3.96), we establish:

$$\lim_{N\to\infty} \mathrm{MSD}_{\mathrm{R}R}^{\mathrm{lt}} = \frac{\mu}{2}\mathrm{Tr}(H^{-1}R_s) + O(\mu^2) \tag{3.100}$$

which is exactly the same expression we have in the streaming data case [11]. If we examine the hyperbolic approximation of MSD, performance is proportional to $tanh(\mu N)$, which

implies the performance will degrade with μN but it will saturate if μN keeps increasing. Equation (3.100) shows that the limit value is the same as the uniform sampling case.

3.6 Concluding Remarks

In conclusion, this chapter studies the performance of stochastic gradient implementations under random reshuffling and provides a detailed analytical justification for the improved performance of these implementations over uniform sampling. The work focuses on constant step-size adaptation, where the agent is continuously learning. The analysis establishes analytically that random reshuffling outperforms uniform sampling by showing that iterates approach a smaller neighborhood of size $O(\mu^2)$ around the minimizer rather than $O(\mu)$. Simulation results illustrate the theoretical findings. We also summarize the conclusions in Table 3.1.

3.A Derivation of (3.13)

Indeed, from Lipschitz continuity of the gradients, we have

$$\begin{aligned}\sum_{i=0}^{N-1}\left\|g_{\boldsymbol{\sigma}^k(i)}(\boldsymbol{w}_i^k)\right\|^2 &\leq \sum_{i=0}^{N-1}\delta^2\left\|\boldsymbol{w}_i^k-\boldsymbol{w}_0^k\right\|^2\\ &=\delta^2\sum_{i=0}^{N-1}\left\|\sum_{j=1}^{i}(\boldsymbol{w}_j^k-\boldsymbol{w}_{j-1}^k)\right\|^2\\ &\overset{(3.12)}{\leq}\delta^2\sum_{i=0}^{N-1}i\sum_{j=1}^{i}\|\boldsymbol{w}_j^k-\boldsymbol{w}_{j-1}^k\|^2\end{aligned}\tag{3.101}$$

Using the equivalence relation

$$\sum_{i=0}^{N-1}\sum_{j=1}^{i}a_{ij}\equiv\sum_{j=1}^{N-1}\sum_{i=j}^{N-1}a_{ij}\tag{3.102}$$

we obtain

$$\begin{aligned}\sum_{i=0}^{N-1}\left\|g_{\boldsymbol{\sigma}^k(i)}(\boldsymbol{w}_i^k)\right\|^2 \leq & \delta^2\sum_{j=1}^{N-1}\sum_{i=j}^{N-1} i\|\boldsymbol{w}_j^k-\boldsymbol{w}_{j-1}^k\|^2\\ \leq & \frac{\delta^2N^2}{2}\sum_{j=1}^{N-1}\|\boldsymbol{w}_j^k-\boldsymbol{w}_{j-1}^k\|^2\\ \leq & \frac{\delta^2N^2}{2}\sum_{j=1}^{N}\|\boldsymbol{w}_j^k-\boldsymbol{w}_{j-1}^k\|^2\end{aligned}\tag{3.103}$$

where in the second inequality we used the fact that

$$\sum_{i=j}^{N-1} i\leq\sum_{i=0}^{N-1} i=\frac{N(N-1)}{2}\leq\frac{N^2}{2},\quad j=1,2,\ldots,N\tag{3.104}$$

We can recursively bound the difference terms in (3.103) as follows. From (1.4), we have

$$\begin{aligned}\|\boldsymbol{w}_j^k-\boldsymbol{w}_{j-1}^k\|^2 =&\mu^2\|\nabla_w Q(w_{j-1};x_{\boldsymbol{\sigma}^k(j)})\|^2\\ \leq&2\mu^2\|\nabla_w Q(w_{j-1};x_{\boldsymbol{\sigma}^k(j)})-\nabla_w Q(w^\star;x_{\boldsymbol{\sigma}^k(j)})\|^2\\ &+2\mu^2\|\nabla_w Q(w^\star;x_{\boldsymbol{\sigma}^k(j)})\|^2\\ \leq&2\mu^2\delta^2\|\widetilde{\boldsymbol{w}}_{j-1}^k\|^2+2\mu^2\|\nabla_w Q(w^\star;x_{\boldsymbol{\sigma}^k(j)})\|^2\\ \leq&4\mu^2\delta^2\|\widetilde{\boldsymbol{w}}_0^k\|^2+4\mu^2\delta^2\|\boldsymbol{w}_{j-1}^k-\boldsymbol{w}_0^k\|^2+2\mu^2\|\nabla_w Q(w^\star;x_{\boldsymbol{\sigma}^k(j)})\|^2\end{aligned}\tag{3.105}$$

Summing over j:

$$\begin{aligned}\sum_{j=1}^{N}\|\boldsymbol{w}_j^k-\boldsymbol{w}_{j-1}^k\|^2\overset{(3.6)}{\leq}&\ 4\mu^2\delta^2N\|\widetilde{\boldsymbol{w}}_0^k\|^2+2\mu^2N\mathcal{K}+4\mu^2\delta^2\sum_{j=1}^{N}\|\boldsymbol{w}_{j-1}^k-\boldsymbol{w}_0^k\|^2\\ =&\ 4\mu^2\delta^2N\|\widetilde{\boldsymbol{w}}_0^k\|^2+2\mu^2N\mathcal{K}+4\mu^2\delta^2\sum_{j=1}^{N}\left\|\sum_{i=1}^{j-1}(\boldsymbol{w}_i^k-\boldsymbol{w}_{i-1}^k)\right\|^2\\ \overset{(3.12)}{=}&\ 4\mu^2\delta^2N\|\widetilde{\boldsymbol{w}}_0^k\|^2+2\mu^2N\mathcal{K}+4\mu^2\delta^2\sum_{j=1}^{N}\sum_{i=1}^{j-1}(j-1)\|\boldsymbol{w}_i^k-\boldsymbol{w}_{i-1}^k\|^2\\ \overset{(3.102)}{=}&\ 4\mu^2\delta^2N\|\widetilde{\boldsymbol{w}}_0^k\|^2+2\mu^2N\mathcal{K}+4\mu^2\delta^2\sum_{i=1}^{N-1}\sum_{j=i+1}^{N}(j-1)\|\boldsymbol{w}_i^k-\boldsymbol{w}_{i-1}^k\|^2\end{aligned}$$

$$\begin{aligned}
&\overset{(3.104)}{\leq} 4\mu^2\delta^2 N\|\widetilde{\boldsymbol{w}}_0^k\|^2 + 2\mu^2 N\mathcal{K} + 2\mu^2\delta^2 N^2\sum_{j=1}^{N-1}\|\boldsymbol{w}_j^k - \boldsymbol{w}_{j-1}^k\|^2 \\
&\leq 4\mu^2\delta^2 N\|\widetilde{\boldsymbol{w}}_0^k\|^2 + 2\mu^2 N\mathcal{K} + 2\mu^2\delta^2 N^2\sum_{j=1}^{N}\|\boldsymbol{w}_j^k - \boldsymbol{w}_{j-1}^k\|^2 \qquad (3.106)
\end{aligned}$$

Rearranging the terms, we get

$$(1-2\mu^2\delta^2N^2)\sum_{j=1}^{N}\|\boldsymbol{w}_j^k - \boldsymbol{w}_{j-1}^k\|^2 \leq 4\mu^2\delta^2 N\|\widetilde{\boldsymbol{w}}_0^k\|^2 + 2\mu^2 N\mathcal{K} \qquad (3.107)$$

After substituting into (3.103) and simplifying, we establish (3.13).

3.B Proof of Lemma 3.1

We employ mathematical induction. First, it is easy to verify that $f(1;X,\beta) = \mathrm{V}ar(X)$. Now, assuming (3.50) is correct for case n, we consider case $n+1$:

$$\begin{aligned}
f(n+1;X,\beta) &= \mathbb{E}\left\|\sum_{j=1}^{n+1}\beta^{n+1-j}x_{\boldsymbol{\sigma}(j)}\right\|^2 \\
&= \mathbb{E}\left\|\beta\sum_{j=1}^{n}\beta^{n-j}x_{\boldsymbol{\sigma}(j)} + x_{\boldsymbol{\sigma}(n+1)}\right\|^2 \\
&= \beta^2\mathbb{E}\left\|\sum_{j=1}^{n}\beta^{n-j}x_{\boldsymbol{\sigma}(j)}\right\|^2 + \mathbb{E}\,\|x_{\boldsymbol{\sigma}(n+1)}\|^2 + 2\beta\mathbb{E}\left(\sum_{j=1}^{n}\beta^{n-j}x_{\boldsymbol{\sigma}(j)}\right)^{\mathsf{T}}x_{\boldsymbol{\sigma}(n+1)}
\end{aligned} \qquad (3.108)$$

From the uniform random reshuffling property (1.7), we know that:

$$\mathbb{E}\,\|x_{\boldsymbol{\sigma}(n+1)}\|^2 = \mathrm{V}ar(x) \qquad (3.109)$$

For the cross terms, we exploit the law of total expectation [70]:

$$\begin{aligned}
\mathbb{E}\left(\sum_{j=1}^{n}\beta^{n-j}x_{\boldsymbol{\sigma}(j)}\right)^{\mathsf{T}}x_{\boldsymbol{\sigma}(n+1)} &= \mathbb{E}_{\boldsymbol{\sigma}(1:n)}\left[\mathbb{E}_{\boldsymbol{\sigma}(n+1)}\left(\sum_{j=1}^{n}\beta^{n-j}x_{\boldsymbol{\sigma}(j)}\right)^{\mathsf{T}}x_{\boldsymbol{\sigma}(n+1)}\Big|\,\boldsymbol{\sigma}(1:n)\right] \\
&\overset{(1.8)}{=} \mathbb{E}_{\boldsymbol{\sigma}(1:n)}\left[\left(\sum_{j=1}^{n}\beta^{n-j}x_{\boldsymbol{\sigma}(j)}\right)^{\mathsf{T}}\left(\frac{1}{N-n}\sum_{j\notin\boldsymbol{\sigma}(1:n)}x_j\right)\right] \\
&= -\frac{1}{N-n}\mathbb{E}_{\boldsymbol{\sigma}(1:n)}\left[\left(\sum_{j=1}^{n}\beta^{n-j}x_{\boldsymbol{\sigma}(j)}\right)^{\mathsf{T}}\sum_{j=1}^{n}x_{\boldsymbol{\sigma}(j)}\right] \\
&= -\frac{1}{N-n}\mathbb{E}_{\boldsymbol{\sigma}(1:n)}\sum_{j=1}^{n}\beta^{n-j}\|x_{\boldsymbol{\sigma}(j)}\|^2 \\
&\quad -\frac{1}{N-n}\mathbb{E}_{\boldsymbol{\sigma}(1:n)}\sum_{i=1}^{n}\beta^{n-i}\left(\sum_{j=1,j\neq i}^{n}x_{\boldsymbol{\sigma}(i)}^{\mathsf{T}}x_{\boldsymbol{\sigma}(j)}\right) \qquad (3.110)
\end{aligned}$$

Without loss of generality, we assume $i<j$ in the following argument. If $i>j$, exchanging the place of $x_{\boldsymbol{\sigma}(i)}$ and $x_{\boldsymbol{\sigma}(j)}$ leads to the same conclusion:

$$\begin{aligned}
\mathbb{E}_{\boldsymbol{\sigma}(1:n)}\left[x_{\boldsymbol{\sigma}(i)}^{\mathsf{T}}x_{\boldsymbol{\sigma}(j)}\right] &= \mathbb{E}_{\boldsymbol{\sigma}(i),\boldsymbol{\sigma}(j)}\left[x_{\boldsymbol{\sigma}(i)}^{\mathsf{T}}x_{\boldsymbol{\sigma}(j)}\right] \\
&= \mathbb{E}_{\boldsymbol{\sigma}(i)}\left\{x_{\boldsymbol{\sigma}(i)}^{\mathsf{T}}\mathbb{E}_{\boldsymbol{\sigma}(j)}[x_{\boldsymbol{\sigma}(j)}\,|\,\boldsymbol{\sigma}(i)]\right\} \\
&\overset{(1.8)}{=} -\frac{1}{N-1}\mathbb{E}_{\boldsymbol{\sigma}(i)}\|x_{\boldsymbol{\sigma}(i)}\|^2 \\
&= -\frac{1}{N-1}\mathrm{V}ar(X) \qquad (3.111)
\end{aligned}$$

Substituting (3.111) into (3.110), we obtain:

$$\begin{aligned}
\mathbb{E}\left(\sum_{j=1}^{n}\beta^{n-j}x_{\boldsymbol{\sigma}(j)}\right)^{\mathsf{T}}x_{\boldsymbol{\sigma}(n+1)} &= -\frac{1}{N-n}\left(\sum_{j=1}^{n}\beta^{n-j}-\sum_{j=1}^{n}\beta^{n-j}\frac{n-1}{N-1}\right)\mathrm{V}ar(X) \\
&= -\frac{1}{N-1}\sum_{j=1}^{n}\beta^{j-1}\mathrm{V}ar(X) \qquad (3.112)
\end{aligned}$$

Combining (3.108), (3.109), and (3.112), we get:

$$\begin{aligned}
f(n+1;X,\beta) =& \beta^2 f(n;X,\beta) + \mathrm{V}ar(X) - \frac{2}{N-1}\sum_{j=1}^{n}\beta^j \mathrm{V}ar(X) \\
=& \left(\beta^2 \frac{(\sum_{i=0}^{n-1}\beta^{2i})N - (\sum_{i=0}^{n-1}\beta^i)^2}{N-1} + 1 - \frac{2\sum_{j=1}^{n}\beta^j}{N-1}\right)\mathrm{V}ar(X) \\
=& \frac{(\sum_{i=1}^{n}\beta^{2i})N - (\sum_{i=1}^{n}\beta^i)^2 + (N-1) - 2\sum_{j=1}^{n}\beta^j}{N-1}\mathrm{V}ar(X) \\
=& \frac{[(\sum_{i=1}^{n}\beta^{2i})N + N] - [(\sum_{i=1}^{n}\beta^i)^2 + 2\sum_{j=1}^{n}\beta^j + 1]}{N-1}\mathrm{V}ar(X) \\
=& \frac{(\sum_{i=0}^{n}\beta^{2i})N - (\sum_{i=0}^{n}\beta^i)^2}{N-1}\mathrm{V}ar(X) \qquad (3.113)
\end{aligned}$$

Hence, we conclude that (3.50) is valid.

Next, the proof of (3.51) is similar. It is easy to verify that $F(1;X,B) = R_x$. Assuming (3.51) is correct for case n, we consider case $n+1$:

$$\begin{aligned}
F(n+1;X,B) =& \mathbb{E}\left[\sum_{j=1}^{n} B^{n-j}x_{\boldsymbol{\sigma}(j)} + x_{\boldsymbol{\sigma}(n+1)}\right]\left[\sum_{j=1}^{n} x_{\boldsymbol{\sigma}(j)}^{\mathsf{T}}B^{n-j} + x_{\boldsymbol{\sigma}(n+1)}\right] \\
=& BF(n;X,B)B + \mathbb{E}\sum_{j=1}^{n} B^{n-j}x_{\boldsymbol{\sigma}(j)}x_{\boldsymbol{\sigma}(n+1)}^{\mathsf{T}} \\
&+ \mathbb{E}\sum_{j=1}^{n} x_{\boldsymbol{\sigma}(n+1)}x_{\boldsymbol{\sigma}(j)}^{\mathsf{T}}B^{n-j} + R_s \\
\overset{(1.8)}{=}& BF(n;X,B)B - \frac{1}{N-n}\mathbb{E}\sum_{j=1}^{n}\sum_{i=1}^{n} B^{n-j}\ x_{\boldsymbol{\sigma}(j)}x_{\boldsymbol{\sigma}(i)}^{\mathsf{T}} \\
&- \frac{1}{N-n}\mathbb{E}\sum_{j=1}^{n}\sum_{i=1}^{n} x_{\boldsymbol{\sigma}(i)}x_{\boldsymbol{\sigma}(j)}^{\mathsf{T}}B^{n-j} + R_s \\
\overset{(a)}{=}& BF(n;X,B)B - \frac{1}{N-1}\sum_{j=1}^{n} B^{n-j}R_s - \frac{1}{N-1}\sum_{j=1}^{n} R_s B^{n-j} + R_s \qquad (3.114)
\end{aligned}$$

where in the step (a) we use the same trick as (3.112):

$$\mathbb{E}\sum_{j=1}^{n}\sum_{i=1}^{n} B^{n-j}x_{\boldsymbol{\sigma}(j)}x_{\boldsymbol{\sigma}(i)}^{\mathsf{T}}$$

$$
\begin{aligned}
&= \mathbb{E}\sum_{j=1}^{n} B^{n-j} x_{\boldsymbol{\sigma}(j)} x_{\boldsymbol{\sigma}(j)}^{\mathsf{T}} + \mathbb{E}\sum_{j=1}^{n}\sum_{i\neq j} B^{n-j} x_{\boldsymbol{\sigma}(j)} x_{\boldsymbol{\sigma}(i)}^{\mathsf{T}} \\
&= \sum_{j=1}^{n} B^{n-j} R_s - \frac{1}{N-1}\mathbb{E}\sum_{j=1}^{n} B^{n-j} x_{\boldsymbol{\sigma}(j)} x_{\boldsymbol{\sigma}(j)} \qquad (3.115)
\end{aligned}
$$

Now if we substitute the $F(n; X, B)$ according to (3.51) into (3.114), we will conclude that the format of (3.51) is still valid for $F(n+1; X, B)$, which completes the proof.

3.C Proof of Theorem 3.2

We introduce the eigen-decomposition [71]

$$
H = U\Lambda U^{\mathsf{T}} \qquad (3.116)
$$

where U is orthogonal and Λ is diagonal with positive entries. Transforming (3.44) into the eigenvector space of H, we obtain:

$$
U^{\mathsf{T}}\widetilde{\boldsymbol{w}}_0^{\prime k+1} = (I-\mu\Lambda)^N U^{\mathsf{T}}\widetilde{\boldsymbol{w}}_0^{\prime k} - \mu U^{\mathsf{T}} s'(\boldsymbol{w}_0^k) \qquad (3.117)
$$

Let

$$
\bar{\boldsymbol{w}}_0^k \;\triangleq\; U^{\mathsf{T}}\widetilde{\boldsymbol{w}}_0^{\prime k} \qquad (3.118)
$$

and introduce any positive-definite matrix Σ. Computing the weighted square norm of both sides of (3.117) and taking expectations we get

$$
\mathbb{E}\,\|\bar{\boldsymbol{w}}_0^{k+1}\|_{\Sigma}^2 \overset{(3.45)}{=} \mathbb{E}\,\|(I-\mu\Lambda)^N\bar{\boldsymbol{w}}_0^k\|_{\Sigma}^2 + \mu^2\mathbb{E}\,\|U^{\mathsf{T}}s'(\boldsymbol{w}_0^k)\|_{\Sigma}^2 \qquad (3.119)
$$

where $\|x\|_{\Sigma}^2 \;\triangleq\; x^{\mathsf{T}}\Sigma x$ and we are free to choose Σ. The cross term is canceled thanks to property (3.45). Letting $k\to\infty$, we get

$$
\lim_{k\to\infty}\mathbb{E}\,\|\bar{\boldsymbol{w}}_0^k\|_{\Sigma-(I-\mu\Lambda)^N\Sigma(I-\mu\Lambda)^N}^2 = \lim_{k\to\infty}\mu^2\mathbb{E}\,\|U^{\mathsf{T}}s'(\boldsymbol{w}_0^k)\|_{\Sigma}^2 \qquad (3.120)
$$

To recover the mean-square-deviation $\mathbb{E}\,\|\bar{\boldsymbol{w}}_0^k\|^2$, we choose Σ as the solution to the Lyapunov equation:

$$\Sigma - (I-\mu\Lambda)^N\Sigma(I-\mu\Lambda)^N = I \tag{3.121}$$

which is given by

$$\Sigma^\star = \sum_{k=0}^{\infty}(I-\mu\Lambda)^{2Nk} = \big(I-(I-\mu\Lambda)^{2N}\big)^{-1} \tag{3.122}$$

The desired MSD is given by:

$$\mathrm{MSD}_{\mathrm{R}R}^{\mathrm{lt}} \;\triangleq\; \lim_{k\to\infty}\mathbb{E}\,\|\widetilde{\boldsymbol{w}}_0'^k\|^2 = \lim_{k\to\infty}\mathbb{E}\,\|\bar{\boldsymbol{w}}_0^k\|^2 \tag{3.123}$$

and, hence,

$$\begin{aligned}
\lim_{k\to\infty}\mathbb{E}\,\|\bar{\boldsymbol{w}}_0^k\|^2 \overset{(3.120)}{=}& \lim_{k\to\infty}\mu^2\mathbb{E}\,\|U^{\mathsf{T}}s'(\boldsymbol{w}_0^k)\|_{\Sigma^\star}^2\\
=& \lim_{k\to\infty}\mu^2\mathrm{Tr}\left(U\Sigma^\star U^{\mathsf{T}}\mathbb{E}\,s'(\boldsymbol{w}_0^k)s'(\boldsymbol{w}_0^k)^{\mathsf{T}}\right)\\
=& \lim_{k\to\infty}\mu^2\mathrm{Tr}\left(U\Sigma^\star U^{\mathsf{T}}\mathbb{E}\,s'(w^\star)s'(w^\star)^{\mathsf{T}}\right)+\\
& \lim_{k\to\infty}\mu^2\mathrm{Tr}\left(U\Sigma^\star U^{\mathsf{T}}\mathbb{E}\,s'(\boldsymbol{w}_0^k)s'(\boldsymbol{w}_0^k)^{\mathsf{T}}-\mathbb{E}\,s'(w^\star)s'(w^\star)^{\mathsf{T}}\right)\\
=&\mu^2\mathrm{Tr}\left(U\Sigma^\star U^{\mathsf{T}}\mathbb{E}\,s'(w^\star)s'(w^\star)^{\mathsf{T}}\right)+O(\mu^4)
\end{aligned} \tag{3.124}$$

The proof of last equality is provided in Appendix 3.E. Combining (3.122) and the fact that U is the eigenvector matrix of H, we get:

$$\begin{aligned}
\mathrm{MSD}_{\mathrm{R}R}^{\mathrm{lt}} =& \mu^2\mathrm{Tr}\left(U\sum_{k=0}^{\infty}(I-\mu\Lambda)^{2Nk}U^{\mathsf{T}}\mathbb{E}\,s'(w^\star)s'(w^\star)^{\mathsf{T}}\right)+O(\mu^4)\\
=& \mu^2\mathrm{Tr}\left(\sum_{k=0}^{\infty}(I-\mu H)^{2Nk}\mathbb{E}\,s'(w^\star)s'(w^\star)^{\mathsf{T}}\right)+O(\mu^4)\\
=& \mu^2\mathrm{Tr}\left(\big(I-(I-\mu H)^{2N}\big)^{-1}R_s'^\star\right)+O(\mu^4)
\end{aligned} \tag{3.125}$$

3.D Proof of Theorem 3.3

Using a similar approach to (3.43), we have

$$\widetilde{\boldsymbol{w}}_i'^k = (I-\mu H)^i \widetilde{\boldsymbol{w}}_0'^k - \mu \underbrace{\sum_{j=1}^{i} (I-\mu H)^{i-j} s_{\boldsymbol{\sigma}^k(i)}(\boldsymbol{w}_0^k)}_{\triangleq\, s_i'(\boldsymbol{w}_0^k)} \tag{3.126}$$

where

$$\mathbb{E}\,[s_i'(\boldsymbol{w}_0^k)|\boldsymbol{w}_0^k] = \mu \sum_{j=1}^{i} (I-\mu H)^{i-j}\mathbb{E}\left[s_{\boldsymbol{\sigma}^k(i)}(\boldsymbol{w}_0^k)|\boldsymbol{w}_0^k\right] = 0$$

Computing the squared norm and taking expectations we get:

$$\mathbb{E}\,\|\widetilde{\boldsymbol{w}}_i'^k\|^2 = \mathbb{E}\,\|(I-\mu H)^i \widetilde{\boldsymbol{w}}_0'^k\|^2 + \mu^2 \mathbb{E}\,\|s_i'(\boldsymbol{w}_0^k)\|^2 \tag{3.127}$$

We assume μ is sufficiently small so that $\|I-\mu H\| \le 1-\mu\nu$, i.e. requiring $\mu \le \frac{2}{\nu+\delta}$ and let $t = (1-\mu\nu)^i$. Then,

$$\mathbb{E}\,\|\widetilde{\boldsymbol{w}}_i'^k\|^2 \le (1-\mu\nu)^{2i}\mathbb{E}\,\|\widetilde{\boldsymbol{w}}_0'^k\|^2 + \mu^2 \mathbb{E}\,\|s_i'(\boldsymbol{w}_0^k)\|^2 \tag{3.128}$$

From Lemma 3.1, we know that

$$\begin{aligned} R_{s,i}'^{\star} &\triangleq \mathbb{E}\, s_i'(w^\star) s_i'(w^\star)^{\mathsf{T}} \\ &= \frac{N\left(\sum_{j=0}^{i-1}(I-\mu H)^j R_s^\star (I-\mu H)^j\right)}{N-1} \\ &\quad - \frac{\left[\sum_{j=0}^{i-1}(I-\mu H)^j\right] R_s^\star \left[\sum_{j=0}^{i-1}(I-\mu H)^j\right]}{N-1} \end{aligned} \tag{3.129}$$

and

$$\mathbb{E}\,\|s_i'(w^\star)\|^2 = \mathrm{Tr}\big(\mathbb{E}\, s_i'(w^\star) s_i'(w^\star)^{\mathsf{T}}\big) \tag{3.130}$$

With $k \to \infty$, we obtain:

$$\lim_{k\to\infty} \mathbb{E}\,\|\widetilde{\boldsymbol{w}}_i'^k\|^2 \leq (1-\mu\nu)^{2i}\mathrm{M}SD_{\mathrm{R}R}^{\mathrm{lt}} + \mu^2\mathrm{Tr}\big(R_{s,i}'^\star\big) + O(\mu^4) \tag{3.131}$$

where the $O(\mu^4)$ term comes from the same argument in (3.124).

Substituting the result (3.125), we have

$$\begin{aligned}\lim_{k\to\infty} \mathbb{E}\,\|\widetilde{\boldsymbol{w}}_i'^k\|^2 \leq &(1-\mu\nu)^{2i}\mu^2\mathrm{Tr}\left(\left(I-(I-\mu H)^{2N}\right)^{-1}R_s'^\star\right)\\ &+\mu^2\mathrm{Tr}\left(R_{s,i}'^\star\right)+O(\mu^4)\end{aligned} \tag{3.132}$$

Lastly, we multiple $\Big(1-(1-\mu\nu)^{2i}\Big)$ and its inverse at the second term of (3.132), which results in (3.59).

3.E Mismatch of Gradient Noise in (3.124)

In this appendix, we will show that

$$\lim_{k\to\infty} \mu^2\mathrm{Tr}\left(U\Sigma^\star U^{\mathsf{T}}\mathbb{E}\,s'(\boldsymbol{w}_0^k)s'(\boldsymbol{w}_0^k)^{\mathsf{T}} - \mathbb{E}\,s'(w^\star)s'(w^\star)^{\mathsf{T}}\right) = O(\mu^4) \tag{3.133}$$

which is equivalent to showing

$$\lim_{k\to\infty} \mathrm{Tr}\left(U\Sigma^\star U^{\mathsf{T}}\mathbb{E}\,s'(\boldsymbol{w}_0^k)s'(\boldsymbol{w}_0^k)^{\mathsf{T}} - \mathbb{E}\,s'(w^\star)s'(w^\star)^{\mathsf{T}}\right) = O(\mu^2) \tag{3.134}$$

Using the inequality that $|\mathrm{Tr}(X)| \leq c\|X\|$ for any square matrix and some constant c, we can just focus on the norm instead of trace:

$$\begin{aligned}&\left\|U\Sigma^\star U^{\mathsf{T}}\big(\mathbb{E}\,s'(\boldsymbol{w}_0^k)s'(\boldsymbol{w}_0^k)^{\mathsf{T}} - \mathbb{E}\,s'(w^\star)s'(w^\star)^{\mathsf{T}}\big)\right\|\\ &\quad\leq \|U\Sigma^\star U^{\mathsf{T}}\|\,\left\|\mathbb{E}\,s'(\boldsymbol{w}_0^k)s'(\boldsymbol{w}_0^k)^{\mathsf{T}} - \mathbb{E}\,s'(w^\star)s'(w^\star)^{\mathsf{T}}\right\|\\ &\quad= O(1/\mu)\,\left\|\mathbb{E}\,s'(\boldsymbol{w}_0^k)s'(\boldsymbol{w}_0^k)^{\mathsf{T}} - \mathbb{E}\,s'(w^\star)s'(w^\star)^{\mathsf{T}}\right\|\end{aligned} \tag{3.135}$$

where the last equality is due to

$$\begin{aligned}\|U\Sigma^{\star}U^{\mathsf{T}}\| \leq & \|\Sigma^{\star}\| \\ = & \left\|\left(I-(I-\mu\Lambda)^{2N}\right)^{-1}\right\| \\ = & \left\|\left(2N\mu\Lambda+O(\mu^2)\right)^{-1}\right\| \\ = & O(1/\mu)\end{aligned} \tag{3.136}$$

This result implies that we now need to show

$$\lim_{k\to\infty}\left\|\mathbb{E}\,s'(\boldsymbol{w}_0^k)s'(\boldsymbol{w}_0^k)^{\mathsf{T}}-\mathbb{E}\,s'(w^{\star})s'(w^{\star})^{\mathsf{T}}\right\| = O(\mu^3) \tag{3.137}$$

Since we have already established an expression for the covariance matrix of the gradient noise in (3.53) we have:

$$\begin{aligned}\mathbb{E}\,[s'(\boldsymbol{w}_0^k)s'(\boldsymbol{w}_0^k)^{\mathsf{T}}\,|\,\boldsymbol{w}_0^k] = & \frac{N\left(\sum_{i=0}^{N-1}(I-\mu H)^i R_s^k(I-\mu H)^i\right)}{N-1} \\ & -\frac{\left[\sum_{i=0}^{N-1}(I-\mu H)^i\right]R_s^k\left[\sum_{i=0}^{N-1}(I-\mu H)^i\right]}{N-1}\end{aligned} \tag{3.138}$$

Thus,

$$\begin{aligned}&\mathbb{E}\,[s'(\boldsymbol{w}_0^k)s'(\boldsymbol{w}_0^k)^{\mathsf{T}}\,|\,\boldsymbol{w}_0^k]-\mathbb{E}\,s'(w^{\star})s'(w^{\star})^{\mathsf{T}} \\ &= \frac{N\left(\sum_{i=0}^{N-1}(I-\mu H)^i\widetilde{R}_s^k(I-\mu H)^i\right)}{N-1}-\frac{\left[\sum_{i=0}^{N-1}(I-\mu H)^i\right]\widetilde{R}_s^k\left[\sum_{i=0}^{N-1}(I-\mu H)^i\right]}{N-1}\end{aligned} \tag{3.139}$$

where

$$\widetilde{R}_s^k \triangleq R_s^k - R_s^{\star} \tag{3.140}$$

$$R_s^k \triangleq \frac{1}{N}\sum_{n=1}^{N}\boldsymbol{s}_n(\boldsymbol{w}_0^k)\boldsymbol{s}_n(\boldsymbol{w}_0^k)^{\mathsf{T}} \tag{3.141}$$

$$R_s^{\star} \triangleq \frac{1}{N}\sum_{n=1}^{N}\boldsymbol{s}_n(w^{\star})\boldsymbol{s}_n(w^{\star})^{\mathsf{T}} \tag{3.142}$$

To simplify the notation, we rewrite the first term as follows:

$$N\left(\sum_{i=0}^{N-1}(I-\mu H)^i\widetilde{R}_s^k(I-\mu H)^i\right)=\sum_{i=0}^{N-1}\sum_{j=0}^{N-1}(I-\mu H)^i\widetilde{R}_s^k(I-\mu H)^i \tag{3.143}$$

Similarly, the second term:

$$\left[\sum_{i=0}^{N-1}(I-\mu H)^i\right]\widetilde{R}_s^k\left[\sum_{i=0}^{N-1}(I-\mu H)^i\right]=\sum_{i=0}^{N-1}\sum_{j=0}^{N-1}(I-\mu H)^i\widetilde{R}_s^k(I-\mu H)^j \tag{3.144}$$

Subtracting (3.143) from (3.144) we obtain (in the following, the notation $O(\mu^m)$ is a matrix where each entry can be bounded by $O(\mu^m)$):

$$\begin{aligned}
&\mathbb{E}\,[s'(\boldsymbol{w}_0^k)s'(\boldsymbol{w}_0^k)^{\mathsf{T}}\,|\,\boldsymbol{w}_0^k]-\mathbb{E}\,s'(w^\star)s'(w^\star)^{\mathsf{T}}\\
&=\frac{1}{N-1}\sum_{i=0}^{N-1}\sum_{j=0}^{N-1}(I-\mu H)^i\widetilde{R}_s^k[(I-\mu H)^i-(I-\mu H)^j]\\
&\overset{(a)}{=}\frac{1}{N-1}\sum_{i=0}^{N-1}\sum_{j=0}^{N-1}(I-\mu H)^i\widetilde{R}_s^k[\mu(j-i)H+O(\mu^2)]\\
&\overset{(b)}{=}\frac{1}{N-1}\mu\sum_{i=0}^{N-1}\sum_{j=0}^{N-1}(I-\mu H)^i\widetilde{R}_s^k(j-i)H+\widetilde{R}_s^kO(\mu^2)\\
&\overset{(c)}{=}\frac{1}{N-1}\mu\sum_{i=0}^{N-1}\sum_{j=0}^{N-1}\big(I+O(\mu)\big)\widetilde{R}_s^k(j-i)H+\widetilde{R}_s^kO(\mu^2)\\
&=\frac{1}{N-1}\mu\underbrace{\sum_{i=0}^{N-1}\sum_{j=0}^{N-1}\widetilde{R}_s^k(j-i)}_{=0}H+O(\mu^2)\widetilde{R}_s^kH+\widetilde{R}_s^kO(\mu^2)\\
&=O(\mu^2)\widetilde{R}_s^kH+\widetilde{R}_s^kO(\mu^2)
\end{aligned}\tag{3.145}$$

where steps (a) and (c) use the binomial expansion, and step (b) assumes the step-size is small enough so that $I-\mu H$ is stable. Next, we conclude:

$$\begin{aligned}
&\big\|\mathbb{E}\,s'(\boldsymbol{w}_0^k)s'(\boldsymbol{w}_0^k)^{\mathsf{T}}-\mathbb{E}\,s'(w^\star)s'(w^\star)^{\mathsf{T}}\big\|\\
&=\Big\|\mathbb{E}_{\boldsymbol{w}_0^k}\Big[\mathbb{E}\,s'(\boldsymbol{w}_0^k)s'(\boldsymbol{w}_0^k)^{\mathsf{T}}\,|\,\boldsymbol{w}_0^k\Big]-\mathbb{E}\,s'(w^\star)s'(w^\star)^{\mathsf{T}}\Big\|
\end{aligned}$$

$$
\begin{aligned}
&\overset{(a)}{\leq} \mathbb{E}_{\boldsymbol{w}_0^k} \left\| \left[\mathbb{E}\, s'(\boldsymbol{w}_0^k) s'(\boldsymbol{w}_0^k)^{\mathsf{T}} \mid \boldsymbol{w}_0^k \right] - \mathbb{E}\, s'(w^\star) s'(w^\star)^{\mathsf{T}} \right\| \\
&\overset{(3.145)}{=} \mathbb{E}_{\boldsymbol{w}_0^k} \| O(\mu^2) \widetilde{R}_s^k H + \widetilde{R}_s^k O(\mu^2) \| \\
&\leq O(\mu^2) \mathbb{E} \| \widetilde{R}_s^k \| \qquad (3.146)
\end{aligned}
$$

where step (a) applies Jensen's inequality. Lastly, we prove

$$
\lim_{k \to \infty} \mathbb{E} \| \widetilde{R}_s^k \| = O(\mu) \qquad (3.147)
$$

From (3.140)-(3.142), we have

$$
\begin{aligned}
\widetilde{R}_s^k =& R_s^k - R_s^\star \\
=& \frac{1}{N} \sum_{n=1}^{N} \left[s_n(\boldsymbol{w}_0^k) s_n(\boldsymbol{w}_0^k)^{\mathsf{T}} - s_n(w^\star) s_n(w^\star)^{\mathsf{T}} \right] \\
=& \frac{1}{N} \sum_{n=1}^{N} \left[s_n(\boldsymbol{w}_0^k) s_n(\boldsymbol{w}_0^k)^{\mathsf{T}} - s_n(\boldsymbol{w}_0^k) s_n(w^\star)^{\mathsf{T}} \right. \\
& \qquad \left. + s_n(\boldsymbol{w}_0^k) s_n(w^\star)^{\mathsf{T}} - s_n(w^\star) s_n(w^\star)^{\mathsf{T}} \right] \\
=& \frac{1}{N} \sum_{n=1}^{N} \left[s_n(\boldsymbol{w}_0^k) [s_n(\boldsymbol{w}_0^k) - s_n(w^\star)]^{\mathsf{T}} \right. \left. + [s_n(\boldsymbol{w}_0^k) - s_n(w^\star)] s_n(w^\star)^{\mathsf{T}} \right] \qquad (3.148)
\end{aligned}
$$

Next, it is easy to verify that $s_n(w)$ is also 2δ-Lipschitz continuity:

$$
\begin{aligned}
\| s_n(\boldsymbol{w}_0^k) - s_n(w^\star) \| &= \| \nabla J(w^\star) - \nabla Q(w^\star; x_{\boldsymbol{\sigma}^k(i)}) - \nabla J(\boldsymbol{w}_0^k) - \nabla Q(\boldsymbol{w}_0^k; x_{\boldsymbol{\sigma}^k(i)}) \| \\
&\leq \| \nabla J(\boldsymbol{w}_0^k) - \nabla J(w^\star) \| + \| \nabla Q(\boldsymbol{w}_0^k; x_n) - \nabla Q(w^\star; x_n) \| \\
&\overset{(3.4)}{\leq} 2\delta \| \widetilde{\boldsymbol{w}}_0^k \| \qquad (3.149)
\end{aligned}
$$

Taking the expectation of the norm of (3.148):

$$
\begin{aligned}
\mathbb{E} \| \widetilde{R}_s^k \| &\leq \frac{1}{N} \sum_{n=1}^{N} \mathbb{E} \left\| s_n(\boldsymbol{w}_0^k) [s_n(\boldsymbol{w}_0^k) - s_n(w^\star)]^{\mathsf{T}} + [s_n(\boldsymbol{w}_0^k) - s_n(w^\star)] s_n(w^\star)^{\mathsf{T}} \right\| \\
&\overset{(3.149)}{\leq} \frac{2}{N} \sum_{n=1}^{N} \mathbb{E} \left(\| s_n(\boldsymbol{w}_0^k) \| \delta \| \widetilde{\boldsymbol{w}}_0^k \| + \delta \| \widetilde{\boldsymbol{w}}_0^k \| \| s_n(w^\star) \| \right)
\end{aligned}
$$

$$\leq \frac{2\delta}{N}\sum_{n=1}^{N}\sqrt{\mathbb{E}\,\|s_n(\boldsymbol{w}_0^k)\|^2\mathbb{E}\,\|\widetilde{\boldsymbol{w}}_0^k\|^2}+\sqrt{\mathbb{E}\,\|\widetilde{\boldsymbol{w}}_0^k\|^2}\|s_n(w^\star)\| \tag{3.150}$$

where the last inequality exploits the Cauchy-Schwartz inequality. Next, as we prove in theorem 3.1, when $k \gg 1$:

$$\begin{aligned}\mathbb{E}\,\|\widetilde{\boldsymbol{w}}_0^k\|^2 &= O(\mu^2),\\ \mathbb{E}\,\|s_n(\boldsymbol{w}_0^k)\|^2 &\leq 2\mathbb{E}\,\|s_n(\boldsymbol{w}_0^k)-s_n(w^\star)\|^2+2\mathbb{E}\,\|s_n(w^\star)\|^2\\ &\leq O(\mu^2)+O(1)=O(1)\end{aligned} \tag{3.151}$$

Substituting the previous results into (3.150), we conclude

$$\begin{aligned}\mathbb{E}\,\|\widetilde{R}_s^k\| &\leq \frac{\delta}{N}\sum_{n=1}^{N}\Big(\sqrt{O(\mu^2)O(1)}+\sqrt{O(\mu^2)}O(1)\Big)\\ &= O(\mu),\quad k\gg 1\end{aligned} \tag{3.152}$$

3.F Bound on Long-term Difference

Subtracting (3.43) from (3.44) and then taking the conditional expectation, we obtain:

$$\begin{aligned}\mathbb{E}\,\big[\,\|\widetilde{\boldsymbol{w}}_0^{k+1}-\widetilde{\boldsymbol{w}}_0'^{k+1}\|^2\,|\,\widetilde{\boldsymbol{w}}_0^k,\widetilde{\boldsymbol{w}}_0'^k\,\big] \leq{}& \frac{1}{t}\|(I-\mu H)^N\|\|\widetilde{\boldsymbol{w}}_0^k-\widetilde{\boldsymbol{w}}_0'^k\|^2\\ &+\frac{2\mu^2}{1-t}\underbrace{\mathbb{E}\,\Big\|\sum_{i=0}^{N-1}(I-\mu H)^{N-i-1}\left(s_{\boldsymbol{\sigma}^k(i)}(\boldsymbol{w}_i^k)-s_{\boldsymbol{\sigma}^k(i)}(\boldsymbol{w}_0^k)\right)\Big\|^2}_{B}\\ &+\frac{2\mu^2}{1-t}\underbrace{\mathbb{E}\,\Big\|\sum_{i=0}^{N-1}(I-\mu H)^{N-i-1}\xi(\boldsymbol{w}_i^k)\Big\|^2}_{C}\end{aligned} \tag{3.153}$$

where we exploit the Jensen's inequality and $0<t<1$. In the following, we assume the step size is sufficiently small so that:

$$\|I-\mu H\|\leq 1-\mu\nu \tag{3.154}$$

Now, we find a tighter bound on the B term:

$$
\begin{aligned}
B &\overset{(a)}{\leq} \mathbb{E}\left(\sum_{i=0}^{N-1}\left\|(I-\mu H)^{N-i-1}\right\|\left\|s_{\boldsymbol{\sigma}^k(i)}(\boldsymbol{w}_i^k)-s_{\boldsymbol{\sigma}^k(i)}(\boldsymbol{w}_0^k)\right\|\right)^2 \\
&= \mathbb{E}\left(\sum_{i=1}^{N}\left\|(I-\mu H)^{N-i}\right\|\left\|s_{\boldsymbol{\sigma}^k(i+1)}(\boldsymbol{w}_{i-1}^k)-s_{\boldsymbol{\sigma}^k(i+1)}(\boldsymbol{w}_0^k)\right\|\right)^2 \\
&\overset{(3.149)}{\leq} \mathbb{E}\left(\sum_{i=1}^{N}\left\|(I-\mu H)^{N-i}\right\| 2\delta\left\|\boldsymbol{w}_{i-1}^k-\boldsymbol{w}_0^k\right\|\right)^2 \\
&= 4\delta^2\sum_{i=1}^{N}\sum_{j=1}^{N}\|I-\mu H\|^{(N-i)(N-j)}\mathbb{E}\left\|\boldsymbol{w}_{i-1}^k-\boldsymbol{w}_0^k\right\|\left\|\boldsymbol{w}_{j-1}^k-\boldsymbol{w}_0^k\right\| \\
&= 4\delta^2\mu^2\sum_{i=1}^{N}\sum_{j=1}^{N}\|I-\mu H\|^{(N-i)(N-j)}\times \\
&\qquad \mathbb{E}\left\|\sum_{n=1}^{i-1}\nabla_w Q(\boldsymbol{w}_{n-1}^k;x_{\boldsymbol{\sigma}^k(n)})\right\|\left\|\sum_{n=1}^{j-1}\nabla_w Q(\boldsymbol{w}_{n-1}^k;x_{\boldsymbol{\sigma}^k(n)})\right\| \\
&\overset{(b)}{\leq} 4\delta^2\mu^2\sum_{i=1}^{N}\sum_{j=1}^{N}(1-\mu\nu)^{(N-i)(N-j)}\times \\
&\qquad \sqrt{\mathbb{E}\left\|\sum_{n=1}^{i-1}\nabla Q(\boldsymbol{w}_{n-1}^k;x_{\boldsymbol{\sigma}^k(n)})\right\|^2\mathbb{E}\left\|\sum_{n=1}^{j-1}\nabla Q(\boldsymbol{w}_{n-1}^k;x_{\boldsymbol{\sigma}^k(n)})\right\|^2} \\
&= 4\delta^2\mu^2\left(\sum_{i=1}^{N}(1-\mu\nu)^{N-i}\sqrt{\mathbb{E}\left\|\sum_{n=1}^{i-1}\nabla_w Q\big(\boldsymbol{w}_{n-1}^k;x_{\boldsymbol{\sigma}^k(n)}\big)\right\|^2}\right)^2
\end{aligned}
\tag{3.155}
$$

where step (a) exploits the triangular inequality, and the sub-multiplicative property of norms, and step (b) uses Cauchy-Schwartz. Then, we establish the following when k is large enough:

$$
\begin{aligned}
\mathbb{E}\left\|\sum_{n=1}^{i-1}\nabla_w Q\big(\boldsymbol{w}_{n-1}^k;x_{\boldsymbol{\sigma}^k(n)}\big)\right\|^2 &= \mathbb{E}\left\|\sum_{n=1}^{i-1}\big(\nabla Q(\boldsymbol{w}_0^k;x_{\boldsymbol{\sigma}^k(n)})-\nabla Q(w^\star;x_{\boldsymbol{\sigma}^k(n)})+\nabla Q(w^\star;x_{\boldsymbol{\sigma}^k(n)})\big)\right\|^2 \\
&\leq 2\mathbb{E}\left\|\sum_{n=1}^{i-1}\nabla_w Q\big(w^\star;x_{\boldsymbol{\sigma}^k(n)}\big)\right\|^2 \\
&\quad + 2\mathbb{E}\left\|\sum_{n=1}^{i-1}\big(\nabla Q(\boldsymbol{w}_0^k;x_{\boldsymbol{\sigma}^k(n)})-\nabla Q(w^\star;x_{\boldsymbol{\sigma}^k(n)})\big)\right\|^2
\end{aligned}
$$

$$= 2\frac{(i-1)N-(i-1)^2}{N-1}\mathcal{K} + O(\mu^2) \tag{3.156}$$

where the last equality is because we already conclude from Lemma 3.1 and (3.6) that

$$\mathbb{E}\left\|\sum_{n=1}^{i-1}\nabla_w Q(w^\star; x_{\boldsymbol{\sigma}^k(n)})\right\|^2 = \frac{(i-1)N-(i-1)^2}{N-1}\mathcal{K} \tag{3.157}$$

Moreover, we know that for sufficiently large k:

$$\begin{aligned}
&\mathbb{E}\left\|\sum_{n=1}^{i-1}\left(\nabla_w Q(\boldsymbol{w}_0^k; x_{\boldsymbol{\sigma}^k(n)}) - \nabla_w Q(w^\star; x_{\boldsymbol{\sigma}^k(n)})\right)\right\|^2 \\
&\leq (i-1)\mathbb{E}\sum_{n=1}^{i-1}\left\|\nabla_w Q(\boldsymbol{w}_0^k; x_{\boldsymbol{\sigma}^k(n)}) - \nabla_w Q(w^\star; x_{\boldsymbol{\sigma}^k(n)})\right\|^2 \\
&\leq \delta^2(i-1)\mathbb{E}\sum_{n=1}^{i-1}\|\widetilde{\boldsymbol{w}}_{i-1}^k\|^2 \\
&= O(\mu^2)
\end{aligned} \tag{3.158}$$

Substituting previous results into (3.155):

$$B \leq 4\delta^2\mu^2\left(\sum_{i=1}^{N}(1-\mu\nu)^{N-i}\sqrt{2\frac{(i-1)N-(i-1)^2}{N-1}+O(\mu^2)}\right)^2\mathcal{K} \tag{3.159}$$

We know for any $0 \leq i \leq N$

$$\frac{(i-1)N-(i-1)^2}{N-1} \leq \frac{N^2}{4(N-1)} \tag{3.160}$$

and, hence,

$$\begin{aligned}
B \leq& 4\delta^2\mu^2\left(\frac{N^2}{2(N-1)}+O(\mu^2)\right)\left(\frac{1-(1-\mu\nu)^N}{\mu\nu}\right)^2\mathcal{K} \\
=& \frac{2\delta^2N^2}{\nu^2(N-1)}(1-(1-\mu\nu)^N)^2\mathcal{K} + O(\mu^2)
\end{aligned} \tag{3.161}$$

Next, we can bound the term C when epoch k is sufficiently large:

$$\begin{aligned}
C \leq & N \sum_{i=1}^{N} \mathbb{E} \|(I-\mu N)^{N-i} \xi(\boldsymbol{w}_{i-1}^k)\|^2 \\
\leq & N \sum_{i=1}^{N} \mathbb{E} \|\xi(\boldsymbol{w}_{i-1}^k)\|^2 \\
\overset{(3.38)}{\leq} & \frac{\kappa^2 N}{4} \sum_{i=1}^{N} \mathbb{E} \|\widetilde{\boldsymbol{w}}_{i-1}^k\|^4 \\
= & O(\mu^4)
\end{aligned} \tag{3.162}$$

where the last equality is due to (3.21):

$$\begin{aligned}
\|\widetilde{\boldsymbol{w}}_0^{k+1}\|^4 \leq & \left(\left(1-\frac{1}{2}\mu N\nu\right)\|\widetilde{\boldsymbol{w}}_0^k\|^2 + \frac{2\mu^3\delta^2 N^3}{\nu}\mathcal{K}\right)^2 \\
\leq & \frac{\left(1-\frac{1}{2}\mu N\nu\right)^2}{s}\|\widetilde{\boldsymbol{w}}_0^k\|^4 + \frac{4\mu^6\delta^4 N^6}{(1-s)\nu^2}\mathcal{K}^2
\end{aligned} \tag{3.163}$$

Let $s = 1-\frac{1}{2}\mu N\nu$, we obtain:

$$\|\widetilde{\boldsymbol{w}}_0^{k+1}\|^4 \leq \left(1-\frac{1}{2}\mu N\nu\right)\|\widetilde{\boldsymbol{w}}_0^k\|^4 + \frac{8\mu^5\delta^4 N^5}{\nu^3}\mathcal{K}^2 \tag{3.164}$$

After letting $k \to \infty$ and taking expectation, we conclude $\mathbb{E}\|\widetilde{\boldsymbol{w}}_0^k\|^4 = O(\mu^4)$.

Lastly, choosing $t = (1-\mu\nu)^N$ in (3.153) and combining (3.161) and (3.162), we establish:

$$\begin{aligned}
& \mathbb{E}\left[\, \|\widetilde{\boldsymbol{w}}_0^{k+1} - \widetilde{\boldsymbol{w}}_0'^{k+1}\|^2\right] \\
& \leq (1-\mu\nu)^N \mathbb{E}\|\widetilde{\boldsymbol{w}}_0^k - \widetilde{\boldsymbol{w}}_0'^k\|^2 \\
& \quad + \frac{2\mu^2}{1-(1-\mu\nu)^N}\frac{2\delta^2 N^2}{\nu^2(N-1)}(1-(1-\mu\nu)^N)^2\mathcal{K} + O(\mu^4)
\end{aligned} \tag{3.165}$$

Letting $k \to \infty$, we conclude

$$\mathbb{E}\|\widetilde{\boldsymbol{w}}_0^k - \widetilde{\boldsymbol{w}}_0'^k\|^2 \leq \frac{4\mu^2\delta^2 N^2}{\nu^2(N-1)}\mathcal{K} + O(\mu^3) \tag{3.166}$$

3.G Convergence for RR under Decaying Step-Sizes

In this section, we provide a sketch of the proof for the convergence of random reshuffling under decaying step-sizes. A more detailed discussion can be found in [72]. Here, we follow an alternate argument by modifying our derivation to arrive at a similar conclusion.

First, observe that inequality (3.23) continues to hold for decaying step-sizes:

$$\mathbb{E}\,\|\widetilde{\boldsymbol{w}}_0^{k+1}\|^2 \leq \left(1-\frac{1}{2}\mu(k)N\nu\right)\mathbb{E}\,\|\widetilde{\boldsymbol{w}}_0^{k}\|^2 + \frac{2\mu(k)^3\delta^2N^3}{\nu}\mathcal{K} \tag{3.167}$$

For simplicity, we only consider step-size sequences of the form:

$$\mu(k) = \frac{c}{k+1}, \quad k \geq 0 \tag{3.168}$$

where c is some positive constant. Then, we can exploit the Chung's lemma [17] or [11, Lemma F.5]:

Lemma 3.3 *Let $u(k) \geq 0$ denote a scalar sequence that satisfies the inequality recursion:*

$$u(k+1) \leq [1-a(k)]u(k) + b(k), k \geq 0 \tag{3.169}$$

When the scalar squences $\{a(k), b(k)\}$ are of the form

$$a(k) = \frac{c}{k+1}, \;\; b(i) = \frac{d}{(k+1)^{p+1}}, \quad c > 0, d > 0, p > 0 \tag{3.170}$$

it holds that, for large enough k, the sequence $u(k)$ converges to zero at one of the following rates depending on the value of c:

$$\begin{cases} u(k) \leq & \left(\frac{d}{c-p}\right)1/k^p + o(1/k^p) \qquad c > p \\ u(k) = & O(\log(k)/k^p), \qquad\qquad\qquad c = p \\ u(k) = & O(1/k^c), \qquad\qquad\qquad c < p \end{cases} \tag{3.171}$$

The fastest convergence rate occurs when $c > p$ and is in the order of $1/k^p$

Comparing (3.167) and (3.169), we have

$$\frac{cN\nu}{2(k+1)} \to a(k), \qquad \frac{2c^3\delta^2N^3\mathcal{K}}{\nu(k+1)^3} \to b(k) \tag{3.172}$$

Through lemma 3.3, we can immediately conclude that the convergence rate of $\mathbb{E}\,\|\widetilde{\boldsymbol{w}}_0^{k+1}\|^2$ is $O(1/k^2)$. The relationship between the number of epoch k and iteration i is linear. Therefore, it also follows that the convergence rate is $O(1/i^2)$.

CHAPTER 4

Variance-Reduced Stochastic Learning under Random Reshuffling

This chapter focuses on random reshuffling algorithm with the variance-reduced stochastic gradient algorithms. The existing convergence results assume uniform data sampling with replacement. However, it has been observed in related works that random reshuffling can deliver superior performance over uniform sampling and, yet, no formal proofs or guarantees of exact convergence exist for variance-reduced algorithms under random reshuffling. This chapter provides a theoretical guarantee of linear convergence under random reshuffling for SAGA in the mean-square sense; the argument is also adaptable to other variance-reduced algorithms.

4.1 Introduction and Motivation

In recent years, several useful variance-reduced stochastic gradient algorithms have been proposed, including SVRG [38], SAGA [39], Finito [40], SDCA [41], and SAG [42], with the intent of reaching the exact minimizer of an empirical risk. Under constant step-sizes and strong-convexity assumptions on the loss functions, these methods have been shown to attain linear convergence towards the exact minimizer when the data samples are uniformly sampled *with* replacement.

However, it has been observed in related works [34,35,62] that implementations that rely instead on random reshuffling (RR) of the data (i.e., sampling *without* replacement) achieve better performance than implementations that rely on uniform sampling with replacement. Under random reshuffling, the algorithm is run multiple times over the finite data set. Each

run is indexed by the integer $t \geq 1$ and is referred to as an epoch. For each epoch, the original data is reshuffled so that the sample of index i becomes the sample of index $\boldsymbol{\sigma}^t(i)$, where the symbol $\boldsymbol{\sigma}$ is used to refer to a uniform random permutation of the indices.

It was shown in [35] that random reshuffling under *decaying* step-sizes can accelerate the convergence rate of stochastic-gradient learning from $O(1/i)$ to $O(1/i^2)$ [17, 68], where i is the iteration index. It was also shown in [69, 73] that random reshuffling under small *constant* step-sizes, μ, can boost the steady-state performance of these algorithms from $O(\mu)$-suboptimal to $O(\mu^2)$-suboptimal around a small neighborhood of the exact minimizer [11]. A similar improvement in convergence rate and performance has been observed for the variance-reduced Finito algorithm [40]. However, no formal proofs or guarantees of exact convergence exist for the class of variance-reduced algorithms *under random reshuffling*, i.e., it is still not known whether these types of algorithms are still guaranteed to converge to the exact minimizer when RR is employed and under what conditions on the data. For example, in [43], another variance-reduction algorithm is proposed under reshuffling; however, no proof of convergence is provided. The closest attempts at proof are the useful arguments given in [3, 36, 74], which deal with special problem formulations. The work [75] deals with the case of incremental aggregated gradients, which corresponds to a deterministic version of RR for SAG, while the work [36] deals with SVRG in the context of ridge regression problems using regret analysis.

The random reshuffling setting has some connection with studies that employ cyclic or incremental implementations such as the incremental aggregate gradient (IAG) algorithm [3,4,74–77]. In this implementation, within each epoch, each sample is also sampled only once but keeps the same order over all epochs. In many cases, the analysis of cyclic/incremental algorithms can apply to the random reshuffling algorithms and vice versa. However, the search direction in these works on incremental algorithms is based on an aggregate gradient construction that is the best choice for studying on the random reshuffling as discussed further ahead. For this reason, different techniques are necessary to carry out the convergence analysis under random reshuffling.

Motivated by these considerations, this chapter makes two contributions. First, it resolves

the open mean-square convergence issue and provides the first theoretical proof and guarantee of linear convergence to the exact minimizer under random reshuffling for SAGA. While the argument is easily adaptable to a wider class of variance-reduced implementations, we illustrate the technique in this work for the SAGA algorithm and and explain how it extends to SVRG as well. This work focuses on providing a linear convergence guarantee on the variance-reduced algorithm under random reshuffling. However, it is still an open question to *analytically* show that variance-reduced algorithms under random reshuffling converge faster than uniform sampling. A second contribution is that, under random reshuffling, we will propose a new amortized variance-reduced gradient (AVRG) algorithm with two benefits: it has constant storage requirements in comparison to SAGA, and it has balanced gradient computations in comparison to SVRG. The balancing in computations is attained by amortizing the full gradient calculation across all iterations. AVRG is also shown analytically to converge linearly.

In preparation for the analysis, we review briefly some of the conditions and notation that are relevant. We consider a generic empirical risk function $J(w) : \mathbb{R}^M \to \mathbb{R}$, which is defined as a sample average of loss values over a possibly large but finite training set of size N:

$$w^\star \triangleq \arg\min_{w \in \mathbb{R}^M} J(w) \triangleq \frac{1}{N} \sum_{n=1}^{N} Q(w; x_n), \tag{4.1}$$

where the $\{x_n\}_{n=1}^N$ represent training data samples.

Assumption 4.1 (Loss function) *The loss function $Q(w; x_n)$ is convex, differentiable, and has a δ-Lipschitz continuous gradient, i.e., for every $n = 1, \ldots, N$ and any $w_1, w_2 \in \mathbb{R}^M$:*

$$\|\nabla_w Q(w_1; x_n) - \nabla_w Q(w_2; x_n)\| \leq \delta \|w_1 - w_2\| \tag{4.2}$$

where $\delta > 0$. We also assume that the empirical risk $J(w)$ is ν-strongly convex, namely,

$$\Big(\nabla_w J(w_1) - \nabla_w J(w_2)\Big)^{\mathsf{T}} (w_1 - w_2) \geq \nu \|w_1 - w_2\|^2 \tag{4.3}$$

4.2 SAGA with Random Reshuffling

We consider the SAGA algorithm [39] in this work, while noting that our analysis can be easily extended to other versions of variance-reduced algorithms; for example, we shall illustrate how the approach applies to the new variant designated by the acronym AVRG. We list the SAGA algorithm without the proximal step but incorporate random reshuffling into the description of the algorithm. We explain the symbols and the operation of the algorithm in Table 4.1. In the listing below, note that, random quantities are being denoted in boldface font, which will be our standard convention in this work.

Table 4.1: Listing of the SAGA algorithm with random reshuffling

SAGA with Random Reshuffling [39]

Initialization: $\boldsymbol{w}_0^0 = 0, \nabla Q(\boldsymbol{\phi}_{0,n}^0; x_n) = 0,\ n = 1, 2, \ldots, N.$
Repeat $t = 0, 1, 2 \ldots, T$ (epoch):
generate a random permutation function $\boldsymbol{\sigma}^t(\cdot)$.
Repeat $i = 0, 1, \ldots N-1$ (iteration):

$$\boldsymbol{n} = \boldsymbol{\sigma}^t(i+1) \tag{4.4}$$

$$\boldsymbol{w}_{i+1}^t = \boldsymbol{w}_i^t - \mu\Big[\nabla Q(\boldsymbol{w}_i^t; x_{\boldsymbol{n}}) - \nabla Q(\boldsymbol{\phi}_{i,\boldsymbol{n}}^t; x_{\boldsymbol{n}}) + \frac{1}{N}\sum_{n=1}^{N} \nabla Q(\boldsymbol{\phi}_{i,n}^t; x_n)\Big] \tag{4.5}$$

$$\boldsymbol{\phi}_{i+1,\boldsymbol{n}}^t = \boldsymbol{w}_{i+1}^t, \ \text{and} \ \boldsymbol{\phi}_{i+1,n}^t = \boldsymbol{\phi}_{i,n}^t, \ \text{for } n \neq \boldsymbol{n} \tag{4.6}$$

End

$$\boldsymbol{w}_0^{t+1} = \boldsymbol{w}_N^t, \ \ \boldsymbol{\phi}_0^{t+1} = \boldsymbol{\phi}_N^t \tag{4.7}$$

End

4.2.1 Operation of the Algorithm

Note that the algorithm runs a total of T times over the data of size N. For each run t, the original data $\{x_n\}_{n=1}^N$ is first randomly reshuffled so that the sample of index $i+1$ becomes the sample of index $\boldsymbol{n} = \boldsymbol{\sigma}^t(i+1)$ in that run. To facilitate the understanding of

the algorithm, we associate a block matrix $\boldsymbol{\Phi}^t$ with each run, as illustrated in Fig. 4.1. This matrix is only introduced for visualization purposes. We denote the block rows of $\boldsymbol{\Phi}^t$ by $\{\boldsymbol{\phi}_i^t\}$; one for each iteration i. Each block row $\boldsymbol{\phi}_i^t$ has size $M \times N$, with its entries generated by the SAGA recursion:

$$\boldsymbol{\phi}_i^t \triangleq \left[\begin{array}{c|c|c|c} \boldsymbol{\phi}_{i,1}^t & \boldsymbol{\phi}_{i,2}^t & \ldots & \boldsymbol{\phi}_{i,N}^t \end{array} \right] \quad (i-\text{th block row}) \tag{4.8}$$

We can therefore view $\boldsymbol{\Phi}^t$ as consisting of cells $\{\boldsymbol{\phi}_{i,n}^t\}$, each having the same $M \times 1$ size as the minimizer $w^\star$. At every iteration i, one random cell in the $(i+1)-$th block row is populated by the iterate $\boldsymbol{w}_{i+1}^t$; the column location of this random cell is determined by the value of $\boldsymbol{n}$.

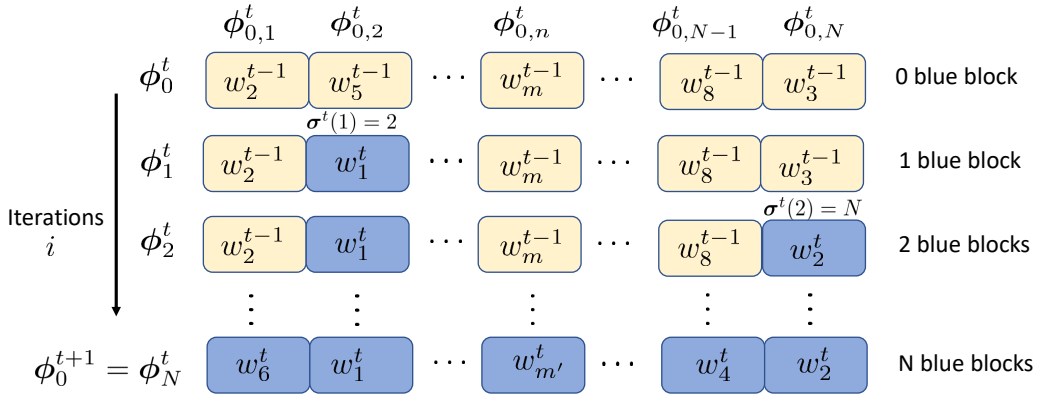

Figure 4.1: An illustration of the evolution of the history variables $\{\boldsymbol{\phi}_{i,n}^t\}$.

We refer to Fig. 1 and explain in greater detail how the cells in the figure are updated. These cells play the role of history variables. To begin with, at iteration $i = 0$, the cells in the first block row $\boldsymbol{\phi}_0^t$ will contain a randomly reshuffled version of all iterates $\{\boldsymbol{w}_1^{t-1}, \boldsymbol{w}_2^{t-1}, \ldots, \boldsymbol{w}_N^{t-1}\}$ generated during the previous run of index $t-1$. A random sample of index $\boldsymbol{n} = \boldsymbol{\sigma}^t(1)$ is selected. Assume this value turns out to be $\boldsymbol{n} = 2$. Then, as indicated in the blue cell in the second block row in the figure, the second cell of $\boldsymbol{\phi}_1^t$ is updated to $\boldsymbol{w}_1^t$ while all other cells in this row remain invariant. Moving to iteration $i = 1$, a new random sample of index $\boldsymbol{n} = \boldsymbol{\sigma}^t(2)$ is selected. Assume this value turns out to be $\boldsymbol{n} = N$. Then, as indicated again in the third block row in the figure, the last cell of $\boldsymbol{\phi}_2^t$ is updated to $\boldsymbol{w}_2^t$ while all other cells in this row remain invariant. The process continues in this manner,

by populating the cell corresponding to location $\boldsymbol{n}$ in the $i-$th block row. By the end of iteration N, all cells of $\boldsymbol{\phi}_N^t$ would have been populated by the iterates $\{\boldsymbol{w}_i^t\}$ generated during the $t-$th run. Observe that, since uniform sampling *without* replacement is used, then all weight iterates $\{\boldsymbol{w}_i^t\}$, from $i = 1$ to $i = N$ will appear in $\boldsymbol{\phi}_N^t$. These iterates appear randomly shuffled in the last row in the figure and they constitute the initial value for $\boldsymbol{\phi}_0^{t+1}$ for the next run.

4.2.2 Properties of the History Variables

Several useful observations can be drawn from Fig. 4.1. These properties will be useful in the convergence proof in subsequent sections.

Observation 1: At the start of each epoch t, the components $\{\boldsymbol{\phi}_{0,n}^t\}_{n=1}^N$ correspond to a permutation of the weight iterates from the previous run, $\{\boldsymbol{w}_i^{t-1}\}_{i=1}^N$. □

Observation 2: At the beginning of the $i-$th iteration of an epoch t, all components of indices $\{\boldsymbol{\sigma}^t(m)\}_{m=1}^i$ will be set to weight iterates obtained during the $t-$th run, namely, $\{\boldsymbol{w}_m^t\}_{m=1}^i$, while the remaining $N - i$ history positions will have values from the previous run, namely, $\{\boldsymbol{w}_{k_n}^{t-1}\}_{n=1}^{N-i}$ for some values $k_n \in \{1, 2, \ldots, N\}$. □

Observation 3: At the beginning of the $i-$th iteration of an epoch t, it holds that

$$\boldsymbol{\phi}_{i,\boldsymbol{n}}^t = \boldsymbol{\phi}_{0,\boldsymbol{n}}^t, \qquad \text{where } \boldsymbol{n} \in \boldsymbol{\sigma}^t(i+1{:}N) \tag{4.9}$$

where $\boldsymbol{\sigma}^t(i{+}1{:}N)$ represents the selected indices for future iterations $i{+}1$ to N. This property holds because, under random reshuffling, sampling is performed without replacement. □

Using these observations, the following two results can be established.

Lemma 4.1 (Distribution of history variables) *Conditioned on the previous $t-1$ epochs, each history variable $\boldsymbol{\phi}_{i,\boldsymbol{n}}^t$ has the following probability distribution at the beginning of the*

$i-th$ ($i < N$) iteration of epoch t:

$$\mathbb{P}(\boldsymbol{\phi}_{i,\boldsymbol{n}}^t|\boldsymbol{\mathcal{F}}_0^t) = \begin{cases} 1/N, & \boldsymbol{\phi}_{i,\boldsymbol{n}}^t = \boldsymbol{w}_1^{t-1} \\ 1/N, & \boldsymbol{\phi}_{i,\boldsymbol{n}}^t = \boldsymbol{w}_2^{t-1} \\ \quad\vdots & \\ 1/N, & \boldsymbol{\phi}_{i,\boldsymbol{n}}^t = \boldsymbol{w}_N^{t-1} \end{cases}, \quad \textit{for } \boldsymbol{n} \in \boldsymbol{\sigma}^t(i+1{:}N) \tag{4.10}$$

where $\boldsymbol{\mathcal{F}}_0^t$ is the collection of all information before iteration 0 at epoch t.

Proof: See Appendix 4.A. ■

Lemma 4.2 (Second-order moment of $\boldsymbol{\phi}_{i,n}^t$) *The aggregate second-order moment of each history variable $\boldsymbol{\phi}_{i,n}^t$ is equal to:*

$$\mathbb{E}\left[\sum_{n=1}^{N}\|\boldsymbol{\phi}_{i,n}^t\|^2\right] = \sum_{n'=1}^{i}\mathbb{E}\,\|\boldsymbol{w}_{n'}^t\|^2 + \frac{N-i}{N}\sum_{n=1}^{N}\mathbb{E}\,\|\boldsymbol{w}_n^{t-1}\|^2 \tag{4.11}$$

Proof: See Appendix 4.B. ■

For comparison purposes, the results obtained so far do not hold for implementations that involve sampling the data *with* replacement. For example, in that case (4.11) would be replaced instead by the following expression derived in [39]:

$$\mathbb{E}\left[\sum_{n=1}^{N}\|\boldsymbol{\phi}_{i,n}^t\|^2\right] = \mathbb{E}\,\|\boldsymbol{w}_i^t\|^2 + \frac{N-1}{N}\sum_{n=1}^{N}\mathbb{E}\,\|\boldsymbol{\phi}_{i-1,n}^t\|^2 \tag{4.12}$$

This result is similar to (4.11) only for $i = 1$. However, observe that (4.12) involves variables $\{\boldsymbol{\phi}_{i-1,n}^t\}$ on the right-hand side, instead of the variables $\{\boldsymbol{w}_n^{t-1}\}$ that appear in (4.11). This is because random reshuffling updates every history variable during each run, while uniform sampling may leave some variables $\boldsymbol{\phi}_{i-1,n}^t$ untouched. As we are going to illustrate in later experiments, this difference helps explain why SAGA under random reshuffling tends to have faster convergence rate.

4.2.3 Biased Nature of the Gradient Estimator

Before launching into the convergence analysis of the variance-reduced algorithm, we first highlight one useful observation, namely, that it is not necessary to insist on *unbiased* gradient estimators for proper operation of stochastic-gradient algorithms. To see this, let us examine first the SAGA implementation assuming uniform data sampling *with* replacement. In a manner similar to (4.5), the SAGA algorithm in this case will employ the following modified gradient direction:

$$\widehat{g}_{\boldsymbol{u}}(\boldsymbol{w}_i^t) \triangleq \nabla Q(\boldsymbol{w}_i^t; x_{\boldsymbol{u}}) - \nabla Q(\boldsymbol{\phi}_{i,\boldsymbol{u}}^t; x_{\boldsymbol{u}}) + \frac{1}{N}\sum_{n=1}^{N} \nabla Q(\boldsymbol{\phi}_{i,n}^t; x_n) \tag{4.13}$$

where the subscript $\boldsymbol{u}$ is used to denote a uniformly distributed random variable, $\boldsymbol{u} \sim \mathcal{U}[1, N]$. As a result, this modified gradient satisfies the *unbiasedness* property [39]:

$$\mathbb{E}_{\boldsymbol{u}}[\widehat{g}_{\boldsymbol{u}}(\boldsymbol{w}_i^t)|\boldsymbol{\mathcal{F}}_i^t] = \nabla J(\boldsymbol{w}_i^t) \tag{4.14}$$

where $\boldsymbol{\mathcal{F}}_i^t$ denotes the collection of all available information before iteration i at epoch t. However, this property no longer holds under random reshuffling! This is because data is now sampled *without* replacement and the selection of one index becomes dependent on the selections made prior to it. Specifically, let

$$\widehat{\boldsymbol{g}}_{\boldsymbol{n}}(\boldsymbol{w}_i^t) \triangleq \nabla Q(\boldsymbol{w}_i^t; x_{\boldsymbol{n}}) - \nabla Q(\boldsymbol{\phi}_{i,\boldsymbol{n}}^t; x_{\boldsymbol{n}}) + \frac{1}{N}\sum_{n=1}^{N} \nabla Q(\boldsymbol{\phi}_{i,n}^t; x_n) \tag{4.15}$$

denote the stochastic gradient that is employed by the SAGA recursion (4.5). It then holds that

$$\mathbb{E}_{\boldsymbol{n}}\big[\widehat{g}_{\boldsymbol{n}}(\boldsymbol{w}_i^t)|\boldsymbol{\mathcal{F}}_i^t\big] = \frac{1}{N-i}\sum_{n\notin\boldsymbol{\sigma}^t(1:i)} \big(\nabla Q(\boldsymbol{w}_i^t; x_n) - \nabla Q(\boldsymbol{\phi}_{i,n}^t; x_n)\big) + \frac{1}{N}\sum_{n=1}^{N} \nabla Q(\boldsymbol{\phi}_{i,n}^t; x_n) \tag{4.16}$$

where $\boldsymbol{n}=\boldsymbol{\sigma}^t(i+1)$ and we exploit the uniform property of random reshuffling when expanded the expectation [69]

$$\mathbb{P}[\boldsymbol{\sigma}^t(i+1)=n\,|\,\boldsymbol{\sigma}^t(1\colon i)]=\begin{cases}\dfrac{1}{N-i}, & n\notin\boldsymbol{\sigma}^t(1{:}i)\\ 0\quad, & n\in\boldsymbol{\sigma}^t(1{:}i)\end{cases}\tag{4.17}$$

where $\boldsymbol{\sigma}^t(1{:}i)$ represents the collection of permuted indices for the original samples numbered 1 through i. . It is not hard to see that the expression on the right-hand side of (4.16) is generally different from $\nabla J(\boldsymbol{w}_i^t)$. Consequently, the gradient estimate that is employed by SAGA under random reshuffling in (4.5) is not an unbiased estimator for the true gradient. Nevertheless, we will establish two useful facts in the following sections. First, the gradient estimate (4.15) becomes asymptotically unbiased when the algorithm converges, as $t\to\infty$. Second, the biased gradient estimation does not harm the convergence rate because we will observe later that SAGA under random reshuffling actually converges faster than SAGA under uniform sampling with replacement in the simulations.

4.2.4 Convergence Analysis

The analysis employs two supporting lemmas. To begin with, we relate the starting iterates for two successive epochs as follows by summing all gradient terms in (4.5) over i:

$$\begin{aligned}\boldsymbol{w}_0^{t+1}&=\boldsymbol{w}_N^t\\&=\boldsymbol{w}_{N-1}^t-\mu\left[\nabla Q(\boldsymbol{w}_{N-1}^t;x_{\boldsymbol{n}_i^t})-\nabla Q(\boldsymbol{\phi}_{N-1,\boldsymbol{n}_{N-1}^t}^t;x_{\boldsymbol{n}_{N-1}^t})+\frac{1}{N}\sum_{n=1}^N\nabla Q(\boldsymbol{\phi}_{N-1,n}^t;x_n)\right]\\&\;\;\vdots\\&=\boldsymbol{w}_0^t-\mu\sum_{i=0}^{N-1}\left[\nabla Q(\boldsymbol{w}_i^t;x_{\boldsymbol{n}_i^t})-\nabla Q(\boldsymbol{\phi}_{i,\boldsymbol{n}_i^t}^t;x_{\boldsymbol{n}_i^t})+\frac{1}{N}\sum_{n=1}^N\nabla Q(\boldsymbol{\phi}_{i,n}^t;x_n)\right]\end{aligned}\tag{4.18}$$

where we are using the notation $\boldsymbol{n}_i^t=\boldsymbol{\sigma}^t(i+1)$. As already alluded to, one main difficulty in the analysis is the fact that the gradient estimate is biased. For this reason, we shall

compare against the gradient at the start of the epoch:

$$\begin{aligned}
\boldsymbol{w}_0^{t+1} &\overset{(a)}{=} \boldsymbol{w}_0^t - \mu N \underline{\nabla J(\boldsymbol{w}_0^t)} \\
&\quad + \mu \sum_{i=0}^{N-1} \Big[\nabla Q(\boldsymbol{\phi}_{0,\boldsymbol{n}_i^t}^t; x_{\boldsymbol{n}_i^t}) - \frac{1}{N}\sum_{n=1}^{N} \underline{\underline{\nabla Q(\boldsymbol{\phi}_{0,n}^t; x_n)}}\Big] \\
&\quad - \mu \sum_{i=0}^{N-1} \Bigg[\nabla Q(\boldsymbol{w}_i^t; x_{\boldsymbol{n}_i^t}) - \underline{\nabla Q(\boldsymbol{w}_0^t; x_{\boldsymbol{n}_i^t})} \\
&\qquad + \frac{1}{N}\sum_{n=1}^{N} \Big(\nabla Q(\boldsymbol{\phi}_{i,n}^t; x_n) - \underline{\underline{\nabla Q(\boldsymbol{\phi}_{0,n}^t; x_n)}}\Big)\Bigg] \\
&\overset{(b)}{=} \boldsymbol{w}_0^t - \mu N \nabla J(\boldsymbol{w}_0^t) \\
&\quad - \mu \sum_{i=0}^{N-1} \Big[\nabla Q(\boldsymbol{w}_i^t; x_{\boldsymbol{n}_i^t}) - \nabla Q(\boldsymbol{w}_0^t; x_{\boldsymbol{n}_i^t}) \\
&\qquad + \frac{1}{N}\sum_{n=1}^{N} \Big(\nabla Q(\boldsymbol{\phi}_{i,n}^t; x_n) - \nabla Q(\boldsymbol{\phi}_{0,n}^t; x_n)\Big)\Big]
\end{aligned} \tag{4.19}$$

where in step (a) we added and subtracted $\{\nabla Q(\boldsymbol{w}_0^t; x_n)\}$ and $\{\nabla Q(\boldsymbol{\phi}_{0,n}^t; x_n)\}$, and we also changed the notation $\nabla Q(\boldsymbol{\phi}_{i,\boldsymbol{n}_i^t}^t; x_{\boldsymbol{n}_i^t})$ into $\nabla Q(\boldsymbol{\phi}_{0,\boldsymbol{n}_i^t}^t; x_{\boldsymbol{n}_i^t})$ because of observation 3; in step (b) we exploited the random reshuffling property that each index is selected only once, i.e.,

$$\sum_{i=0}^{N-1} \nabla Q(\boldsymbol{\phi}_{0,\boldsymbol{n}_i^t}^t; x_{\boldsymbol{n}_i^t}) \equiv \sum_{n=0}^{N-1} \nabla Q(\boldsymbol{\phi}_{0,n}^t; x_n) \tag{4.20}$$

We also need to appeal to a second recursion (within epoch t). By moving $\boldsymbol{w}_i^t$ in (4.5) to the left-hand side and computing the squared norm, we obtain:

$$\begin{aligned}
\|\boldsymbol{w}_{i+1}^t - \boldsymbol{w}_i^t\|^2 &= \mu^2 \Big\|\nabla Q(\boldsymbol{w}_i^t; x_{\boldsymbol{n}}) - \nabla Q(\boldsymbol{\phi}_{i,\boldsymbol{n}}^t; x_{\boldsymbol{n}}) + \frac{1}{N}\sum_{n=1}^{N} \nabla Q(\boldsymbol{\phi}_{i,n}^t; x_n)\Big\|^2 \\
&\overset{(a)}{=} \mu^2 \Big\|\nabla Q(\boldsymbol{w}_i^k; x_{\boldsymbol{j}}) - \nabla Q(\boldsymbol{w}_0^k; x_{\boldsymbol{j}}) - \nabla Q(\boldsymbol{\phi}_{i,\boldsymbol{j}}^k; x_{\boldsymbol{j}}) + \nabla Q(\boldsymbol{w}_N^{k-1}; x_{\boldsymbol{j}}) \\
&\qquad + \frac{1}{N}\sum_{n=1}^{N} [\nabla Q(\boldsymbol{\phi}_{i,n}^k; x_n) - \nabla Q(w^\star; x_n)]\Big\|^2
\end{aligned}$$

$$
\begin{aligned}
&\overset{(a)}{\leq} 3\mu^2\Big\|\nabla Q(\boldsymbol{w}_i^t;x_{\boldsymbol{n}}) - \nabla Q(\boldsymbol{w}_0^t;x_{\boldsymbol{n}})\Big\|^2 \\
&\quad + 3\mu^2\Big\|\nabla Q(\boldsymbol{\phi}_{i,\boldsymbol{n}}^t;x_{\boldsymbol{n}}) - \nabla Q(\boldsymbol{w}_N^{t-1};x_{\boldsymbol{n}})\Big\|^2 \\
&\quad + 3\mu^2\Big\|\frac{1}{N}\sum_{n=1}^{N}[\nabla Q(\boldsymbol{\phi}_{i,n}^t;x_n) - \nabla Q(w^\star;x_n)]\Big\|^2 \\
&\overset{(b)}{\leq} 3\delta^2\mu^2\|\boldsymbol{w}_i^t - \boldsymbol{w}_0^t\|^2 + 3\delta^2\mu^2\|\boldsymbol{w}_N^{t-1} - \boldsymbol{\phi}_{i,\boldsymbol{n}}^t\|^2 + \frac{3\delta^2\mu^2}{N}\sum_{n=1}^{N}\|\boldsymbol{\phi}_{i,n}^t - w^\star\|^2 \quad (4.21)
\end{aligned}
$$

where in step (a) we first added and subtracted $\nabla Q(\boldsymbol{w}_0^t;x_{\boldsymbol{n}})$ and used the fact that

$$
\frac{1}{N}\sum_{n=1}^{N}\nabla Q(w^\star;x_n) = 0 \tag{4.22}
$$

then, we employed Jensen's inequality; and step (b) is because of the assumed Lipschitz condition (4.2). Using (4.19) and (4.21), and further introducing the error quantity $\widetilde{\boldsymbol{w}}_i^t = w^\star - \boldsymbol{w}_i^t$, we can establish the following auxiliary lemmas.

Lemma 4.3 (Mean-square error recursion) *The mean-square-error at the start of each epoch satisfies the following inequality recursion for step sizes $\mu \leq 2/(N(\nu+\delta))$:*

$$
\begin{aligned}
\mathbb{E}\,\|\widetilde{\boldsymbol{w}}_0^{t+1}\|^2 \;&\leq (1-\mu\nu N)\,\mathbb{E}\,\|\widetilde{\boldsymbol{w}}_0^t\|^2 \\
&\quad + 4\mu\frac{\delta^2}{\nu}\left(\sum_{i=1}^{N-1}\mathbb{E}\,\|\boldsymbol{w}_i^t - \boldsymbol{w}_0^t\|^2 + \sum_{n'=1}^{N-1}\mathbb{E}\,\|\boldsymbol{w}_N^{t-1} - \boldsymbol{w}_{n'}^{t-1}\|^2\right) \qquad (4.23)
\end{aligned}
$$

Proof: See Appendix 4.C. ■

Roughly, the above result shows that the mean-square error across epochs evolves according to a dynamics that is determined by the scaling factor

$$
\alpha \;\overset{\Delta}{=}\; (1-\mu\nu N) \tag{4.24}
$$

which is smaller than one under the required condition in lemma 4.3. In addition, there are two driving terms in (4.23). We will refer $\sum_{i=1}^{N-1}\mathbb{E}\,\|\boldsymbol{w}_i^t - \boldsymbol{w}_0^t\|^2$ as the forward inner difference

term and to $\sum_{n'=1}^{N-1}\mathbb{E}\,\|\boldsymbol{w}_N^{t-1}-\boldsymbol{w}_{n'}^{t-1}\|^2$ as the backward inner difference term.

Lemma 4.4 (Inner differences) *The forward inner difference satisfies:*

$$\begin{aligned}&\sum_{i=1}^{N-1}\mathbb{E}\,\|\boldsymbol{w}_i^t-\boldsymbol{w}_0^t\|^2\\&\leq 5\delta^2\mu^2N^2\left(\sum_{i=1}^{N-1}\mathbb{E}\,\|\boldsymbol{w}_i^t-\boldsymbol{w}_0^t\|^2+\sum_{i=1}^{N-1}\mathbb{E}\,\|\boldsymbol{w}_N^{t-1}-\boldsymbol{w}_i^{t-1}\|^2\right)+3\delta^2\mu^2N^3\mathbb{E}\,\|\widetilde{\boldsymbol{w}}_0^t\|^2\end{aligned}\tag{4.25}$$

while the backward inner difference satisfies:

$$\begin{aligned}&\sum_{i=1}^{N-1}\mathbb{E}\,\|\boldsymbol{w}_N^{t-1}-\boldsymbol{w}_i^{t-1}\|^2\\&\leq 5\delta^2\mu^2N^2\left(\sum_{i=1}^{N-1}\mathbb{E}\|\boldsymbol{w}_i^{t-1}-\boldsymbol{w}_0^{t-1}\|^2+\sum_{i=1}^{N-1}\mathbb{E}\|\boldsymbol{w}_N^{t-2}-\boldsymbol{w}_i^{t-2}\|^2\right)+3\delta^2\mu^2N^3\mathbb{E}\,\|\widetilde{\boldsymbol{w}}_0^{t-1}\|^2\end{aligned}\tag{4.26}$$

Proof: See Appendix 4.D. ■

Combining the above lemmas, we arrive at the following theorem. Let $\widetilde{\boldsymbol{w}}_0^t\ \triangleq\ w^\star-\boldsymbol{w}_0^t$ and introduce the energy function:

$$V_{t+1}\ \triangleq\ \mathbb{E}\,\|\widetilde{\boldsymbol{w}}_0^{t+1}\|^2+\frac{5}{8}\gamma\left(\frac{1}{N}\sum_{i=1}^{N-1}\mathbb{E}\,\|\boldsymbol{w}_i^{t+1}-\boldsymbol{w}_0^{t+1}\|^2+\frac{1}{N}\sum_{i=1}^{N-1}\mathbb{E}\,\|\boldsymbol{w}_N^t-\boldsymbol{w}_i^t\|^2\right)$$

where $\gamma=8\mu\delta^2N/\nu$.

Theorem 4.1 (Linear convergence of SAGA) *For sufficiently small step-sizes, namely, for $\mu\leq\frac{\nu}{7\delta^2N}$, the quantity V_{t+1} converges linearly:*

$$V_{t+1}\leq\alpha V_t\tag{4.27}$$

where

$$\alpha=\frac{1-\mu\nu N/2}{1-24\delta^4\mu^3N^3/\nu}<1\tag{4.28}$$

It follows that $\mathbb{E}\,\|\widetilde{\boldsymbol{w}}_0^t\|^2\leq\alpha^tV_0$.

Proof: See Appendix 4.E. ■

Remark: To achieve an ϵ-optimal solution, the number of iterations required is close to $O(N\delta^2/\nu^2 \log(1/\epsilon))$, which is slower than the rate proved under sampling with replacement in [39]. The main reason is that the dependency between the samples makes it difficult to obtain a tight bound. Also, to ensure the stability of the algorithm, the stepsize upper bound is small. Meanwhile, as we will observe in the simulations later, in practice, the convergence can be faster than the original SAGA and the choice of stepsize is almost independent of size of dataset.

4.3 Amortized Variance-Reduced Gradient (AVRG) Learning

One inconvenience of the SAGA implementation is its high storage requirement, which refers to the need to track the history variables $\{\boldsymbol{\phi}_{i,n}^t\}$ or the gradients for use in (4.5). There is a need to store $O(N)$ variables. In big data applications, the size of N can be prohibitive. The same storage requirement applies to the variant with reshuffling proposed in [43]. An alternative method is the stochastic variance-reduced gradient (SVRG) algorithm [38], which is listed below (again with random reshuffling) for ease of reference.

This method replaces the history variables $\{\boldsymbol{\phi}_{i,\boldsymbol{n}}^t\}$ of SAGA by a fixed initial condition $\boldsymbol{w}_0^t$ for each epoch. This simplification greatly reduces the storage requirement. However, each epoch in SVRG is preceded by an aggregation step to compute a gradient estimate, which is time-consuming for large data sets. It also causes the operation of SVRG to become *unbalanced*, with a larger time interval needed before each epoch, and shorter time intervals needed within the epoch. Motivated by these two important considerations, we propose a new *amortized* implementation, referred to as AVRG. This new algorithm removes the initial aggregation step from SVRG and replaces it by an estimate $\boldsymbol{g}^{t+1}$. This estimate is computed iteratively within the inner loop by re-using the gradient, $\nabla Q(\boldsymbol{w}_i^t; x_{\boldsymbol{n}})$, to reduce complexity.

Table 4.2: Listing of SVRG with random reshuffling

SVRG with Random Reshuffling [38]

Initialization: $\boldsymbol{w}_0^0 = 0$.
Repeat $t = 0, 1, 2 \ldots, T$ (epochs):

$$\nabla J(\boldsymbol{w}_0^t) \;=\; \frac{1}{N}\sum_{n=1}^{N} \nabla Q(\boldsymbol{w}_0^t; x_n)$$

generate a random permutation function $\boldsymbol{\sigma}^t(\cdot)$.
Repeat $i = 0, 1, \ldots N-1$ (iteration):

$$\boldsymbol{n} = \boldsymbol{\sigma}^t(i+1) \tag{4.29}$$

$$\boldsymbol{w}_{i+1}^t = \boldsymbol{w}_i^t - \mu\left[\nabla Q(\boldsymbol{w}_i^t; x_{\boldsymbol{n}}) - \nabla Q(\boldsymbol{w}_0^t; x_{\boldsymbol{n}}) + \nabla J(\boldsymbol{w}_0^t)\right] \tag{4.30}$$

End

$$\boldsymbol{w}_0^{t+1} = \boldsymbol{w}_N^t \tag{4.31}$$

End

4.3.1 Useful Properties

Several properties stand out when we compare the proposed AVRG implementation with the previous algorithms. First, observe that the storage requirement for AVRG in each epoch is just the variables $\boldsymbol{g}^t$, $\boldsymbol{g}^{t+1}$, and $\boldsymbol{w}_0^t$, which is similar to SVRG and considerably less than SAGA.

Second, since the gradient vector $Q(\boldsymbol{w}_i^t; x_{\boldsymbol{n}})$ used in (4.34) has already been computed in (4.33), every iteration i will only require two gradients to be evaluated. Thus, the effective computation of gradients per epoch is smaller in AVRG than in SVRG.

Third, observe from Eq. (4.34) how the estimated $\boldsymbol{g}^t$ is computed by averaging the loss values at successive iterates. This construction is feasible because of the use of random reshuffling. Under random reshuffling, the collection of gradients $\{Q(\boldsymbol{w}_i^t; x_{\boldsymbol{n}})\}$ that are used in (4.34) during each epoch will end up covering the entire set of data, $\{x_n\}_{n=1}^N$. This is not necessarily the case for operation under uniform sampling with replacement. Therefore, the AVRG procedure assumes the use of random reshuffling. We will simply refer to it as

Table 4.3: Listing of AVRG with random reshuffling

AVRG with Random Reshuffling

Initialization: $\boldsymbol{w}_0^0 = 0,\ \ \boldsymbol{g}^0 = 0,\ \ \nabla Q(\boldsymbol{w}_0^0; x_n) \leftarrow 0,\ n = 1, 2, \ldots, N$
Repeat $t = 0, 1, 2 \ldots, T$ (epoch):
generate a random permutation function $\boldsymbol{\sigma}^t(\cdot)$
set $\boldsymbol{g}^{t+1} = 0$
Repeat $i = 0, 1, \ldots N-1$ (iteration):

$$\boldsymbol{n} = \boldsymbol{\sigma}^t(i+1) \tag{4.32}$$

$$\boldsymbol{w}_{i+1}^t = \boldsymbol{w}_i^t - \mu \left[\nabla Q(\boldsymbol{w}_i^t; x_{\boldsymbol{n}}) - \nabla Q(\boldsymbol{w}_0^t; x_{\boldsymbol{n}}) + \boldsymbol{g}^t\right] \tag{4.33}$$

$$\boldsymbol{g}^{t+1} \leftarrow \boldsymbol{g}^{t+1} + \frac{1}{N}\nabla Q(\boldsymbol{w}_i^t; x_{\boldsymbol{n}}) \tag{4.34}$$

End

$$\boldsymbol{w}_0^{t+1} = \boldsymbol{w}_N^t \tag{4.35}$$

End

AVRG, rather than AVRG under RR.

Fourth, unlike the SVRG algorithm, which requires a step to compute the full gradient, the AVRG implementation is amenable to decentralized implementations (i.e., to fully-decentralized implementations with no master nodes). and also to asynchronous operation [78]. The unbalanced gradient computation in SVRG poses difficulties for fully-decentralized solutions [11, 79, 80] (instead of master-slave model) and introduces idle times when multiple devices/agents with different amounts of data cooperate to solve an optimization problem. The amenability to effective decentralized solutions is a powerful convenience of the AVRG framework and one main motivation for introducing it, as explained in the related work [80].

Finally, the modified gradient direction that is employed in (4.33) by AVRG has distinctive properties in relation to the modified gradient direction (4.5) in SAGA. To see this, we

Table 4.4: Comparison of the variance-reduced implementations: SAGA, SVRG, SAG, and AVRG.

	SVRG	SVRG+RR	AVRG	SAG	SAGA	SAGA+RR
gradient computation per epoch	$2.5N$	$2.5N$	$2N$	N	N	N
extra storage requirement	$O(1)$	$O(1)$	$O(1)$	$O(N)$	$O(N)$	$O(N)$
balanced gradient computation	No	No	Yes	Yes	Yes	Yes
unbiased gradient estimator	Yes	No	No	No	Yes	No

note that the gradient direction in (4.33) can be written as

$$\begin{aligned}\widehat{g}_{\boldsymbol{n}}(\boldsymbol{w}_i^t) \;&\triangleq \nabla Q(\boldsymbol{w}_i^t; x_{\boldsymbol{n}}) - \nabla Q(\boldsymbol{w}_0^t; x_{\boldsymbol{n}}) \\ &\quad + \frac{1}{N}\sum_{n=0}^{N-1} \nabla Q(\boldsymbol{w}_n^{t-1}; x_{\boldsymbol{\sigma}^{t-1}(n+1)}) \end{aligned} \tag{4.36}$$

It is clear that even when the index $\boldsymbol{n}$ is chosen uniformly, the above vector cannot be an unbiased estimator for true gradient in general. What is more critical for convergence is that the modified gradient direction of an algorithm should satisfy the useful property that as the weight iterate gets closer to the optimal value, i.e., as $\|w^\star - \boldsymbol{w}_i^t\| \leq \epsilon$, for arbitrary small ϵ and large enough t, the modified and true gradients will also get arbitrarily close to each. This property holds for (4.36) since

$$\begin{aligned}\|\widehat{g}_{\boldsymbol{n}}(\boldsymbol{w}_i^t) - \nabla J(w^\star)\| &\leq \big\|\nabla Q(\boldsymbol{w}_i^t; x_{\boldsymbol{n}}) - \nabla Q(\boldsymbol{w}_0^t; x_{\boldsymbol{n}})\big\| \\ &\quad + \left\|\frac{1}{N}\sum_{n=1}^{N} \nabla Q(\boldsymbol{w}_{n-1}^{t-1}; x_{\boldsymbol{\sigma}^{t-1}(n)}) - \frac{1}{N}\sum_{n=1}^{N} \nabla Q(w^\star; x_n)\right\| \\ &\leq \delta\|\boldsymbol{w}_i^t - w^\star\| + \delta\|\boldsymbol{w}_0^t - w^\star\| + \frac{\delta}{N}\sum_{n=1}^{N-1}\big\|\boldsymbol{w}_{n-1}^{t-1} - w^\star\big\| \\ &\leq 3\delta\epsilon \end{aligned} \tag{4.37}$$

where in the second inequality we exploited Jensen's inequality, the triangle inequality, Lipschitz assumption, and the fact that $\boldsymbol{\sigma}^{t-1}(n)$ corresponds to sampling without replacement. Because ϵ can be chosen arbitrary small, then $\widehat{g}_{\boldsymbol{n}}(\boldsymbol{w}_i^t)$ must approach the true gradient at $w^\star$. This result implies the aforementioned asymptotic unbiasedness property of the gradient estimate. Actually, this property holds for all previous modified gradients in SAGA, SVRG, SAG, Finito, and AVRG. The work [81] also discusses a case where there is an extra error term in the gradient calculation, which supports the observation that a small gradient bias does not necessarily harm convergence. For ease of reference, Table 4.3 compares the trade-offs between storage and computational complexity of different variance-reduced algorithms with and without random reshuffling.

4.3.2 Convergence Analysis

The same approach used to establish the convergence of SAGA under RR is also suitable for AVRG. For this reason, we can be brief. First, similar to (4.19), we derive the main recursion for one epoch:

$$\begin{aligned}\boldsymbol{w}_0^{t+1}=&\boldsymbol{w}_0^t-\mu N\nabla J(\boldsymbol{w}_0^t)\\&-\mu\sum_{i=0}^{N-1}\left[\nabla Q(\boldsymbol{w}_i^t;x_{\boldsymbol{n}_i^t})-\nabla Q(\boldsymbol{w}_0^t;x_{\boldsymbol{n}_i^t})\right]\\&+\mu\sum_{i=0}^{N-1}\left[\nabla Q(\boldsymbol{w}_0^t;x_{\boldsymbol{n}_i^{t-1}})-\nabla Q(\boldsymbol{w}_i^{t-1};x_{\boldsymbol{n}_i^{t-1}})\right]\end{aligned}\tag{4.38}$$

where, for compactness of notation, we introduce $\boldsymbol{n}_i^{t-1}=\boldsymbol{\sigma}^{t-1}(i+1)$. Second, similar to (4.21), we derive an inner difference recursion:

$$\begin{aligned}\|\boldsymbol{w}_{i+1}^t-\boldsymbol{w}_i^t\|^2&=\mu^2\left\|\nabla Q(\boldsymbol{w}_i^t;x_{\boldsymbol{n}})-\nabla Q(\boldsymbol{w}_0^t;x_{\boldsymbol{n}})+\boldsymbol{g}^t\right\|^2\\&\leq 3\mu^2\delta^2\left(\|\boldsymbol{w}_i^t-\boldsymbol{w}_0^t\|^2+\frac{1}{N}\sum_{i=0}^{N-1}\|\boldsymbol{w}_i^{t-1}-\boldsymbol{w}_N^{t-1}\|^2+\|\widetilde{\boldsymbol{w}}_0^t\|^2\right)\end{aligned}\tag{4.39}$$

Next, we establish recursions related to $\widetilde{\boldsymbol{w}}_0^t$, and the forward and backward difference terms.

Lemma 4.5 (Recursions for AVRG analysis) *The mean-square-error at the start of each epoch satisfies the following inequality for step-sizes $\mu\leq 2/(N(\nu+\delta))$:*

$$\begin{aligned}\mathbb{E}\,\|\widetilde{\boldsymbol{w}}_0^{t+1}\|^2\leq&\,(1-\mu\nu N)\,\mathbb{E}\,\|\widetilde{\boldsymbol{w}}_0^t\|^2\\&+\frac{2\mu\delta^2}{\nu}\left(\sum_{i=0}^{N-1}\mathbb{E}\,\|\boldsymbol{w}_i^t-\boldsymbol{w}_0^t\|^2+\sum_{i=0}^{N-1}\mathbb{E}\,\|\boldsymbol{w}_N^{t-1}-\boldsymbol{w}_i^{t-1}\|^2\right)\end{aligned}\tag{4.40}$$

Moreover, the forward inner difference satisfies:

$$\sum_{i=0}^{N-1}\mathbb{E}\,\|\boldsymbol{w}_i^t-\boldsymbol{w}_0^t\|^2\leq\;3\mu^2\delta^2N^2\sum_{i=0}^{N-1}\mathbb{E}\,\|\boldsymbol{w}_i^t-\boldsymbol{w}_0^t\|^2$$

$$+\mu^2\delta^2N^2\left(\sum_{i=0}^{N-1}\mathbb{E}\,\|\boldsymbol{w}_N^{t-1}-\boldsymbol{w}_i^{t-1}\|^2+N\mathbb{E}\,\|\widetilde{\boldsymbol{w}}_0^t\|^2\right) \tag{4.41}$$

while the backward inner difference satisfies:

$$\begin{aligned}&\sum_{i=0}^{N-1}\mathbb{E}\,\|\boldsymbol{w}_N^t-\boldsymbol{w}_i^t\|^2\\&\quad\leq 3\mu^2\delta^2N^2\sum_{i=0}^{N-1}\mathbb{E}\,\|\boldsymbol{w}_i^t-\boldsymbol{w}_0^t\|^2+3\mu^2\delta^2N^2\left(\sum_{i=0}^{N-1}\mathbb{E}\,\|\boldsymbol{w}_i^{t-1}-\boldsymbol{w}_N^{t-1}\|^2+N\mathbb{E}\,\|\widetilde{\boldsymbol{w}}_0^t\|^2\right)\end{aligned} \tag{4.42}$$

Proof: See Appendix 4.F. ■

Likewise, we introduce the energy function:

$$V_{t+1}\;\triangleq\;\mathbb{E}\,\|\widetilde{\boldsymbol{w}}_0^{t+1}\|^2+\frac{13}{16}\gamma\left(\frac{1}{N}\sum_{i=1}^{N-1}\mathbb{E}\,\|\boldsymbol{w}_i^{t+1}-\boldsymbol{w}_0^{t+1}\|^2+\frac{1}{N}\sum_{i=1}^{N-1}\mathbb{E}\,\|\boldsymbol{w}_N^t-\boldsymbol{w}_i^t\|^2\right) \tag{4.43}$$

where $\gamma=5\mu\delta^2N/\nu$, and state the relevant convergence theorem.

Theorem 4.2 (Linear convergence of AVRG) *For sufficiently small step-sizes, namely, for $\mu\leq\frac{\nu}{6\delta^2N}$, the quantity V_{t+1} converges linearly:*

$$V_{t+1}\leq\alpha V_t \tag{4.44}$$

where

$$\alpha=\frac{1-\mu\nu N/2}{1-15\delta^4\mu^3N^3/\nu}<1 \tag{4.45}$$

It follows that $\mathbb{E}\,\|\widetilde{\boldsymbol{w}}_0^t\|^2\leq\alpha^tV_0$.

Proof: See Appendix 4.G. ■

Remark: This is similar to the theorem for SAGA under RR except for the scaling coefficients. However, in practice, the convergence curve of AVRG will be different from the one of SAGA under RR.

4.3.3 Sketch of Proof for SVRG under Random Reshuffling

We provide a brief explanation of how the same proof technique applies to SVRG.

Theorem 4.3 (Linear convergence of RR-SVRG) *For sufficiently small step-sizes, namely, for* $\mu \leq \frac{\nu}{2\delta^2 N}$*, we have:*

$$\mathbb{E}\,\|\boldsymbol{w}_0^{t+1}\|^2 \leq (1-\frac{2}{3}\mu\nu N)\mathbb{E}\,\|\boldsymbol{w}_0^t\|^2 \tag{4.46}$$

Proof: First, similar to (4.19), we derive the main recursion for one epoch:

$$w_0^{t+1} = w_0^t - \mu N\nabla J(w_0^t) - \mu\sum_{i=0}^{N-1}[\nabla Q(\boldsymbol{w}_i^t; x_{\boldsymbol{n}_i^t}) - \nabla Q(\boldsymbol{w}_0^t; x_{\boldsymbol{n}_i^t})] \tag{4.47}$$

Under $\mu \leq \frac{2}{\mu+\delta}$, after expectations and simplifying, we arrive at

$$\begin{aligned}\mathbb{E}\,\|\widetilde{\boldsymbol{w}}_0^{t+1}\|^2 &\leq \frac{(1-\mu\nu N)^2}{t}\mathbb{E}\,\|\widetilde{\boldsymbol{w}}_0^t\|^2 + \frac{\mu^2\delta^2 N}{1-t}\sum_{i=0}^{N-1}\mathbb{E}\,\|\boldsymbol{w}_i^t - \boldsymbol{w}_0^t\|^2 \\ &= (1-\mu\nu N)\,\mathbb{E}\,\|\widetilde{\boldsymbol{w}}_0^t\|^2 + \frac{\mu\delta^2}{\nu}\sum_{i=0}^{N-1}\mathbb{E}\,\|\boldsymbol{w}_i^t - \boldsymbol{w}_0^t\|^2 \end{aligned} \tag{4.48}$$

Second, similar to (4.21), we derive an inner difference recursion:

$$\begin{aligned}\|\boldsymbol{w}_{i+1}^t - \boldsymbol{w}_i^t\|^2 &= \mu^2\left\|\nabla Q(\boldsymbol{w}_i^t; x_{\boldsymbol{n}}) - \nabla Q(\boldsymbol{w}_0^t; x_{\boldsymbol{n}}) + \nabla J(w_0^t)\right\|^2 \\ &\leq 2\mu^2\delta^2\big(\|\boldsymbol{w}_i^t - \boldsymbol{w}_0^t\|^2 + \|\widetilde{\boldsymbol{w}}_0^t\|^2\big)\end{aligned} \tag{4.49}$$

Then, similar to the approach in Appendix 4.D, we establish:

$$\begin{aligned}\sum_{i=0}^{N-1}\|\boldsymbol{w}_i^t - \boldsymbol{w}_0^t\|^2 &\leq \sum_{i=0}^{N-1}\Big\|\sum_{j=1}^{i}(\boldsymbol{w}_j^t - \boldsymbol{w}_{j-1}^t)\Big\|^2 \\ &\leq \sum_{i=0}^{N-1}\sum_{j=1}^{i} i\Big\|\boldsymbol{w}_j^t - \boldsymbol{w}_{j-1}^t\Big\|^2\end{aligned}$$

$$
\begin{aligned}
&= \sum_{j=1}^{N-1} \sum_{i=j}^{N-1} i \left\| \boldsymbol{w}_j^t - \boldsymbol{w}_{j-1}^t \right\|^2 \\
&\leq \frac{N^2}{2} \sum_{j=1}^{N-1} \left\| \boldsymbol{w}_j^t - \boldsymbol{w}_{j-1}^t \right\|^2 \\
&\leq \mu^2 \delta^2 N^2 \sum_{j=1}^{N-1} \left(\|\boldsymbol{w}_i^t - \boldsymbol{w}_0^t\|^2 + \|\widetilde{\boldsymbol{w}}_0^t\|^2 \right)
\end{aligned} \tag{4.50}
$$

Note that SVRG computes the full gradient at the start of the epoch. Therefore, RR-SVRG will have a forward difference term only. Rearranging terms we get:

$$
\sum_{i=0}^{N-1} \|\boldsymbol{w}_i^t - \boldsymbol{w}_0^t\|^2 \leq \frac{\mu^2 \delta^2 N^3}{1 - \mu^2 \delta^2 N^2} \|\widetilde{\boldsymbol{w}}_0^t\|^2 \tag{4.51}
$$

Substituting back, we have

$$
\mathbb{E} \|\widetilde{\boldsymbol{w}}_0^{t+1}\|^2 \leq (1 - \mu\nu N) \mathbb{E} \|\widetilde{\boldsymbol{w}}_0^t\|^2 + \frac{\delta}{\nu} \frac{\mu^3 \delta^3 N^3}{(1 - \mu^2 \delta^2 N^2)} \mathbb{E} \|\widetilde{\boldsymbol{w}}_0^t\|^2 \tag{4.52}
$$

Supposing $\mu \leq \frac{1}{2\delta N}$, so that

$$
\frac{\mu^3 \delta^3 N^3}{(1 - \mu^2 \delta^2 N^2)} \leq \frac{4}{3} \mu^3 \delta^3 N^3 \tag{4.53}
$$

and we obtain

$$
\mathbb{E} \|\widetilde{\boldsymbol{w}}_0^{t+1}\|^2 \leq (1 - \mu\nu N) \mathbb{E} \|\widetilde{\boldsymbol{w}}_0^t\|^2 + 4\mu^3 \delta^4 N^3 / (3\nu) \mathbb{E} \|\widetilde{\boldsymbol{w}}_0^t\|^2 \tag{4.54}
$$

Under the further assumption $\mu \leq \frac{\nu}{2\delta^2 N}$, we have

$$
4\mu^3 \delta^4 N^3 / (3\nu) \leq \frac{1}{3} \mu\nu N \tag{4.55}
$$

and we conclude that

$$
\mathbb{E} \|\widetilde{\boldsymbol{w}}_0^{t+1}\|^2 \leq \left(1 - \frac{2}{3} \mu\nu N\right) \mathbb{E} \|\widetilde{\boldsymbol{w}}_0^t\|^2 \tag{4.56}
$$

■

4.3.4 Algorithms with Incremental Strategies

There is also the possibility of running through the data in a cyclic or incremental manner [66,75,82,83]. The incremental aggregated gradient (IAG) algorithm [3,4,74–77] is one such example. However, the algorithm is based on the following gradient construction

$$g_i \triangleq \frac{1}{N}\sum_{n=1}^{N}\nabla Q(w_{\tau_{i,n}};x_n) \tag{4.57}$$

where $\tau_{i,n}$ represents the gradient of the $i-$th component function sampled at time $\tau_{i,n}$. They assume that each component function is sampled at least once in the past $K \geq 0$ iterations, i.e., we have

$$i-K \leq \tau_{i,n} \leq K, \quad \forall n \in 1,2,\ldots,N \tag{4.58}$$

For the cyclic case, that constant $K = N$ and for random reshuffling $K = 2N$. This leads to a counterintuitive conclusion that cyclic algorithm is to random reshuffling due to this factor 2, which we know is not true. For this reason, different proof techniques are needed to handle random reshuffling implementations.

4.4 Simulation Results

In this section, we illustrate the convergence performance of various algorithms by numerical simulations. We consider the following regularized logistic regression problem:

$$\begin{aligned}\min_{w} \quad J(w) &= \frac{1}{N}\sum_{n=1}^{N} Q(w;h_n,\gamma(n)) \\ &\triangleq \frac{1}{N}\sum_{n=1}^{N}\Big(\frac{\rho}{2}\|w\|^2+\ln\big(1+\exp(-\gamma(n)h_n^{\mathsf{T}}w)\big)\Big)\end{aligned} \tag{4.59}$$

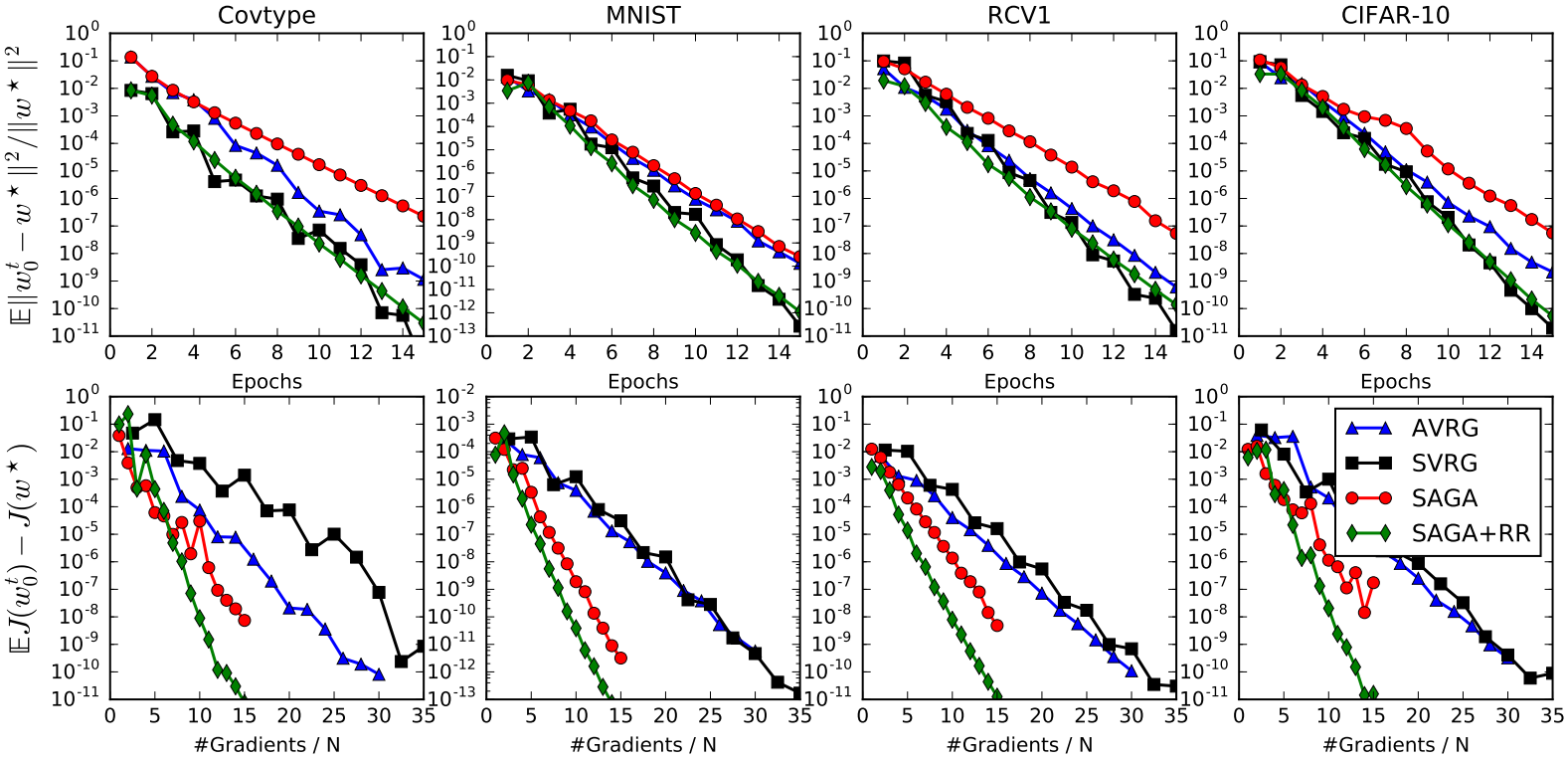

Figure 4.2: Comparison of various variance-reduced algorithms over three datasets: Covtype, MNIST, RCV1, and CIFAR-10. The top three plots compare the relative mean-square-error performance versus the epoch index, t, while the bottom three plots compare the excess risk values versus the number of gradients computed.

where $h_n \in \mathbb{R}^M$ is the feature vector, $\gamma(n) \in \{\pm 1\}$ is the class label. In all our experiments, we set $\rho = 1/N$. The optimal $w^\star$ and the corresponding risk value are calculated by means of the Scikit-Learn package. We run simulations over four datasets: covtype.binary[1], rcv1.binary [2], MNIST[3], and CIFAR-10[4]. The last two datasets have been transformed into binary classification problems by considering data with labels 0 and 1, i.e., digital zero and one classes for MNIST and airplane and automobile classes for CIFAR-10. All features have been preprocessed and normalized to the unit vector [84]. The results are exhibited in Fig. 4.2. To enable fair comparisons, we tune the step-size parameter of each algorithm for fastest convergence in each case. The plots are based on measuring the relative mean-square-error, $\mathbb{E}\,\|\boldsymbol{w}_0^t - w^\star\|^2/\|w^\star\|^2$, and the excess risk value, $\mathbb{E}\,J(\boldsymbol{w}_0^t) - J(w^\star)$. Two key facts to observe from these simulations are that 1) SAGA with RR is consistently faster than SAGA, and 2) without the high memory cost of SAGA and without the unbalanced structure of SVRG, the proposed AVRG technique is able to match their performance reasonably well. Moreover, as we shall show in future work [80], the AVRG technique enables effective distributed implementations.

---

[1] `http://www.csie.ntu.edu.tw/~cjlin/libsvmtools/datasets/`

[2] `http://www.csie.ntu.edu.tw/~cjlin/libsvmtools/datasets/`

[3] `http://yann.lecun.com/exdb/mnist/`

[4] `http://www.cs.toronto.edu/~kriz/cifar.html`

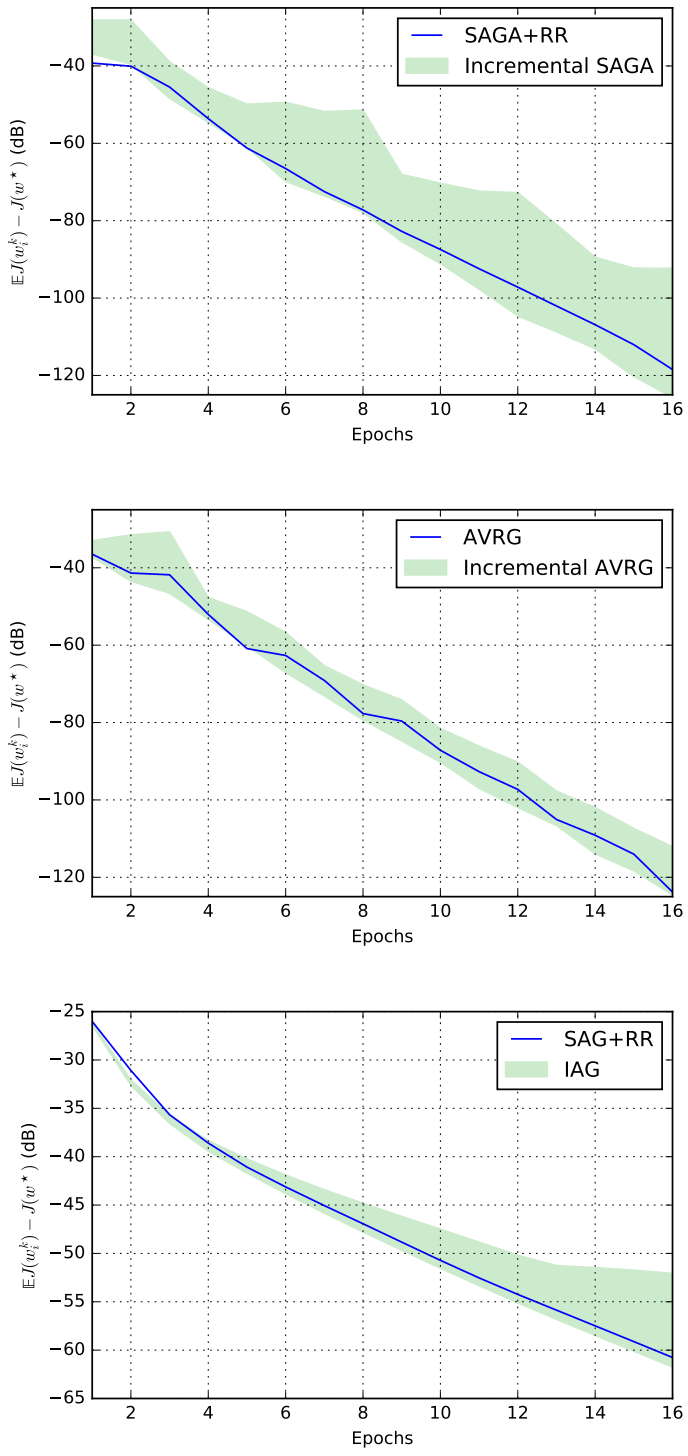

Figure 4.3: Comparison of SAGA, AVRG, and IAG [3, 4] under random reshuffling and incremental settings on the MNIST dataset. The green region is the possible learning curve under incremental setting.

We also compare the performance of random reshuffling and incremental implementations in Fig. 4.3. The plot is generated by running the logistic regression problem over the MNIST dataset with same hyperparameter setting as before. We run the incremental SAGA, SAG, and AVRG over multiple experiments with different permutation orders. The green region represents the difference between the best and worse cases over 25 different permutation orders. The plot suggests that incremental and random reshuffling implementations have similar convergence rates. However, random reshuffling has superior performance in most cases.

4.5 Discussion and Future Work

The statements of Theorems 6.1 and 7.2 are similar. This suggests that the analysis approach is applicable to a wider class of variance-reduced implementations. The statements also suggest that these types of algorithms are able to deliver linear convergence for sufficiently small constant step-sizes. One useful extension for future study is to consider situations with non-smooth loss functions. It is also useful to note that the stability ranges and convergence rates derived from the theoretical analysis tend to be more conservative than what is actually observed in experiments.

4.A Proof of Lemma 6.1

For $\boldsymbol{n} = \boldsymbol{\sigma}^t(i+1)$ and any $\boldsymbol{w}_i^{t-1}$, $i = 1, 2, \ldots, N$, it holds that

$$\begin{aligned}
\mathbb{P}(\boldsymbol{\phi}_{i,\boldsymbol{n}}^t = \boldsymbol{w}_i^{t-1} | \boldsymbol{\mathcal{F}}_0^t) &= \sum_{\boldsymbol{\sigma}^t} \mathbb{P}(\boldsymbol{\sigma}^t) \mathbb{P}(\boldsymbol{\phi}_{i,\boldsymbol{n}}^t = \boldsymbol{w}_i^{t-1} | \boldsymbol{\mathcal{F}}_0^t, \boldsymbol{\sigma}^t) \\
&= \sum_{\boldsymbol{\sigma}^t} \frac{1}{N!} \mathbb{P}(\boldsymbol{\phi}_{i,\boldsymbol{n}}^t = \boldsymbol{w}_i^{t-1} | \boldsymbol{\mathcal{F}}_0^t, \boldsymbol{\sigma}^t) \\
&= \sum_{\boldsymbol{\sigma}^t} \frac{1}{N!} \mathbb{P}(\boldsymbol{\phi}_{0,\boldsymbol{n}}^t = \boldsymbol{w}_i^{t-1} | \boldsymbol{\mathcal{F}}_0^t, \boldsymbol{\sigma}^t) \\
&= \frac{1}{N!} \sum_{\boldsymbol{\sigma}^t} \mathbb{I}\left[\boldsymbol{\phi}_{0,\boldsymbol{n}}^t = \boldsymbol{w}_i^{t-1} | \boldsymbol{\mathcal{F}}_0^t, \boldsymbol{\sigma}^t\right] \qquad (4.60)
\end{aligned}$$

The second equality is because all permutation sequences are equally probable; the third equality applies observation 3. The last equality follows from noting that, given $\boldsymbol{\mathcal{F}}_0^t$ and $\boldsymbol{\sigma}^t$, the quantity $\boldsymbol{\phi}_{0,\boldsymbol{n}}^t$ becomes a deterministic variable. In this case, the probability $\mathbb{P}(\boldsymbol{\phi}_{0,\boldsymbol{n}}^t|\boldsymbol{\mathcal{F}}_0^t,\boldsymbol{\sigma}^t)$ is either 1 or 0. We therefore express it in terms of the indicator function, where the notation $\mathbb{I}[a]=1$ when the statement a is true and is zero otherwise. Next note that there are $(N-1)!$ permutations $\boldsymbol{\sigma}^t$ with the $\boldsymbol{n}-$th position storing w_i^{t-1}. Substituting back, we get

$$\mathbb{P}(\boldsymbol{\phi}_{i,\boldsymbol{n}}^t=\boldsymbol{w}_i^{t-1}|\boldsymbol{\mathcal{F}}_0^t)=\frac{(N-1)!}{N!} \;=\; \frac{1}{N} \tag{4.61}$$

■

4.B Proof of Lemma 4.2

Conditioning on the information in the past epochs:

$$\begin{aligned}
\mathbb{E}\left[\left(\sum_{n=1}^{N}\|\boldsymbol{\phi}_{i,n}^t\|^2\right)\middle|\boldsymbol{\mathcal{F}}_0^t\right] &= \mathbb{E}\left[\left(\sum_{n\in\boldsymbol{\sigma}^t(1:i)}\|\boldsymbol{\phi}_{i,n}^t\|^2\right)\middle|\boldsymbol{\mathcal{F}}_0^t\right]+\mathbb{E}\left[\left(\sum_{n\notin\boldsymbol{\sigma}^t(1:i)}\|\boldsymbol{\phi}_{i,n}^t\|^2\right)\middle|\boldsymbol{\mathcal{F}}_0^t\right]\\
&= \mathbb{E}\left[\left(\sum_{n'=1}^{i}\|\boldsymbol{w}_{n'}^t\|^2\right)\middle|\boldsymbol{\mathcal{F}}_0^t\right]+\mathbb{E}\left[\left(\sum_{i'=i+1}^{N}\|\boldsymbol{\phi}_{i,\boldsymbol{\sigma}^t(i')}^t\|^2\right)\middle|\boldsymbol{\mathcal{F}}_0^t\right]\\
&\overset{(4.10)}{=} \sum_{n'=1}^{i}\mathbb{E}\left[\|\boldsymbol{w}_{n'}^t\|^2|\boldsymbol{\mathcal{F}}_0^t\right]+\sum_{i'=i+1}^{N}\frac{1}{N}\sum_{n=1}^{N}\|\boldsymbol{w}_n^{t-1}\|^2\\
&= \sum_{n'=1}^{i}\mathbb{E}\left[\|\boldsymbol{w}_{n'}^t\|^2|\boldsymbol{\mathcal{F}}_0^t\right]+\frac{N-i}{N}\sum_{n=1}^{N}\|\boldsymbol{w}_n^{t-1}\|^2
\end{aligned} \tag{4.62}$$

Taking expectation over $\boldsymbol{\mathcal{F}}_0^t$, we arrive at (4.11). ■

4.C Proof of Lemma 4.3

By introducing the error quantity $\widetilde{\boldsymbol{w}}_i^t = w^\star - \boldsymbol{w}_i^t$, we easily arrive at the following recursion for the evolution of the error dynamics:

$$\widetilde{\boldsymbol{w}}_0^{t+1} = \widetilde{\boldsymbol{w}}_0^t + \mu N \nabla J(\boldsymbol{w}_0^t) + \mu \sum_{i=0}^{N-1} \Bigg[\nabla Q(\boldsymbol{w}_i^t; x_{\boldsymbol{n}_i^t}) - \nabla Q(\boldsymbol{w}_0^t; x_{\boldsymbol{n}_i^t}) + \frac{1}{N} \sum_{n=1}^{N} \big(\nabla Q(\boldsymbol{\phi}_{i,n}^t; x_n) - \nabla Q(\boldsymbol{\phi}_{0,n}^t; x_n) \big) \Bigg] \tag{4.63}$$

Computing the conditional mean-square-error of both sides of (4.63), and appealing to Jensen's inequality, gives:

$$\begin{aligned}
\mathbb{E}\left[\left\| \widetilde{\boldsymbol{w}}_0^{t+1} \right\|^2 \mid \boldsymbol{\mathcal{F}}_0^t \right] &\overset{(a)}{\leq} \frac{1}{1-a} \left\| \widetilde{\boldsymbol{w}}_0^t + \mu N \nabla J(\boldsymbol{w}_0^t) \right\|^2 \\
&\quad + \frac{\mu^2}{a} \mathbb{E} \left\{ \left\| \sum_{i=0}^{N-1} \big[\nabla Q(\boldsymbol{w}_i^t; x_{\boldsymbol{n}_i^t}) - \nabla Q(\boldsymbol{w}_0^t; x_{\boldsymbol{n}_i^t}) \right. \right. \\
&\qquad \left. \left. + \frac{1}{N} \sum_{n=1}^{N} \big(\nabla Q(\boldsymbol{\phi}_{i,n}^t; x_n) - \nabla Q(\boldsymbol{\phi}_{0,n}^t; x_n) \big) \right] \right\|^2 \Big| \boldsymbol{\mathcal{F}}_0^t \right\} \\
&\overset{(b)}{\leq} \frac{1}{1-a} \left\| \widetilde{\boldsymbol{w}}_0^t + \mu N \nabla J(\boldsymbol{w}_0^t) \right\|^2 \\
&\quad + \frac{\mu^2 N}{a} \sum_{i=0}^{N-1} \mathbb{E} \left[\left\| \nabla Q(\boldsymbol{w}_i^t; x_{\boldsymbol{n}_i^t}) - \nabla Q(\boldsymbol{w}_0^t; x_{\boldsymbol{n}_i^t}) \right. \right. \\
&\qquad \left. \left. + \frac{1}{N} \sum_{n=1}^{N} \big(\nabla Q(\boldsymbol{\phi}_{i,n}^t; x_n) - \nabla Q(\boldsymbol{\phi}_{0,n}^t; x_n) \big) \right\|^2 \Big| \boldsymbol{\mathcal{F}}_0^t \right] \\
&\overset{(c)}{\leq} \frac{1}{1-a} \left\| \widetilde{\boldsymbol{w}}_0^t + \mu N \nabla J(\boldsymbol{w}_0^t) \right\|^2 \\
&\quad + \frac{2\mu^2 N}{a} \sum_{i=0}^{N-1} \mathbb{E} \left[\left\| \nabla Q(\boldsymbol{w}_i^t; x_{\boldsymbol{n}_i^t}) - \nabla Q(\boldsymbol{w}_0^t; x_{\boldsymbol{n}_i^t}) \right\|^2 \mid \boldsymbol{\mathcal{F}}_0^t \right] \\
&\quad + \frac{2\mu^2 N}{a} \sum_{i=0}^{N-1} \mathbb{E} \left[\left\| \frac{1}{N} \sum_{n=1}^{N} \nabla Q(\boldsymbol{\phi}_{i,n}^t; x_n) - \nabla Q(\boldsymbol{\phi}_{0,n}^t; x_n) \right\|^2 \Big| \boldsymbol{\mathcal{F}}_0^t \right] \\
&= \frac{1}{1-a} \left\| \widetilde{\boldsymbol{w}}_0^t + \mu N \nabla J(\boldsymbol{w}_0^t) \right\|^2
\end{aligned}$$

$$+\frac{2\mu^2 N}{a}\sum_{i=1}^{N-1}\mathbb{E}\Big[\big\|\nabla Q(\boldsymbol{w}_i^t;x_{\boldsymbol{n}})-\nabla Q(\boldsymbol{w}_0^t;x_{\boldsymbol{n}})\big\|^2\mid\boldsymbol{\mathcal{F}}_0^t\Big]$$
$$+\frac{2\mu^2 N}{a}\sum_{i=1}^{N-1}\mathbb{E}\left[\left\|\frac{1}{N}\sum_{n=1}^{N}\big(\nabla Q(\boldsymbol{\phi}_{i,n}^t;x_n)-\nabla Q(\boldsymbol{\phi}_{0,n}^t;x_n)\big)\right\|^2\Big|\boldsymbol{\mathcal{F}}_0^t\right] \quad (4.64)$$

where step (a) follows from Jensen's inequality and a can be chosen arbitrarily in the open interval $a\in(0,1)$; and steps (b) and (c) also follow from the following corollary of Jensen's inequality:

$$\left\|\sum_{i=1}^{N}y_i\right\|^2=N^2\left\|\sum_{i=1}^{N}\frac{1}{N}y_i\right\|^2\leq N\sum_{i=1}^{N}\|y_i\|^2 \quad (4.65)$$

We further know from the Lipschitz condition (4.2) that:

$$\mathbb{E}\left[\big\|\nabla Q(\boldsymbol{w}_i^t;x_{\boldsymbol{n}})-\nabla Q(\boldsymbol{w}_0^t;x_{\boldsymbol{n}})\big\|^2\Big|\boldsymbol{\mathcal{F}}_0^t\right]\leq\delta^2\mathbb{E}\left[\|\boldsymbol{w}_i^t-\boldsymbol{w}_0^t\|^2\Big|\boldsymbol{\mathcal{F}}_0^t\right] \quad (4.66)$$

and

$$\begin{aligned}
&\mathbb{E}\left[\left\|\frac{1}{N}\sum_{n=1}^{N}\big(\nabla Q(\boldsymbol{\phi}_{i,n}^t;x_n)-\nabla Q(\boldsymbol{\phi}_{0,n}^t;x_n)\big)\right\|^2\;\Big|\;\boldsymbol{\mathcal{F}}_0^t\right]\\
&\overset{(a)}{=}\mathbb{E}\left[\left\|\frac{1}{N}\sum_{n=1}^{i}\nabla Q(\boldsymbol{w}_n^t;x_{\boldsymbol{\sigma}^t(n)})-\nabla Q(\boldsymbol{\phi}_{0,\boldsymbol{\sigma}^t(n)}^t;x_{\boldsymbol{\sigma}^t(n)})\right\|^2\;\Big|\;\boldsymbol{\mathcal{F}}_0^t\right]\\
&\overset{(b)}{\leq}\frac{i\delta^2}{N^2}\sum_{n=1}^{i}\mathbb{E}\left[\|\boldsymbol{w}_n^t-\boldsymbol{\phi}_{0,\boldsymbol{\sigma}^t(n)}^t\|^2\,|\,\boldsymbol{\mathcal{F}}_0^t\right]\\
&=\frac{i\delta^2}{N^2}\sum_{n=1}^{i}\mathbb{E}\left[\|\boldsymbol{w}_n^t-\boldsymbol{w}_0^t+\boldsymbol{w}_N^{t-1}-\boldsymbol{\phi}_{0,\boldsymbol{\sigma}^t(n)}^t\|^2\,|\,\boldsymbol{\mathcal{F}}_0^t\right]\\
&\overset{(c)}{\leq}\frac{i\delta^2}{N^2}\sum_{n=1}^{i}\Big(2\mathbb{E}\left[\|\boldsymbol{w}_n^t-\boldsymbol{w}_0^t\|^2|\boldsymbol{\mathcal{F}}_0^t\right]+2\mathbb{E}\left[\|\boldsymbol{w}_N^{t-1}-\boldsymbol{\phi}_{0,\boldsymbol{\sigma}^t(n)}^t\|^2|\boldsymbol{\mathcal{F}}_0^t\right]\Big)\\
&=\frac{i\delta^2}{N^2}\left(\sum_{n=1}^{i}2\mathbb{E}\left[\|\boldsymbol{w}_n^t-\boldsymbol{w}_0^t\|^2\,|\,\boldsymbol{\mathcal{F}}_0^t\right]+\frac{2}{N}\sum_{n'=1}^{N}\|\boldsymbol{w}_N^{t-1}-\boldsymbol{w}_{n'}^{t-1}\|^2\right)
\end{aligned} \quad (4.67)$$

where step (a) holds because of observation 2, steps (b) and (c) apply Jensen's inequality; and the last equality is because of uniform random reshuffling. Next, using the strong-convexity

of the empirical risk, we have

$$\begin{aligned}\left\|\widetilde{\boldsymbol{w}}_0^t+\mu N\nabla J(\boldsymbol{w}_0^t)\right\|^2 =&\|\widetilde{\boldsymbol{w}}_0^t\|^2+\mu^2N^2\|\nabla J(\boldsymbol{w}_0^t)\|^2+2\mu N(\widetilde{\boldsymbol{w}}_0^t)^{\mathsf{T}}\nabla J(\boldsymbol{w}_0^t)\\ \leq&\Big(1-2\mu N\frac{\nu\delta}{\delta+\nu}\Big)\|\widetilde{\boldsymbol{w}}_0^t\|^2+\mu N(\mu N-\frac{2}{\delta+\nu})\|\nabla J(\boldsymbol{w}_0^t)\|^2 \end{aligned}\tag{4.68}$$

where in the first inequality we exploit the co-coercivity inequality [68] that

$$(\nabla J(x)-\nabla J(y))^{\mathsf{T}}(x-y)\geq\frac{\nu\delta}{\delta+\nu}\|x-y\|^2+\frac{1}{\delta+\nu}\|\nabla J(x)-\nabla J(y)\|^2\tag{4.69}$$

Next we require the step size to satisfy

$$\mu\leq\frac{2}{N(\delta+\nu)}\tag{4.70}$$

Then, the coefficient of the last term in (4.68) is negative. Combining with the strongly convexity property $\|\nabla J(\boldsymbol{w}_0^t)-\nabla J(w^\star)\|\geq\nu\|\widetilde{\boldsymbol{w}}_0^t\|$, we have

$$\begin{aligned}\left\|\widetilde{\boldsymbol{w}}_0^t+\mu N\nabla J(\boldsymbol{w}_0^t)\right\|^2\leq&\Big(1-2\mu N\frac{\nu\delta}{\delta+\nu}\Big)\|\widetilde{\boldsymbol{w}}_0^t\|^2+\mu N\nu^2(\mu N-\frac{2}{\delta+\nu})\|\widetilde{\boldsymbol{w}}_0^t\|^2\\ =&\Big(1-\mu\nu N)^2\|\widetilde{\boldsymbol{w}}_0^t\|^2\end{aligned}\tag{4.71}$$

Substituting (4.66), (4.67), and (4.71) into (4.64) and letting $a=\mu N\nu$, we get

$$\begin{aligned}\mathbb{E}\left[\|\widetilde{\boldsymbol{w}}_0^{t+1}\|^2\,|\,\boldsymbol{\mathcal{F}}_0^t\right]\;\leq&\,(1-\mu\nu N)\,\|\widetilde{\boldsymbol{w}}_0^t\|^2+2\mu\frac{\delta^2}{\nu}\sum_{i=1}^{N-1}\mathbb{E}\left[\|\boldsymbol{w}_i^t-\boldsymbol{w}_0^t\|^2\,|\,\boldsymbol{\mathcal{F}}_0^t\right]\\ &+2\mu\frac{\delta^2}{\nu}\sum_{i=1}^{N-1}\frac{i}{N^2}\left(\sum_{n=1}^{i}2\mathbb{E}\left[\|\boldsymbol{w}_n^t-\boldsymbol{w}_0^t\|^2\,|\,\boldsymbol{\mathcal{F}}_0^t\right]\;+\frac{2}{N}\sum_{n'=1}^{N}\|\boldsymbol{w}_N^{t-1}-\boldsymbol{w}_{n'}^{t-1}\|^2\right)\\ \overset{(a)}{=}&\,(1-\mu\nu N)\,\|\widetilde{\boldsymbol{w}}_0^t\|^2+2\mu\frac{\delta^2}{\nu}\sum_{i=1}^{N-1}\mathbb{E}\left[\|\boldsymbol{w}_i^t-\boldsymbol{w}_0^t\|^2\,|\,\boldsymbol{\mathcal{F}}_0^t\right]\\ &+2\mu\frac{\delta^2}{\nu}\sum_{n=1}^{N-1}\sum_{i=n}^{N-1}\frac{i}{N^2}\left(2\mathbb{E}\left[\|\boldsymbol{w}_n^t-\boldsymbol{w}_0^t\|^2\,|\,\boldsymbol{\mathcal{F}}_0^t\right]\;+\frac{2}{N}\sum_{n'=1}^{N}\|\boldsymbol{w}_N^{t-1}-\boldsymbol{w}_{n'}^{t-1}\|^2\right)\\ \overset{(b)}{\leq}&\,(1-\mu\nu N)\,\|\widetilde{\boldsymbol{w}}_0^t\|^2+2\mu\frac{\delta^2}{\nu}\sum_{i=1}^{N-1}\mathbb{E}\left[\|\boldsymbol{w}_i^t-\boldsymbol{w}_0^t\|^2\,|\,\boldsymbol{\mathcal{F}}_0^t\right]\end{aligned}$$

$$
\begin{aligned}
&\quad + 2\mu\frac{\delta^2}{\nu}\sum_{n=1}^{N-1}\frac{1}{2}\left(2\mathbb{E}\,\left[\|\boldsymbol{w}_n^t-\boldsymbol{w}_0^t\|^2\,|\,\boldsymbol{\mathcal{F}}_0^t\right]+\frac{2}{N}\sum_{n'=1}^{N}\|\boldsymbol{w}_N^{t-1}-\boldsymbol{w}_{n'}^{t-1}\|^2\right)\\
&\le (1-\mu\nu N)\,\|\widetilde{\boldsymbol{w}}_0^t\|^2\\
&\quad + 2\mu\frac{\delta^2}{\nu}\sum_{i=1}^{N-1}\mathbb{E}\left[\|\boldsymbol{w}_i^t-\boldsymbol{w}_0^t\|^2|\boldsymbol{\mathcal{F}}_0^t\right]+2\mu\frac{\delta^2}{\nu}\sum_{n'=1}^{N}\|\boldsymbol{w}_N^{t-1}-\boldsymbol{w}_{n'}^{t-1}\|^2\\
&\le (1-\mu\nu N)\,\|\widetilde{\boldsymbol{w}}_0^t\|^2\\
&\quad + 4\mu\frac{\delta^2}{\nu}\left(\sum_{i=1}^{N-1}\mathbb{E}\left[\|\boldsymbol{w}_i^t-\boldsymbol{w}_0^t\|^2\,|\,\boldsymbol{\mathcal{F}}_0^t\right]+\sum_{n'=1}^{N-1}\|\boldsymbol{w}_N^{t-1}-\boldsymbol{w}_{n'}^{t-1}\|^2\right) \qquad (4.72)
\end{aligned}
$$

where in step (a) and in several similar steps later, we are using the equality:

$$
\sum_{i=1}^{N-1}\sum_{n=1}^{i}f(n,i)\equiv\sum_{n=1}^{N-1}\sum_{i=n}^{N-1}f(n,i) \qquad (4.73)
$$

As for step (b), the factor $\frac{1}{2}$ is because:

$$
\sum_{i=n}^{N-1}\frac{i}{N^2}=\frac{(N-n)(N+n-1)}{2N^2}\le\frac{1}{2},\quad 1\le n\le N-1 \qquad (4.74)
$$

The last step (4.72) is unnecessary; it is used to introduce symmetry into the expression and facilitate the treatment. Taking expectation over the past history $\boldsymbol{\mathcal{F}}_0^t$ leads to (4.23).

4.D Proof of Lemma 4.4

Using (4.21), we can establish an upper bound for any inner difference based on $\boldsymbol{w}_0^t$ as follows:

$$
\begin{aligned}
\|\boldsymbol{w}_i^t-\boldsymbol{w}_0^t\|^2&=\|\boldsymbol{w}_i^t-\boldsymbol{w}_{i-1}^t+\boldsymbol{w}_{i-1}^t-\cdots-\boldsymbol{w}_0^t\|^2\\
&=i^2\left\|\frac{1}{i}\left(\boldsymbol{w}_i^t-\boldsymbol{w}_{i-1}^t+\boldsymbol{w}_{i-1}^t-\cdots-\boldsymbol{w}_0^t\right)\right\|^2\\
&\le i\sum_{m=0}^{i-1}\|\boldsymbol{w}_{m+1}^t-\boldsymbol{w}_m^t\|^2
\end{aligned}
$$

$$\overset{(4.21)}{\leq} 3\delta^2\mu^2 i \sum_{m=0}^{i-1}\left(\|\boldsymbol{w}_m^t-\boldsymbol{w}_0^t\|^2+\|\boldsymbol{w}_N^{t-1}-\boldsymbol{\phi}_{m,\boldsymbol{n}}^t\|^2+\frac{1}{N}\sum_{n=1}^{N}\|\widetilde{\boldsymbol{\phi}}_{m,n}^t\|^2\right) \tag{4.75}$$

where $\widetilde{\boldsymbol{\phi}}_{m,n}^t \triangleq w^\star - \boldsymbol{\phi}_{m,n}^t$. It is important to remark here that now $\boldsymbol{n} = \boldsymbol{\sigma}^t(m+1)$, i.e., $\boldsymbol{n}$ is always associated with the index before it. Summing over i, we have

$$\begin{aligned}
\sum_{i=1}^{N-1}\|\boldsymbol{w}_i^t-\boldsymbol{w}_0^t\|^2 &\leq 3\delta^2\mu^2\sum_{i=1}^{N-1} i\sum_{m=0}^{i-1}\Bigg(\|\boldsymbol{w}_m^t-\boldsymbol{w}_0^t\|^2+\|\boldsymbol{w}_N^{t-1}-\boldsymbol{\phi}_{m,\boldsymbol{n}}^t\|^2 \\
&\qquad\qquad +\frac{1}{N}\sum_{n=1}^{N}\|\widetilde{\boldsymbol{\phi}}_{m,n}^t\|^2\Bigg) \\
&\overset{(4.73)}{=} 3\delta^2\mu^2\sum_{m=0}^{N-2}\sum_{i=m+1}^{N-1} i\Bigg(\|\boldsymbol{w}_m^t-\boldsymbol{w}_0^t\|^2+\|\boldsymbol{w}_N^{t-1}-\boldsymbol{\phi}_{m,\boldsymbol{n}}^t\|^2 \\
&\qquad\qquad +\frac{1}{N}\sum_{n=1}^{N}\|\widetilde{\boldsymbol{\phi}}_{m,n}^t\|^2\Bigg) \\
&\overset{(a)}{\leq} \frac{3}{2}\delta^2\mu^2N^2\sum_{m=0}^{N-2}\Bigg(\|\boldsymbol{w}_m^t-\boldsymbol{w}_0^t\|^2+\|\boldsymbol{w}_N^{t-1}-\boldsymbol{\phi}_{m,\boldsymbol{n}}^t\|^2 \\
&\qquad\qquad +\frac{1}{N}\sum_{n=1}^{N}\|\widetilde{\boldsymbol{\phi}}_{m,n}^t\|^2\Bigg) \\
&\overset{(b)}{=} \frac{3}{2}\delta^2\mu^2N^2\sum_{m=0}^{N-2}\Bigg(\|\boldsymbol{w}_m^t-\boldsymbol{w}_0^t\|^2+\|\boldsymbol{w}_N^{t-1}-\boldsymbol{w}_{m+1}^{t-1}\|^2 \\
&\qquad\qquad +\frac{1}{N}\sum_{n=1}^{N}\|\widetilde{\boldsymbol{\phi}}_{m,n}^t\|^2\Bigg) \\
&= \frac{3}{2}\delta^2\mu^2N^2\Bigg(\sum_{m=0}^{N-2}\|\boldsymbol{w}_m^t-\boldsymbol{w}_0^t\|^2+\sum_{i=1}^{N-1}\|\boldsymbol{w}_N^{t-1}-\boldsymbol{w}_i^{t-1}\|^2 \\
&\qquad\qquad +\sum_{m=0}^{N-2}\frac{1}{N}\sum_{n=1}^{N}\|\widetilde{\boldsymbol{\phi}}_{m,n}^t\|^2\Bigg) \\
&\leq \frac{3}{2}\delta^2\mu^2N^2\Bigg(\sum_{i=1}^{N-1}\|\boldsymbol{w}_i^t-\boldsymbol{w}_0^t\|^2+\sum_{i=1}^{N-1}\|\boldsymbol{w}_N^{t-1}-\boldsymbol{w}_i^{t-1}\|^2 \\
&\qquad\qquad +\sum_{i=0}^{N-2}\frac{1}{N}\sum_{n=1}^{N}\|\widetilde{\boldsymbol{\phi}}_{i,n}^t\|^2\Bigg)
\end{aligned} \tag{4.76}$$

where step (a) is because $\sum_{i=m+1}^{N-1} i$ is bounded by $\frac{N^2}{2}$, and step (b) uses the fact that $\boldsymbol{\phi}_{i,\boldsymbol{n}}^t = \boldsymbol{w}_{m+1}^t$ by construction. Then, computing the conditional expectation, we get:

$$\begin{aligned}
&\sum_{i=1}^{N} \mathbb{E}\left[\|\boldsymbol{w}_i^t - \boldsymbol{w}_0^t\|^2 \,|\, \boldsymbol{\mathcal{F}}_0^t\right] \\
&\leq \frac{3}{2}\delta^2\mu^2 N^2 \left(\sum_{i=1}^{N-1} \mathbb{E}\left[\|\boldsymbol{w}_i^t - \boldsymbol{w}_0^t\|^2 \,|\, \boldsymbol{\mathcal{F}}_0^t\right] + \sum_{i=1}^{N-1} \|\boldsymbol{w}_N^{t-1} - \boldsymbol{w}_i^{t-1}\|^2 \right. \\
&\qquad \left. + \sum_{i=0}^{N-2} \frac{1}{N} \sum_{n=1}^{N} \mathbb{E}\left[\|\widetilde{\boldsymbol{\phi}}_{i,n}^t\|^2 \,|\, \boldsymbol{\mathcal{F}}_0^t\right] \right) \qquad (4.77)
\end{aligned}$$

To bound the last term, we first separate it into two quantities:

$$\begin{aligned}
\mathbb{E}\left[\|\widetilde{\boldsymbol{\phi}}_{i,n}^t\|^2 \,|\, \boldsymbol{\mathcal{F}}_0^t\right] &= \mathbb{E}\left[\|\widetilde{\boldsymbol{\phi}}_{i,n}^t - \widetilde{\boldsymbol{w}}_0^t + \widetilde{\boldsymbol{w}}_0^t\|^2 \,|\, \boldsymbol{\mathcal{F}}_0^t\right] \\
&\leq 2\mathbb{E}\left[\|\boldsymbol{\phi}_{i,n}^t - \boldsymbol{w}_0^t\|^2 \,|\, \boldsymbol{\mathcal{F}}_0^t\right] + 2\|\widetilde{\boldsymbol{w}}_0^t\|^2 \qquad (4.78)
\end{aligned}$$

Using an argument similar to Lemma 4.2, we can establish that:

$$\mathbb{E}\left[\sum_{n=1}^{N} \|\boldsymbol{\phi}_{i,n}^t - \boldsymbol{w}_0^t\|^2 \,\middle|\, \boldsymbol{\mathcal{F}}_0^t\right] = \sum_{n=1}^{i} \mathbb{E}\left[\|\boldsymbol{w}_i^t - \boldsymbol{w}_0^t\|^2 \,\middle|\, \boldsymbol{\mathcal{F}}_0^t\right] + \frac{N-i}{N} \sum_{n'=1}^{N} \|\boldsymbol{w}_N^{t-1} - \boldsymbol{w}_{n'}^{t-1}\|^2 \quad (4.79)$$

Combining results (4.78) and (4.79), we can bound the last term of (4.77):

$$\begin{aligned}
&\sum_{i=0}^{N-2} \frac{1}{N} \sum_{n=1}^{N} \mathbb{E}\left[\|\widetilde{\boldsymbol{\phi}}_{i,n}^t\|^2 \,|\, \boldsymbol{\mathcal{F}}_0^t\right] \\
&\leq \sum_{i=0}^{N-1} \frac{2}{N} \left(\sum_{n=1}^{i} \mathbb{E}\left[\|\boldsymbol{w}_i^t - \boldsymbol{w}_0^t\|^2 \,|\, \boldsymbol{\mathcal{F}}_0^t\right] + \frac{N-i}{N} \sum_{n'=1}^{N} \|\boldsymbol{w}_N^{t-1} - \boldsymbol{w}_{n'}^{t-1}\|^2 \right) + 2N\|\widetilde{\boldsymbol{w}}_0^t\|^2 \\
&\leq \frac{2}{N} \sum_{i=0}^{N-1} \sum_{n=1}^{i} \mathbb{E}\left[\|\boldsymbol{w}_i^t - \boldsymbol{w}_0^t\|^2 \,|\, \boldsymbol{\mathcal{F}}_0^t\right] + \frac{N+1}{N} \sum_{n'=1}^{N} \|\boldsymbol{w}_N^{t-1} - \boldsymbol{w}_{n'}^{t-1}\|^2 + 2N\|\widetilde{\boldsymbol{w}}_0^t\|^2 \\
&\leq 2\sum_{i=0}^{N-1} \mathbb{E}\left[\|\boldsymbol{w}_i^t - \boldsymbol{w}_0^t\|^2 \,|\, \boldsymbol{\mathcal{F}}_0^t\right] + 2\sum_{n'=1}^{N} \|\boldsymbol{w}_N^{t-1} - \boldsymbol{w}_{n'}^{t-1}\|^2 + 2N\|\widetilde{\boldsymbol{w}}_0^t\|^2 \\
&= 2\sum_{i=1}^{N-1} \mathbb{E}[\|\boldsymbol{w}_i^t - \boldsymbol{w}_0^t\|^2 | \boldsymbol{\mathcal{F}}_0^t] + 2\sum_{n'=1}^{N-1} \|\boldsymbol{w}_N^{t-1} - \boldsymbol{w}_{n'}^{t-1}\|^2 + 2N\|\widetilde{\boldsymbol{w}}_0^t\|^2 \qquad (4.80)
\end{aligned}$$

Substituting back into (4.77), we have:

$$
\begin{aligned}
&\sum_{i=1}^{N-1}\mathbb{E}\left[\|\boldsymbol{w}_i^t-\boldsymbol{w}_0^t\|^2\,|\,\boldsymbol{\mathcal{F}}_0^t\right]\\
&\le\frac{3}{2}\delta^2\mu^2N^2\left(\sum_{i=1}^{N-1}\mathbb{E}\left[\|\boldsymbol{w}_i^t-\boldsymbol{w}_0^t\|^2\,|\,\boldsymbol{\mathcal{F}}_0^t\right]+\sum_{i=1}^{N-1}\|\boldsymbol{w}_N^{t-1}-\boldsymbol{w}_i^{t-1}\|^2\right.\\
&\qquad\qquad+2\sum_{i=1}^{N-1}\mathbb{E}\left[\|\boldsymbol{w}_i^t-\boldsymbol{w}_0^t\|^2\,|\,\boldsymbol{\mathcal{F}}_0^t\right]\\
&\qquad\qquad\left.+2\sum_{n'=1}^{N-1}\|\boldsymbol{w}_N^{t-1}-\boldsymbol{w}_{n'}^{t-1}\|^2+2N\|\widetilde{\boldsymbol{w}}_0^t\|^2\right)\\
&\le\frac{9}{2}\delta^2\mu^2N^2\left(\sum_{i=1}^{N-1}\mathbb{E}\left[\|\boldsymbol{w}_i^t-\boldsymbol{w}_0^t\|^2\,|\,\boldsymbol{\mathcal{F}}_0^t\right]+\sum_{i=1}^{N-1}\|\boldsymbol{w}_N^{t-1}-\boldsymbol{w}_i^{t-1}\|^2\right)\\
&\quad+3\delta^2\mu^2N^3\|\widetilde{\boldsymbol{w}}_0^t\|^2\\
&\le5\delta^2\mu^2N^2\left(\sum_{i=1}^{N-1}\mathbb{E}[\|\boldsymbol{w}_i^t-\boldsymbol{w}_0^t\|^2\,|\,\boldsymbol{\mathcal{F}}_0^t]+\sum_{i=1}^{N-1}\|\boldsymbol{w}_N^{t-1}-\boldsymbol{w}_i^{t-1}\|^2\right)\\
&\quad+3\delta^2\mu^2N^3\|\widetilde{\boldsymbol{w}}_0^t\|^2
\end{aligned}
\tag{4.81}
$$

Taking expectation over the filtration leads to (4.25).

Next, following similar arguments, we have the following for backward inner difference term:

$$
\begin{aligned}
&\|\boldsymbol{w}_N^{t-1}-\boldsymbol{w}_i^{t-1}\|^2\\
&=\|\boldsymbol{w}_N^{t-1}-\boldsymbol{w}_{N-1}^{t-1}+\boldsymbol{w}_{N-1}^{t-1}-\cdots-\boldsymbol{w}_i^{t-1}\|^2\\
&\le(N-i)\sum_{m=i}^{N-1}\|\boldsymbol{w}_{m+1}^{t-1}-\boldsymbol{w}_m^{t-1}\|^2\\
&\overset{(4.21)}{\le}3\delta^2\mu^2(N-i)\sum_{m=i}^{N-1}\left(\|\boldsymbol{w}_m^{t-1}-\boldsymbol{w}_0^{t-1}\|^2+\|\boldsymbol{w}_N^{t-2}-\boldsymbol{\phi}_{m,\boldsymbol{n}}^{t-1}\|^2+\frac{1}{N}\sum_{n=1}^{N}\|\widetilde{\boldsymbol{\phi}}_{m,n}^{t-1}\|^2\right)
\end{aligned}
\tag{4.82}
$$

where $\widetilde{\boldsymbol{\phi}}_{m,n}^{t-1} \triangleq w^\star - \boldsymbol{\phi}_{m,n}^{t-1}$ and now $\boldsymbol{n} = \boldsymbol{\sigma}^{t-1}(m+1)$. Summing over i, we have

$$\begin{aligned}
\sum_{i=1}^{N-1} \|\boldsymbol{w}_N^{t-1} - \boldsymbol{w}_i^{t-1}\|^2 &\leq 3\delta^2\mu^2 \sum_{i=1}^{N-1}(N-i) \sum_{m=i}^{N-1} \Bigg(\|\boldsymbol{w}_m^{t-1} - \boldsymbol{w}_0^{t-1}\|^2 + \|\boldsymbol{w}_N^{t-2} - \boldsymbol{\phi}_{m,\boldsymbol{n}}^{t-1}\|^2 \\
&\qquad + \frac{1}{N}\sum_{n=1}^{N} \|\widetilde{\boldsymbol{\phi}}_{m,n}^{t-1}\|^2 \Bigg) \\
&= 3\delta^2\mu^2 \sum_{m=1}^{N-2}\sum_{i=m}^{N-1}(N-i) \Bigg(\|\boldsymbol{w}_m^{t-1} - \boldsymbol{w}_0^{t-1}\|^2 + \|\boldsymbol{w}_N^{t-2} - \boldsymbol{\phi}_{m,\boldsymbol{n}}^{t-1}\|^2 \\
&\qquad + \frac{1}{N}\sum_{n=1}^{N} \|\widetilde{\boldsymbol{\phi}}_{m,n}^{t-1}\|^2 \Bigg) \\
&\leq \frac{3}{2}\delta^2\mu^2 N^2 \sum_{m=1}^{N-2} \Bigg(\|\boldsymbol{w}_m^{t-1} - \boldsymbol{w}_0^{t-1}\|^2 + \|\boldsymbol{w}_N^{t-2} - \boldsymbol{\phi}_{m,\boldsymbol{n}}^{t-1}\|^2 \\
&\qquad + \frac{1}{N}\sum_{n=1}^{N} \|\widetilde{\boldsymbol{\phi}}_{m,n}^{t-1}\|^2 \Bigg) \\
&= \frac{3}{2}\delta^2\mu^2 N^2 \sum_{m=1}^{N-2} \Bigg(\|\boldsymbol{w}_m^{t-1} - \boldsymbol{w}_0^{t-1}\|^2 + \|\boldsymbol{w}_N^{t-2} - \boldsymbol{w}_{m+1}^{t-2}\|^2 \\
&\qquad + \frac{1}{N}\sum_{n=1}^{N} \|\widetilde{\boldsymbol{\phi}}_{m,n}^{t-1}\|^2 \Bigg) \\
&= \frac{3}{2}\delta^2\mu^2 N^2 \Bigg(\sum_{i=1}^{N-2} \|\boldsymbol{w}_i^{t-1} - \boldsymbol{w}_0^{t-1}\|^2 + \sum_{i=2}^{N-1} \|\boldsymbol{w}_N^{t-2} - \boldsymbol{w}_i^{t-2}\|^2 \\
&\qquad + \sum_{i=1}^{N-2}\frac{1}{N}\sum_{n=1}^{N} \|\widetilde{\boldsymbol{\phi}}_{i,n}^{t-1}\|^2 \Bigg) \\
&\leq \frac{3}{2}\delta^2\mu^2 N^2 \Bigg(\sum_{i=1}^{N-1} \|\boldsymbol{w}_i^{t-1} - \boldsymbol{w}_0^{t-1}\|^2 + \sum_{i=1}^{N-1} \|\boldsymbol{w}_N^{t-2} - \boldsymbol{w}_i^{t-2}\|^2 \\
&\qquad + \sum_{i=0}^{N-2}\frac{1}{N}\sum_{n=1}^{N} \|\widetilde{\boldsymbol{\phi}}_{i,n}^{t-1}\|^2 \Bigg)
\end{aligned} \tag{4.83}$$

The above result is similar to (4.76) with t replaced by $t-1$. Therefore, the same procedure can now be followed to arrive at (4.26).

4.E Proof of Theorem 6.1

To simplify the notation, we introduce the symbols:

$$a_t^2 \triangleq \frac{1}{N}\sum_{i=1}^{N-1}\mathbb{E}\,\|\boldsymbol{w}_i^t-\boldsymbol{w}_0^t\|^2,\;\; b_{t-1}^2 \triangleq \frac{1}{N}\sum_{i=1}^{N-1}\mathbb{E}\,\|\boldsymbol{w}_N^{t-1}-\boldsymbol{w}_i^{t-1}\|^2 \tag{4.84}$$

Then, the results of the previous three lemmas can be rewritten in the form:

$$\mathbb{E}\,\|\widetilde{\boldsymbol{w}}_0^{t+1}\|^2 \le (1-\mu\nu N)\,\mathbb{E}\|\widetilde{\boldsymbol{w}}_0^t\|^2 + 4\mu N\frac{\delta^2}{\nu}(a_t^2+b_{t-1}^2) \tag{4.85}$$

$$a_{t+1}^2 \le 5\delta^2\mu^2N^2(a_{t+1}^2+b_t^2) + 3\delta^2\mu^2N^2\mathbb{E}\,\|\widetilde{\boldsymbol{w}}_0^{t+1}\|^2 \tag{4.86}$$

$$b_t^2 \le 5\delta^2\mu^2N^2(a_t^2+b_{t-1}^2) + 3\delta^2\mu^2N^2\mathbb{E}\,\|\widetilde{\boldsymbol{w}}_0^{t}\|^2 \tag{4.87}$$

Before continuing, we first give a more strict condition on the step size than the one in (4.70):

$$\boxed{\text{Condition } \#1:\;\; \mu \le \frac{1}{4(\delta+\nu)N}} \tag{4.88}$$

Let γ denote an arbitrary positive scalar that we are free to choose. Multiplying relations (4.86) and (4.87) by γ and adding to (4.85) we obtain:

$$\begin{aligned}\mathbb{E}\,\|\widetilde{\boldsymbol{w}}_0^{t+1}\|^2 + \gamma(a_{t+1}^2+b_t^2) \le\;& (1-\mu\nu N)\,\mathbb{E}\,\|\widetilde{\boldsymbol{w}}_0^t\|^2 + 4\mu N\frac{\delta^2}{\nu}(a_t^2+b_{t-1}^2)\\ &+ 5\gamma\delta^2\mu^2N^2(a_{t+1}^2+b_t^2) + 3\gamma\delta^2\mu^2N^2\mathbb{E}\,\|\widetilde{\boldsymbol{w}}_0^{t+1}\|^2\\ &+ 5\gamma\delta^2\mu^2N^2(a_t^2+b_{t-1}^2) + 3\gamma\delta^2\mu^2N^2\mathbb{E}\,\|\widetilde{\boldsymbol{w}}_0^{t}\|^2\end{aligned} \tag{4.89}$$

which simplifies to

$$\begin{aligned}&(1-3\gamma\delta^2\mu^2N^2)\mathbb{E}\,\|\widetilde{\boldsymbol{w}}_0^{t+1}\|^2 + \gamma(1-5\delta^2\mu^2N^2)(a_{t+1}^2+b_t^2)\\ &\quad\le \left(1-\mu\nu N+3\gamma\delta^2\mu^2N^2\right)\mathbb{E}\,\|\widetilde{\boldsymbol{w}}_0^t\|^2 + \left(4\mu N\frac{\delta^2}{\nu}+5\gamma\delta^2\mu^2N^2\right)(a_t^2+b_{t-1}^2)\end{aligned} \tag{4.90}$$

Under the condition $1-3\gamma\delta^2\mu^2N^2>0$, which is equivalent to

$$\boxed{\text{Condition } \#2:\ \mu^2\gamma<\frac{1}{3\delta^2N^2}} \tag{4.91}$$

it holds that

$$\begin{aligned}\mathbb{E}\,\|\widetilde{\boldsymbol{w}}_0^{t+1}\|^2+\gamma\frac{1-5\delta^2\mu^2N^2}{1-3\gamma\delta^2\mu^2N^2}(a_{t+1}^2+b_t^2)\leq&\frac{1-\mu\nu N+3\gamma\delta^2\mu^2N^2}{1-3\gamma\delta^2\mu^2N^2}\mathbb{E}\,\|\widetilde{\boldsymbol{w}}_0^t\|^2\\&+\frac{4\mu N\frac{\delta^2}{\nu}+5\gamma\delta^2\mu^2N^2}{1-3\gamma\delta^2\mu^2N^2}(a_t^2+b_{t-1}^2)\\=&\frac{1-\mu\nu N+3\gamma\delta^2\mu^2N^2}{1-3\gamma\delta^2\mu^2N^2}\times\\&\left(\mathbb{E}\,\|\widetilde{\boldsymbol{w}}_0^t\|^2+\frac{4\mu N\frac{\delta^2}{\nu}+5\gamma\delta^2\mu^2N^2}{1-\mu\nu N+3\gamma\delta^2\mu^2N^2}(a_t^2+b_{t-1}^2)\right)\end{aligned} \tag{4.92}$$

This relation in turn implies that

$$\begin{aligned}&\mathbb{E}\,\|\widetilde{\boldsymbol{w}}_0^{t+1}\|^2+\gamma(1-5\delta^2\mu^2N^2)(a_{t+1}^2+b_t^2)\\&\leq\frac{1-\mu\nu N+3\gamma\delta^2\mu^2N^2}{1-3\gamma\delta^2\mu^2N^2}\times\left(\mathbb{E}\,\|\widetilde{\boldsymbol{w}}_0^t\|^2+\frac{4\mu N\frac{\delta^2}{\nu}+5\gamma\delta^2\mu^2N^2}{1-\mu\nu N+3\gamma\delta^2\mu^2N^2}(a_t^2+b_{t-1}^2)\right)\end{aligned} \tag{4.93}$$

We can again simplify the result by noting that

$$\begin{aligned}1-\mu\nu N+3\gamma\delta^2\mu^2N^2=&1-\mu\nu N/2-(\mu\nu N/2-3\gamma\delta^2\mu^2N^2)\\\leq&1-\mu\nu N/2\end{aligned} \tag{4.94}$$

where the inequality holds when $\mu\nu N/2-3\gamma\delta^2\mu^2N^2\geq0$, i.e.,

$$\boxed{\text{Condition } \#3:\ \mu\gamma\leq\frac{\nu}{6\delta^2N}} \tag{4.95}$$

In addition, we have the lower bound

$$1 - \mu\nu N + 3\gamma\delta^2\mu^2 N^2 \geq 1 - \mu\nu N \tag{4.96}$$

Using condition #1 from Eq. (4.88), we have

$$1 - \mu\nu N \geq 1 - \frac{\nu}{4(\delta+\nu)} \geq \frac{3}{4} \tag{4.97}$$

In a similar manner,

$$\begin{aligned} 4\mu N\frac{\delta^2}{\nu} + 5\gamma\delta^2\mu^2 N^2 &\leq 4\mu N\frac{\delta^2}{\nu} + \mu N\frac{\delta^2}{\nu} \\ &= 5\mu N\frac{\delta^2}{\nu} \end{aligned} \tag{4.98}$$

where the last inequality holds when $\mu\gamma \leq \frac{1}{5\nu N}$, which is always valid under condition #3 in Eq. (4.95) since the latter implies that $\mu\gamma \leq \frac{1}{6\delta N}$. Substituting (4.94), (4.97), and (4.98) into (4.93), we find that

$$\begin{aligned} &\mathbb{E}\,\|\widetilde{\boldsymbol{w}}_0^{t+1}\|^2 + \gamma(1 - 5\delta^2\mu^2N^2)(a_{t+1}^2 + b_t^2) \\ &\quad \leq \frac{1-\mu\nu N/2}{1-3\gamma\delta^2\mu^2N^2}\left(\mathbb{E}\,\|\widetilde{\boldsymbol{w}}_0^t\|^2 + \frac{4}{3}\cdot 5\mu N\frac{\delta^2}{\nu}(a_t^2 + b_{t-1}^2)\right) \end{aligned} \tag{4.99}$$

Under condition #1 in Eq. (4.88), we have

$$1 - 5\delta^2\mu^2N^2 \geq 1 - 5\delta^2N^2\frac{1}{16(\delta+\nu)^2N^2} \geq 1 - \frac{5}{16} > \frac{5}{8} \tag{4.100}$$

and, hence,

$$\begin{aligned} \mathbb{E}\,\|\widetilde{\boldsymbol{w}}_0^{t+1}\|^2 + \frac{5}{8}\gamma(a_{t+1}^2 + b_t^2) &\leq \frac{1-\mu\nu N/2}{1-3\gamma\delta^2\mu^2N^2}\left(\mathbb{E}\,\|\widetilde{\boldsymbol{w}}_0^t\|^2 + \frac{20}{3}\mu N\frac{\delta^2}{\nu}(a_t^2 + b_{t-1}^2)\right) \\ &\leq \frac{1-\mu\nu N/2}{1-3\gamma\delta^2\mu^2N^2}\left(\mathbb{E}\,\|\widetilde{\boldsymbol{w}}_0^t\|^2 + 5\mu N\frac{\delta^2}{\nu}(a_t^2 + b_{t-1}^2)\right) \end{aligned} \tag{4.101}$$

where the last inequality is unnecessary but is introduced for convenience. Recall that we

are free to choose γ, so assume we choose it to satisfy

$$\frac{5}{8}\gamma = 5\mu N\frac{\delta^2}{\nu} \implies \gamma = 8\mu N\frac{\delta^2}{\nu} \tag{4.102}$$

It then follows that:

$$\begin{aligned}\mathbb{E}\,\|\widetilde{\boldsymbol{w}}_0^{t+1}\|^2 + \frac{5}{8}\gamma(a_{t+1}^2 + b_t^2) &\leq \frac{1-\mu\nu N/2}{1-3\gamma\delta^2\mu^2N^2}\left(\mathbb{E}\,\|\widetilde{\boldsymbol{w}}_0^{t}\|^2 + 5\gamma(a_t^2 + b_{t-1}^2)\right) \\ &\triangleq \alpha\left(\mathbb{E}\,\|\widetilde{\boldsymbol{w}}_0^{t}\|^2 + 5\gamma(a_t^2 + b_{t-1}^2)\right)\end{aligned} \tag{4.103}$$

where we introduced the positive parameter

$$\alpha \triangleq \frac{1-\mu\nu N/2}{1-3\gamma\delta^2\mu^2N^2} \tag{4.104}$$

This parameter controls the speed of convergence. It will hold that $\alpha < 1$ when

$$\frac{1-\mu\nu N/2}{1-3\gamma\delta^2\mu^2N^2} = \frac{1-\mu\nu N/2}{1-24\delta^4\mu^3N^3/\nu} < 1 \iff \mu < \sqrt{\frac{1}{48}}\frac{\nu}{\delta^2 N} \tag{4.105}$$

Let us now re-examine conditions #1 through #3, along with (4.105), when γ is chosen according to (4.102). In this case, conditions #1 through #3 become

$$\boxed{\mu \leq \frac{1}{2(\delta+\nu)N},\;\; \mu < \frac{1}{3\delta N}\left(\frac{\nu}{\delta}\right)^{1/3},\;\; \mu \leq \sqrt{\frac{1}{48}}\frac{\nu}{\delta^2 N}} \tag{4.106}$$

All three conditions and condition (4.105) can be satisfied by the following single sufficient bound on the step-size parameter (since $7^2 > 48$):

$$\boxed{\mu \leq \frac{\nu}{7\delta^2 N}} \tag{4.107}$$

4.F Proof of Lemma 4.5

Subtracting $w^\star$ from both sides of (4.38), we obtain:

$$\begin{aligned}\widetilde{\boldsymbol{w}}_0^{t+1} =&\widetilde{\boldsymbol{w}}_0^t + \mu N \nabla J(\boldsymbol{w}_0^t) + \mu \sum_{i=0}^{N-1} \big[\nabla Q(\boldsymbol{w}_i^t; x_{\boldsymbol{n}}) - \nabla Q(\boldsymbol{w}_0^t; x_{\boldsymbol{n}})\big] \\ &- \mu \sum_{i=0}^{N-1} \big[\nabla Q(\boldsymbol{w}_0^t; x_{\boldsymbol{n}'}) - \nabla Q(\boldsymbol{w}_i^{t-1}; x_{\boldsymbol{n}'})\big] \end{aligned} \tag{4.108}$$

Then, taking the squared norm and applying Jensen's inequality, we establish the first recursion for any $t \in (0,1)$:

$$\begin{aligned}\|\widetilde{\boldsymbol{w}}_0^{t+1}\|^2 \le& \frac{1}{t}\left\|\widetilde{\boldsymbol{w}}_0^t + \mu N \nabla J(\boldsymbol{w}_0^t)\right\|^2 \\ &+ \frac{2\mu^2}{1-t}\Big\| \sum_{i=0}^{N-1} \big[\nabla Q(\boldsymbol{w}_i^t; x_{\boldsymbol{n}}) - \nabla Q(\boldsymbol{w}_0^t; x_{\boldsymbol{n}})\big] \Big\|^2 \\ &+ \frac{2\mu^2}{1-t}\Big\| \sum_{i=0}^{N-1} \big[\nabla Q(\boldsymbol{w}_0^t; x_{\boldsymbol{n}'}) - \nabla Q(\boldsymbol{w}_i^{t-1}; x_{\boldsymbol{n}'})\big] \Big\|^2 \\ \le& \frac{1}{t}\|\widetilde{\boldsymbol{w}}_0^t + \mu N \nabla J(\boldsymbol{w}_0^t)\|^2 + \frac{2\mu^2\delta^2 N}{1-t}\sum_{i=0}^{N-1}\|\boldsymbol{w}_i^t - \boldsymbol{w}_0^t\|^2 \\ &+ \frac{2\mu^2\delta^2 N}{1-t}\sum_{i=0}^{N-1}\|\boldsymbol{w}_N^{t-1} - \boldsymbol{w}_i^{t-1}\|^2 \end{aligned} \tag{4.109}$$

Using an argument similar to (4.71) and letting $t = 1 - \mu N\nu$, assuming $\mu \le 2/(N(\delta+\nu))$, we obtain:

$$\|\widetilde{\boldsymbol{w}}_0^{t+1}\|^2 \le (1-\mu\nu N)\,\|\widetilde{\boldsymbol{w}}_0^t\|^2 + \frac{2\mu\delta^2}{\nu}\left(\sum_{i=0}^{N-1}\|\boldsymbol{w}_i^t - \boldsymbol{w}_0^t\|^2 + \sum_{i=0}^{N-1}\|\boldsymbol{w}_N^{t-1} - \boldsymbol{w}_i^{t-1}\|^2\right) \tag{4.110}$$

Taking the expectation of both sides, we establish (4.40). The forward inner difference recursion can be obtain by following the same procedure as in (4.75):

$$\|\boldsymbol{w}_i^t - \boldsymbol{w}_0^t\|^2 \le i\sum_{m=0}^{i-1}\|\boldsymbol{w}_{m+1}^t - \boldsymbol{w}_m^t\|^2$$

$$
\begin{aligned}
&\overset{(4.39)}{\leq} 3\mu^2\delta^2 i \sum_{m=0}^{i-1}\left(\|\boldsymbol{w}_m^t-\boldsymbol{w}_0^t\|^2+\frac{1}{N}\sum_{n'=0}^{N-1}\|\boldsymbol{w}_{n'}^{t-1}-\boldsymbol{w}_N^{t-1}\|^2+\|\widetilde{\boldsymbol{w}}_0^t\|^2\right)\\
&\quad= 3\mu^2\delta^2 i\sum_{m=0}^{i-1}\|\boldsymbol{w}_m^t-\boldsymbol{w}_0^t\|^2\\
&\qquad+3\mu^2\delta^2 i^2\left(\frac{1}{N}\sum_{n'=0}^{N-1}\|\boldsymbol{w}_{n'}^{t-1}-\boldsymbol{w}_N^{t-1}\|^2+\|\widetilde{\boldsymbol{w}}_0^t\|^2\right) \qquad (4.111)
\end{aligned}
$$

Summing over i, we have

$$
\begin{aligned}
\sum_{i=0}^{N-1}\|\boldsymbol{w}_i^t-\boldsymbol{w}_0^t\|^2 &\leq 3\mu^2\delta^2\left(\sum_{i=0}^{N-1} i\sum_{m=0}^{i-1}\|\boldsymbol{w}_m^t-\boldsymbol{w}_0^t\|^2\right.\\
&\qquad\left.+\sum_{i=0}^{N-1} i^2\left(\frac{1}{N}\sum_{n'=0}^{N-1}\|\boldsymbol{w}_{n'}^{t-1}-\boldsymbol{w}_N^{t-1}\|^2+\|\widetilde{\boldsymbol{w}}_0^t\|^2\right)\right)\\
&= 3\mu^2\delta^2\left(\sum_{m=0}^{N-1}\sum_{i=m+1}^{N-1} i\|\boldsymbol{w}_m^t-\boldsymbol{w}_0^t\|^2\right.\\
&\qquad\left.+\sum_{i=0}^{N-1} i^2\left(\frac{1}{N}\sum_{n'=0}^{N-1}\|\boldsymbol{w}_{n'}^{t-1}-\boldsymbol{w}_N^{t-1}\|^2+\|\widetilde{\boldsymbol{w}}_0^t\|^2\right)\right)\\
&\overset{(a)}{\leq} 3\mu^2\delta^2N^2\sum_{m=0}^{N-1}\|\boldsymbol{w}_m^t-\boldsymbol{w}_0^t\|^2\\
&\qquad+\mu^2\delta^2N^2\left(\sum_{n'=0}^{N-1}\|\boldsymbol{w}_N^{t-1}-\boldsymbol{w}_{n'}^{t-1}\|^2+N\|\widetilde{\boldsymbol{w}}_0^t\|^2\right)\\
&= 3\mu^2\delta^2N^2\sum_{i=0}^{N-1}\|\boldsymbol{w}_i^t-\boldsymbol{w}_0^t\|^2\\
&\qquad+\mu^2\delta^2N^2\left(\sum_{i=0}^{N-1}\|\boldsymbol{w}_N^{t-1}-\boldsymbol{w}_i^{t-1}\|^2+N\|\widetilde{\boldsymbol{w}}_0^t\|^2\right) \qquad (4.112)
\end{aligned}
$$

where step (a) is because:

$$
\sum_{m+1}^{N-1} i\leq N^2,\quad \sum_{i=0}^{N-1} i^2=\frac{(N-1)N(2N-1)}{6}\leq\frac{N^3}{3} \qquad (4.113)
$$

Lastly, we establish the backwards inner difference term using the same argument as in

(4.82):

$$\begin{aligned}
\|\boldsymbol{w}_N^t - \boldsymbol{w}_i^t\|^2 &= \|\boldsymbol{w}_N^t - \boldsymbol{w}_{N-1}^t + \boldsymbol{w}_{N-1}^t - \cdots + \boldsymbol{w}_{i+1}^t - \boldsymbol{w}_i^t\|^2 \\
&\leq (N-i)\sum_{m=i}^{N-1} \|\boldsymbol{w}_{m+1}^t - \boldsymbol{w}_m^t\|^2 \\
&\leq 3\mu^2\delta^2(N-i)\sum_{m=i}^{N-1}\left(\|\boldsymbol{w}_m^t - \boldsymbol{w}_0^t\|^2 + \frac{1}{N}\sum_{n'=0}^{N-1}\|\boldsymbol{w}_{n'}^{t-1} - \boldsymbol{w}_N^{t-1}\|^2 + \|\widetilde{\boldsymbol{w}}_0^t\|^2\right) \\
&\leq 3\mu^2\delta^2(N-i)\sum_{m=i}^{N-1}\|\boldsymbol{w}_m^t - \boldsymbol{w}_0^t\|^2 \\
&\quad + \frac{3\mu^2\delta^2(N-i)^2}{N}\sum_{n'=0}^{N-1}\|\boldsymbol{w}_{n'}^{t-1} - \boldsymbol{w}_N^{t-1}\|^2 + 3\mu^2\delta^2(N-i)^2\|\widetilde{\boldsymbol{w}}_0^t\|^2 \qquad (4.114)
\end{aligned}$$

Observing that this backward term is summing from 0 to $N-1$, rather than from 1 to $N-1$ as in SAGA with RR, we have

$$\begin{aligned}
\sum_{i=0}^{N-1}\|\boldsymbol{w}_N^t - \boldsymbol{w}_i^t\|^2 &\leq 3\mu^2\delta^2\sum_{i=0}^{N-1}(N-i)\sum_{m=i}^{N-1}\|\boldsymbol{w}_m^t - \boldsymbol{w}_0^t\|^2 \\
&\quad + 3\mu^2\delta^2\sum_{i=0}^{N-1}(N-i)^2\left(\frac{1}{N}\sum_{n'=0}^{N-1}\|\boldsymbol{w}_{n'}^{t-1} - \boldsymbol{w}_N^{t-1}\|^2 + \|\widetilde{\boldsymbol{w}}_0^t\|^2\right) \\
&= 3\mu^2\delta^2\sum_{m=0}^{N-1}\sum_{i=0}^{m}(N-i)\|\boldsymbol{w}_m^t - \boldsymbol{w}_0^t\|^2 \\
&\quad + 3\mu^2\delta^2\frac{N(N+1)(2N+1)}{6}\left(\frac{1}{N}\sum_{n'=0}^{N-1}\|\boldsymbol{w}_{n'}^{t-1} - \boldsymbol{w}_N^{t-1}\|^2 + \|\widetilde{\boldsymbol{w}}_0^t\|^2\right) \\
&\leq 3\mu^2\delta^2N^2\sum_{m=0}^{N-1}\|\boldsymbol{w}_m^t - \boldsymbol{w}_0^t\|^2 \\
&\quad + 3\mu^2\delta^2N^2\left(\sum_{n'=0}^{N-1}\|\boldsymbol{w}_{n'}^{t-1} - \boldsymbol{w}_N^{t-1}\|^2 + N\|\widetilde{\boldsymbol{w}}_0^t\|^2\right) \\
&= 3\mu^2\delta^2N^2\sum_{i=0}^{N-1}\|\boldsymbol{w}_i^t - \boldsymbol{w}_0^t\|^2 \\
&\quad + 3\mu^2\delta^2N^2\left(\sum_{i=0}^{N-1}\|\boldsymbol{w}_i^{t-1} - \boldsymbol{w}_N^{t-1}\|^2 + N\|\widetilde{\boldsymbol{w}}_0^t\|^2\right) \qquad (4.115)
\end{aligned}$$

where in the last inequality we used the fact that

$$\frac{N(N+1)(2N+1)}{6} \leq N^3, \quad \forall N \tag{4.116}$$

4.G Proof of Theorem 7.2

We let

$$a_t \triangleq \frac{1}{N}\sum_{i=0}^{N-1}\mathbb{E}\,\|\boldsymbol{w}_i^t - \boldsymbol{w}_0^t\|^2, \quad b_t \triangleq \frac{1}{N}\sum_{i=0}^{N-1}\mathbb{E}\,\|\boldsymbol{w}_N^t - \boldsymbol{w}_i^t\|^2 \tag{4.117}$$

The recursions available so far for AVRG are:

$$\mathbb{E}\,\|\widetilde{\boldsymbol{w}}^{t+1}\|^2 \leq (1-\mu\nu N)\,\mathbb{E}\,\|\widetilde{\boldsymbol{w}}_0^t\|^2 + \frac{2\mu\delta^2 N}{\nu}\,(a_t + b_{t-1}) \tag{4.118}$$

$$a_{t+1} \leq 3\mu^2\delta^2N^2a_{t+1} + \mu^2\delta^2N^2b_t + \mu^2\delta^2N^2\mathbb{E}\,\|\widetilde{\boldsymbol{w}}_0^{t+1}\|^2 \tag{4.119}$$

$$b_t \leq 3\mu^2\delta^2N^2(a_t + b_{t-1}) + 3\mu^2\delta^2N^2\mathbb{E}\,\|\widetilde{\boldsymbol{w}}_0^t\|^2 \tag{4.120}$$

which have exactly the same form as recursions (4.85)—(4.87) except for the coefficients. To simplify the argument, we can replace (4.119) by:

$$a_{t+1} \leq 3\mu^2\delta^2N^2(a_{t+1} + b_t) + \mu^2\delta^2N^2\mathbb{E}\,\|\widetilde{\boldsymbol{w}}_0^{t+1}\|^2 \tag{4.121}$$

Similarly, we first give a more strict condition for the step size than the one in (4.70):

$$\boxed{\text{Condition } \#1: \ \mu \leq \frac{1}{2(\delta+\nu)N}} \tag{4.122}$$

Under the condition $1 - \gamma\delta^2\mu^2N^2 > 0$, which is equivalent to

$$\boxed{\text{Condition } \#2: \ \mu^2\gamma < \frac{1}{\delta^2N^2}} \tag{4.123}$$

it further holds that

$$\begin{aligned}
&\mathbb{E}\,\|\widetilde{\boldsymbol{w}}^{t+1}\|^2 + \gamma(1-3\mu^2\delta^2N^2)(a_{t+1}+b_t) \\
&\leq \frac{1-\mu\nu N+3\gamma\mu^2\delta^2N^2}{1-\gamma\delta^2\mu^2N^2} \times \left(\mathbb{E}\,\|\widetilde{\boldsymbol{w}}_0^t\|^2 + \frac{\frac{2\mu\delta^2N}{\nu}+3\gamma\mu^2\delta^2N^2}{1-\mu\nu N+3\gamma\mu^2\delta^2N^2}\,(a_t+b_{t-1})\right)
\end{aligned} \tag{4.124}$$

Note that the numerator $1-\mu\nu N+3\gamma\mu^2\delta^2N^2$ is the same as SAGA in (4.94). Thus, under condition:

$$\boxed{\textit{Condition \#3}:\ \mu\gamma \leq \frac{\nu}{6\delta^2N}} \tag{4.125}$$

we have:

$$\frac{3}{4} \leq\ 1-\mu\nu N+3\gamma\delta^2\mu^2N^2 \leq 1-\mu\nu N \tag{4.126}$$

Lastly, we can verify that

$$\begin{aligned}
\frac{2\mu\delta^2N}{\nu}+3\gamma\mu^2\delta^2N^2 &\leq \frac{2\mu\delta^2N}{\nu}+\frac{\mu\delta^2N}{\nu} \\
&\leq \frac{3\mu\delta^2N}{\nu}
\end{aligned} \tag{4.127}$$

where the last inequality holds when $\mu\gamma \leq \frac{1}{3\nu N}$, which is always valid under condition #3. Now, collecting the results, we have

$$\begin{aligned}
\mathbb{E}\,\|\widetilde{\boldsymbol{w}}^{t+1}\|^2 + \gamma\frac{13}{16}(a_{t+1}+b_t) &\leq \frac{1-\mu\nu N/2}{1-\gamma\mu^2\delta^2N^2}\left(\mathbb{E}\,\|\widetilde{\boldsymbol{w}}_0^t\|^2 + 3\times\frac{4}{3}\mu N\frac{\delta^2}{\nu}\,(a_t+b_{t-1})\right) \\
&\leq \frac{1-\mu\nu N/2}{1-\gamma\mu^2\delta^2N^2}\left(\mathbb{E}\,\|\widetilde{\boldsymbol{w}}_0^t\|^2 + \frac{65}{16}\mu N\frac{\delta^2}{\nu}\,(a_t+b_{t-1})\right)
\end{aligned} \tag{4.128}$$

Assume we choose γ such that

$$\gamma\frac{13}{16} = \frac{65}{16}\mu N\frac{\delta^2}{\nu} \implies \gamma = 5\mu N\frac{\delta^2}{\nu} \tag{4.129}$$

It then follows that:

$$\begin{aligned}\mathbb{E}\,\|\widetilde{\boldsymbol{w}}_0^{t+1}\|^2+\frac{13}{16}\gamma(a_{t+1}^2+b_t^2) &\leq \frac{1-\mu\nu N/2}{1-3\gamma\delta^2\mu^2N^2}\left(\mathbb{E}\,\|\widetilde{\boldsymbol{w}}_0^{t}\|^2+\frac{13}{16}\gamma(a_t^2+b_{t-1}^2)\right)\\ &\triangleq\ \alpha\left(\mathbb{E}\,\|\widetilde{\boldsymbol{w}}_0^{t}\|^2+\frac{13}{16}\gamma(a_t^2+b_{t-1}^2)\right)\end{aligned}\tag{4.130}$$

where we introduced the positive parameter

$$\alpha\ \triangleq\ \frac{1-\mu\nu N/2}{1-3\gamma\delta^2\mu^2N^2}\tag{4.131}$$

This parameter satisfies $\alpha<1$ for

$$\frac{1-\mu\nu N/2}{1-3\gamma\delta^2\mu^2N^2}=\frac{1-\mu\nu N/2}{1-15\delta^4\mu^3N^3/\nu}<1\iff\mu<\sqrt{\frac{1}{30}}\frac{\nu}{\delta^2N}\tag{4.132}$$

We re-examine conditions #1–#3 when γ is chosen according to (4.129). In this case, these conditions become

$$Conditions\ 1-3:\mu\leq\frac{1}{4(\delta+\nu)N},\ \mu^3<\frac{\nu}{5\delta^4N^3},\ \ \mu^2\leq\frac{\nu^2}{30\delta^4N^2}\tag{4.133}$$

which can be met by:

$$\boxed{\mu\leq\frac{1}{4(\delta+\nu)N},\ \ \mu<\frac{1}{2\delta N}\left(\frac{\nu}{\delta}\right)^{1/3},\ \ \mu\leq\frac{\nu}{6\delta^2N}}\tag{4.134}$$

All these three conditions and the condition for $\alpha<1$ can be satisfied by the following single sufficient bound on the step-size parameter:

$$\boxed{\mu\leq\frac{\nu}{6\delta^2N}}\tag{4.135}$$

CHAPTER 5

Diffusion Gradient Boosting for Networked Learning

Using duality arguments from optimization theory, this chapter develops an effective distributed gradient boosting strategy for inference and classification by networked clusters of learners. By sharing local dual variables with their immediate neighbors through a diffusion learning protocol, the clusters are able to match the performance of centralized boosting solutions even when the individual clusters only have access to partial information about the feature space. The performance of the resulting fully-distributed procedure is illustrated on two data sets with superior results in comparison to a centralized boosting solution.

5.1 Introduction and Motivation

In statistics and machine learning, ensemble learning is a formidable technique that is able to aggregate the recommendations of weak classifiers into a more powerful classification structure with enhanced predictive abilities [9,10,85,86]. One prominent example is the AdaBoost algorithm [85, 87, 88], which has achieved great prominence due to its efficient implementation structure, strong performance, and its ability to limit over-fitting [9]. It also satisfies a useful optimality property in that it can be derived from the minimization of exponential risk functions [89]. This connection between exponential risks and AdaBoost has motivated useful generalizations of boosting solutions using other choices for the risk function. Two of the main generalizations introduced in [90] and [91] are the Gradient Boosting Machine and the AnyBoost solution. These useful works helped solidify the connection between boosting techniques and gradient-descent methods from an optimization theory perspective.

In this chapter, we exploit this connection more broadly and consider *distributed* im-

plementations. Specifically, we show how to develop *cooperative* boosting strategies by exploiting strong duality arguments from optimization theory [15, 46] and powerful diffusion strategies from distributed learning [11, 28]. We examine the important case in which the weak classifiers are not co-located but are dispersed, either geographically over space or by design through the intentional partitioning of the classifier set. The main challenge is to devise suitable cooperation and communication protocols that are able to bring forth these enhancements. The scenario where the classifier set is partitioned into smaller sets arises when it is desired to reduce the possibility of overfitting or when the feature space itself can be partitioned into lower-dimensional subspaces. Regardless of the motivation for splitting the classifier set, we assume that the classifier set is partitioned into smaller groups, where the elements in each group may only have access to lower dimensional subspaces of the feature space. The groups are also networked by an outer topology – see Fig. 6.1 and 5.1 further ahead. The objective is to endow the dispersed groups of classifiers, through localized cooperation, with a distributed learning mechanism that ensures that the quality of their predictions is as close to what would result from a centralized solution with access to the entire feature space.

One earlier approach to distributed boosting is studied in [92–94]. It is based on learning from subsets of the training data and then combining the weak classifiers through an aggregation procedure. This formulation is different from the approach pursued in this work, which is fully decentralized and does not involve fusing the information from across all classifiers. A second example of a distributed implementation is the Ivote procedure [95] and its distributed version DIvote [96, 97]. These procedures, however, do not rely on boosting and their theoretical limits of performance have not been analyzed as closely as AdaBoost. While these various methods work well in some circumstances, they may still suffer from over-fitting or get trapped at local minima. In comparison, our main objective in this work is to devise truly distributed boosting solutions with performance guarantees by relying on strong duality arguments [44, 46] and the theory of diffusion adaptation [11, 28].

5.2 Gradient Boosting Algorithm

In order to prepare for the derivation of the distributed strategy, we briefly review the well-known gradient boosting technique [90, 91, 98] in the context of supervised machine learning problems. Thus, consider a collection of N data pairs:

$$\mathcal{D} = \Big\{\{h_1, \gamma(1)\}, \{h_2, \gamma(2)\}, \cdots, \{h_N, \gamma(N)\}\Big\} \tag{5.1}$$

where $h_n \in \mathbb{R}^M$ are feature vectors and $\gamma(n)$ represent the class variable. In this chapter, we assume that there are two classes and $\gamma(n) \in \{\pm 1\}$. A generic classifier, denoted by $c(h)$, is a transformation that maps a feature vector h into a class value, $\gamma(h)$. Assume we have a collection of L weak classifiers:

$$\mathcal{C} = \{c_1(h), c_2(h), \cdots, c_L(h)\} \quad (L \text{ can be larger than the data size, } N) \tag{5.2}$$

We would like to select combination coefficients $\{\alpha(\ell)\}$ to construct a prediction for the class variable $\gamma(h)$ by combining the above learners into a more powerful classifier:

$$\widehat{\gamma}(h) \;\triangleq\; \sum_{\ell=1}^{L} \alpha(\ell) c_\ell(h) \tag{5.3}$$

The coefficients $\{\alpha(\ell)\}$ are determined by minimizing a surrogate risk of the following form

$$J_{emp}(\widehat{\gamma}) \;\triangleq\; \frac{1}{N} \sum_{n=1}^{N} Q\Big(\widehat{\gamma}(n); \gamma(n)\Big) \tag{5.4}$$

where the symbol $Q(\cdot)$ denotes a generic loss function, assumed convex and first-order differentiable. Some popular choices for $Q(\cdot)$ include exponential loss, quadratic loss, hinge loss, and logistic loss [9, 10, 99]. Gradient boosting provides a solution technique by applying a greedy strategy to the minimization of (5.4) [9, 90, 98], where one coefficient $\alpha(\ell)$ and one classifier $c_\ell(h)$ are selected at a time. Specifically, assume that by the end of iteration $t-1$, we have already succedded in identifying classifiers $\{c_1^o(h), \ldots, c_{t-1}^o(h)\}$ and weights

Table 5.1: Listing of gradient boosting algorithm

Gradient boosting algorithm [90]

Initialization: choose $\widehat{\gamma}^{(0)}(n),\ n = 1, 2, \ldots, N$
Repeat $t = 1, 2, \ldots, T$:

$$g_t(n) = -\left.\frac{\partial Q\big(\widehat{\gamma}(n)\,;\,\gamma(n)\big)}{\partial\widehat{\gamma}(n)}\right|_{\widehat{\gamma}(n)=\widehat{\gamma}^{(t-1)}(n)} \tag{5.7}$$

$$\{\ell^o, \beta^o\} = \underset{\{\ell,\beta\}}{\arg\min} \sum_{n=1}^{N} \left(g_t(n) - \beta c_\ell(h_n)\right)^2 \implies set\ c_t^o(h) = c_{\ell^o}(h) \tag{5.8}$$

$$\alpha^o(t) = \underset{\alpha}{\arg\min} \sum_{n=1}^{N} Q\left(\widehat{\gamma}^{(t-1)}(h_n) + \alpha c_t^o(h_n)\,;\,\gamma(n)\right) \tag{5.9}$$

$$\widehat{\gamma}^{(t)}(h) = \widehat{\gamma}^{(t-1)}(h) + \alpha^o(t)c_t^o(h) \tag{5.10}$$

End

$\{\alpha^o(1), \ldots, \alpha^o(t-1)\}$. Gradient boosting selects the next classifier, $c_t(h)$, and its associated weight, $\alpha(t)$, in order to enlarge enlarge the aggregate prediction from iteration $t-1$ as follows:

$$\widehat{\gamma}^{(t)}(h) = \sum_{s=1}^{t-1} \alpha^o(s)c_s^o(h) + \alpha(t)c_t(h) = \widehat{\gamma}^{(t-1)}(h) + \alpha(t)c_t(h) \tag{5.5}$$

This expression involves correcting the previous construction by adding the term $\alpha(t)c_t(h)$. In order to determine the optimal choices $\{c_t^o(h), \alpha^o(t)\}$, the algorithm evaluates the negative gradient of the empirical risk (5.4):

$$\text{negative gradient for stage } t \triangleq \begin{bmatrix} g_t(1) \\ \vdots \\ g_t(N) \end{bmatrix},\ \ g_t(n) \triangleq -\left.\frac{\partial Q(\widehat{\gamma}(n)\,;\,\gamma(n))}{\partial\widehat{\gamma}(n)}\right|_{\widehat{\gamma}(n)=\widehat{\gamma}^{(t-1)}(n)} \tag{5.6}$$

and selects the classifier and its weight optimally as summarized below: the index ℓ^o is selected through (5.8) and the coefficient $\alpha^o(t)$ through (5.9).

5.3 Diffusion Gradient Boosting

We now move on to motivate and derive an effective decentralized strategy for boosting assuming distributed classifier groups and partial information at the groups. We first describe a formulation that involves the *centralized* fusion of predictions from a collection of dispersed learning groups. We subsequently apply duality arguments from optimization theory to show that this can be transformed into a distributed implementation that relies solely on local interactions among the groups.

5.3.1 Networked Groups and Partitioning Model

We consider a scenario in which the L classifiers happen to be divided into K groupings. We index the groups by the letter k, with $k = 1, 2, \ldots, K$. We denote the classifiers that are available in group k:

$$\mathcal{C}_k = \{c_{k,1}(h), c_{k,2}(h), \ldots, c_{k,L_k}(h)\}, \;\; (L_k \text{ classifiers at group or agent } k) \tag{5.11}$$

Note that we are attaching a subscript k to the classifiers to indicate that these are the ones used by group k. We also allow classifiers to be repeated across groups. Moreover, for further generality, we assume that each group k may have access to only a *subset* of the feature space for its classification decisions. This situation is common. For example, weak classifiers are often chosen as shallow decision trees, or simply stumps [9, 100]. When a number of these weak classifiers is present at a particular group k, then this group will be relying on a subset of the feature entries. To reflect this fact, for any of the training vectors $h_n \in \mathbb{R}^M$, we shall adopt the notation $h_{k,n} \in \mathbb{R}^{M_k}$ to refer to the subset of the feature vector h_n that is used by group k. Again, we allow for feature entries to be repeated across groups. Accordingly, when we write, for example, $c_{k,1}(h_{k,n})$, this notation is meant to refer to the classifier $c_1(\cdot)$ that is present in group k and which operates on the sub-features of h_n that are included in $h_{k,n}$.

We further assume that there is a graph structure that ties the groups together — see

Fig. 5.1 so that if groups k and k' are connected by some edge, then this means that these groups can exchange information over this edge. A non-negative scalar $a_{k',k}$ is assigned to the edge connecting k' to k. These scalars are convex combination coefficients and satisfy:

$$a_{k,k'} = a_{k',k} \geq 0, \quad \sum_{k' \in \mathcal{N}_k} a_{k',k} = 1, \tag{5.12}$$

where $\mathcal{N}_k$ denotes the set of neighbors of group k; these are the groups that are directly connected to k by edges. If we collect the scalars $\{a_{k',k}\}$ into a $K \times K$ matrix $A = [a_{k',k}]$, then the above property implies that A is a symmetric matrix, and each column and each row of A adds up to one. We say that A is a doubly-stochastic matrix. There are many possible choices for such doubly-stochastic matrices. One popular choice is the Metropolis rule [28].

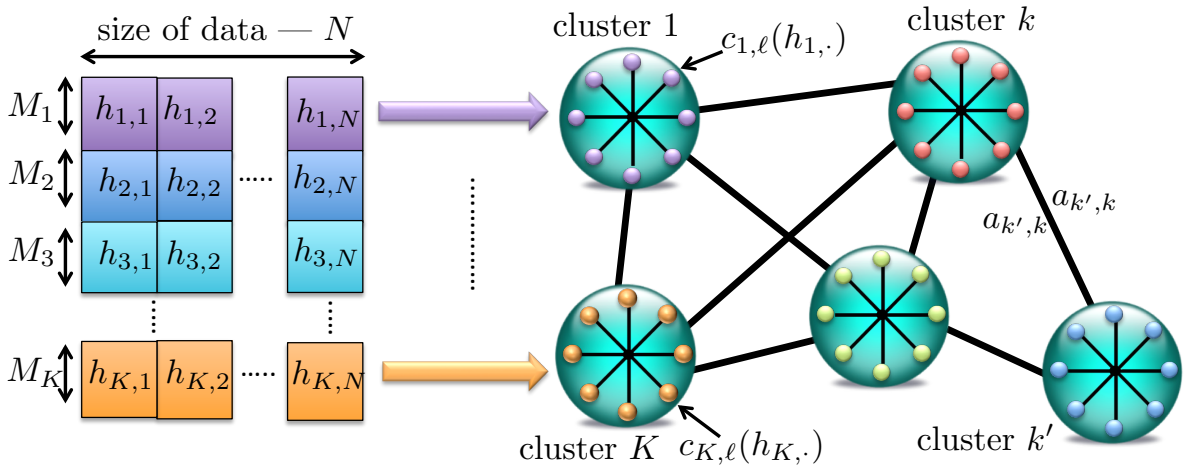

Figure 5.1: Partitioning of the feature space and network topology.

5.3.2 Centralized Processing at the Groups

In principle, each group k can run its own gradient boosting procedure and, after $T \leq L_k$ iterations, come up with its own prediction, one that is based on the classifiers in $\mathcal{C}_k$. More broadly, a centralized solution would seek to determine a global prediction function, $\widehat{\gamma}(h)$,that

combines all group classifiers in the following manner:

$$\widehat{\gamma}^{(T)}(h) \;\triangleq\; \sum_{k=1}^{K}\sum_{t=1}^{T}\alpha_k(t)c_{k,t}(h_{k,\cdot}) \tag{5.13}$$

in terms of some coefficients $\{\alpha_k(t)\}$ that need to be determined. Observe that we are now attaching a subscript k to coefficients arising from group k. The main difference in the derivation that follows in relation to the gradient boosting derivation from the previous section is that now we need to select a total of K classifiers, one for each group, for each iteration t, along with their corresponding weights. That is, every stage t now involves determining K pairs $\{c^o_{k,t}(h_{k,\cdot}), \alpha^o_k(t)\}$ for $k = 1, 2, \ldots, K$.

Assume that by the end of iteration $t-1$, each group k has already identified the optimal classifiers $\{c^o_{k,1}(h),\, c^o_{k,2}(h), \ldots,\, c^o_{k,t-1}(h)\}$ and their combination weights $\{\alpha^o_k(1),\, \alpha^o_k(2), \ldots, \alpha^o_k(t-1)\}$. Next, we would like to select the next K classifiers, denoted generically by $\{c_{k,t}(h)\}$, and their associated weights $\{\alpha_k(t)\}$, for $k = 1, 2, \ldots, K$, in order to enlarge the aggregate prediction from stage $t-1$ by adding to it a term of the following form:

$$\widehat{\gamma}^{(t)}(h) = \sum_{k=1}^{K}\sum_{s=1}^{t-1}\alpha^o_k(s)c^o_{k,s}(h_{k,\cdot}) + \sum_{k=1}^{K}\alpha_k(t)c_{k,t}(h_{k,\cdot}) = \widehat{\gamma}^{(t-1)}(h) + \sum_{k=1}^{K}\alpha_k(t)c_{k,t}(h_{k,\cdot}) \tag{5.14}$$

Observe that the aggregate update now involves the sum of K weak classifiers: one from each group k. In order to determine the optimal choices $\{c^o_{k,t}(h), \alpha^o_k(t)\}$ for $k = 1, 2, \ldots, K$, we evaluate the negative gradient of this empirical risk:

$$\text{negative gradient by group } k \;\triangleq\; \begin{bmatrix} g_{k,t}(1) \\ \vdots \\ g_{k,t}(N) \end{bmatrix}, \quad g_{k,t}(n) \triangleq -\frac{\partial Q\big(\widehat{\gamma}(n)\,;\,\gamma(n)\big)}{\partial\widehat{\gamma}(n)}\bigg|_{\widehat{\gamma}(n)=\widehat{\gamma}^{(t-1)}(n)} \tag{5.15}$$

and set $c^o_{k,t}(h) = c_{k,\ell^o_k}(h)$, where the optimal index ℓ^o_k, for group k, is obtained by solving:

$$\{\ell^o_k, \beta^o_k\} = \underset{\{1\leq \ell \leq L_k, \beta_k\}}{\arg\min} \sum_{n=1}^{N} \Big(g_{k,t}(n) - \beta_k\, c_{k,\ell}(h_{k,n})\Big)^2 \tag{5.16}$$

Once the $\{c^o_{k,t}(h)\}$ are selected, we then choose the weights $\{\alpha_k(t)\}$ for the K groups in order to result in steepest decline in the value of the empirical risk, namely,

$$\{\alpha^o_k(t)\} = \underset{\{\alpha_k\}}{\arg\min} \sum_{n=1}^{N} Q\Big(\widehat{\gamma}^{(t-1)}(h_n) + \sum_{k=1}^{K} \alpha_k c^o_{k,t}(h_{k,n})\,;\, \gamma(n)\Big) \tag{5.17}$$

Notice this function has same format as (6.1) in the introduction. With the $\{c^o_{k,t}(h), \alpha^o_k(t)\}$ so determined, we can rewrite (5.14) it in terms of these optimal choices:

$$\widehat{\gamma}^{(t)}(h) = \widehat{\gamma}^{(t-1)}(h) + \sum_{k=1}^{K} \alpha^o_k(t) c^o_{k,t}(h_{k,\cdot}) \tag{5.18}$$

The resulting algorithm is *non-distributed*; nevertheless, it solves the problem of selecting K optimal classifiers and their weights at each stage in order to reduce the empirical risk value sequentially. Since this implementation requires access to global information from across all groups, we shall refer to it as a centralized solution.

5.3.3 Equivalence via Duality Argument

The centralized implementation of gradient boosting involves two steps, (5.17) and (5.18), where agents require access to global information from across all other agents. Our purpose now is to device a fully-decentralized scheme whereby groups rely solely on their local information and on exchanges with their immediate group neighbors in order to construct the aggregate classifier without the need to access global information.

Let us consider first problem (5.17). For generality, we consider a regularized version of

(5.17), say,

$$\{\alpha_k^o(t)\} = \arg\min_{\{\alpha_k\}} \; \rho \sum_{k=1}^{K} q(\alpha_k) \; + \; \sum_{n=1}^{N} Q\left(\widehat{\gamma}^{(t-1)}(h_n) + \sum_{k=1}^{K} \alpha_k c_{k,t}^o(h_{k,n})\,;\, \gamma(n)\right) \tag{5.19}$$

where $\rho > 0$ is a regularization parameter and $q(\cdot)$ is a convex regularization function. The key observation is that the objective function in (5.19) has the form of a "cost-of-sum" since the argument of Q involves a sum in terms of the unknowns, $\{\alpha_k\}$. The duality argument will show that this "cost-of-sum" form can be transformed into an equivalent "sum-of-cost" form, which are particularly amenable to distributed implementations [54].

We start by introducing, for every $n = 1, 2, \ldots, N$, a dummy scalar variable $z(n)$ and transform the above optimization problem into the following constrained form:

$$\begin{aligned} \min_{\{z,\alpha\}} \quad & \rho \sum_{k=1}^{K} q(\alpha_k) + \sum_{n=1}^{N} Q\left(\widehat{\gamma}^{(t-1)}(h_n) + z(n)\,;\,\gamma(n)\right) \\ \text{subject to} \quad & z(n) = \sum_{k=1}^{K} \alpha_k c_{k,t}^o(h_{k,n}), \;\; \text{for } n = 1, 2, \ldots, N \end{aligned} \tag{5.20}$$

It is easy to see that problem (5.20) is a standard convex optimization problem and that, under the linear equality constraints, strong duality holds [46]. As such, we can seek the minimizer(s) $\{z^o(n), \alpha_k^o(t)\}$ by means of Lagrangian duality. Let

$$\mathcal{L}(z, \alpha, \lambda) \triangleq \sum_{n=1}^{N} Q\Big(\widehat{\gamma}^{(t-1)}(h_n) + z(n)\,;\,\gamma(n)\Big) + \rho \sum_{k=1}^{K} q(\alpha_k) + \sum_{n=1}^{N} \lambda(n) \left(z(n) - \sum_{k=1}^{K} \alpha_k c_{k,t}^o(h_{k,n})\right) \tag{5.21}$$

where the $\{\lambda(n)\}$ are scalar Lagrange multipliers. The corresponding dual function is given by:

$$\mathcal{D}(\lambda) \; \triangleq \; \sum_{n=1}^{N} \inf_{z(n)} \Big\{ Q\Big(\widehat{\gamma}^{(t-1)}(h_n) + z(n)\Big) + \lambda(n) z(n)\,;\,\gamma(n)\Big\} \tag{5.22}$$

$$+ \rho \sum_{k=1}^{K} \inf_{\alpha_k} \left\{ q(\alpha_k) - \sum_{n=1}^{N} \frac{\lambda(n) c_{k,t}^o(h_{k,n})}{\rho} \alpha_k \right\} \tag{5.23}$$

where the primal variables $\{\alpha_k^o(t,\lambda), z^o(n,\lambda)\}$ and the dual variable λ are related via:

$$\{z^o(n,\lambda), \alpha_k^o(t,\lambda)\} = \arg\min_{\{z,\alpha\}} \mathcal{L}(z,\alpha,\lambda) \tag{5.24}$$

It will be shown later that the primal variables can be recovered in a distributed manner. We can now call upon the concept of a conjugate function. For any function $r(x)$ of a scalar variable x, the conjugate function is denoted by $r^\star(\nu)$ [46], where ν is a scalar argument, and defined as

$$r^\star(\nu) \;\stackrel{\Delta}{=}\; \sup_x \left(\nu x - r(x)\right) \tag{5.25}$$

For many common regularization forms, closed form expressions exist for $q^\star(\nu)$ — see [54, 101]. The first minimization in (5.22) can also be expressed in closed form in important cases, such as when the loss function $Q(\cdot)$ is chosen as the exponential loss, or the square loss, or the logit loss. For now, we denote the minimum value of the first term generically as

$$Q^o(n,\lambda) \;\stackrel{\Delta}{=}\; -\inf_{z(n)} \left\{Q\left(\widehat{\gamma}^{(t-1)}(h_n) + z(n)\,;\,\gamma(n)\right) \;+\; \lambda(n)z(n)\right\} \tag{5.26}$$

so that the expression (5.22) can be written as:

$$\mathcal{D}(\lambda) = -\sum_{n=1}^{N} Q^o(n,\lambda) - \rho\sum_{k=1}^{K} q^\star\left(\sum_{n=1}^{N}\frac{\lambda(n)c_{k,t}^o(h_{k,n})}{\rho}\right) \tag{5.27}$$

We introduce the following compact notation for the cost at group k:

$$\mathcal{D}(\lambda) = -\sum_{k=1}^{K} J_k(\lambda), \;\; \text{where } J_k(\lambda) \;\stackrel{\Delta}{=}\; \rho q^\star\left(\sum_{n=1}^{N}\frac{\lambda(n)c_{k,t}^o(h_{k,n})}{\rho}\right) + \frac{1}{K}\sum_{n=1}^{N} Q^o(n,\lambda) \tag{5.28}$$

Therefore, the problem of determining the optimal dual variable, λ^o, can be equivalently stated as

$$\min_{\lambda} \sum_{k=1}^{K} J_k(\lambda) \;\;\Longrightarrow\;\; \lambda^o \tag{5.29}$$

The purpose of the duality argument employed so far has been to transform the original problem (5.19) into the equivalent problem (5.29), which involves minimizing an objective

function that is written in the form of a "sum-of-costs." Such forms admit efficient distributed implementations, meaning that each group k is now able to estimate λ^o on its own by interacting solely with its neighbors. We shall denote these local estimates by λ_k^o, with a subscript k to indicate the group index. These estimates can be computed in a distributed manner by employing consensus or diffusion strategies [11, 28].

5.3.4 Diffusion Learning

In the diffusion implementation, at every stage t and starting from some initial value, each group k repeats the following computations a couple of times until its estimate for the vector λ, denoted by $\lambda_{k,i}$ at time i, converges close enough towards a limiting value, denoted by λ_k^o; this limiting value is the local estimate for the desired dual variable λ^o:

Diffusion strategy [11] (run by every group k)

Repeat $i = 1, 2, \ldots, I$:

$$\begin{aligned} \phi_{k,i} &= \lambda_{k,i-1} - \mu \nabla_\lambda J_k(\lambda_{k,i-1}) \\ \lambda_{k,i} &= \sum_{k' \in \mathcal{N}_k} a_{k'k} \phi_{k',i} \end{aligned} \tag{5.30}$$

End

Set $\lambda_k^o = \lambda_{k,I}$

In the above adapt-then-combine (ATC) diffusion strategy [11, 28, 102], for every i, agent k first move along the negative direction of its cost gradient to generate the intermediate estimate $\phi_{k,i}$, followed by a consultation step where it combines the intermediate estimates $\{\phi_{k',i}\}$ from its neighbors to obtain $\lambda_{k,i}$. We shall represent the diffusion strategy more compactly by writing

$$\lambda_k^o = \text{diffusion}\,\{J_k(\lambda), \mathcal{N}_k, I\} \tag{5.31}$$

where $\mathcal{N}_k$ denotes the neighborhood of group k, and I denotes the number of iterations to be used; this parameter is set by the designer. All groups apply the diffusion strategy

simultaneously. Consequently, every group k will end up with a local version, λ_k^o, for the global dual variable λ^o. In this way, each group k can now compute a local version for $z^o(n)$ and its optimal coefficient $\alpha_k^o(t)$ by solving:

$$z_k^o(n) = \arg\min_z \ \left\{ Q\left(\widehat{\gamma}_k^{(t-1)}(h_n) + z\,;\,\gamma(n)\right) + \lambda_k^o(n) z \right\} \tag{5.32}$$

$$\alpha_k^o(t) = \arg\min_{\alpha_k} \left\{ q(\alpha_k) \ - \ \left(\sum_{n=1}^{N} \frac{\lambda_k^o(n) c_{k,t}^o(h_{k,n})}{\rho} \right) \alpha_k \right\} \tag{5.33}$$

We now explain how the prediction variables can be estimated for arbitrary features, h. Indeed, note that after completing T stages of the diffusion strategy to learn the dual variable, each agent k will have available its optimal coefficients $\alpha_k^o(t)$ and classifier selections $c_{k,t}^o(\cdot)$. During testing, when a new feature vector h is received, each agent k is able to use this local information to evaluate:

$$b_k^{(T)}(h) \ \triangleq \ \sum_{t=1}^{T} \alpha_k^o(t) c_{k,t}^o(h_{k,\cdot}) \tag{5.34}$$

Then, from the general form (5.13) we know that the overall prediction variable is the aggregate sum of these individual decision variables. When A is doubly-stochastic, this sum can be evaluated in a distributed manner by each agent k run the traditional consensus iteration [11, 103] to combine repeatedly the local values at its neighbors.

Local averaging (run by every group k)

Initialization : start from $s_k^{(0)}(h) = b_k^{(T)}(h)$

Repeat $j = 1, 2, \ldots, J$:

$$s_k^{(j)}(h) = \sum_{k' \in \mathcal{N}_k} a_{k'k}\, s_k^{(j-1)}(h)$$

End

Set $\widehat{\gamma}_k^{(T)}(h) = K \cdot s_k^{(J)}(h)$

(5.35)

We shall represent the above averaging procedure more compactly by writing:

$$\widehat{\gamma}_k^{(T)}(h) = K \cdot \text{average}\left\{b_k^{(T)}(h), \mathcal{N}_k, J\right\} \tag{5.36}$$

where J is the number of iterations. In summary, the algorithm is listed as follows:

Table 5.2: Listing of diffusion gradient boosting algorithm

Diffusion gradient boosting algorithm

Initialization:

choose $\widehat{\gamma}_k^{(0)}(n)$, for $n = 1 \ldots N$, and $k = 1 \ldots K$

Repeat $t = 1, 2, \ldots, T$:

for every agent (in parallel) $k = 1, 2, \ldots, K$:

$$g_{k,t}(n) = -\left.\frac{\partial Q\big(\widehat{\gamma}\,;\,\gamma\big)}{\partial \widehat{\gamma}(n)}\right|_{\widehat{\gamma}=\widehat{\gamma}_k^{(t-1)}(n)} \tag{5.37}$$

$$\{\ell_k^o, \beta_k^o\} = \underset{\{1\le\ell\le L_k, \beta_k\}}{\arg\min} \sum_{n=1}^{N} \Big(g_{k,t}(n) - \beta_k\, c_{k,\ell}(h_{k,n})\Big)^2 \implies \text{set } c_{k,t}^o(h) = c_{k,\ell_k^o}(h) \tag{5.38}$$

$$J_k(\lambda) = \text{expression (5.28)} \tag{5.39}$$

$$\lambda_k^o = \text{diffusion}\,\{J_k(\lambda), \mathcal{N}_k, I\} \tag{5.40}$$

$$\alpha_k^o(t) = \underset{\alpha_k}{\arg\min}\left\{q(\alpha_k) - \left(\sum_{n=1}^{N} \frac{\lambda_k^o(n) c_{k,t}^o(h_{k,n})}{\rho}\right)\alpha_k\right\} \tag{5.41}$$

$$b_k^{(t)}(h) = b_k^{(t-1)}(h) + \alpha_k^o(t) c_{k,t}^o(h) \tag{5.42}$$

End

End

$$\widehat{\gamma}_k^{(t)}(h) = K \cdot \text{average}\left\{b_k^{(t)}(h), \mathcal{N}_k, J\right\} \tag{5.43}$$

5.4 Simulation on Special Loss Functions

In this section, we consider the exponential loss function, $Q(\gamma, \widehat{\gamma}) = e^{-\gamma\widehat{\gamma}}$, which is associated with AdaBoost learning and will simplify several steps since some minimization operation can be carried out in closed form. To explain the simplifications, we start by noting that:

$$-\left.\frac{\partial Q\big(\widehat{\gamma}(n)\,;\,\gamma(n)\big)}{\partial\widehat{\gamma}(n)}\right|_{\widehat{\gamma}(n)=\widehat{\gamma}_k^{(t-1)}(n)} = \gamma(n)\tau_{k,t}(n), \quad \text{where } \tau_{k,t}(n) \triangleq e^{-\gamma(n)\widehat{\gamma}_k^{(t-1)}(n)} \tag{5.44}$$

Now, it is easy to verify following equivalence by exploiting the fact that $\gamma(n), c_{k,\ell}(h) \in \{\pm 1\}$:

$$\ell_k^o = \arg\min_{1\le\ell\le L_k} \sum_{n=1}^{N} \left(\gamma(n)\tau_{k,t}(n) - c_{k,\ell}(h_{k,n})\right)^2 \Longleftrightarrow \arg\min_{1\le\ell\le L_k} \sum_{n=1}^{N} \tau_{k,t}(n)\,\mathbb{I}\left[c_{k,\ell}(h_{k,n}) \neq \gamma(n)\right] \tag{5.45}$$

where $\mathbb{I}[x]$ denotes the indicator function; it is equal to one when its argument is true and zero otherwise. Result (5.45) indicates that the optimal classifier ℓ_k^o is selected as the one that results in the smallest sum of weights $\tau_{k,t}(n)$ over the misclassified data.

Next, we can evaluate the function $Q^o(n,\lambda)$ defined by (5.26), which in this case is given by:

$$z_k^o(n,\lambda) = -\gamma(n)\ln\left(\frac{\gamma(n)\lambda(n)}{\tau_{k,t}(n)}\right) \implies Q_{k,t}^o(n,\lambda) = \gamma(n)\lambda(n)\left[\ln\left(\frac{\gamma(n)\lambda(n)}{\tau_{k,t}(n)}\right) - 1\right] \tag{5.46}$$

Assume we select the regularization function as the following elastic-net function:

$$q(x) = \delta|x| + \frac{1}{2}|x|^2 \iff q^\star(\nu) = \frac{1}{2}\big|\mathcal{T}_\delta(\nu)\big|^2 \tag{5.47}$$

where $\mathcal{T}_\delta(\nu)$ represents the soft-threshold operator:

$$\mathcal{T}_\delta(\nu) = sgn(\nu)\cdot\max(|\nu| - \delta, 0) \tag{5.48}$$

It follows that the individual costs, $J_k(\lambda)$, defined by (5.28) are given by

$$J_{k,t}(\lambda) = \frac{\rho}{2}\left|\mathcal{T}_\delta\left(\frac{1}{\rho}\sum_{n=1}^{N}\lambda(n)c_{k,t}^o(h_{k,\cdot})\right)\right|^2 + \frac{1}{K}\sum_{n=1}^{N}\gamma(n)\lambda(n)\left[\ln\left(\frac{\gamma(n)\lambda(n)}{\tau_{k,t}(n)}\right)-1\right] \tag{5.49}$$

It also follows from (5.33) that

$$\alpha_k^o(t) = \mathcal{T}_\delta\left(\frac{1}{\rho}\sum_{n=1}^{N}\lambda_{k,t}^o(n)c_{k,t}^o(h_{k,\cdot})\right) \tag{5.50}$$

Furthermore, using expression (5.46) we now have

$$\widehat{\gamma}_k^{(t)}(n) = \widehat{\gamma}_k^{(t-1)}(n) - \gamma(n)\ln\left(\frac{\gamma(n)\lambda_{k,t}^o(n)}{\tau_{k,t}(n)}\right) \tag{5.51}$$

We can use this update to derive an alternative expression for the weight $\tau_{k,t}(n)$ in terms of the local dual variable $\lambda_{k,t}^o(n)$ as follows:

$$\tau_{k,t+1}(n) = \gamma(n)\lambda_{k,t}^o(n) \tag{5.52}$$

It is useful to assign $\lambda_{k,t-1}^o(n)$ as initial starting point for $\lambda_{k,t}^o(n)$ in the diffusion update (5.30), which can reduce the number of iterations due to (5.52).

We now compare the performance of the diffusion Adaboost implementation with elastic-net regularization against the standard (centralized) Adaboost algorithm on two test datasets. Both implementations will be based on the same set of weak classifiers, which are chosen as

$$c_\ell(h_n) = \text{sign}(h_n(p) > \text{thres}_p) \tag{5.53}$$

Each of these classifiers simply compares the $p-$th entry of the feature vectors against a threshold value. For the diffusion AdaBoost setting, we assigned 10 groups, which are connected through a random doubly stochastic matrix. Each group is in charge of one-tenth of total number of feature entries and corresponding weak classifiers.

The test data is obtained from the LIBSVM website[1]. We first use the Adult dataset after preprocessing [104] with 11,220 training data and 21,341 testing data in 123 feature dimensions. The parameter setting is as follows: regularization term coefficients $\rho = 0.005$, ℓ_1-norm ratio $\delta = 0.1$, and the step-size $\mu = 1 \times 10^{-6}$. We also examine the performance of the algorithm on another large-scale dataset, namely, the Reuters Corpus Volume I (RCV1) data with 20242 training data and 253843 testing data consisting of 47236 feature dimensions. The agent setting is the same and the parameter setting is $\rho = 0.01$, $\delta = 0.2$, and $\mu = 1 \times 10^{-6}$.

One observation stands out from these results. The dotted lines in the figure confirm that if the individual groups were to rely solely on their classifiers to solve the inference task, then their performance will be poor. However, once they start cooperating locally and sharing local information, the network of dispersed groups is able to match the information of the centralized Adaboost solution.

[1] `http://www.csie.ntu.edu.tw/~cjlin/libsvmtools/datasets/binary.html`

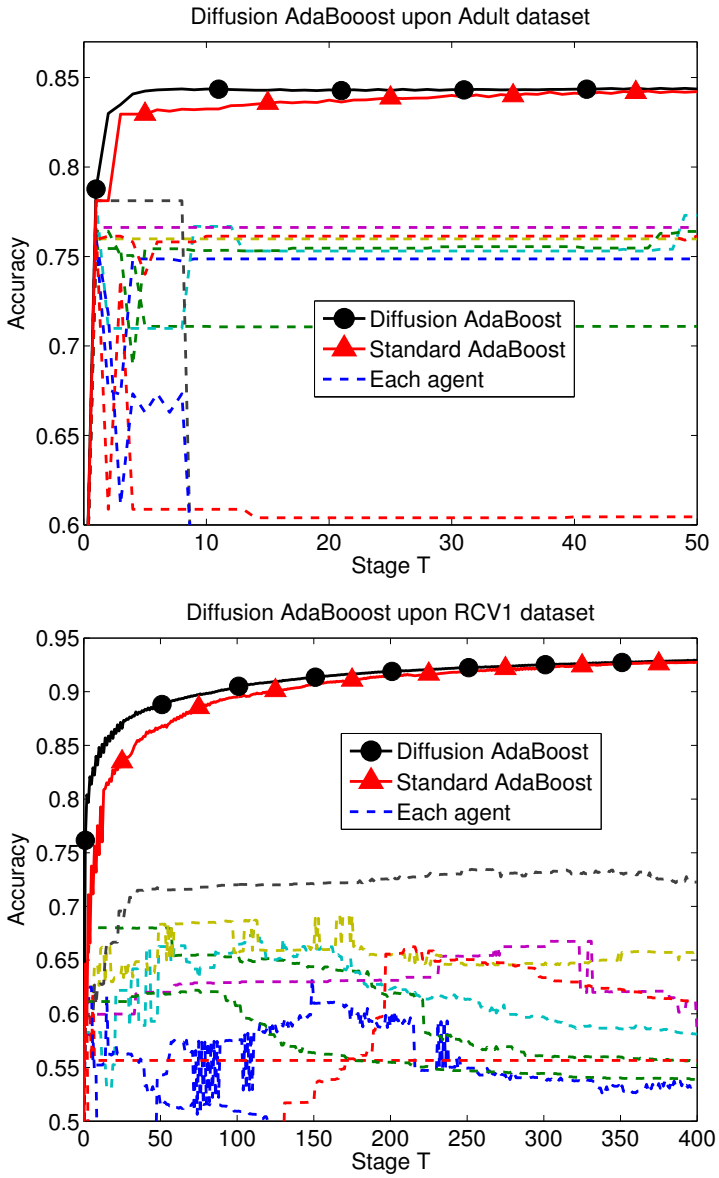

Figure 5.2: Evolution of the performance curves. There are ten dotted lines. Each represents the performance of one group based on its own weak classifiers.

CHAPTER 6

Learning Under Distributed Features

This chapter continues to study the problem of learning under both large datasets and large-dimensional feature space scenarios. The feature information is assumed to be spread across agents in a network, where each agent observes some of the features. Through local cooperation, the agents are supposed to interact with each other to solve an inference problem and converge towards the global minimizer of an empirical risk. We study this problem exclusively in the primal domain, and propose new and effective distributed solutions with guaranteed convergence to the minimizer. This is achieved by combining a dynamic diffusion construction, a pipeline strategy, and variance-reduced techniques.

6.1 Introduction and Problem Formulation

For easy reference, we illustrate our netwokred agent model in Fig. 6.1 again:

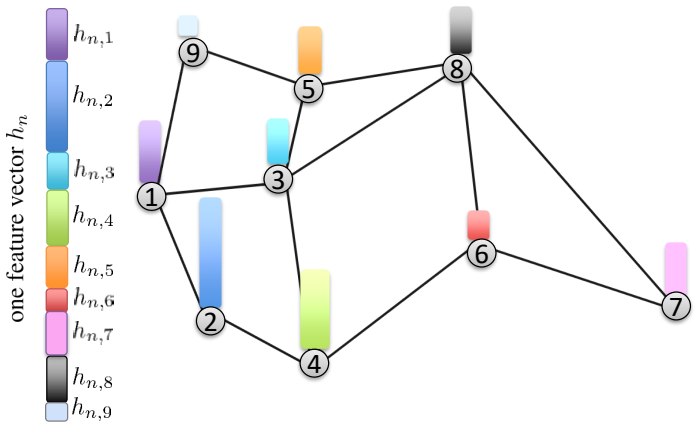

Figure 6.1: Distributing the feature across the networked agents.

Also, we list the empirical risk function considered in this chapter here again:

$$J(w) = \frac{1}{N}\sum_{n=1}^{N} Q\left(\sum_{k=1}^{K} h_{n,k}^{\mathsf{T}} w_k; \gamma_n\right) + \sum_{k=1}^{K} r(w_k) \tag{6.1}$$

where we are also assuming that the regularization term satisfies an additive factorization of the form

$$r(w) = \sum_{k=1}^{K} r(w_k) \tag{6.2}$$

with regularization applied to each sub-vector w_k.

6.1.1 Related Works

Problems of this type have been pursued before in the literature by using duality arguments, such as those in [15, 54, 105–107]. One common way to do that is to transform problem (6.1) into a constrained problem, say, as:

$$\begin{aligned} \min_{w\in\mathbb{R}^M} \quad & J(w) = \frac{1}{N}\sum_{n=1}^{N} Q\left(z_n; \gamma_n\right) + \sum_{k=1}^{K} r(w_k) \\ \text{s.}t. \quad & z_n = \sum_{k=1}^{K} h_{n,k}^{\mathsf{T}} w_k \qquad n = 1, 2, \ldots, N \end{aligned} \tag{6.3}$$

Next, exploiting a duality argument, problem (6.3) is equivalent to solving the following dual problem:

$$\min_{y\in\mathbb{R}^N} D(y) = \sum_{k=1}^{K}\left\{\frac{K}{N}\sum_{n=1}^{N} Q^{\star}(-y_n; \gamma_n) + r^{\star}\left(\frac{1}{N}\sum_{n=1}^{N} y_n h_{n,k}\right)\right\} \tag{6.4}$$

where the scalar y_n denotes the dual variable corresponding to the $n-$th constraint, and $Q^{\star}(\cdot)$ and $r^{\star}(\cdot)$ represent the conjugate functions, i.e., $f^{\star}(y) = \sup_x(y^{\mathsf{T}}x - f(x))$, of $Q(\cdot)$ and $r(\cdot)$, respectively. Note that the function in (6.4) has the format of a "sum-of-cost" and each term inside the summation can be computed by each agent alone. Therefore, problem

(6.4) can be solved in a number of traditional distributed algorithms [11, 57, 58, 108, 109]. However, the resulting algorithms suffer from some limitations. One limitation is that the term $\frac{1}{N}\sum_{n=1}^{N} y_n h_{n,k}$ inside of $r^\star(x)$ has complexity $O(N)$ to compute. Another limitation is that the resulting algorithms rely on the use of conjugate functions, which are not always available in closed form; this will greatly depend on the nature of the loss function $Q(\cdot)$. This limitation is worse for nonlinear models, say, for:

$$J(w) = \frac{1}{N}\sum_{n=1}^{N} Q\Big(\sum_{k=1}^{K} f(w_k, h_{n,k}); \gamma_n\Big) + r(w) \tag{6.5}$$

where $f(w_k, h_{n,k})$ is some nonlinear function. These difficulties do not arise if we pursue a solution directly in the primal domain. For example, the case of nonlinear models can be handled through the chain rule of differentiation (as in backpropagation), when deriving stochastic gradient algorithms. Furthermore, we are often interested more directly in the primal rather than the dual variable.

With regards to the large feature space, one may be motivated to consider coordinate descent techniques [110, 111], which pursue optimization one coordinate y_n at a time. However, these techniques still face difficulties in both the primal and dual domains. For instance, in the primal domain [112–114], they will generally require two time-scales: one scale governs the rate for updating the gradients and a second faster scale for running averaging iterations multiple times. This feature limits the rate at which data can be sampled because the inner calculation will need to be completed before the computation of the next datum. The same difficulties mentioned above for the dual methods will again arise if coordinate descent solutions are implemented in the dual domain [106, 115, 116]. For these reasons, the approach proposed in this chapter does not rely directly on coordinate descent implementations.

Other useful approaches to solving problems of the form (6.1) are the Alternating Direction Method of Multipliers (ADMM) [105, 117, 118] and primal dual-methods [57, 108, 119, 120]. These techniques have good convergence properties but continue to suffer from high computational costs and two-time scale communications.

6.1.2 Novelty and Contributions

In this chapter, we propose a *stochastic* solution method to (6.1) that operates directly in the *primal* domain. By avoiding the dual formulation, we will arrive at a simple and effective method even for large scale applications. We exploit the idea of dynamic consensus algorithm [121, 122], which has been adopted in the distributed optimization algorithms to track the average of gradients, see [113, 123–126]. Meanwhile, we are interested in tracking the sum of score, $\sum_{k=1}^{K} h_{n,k}^{\mathsf{T}} w_k$, due to the different problem setting. More importantly, we will show that the proposed method is able to converge at a *linear* rate to the *exact* minimizer of the empirical risk $J(w)$ even under constant step-size learning. We will also exploit variance-reduced techniques [39] and a pipeline strategy [127] to obtain a reduced complexity algorithm that requires only $O(1)$ operations per iteration. The algorithm will not require two (or separate) time-scales and will not necessitate the solution of auxiliary sub-optimization problems as is common in prior methods in the literature.

Problems similar to (6.1) were also studied in [5] using similar primal methods like us but in a deterministic setting, but the approach is not well-suited for big-data applications.

Notation: We use plain letters for deterministic variables, and boldface letters for random variables. We also use $\mathbb{E}_x$ to denote the expectation with respect to x, $col\{x_1, \cdots, x_n\}$ to denote a column vector formed by stacking $x_1, \cdots, x_n$, $(\cdot)^{\mathsf{T}}$ to denote transposition, and $\|\cdot\|$ for the 2-norm of a matrix or the Euclidean norm of a vector. Through this chapter, we use the subscript n as the index of data, the subscript i, j as the index of iteration/time, and k, ℓ as the index of agent. We also put i, j as the superscript with the same meaning, i.e., the index of iteration/time. The notation $\mathbb{1}_N = col\{1, \ldots, 1\} \in \mathbb{R}^N$.

6.2 Preliminary Naïve Solution

We first propose a simple and naïve solution, which will be used to motivate the more advanced algorithms in later sections.

6.2.1 Networked Agent Model and Consensus Strategy

To begin with, we introduce several preliminary concepts that will be exploited in later sections.

We consider the graph model shown in Fig. 6.1. In this construction, the communication network of the agents is modeled as a fixed directed graph $\mathcal{G} = (\{1, \cdots, K\}; \mathcal{E})$, where $\mathcal{E} \subseteq \{1, \cdots, K\} \times \{1, \cdots, K\}$ is the set of edges. The edge (ℓ, k) means that agent ℓ can send a message to agent k, where we associate the weight $a_{\ell k}$ as a nonnegative factor that scales the information from agent ℓ to agent k. We assume the combination matrix $A = [a_{\ell k}]$ is symmetric and doubly-stochastic, i.e.,

$$\sum_{\ell=1}^{K} a_{\ell k} = 1, \quad \sum_{k=1}^{K} a_{\ell k} = 1 \tag{6.6}$$

We also assume that $a_{kk} > 0$ for at least one agent k and the underlying graph $\mathcal{G}$ is strongly connected.

Now assume there is a signal d_k at each agent k. Then, a well-studied and traditional algorithm for the agents to learn the average value of all the $\{d_k\}$ signals is to run the consensus iteration [11, 58, 109, 128]:

$$w_{i+1,k} = \sum_{\ell \in \mathcal{N}_k} a_{\ell k} w_{i,\ell}, \quad where\ w_{0,k} = d_k \tag{6.7}$$

where the notation $\mathcal{N}_k$ denotes the set of neighbors of agent k. In this way, each agents starts from its own observation vector d_k and continually averages the state values of its neighbors. After sufficient iterations, it is well-known that

$$w_{i,k} \to \frac{1}{K} \sum_{k=1}^{K} d_k \tag{6.8}$$

under some mild conditions on A [28, 58, 128–130].

6.2.2 Naïve Solution

Now, let us consider the problem of minimizing (6.1) by means of a stochastic gradient recursion. Let $\alpha_{n,k} = h_{n,k}^{\mathsf{T}} w_k$ denote the inner product that is available at agent k at time n and define

$$z_n \triangleq \sum_{k=1}^{K} \alpha_{n,k} \tag{6.9}$$

which is the argument of $Q(\cdot)$ in (6.1):

$$J(w) = \frac{1}{N} \sum_{n=1}^{N} Q\left(z_n; \gamma_n\right) + r(w) \tag{6.10}$$

If we denote the average of the local inner products by

$$\bar{\alpha}_n \triangleq \frac{1}{K} \sum_{k=1}^{K} \alpha_{n,k} \tag{6.11}$$

then the variable z_n is a scaled multiple of $\bar{\alpha}_n$, namely, $z_n = K\bar{\alpha}_n$. Now, the stochastic-gradient step to solving (6.10) will involve approximating the true gradient vector of $J(w)$ by the gradient vector of the loss function evaluated at some randomly selected data pair $(\boldsymbol{z}_{\boldsymbol{n}_i}, \gamma_{\boldsymbol{n}_i})$, where $\boldsymbol{n}_i$ at iteration i denotes the index of the sample pair selected uniformly at random from the index set $\{1, 2, \ldots, N\}$. Doing so, the stochastic gradient recursion will take the form:

$$\boldsymbol{w}_{i+1} = \boldsymbol{w}_i - \mu \nabla_z Q(\boldsymbol{z}_{\boldsymbol{n}_i}; \gamma_{\boldsymbol{n}_i}) h_{\boldsymbol{n}_i} - \mu \nabla_w r(\boldsymbol{w}_i) \tag{6.12}$$

Note that $\boldsymbol{n}_i$ is independent of the iterates $\{\boldsymbol{w}_j\}_{j=0}^{i}$. Recalling that h_n and w are partitioned into K blocks, we can decompose (6.12) into K parallel recursions run at the local agents:

$$\boldsymbol{w}_{i+1,k} = \boldsymbol{w}_{i,k} - \mu \nabla_z Q\left(\boldsymbol{z}_{\boldsymbol{n}_i}; \gamma_{\boldsymbol{n}_i}\right) h_{\boldsymbol{n}_i,k} - \mu \nabla_w r(\boldsymbol{w}_{i,k}) \tag{6.13}$$

One main problem with (6.13) is that it is *not* a distributed solution because agents need to calculate $\nabla_z Q(z;\gamma)$ at z whose value depends on all sub-vectors $\{w_k\}$ from all agents and not just on w_k from agent k. This difficulty suggests one initial solution method.

Since the desired variable z_n is proportional to the average value $\bar{\alpha}_n$, then a consensus-type construction can be used to approximate this average. For some total number of iterations J, each agent would start from $\bar{\alpha}_{n,k}^{(0)} = \alpha_{n,k}$ and repeat the following calculations J times:

$$\bar{\alpha}_{n,k}^{(j)} \leftarrow \sum_{\ell \in \mathcal{N}_k} a_{\ell k} \bar{\alpha}_{n,\ell}^{(j-1)}, \quad j = 1, 2, \ldots, J \tag{6.14}$$

However, this mode of operation requires the agents to complete J consensus updates between two data arrivals and requires a two-time scale operation: a faster time-scale for the consensus iterations and a slower time-scale for the data sampling and computing the gradient. One simplification is to set $J = 1$ and to have each agent perform only one single combination step to generate the variable:

$$\widehat{\boldsymbol{z}}_{\boldsymbol{n}_i,k} = \sum_{\ell \in \mathcal{N}_k} a_{\ell k} K h_{\boldsymbol{n}_i,\ell}^{\mathsf{T}} w_{i,\ell} \tag{6.15}$$

where we are expressing the result of this single combination by $\widehat{\boldsymbol{z}}_{\boldsymbol{n}_i,k}$ to indicate that this is the estimate for $z_{\boldsymbol{n}_i}$ that is computed at agent k at iteration i. Observe also that we are scaling the quantity inside the sum by K since, as explained before, $z_n = K\bar{\alpha}_n$. We list the resulting algorithm in (6.16)–(6.18).

Observe that this implementation requires all agents to use the same random index $\boldsymbol{n}_i$ at iteration i. Although this requirement may appear restrictive, it can still be implemented in distributed architectures. For example, each agent can be set with the same random seed so that they can generate the same index variable $\boldsymbol{n}_i$ at iteration i. To agree on the same random seed in a fully distributed manner, one way is to run the consensus algorithm on the seed in the setting phase. Specifically, each agent generates a random seed number, then runs the consensus algorithm until it converges and rounds the result to the nearest integer. Alternatively, agents can sample the data in a cyclic manner instead of uniform sampling.

Table 6.1: Listing of Naïve feature-distributed method for agent k

Algorithm 6.1 (Naïve feature-distributed method for agent k)

Initialization: Set $w_{0,k} = 0$.
Repeat for $i = 1, 2, \ldots$:

$$\boldsymbol{n}_i \sim \mathcal{U}[1, N] \quad \text{(uniformly sampled)} \tag{6.16}$$

$$\widehat{\boldsymbol{z}}_{\boldsymbol{n}_i,k} = \sum_{\ell \in \mathcal{N}_k} a_{\ell k} (K h_{\boldsymbol{n}_i,k}^{\mathsf{T}} w_{i,k}) \tag{6.17}$$

$$\boldsymbol{w}_{i+1,k} = \boldsymbol{w}_{i,k} - \mu \nabla_z Q\Big(\widehat{\boldsymbol{z}}_{\boldsymbol{n}_i,k}; \gamma_{\boldsymbol{n}_i}\Big) h_{\boldsymbol{n}_i,k} - \mu \nabla_w r(\boldsymbol{w}_{i,k}) \tag{6.18}$$

End

But in the main body of this chapter, we assume each agent k will sample the same index $\boldsymbol{n}_i$ at iteration i for simplicity.

6.2.3 Limitations

Algorithm 1 is easy to implement. However, it suffers from two major drawbacks. First, the variable $\widehat{\boldsymbol{z}}_{\boldsymbol{n}_i,k}$ generated by the combination step (6.17) is not generally a good approximation for the global variable $\boldsymbol{z}_{\boldsymbol{n}_i}$. This approximation error affects the stability of the algorithm and requires the use of very small step-sizes. A second drawback is that the stochastic-gradient implementation (6.16)–(6.18) will converge to a small neighborhood around the exact minimizer rather than to the exact minimizer itself [11, 13]. In the following sections, we will design a more effective solution.

6.3 Correcting the Approximation Error

6.3.1 Dynamic Diffusion Strategy

Motivated by the dynamic average consensus method [121, 122], we will design a stochastic diffusion-based algorithm to correct the error introduced by (6.17). To motivate the algorithm, let us step back and assume that each agent k in the network is observing some dynamic input signal, $d_{i,k} \in \mathbb{R}^N$, that changes with time i. Assume we want to develop

a scheme to ensure that each agent k is able to track the average of all local signals, i.e., $\bar{d}_i = \frac{1}{K}\sum_{k=1}^{K} d_{i,k}$. For that purpose, we consider an optimization problem of the form:

$$\min_{x\in\mathbb{R}^P} \; C_i(x) = \sum_{k=1}^{K} \frac{1}{2}\|x - d_{i,k}\|^2 \tag{6.19}$$

where the cost function $C_i(x)$ is changing with time i. The global minimizer of $C_i(x)$ is the desired average $\bar{r}_i$. However, we would like the agents to attain this solution in a distributed fashion. To this end, we can apply the exact diffusion algorithm developed in [57, 131] to solve (6.19). For this case, the algorithm simplifies to the following recursions:

$$\text{Adapt:} \quad \psi_{i+1,k} = x_{i,k} - \mu(x_{i,k} - d_{i+1,k}) \tag{6.20}$$

$$\text{Correct:} \quad \phi_{i+1,k} = \psi_{i+1,k} + x_{i,k} - \psi_{i,k} \tag{6.21}$$

$$\text{Combine:} \quad x_{i+1,k} = \sum_{\ell\in\mathcal{N}_k} a_{\ell k}\phi_{i+1,\ell} \tag{6.22}$$

Each agent k has a state variable $x_{i,k}$ at time i. Step (6.20) uses the input signal $d_{i+1,k}$ at agent k to update its state to the intermediate value $\psi_{i+1,k}$. The second step (6.21) corrects $\psi_{i+1,k}$ to $\phi_{i+1,k}$, and the third step (6.22) combines the intermediate values in the neighborhood of agent k to obtain the update state $x_{i+1,k}$. The process continues in this manner at all agents. Based on the results from [131] applied to (6.19), we can set $\mu = 1$ in (6.20) and combine three recursions to get

$$x_{i+1,k} = \sum_{\ell\in\mathcal{N}_k} a_{\ell k}\left(x_{i,\ell} + d_{i+1,\ell} - d_{i,\ell}\right) \tag{6.23}$$

with $x_{0,k} = d_{0,k}$ for any k. It can be shown that if the signals change slowly, the $x_{i,k}$ will track the mean $\bar{d}_i = \frac{1}{K}\sum_{k=1}^{K} d_{i,k}$ well [121, 122]. Also, using induction and the initial boundary conditions, it is easy to verify that (6.23) has the unbiasedness property:

$$\sum_{k=1}^{K} x_{i,k} = \sum_{k=1}^{K} d_{i,k}, \quad \forall i > 0 \tag{6.24}$$

In this chapter, we are interested in a second useful property that:

$$\lim_{i\to\infty} x_{i,k} = \frac{1}{K}\sum_{k=1}^{K} d_k, \quad \text{when } \lim_{i\to\infty} d_{i,k} = d_k, \quad \forall k \tag{6.25}$$

This means if the signals $d_{i,k}$ converge, then the $x_{i,k}$ of all agents will converge to the mean of the limit values. We refer to (6.23) as the dynamic diffusion method.

We now apply this intermediate result to the earlier recursion (6.13) to transform it into a distributed solution. Recall that there we need to evaluate the variable

$$\boldsymbol{z}_{\boldsymbol{n}_i} = \sum_{k=1}^{K} h_{\boldsymbol{n}_i,k}^{\mathsf{T}} \boldsymbol{w}_{i,k} \tag{6.26}$$

Calculating this quantity is similar to solving problem (6.19), where each $d_{i,k}$ corresponds to the inner product $h_{\boldsymbol{n}_i,k}^{\mathsf{T}}\boldsymbol{w}_{i,k}$. However, there is one key difference: the signal $h_{\boldsymbol{n}_i}$ is not *deterministic* but *stochastic* and it varies randomly with the data index $\boldsymbol{n}_i$. At any particular iteration i, we do not know beforehand which random index $\boldsymbol{n}_i$ is going to be chosen. This suggests that in principle we should keep track of N variables z_n, one for each possible $n = 1, 2, \ldots, N$. For large datasets, this is of course too costly to implement it. Instead, we propose a more efficient solution where the data is sparsely sampled. Assume first, for the sake of argument only, that we move ahead and compute the variable z_n for every possible value of n. If we do so, we would need to repeat construction (6.23) a total of N times at each node k, one for each n, as follows:

$$\boldsymbol{z}_{1,k}^{i+1} = \sum_{\ell\in\mathcal{N}_k} a_{\ell k}\left(\boldsymbol{z}_{1,\ell}^{i} + Kh_{1,\ell}^{\mathsf{T}}\boldsymbol{w}_{i,\ell} - Kh_{1,\ell}^{\mathsf{T}}\boldsymbol{w}_{i-1,\ell}\right) \tag{6.27}$$

$$\boldsymbol{z}_{2,k}^{i+1} = \sum_{\ell\in\mathcal{N}_k} a_{\ell k}\left(\boldsymbol{z}_{2,\ell}^{i} + Kh_{2,\ell}^{\mathsf{T}}\boldsymbol{w}_{i,\ell} - Kh_{2,\ell}^{\mathsf{T}}\boldsymbol{w}_{i-1,\ell}\right) \tag{6.28}$$

$$\vdots$$

$$\boldsymbol{z}_{N,k}^{i+1} = \sum_{\ell\in\mathcal{N}_k} a_{\ell k}\left(\boldsymbol{z}_{N,\ell}^{i} + Kh_{N,\ell}^{\mathsf{T}}\boldsymbol{w}_{i,\ell} - Kh_{N,\ell}^{\mathsf{T}}\boldsymbol{w}_{i-1,\ell}\right) \tag{6.29}$$

In this description, we are adding a superscript i to each $\boldsymbol{z}_{n,k}^{i}$ to indicate the iteration index.

In this way, each $\boldsymbol{z}_{n,k}^i$ will be able to track the sum $\sum_{k=1}^K h_{n,k}^{\mathsf{T}} w_{i,k}$. However, since the data size N is usually very large, it is too expensive to communicate and update all $\{\boldsymbol{z}_{n,k}\}_{n=1}^N$ per iteration. We propose a stochastic algorithm in which only *one* datum $h_{\boldsymbol{n}_i,k}$ is selected at iteration i and only the corresponding entry $z_{\boldsymbol{n}_i,k}^{i+1}$ be updated while all other $\boldsymbol{z}_{n,k}^{i+1}$ will stay unchanged for $n \neq \boldsymbol{n}_i$:

$$\begin{cases} \boldsymbol{z}_{\boldsymbol{n}_i,k}^{i+1} = \displaystyle\sum_{\ell \in \mathcal{N}_k} a_{\ell k} \left(\boldsymbol{z}_{\boldsymbol{n}_i,\ell}^{j} + K h_{\boldsymbol{n}_i,\ell}^{\mathsf{T}} \boldsymbol{w}_{i,\ell} - K h_{\boldsymbol{n}_i,\ell}^{\mathsf{T}} \boldsymbol{w}_{j-1,\ell} \right) \\ \boldsymbol{z}_{n,k}^{i+1} = \boldsymbol{z}_{n,k}^{i}, \qquad n \neq \boldsymbol{n}_i \end{cases} \tag{6.30}$$

where the index j in the first equation refers to the most recent iteration where the same index $\boldsymbol{n}_i$ was chosen the last time. Note that the value j depends on $\boldsymbol{n}_i$ and the history of sampling, and therefore we need to store the inner product value that is associated with it. To fetch $\boldsymbol{z}_{n,\ell}^j$ and $K h_{\boldsymbol{n}_i,\ell}^{\mathsf{T}} \boldsymbol{w}_{j-1,\ell}$ easily, we introduce two auxiliary variables:

$$\boldsymbol{u}_{\boldsymbol{n}_i,\ell}^i \leftarrow \boldsymbol{z}_{\boldsymbol{n}_i,\ell}^j, \quad \boldsymbol{v}_{\boldsymbol{n}_i,\ell}^i \leftarrow K h_{\boldsymbol{n}_j,\ell}^{\mathsf{T}} \boldsymbol{w}_{j-1,\ell} \tag{6.31}$$

If we view $\{\boldsymbol{z}_{1,k}^{i+1}, \boldsymbol{z}_{2,k}^{i+1}, \ldots, \boldsymbol{z}_{N,k}^{i+1}\}$ as one long vector, the update (6.27)—(6.29) resembles a coordinate descent algorithm [111, 113].

An unbiasedness property similar to (6.24) will continue to hold:

$$\sum_{k=1}^K \boldsymbol{u}_{n,k}^i = \sum_{k=1}^K \boldsymbol{v}_{n,k}^i = \sum_{k=1}^K K h_{n,k}^{\mathsf{T}} \boldsymbol{w}_{j-1,k}, \quad \forall\, n, i > 0 \tag{6.32}$$

It is easy to verify the validity of (6.32) by taking the summation over (6.30) and combining the initialization conditions.

6.3.2 Variance-Reduction Algorithm

We can enhance the algorithm further by accounting for the gradient noise that is present in (6.12): this noise is the difference between the gradient of the risk function $J(w)$, which is unavailable, and the gradient of the loss function that is used in (6.12). It is known, e.g.,

from [11–13, 17] that under constant step-size adaptation, and due to this gradient noise, recursion (6.12) will only approach an $O(\mu)-$neighborhood around the global minimizer of (6.1). We can modify the recursion to ensure convergence to the exact minimizer as follows.

There is a family of variance-reduction algorithms such as SVRG [38], SAGA [39], and AVRG [31] that can approach the exact solution of the empirical risk function with constant step-size. In this chapter, we exploit the SAGA construction because the variables $\{\boldsymbol{u}_{n,k}\}$ can readily be used in that implementation. Let us consider an agent k in a non-cooperative scenario where the vanilla SAGA recursion would be

$$\begin{aligned}\boldsymbol{w}_{i+1,k}=\boldsymbol{w}_{i,k}-\mu\nabla_z Q(h_{\boldsymbol{n}_i,k}^{\mathsf{T}}\boldsymbol{w}_{i,k};\gamma_{\boldsymbol{n}_i})h_{\boldsymbol{n}_i,k} \\ +\mu\nabla_z Q(\boldsymbol{u}_{\boldsymbol{n}_i,k}^{i};\gamma_{\boldsymbol{n}_i})h_{\boldsymbol{n}_i,k}-\frac{\mu}{N}\sum_{n=1}^{N}\nabla_z Q(\boldsymbol{u}_{n,k}^{i};\gamma_n)h_{n,k}\end{aligned} \tag{6.33}$$

$$\boldsymbol{u}_{n,k}^{i+1}=\begin{cases}h_{\boldsymbol{n}_i,k}^{\mathsf{T}}\boldsymbol{w}_{i,k}, & \text{if } n=\boldsymbol{n}_i\\ \boldsymbol{u}_{n,k}^{i}, & \text{otherwise}\end{cases} \tag{6.34}$$

It is proved in [39] that the variance of the gradient noise, i.e., the difference between the gradient step in 6.33 and the full gradient, introduced in SAGA will vanish in expectation. Therefore, SAGA will converge to the exact solution of problem (1.20). Also, note that the N-summation term $\sum_{n=1}^{N}\nabla_z Q(\boldsymbol{u}_n^i;\gamma_n)h_n$ can be calculated in an online manner since only one term is updated, i.e.,

$$\sum_{n=1}^{N}\nabla_z Q(\boldsymbol{u}_n^{i+1};\gamma_n)h_n=\sum_{n=1}^{N}\nabla_z Q(\boldsymbol{u}_n^{i};\gamma_n)h_n-\nabla_z Q(\boldsymbol{u}_{\boldsymbol{n}_i}^{i};\gamma_{\boldsymbol{n}_i})h_{\boldsymbol{n}_i}+\nabla_z Q(\boldsymbol{u}_{\boldsymbol{n}_i}^{i+1};\gamma_{\boldsymbol{n}_i})h_{\boldsymbol{n}_i} \tag{6.35}$$

This online calculation results in $O(1)$ complexity per iteration.

Note that the vanilla SAGA needs to store the gradient $\nabla_z Q(\boldsymbol{u}_{n,k}^{i};\gamma_n)h_{n,k}$ for $n=1,2,\ldots,N$. However, we have already stored $\boldsymbol{u}_{n,k}^{i}$ and the additional storage of the gradient is unnecessary.

Hence, the stochastic gradient recursion(6.13) at each agent k will be modified to (6.38) with two correction terms. The resulting algorithm is summarized in Table 6.2.

Table 6.2: Listing of variance-reduced dynamic diffusion algorithm

Algorithm 6.2 [Variance-Reduced Dynamic Diffusion (VRD^2)]

Initialization: Set $w_{0,k} = 0$; $\boldsymbol{u}_{n,k}^0 = 0$; $\boldsymbol{v}_{n,k}^0 = 0$.
Repeat for $i = 1, 2, \ldots$:

$$\boldsymbol{n}_i \sim \mathcal{U}[1, N] \qquad \text{(uniformly sampled)} \tag{6.36}$$

$$\boldsymbol{z}_{\boldsymbol{n}_i,k} = \sum_{\ell \in \mathcal{N}_k} a_{\ell k} \big(\boldsymbol{u}_{\boldsymbol{n}_i,\ell}^i + K h_{\boldsymbol{n}_i,\ell}^{\mathsf{T}} \boldsymbol{w}_{i,\ell} - \boldsymbol{v}_{\boldsymbol{n}_i,\ell}^i \big) \tag{6.37}$$

$$\begin{aligned} \boldsymbol{w}_{i+1,k} = \boldsymbol{w}_{i,k} - \mu \Bigg\{ & \Big[\nabla_z Q\big(\boldsymbol{z}_{\boldsymbol{n}_i,k}; \gamma_{\boldsymbol{n}_i}\big) - \nabla_z Q\big(\boldsymbol{u}_{\boldsymbol{n}_i,k}^i; \gamma_{\boldsymbol{n}_i}\big) \Big] h_{\boldsymbol{n}_i,k} \\ & + \frac{1}{N} \sum_{n=1}^{N} \nabla_z Q\Big(\boldsymbol{u}_{n,k}^i; \gamma_n\Big) h_{n,k} + \nabla_w r(\boldsymbol{w}_{i,k}) \Bigg\} \end{aligned} \tag{6.38}$$

$$\boldsymbol{u}_{n,k}^{i+1} = \begin{cases} \boldsymbol{z}_{\boldsymbol{n}_i,k}^{i+1}, & \text{if } n = \boldsymbol{n}_i \\ \boldsymbol{u}_{n,k}^i, & \text{otherwise} \end{cases} \tag{6.39}$$

$$\boldsymbol{v}_{n,k}^{i+1} = \begin{cases} K h_{\boldsymbol{n}_i,k}^{\mathsf{T}} \boldsymbol{w}_{i,k}, & \text{if } n = \boldsymbol{n}_i \\ \boldsymbol{v}_{n,k}^i, & \text{otherwise} \end{cases} \tag{6.40}$$

End

6.4 Acceleration with pipeline

Algorithm 6.2 can be shown to converge to the solution of problem (1.20) for sufficiently small step-sizes. However, it is observed in numerical experiments that its convergence rate can be slow. One reason is that the variable $\boldsymbol{z}_{n,k}$ generated by (6.37) converges slowly to $\sum_{k=1}^{K} h_{n,k}^{\mathsf{T}} \boldsymbol{w}_{i,k}$. To accelerate convergence, it is necessary to run (6.37) multiple times before the gradient descent step (6.38), which, however, will take us back to in a two-time-scale algorithm. In this section, we propose a pipeline method that accelerates the convergence of $\boldsymbol{z}_{n,k}$ while maintaining the one-time-scale structure.

A pipeline is a set of data processing elements connected in series, where the output of one element is the input to the next element [127]. We assume each agent k stores J variables at iteration i:

$$[\boldsymbol{z}_{\boldsymbol{n}_i,k}^{(0)}, \boldsymbol{z}_{\boldsymbol{n}_{i-1},k}^{(1)}, \cdots \boldsymbol{z}_{\boldsymbol{n}_{i-J+1},k}^{(J-1)}] \in \mathbb{R}^J \tag{6.41}$$

At every iteration, agent k runs a *one-step* average consensus recursion on its state vector (6.41):

$$[\boldsymbol{z}_{\boldsymbol{n}_i,k}^{(1)}, \boldsymbol{z}_{\boldsymbol{n}_{i-1},k}^{(2)}, \cdots \boldsymbol{z}_{\boldsymbol{n}_{i-J+1},k}^{(J)}] = \sum_{\ell \in \mathcal{N}_k} a_{\ell k} [\boldsymbol{z}_{\boldsymbol{n}_i,\ell}^{(0)}, \boldsymbol{z}_{\boldsymbol{n}_{i-1},\ell}^{(1)}, \cdots \boldsymbol{z}_{\boldsymbol{n}_{i-J+1},\ell}^{(J-1)}] \tag{6.42}$$

Then, agent k pops up the variable $\boldsymbol{z}_{\boldsymbol{n}_{i-J+1},k}^{(J)}$ from memory and uses it to continue the stochastic gradient descent steps. Note that the variable $\boldsymbol{z}_{n,k}^{(J)}$ can be interpreted as the result of applying J consensus iterations and, therefore, it is a better approximation for $\sum_{k=1}^{K} h_{n,k}^{\mathsf{T}} \boldsymbol{w}_{n,k}$. At iteration $i+1$, agent k will push a new variable $\boldsymbol{z}_{\boldsymbol{n}_{i+1},k}^{0} = K \boldsymbol{h}_{\boldsymbol{n}_{i+1},k}^{\mathsf{T}} \boldsymbol{w}_{i+1,k}$ into the buffer and update its state to

$$[\boldsymbol{z}_{\boldsymbol{n}_{i+1},k}^{(0)}, \boldsymbol{z}_{\boldsymbol{n}_i,k}^{(1)}, \cdots \boldsymbol{z}_{\boldsymbol{n}_{i-J+2},k}^{(J-1)}] \in \mathbb{R}^J. \tag{6.43}$$

Recursion (6.42) employs the pipeline strategy. For example, variable $\boldsymbol{z}_{\boldsymbol{n}_i,k}^{(1)}$ is updated

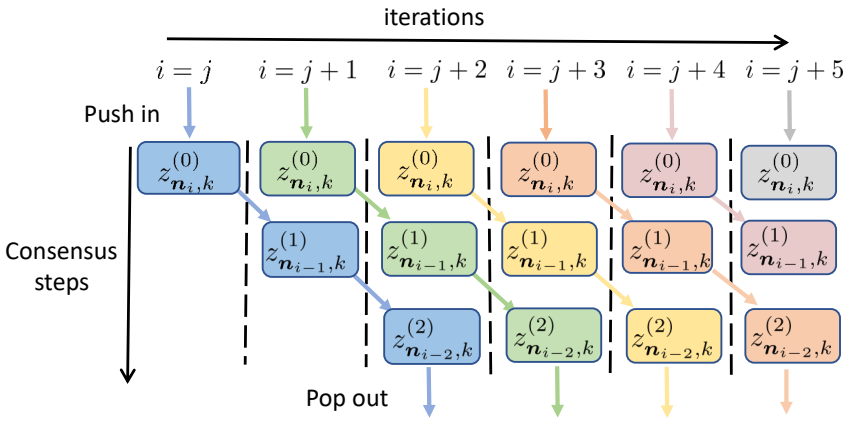

Figure 6.2: Illustration of the pipeline strategy with buffer length $J = 3$.

at iteration i. This new output $\boldsymbol{z}_{\boldsymbol{n}_i,k}^{(1)}$ will become the second input at iteration i and is used to produce the output $\boldsymbol{z}_{\boldsymbol{n}_i,k}^{(2)}$. Next, the output $\boldsymbol{z}_{\boldsymbol{n}_i,k}^{(2)}$ will be the third input at iteration $i+2$ and is used to produce the output $\boldsymbol{z}_{\boldsymbol{n}_i,k}^{(3)}$. If we follow this procedure, the output $z_{\boldsymbol{n}_i,k}^{(J)}$ will be reached at iteration $i+J-1$. At that time, we can pop up $z_{\boldsymbol{n}_i,k}^{(J)}$ and use it in the stochastic gradient update. The pipeline procedure is summarized in the "Pipeline function" shown above, and Fig. 6.2 illustrates the pipeline strategy.

The pipeline strategy has two advantages. First, it is able to calculate $z_{n,k}^{(J)}$ without inner

Table 6.3: Listing of pipeline function

Pipeline function

Initialization: $\boldsymbol{z}_{\boldsymbol{n}_i,k} = 0$ for any $i \leq 0$

Function Pipeline $\left(\boldsymbol{z}_{\boldsymbol{n}_i,k}^{(0)}, \boldsymbol{v}_{\boldsymbol{n}_i,k}^{i+J-1}\right)$

$$\text{Push} \quad \left[\boldsymbol{z}_{\boldsymbol{n}_i,k}^{(0)}, \boldsymbol{v}_{\boldsymbol{n}_i,k}^{i+J-1}\right] \quad \text{into the queue} \tag{6.44}$$

$$[\boldsymbol{z}_{\boldsymbol{n}_i,k}^{(1)}, \boldsymbol{z}_{\boldsymbol{n}_{i-1},k}^{(2)}, \cdots \boldsymbol{z}_{\boldsymbol{n}_{i-J+1},k}^{(J)}] = \sum_{\ell \in \mathcal{N}_k} a_{\ell k} [\boldsymbol{z}_{\boldsymbol{n}_i,\ell}^{(0)}, \boldsymbol{z}_{\boldsymbol{n}_{i-1},\ell}^{(1)}, \cdots \boldsymbol{z}_{\boldsymbol{n}_{i-J+1},\ell}^{(J-1)}] \tag{6.45}$$

$$\text{Pop} \quad \left[\boldsymbol{z}_{\boldsymbol{n}_{i-J+1},k}^{(J)}, \boldsymbol{v}_{\boldsymbol{n}_{i-J+1},k}^{i}\right] \quad \text{out of the queue} \tag{6.46}$$

Return $\left[\boldsymbol{z}_{\boldsymbol{n}_{i-J+1},k}^{(J)}, \boldsymbol{v}_{\boldsymbol{n}_{i-J+1},k}^{i}\right]$

loop, which accelerates the algorithm and maintains the one-time-scale structure. Second, in one iteration, the two-time-scale solution sends a scalar J times while the pipeline solution

Table 6.4: Listing of pipelined variance-reduced dynamic diffusion algorithm

Algorithm 6.4 [Pipelined Variance-Reduced Dynamic Diffusion (PVRD^2)]

Initialization: Set $w_{0,k}=0$; $\boldsymbol{u}_{n,k}^0=0$; $\boldsymbol{v}_{n,k}^0=0$.
Repeat for $i=1,2,\ldots$:

$$\boldsymbol{n}_i \sim \mathcal{U}[1,N] \quad \text{(uniformly sampling)} \tag{6.47}$$

$$\Big[\boldsymbol{z}_{\boldsymbol{n}_i',k}^{(J)}, \boldsymbol{v}_{\boldsymbol{n}_i',k}^{i+1}\Big] = \text{Pipeline}\big(\boldsymbol{u}_{\boldsymbol{n}_i,k}^i + Kh_{\boldsymbol{n}_i,k}^{\mathsf{T}}\boldsymbol{w}_{i,k} - \boldsymbol{v}_{\boldsymbol{n}_i,k}^i,\ Kh_{\boldsymbol{n}_i,k}^{\mathsf{T}}\boldsymbol{w}_{i,k}\big) \tag{6.48}$$

$$\text{(denote } \boldsymbol{n}_i' \triangleq \boldsymbol{n}_{i-J+1}\text{)}$$

$$\boldsymbol{w}_{i+1,k} = \boldsymbol{w}_{i,k} - \mu\Bigg\{\Big[\nabla_z Q\big(\boldsymbol{z}_{\boldsymbol{n}_i',k}^{(J)};\gamma_{\boldsymbol{n}_i'}\big) - \nabla_z Q\big(\boldsymbol{u}_{\boldsymbol{n}_i',k}^i;\gamma_{\boldsymbol{n}_i'}\big)\Big]h_{\boldsymbol{n}_i',k} + \frac{1}{N}\sum_{n=1}^{N}\nabla_z Q\Big(\boldsymbol{u}_{n,k}^i;\gamma_n\Big)h_{n,k} + \nabla_w r(\boldsymbol{w}_{i,k})\Bigg\} \tag{6.49}$$

$$\boldsymbol{u}_{n,k}^{i+1} = \begin{cases} \boldsymbol{z}_{\boldsymbol{n}_i',k}^{(J)}, & \text{if } n=\boldsymbol{n}_i' \\ \boldsymbol{u}_{n,k}^i, & \text{otherwise} \end{cases} \tag{6.50}$$

$$\boldsymbol{v}_{n,k}^{i+1} = \begin{cases} \boldsymbol{v}_{\boldsymbol{n}_i',k}^{i+1}, & \text{if } n=\boldsymbol{n}_i' \\ \boldsymbol{v}_{n,k}^i, & \text{otherwise} \end{cases} \tag{6.51}$$

End

sends a J-length vector just once. Though the communication load is the same, the pipeline solution is usually faster than the two-time-scale approach in practice. That is because sending a scalar with J time needs all agents to synchronize for J times, which can take longer time than the one-time communication.

Observe that when $z_{\boldsymbol{n}_i,k}^{(0)}$ is popped out of the pipeline after J iterations, the required $\boldsymbol{v}_{\boldsymbol{n}_i,k}^i$ should also be from J iterations ago. This means that we need to store the past $\boldsymbol{v}_{\boldsymbol{n}_i,k}^i$ values. One solution is to push the auxiliary $\boldsymbol{v}_{\boldsymbol{n}_i,k}^i$ along with $z_{\boldsymbol{n}_i,k}^{(0)}$ into the pipeline but without doing any processing inside the pipeline. Since this value will be used J iterations into the future, this variable should be denoted by $\boldsymbol{v}_{\boldsymbol{n}_i,k}^{i+J-1}$.

6.5 Algorithm Analysis

6.5.1 Delayed gradient calculations

First, we have to point out that the pipeline solution is not equivalent to the two-time-scale solution. It is observed that the gradient used in the stochastic gradient descent step (6.49) at iteration i is $\nabla_z Q(\boldsymbol{z}^{(J)}_{\boldsymbol{n}'_i,k};\gamma_{\boldsymbol{n}'_i})$ where $\boldsymbol{n}'_i = \boldsymbol{n}_{i-J+1}$, which is J-time out-of-date. The cause of the delay is that the variable $\boldsymbol{z}_{n,k}$ has to conduct J updates in the pipeline. When $\boldsymbol{z}^{(J)}_{\boldsymbol{n}_i,k}$ pops up from the pipeline, the iteration index has arrived to $i+J-1$. As a result, the pipeline solution introduces delays in the gradient. Due to this delay, it does not necessarily follow that deeper pipelines lead to better performance. Actually there is a trade off between depth and performance.

Fortunately, problems involving delays in gradient calculations are well-studied in the distributed machine learning and signal processing literature [132–134]. These works show that convergence can still occur albeit the stability ranges are affected. Although, in most literature, the delayed gradient is usually caused by unbalanced computation loads or fragile communication environments instead of pipeline structure, the proof in this chapter is inspired from these investigations to some extent.

6.5.2 Convergence Analysis

To establish the convergence theorem, we require two assumptions

Assumption 6.1 (Risk Function) *The loss function* $Q(z;\gamma_n) \triangleq Q(h_n^{\mathsf{T}}w;\gamma_n)$ *is differentiable, and has an L-Lipschitz continuous gradient with respect to* w *and a* δ*-Lipschitz continuous gradient with respect to* z*, for every* $n=1,\ldots,N$*, i.e., for any* $w_1, w_2 \in \mathbb{R}^M$*:*

$$\begin{aligned}\|\nabla_w Q(h_n^{\mathsf{T}}w_1;\gamma_n) - \nabla_w Q(h_n^{\mathsf{T}}w_2;\gamma_n)\| &= \|\nabla_z Q(h_n^{\mathsf{T}}w_1;\gamma_n)h_n - \nabla_z Q(h_n^{\mathsf{T}}w_2;\gamma_n)h_n\| \\ &\leq L\|w_1 - w_2\| \qquad (6.52)\end{aligned}$$

$$\|\nabla_z Q(z_1;\gamma_n) - \nabla_z Q(z_2;\gamma_n)\| \leq \delta\|z_1 - z_2\| \qquad (6.53)$$

where $L > 0$ and $\delta > 0$. For the regularization term $r(w)$, it is convex and has η-Lipschitz continuous gradient:

$$\|\nabla_w r(w_1) - \nabla_w r(w_2)\| \leq \eta \|w_1 - w_2\| \tag{6.54}$$

We also assume that the risk $J(w)$ is ν-strongly convex, namely,

$$\Big(\nabla_w J(w_1) - \nabla_w J(w_2)\Big)^{\mathsf{T}} (w_1 - w_2) \geq \nu \|w_1 - w_2\|^2 \tag{6.55}$$

□

Assumption 6.2 (Topology) *The underlying topology is strongly connected, and the combination/weighted adjacency matrix A is symmetric and doubly stochastic, i.e.,*

$$A = A^{\mathsf{T}} \text{ and } A\mathbb{1}_K = \mathbb{1}_K \tag{6.56}$$

where $\mathbb{1}$ is a vector with all unit entries. We further assume that $a_{kk} > 0$ for at least one agent k. □

Under assumption 6.2, we can show that matrix A is primitive [11, 129] and that the second largest magnitude of the eigenvalue of A, denoted by λ, satisfies [130]:

$$0 < \lambda < 1 \tag{6.57}$$

Theorem 6.1 (Convergence of PVRD^2) *Algorithm PVRD^2 converges at a linear rate for sufficiently small step-sizes μ, i.e.,*

$$\mathbb{E}\,\|\boldsymbol{w}_{i,k} - w^\star\|^2 \leq \rho^i C, \quad \forall k, i > 0 \tag{6.58}$$

for some constant C, where:

$$\rho = \max\left(1 - \frac{1-\lambda^J}{2N}, 1 - \mu\nu/5\right) \tag{6.59}$$

Proof: See Appendix 6.A. ■

This theorem indicates that the convergence rate of the algorithm depends on the network topology through λ, the depth of pipeline J, and the strong convexity parameter ν. When J is not very large and the first term in (6.59) is larger than the second one, the convergence rate will depend more on the network topology and the depth of the pipeline. However, when J is large enough so that the second term is dominant, the algorithm performance will depend on the strong convexity parameter ν, as in the single agent case.

Remark: The theorem indicates that the algorithm converges to the minimizer of empirical risk (6.1) exactly. In many cases we are instead interested in the minimizer of the average risk $J_E(w) = \mathbb{E}_{\boldsymbol{x}} Q(w; \boldsymbol{x}) + r(w)$. By an ergodicity argument, the two minimizers will become closer when the number of data samples increases. □

Notice that algorithm VRD^2 is a special case of algorithm PVRD^2 by setting $J = 1$. Thus, we establish the following corollary from Theorem 6.1.

Corollary 6.1 (Convergence of VRD^2) *Algorithm VRD^2 converges at a linear rate for sufficiently small step-sizes μ, i.e.,*

$$\mathbb{E}\,\|\boldsymbol{w}_{i,k} - w^\star\|^2 \leq \rho^i C, \quad \forall k, i > 0 \tag{6.60}$$

for some constant C, where:

$$\rho = \max\left(1 - \frac{1-\lambda}{2N}, 1 - \mu\nu/4\right) \tag{6.61}$$

■

6.6 Simulations

We illustrate the performance of the PVRD^2 algorithm on the MNIST dataset, which consists of 50000 28×28 handwritten digits[1]. In the simulation, we consider the classification task of predicting digit 0 or digit 1. We separate the features over 8 networked agents. The loss function we use is logistic regression:

$$J(w) = \frac{1}{N} \sum_{n=1}^{N} \ln \left(1 + \exp \left(-\gamma_n \sum_{k=1}^{K} h_{n,k}^{\mathsf{T}} w_k \right) \right) + \rho \sum_{k=1}^{K} \|w_k\|^2 \tag{6.62}$$

In the simulation, we set $\rho = 1 \times 10^{-4}$. From two subplots in Fig. 6.3, we see that each agent is in charge of part of w, and each converges to its corresponding part of $w^\star$. Next, we compare our algorithm to the method proposed in [5] with some modification, which can be viewed as the deterministic full-gradient version of our algorithm without pipeline. To make a fair comparison, we plot the convergence curve based on the count of gradients calculated and the combination step in Fig. 6.4. Notice that when the pipeline step J is larger, we need to do more operation on the combination step (6.45). Therefore, we use a different mini-batch B for different J but we keep the sum, $J + B$ equal 30. The curve shows that the larger J we set, the faster the algorithm converges until it is large enough to trigger the second term in the convergence rate.

Next, we compare the communication cost on the CIFAR-10 dataset, which consists of 50000 32x32 color images in 10 classes, with 5000 images per class[2]. For this multi class problem, we use the softmax as the loss function:

$$J(W) = -\frac{1}{N} \sum_{n=1}^{N} \ln \left(\frac{\exp(\sum_{k=1}^{K} W_k[:, \gamma_n]^{\mathsf{T}} h_{n,k})}{\sum_{c=1}^{C} \exp(\sum_{k=1}^{K} W_k[:, c]^{\mathsf{T}} h_{n,k})} \right) + \rho \sum_{k=1}^{K} \|W_k\|_F^2 \tag{6.63}$$

where $\|\cdot\|_F$ represents the Frobenius norm, $W_k \in \mathbb{R}^{M \times C}$ is a matrix that has dimensions $M \times C$, where $C = 10$ in CIFAR-10 problem, and $W[:, c]$ is the C-th column of matrix W.

[1] `http://yann.lecun.com/exdb/mnist/`

[2] `https://www.cs.toronto.edu/~kriz/cifar.html`

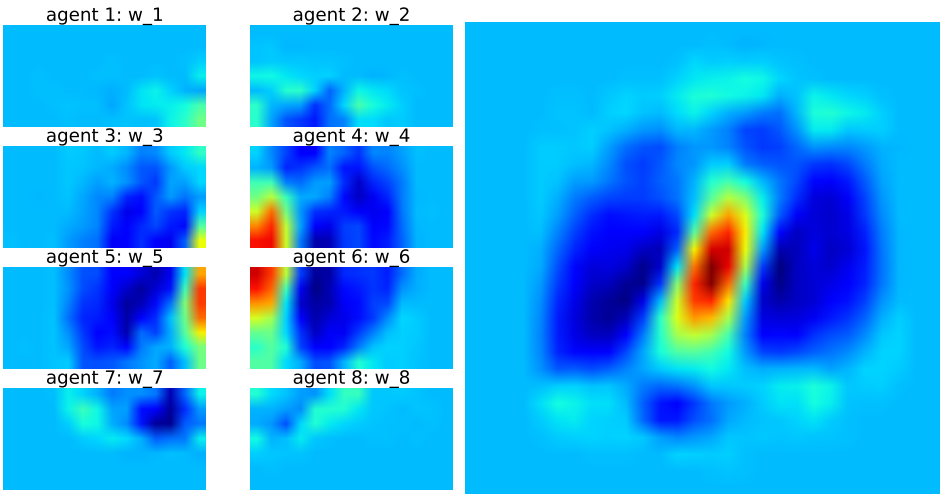

Figure 6.3: Left: Visualization of $\boldsymbol{w}_{i,k}$; Right: Visualization of $w^\star$.

Notice that this model is a little bit different from (1.20). However, it is still easy to adapt our algorithm for the softmax cost. Notice that we now need to find the summation of C-classes, i.e.

$$z_{n,k} \triangleq \begin{bmatrix} z_{n,k}(1) \\ z_{n,k}(2) \\ \vdots \\ z_{n,k}(C) \end{bmatrix} \triangleq \begin{bmatrix} \sum_{k=1}^{K} W_k[:,1]^{\mathsf{T}} h_{n,k} \\ \sum_{k=1}^{K} W_k[:,2]^{\mathsf{T}} h_{n,k} \\ \vdots \\ \sum_{k=1}^{K} W_k[:,c]^{\mathsf{T}} h_{n,k} \end{bmatrix} \tag{6.64}$$

As long as the agent has the information of $z_{n,k}$, it can compute the gradient locally. For simplicity, we let

$$\begin{aligned} prob(z_{n,k},\gamma_n) \triangleq & \frac{\exp\Big(z_{n,k}(\gamma_n)\Big)}{\sum_{c=1}^{C}\exp\Big(z_{n,k}(c)\Big)} \\ = & \frac{\exp\Big(\sum_{k=1}^{K} W_k[:,\gamma_n]^{\mathsf{T}} h_{n,k}\Big)}{\sum_{c=1}^{C}\exp\Big(\sum_{k=1}^{K} W_k[:,c]^{\mathsf{T}} h_{n,k}\Big)} \end{aligned} \tag{6.65}$$

After some algebraic manipulations, we know that

$$\nabla_{W_k[:,\gamma_n]} J_n(W) = \Big(\text{prob}(z_{n,k},\gamma_n) - 1\Big) h_{n,k} + \rho W_k[:,\gamma_n] \tag{6.66}$$

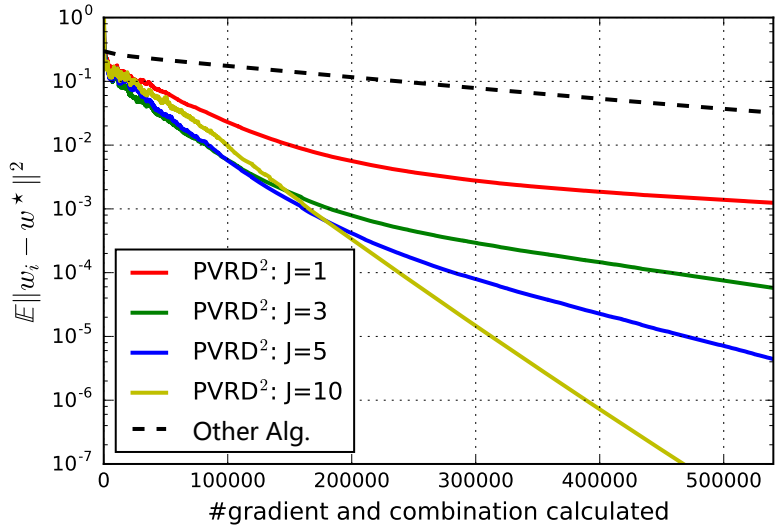

Figure 6.4: Illustration of the convergence behavior with different pipeline depth J over MNIST dataset caption. The dotted line is the algorithm proposed in [5]

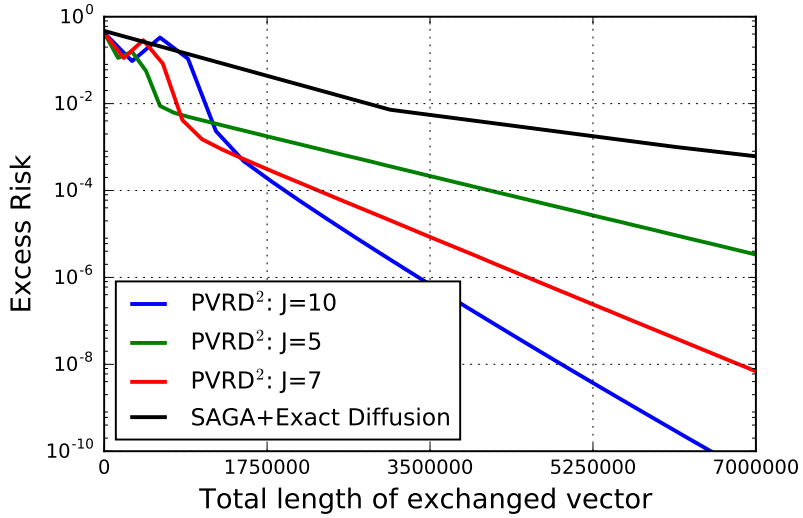

Figure 6.5: Comparison of communication vector length between PVRD^2 and SAGA+Exact Diffusion over CIFAR-10 Dataset.

and for the other columns where $c \neq \gamma_n$, we have

$$\nabla_{W_k[:,c]} J_n(W) = \mathrm{p}rob(z_{n,k}, c) h_{n,k} + \rho W_k[:, c] \tag{6.67}$$

Thus, the only modification that is necessary to our algorithm is changing from sending a scalar $z_{n,k}$ into sending a vector $z_{n,k}$. In the simulation, we compare this implementation it with SAGA+Exact diffusion as proposed in [80]. In the simulation, as plotted in Fig. 6.5, we set $K = 10$, $J = 10$, $\rho = 1e-4$ and mini-batch is $B = 10$. In Fig. 6.5, the curve is plotted based on the total number of communicated length versus the excess risk, i.e., $J(w_i) - J(w^\star)$. More specifically, for PVRD^2 algorithm, at every iteration, each edge of the network will communicate a length of $J \times C \times B = 1000$. Meanwhile, for Exact Diffusion, it needs $32 \times 32 \times 3 \times 10 = 30720$. Hence, from Fig. 6.5, it is not surprising to find that PVRD^2 is more communication-efficient.

Lastly, we provide a simulation to show the influence of network structure on algorithm performance in Fig. 6.6. The approach we utilized to generate the network is the random geometric graph model, which places N nodes uniformly at random in the unit square and two nodes are joined by an edge if the Euclidean distance between the nodes is less than some radius threshold [135]. The simulation problem is the same as the previous MNIST problem except we distribute the feature over 28 agents. We generate four network topologies, whose Euclidean distance threshold Δ are 0.3, 0.4, 0.6, and $\sqrt{2}$—full connected networks, respectively. For all network topologies, we fix the pipeline depth at 20 and mini-batch at 10. Figure 6.6 confirms the conclusion from Theorem 6.1 that the denser the topology is, the faster it converges.

6.A Proof of Theorem 6.1

6.A.1 Supporting lemma and proof sketch

Before we start the proof of the main theorem, we first give two useful lemmas here.

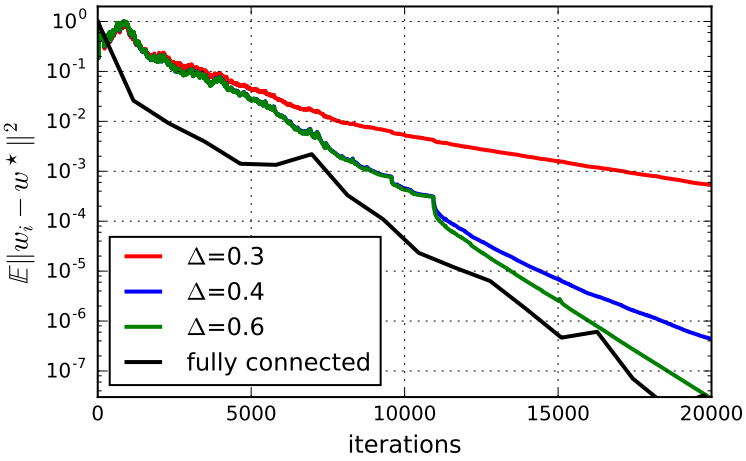

Figure 6.6: Comparison of PVRD^2 with different network topologies over MNIST Dataset.

Lemma 6.1 *For a ν-strongly convex and L-Lipschitz gradient continuous function $f(\cdot)$, the following inequality holds:*

$$(\nabla f(z) - \nabla f(y))^{\mathsf{T}}(x - z) \leq \frac{L}{2}\|x - y\|^2 - \frac{\nu}{2}\|z - y\|^2 - \frac{\nu}{2}\|x - z\|^2 \tag{6.68}$$

Proof: From the convexity and Lipschitz conditions, we have:

$$f(x) \leq f(y) + \nabla f(y)^{\mathsf{T}}(x - y) + \frac{L}{2}\|x - y\|^2 \tag{6.69}$$

$$f(z) \geq f(y) + \nabla f(y)^{\mathsf{T}}(z - y) + \frac{\nu}{2}\|z - y\|^2 \tag{6.70}$$

$$f(x) \geq f(z) + \nabla f(z)^{\mathsf{T}}(x - z) + \frac{\nu}{2}\|x - z\|^2 \tag{6.71}$$

Combining these inequalities, we get:

$$\begin{aligned} f(x) - f(z) \leq & f(y) + \nabla f(y)^{\mathsf{T}}(x - y) + \frac{L}{2}\|x - y\|^2 \\ & - f(y) - \nabla f(y)^{\mathsf{T}}(z - y) - \frac{\nu}{2}\|z - y\|^2 \\ \leq & \nabla f(y)^{\mathsf{T}}(x - z) + \frac{L}{2}\|x - y\|^2 - \frac{\nu}{2}\|z - y\|^2 \end{aligned} \tag{6.72}$$

$$f(x) - f(z) \geq \nabla f(z)^{\mathsf{T}}(x - z) + \frac{\nu}{2}\|x - z\|^2 \tag{6.73}$$

Combining the above two inequalities and rearranging terms, we establish (6.68). ■

Lemma 6.2 *Consider the inequality recursion for non-negative numbers α_i:*

$$\alpha_{i+1} \leq \lambda \alpha_i + \rho^i C \tag{6.74}$$

where $1 > \rho > \lambda > 0$. The sequence α_i converges to zero according to

$$\alpha_{i+1} \leq \rho^{i+1} \left(\alpha_0 + \frac{1}{\rho - \lambda} C \right) \tag{6.75}$$

Proof: Iterating, we get

$$\begin{aligned}
\alpha_{i+1} \leq & \lambda^2 \alpha_{i-1} + \lambda \rho^{i-1} C + \rho^i C \\
\leq & \lambda^3 \alpha_{i-2} + \lambda^2 \rho^{i-2} C + \lambda \rho^{i-1} C + \rho^i C \\
& \vdots \\
\leq & \lambda^{i+1} a_0 + \sum_{j=0}^{i} \lambda^j \rho^{i-d} C \\
\leq & \lambda^{i+1} \alpha_0 + \frac{\rho^i}{1 - \lambda/\rho} C \\
\leq & \rho^{i+1} \left(\alpha_0 + \frac{1}{\rho - \lambda} C \right)
\end{aligned} \tag{6.76}$$

■

The procedure of proving the linear convergence in Theorem 6.1 consists of two steps. The first step is to construct an error recursion for $\mathbb{E}\|\boldsymbol{w}_i - w^\star\|^2$. However, because of many auxiliary variables, eventually, we establish six recursions— (6.97), (6.100), (6.104), (6.105), (6.107), and (6.110). Note that a simple energy or Lyapunov function cannot establish the linear convergence easily due to the delay introduced by the pipeline strategy. Instead, we employ a mathematical induction method. After assuming the quantity $\mathbb{E}\|\boldsymbol{w}_i - \boldsymbol{w}_{i-1}\|^2$ converges linearly, the six error recursions will have the same format as (6.74) in Lemma 6.2. So that we conclude the proof by showing $\mathbb{E}\|\boldsymbol{w}_i - \boldsymbol{w}_{i-1}\|^2$ converges linearly in view of Lemma 6.2.

6.A.2 Error Recursion

In order to shorten the notation, we let $d = J-1$, such that $\boldsymbol{n}' = \boldsymbol{n}_{i-d}$. First, we rewrite the main recursion in vector notation:

$$\boldsymbol{w}_{i+1} = \boldsymbol{w}_i - \mu\Big(\big[\nabla_z Q(\mathcal{Z}^{(J)}_{\boldsymbol{n}',i-d};\gamma_{\boldsymbol{n}'}) - \nabla_z Q(\mathcal{U}^i_{\boldsymbol{n}'};\gamma_{\boldsymbol{n}'})\big]h_{\boldsymbol{n}'} + \frac{1}{N}\sum_{n=1}^{N}\nabla_z Q(\mathcal{U}^i_n;\gamma_n)h_n + \nabla_w r(\boldsymbol{w}_i)\Big) \tag{6.77}$$

where:

$$\mathcal{Z}^{(J)}_{\boldsymbol{n}',i-d} \triangleq col\{\boldsymbol{z}^{(J)}_{\boldsymbol{n}',1}, \boldsymbol{z}^{(J)}_{\boldsymbol{n}',2}, \cdots, \boldsymbol{z}^{(J)}_{\boldsymbol{n}',K}\} \in \mathbb{R}^{K\times 1} \tag{6.78}$$

$$\mathcal{U}^i_{\boldsymbol{n}'} \triangleq col\{\boldsymbol{u}^i_{\boldsymbol{n}',1}, \boldsymbol{u}^i_{\boldsymbol{n}',2}, \cdots, \boldsymbol{u}^i_{\boldsymbol{n}',K}\} \in \mathbb{R}^{K\times 1} \tag{6.79}$$

$$\nabla_z Q(\mathcal{Z}^{(J)}_{\boldsymbol{n}',i-d};\gamma_{\boldsymbol{n}'}) \triangleq blockdiag\{\nabla_z Q(\boldsymbol{z}^{(J)}_{\boldsymbol{n}',1};\gamma_{\boldsymbol{n}'})I_{M_1}, \cdots, \nabla_z Q(\boldsymbol{z}^{(J)}_{\boldsymbol{n}',K};\gamma_{\boldsymbol{n}'})I_{M_K}\} \in \mathbb{R}^{M\times M} \tag{6.80}$$

$$\nabla_z Q(\mathcal{U}^i_{\boldsymbol{n}'};\gamma_{\boldsymbol{n}'}) \triangleq blockdiag\{\nabla_z Q(\boldsymbol{u}^i_{\boldsymbol{n}',1};\gamma_{\boldsymbol{n}'})I_{M_1}, \cdots, \nabla_z Q(\boldsymbol{u}^i_{\boldsymbol{n}',K};\gamma_{\boldsymbol{n}'})I_{M_K}\} \in \mathbb{R}^{M\times M} \tag{6.81}$$

In our notation convention, all calligraphic symbols represent stacked vectors or matrices over agents. Notice the subscript $i-d$ in $\mathcal{Z}$ is used to represents the $\boldsymbol{z}$ is computed from $\boldsymbol{w}_{i-d}$ instead of data or agent index. Introducing the error quantity $\widetilde{\boldsymbol{w}}_i \triangleq w^\star - \boldsymbol{w}_i$, we obtain the error recursion:

$$\widetilde{\boldsymbol{w}}_{i+1} = \widetilde{\boldsymbol{w}}_i + \mu\Big(\big[\nabla_z Q(\mathcal{Z}^{(J)}_{\boldsymbol{n}',i-d};\gamma_{\boldsymbol{n}'}) - \nabla_z Q(\mathcal{U}^i_{\boldsymbol{n}'};\gamma_{\boldsymbol{n}'})\big]h_{\boldsymbol{n}'} + \frac{1}{N}\sum_{n=1}^{N}\nabla_z Q(\mathcal{U}^i_n;\gamma_n)h_n + \nabla_w r(\boldsymbol{w}^k_i)\Big) \tag{6.82}$$

We introduce the filtration $\boldsymbol{\mathcal{F}}_i$, which consists of all previous information $\boldsymbol{w}_j,\ j \le i$ and indices $\boldsymbol{n}_j,\ j \le i-J$. Then, the modified gradient step satisfies the unbiasedness property:

$$\mathbb{E}_{\boldsymbol{n}'}\left[\Big(\big[\nabla_z Q(\mathcal{Z}^{(J)}_{\boldsymbol{n}',i-d};\gamma_{\boldsymbol{n}'}) - \nabla_z Q(\mathcal{U}^i_{\boldsymbol{n}'};\gamma_{\boldsymbol{n}'})\big]h_{\boldsymbol{n}'} + \frac{1}{N}\sum_{n=1}^{N}\nabla_z Q(\mathcal{U}^i_n;\gamma_n)h_n + \nabla_w r(\boldsymbol{w}_i)\Big)\Big|\boldsymbol{\mathcal{F}}_i\right]$$
$$= \frac{1}{N}\sum_{n=1}^{N}\nabla_z Q(\mathcal{Z}^{(J)}_{n,i-d};\gamma_n)h_n + \nabla_w r(\boldsymbol{w}_i) \tag{6.83}$$

Thus, we have the following error recursion:

$$\begin{aligned}
\mathbb{E}_{\boldsymbol{n}'}[\|\widetilde{\boldsymbol{w}}_{i+1}\|^2|\boldsymbol{\mathcal{F}}_i] =&\|\widetilde{\boldsymbol{w}}_i\|^2 + \frac{2\mu}{N}\sum_{n=1}^{N}\widetilde{\boldsymbol{w}}_i^{\mathsf{T}}[\nabla_z Q(\boldsymbol{z}_{n,i-d}^{(J)};\gamma_n)h_n + \nabla_w r(\boldsymbol{w}_i)] \\
&+\mu^2\mathbb{E}\left[\Big\|\big[\nabla_z Q(\boldsymbol{z}_{\boldsymbol{n}',i-d}^{(J)};\gamma_{\boldsymbol{n}'}) - \nabla_z Q(\boldsymbol{u}_{\boldsymbol{n}'}^{i};\gamma_{\boldsymbol{n}'})\big]h_{\boldsymbol{n}'}\right. \\
&\left.+\frac{1}{N}\sum_{n=1}^{N}\nabla_z Q(u_n^i;\gamma_n)h_n + \nabla_w r(\boldsymbol{w}_i)\Big\|^2\Big|\boldsymbol{\mathcal{F}}_i\right] \\
\leq& \|\widetilde{\boldsymbol{w}}_i\|^2+\frac{2\mu}{N}\sum_{n=1}^{N}\widetilde{\boldsymbol{w}}_i^{\mathsf{T}}[\nabla_z Q(\boldsymbol{z}_{n,i-d}^{(J)};\gamma_n)h_n + \nabla_w r(\boldsymbol{w}_i)] \\
&+4\mu^2\mathbb{E}\Big\|\big[\nabla_z Q(\boldsymbol{z}_{\boldsymbol{n}',i-d}^{(J)};\gamma_{\boldsymbol{n}'})-\nabla_z Q(h_{\boldsymbol{n}'}^{\mathsf{T}}w^\star;\gamma_{\boldsymbol{n}'})I_M\big]h_{\boldsymbol{n}'}\Big\|^2 \\
&+4\mu^2\mathbb{E}\Big\|\big[\nabla_z Q(\boldsymbol{u}_{\boldsymbol{n}'}^{i};\gamma_{\boldsymbol{n}'})-\nabla_z Q(h_{\boldsymbol{n}'}^{\mathsf{T}}w^\star;\gamma_{\boldsymbol{n}'})I_M\big]h_{\boldsymbol{n}'}\Big\|^2 \\
&+4\mu^2\frac{1}{N}\sum_{n=1}^{N}\Big\|[\nabla_z Q(u_n^i;\gamma_n)-\nabla_z Q(h_n^{\mathsf{T}}w^\star;\gamma_n)I_M]h_n\Big\|^2 \\
&+4\mu^2\Big\|\nabla_w r(\boldsymbol{w}_i) - \nabla_w r(w^\star)\Big\|^2
\end{aligned} \tag{6.84}$$

where we appealed to Jensen's inequality and I_M is the $M\times M$ identity matrix. Next, we focus on the cross term:

$$\begin{aligned}
\frac{1}{N}\sum_{n=1}^{N}\widetilde{\boldsymbol{w}}_i^{\mathsf{T}}[\nabla_z Q(\boldsymbol{z}_{n,i-d}^{(J)};\gamma_n)h_n + \nabla_w r(\boldsymbol{w}_i)] =&\frac{1}{N}\sum_{n=1}^{N}\widetilde{\boldsymbol{w}}_i^{\mathsf{T}}[\nabla_z Q(\boldsymbol{z}_{n,i-d}^{(J)};\gamma_n)h_n + \nabla_w r(\boldsymbol{w}_i) - \nabla_w J(w^\star)] \\
\overset{(a)}{=}&\widetilde{\boldsymbol{w}}_i^{\mathsf{T}}\left[\frac{1}{N}\sum_{n=1}^{N}\nabla_z Q(\bar{\boldsymbol{z}}_{n,i-d};\gamma_n)h_n + \nabla_w r(\boldsymbol{w}_i)-\nabla_w J(w^\star)\right] \\
&+\frac{1}{N}\sum_{n=1}^{N}\widetilde{\boldsymbol{w}}_i^{\mathsf{T}}[\nabla_z Q(\boldsymbol{z}_{n,i-d}^{(J)};\gamma_n)h_n-\nabla_z Q(\bar{\boldsymbol{z}}_{n,i-d};\gamma_n)h_n] \\
\overset{(b)}{=}&[\nabla_w J(\boldsymbol{w}_{i-d}) - \nabla_w J(w^\star)]^{\mathsf{T}}(w^\star - \boldsymbol{w}_i) \\
&+[\nabla_w r(\boldsymbol{w}_i) - \nabla_w r(\boldsymbol{w}_{i-d})]^{\mathsf{T}}(w^\star - \boldsymbol{w}_i) \\
&+\frac{1}{N}\sum_{n=1}^{N}\widetilde{\boldsymbol{w}}_i^{\mathsf{T}}[\nabla_z Q(\boldsymbol{z}_{n,i-d}^{(J)};\gamma_n)h_n-\nabla_z Q(\bar{\boldsymbol{z}}_{n,i-d};\gamma_n)h_n]
\end{aligned} \tag{6.85}$$

where we define

$$\bar{\boldsymbol{z}}_{n,i-d} \triangleq \frac{1}{K}\mathbb{1}_K^{\mathsf{T}}\boldsymbol{z}_{n,i-d}^{(0)} \in \mathbb{R} \tag{6.86}$$

$$\bar{z}_{n,i-d} \triangleq \mathrm{col}\{\bar{\boldsymbol{z}}_{n,i-d},\cdots,\bar{\boldsymbol{z}}_{n,i-d}\} = \bar{\boldsymbol{z}}_{n,i-d}\mathbb{1}_K \in \mathbb{R}^{K\times 1} \tag{6.87}$$

In step (a), we add and subtract the same term about $\nabla_z Q(\bar{z}_{n,i-d};\gamma_n)h_n$; In step (b), we exploit the property that $\bar{\boldsymbol{z}}_{n,i-d} = h_n^{\mathsf{T}}\boldsymbol{w}_{i-d}$. Indeed, note that

$$\begin{aligned}
\bar{\boldsymbol{z}}_{n,i-d} =& \frac{1}{K}\mathbb{1}_K^{\mathsf{T}}\boldsymbol{z}_{n,i-d}^{(0)} \\
\overset{(6.48)}{=}& \frac{1}{K}\sum_{k=1}^{K}(Kh_{n,k}^{\mathsf{T}}\boldsymbol{w}_{i-d,k} + \boldsymbol{u}_{n,k}^{i-d} - \boldsymbol{v}_{n,k}^{i-d}) \\
\overset{(6.32)}{=}& \sum_{k=1}^{K} h_{n,k}^{\mathsf{T}}\boldsymbol{w}_{i-d,k} \\
=& h_n^{\mathsf{T}}\boldsymbol{w}_{i-d}
\end{aligned} \tag{6.88}$$

Now, to bound the first term in (6.85), we apply the result (6.68) from Lemma 6.1 so that we immediately get

$$\begin{aligned}
[\nabla_w J(\boldsymbol{w}_{i-d}) - \nabla_w J(w^\star)]^{\mathsf{T}}(w^\star - \boldsymbol{w}_i) \leq& \frac{L+\eta}{2}\|\boldsymbol{w}_i - \boldsymbol{w}_{i-d}\|^2 - \frac{\nu}{2}\|w^\star - \boldsymbol{w}_{i-d}\|^2 - \frac{\nu}{2}\|\boldsymbol{w}_i - w^\star\|^2 \\
\leq& \frac{(L+\eta)d}{2}\sum_{j=i-d}^{i-1}\|\boldsymbol{w}_{j+1} - \boldsymbol{w}_j\|^2 - \frac{\nu}{2}\|\widetilde{\boldsymbol{w}}_{i-d}\|^2 - \frac{\nu}{2}\|\widetilde{\boldsymbol{w}}_i\|^2
\end{aligned} \tag{6.89}$$

and we have

$$\begin{aligned}
[\nabla_w r(\boldsymbol{w}_i) - \nabla_w r(\boldsymbol{w}_{i-d})]^{\mathsf{T}}(w^\star - \boldsymbol{w}_i) \leq& \frac{\eta\epsilon}{2}\|\boldsymbol{w}_i - \boldsymbol{w}_{i-d}\|^2 + \frac{1}{2\epsilon}\|\widetilde{\boldsymbol{w}}_i\|^2 \\
\leq& \frac{\eta d}{\nu}\sum_{j=i-d}^{i-1}\|\boldsymbol{w}_{j+1} - \boldsymbol{w}_j\|^2 + \frac{\nu}{4}\|\widetilde{\boldsymbol{w}}_i\|^2
\end{aligned} \tag{6.90}$$

where the first inequality is due to Young's inequality $a^{\mathsf{T}}b \leq \frac{\epsilon}{2}\|a\|^2 + \frac{1}{2\epsilon}\|b\|^2$ and ϵ can be any positive number. And in second step we set $\epsilon = \frac{2}{\nu}$.

For the third term in (6.85), we obtain:

$$\begin{aligned}\widetilde{\boldsymbol{w}}_i^{\mathsf{T}}[\nabla_z Q(\boldsymbol{z}_{n,i-d}^{(J)};\gamma_n)h_n-\nabla_z Q(\bar{\boldsymbol{z}}_{n,i-d};\gamma_n)h_n]\leq&\frac{\epsilon}{2}\|\widetilde{\boldsymbol{w}}_i\|^2+\frac{1}{2\epsilon}\Big\|\nabla Q(\boldsymbol{z}_{n,i-d}^{(J)};\gamma_n)h_n-\nabla Q(\bar{\boldsymbol{z}}_{n,i-d};\gamma_n)h_n\Big\|^2\\ \leq&\frac{\epsilon}{2}\|\widetilde{\boldsymbol{w}}_i\|^2+\frac{\delta^2}{2\epsilon}\Big\|\boldsymbol{z}_{n,i-d}^{(J)}-\bar{\boldsymbol{z}}_{n,i-d}\Big\|^2\|h_n\|^2 \qquad (6.91)\end{aligned}$$

Next, for the remaining terms in the main recursion (6.84), we have the following bounds through the Lipschitz condition:

$$\begin{aligned}&\mathbb{E}_{\boldsymbol{n}'}\Big\|\big[\nabla_z Q(\boldsymbol{z}_{\boldsymbol{n}',i-d}^{(J)};\gamma_{\boldsymbol{n}'})-\nabla_z Q(h_{\boldsymbol{n}'}^{\mathsf{T}}w^\star;\gamma_{\boldsymbol{n}'}I_M)\big]h_{\boldsymbol{n}'}\Big\|^2\\ &\quad\leq 2\mathbb{E}_{\boldsymbol{n}'}\Big\|\big[\nabla_z Q(\boldsymbol{z}_{\boldsymbol{n}',i-d}^{(J)};\gamma_{\boldsymbol{n}'})-\nabla_z Q(\bar{\boldsymbol{z}}_{\boldsymbol{n}',i-d};\gamma_{\boldsymbol{n}'})\big]h_{\boldsymbol{n}'}\Big\|^2\\ &\qquad+2\mathbb{E}_{\boldsymbol{n}'}\Big\|\big[\nabla_z Q(h_{\boldsymbol{n}'}^{\mathsf{T}}\boldsymbol{w}_{i-d};\gamma_{\boldsymbol{n}'})-\nabla_z Q(h_{\boldsymbol{n}'}^{\mathsf{T}}w^\star;\gamma_{\boldsymbol{n}'})\big]h_{\boldsymbol{n}'}\Big\|^2\\ &\quad\leq\frac{2\delta^2}{N}\sum_{n=1}^N\|\boldsymbol{z}_{n,i-d}^{(J)}-\bar{\boldsymbol{z}}_{n,i-d}\|^2\|h_n\|^2+2L^2\|\widetilde{\boldsymbol{w}}_{i-d}\|^2 \qquad (6.92)\end{aligned}$$

and

$$\begin{aligned}\mathbb{E}_{\boldsymbol{n}'}\Big\|\big[\nabla_z Q(\boldsymbol{u}_{\boldsymbol{n}'}^i;\gamma_{\boldsymbol{n}'})-\nabla_z Q(h_{\boldsymbol{n}'}^{\mathsf{T}}w^\star;\gamma_{\boldsymbol{n}'})I_M\big]h_{\boldsymbol{n}'}\Big\|^2&\leq\mathbb{E}\,\|\nabla_z Q(\boldsymbol{u}_{\boldsymbol{n}'}^i;\gamma_{\boldsymbol{n}'})-\nabla_z Q(h_{\boldsymbol{n}'}^{\mathsf{T}}w^\star;\gamma_{\boldsymbol{n}'})I_M\|^2\|h_{\boldsymbol{n}'}\|^2\\ &\leq\frac{\delta^2}{N}\sum_{n=1}^N\|\boldsymbol{u}_n^i-h_n^{\mathsf{T}}w^\star\mathbb{1}_K\|^2\|h_n\|^2 \qquad (6.93)\end{aligned}$$

and

$$\frac{1}{N}\sum_{n=1}^N\Big\|[\nabla_z Q(\boldsymbol{u}_n^i;\gamma_n)-\nabla_z Q(h_n^{\mathsf{T}}w^\star;\gamma_n)I_M]h_n\Big\|^2\leq\frac{\delta^2}{N}\sum_{n=1}^N\|\boldsymbol{u}_n^i-h_n^{\mathsf{T}}w^\star\mathbb{1}_K\|^2\|h_n\|^2. \qquad (6.94)$$

and

$$4\mu^2\Big\|[\nabla_w r(\boldsymbol{w}_i)-\nabla_w r(w^\star)\Big\|^2\leq 4\mu^2\eta^2\|\widetilde{\boldsymbol{w}}_i\|^2 \qquad (6.95)$$

Substituting (6.89) – (6.95) into the main error recursion (6.84), we have:

$$\begin{aligned}\mathbb{E}_{\boldsymbol{n}'}[\|\widetilde{\boldsymbol{w}}_{i+1}\|^2|\boldsymbol{\mathcal{F}}_i] \leq & (1+4\mu^2\eta^2)\|\widetilde{\boldsymbol{w}}_i\|^2+\mu d(L+\eta+\frac{2\eta}{\nu})\sum_{j=i-d}^{i-1}\|\boldsymbol{w}_{j+1}-\boldsymbol{w}_j\|^2\\ &-\nu\mu\|\widetilde{\boldsymbol{w}}_{i-d}\|^2-\frac{\mu\nu}{2}\|\widetilde{\boldsymbol{w}}_i\|^2+\mu\epsilon\|\widetilde{\boldsymbol{w}}_i\|^2\\ &+\left(\frac{\mu\delta^2}{\epsilon}+8\mu^2\delta^2\right)\frac{1}{N}\sum_{n=1}^{N}\|\mathcal{z}_{n,i-d}^{(J)}-\bar{\mathcal{z}}_{n,i-d}\|^2\|h_n\|^2\\ &+8\mu^2L^2\|\widetilde{\boldsymbol{w}}_{i-d}\|^2+\frac{8\mu^2\delta^2}{N}\sum_{n=1}^{N}\|\mathcal{u}_n^i-h_n^{\mathsf{T}}w^{\star}\mathbb{1}_K\|^2\|h_n\|^2 \end{aligned} \tag{6.96}$$

Let $\epsilon=\frac{\nu}{6}$ and denote $L'\triangleq L+\eta+\frac{2\eta}{\nu}$. Rearranging terms we get:

$$\begin{aligned}\mathbb{E}_{\boldsymbol{n}'}[\|\widetilde{\boldsymbol{w}}_{i+1}\|^2|\boldsymbol{\mathcal{F}}_i] \leq & (1-\mu\nu/3+4\mu^2\eta^2)\|\widetilde{\boldsymbol{w}}_i\|^2+\mu dL'\sum_{j=i-d}^{i-1}\|\boldsymbol{w}_{j+1}-\boldsymbol{w}_j\|^2\\ &-(\nu\mu-8\mu^2L^2)\|\widetilde{\boldsymbol{w}}_{i-d}\|^2+\frac{8\mu\delta^2}{\nu N}\sum_{n=1}^{N}\left\|\mathcal{z}_{n,i-d}^{(J)}-\bar{\mathcal{z}}_{n,i-d}\right\|^2\|h_n\|^2\\ &+\frac{8\delta^2\mu^2}{N}\sum_{n=1}^{N}\|\mathcal{u}_n^i-h_n^{\mathsf{T}}w^{\star}\mathbb{1}_K\|^2\|h_n\|^2 \end{aligned} \tag{6.97}$$

where we relaxed $\frac{6\mu\delta^2}{\nu}+8\mu^2\delta^2$ into the upper bound $\frac{8\mu\delta^2}{\nu}$, which requires

$$\frac{8\mu\delta^2}{\nu}\geq\frac{6\mu\delta^2}{\nu}+8\mu^2\delta^2\Longleftrightarrow\mu\leq\frac{1}{4\nu} \tag{6.98}$$

So far we have established (6.97). Next, we seek a recursion for the inner difference:

$$\begin{aligned}\mathbb{E}\,[\|\boldsymbol{w}_{i+1}-\boldsymbol{w}_i\|^2|\boldsymbol{\mathcal{F}}_i] = & \mu^2\mathbb{E}_{\boldsymbol{n}'}\Bigg[\Big\|\nabla_z Q(\mathcal{z}_{\boldsymbol{n}',i-d}^{(J)};\gamma_{\boldsymbol{n}'})-\nabla_z Q(\mathcal{u}_{\boldsymbol{n}'}^i;\gamma_{\boldsymbol{n}'})\big]h_{\boldsymbol{n}'}\\ &+\frac{1}{N}\sum_{n=1}^{N}\nabla_z Q(\mathcal{u}_n^i;\gamma_n)h_n+\nabla_w r(\boldsymbol{w}_i)\Big\|^2\Big|\boldsymbol{\mathcal{F}}_i\Bigg] \end{aligned} \tag{6.99}$$

Applying Jensen's inequality and combining results (6.92), (6.93), (6.94), and (6.95) we get:

$$\begin{aligned}\mathbb{E}\left[\|\boldsymbol{w}_{i+1}-\boldsymbol{w}_i\|^2|\boldsymbol{\mathcal{F}}_i\right] \leq & \frac{8\mu^2\delta^2}{N}\sum_{n=1}^{N}\|\mathcal{z}_{n,i-d}^{(J)}-\bar{\mathcal{z}}_{n,i-d}\|^2\|h_n\|^2 \\ & + \frac{8\mu^2\delta^2}{N}\sum_{n=1}^{N}\|\mathcal{u}_n^i-h_n^{\mathsf{T}}w^\star\mathbb{1}_K\|^2\|h_n\|^2 \\ & + 8\mu^2L^2\|\widetilde{\boldsymbol{w}}_{i-d}\|^2+4\mu^2\eta^2\|\widetilde{\boldsymbol{w}}_i\|^2 \end{aligned} \tag{6.100}$$

Then, we derive the result:

$$\begin{aligned}\|\mathcal{z}_{i-d,n}^{(J)}-\bar{\mathcal{z}}_{i-d,n}\|^2 = & \|\mathcal{z}_{i-d,n}^{(J)}-\frac{1}{K}\mathbb{1}\mathbb{1}^{\mathsf{T}}\mathcal{z}_{i-d,n}^{(J)}\|^2 \\ \overset{(6.45)}{=} & \left\|\left(I-\frac{1}{K}\mathbb{1}\mathbb{1}^{\mathsf{T}}\right)A^J(KH_n^{\mathsf{T}}\boldsymbol{w}_{i-d}+\mathcal{u}_n^i-\mathcal{v}_n^i)\right\|^2 \\ \overset{(a)}{=} & \left\|\left(A^J-\frac{1}{K}\mathbb{1}\mathbb{1}^{\mathsf{T}}\right)(KH_n^{\mathsf{T}}\boldsymbol{w}_{i-d}-\mathcal{v}_n^i+\mathcal{u}_n^i-\bar{\mathcal{u}}_n^i)\right\|^2 \\ \overset{(b)}{\leq} & \lambda^J\|\mathcal{u}_n^i-\bar{\mathcal{u}}_n^i\|^2+\frac{\lambda^{2J}}{1-\lambda^J}\left\|KH_n^{\mathsf{T}}\boldsymbol{w}_{i-d}-\mathcal{v}_n^i\right\|^2 \end{aligned} \tag{6.101}$$

where A^J arises from the definition of $z_{i-d,n}^{(J)}$, which is the stacked vector of $z_{i-d,n}^{(0)}$ after J-step consensus, and where we denote:

$$H_n \triangleq \begin{bmatrix} h_{1,n} & & \\ & \ddots & \\ & & h_{K,n} \end{bmatrix} \in \mathbb{R}^{M\times K} \tag{6.102}$$

Step (a) holds because $\left(A^J-\frac{1}{K}\mathbb{1}\mathbb{1}^{\mathsf{T}}\right)\bar{\mathcal{u}}_n^i=0$, and

$$\bar{\mathcal{u}}_n^i \triangleq \frac{1}{K}\mathbb{1}\mathbb{1}_K^{\mathsf{T}}\mathcal{u}_n^i=\left(\frac{1}{K}\sum_{k=1}^{k}\boldsymbol{u}_{n,k}^i\right)\mathbb{1}\in\mathbb{R}^{K\times 1} \tag{6.103}$$

In step (b), λ is the second largest eigenvalue of A. Multiplying by $\|h_n\|^2$, taking expectation

and averaging over n:

$$\frac{1}{N}\sum_{n=1}^{N}\mathbb{E}\,\|\mathcal{z}_{i-d,n}^{(J)}-\bar{z}_{i-d,n}\|^2\|h_n\|^2\le\lambda^J\frac{1}{N}\sum_{n=1}^{N}\mathbb{E}\,\|\mathcal{u}_n^i-\bar{u}_n^i\|^2\|h_n\|^2 + \frac{\lambda^{2J}}{1-\lambda^J}\frac{1}{N}\sum_{n=1}^{N}\mathbb{E}\,\left\|KH_n^{\mathsf{T}}\boldsymbol{w}_{i-d}-\nu_n^i\right\|^2\|h_n\|^2 \quad (6.104)$$

Next, using the uniform sampling property, we establish a similar recursion for $\mathcal{u}_n^i$:

$$\begin{aligned}&\frac{1}{N}\sum_{n=1}^{N}\mathbb{E}_{\boldsymbol{n}'}\left[\|\mathcal{u}_n^{i+1}-\bar{u}_n^{i+1}\|^2\|h_n\|^2\,\middle|\,\boldsymbol{\mathcal{F}}_i\right]\\ &\overset{(6.50)}{=}\frac{N-1}{N^2}\sum_{n=1}^{N}\|\mathcal{u}_n^i-\bar{u}_n^i\|^2\|h_n\|^2+\frac{1}{N^2}\sum_{n=1}^{N}\|\mathcal{z}_{n,i-d}^{(J)}-\bar{z}_{n,i-d}\|^2\|h_n\|^2\end{aligned} \quad (6.105)$$

Similarly, we get:

$$\begin{aligned}\mathbb{E}_{\boldsymbol{n}'}\left[\|\mathcal{u}_n^i-h_n^{\mathsf{T}}w^\star\mathbb{1}\|^2|\boldsymbol{\mathcal{F}}_i\right]&\overset{(6.50)}{=}\frac{1}{N}\|\mathcal{z}_{n,i-d}^{(J)}-h_n^{\mathsf{T}}w^\star\mathbb{1}\|^2+\frac{N-1}{N}\|\mathcal{u}_n^{i-1}-h_n^{\mathsf{T}}w^\star\mathbb{1}\|^2\\ &\le\frac{2}{N}\|\mathcal{z}_{n,i-d}^{(J)}-\bar{z}_{n,i-d}\|^2+\frac{2}{N}\|\bar{z}_{n,i-d}-h_n^{\mathsf{T}}w^\star\mathbb{1}\|^2\\ &\quad+\frac{N-1}{N}\sum_{n=1}^{N}\|\mathcal{u}_n^{i-1}-h_n^{\mathsf{T}}w^\star\mathbb{1}\|^2\\ &\overset{(6.88)}{=}\frac{2}{N}\|\mathcal{z}_{n,i-d}^{(J)}-\bar{z}_{n,i-d}\|^2+\frac{2}{N}\|h_n^{\mathsf{T}}\widetilde{\boldsymbol{w}}_{i-d}\|^2\\ &\quad+\frac{N-1}{N}\|\mathcal{u}_n^{i-1}-h_n^{\mathsf{T}}w^\star\mathbb{1}\|^2\end{aligned} \quad (6.106)$$

Multiplying $\|h_n\|^2$, taking expectation over filtration and averaging over n:

$$\frac{1}{N}\sum_{n=1}^{N}\mathbb{E}\,\|\mathcal{u}_n^i-h_n^{\mathsf{T}}w^\star\mathbb{1}\|^2\|h_n\|^2\le\frac{2}{N^2}\sum_{n=1}^{N}\mathbb{E}\,\|\mathcal{z}_{n,i-d}^{(J)}-\bar{z}_{n,i-d}\|^2\|h_n\|^2+\frac{2}{N^2}\sum_{n=1}^{N}\mathbb{E}\,\|h_n\|^4\|\widetilde{\boldsymbol{w}}_{i-d}\|^2 + \frac{N-1}{N^2}\sum_{n=1}^{N}\mathbb{E}\,\|\mathcal{u}_n^{i-1}-h_n^{\mathsf{T}}w^\star\mathbb{1}\|^2\|h_n\|^2 \quad (6.107)$$

Lastly, we obtain:

$$\mathbb{E}_{\boldsymbol{n}'}\left[\|KH_n^{\mathsf{T}}\boldsymbol{w}_{i-d}-\nu_n^i\|^2|\boldsymbol{\mathcal{F}}_i\right]=\frac{1}{N}\|KH_n(\boldsymbol{w}_{i-d}-\boldsymbol{w}_{i-d-1})\|^2+\frac{N-1}{N}\|KH_n^{\mathsf{T}}\boldsymbol{w}_{i-d}-\nu_n^{i-1}\|^2$$

$$\leq \frac{1}{N}\|KH_n(\boldsymbol{w}_{i-d}-\boldsymbol{w}_{i-d-1})\|^2+\frac{N-1}{tN}\|KH_n^{\mathsf{T}}\boldsymbol{w}_{i-d-1}-\nu_n^{i-1}\|^2$$
$$+\frac{N-1}{(1-t)N}\|KH_n^{\mathsf{T}}(\boldsymbol{w}_{i-d}-\boldsymbol{w}_{i-d-1})\|^2 \tag{6.108}$$

If we choose $t=\frac{N-1}{N-1/2}$, then

$$\mathbb{E}_{\boldsymbol{n}'}\left[\|KH_n^{\mathsf{T}}\boldsymbol{w}_{i-d}-\nu_n^i\|^2|\boldsymbol{\mathcal{F}}_i\right]\leq\frac{1+2(N-1)^2}{N}\|KH_n(\boldsymbol{w}_{i-d}-\boldsymbol{w}_{i-d-1})\|^2$$
$$+\frac{N-1/2}{N}\|KH_n^{\mathsf{T}}\boldsymbol{w}_{i-d-1}-\nu_n^{i-1}\|^2 \tag{6.109}$$

Notice that $\|H_n\|^2\leq\|H_n\|_F^2=\|h_n\|^2$. After multiplying by $\|h_n\|^2$, taking expectation over the filtration and averaging over n, we have

$$\frac{1}{N}\sum_{n=1}^{N}\mathbb{E}\,\|KH_n^{\mathsf{T}}\boldsymbol{w}_{i-d}-\nu_n^i\|^2\|h_n\|^2\leq\frac{K^2+2K^2(N-1)^2}{N^2}\sum_{n=1}^{N}\mathbb{E}\,\|(\boldsymbol{w}_{i-d}-\boldsymbol{w}_{i-d-1})\|^2\|h_n\|^4$$
$$+\frac{N-1/2}{N^2}\sum_{n=1}^{N}\mathbb{E}\,\|KH_n^{\mathsf{T}}\boldsymbol{w}_{i-d-1}-\nu_n^{i-1}\|^2 \tag{6.110}$$

To simplify the notation, we introduce

$$a_i\ \triangleq\ \mathbb{E}\,\|\widetilde{\boldsymbol{w}}_i\|^2 \tag{6.111}$$
$$b_i\ \triangleq\ \mathbb{E}\,\|\boldsymbol{w}_i-\boldsymbol{w}_{i-1}\|^2 \tag{6.112}$$
$$c_i\ \triangleq\ \frac{1}{N}\sum_{n=1}^{N}\mathbb{E}\Big\|\boldsymbol{z}_{n,i-d}^{(J)}-\bar{\boldsymbol{z}}_{n,i-d}\|^2\|h_n\|^2 \tag{6.113}$$
$$d_i\ \triangleq\ \frac{1}{N}\sum_{n=1}^{N}\mathbb{E}\,\|u_n^i-\bar{u}_n^i\|^2\|h_n\|^2 \tag{6.114}$$
$$e_i\ \triangleq\ \frac{1}{N}\sum_{n=1}^{N}\mathbb{E}\,\|u_n^i-h_n^{\mathsf{T}}w^\star\mathbb{1}\|^2\|h_n\|^2 \tag{6.115}$$
$$f_i\ \triangleq\ \frac{1}{N}\sum_{n=1}^{N}\mathbb{E}\,\|KH_n^{\mathsf{T}}\boldsymbol{w}_{i-d}-\nu_n^i\|^2\|h_n\|^2 \tag{6.116}$$
$$h^4\ \triangleq\ \frac{1}{N}\sum_{n=1}^{N}\|h_n\|^4 \tag{6.117}$$

Using the above notation, we have established so far the following six recursions from (6.97), (6.100), (6.104), (6.105), (6.107), (6.110):

$$\begin{aligned} a_{i+1} \leq (1-\mu\nu/3+4\mu^2\eta^2)a_i + \mu d L' \sum_{j=i-d}^{i-1} b_{j+1} \\ - (\nu\mu - 8\mu^2 L^2)a_{i-d} + \frac{8\mu\delta^2}{\nu}c_i + 8\mu^2\delta^2 e_i \end{aligned} \tag{6.118}$$

$$b_{i+1} \leq 8\mu^2\delta^2 c_i + 8\mu^2\delta^2 e_i + 8\mu^2 L^2 a_{i-d} + 4\mu^2\eta^2 a_i \tag{6.119}$$

$$c_i \leq \lambda^J d_i + \frac{\lambda^{2J}}{1-\lambda^J} f_i \tag{6.120}$$

$$d_{i+1} = \frac{1}{N}c_i + \frac{N-1}{N}d_i \tag{6.121}$$

$$e_i = \frac{2}{N}h^4 a_{i-d} + \frac{2}{N}c_i + \frac{N-1}{N}e_{i-1} \tag{6.122}$$

$$f_i \leq \frac{N-1/2}{N}f_{i-1} + \frac{K^2+2K^2(N-1)^2}{N}h^4 b_{i-d} \tag{6.123}$$

6.A.3 Linear Convergence of Error Recursion

Finally, we establish convergence of the PVRD^2 algorithm using Lemma 6.2 and mathematical induction. Assuming $b_i \leq \rho^i C_0$ for $j \leq i-1$, where we set

$$\rho = \max\left(1 - \frac{1-\lambda^J}{2N}, 1-\mu\nu/5\right) \tag{6.124}$$

Then, from (6.123), we have

$$\begin{aligned} f_i \leq & \frac{N-1/2}{N}f_{i-1} + \frac{K^2+K^2(N-1)^2}{N\rho^{d-1}}h^4 C_0\rho^{i-1} \\ \leq & \rho^i\left(f_0 + \frac{1}{\rho - \frac{N-1/2}{N}}\frac{K^2+2K^2(N-1)^2}{N\rho^{d-1}}h^4 C_0\right) \\ \triangleq & \rho^i C_1 \end{aligned} \tag{6.125}$$

Substituting (6.120) into (6.121), we have

$$d_{i+1} \leq \frac{\lambda^J}{N}d_i + \frac{\lambda^{2J}}{N(1-\lambda^J)}f_i + \frac{N-1}{N}d_i \tag{6.126}$$

$$
\begin{aligned}
=&\frac{\lambda^J+N-1}{N}d_i+\frac{\lambda^{2J}}{N(1-\lambda^J)}f_i\\
\leq&\left(1-\frac{1-\lambda^J}{N}\right)d_i+\frac{\lambda^{2J}}{N(1-\lambda^J)}\rho^iC_1\\
\leq&\rho^{i+1}\left(d_0+\frac{1}{\rho-1+\frac{1-\lambda^J}{N}}\frac{\lambda^{2J}}{N(1-\lambda^J)}C_1\right)\\
\triangleq&\ \rho^{i+1}C_2
\end{aligned}
\tag{6.127}
$$

From (6.120):

$$
\begin{aligned}
c_i\leq&\lambda^J\rho^iC_2+\frac{\lambda^{2J}}{1-\lambda^J}\rho^iC_1\\
=&\rho^i\left(\lambda^JC_2+\frac{\lambda^{2J}}{1-\lambda^J}C_1\right)\\
\triangleq&\ \rho^iC_3
\end{aligned}
\tag{6.128}
$$

Adding (6.118) and (6.122) with a positive coefficient γ, we have

$$
\begin{aligned}
a_{i+1}+\gamma e_i\leq&(1-\mu\nu/3+4\mu^2\eta^2)a_i+\mu dL'\sum_{j=i-d}^{i-1}b_{j+1}\\
&-(\nu\mu-8\mu^2L^2)a_{i-d}+\frac{8\mu\delta^2}{\nu}c_i+8\mu^2\delta^2e_i\\
&+\frac{2\gamma}{N}h^4a_{i-d}+\frac{2\gamma}{N}c_i+\gamma\frac{N-1}{N}e_{i-1}
\end{aligned}
\tag{6.129}
$$

We require that $\mu\leq\frac{\nu}{48\eta^2}$ such that

$$
1-\mu\nu/3+4\mu^2\eta^2\leq1-\mu\nu/4 \tag{6.130}
$$

Rearranging terms:

$$
\begin{aligned}
a_{i+1}+(\gamma-8\mu^2\delta^2)e_i\leq&(1-\mu\nu/4)a_i+\mu dL'\sum_{j=i-d}^{i-1}b_{j+1}-\left(\nu\mu-8\mu^2L^2-\frac{2\gamma}{N}h^4\right)a_{i-d}\\
&+\left(\frac{4\mu\delta^2}{\nu}+\frac{2\gamma}{N}\right)c_i+\gamma\frac{N-1}{N}e_{i-1}
\end{aligned}
$$

$$
\begin{aligned}
&= (1-\mu\nu/4)\left(a_i + \gamma\frac{N-1}{(1-\mu\nu/4)N}e_{i-1}\right) \\
&\quad + \left(\frac{8\mu\delta^2}{\nu} + \frac{2\gamma}{N}\right)c_i - \left(\nu\mu - 8\mu^2L^2 - \frac{2\gamma}{N}h^4\right)a_{i-d} + \mu dL\sum_{j=i-d}^{i-1} b_{j+1}
\end{aligned} \tag{6.131}
$$

We require that

$$
\begin{aligned}
\gamma - 8\mu^2\delta^2 =& \gamma\frac{N-1}{(1-\mu\nu/4)N} \\
\iff \quad [1-\mu\nu N/4]\gamma =& \, 8\mu^2\delta^2 N(1-\mu\nu/4) \\
\iff \qquad \gamma =& \frac{8\mu^2\delta^2 N(1-\mu\nu/4)}{1-\mu\nu N/4} = O(\mu^2)
\end{aligned} \tag{6.132}
$$

We can further require that $\mu \leq \frac{1}{2\nu N}$ so that γ has the upper bound:

$$
\gamma \leq 10\mu^2\delta^2 N \leq 5\mu\frac{\delta^2}{\nu} \tag{6.133}
$$

Let $s_{i+1} = a_{i+1} + (\gamma - 8\mu^2\delta^2)e_i$, we have

$$
s_{i+1} \leq (1-\mu\nu/3)s_i + \mu dL\sum_{j=i-d}^{i-1} b_{j+1} + \left(\frac{8\mu\delta^2}{\nu} + \frac{2\gamma}{N}\right)c_i \tag{6.134}
$$

where we discard the term about a_{i-d}, which requires:

$$
\begin{aligned}
&\nu\mu - 8\mu^2L^2 - \frac{2\gamma}{N}h^4 \geq 0 \\
\iff &\nu\mu \geq 8\mu^2L^2 + \frac{2\gamma}{N}h^4 \\
\overset{(6.133)}{\impliedby} &\nu\mu \geq 8\mu^2L^2 + 20\mu^2\delta^2h^4 \\
\iff &\mu \leq \frac{\nu}{8L^2 + 20\delta^2h^4}
\end{aligned} \tag{6.135}
$$

After substituting (6.133), we have

$$\begin{aligned} s_{i+1} \leq & (1-\mu\nu/4)s_i + \mu d L' \sum_{j=i-d}^{i-1} b_{j+1} + \left(\frac{8\mu\delta^2}{\nu} + \frac{10\mu\delta^2}{\nu N}\right) c_i \\ \leq & (1-\mu\nu/4)s_i + \mu d L' \sum_{j=i-d}^{i-1} \rho^{j+1} C_0 + \frac{18\mu\delta^2}{\nu}\rho^i C_3 \\ \leq & (1-\mu\nu/4)s_i + \rho^i \left(\mu d L' \sum_{j=0}^{d-1} \rho^{-j} C_0 + \frac{18\mu\delta^2}{\nu} C_3\right) \\ \leq & \rho^{i+1}\left(s_0 + \frac{\mu}{\rho - 1 + \mu\nu/4}\left(dL' \sum_{j=0}^{d-1} \rho^{-j} C_0 + \frac{18\delta^2}{\nu} C_3\right)\right) \\ \triangleq & \rho^{i+1} C_4 \end{aligned} \tag{6.136}$$

This also implies

$$a_i \leq \rho^i C_4 \tag{6.137}$$

Revisiting (6.122), we have

$$\begin{aligned} e_i \leq & \frac{N-1}{N} e_{i-1} + \rho^{i-d}\frac{h^4}{N} C_4 \\ \leq & \rho^i \left(e_0 + \frac{1}{\rho - \frac{N-1}{N}} \frac{h^4}{\rho^{d+1} N} C_4\right) \\ \triangleq & \rho^i C_5 \end{aligned} \tag{6.138}$$

Lastly, we substitute foregoing results into (6.119):

$$\begin{aligned} b_{i+1} \leq & 8\mu^2\delta^2\rho^i C_3 + 8\mu^2\delta^2\rho^i C_5 + 8\mu^2 L^2 \rho^{i-d} C_4 + 4\mu^2\eta^2\rho^i C_4 \\ = & \rho^i \mu^2 \left(8\delta^2 C_3 + 8\delta^2 C_5 + \frac{C_4}{\rho^d} + 4\eta^2 C_4\right) \end{aligned} \tag{6.139}$$

And we require the following to complete the mathematical induction:

$$\mu^2 \left(6\delta^2 C_3 + 6\delta^2 C_5 + \frac{C_4}{\rho^d} + 4\eta^2 C_4\right) < \rho C_0 \tag{6.140}$$

In the following, we will show that C_3, C_4 and C_5 can be upper bounded by constants that are independent of step-size μ. From (6.124), we have

$$\rho \geq 1 - \frac{1-\lambda^J}{2N}, \tag{6.141}$$

$$\rho \geq 1 - \frac{\mu\nu}{4}. \tag{6.142}$$

Moreover, since $\mu \leq 1/2\nu N$, from (6.142) we also have

$$\rho \geq 1 - \frac{\mu\nu}{4} \geq 1 - \frac{1}{8N}. \tag{6.143}$$

With (6.141) and (6.142), it holds that

$$\frac{1}{\rho - 1 + \frac{1}{2N}} \overset{(6.141)}{\leq} \frac{2N}{\lambda^J}, \tag{6.144}$$

$$\frac{1}{\rho - 1 + \frac{1-\lambda^J}{N}} \overset{(6.141)}{\leq} \frac{2N}{1-\lambda^J}, \tag{6.145}$$

$$\frac{\mu}{\rho - 1 + \mu\nu/4} \overset{(6.142)}{\leq} \frac{20}{\nu}, \tag{6.146}$$

$$\frac{1}{N\rho - N + 1} \overset{(6.141)}{\leq} \frac{2}{1+\lambda^J}. \tag{6.147}$$

Now we examine the upper bounds on C_3, C_4 and C_5. Note that

$$\begin{aligned} C_1 &= f_0 + \frac{1}{\rho - \frac{N-1/2}{N}} \frac{K^2 + 2K^2(N-1)^2}{N\rho^{d-1}} h^4 C_0 \\ &\overset{(6.144)}{\leq} f_0 + 2K^2 \frac{1 + 2(N-1)^2}{\lambda^J \rho^{d-1}} h^4 C_0 \\ &\overset{(6.143)}{\leq} f_0 + 2K^2 \frac{1 + 2(N-1)^2}{\lambda^J (1 - 1/(8N))^{d-1}} h^4 C_0 = O(1), \end{aligned} \tag{6.148}$$

and

$$\begin{aligned} C_2 &= d_0 + \frac{1}{\rho - 1 + \frac{1-\lambda^J}{N}} \frac{\lambda^{2J}}{N(1-\lambda^J)} C_1 \\ &\overset{(6.145)}{\leq} d_0 + \frac{2\lambda^{2J}}{(1-\lambda^J)^2} C_1 = O(1). \end{aligned} \tag{6.149}$$

With C_1 and C_2, we have

$$C_3 = \lambda^J C_2 + \frac{\lambda^{2J}}{1-\lambda^J} C_1 \triangleq C_3' = O(1) \tag{6.150}$$

Next we examine C_4:

$$\begin{aligned} C_4 &= s_0 + \frac{\mu}{\rho - 1 + \mu\nu/4} \left(dL' \sum_{j=0}^{d-1} \rho^{-j} C_0 + \frac{18\delta^2}{\nu} C_3 \right) \\ &\overset{(6.146)}{\leq} s_0 + \frac{20}{\nu} \left(\frac{d^2 L'}{(1-\frac{1}{8N})^d} C_0 + \frac{18\delta^2}{\nu} C_3 \right) \triangleq C_4' = O(1). \end{aligned} \tag{6.151}$$

For term C_5:

$$\begin{aligned} C_5 &= e_0 + \frac{h^4}{\rho^{d+1}(N\rho - N + 1)} C_4 \\ &\overset{(6.147)}{\leq} e_0 + \frac{2h^4}{\rho^{d+1}(1+\lambda^J)} C_4 \\ &\leq e_0 + \frac{2h^4}{(1-1/(8N))^{d+1}(1+\lambda^J)} C_4 \triangleq C_5' = O(1). \end{aligned} \tag{6.152}$$

Note that all constants C_3', C_4' and C_5' are independent of μ. Finally, note that

$$\begin{aligned} & 6\delta^2 C_3 + 6\delta^2 C_5 + \frac{C_4}{\rho^d} + 4\eta^2 C_4 \\ & \leq 6\delta^2 C_3' + 6\delta^2 C_5' + \frac{C_4'}{\rho^d} + 4\eta^2 C_4' \\ & \leq 6\delta^2 C_3' + 6\delta^2 C_5' + \frac{C_4'}{(1-1/(8N))^d} + 4\eta^2 C_4' \\ & \triangleq B, \end{aligned} \tag{6.153}$$

and B is independent of step-size μ. Also, we have have

$$\rho C_0 \geq \left(1 - \frac{1}{8N}\right) C_0. \tag{6.154}$$

To prove (6.140), it is enough to choose μ such that

$$\mu^2 B \leq \left(1 - \frac{1}{8N}\right) C_0 \Longleftrightarrow \mu \leq \sqrt{\left(1 - \frac{1}{8N}\right) \frac{C_0}{B}}. \tag{6.155}$$

Combining with $1/2\nu N$, we can set μ as

$$\mu \leq \min\left\{\sqrt{\left(1 - \frac{1}{8N}\right) \frac{C_0}{B}}, \frac{1}{2\nu N}\right\} \tag{6.156}$$

CHAPTER 7

Stochastic Subgradient Learning

This chapter examines the performance of stochastic sub-gradient learning strategies under weaker conditions than usually considered in the literature. The conditions are shown to be automatically satisfied by several important cases of interest including the construction of Linear-SVM, LASSO, and Total-Variation denoising formulations. In comparison, these problems do not satisfy the traditional assumptions automatically and, therefore, conclusions derived based on these earlier assumptions are not directly applicable to these problems. The analysis establishes that stochastic sub-gradient strategies can attain exponential convergence rates, as opposed to sub-linear rates, to the steady-state. A realizable exponential-weighting procedure is proposed to smooth the intermediate iterates by the sub-gradient procedure and to guarantee the established performance bounds in terms of convergence rate and excessive risk performance. Both single-agent and multi-agent scenarios are studied, where the latter case assumes that a collection of agents are interconnected by a topology and can only interact locally with their neighbors. The theoretical conclusions are illustrated by several examples and simulations, including comparisons with the FISTA procedure.

7.1 Introduction and Motivation

The minimization of *non-differentiable* convex cost functions is a critical step in the solution of many design problems [17,44,136], including the design of sparse-aware (LASSO) solutions [9,137], support-vector machine (SVM) learners [8,10,60,138,139], or total-variation based image denoising solutions [7,140]. Several powerful techniques have been proposed in the literature to deal with the non-differentiability aspect of the problem formulation, including

methods that employ of sub-gradient iterations [17, 44, 136], cutting-plane techniques [141], or proximal iterations [142, 143]. This work focuses on the class of sub-gradient methods for the reasons explained in the sequel. The sub-gradient technique is closely related to the traditional gradient-descent method where the actual gradient is replaced by a sub-gradient at the points of non-differentiability. It is one of the simplest methods in current practice but is known to suffer from slow convergence. For instance, it is shown in [136] that, for convex cost functions, the optimal convergence rate that can be delivered by sub-gradient methods in *deterministic* optimization problems cannot be faster than $O(1/\sqrt{i})$, where i is the iteration index.

Still, there are at least three strong reasons that motivate a closer examination of the limits of performance of sub-gradient learning algorithms. First, the explosive interest in large-scale and big data scenarios favors the use of simple and computer-efficient algorithmic structures, of which the sub-gradient technique is a formidable example. Second, it is becoming increasingly evident that more sophisticated optimization iterations do not necessarily ensure improved performance when dealing with complex models and data structures [17, 19, 144, 145]. This is because the assumed models, or the adopted cost functions, do not always reflect faithfully the underlying problem structure. In addition, the presence of noise in the data generally implies that a solution that may be perceived to be optimal is actually sub-optimal due to perturbations in the data and models. Third, it turns out that a clear distinction needs to be made between optimizing *deterministic* costs [17, 44, 136], where the cost function is known completely beforehand, and optimizing *stochastic* costs, where the cost function is actually unavailable due to its dependence on the unknown probability distribution of the data. Stochastic problem formulations are very common in applications arising in machine learning problems, adaptation, and estimation. We will show that sub-gradient algorithms have surprisingly favorable behavior in the stochastic setting.

Motivated by these remarks, we therefore examine in some detail the performance of *stochastic sub-gradient* algorithms for the minimization of non-differentiable convex costs. Our analysis will reveal some interesting properties when these algorithms are used in the context of continuous adaptation and learning (i.e., when actual sub-gradients cannot be

evaluated but need to be *approximated* continually in an *online* manner). The study is carried out for both cases of single stand-alone agents and multi-agent networks [11,58,102,146,147]. We start with single-agent learning and establish some revealing conclusions about how fast and how well the agent is able to learn. Extension of the results to the multi-agent case will require additional effort due to the coupling that exists among neighboring agents. Nevertheless, the same broad conclusions will continue to hold in this case with proper adjustments.

In order to examine the performance of *stochastic* sub-gradient implementations, it is necessary to introduce some assumptions on the gradient noise process (which is the difference between a true sub-gradient and its approximation). Here we diverge in a noticeable way from assumptions commonly used in the literature for two reasons (see Sec. 7.3 for further explanations). First, we introduce weaker assumptions than usually adopted in prior works and, secondly and more importantly, we show that our assumptions are automatically satisfied for important cases of interest (such as SVM, LASSO, Total Variation). In contrast, these applications do not satisfy the traditional assumptions used in the literature and, therefore, conclusions derived based on these earlier assumptions are not directly applicable to these problems. For example, it is common in the literature to assume that the cost function has a bounded gradient [17,22,58,147–149]; this condition is not even satisfied by quadratic costs whose gradient vectors are affine in their parameter. This condition is also in direct conflict with strongly-convex costs [149]. By weakening the assumptions, the analysis in this chapter becomes more challenging (as the material in the appendices reveal). At the same time, the conclusions become stronger and more revealing, and they apply to a broader class of algorithms and scenarios.

A second aspect of our study is that we focus on the use of *constant* step-sizes in order to enable continuous adaptation and learning. Since the step-size is assumed to remain constant, the effect of gradient noise is always present and does not die out, as would occur if we were using instead a diminishing step-size, say, of the form $\mu(i) = \tau/i$ for some $\tau > 0$ [58, 60, 148]. Such diminishing step-sizes annihilate the gradient noise term asymptotically albeit at the cost of turning off adaptation in the long run. When this happens, the learning

algorithm loses its ability to track drifts in the solution. In contrast, a constant step-size keeps adaptation alive and endows the learning algorithm with an inherent tracking mechanism: if the minimizer that we are seeking drifts with time due, for example, to changes in the statistical properties of the data, then the algorithm will be able to track the new location since it is continually adapting [150]. This useful tracking feature comes at the expense of a persistent gradient noise term that never dies out. The challenge in analyzing the performance of learning algorithms in the constant adaptation regime is to show that their feedback mechanism induces a stable behavior that reduces the size (variance) of the gradient noise to a small level and that ensures convergence of the iterates to within $O(\mu)$ of the desired optimal solution. Moreover, and importantly, it turns out that constant step-size adaptation is not only useful under non-stationary conditions when drifts in the data occur, but it is also useful even under stationary conditions when the minimizer does not vary with time. This is because, as we will see, the convergence towards the steady-state regime will now be guaranteed to occur at an exponential rate, $O(\alpha^i)$ for some $\alpha \in (0,1)$, which is much faster than the $O(1/i)$ rate that would be observed under a diminishing step-size implementation for strongly-convex costs.

A third aspect of our contribution is that it is known that sub-gradient methods are not *descent* methods. For this reason, it is customary to employ pocket variables (i.e., the best iterate) [44, 136, 151, 152] or arithmetic averages [60] to smooth out the output. However, as the analysis will reveal, the pocket method is not practical in the *stochastic* setting (its implementation requires knowledge of unavailable information), and the use of arithmetic averages [20] does not match the convergence rate derived later in Sec. 7.4.3. We shall propose an alternative weighted averaging scheme with an exponentially-decaying weight and show that this technique does not degrade convergence while providing the desired smoothing effect.

7.2 Problem Formulation: Single Agent Case

7.2.1 Problem Formulation

We consider the problem of minimizing a risk function, $J(w) : \mathbb{R}^M \rightarrow \mathbb{R}$, which is assumed to be expressed as the expected value of some loss function, $Q(w; \boldsymbol{x})$, namely,

$$w^\star \triangleq \arg\min_w J(w), \tag{7.1}$$

where we are denoting the minimizer by $w^\star$ and where

$$J(w) \triangleq \mathbb{E}\, Q(w; \boldsymbol{x}). \tag{7.2}$$

Here, the letter $\boldsymbol{x}$ represents the random data and the expectation operation is performed over the distribution of this data. Many problems in adaptation and learning involve risk functions of this form, including, for example, mean-square-error designs and support vector machine (SVM) solutions — see, e.g., [10,139,150]. For generality, we allow the risk function $J(w)$ to be *non-differentiable*. This situation is common in machine learning formulations, e.g., in SVM costs and in regularized sparsity-inducing formulations; examples to this effect are provided in the sequel.

In this chapter, we examine in some detail the performance of stochastic *sub-gradient* algorithms for the minimization of (7.1) and reveal some interesting properties when these algorithms are used in the context of continuous adaptation and learning (i.e., when actual sub-gradients cannot be evaluated but need to be *approximated* continually in an online manner). This situation arises when the probability distribution of the data is not known beforehand, as is common in practice. In most applications, we only have access to data realizations but not to their actual distribution. Our study is carried out for both cases of single stand-alone agents and multi-agent networks. We start with single-agent learning and establish some revealing conclusions about how fast and how well the agent is able to learn. Extension of the results to the multi-agent case will require additional effort due to the

coupling that exists among neighboring agents, and is pursued in a future section. The same broad conclusions will continue to hold in this case as well with some proper adjustments.

7.2.2 Stochastic Sub-Gradient Algorithm

To describe the sub-gradient algorithm in the single-agent case, we first recall that the sub-gradient of a function $J(w)$ at any arbitrary point w_0 is defined as any vector $g \in \mathbb{R}^M$ that satisfies:

$$J(w) \;\geq\; J(w_0) + g^{\mathsf{T}}(w - w_0), \;\; \forall w \tag{7.3}$$

We shall often write $g(w_0)$, instead of simply g, in order to emphasize that it is a sub-gradient vector at location w_0. We note that sub-gradients are generally non-unique. Accordingly, a related concept is that of the *sub-differential* of $J(w)$ at w_0, denoted by $\partial J(w_0)$. The sub-differential is defined as the set of all possible sub-gradient vectors at w_0:

$$\partial J(w_0) \;\stackrel{\Delta}{=}\; \big\{ g \mid J(w) \geq J(w_0) + g^{\mathsf{T}}(w - w_0), \;\; \forall w \big\}. \tag{7.4}$$

In general, the sub-differential $\partial J(w_0)$ is a set and it will collapse to a single point if, and only if, the cost function is differentiable at w_0 [136]; in that case, the sub-gradient vector will coincide with the actual gradient vector at location w_0.

Now, referring back to problem (7.1), the traditional sub-gradient method to minimizing the risk function $J(w)$ takes the form:

$$w_i = w_{i-1} - \mu\, g(w_{i-1}), \quad i \geq 0 \tag{7.5}$$

where $g(w_{i-1})$ refers to a sub-gradient vector for $J(w)$ at location w_{i-1}, and $\mu > 0$ is a small step-size parameter. However, in the context of adaptation and learning, we do not know the exact form of $J(w)$ because the distribution of the data is not known to enable computation of $\mathbb{E}\, Q(w; \boldsymbol{x})$. As such, true sub-gradient vectors for $J(w)$ cannot be determined and they will need to be replaced by stochastic approximations evaluated from streaming data; examples to this effect are provided in the sequel in the context of support-vector

machines and LASSO sparse designs. Accordingly, we replace the deterministic iteration (7.5) by the following stochastic iteration [44, 136, 151, 152]:

$$\boldsymbol{w}_i = \boldsymbol{w}_{i-1} - \mu\,\widehat{g}(\boldsymbol{w}_{i-1}), \tag{7.6}$$

where the successive iterates, $\{\boldsymbol{w}_i\}$, are now random variables (denoted in boldface) and $\widehat{g}(\cdot)$ represents an approximate sub-gradient vector at location $\boldsymbol{w}_{i-1}$ estimated from data available at time i. The difference between an actual sub-gradient vector and its approximation is referred to as *gradient noise* and is denoted by

$$\boldsymbol{s}_i(\boldsymbol{w}_{i-1}) \;\stackrel{\Delta}{=}\; \widehat{g}(\boldsymbol{w}_{i-1}) - g(\boldsymbol{w}_{i-1}). \tag{7.7}$$

7.2.3 Examples: SVM and LASSO

To illustrate the construction, we list two examples dealing with support vector machines (SVM) [138] and the LASSO problem [137]; the latter is also known as the sparse LMS problem or basis pursuit [9, 153, 154]. We will be using these two problems throughout the manuscript to illustrate our findings.

Example 1 (SVM problem). The two-class SVM formulation deals with the problem of determining a separating hyperplane, $w \in \mathbb{R}^M$, in order to classify feature vectors, denoted by $\boldsymbol{h} \in \mathbb{R}^M$, into one of two classes: $\boldsymbol{\gamma} = +1$ or $\boldsymbol{\gamma} = -1$. The regularized SVM risk function is of the form:

$$J^{svm}(w) \quad \stackrel{\Delta}{=} \quad \frac{\rho}{2}\|w\|^2 + \mathbb{E}\left(\max\left\{0, 1 - \boldsymbol{\gamma}\boldsymbol{h}^{\mathsf{T}}w\right\}\right), \tag{7.8}$$

where $\rho > 0$ is a regularization parameter. We are generally given a collection of independent training data, $\{\boldsymbol{\gamma}(i), \boldsymbol{h}_i\}$, consisting of feature vectors and their class designations and assumed to arise from joint wide-sense stationary processes. Using this data, the loss function

at time i is given by

$$Q^{svm}(w;\{\boldsymbol{\gamma}(i),\boldsymbol{h}_i\}) = \frac{\rho}{2}\|w\|^2 + \max\left\{0, 1-\boldsymbol{\gamma}(i)\boldsymbol{h}_i^{\mathsf{T}}w\right\}, \tag{7.9}$$

where the second term on the right-hand side, which is also known as the hinge function, is non-differentiable at all points w satisfying $1-\boldsymbol{\gamma}(i)\boldsymbol{h}_i^{\mathsf{T}}w = 0$. One choice to approximate the sub-gradient vector of $J^{svm}(w)$ is to employ the following instantaneous approximation (which follows from "differentiating" the loss function (7.9)):

$$\widehat{g}^{svm}(\boldsymbol{w}_{i-1}) = \rho\boldsymbol{w}_{i-1} + \boldsymbol{\gamma}(i)\boldsymbol{h}_i\,\mathbb{I}[\boldsymbol{\gamma}(i)\boldsymbol{h}_i^{\mathsf{T}}\boldsymbol{w}_{i-1} \le 1]. \tag{7.10}$$

In this expression, the indicator function $\mathbb{I}[a]$ is defined as follows:

$$\mathbb{I}[a] \;=\; \begin{cases} 1, & \text{if statement } a \text{ is true} \\ 0, & \text{otherwise} \end{cases} \tag{7.11}$$

It then follows that the gradient noise process in the SVM formulation is given by

$$\boldsymbol{s}_i^{svm}(\boldsymbol{w}_{i-1}) = \boldsymbol{\gamma}(i)\boldsymbol{h}_i\,\mathbb{I}[\boldsymbol{\gamma}(i)\boldsymbol{h}_i^{\mathsf{T}}\boldsymbol{w}_{i-1} \le 1] - \mathbb{E}\,\boldsymbol{\gamma}\boldsymbol{h}\,\mathbb{I}[\boldsymbol{\gamma}\boldsymbol{h}^{\mathsf{T}}\boldsymbol{w}_{i-1} \le 1]. \tag{7.12}$$

□

Example 2 (LASSO problem). The least-mean-squares LASSO formulation deals with the problem of estimating a sparse weight vector by minimizing a risk function of the form [155, 156]:[1]

$$J^{lasso}(w) \;\triangleq\; \frac{1}{2}\mathbb{E}\,\|\boldsymbol{\gamma} - \boldsymbol{h}^{\mathsf{T}}w\|^2 + \delta\|w\|_1, \tag{7.13}$$

where $\delta > 0$ is a regularization parameter and $\|w\|_1$ denotes the ℓ_1-norm of w. In this problem formulation, the variable $\boldsymbol{\gamma}$ now plays the role of a desired signal, while $\boldsymbol{h}$ plays the role of a regression vector. It is assumed that the data are zero-mean wide-sense stationary

[1]Traditionally, LASSO refers to minimize a deterministic cost function, like $\|y-Ax\|^2+\lambda\|x\|_1$. However, since here we are more interested in stochastic version, we only consider the case like (7.13).

with second-order moments denoted by

$$\boldsymbol{r}_{h\gamma} \triangleq \mathbb{E}\,\boldsymbol{h}\boldsymbol{\gamma}, \quad R_h \triangleq \mathbb{E}\,\boldsymbol{h}\boldsymbol{h}^{\mathsf{T}}. \tag{7.14}$$

It is generally assumed that $\{\boldsymbol{\gamma}, \boldsymbol{h}\}$ satisfy a linear regression model of the form:

$$\boldsymbol{\gamma} = \boldsymbol{h}^{\mathsf{T}} w^o + \boldsymbol{n}, \tag{7.15}$$

where $w^o \in \mathbb{R}^M$ is the desired unknown sparse vector, and $\boldsymbol{n}$ refers to an additive zero-mean noise component with finite variance σ_n^2 and independent of $\boldsymbol{h}$. If we multiply both sides of (7.15) by $\boldsymbol{h}$ from the left and compute expectations, we find that w^o satisfies the normal equations:

$$r_{h\gamma} = R_h w^o. \tag{7.16}$$

We are again given a collection of independent training data, $\{\boldsymbol{\gamma}(i), \boldsymbol{h}_i\}$, consisting of regression vectors and their noisy measured signals. Using this data, the loss function at time i is given by

$$Q^{lasso}(w; \{\boldsymbol{\gamma}(i), \boldsymbol{h}_i\}) = \frac{1}{2}(\boldsymbol{\gamma}(i) - \boldsymbol{h}_i^{\mathsf{T}} w)^2 + \delta\|w\|_1, \tag{7.17}$$

where the second term on the right-hand side is again non-differentiable. One choice for the approximate sub-gradient vector of $J^{lasso}(w)$ is to employ the following instantaneous approximation (which follows from "differentiating" the loss function (7.17)):

$$\begin{aligned} \widehat{g}^{\,lasso}(\boldsymbol{w}_{i-1}) &= -\boldsymbol{h}_i(\boldsymbol{\gamma}(i) - \boldsymbol{h}_i^{\mathsf{T}}\boldsymbol{w}_{i-1}) + \delta \cdot \mathrm{s}gn(\boldsymbol{w}_{i-1}) \\ &= -\boldsymbol{h}_i\boldsymbol{h}_i^{\mathsf{T}}(w^\circ - \boldsymbol{w}_{i-1}) + \delta \cdot \mathrm{s}gn(\boldsymbol{w}_{i-1}) - \boldsymbol{h}_i\boldsymbol{n}(i), \end{aligned} \tag{7.18}$$

where the notation $\mathrm{s}gn(a)$, for a scalar a, refers to the sign function:

$$\mathrm{sgn}[a] = \begin{cases} +1, & a \geq 0 \\ -1, & \text{otherwise} \end{cases}. \tag{7.19}$$

When applied to a vector a, as is the case in (7.18), the sgn function is a vector consisting

of the signs of the individual entries of a. It then follows that the gradient noise process in the LASSO formulation is given by

$$\boldsymbol{s}_i^{lasso}(\boldsymbol{w}_{i-1}) = (R_h - \boldsymbol{h}_i\boldsymbol{h}_i^{\mathsf{T}})(w^\circ - \boldsymbol{w}_{i-1}) - \boldsymbol{h}_i\boldsymbol{n}(i). \tag{7.20}$$

□

7.3 Modeling Conditions

In order to examine the performance of the stochastic sub-gradient implementation (7.6) for single-agent adaptation and learning, and later for multi-agent networks, it is necessary to introduce some assumptions on the gradient noise process. We diverge here from assumptions that are commonly used in the literature for two reasons. First, we introduce weaker assumptions than usually adopted in prior works and, secondly and more importantly, we show that our assumptions are automatically satisfied by important cases of interest (such as SVM and LASSO). In contrast, these applications do not satisfy the traditional assumptions used in the literature and, therefore, conclusions derived based on these earlier assumptions are not directly applicable to SVM and LASSO problems. We clarify these remarks in the sequel.

Recall from (7.1) that $w^\star$ denotes the global minimizer that we are seeking. The first set of conditions on the gradient noise process below essentially require that the construction of the approximate sub-gradient vector should not introduce bias and that its error variance should decrease as the quality of the iterate approaches the optimal solution, $w^\star$. Both of these conditions are sensible and, moreover, they will be shown to be satisfied by, for example, SVM and LASSO constructions. More formally, we require the gradient noise process to satisfy the following two conditions.

Assumption 7.1 (Conditions on gradient noise) *The first and second-order conditional*

moments of the gradient noise process satisfy the following conditions:

$$\begin{aligned}\mathbb{E}\,[\,s_i(\boldsymbol{w}_{i-1})\,|\,\boldsymbol{\mathcal{F}}_{i-1}\,] &= 0, &(7.21)\\ \mathbb{E}\,[\,\|s_i(\boldsymbol{w}_{i-1})\|^2\,|\,\boldsymbol{\mathcal{F}}_{i-1}\,] &\leq \beta^2\|w^\star-\boldsymbol{w}_{i-1}\|^2+\sigma^2, &(7.22)\end{aligned}$$

for some constants $\beta^2 \geq 0$ and $\sigma^2 \geq 0$, and where the notation $\boldsymbol{\mathcal{F}}_{i-1}$ denotes the filtration (collection) corresponding to all past iterates:

$$\boldsymbol{\mathcal{F}}_{i-1} \;=\; \textit{filtration by } \{\boldsymbol{w}_j,\; j \leq i-1\}. \tag{7.23}$$

■

Assumption 7.2 (Strongly-convex risk function) *The risk function is assumed to be $\eta-$strongly-convex (or, simply, strongly-convex), i.e., there exists an $\eta > 0$ such that*

$$J(\theta w_1+(1-\theta)w_2) \;\leq\; \theta J(w_1)+(1-\theta)J(w_2)-\frac{\eta}{2}\theta(1-\theta)\|w_1-w_2\|^2, \tag{7.24}$$

for any $\theta \in [0,1]$, w_1, and w_2. The above condition is equivalent to requiring [17]:

$$J(w_1) \geq J(w_2)+g(w_2)^{\mathsf{T}}(w_1-w_2)+\frac{\eta}{2}\|w_1-w_2\|^2. \tag{7.25}$$

Under this condition, the minimizer $w^\star$ exists and is unique. ■

Assumption 7.2 is relatively rare in works on non-differentiable function optimization because it is customary for these works to focus on studying *piece-wise* linear risks; these are important examples of non-smooth functions but they do not satisfy the strong-convexity condition. In our case, strong-convexity is not a restriction because in the context of adaptation and learning, it is common for the risk functions to include a regularization term, which generally helps ensure strong-convexity.

Assumption 7.3 (Sub-gradient is Affine-Lipschitz) *It is assumed that the sub-gradient of the risk function, $J(w)$, is affine Lipschitz, which means that there exist constants $c \geq 0$*

and $d \geq 0$ such that

$$\|g(w_1) - g(w_2)\| \;\leq\; c\|w_1 - w_2\| + d, \quad \forall w_1, w_2, \tag{7.26}$$

and for any choice $g(\cdot) \in \partial J(w)$. ■

More critically, though, it is customary in the literature to use in place of Assumption 7.3 a more restrictive condition that requires the sub-gradient to be bounded [22,44,58,147,149], i.e.,

$$\|g(w)\| \leq d_1, \quad \forall w,\ g \in \partial J(w). \tag{7.27}$$

which is also equivalent to assuming the risk function is Lipschitz:

$$\|J(w_1) - J(w_2)\| \leq d_1\|w_1 - w_2\|, \quad \forall w_1, w_2 \tag{7.28}$$

Such a requirement does not even hold for quadratic risk functions, $J(w)$, whose gradient vectors are affine in w and, therefore, grow unbounded! Even more, it can be easily seen that requirement (7.27) is always conflicted with the strong-convexity assumption. For example, if we set $w_1 = w$ and $w_2 = w^\star$ in (7.25), we would obtain:

$$J(w) \geq J(w^\star) + \frac{\eta}{2}\|w - w^\star\|^2. \tag{7.29}$$

Likewise, if we instead set $w_1 = w^\star$ and $w_2 = w$ in (7.25), we would obtain:

$$J(w^\star) \geq J(w) + g(w)^{\mathsf{T}}(w^\star - w) + \frac{\eta}{2}\|w - w^\star\|^2. \tag{7.30}$$

Adding relations (7.29)–(7.30) we arrive at the so-called strong monotonicity property:

$$g(w)^{\mathsf{T}}(w - w^\star) \;\;\geq\;\; \eta\|w - w^\star\|^2, \tag{7.31}$$

which implies, in view of the Cauchy-Schwarz inequality, that

$$\|g(w)\| \geq \eta \|w - w^\star\|. \tag{7.32}$$

In other words, the strong-convexity condition (7.25) implies that the sub-gradient satisfies (7.32); and this condition is in clear conflict with the bounded requirement in (7.27).

One way to circumvent this problem is to restrict the domain of $J(w)$ to some bounded convex set, say, $w \in \mathcal{W}$, in order to bound its sub-gradient vectors, and then employ a projection-based sub-gradient method (i.e., one in which each iteration is followed by projecting $\boldsymbol{w}_i$ onto $\mathcal{W}$). However, this approach has at least three difficulties. First, the unconstrained problem is transformed into a more demanding constrained problem involving an extra projection step. Second, the projection step may not be straightforward to carry out unless the set $\mathcal{W}$ is simple enough. Third, the bound that results on the sub-gradient vectors by limiting w to $\mathcal{W}$ can be very loose, which will be dependent on the diameter of convex set $\mathcal{W}$.

For these reasons, we do not rely on the restrictive condition (7.27) and introduce instead the more relaxed affine-Lipschitz condition (7.26). This condition is weaker than (7.27). Indeed, it can be verified that (7.27) implies (7.26) but not the other way around. To see this, assume (7.27) holds. Then, using the triangle inequality of norms we have

$$\begin{aligned} \|g(w_1) - g(w_2)\| &\leq \|g(w_1)\| + \|g(w_2)\| \\ &\leq d_1 \ + \ d_1 \\ &= 2d_1, \end{aligned} \tag{7.33}$$

which is a special case of (7.26) with $c = 0$ and $d = 2d_1$. We now verify that important problems of interest satisfy Assumption 7.3 but *not* the traditional condition (7.27).

Example 3 (SVM problem). We revisit the SVM formulation from Example 1. The risk function (7.8) is strongly convex due to the presence of the quadratic regularization term,

$\frac{\rho}{2}\|w\|^2$, and since the hinge function $\mathbb{E}\max\{0, 1-\boldsymbol{\gamma}\boldsymbol{h}^{\mathsf{T}}w\}$ is convex. The zero-mean property of the gradient noise process is obvious in this case. With respect to the variance condition, we note that

$$\begin{aligned}\mathbb{E}\,[\|s_i^{svm}(\boldsymbol{w}_{i-1})\|^2|\boldsymbol{\mathcal{F}}_{i-1}] &= \mathbb{E}\,\boldsymbol{h}_i^{\mathsf{T}}\boldsymbol{h}_i\,\mathbb{I}[\boldsymbol{\gamma}(i)\boldsymbol{h}_i^{\mathsf{T}}\boldsymbol{w}_{i-1}\leq 1] - \|\mathbb{E}\,\boldsymbol{\gamma}\boldsymbol{h}\,\mathbb{I}[\boldsymbol{\gamma}\boldsymbol{h}^{\mathsf{T}}\boldsymbol{w}_{i-1}\leq 1]\|^2\\ &\leq \mathbb{E}\,\boldsymbol{h}_i^{\mathsf{T}}\boldsymbol{h}_i\,\mathbb{I}[\boldsymbol{\gamma}(i)\boldsymbol{h}_i^{\mathsf{T}}\boldsymbol{w}_{i-1}\leq 1]\\ &\leq \mathbb{E}\,\boldsymbol{h}_i^{\mathsf{T}}\boldsymbol{h}_i\\ &= \mathrm{Tr}(R_h),\end{aligned} \tag{7.34}$$

so that Assumption 7.1 is satisfied with $\beta^2 = 0$ and $\sigma^2 = \mathrm{Tr}(R_h)$. Let us now verify Assumption 7.3. For that purpose, we first note that:

$$\|g^{svm}(w_1) - g^{svm}(w_2)\| \leq \rho\|w_1 - w_2\| + \|\mathbb{E}\,\boldsymbol{\gamma}\boldsymbol{h}\,\mathbb{I}[\boldsymbol{\gamma}\boldsymbol{h}^{\mathsf{T}}w_1\leq 1]\| + \big\|\mathbb{E}\,\boldsymbol{\gamma}\boldsymbol{h}\,\mathbb{I}[\boldsymbol{\gamma}\boldsymbol{h}^{\mathsf{T}}w_2\leq 1]\big\|. \tag{7.35}$$

Additionally, we have

$$\begin{aligned}\big\|\mathbb{E}\,\boldsymbol{\gamma}\boldsymbol{h}\,\mathbb{I}[\boldsymbol{\gamma}\boldsymbol{h}^{\mathsf{T}}w<1]\big\|^2 &\overset{(a)}{\leq} \mathbb{E}\,\big\|\boldsymbol{\gamma}\boldsymbol{h}\,\mathbb{I}[\boldsymbol{\gamma}\boldsymbol{h}^{\mathsf{T}}w\leq 1]\big\|^2\\ &= \mathbb{E}\,\boldsymbol{h}^{\mathsf{T}}\boldsymbol{h}\,\mathbb{I}[\boldsymbol{\gamma}\boldsymbol{h}^{\mathsf{T}}w\leq 1]\\ &\leq \mathrm{Tr}(R_h),\end{aligned} \tag{7.36}$$

where step (a) uses Jensen's inequality [46]. Substituting into (7.35) gives

$$\|g^{svm}(w_1) - g^{svm}(w_2)\| \leq \rho\|w_1 - w_2\| + 2[\mathrm{Tr}(R_h)]^{1/2}, \tag{7.37}$$

which is of the same form as (7.26) with parameters $c = \rho$ and $d = 2[\mathrm{Tr}(R_h)]^{1/2}$. □

Example 4 (LASSO problem). We revisit the LASSO formulation from Example 2. Under the condition that $R_h > 0$, the risk function (7.13) is again strongly-convex because the quadratic term, $\frac{1}{2}\mathbb{E}\,\|\boldsymbol{\gamma} - \boldsymbol{h}^{\mathsf{T}}w\|^2$, is strongly convex and the regularization term, $\delta\|w\|_1$, is convex. With regards to the gradient noise process, it was already shown in Eq. (3.22) in [11]

that a gradient noise process of the form (7.20) is zero-mean and its conditional variance satisfies:

$$\begin{aligned}\mathbb{E}\,[\|s_i^{lasso}(\boldsymbol{w}_{i-1})\|^2|\boldsymbol{\mathcal{F}}_{i-1}] &\leq a\|w^\circ-\boldsymbol{w}_{i-1}\|^2+\sigma_n^2\mathrm{Tr}(R_h)\\ &\leq 2a\|w^\star-\boldsymbol{w}_{i-1}\|^2+\sigma_n^2\mathrm{Tr}(R_h)+2a\|w^\circ-w^\star\|^2,\end{aligned}\tag{7.38}$$

where $a=2\mathbb{E}\,\|R_h-\boldsymbol{h}_i\boldsymbol{h}_i^{\mathsf{T}}\|^2$. It follows that Assumption 7.1 is satisfied with $\beta^2=2a$ and $\sigma^2=\sigma_n^2\mathrm{Tr}(R_h)+2a\|w^\circ-w^\star\|^2$. Let us now verify Assumption 7.3. For that purpose, we first note that:

$$\begin{aligned}\|g^{lasso}(w_1)-g^{lasso}(w_2)\| &= \|R_hw_1-R_hw_2+\delta(sgn(w_1)-sgn(w_2))\|\\ &\leq \|R_h\|\|w_1-w_2\|+2\delta\|\mathbb{1}\|\\ &= \|R_h\|\|w_1-w_2\|+2\delta M^{1/2},\end{aligned}\tag{7.39}$$

where $\mathbb{1}$ is the column vector with all its entries equal to one. We again arrive at a relation of the same form as (7.26) with parameters $c=\|R_h\|$ and $d=2\delta M^{1/2}$. □

7.4 Performance Analysis: Single Agent Case

We now carry out a detailed mean-square-error analysis of the stability and performance of the stochastic sub-gradient recursion (7.6) in the presence of gradient noise and for *constant* step-size adaptation.

7.4.1 Continuous Adaptation

Since the step-size is assumed to remain constant, the effect of gradient noise is continually present and does not die out, as would occur if we were using instead a diminishing step-size, say, of the form $\mu(i)=\tau/i$. Such diminishing step-sizes annihilate the gradient noise term asymptotically albeit at the expense of turning off adaptation in the long run. In that case, the learning algorithm will lose its tracking ability. In contrast, a constant step-size keeps

adaptation alive and endows the learning algorithm with a tracking mechanism and, as the analysis will show, enables convergence towards the steady-state regime at an exponential rate, $O(\alpha^i)$, for some $\alpha \in (0,1)$.

7.4.2 A Useful Bound

In preparation for the analysis, we first conclude from (7.26) that the following useful condition also holds, involving squared-norms as opposed to the actual norms:

$$\|g(w_1) - g(w_2)\|^2 \leq e^2\|w_1 - w_2\|^2 + f^2 \quad \forall w_1, w_2, \;\; g \in \partial J, \tag{7.40}$$

for some nonnegative constants, e^2, f^2. Indeed, if we square both sides of (7.26) we get

$$\begin{aligned} \|g(w_1) - g(w_2)\|^2 \leq \;& c^2\|w_1 - w_2\|^2 + 2cd\|w_1 - w_2\| + d^2 \\ \overset{(a)}{\leq} \;& \left(c^2 + \frac{2cd}{R}\right)\|w_1 - w_2\|^2 + d^2 + 2cdR, \end{aligned} \tag{7.41}$$

where the constant R is any positive number that we are free to choose, and step (a) is because

$$\|w_1 - w_2\| \leq \begin{cases} R, & \text{if } \|w_1 - w_2\| < R \\ \|w_1 - w_2\|^2/R, & \text{if } \|w_1 - w_2\| \geq R \end{cases}. \tag{7.42}$$

It follows that the constants $\{e^2, f^2\}$ in (7.40) can be taken as

$$e^2 \;\triangleq\; c^2 + \frac{2cd}{R} \geq 0 \tag{7.43}$$

$$f^2 \;\triangleq\; d^2 + 2cdR \geq 0. \tag{7.44}$$

There is a second easier derivation for a bound of the form (7.40) without the need to introduce the parameter R but it generally leads to a looser bound. For example, observe that by squaring and appealing to Jensen's inequality we get:

$$\|g(w_1) - g(w_2)\|^2 \;\leq\; \Big(c\|w_1 - w_2\| + d\Big)^2 \tag{7.45}$$

$$\leq \quad 2c^2\|w_1 - w_2\|^2 + 2d^2. \tag{7.46}$$

In this case, we would select $e^2 = 2c^2$ and $f^2 = 2d^2$. We continue with (7.41).

7.4.3 Stability and Convergence

We are now ready to establish the following important conclusion regarding the stability and performance of the stochastic sub-gradient algorithm (7.6); the conclusion indicates that the algorithm is stable and converges exponentially fast for sufficiently small step-sizes. But first, we explain our notation and the definition of a "best" iterate, denoted by $\boldsymbol{w}_i^{best}$ [136]. This variable is useful in the context of sub-gradient implementations because it is known that (negative) sub-gradient directions do not necessarily correspond to real descent directions (as is the case with actual gradient vectors for differentiable functions).

At every iteration i, the risk value that corresponds to the iterate $\boldsymbol{w}_i$ is $J(\boldsymbol{w}_i)$. This value is obviously a random variable due to the randomness in the data used to run the algorithm. We denote the mean risk value by $\mathbb{E}\, J(\boldsymbol{w}_i)$. The next theorem examines how fast and how close this mean value approaches the optimal value, $J(w^\star)$. To do so, the statement in the theorem relies on the *best pocket* iterate, denoted by $\boldsymbol{w}_i^{\rm best}$, and which is defined as follows. At any iteration i, the value that is saved in this pocket variable is the iterate, $\boldsymbol{w}_j$, that has generated the smallest mean risk value up to that point in time, i.e.,

$$\boldsymbol{w}_i^{\rm best} \quad \stackrel{\Delta}{=} \quad \arg\min_{0\leq j\leq i} \ \mathbb{E}\ J(\boldsymbol{w}_j). \tag{7.47}$$

The statement below then proves that $\mathbb{E}\, J(\boldsymbol{w}_i^{best})$ approaches a small neighborhood of size $O(\mu)$ around $J(w^\star)$ exponentially fast:

$$\lim_{i\to\infty} \ \mathbb{E}\ J(\boldsymbol{w}_i^{\rm best}) \ \leq \ J(w^\star) \ + \ O(\mu), \tag{7.48}$$

where the big-O notation $O(\mu)$ means in the order of μ.

Theorem 7.1 (Single agent performance) *Consider using the stochastic sub-gradient*

algorithm (7.6) to seek the unique minimizer, $w^\star$, of the optimization problem (7.1), where the risk function, $J(w)$, is assumed to satisfy Assumptions 7.1–7.3. If the step-size parameter satisfies (i.e., if it is small enough):

$$\mu < \frac{\eta}{e^2 + \beta^2}, \tag{7.49}$$

then it holds that

$$\mathbb{E}\, J(\boldsymbol{w}_i^{best}) - J(w^\star) \;\leq\; \xi \cdot \alpha^i \mathbb{E}\left\|\boldsymbol{w}_0 - w^\star\right\|^2 + \mu(f^2 + \sigma^2)/2, \quad \forall\, i \tag{7.50}$$

That is the convergence of $\mathbb{E}\, J(\boldsymbol{w}_i^{best})$ towards the $O(\mu)-$neighborhood around $J(w^\star)$ occurs at linear rate, $O(\alpha^i)$, dictated by the parameter

$$\alpha \;\stackrel{\Delta}{=}\; 1 - \mu\eta + \mu^2(e^2 + \beta^2) = 1 - O(\mu) \tag{7.51}$$

Condition (7.49) ensures $\alpha \in (0, 1)$. In the limit:

$$\lim_{i\to\infty} \mathbb{E}\, J(\boldsymbol{w}_i^{best}) - J(w^\star) \;\leq\; \mu(f^2 + \sigma^2)/2. \tag{7.52}$$

That is, for large i, $\mathbb{E}\, J(\boldsymbol{w}_i^{best})$ is approximately $\mu(f^2 + \sigma^2)/2$-suboptimal.

Proof: We introduce the error vector, $\widetilde{\boldsymbol{w}}_i = w^\star - \boldsymbol{w}_i$, and use it to deduce from (7.6)–(7.7) the following error recursion:

$$\widetilde{\boldsymbol{w}}_i = \widetilde{\boldsymbol{w}}_{i-1} + \mu g(\boldsymbol{w}_{i-1}) + \mu s_i(\boldsymbol{w}_{i-1}). \tag{7.53}$$

Squaring both sides and computing the conditional expectation we obtain:

$$\begin{aligned}
\mathbb{E}\,[\|\widetilde{\boldsymbol{w}}_i\|^2 \,|\, \boldsymbol{\mathcal{F}}_{i-1}\,] &= \mathbb{E}\,[\,\|\widetilde{\boldsymbol{w}}_{i-1} + \mu g(\boldsymbol{w}_{i-1}) + \mu \boldsymbol{s}_i(\boldsymbol{w}_{i-1})\|^2 \,|\, \boldsymbol{\mathcal{F}}_{i-1}\,] \\
&\stackrel{(a)}{=} \|\widetilde{\boldsymbol{w}}_{i-1} + \mu g(\boldsymbol{w}_{i-1})\|^2 \;+\; \mu^2 \mathbb{E}\,[\|\boldsymbol{s}_i(\boldsymbol{w}_{i-1})\|^2 \,|\, \boldsymbol{\mathcal{F}}_{i-1}\,] \\
&= \|\widetilde{\boldsymbol{w}}_{i-1}\|^2 + 2\mu g(\boldsymbol{w}_{i-1})^{\mathsf{T}} \widetilde{\boldsymbol{w}}_{i-1} + \mu^2 \|g(\boldsymbol{w}_{i-1})\|^2 + \mu^2 \mathbb{E}\,[\|\boldsymbol{s}_i(\boldsymbol{w}_{i-1})\|^2 \,|\, \boldsymbol{\mathcal{F}}_{i-1}\,].
\end{aligned} \tag{7.54}$$

In step (a), we eliminated the cross term because, conditioned on $\boldsymbol{\mathcal{F}}_{i-1}$, the gradient noise process has zero-mean. Now, from the strong convexity condition (7.25), it holds that

$$g(\boldsymbol{w}_{i-1})^{\mathsf{T}}\widetilde{\boldsymbol{w}}_{i-1} \leq J(w^\star) - J(\boldsymbol{w}_{i-1}) - \frac{\eta}{2}\|\widetilde{\boldsymbol{w}}_{i-1}\|^2. \tag{7.55}$$

Substituting into (7.54) gives

$$\begin{aligned}\mathbb{E}\,[\|\widetilde{\boldsymbol{w}}_i\|^2 \,|\, \boldsymbol{\mathcal{F}}_{i-1}\,] \leq \|\widetilde{\boldsymbol{w}}_{i-1}\|^2 + 2\mu\left(J(w^\star) - J(\boldsymbol{w}_{i-1}) - \frac{\eta}{2}\|\widetilde{\boldsymbol{w}}_{i-1}\|^2\right) + \\ \mu^2\|g(\boldsymbol{w}_{i-1})\|^2 + \mu^2\mathbb{E}\,[\|s_i(\boldsymbol{w}_{i-1})\|^2 \,|\, \boldsymbol{\mathcal{F}}_{i-1}\,].\end{aligned} \tag{7.56}$$

Referring to (7.40), if we set $w_1 = \boldsymbol{w}_{i-1}$, $w_2 = w^\star$, and use the fact that $g(w^\star) = 0$, we obtain:

$$\|g(\boldsymbol{w}_{i-1})\|^2 \leq e^2\|\widetilde{\boldsymbol{w}}_{i-1}\|^2 + f^2. \tag{7.57}$$

Substituting into (7.56), we get

$$\begin{aligned}&\mathbb{E}\,[\|\widetilde{\boldsymbol{w}}_i\|^2 \,|\, \boldsymbol{\mathcal{F}}_{i-1}\,] \\ &\leq (1-\mu\eta+\mu^2e^2)\|\widetilde{\boldsymbol{w}}_{i-1}\|^2 + 2\mu J(w^\star) - 2\mu J(\boldsymbol{w}_{i-1}) + \mu^2 f^2 + \mu^2\mathbb{E}\,[\|s_i(\boldsymbol{w}_{i-1})\|^2 \,|\, \boldsymbol{\mathcal{F}}_{i-1}\,] \\ &\overset{(7.22)}{\leq} (1-\mu\eta+\mu^2(e^2+\beta^2))\|\widetilde{\boldsymbol{w}}_{i-1}\|^2 + 2\mu J(w^\star) - 2\mu J(\boldsymbol{w}_{i-1}) + \mu^2 f^2 + \mu^2\sigma^2.\end{aligned} \tag{7.58}$$

Taking expectation again we eliminate the conditioning on $\boldsymbol{\mathcal{F}}_{i-1}$ and arrive at:

$$2\mu(\mathbb{E}\,J(\boldsymbol{w}_{i-1}) - J(w^\star)) \;\leq\; (1-\mu\eta+\mu^2(e^2+\beta^2))\mathbb{E}\,\|\widetilde{\boldsymbol{w}}_{i-1}\|^2 - \mathbb{E}\,\|\widetilde{\boldsymbol{w}}_i\|^2 + \mu^2(f^2+\sigma^2). \tag{7.59}$$

To proceed, we simplify the notation and introduce the scalars

$$a(i) \;\triangleq\; \mathbb{E}\,J(\boldsymbol{w}_{i-1}) - J(w^\star) \tag{7.60}$$

$$b(i) \;\triangleq\; \mathbb{E}\,\|\widetilde{\boldsymbol{w}}_i\|^2 \tag{7.61}$$

$$\alpha \;\triangleq\; 1-\mu\eta+\mu^2(e^2+\beta^2) \tag{7.62}$$

$$\tau^2 \;\triangleq\; f^2 + \sigma^2 \tag{7.63}$$

Note that since $w^\star$ is the unique global minimizer of $J(w)$, then it holds that $J(\boldsymbol{w}_{i-1}) \geq J(w^\star)$ so that $a(i) \geq 0$ for all i. The variable $a(i)$ represents the average *excess risk*. Now, we can rewrite (7.59) more compactly as

$$2\mu a(i) \leq \alpha b(i-1) - b(i) + \mu^2\tau^2. \tag{7.64}$$

Iterating over $0 \leq i \leq L$, gives

$$\sum_{i=0}^{L} \alpha^{L-i}(2\mu a(i) - \mu^2\tau^2) \;\;\leq\;\; \alpha^{L+1}b(-1) - b(L) \;\leq\; \alpha^{L+1}b(-1). \tag{7.65}$$

Let us verify that $\alpha \in (0,1)$. First, observe from expression (7.62) for α that $\alpha(\mu)$ is a quadratic function in μ. This function attains its minimum at location $\mu^o = \eta/(2e^2 + 2\beta^2)$. For any μ, the value of $\alpha(\mu)$ is larger than the minimum value of the function at μ^o, i.e., it holds that

$$\alpha \geq 1 - \frac{\eta^2}{4(e^2+\beta^2)}. \tag{7.66}$$

Now, comparing relations (7.32) and (7.26), we find that the sub-gradient vector satisfies:

$$\eta\|w - w^\star\| \leq \|g(w)\| \leq c\|w - w^\star\| + d, \quad \forall w, \tag{7.67}$$

which implies that $\eta \leq c$ since the above inequality must hold for all w. It then follows from (7.43) that $e^2 > \eta^2$ and from (7.66) that

$$\alpha \geq 1 - \frac{\eta^2}{4\eta^2} > 0. \tag{7.68}$$

In other words, the parameter α is positive. Furthermore, some straightforward algebra using (7.62) shows that condition (7.49) implies $\alpha < 1$. We therefore established that $\alpha \in (0,1)$, as desired.

Returning to (7.64), we note that because the (negative) sub-gradient direction is not

necessarily a descent direction, we cannot ensure that $a(i) < a(i-1)$. However, we can still arrive at a useful conclusion by introducing a pocket variable, denoted by $a^{best}(L) \geq 0$. This variable saves the value of the smallest increment, $a(j)$, up to time L, i.e.,

$$a^{best}(L) \stackrel{\Delta}{=} \min_{0 \leq i \leq L} a(i) \; . \tag{7.69}$$

Let further $\boldsymbol{w}_L^{best}$ denote the corresponding iterate $\boldsymbol{w}_i$ where this best value is achieved. Replacing $a(i)$ by $a^{best}(L)$ in (7.65) gives

$$\begin{aligned} (2\mu a^{best}(L) - \mu^2\tau^2) &\leq \alpha^{L+1} b(-1) \left(\sum_{i=0}^{L} \alpha^{L-i} \right)^{-1} \\ &= \alpha^{L+1} b(-1) \left(\frac{1-\alpha^{L+1}}{1-\alpha} \right)^{-1} \\ &= b(-1) \cdot \frac{\alpha^{L+1}(1-\alpha)}{1-\alpha^{L+1}}, \end{aligned} \tag{7.70}$$

or, equivalently,

$$2\mu a^{best}(L) \; \leq \; \mu^2\tau^2 \; + \; b(-1) \cdot \frac{\alpha^{L+1}(1-\alpha)}{1-\alpha^{L+1}}. \tag{7.71}$$

Plugging the definition of $a^{best}(L)$ and $b(-1)$, we will arrive at

$$\mathbb{E}\, J(\boldsymbol{w}_L^{best}) - J(w^\star) \leq \alpha^L \frac{\alpha(1-\alpha)}{2\mu(1-\alpha^{L+1})} \mathbb{E}\, \|\boldsymbol{w}_{-1} - w^\star\|^2 + \mu(f^2+\sigma^2)/2 \tag{7.72}$$

Taking the limit as $L \to \infty$, we conclude that

$$\lim_{L \to \infty} \mathbb{E}\, J(\boldsymbol{w}_L^{best}) - J(w^\star) \leq \mu\tau^2/2 = \mu(f^2+\sigma^2)/2. \tag{7.73}$$

■

Remark #1: It is important to note that result (50) extends and enhances a useful result derived by [149] where the following *lower bound* was established (using our notation):

$$\mathbb{E}\, J(w_i) - J(w^\star) \geq \frac{\zeta_1}{i} \left[\mathbb{E}\, \|w_{-1} - w^\star\|^2 \right] \tag{7.74}$$

for some constant $\zeta_1 > 0$. This result shows that the convergence of $J(w_i)$ towards $J(w^\star)$ cannot be better than a linear rate when one desires convergence towards the exact minimum value. In contrast, our analysis that led to (50) establishes the following *upper bound*:

$$\mathbb{E}\, J(w_i) - J(w^\star) \leq \zeta \alpha^i \left[\mathbb{E}\, \|w_{-1} - w^\star\|^2\right] \;+\; O(\mu) \tag{7.75}$$

for some constant $\zeta > 0$. Observe that this expression is showing that $J(w_i)$ can actually approach a small $O(\mu)-$neighborhood around $J(w_i)$ *exponentially fast* at a rate that is dictated by the scalar $0 < \alpha < 1$. It is clear that the two results (7.74) and (7.75) on the convergence rate do not contradict each other. On the contrary, they provide complementary views on the convergence behavior (from below and from above). Still, it is useful to remark that the analysis employed by Agarwal and Bartlett (2012) imposes a stronger condition on the risk function than our condition: they require the risk function to be Lipschitz continuous. In comparison, we require the subgradient (and not the risk function) to be affine Lipschitz, which makes the current results applicable to a borader class of problems.

■

Remark #2: We can similarly comment on how our result (7.50) relates to the useful results that appear in [157]. Using our notation, the main conclusion that follows from propositions 3.2 and 3.3 is that

$$\mathbb{E}\, J(w_i^{best}) - J(w^\star) \leq \frac{\zeta_2}{i} \mathbb{E}\, \|w_{-1} - w^\star\|^2 + O(\mu) \tag{7.76}$$

for some constant $\zeta_2 > 0$. This result only ensures a sub-linear rate of convergence. In contrast, our convergence analysis leads to the following result

$$\mathbb{E}\, J(w_i^{best}) - J(w^\star) \leq \zeta \alpha^i \left[\mathbb{E}\, \|w_{-1} - w^\star\|^2\right] \;+\; O(\mu) \tag{7.77}$$

for some scalar $0 < \alpha < 1$, which shows that convergence actually occurs at a linear rate. This conclusion is clearly more powerful. Furthermore, as was the case with the treatment in [149], the result (7.76) is derived in [157] by assuming the sub-gradient vectors are bounded, which

is a stronger condition than the affine Lipschitz condition used in the current manuscript. ■

The above theorem only clarifies the performance of the best pocket value, which is not readily available during the algorithm implementation since the risk function itself cannot be evaluated. That is, $J(\boldsymbol{w}_i)$ cannot be computed because $J(w)$ is not known due to the lack of knowledge about the probability distribution of the data. However, a more practical conclusion can be deduced from the statement of the theorem as follows. Introduce the geometric sum:

$$S_L \;\triangleq\; \sum_{j=0}^{L} \alpha^{L-j} \;=\; \frac{1-\alpha^{L+1}}{1-\alpha}, \tag{7.78}$$

as well as the normalized and convex-combination coefficients:

$$r_L(j) \;\triangleq\; \frac{\alpha^{L-j}}{S_L}, \quad j = 0, 1, \ldots, L. \tag{7.79}$$

Using these coefficients, we define the weighted iterate

$$\bar{\boldsymbol{w}}_L \;\triangleq\; \sum_{j=0}^{L} r_L(j)\boldsymbol{w}_j = \frac{1}{S_L}\left[\alpha^L \boldsymbol{w}_0 + \alpha^{L-1}\boldsymbol{w}_1 + \ldots + \alpha \boldsymbol{w}_{L-1} + \boldsymbol{w}_L\right]. \tag{7.80}$$

Observe that, in contrast to $\boldsymbol{w}_L^{best}$, the above weighted iterate is computable since its value depends on the successive iterates $\{\boldsymbol{w}_j\}$ and these are available during the operation of the algorithm. Observe further that $\bar{\boldsymbol{w}}_L$ satisfies the recursive construction:

$$\bar{\boldsymbol{w}}_L \;=\; \left(1 - \frac{1}{S_L}\right)\bar{\boldsymbol{w}}_{L-1} \;+\; \frac{1}{S_L}\boldsymbol{w}_L. \tag{7.81}$$

In particular, as $L \to \infty$, we have $S_L \to 1/(1-\alpha)$, and the above recursion simplifies in the limit to

$$\bar{\boldsymbol{w}}_L \;=\; \alpha\bar{\boldsymbol{w}}_{L-1} \;+\; (1-\alpha)\boldsymbol{w}_L. \tag{7.82}$$

Now, since $J(\cdot)$ is a convex function, it holds that

$$J(\bar{\boldsymbol{w}}_L) \;=\; J\left(\sum_{j=0}^{L} r_L(j)\boldsymbol{w}_j\right) \;\leq\; \sum_{j=0}^{L} r_L(j) J(\boldsymbol{w}_j). \tag{7.83}$$

Using this fact, the following corollary derives a result similar to (7.52) albeit applied to $\bar{\boldsymbol{w}}_L$.

Corollary 7.1 (Weighted iterate) *Under the same conditions as in Theorem 7.1, it holds that*

$$\lim_{L\to\infty} \mathbb{E}\, J(\bar{\boldsymbol{w}}_L) - J(w^\star) \leq \mu(f^2+\sigma^2)/2, \tag{7.84}$$

and the convergence of $\mathbb{E}\, J(\bar{\boldsymbol{w}}_L)$ towards $J(w^\star)$ continues to occur at the same exponential rate, $O(\alpha^L)$.

Proof: We start from (7.65), namely,

$$\sum_{i=0}^{L} \alpha^{L-i}(2\mu\mathbb{E}\, J(\boldsymbol{w}_i) - 2\mu J(w^\star) - \mu^2\tau^2) \leq \alpha^{L+1} b(-1), \tag{7.85}$$

and divide both sides by the same sum:

$$\sum_{i=0}^{L} \left(\frac{\alpha^{L-i}}{\sum_{j=0}^{L}\alpha^{L-j}}\right)(2\mu\mathbb{E}\, J(\boldsymbol{w}_i) - 2\mu J(w^\star) - \mu^2\tau^2) \leq \left(\frac{\alpha^{L+1}}{\sum_{j=0}^{L}\alpha^{L-j}}\right) b(-1), \tag{7.86}$$

which gives

$$\sum_{i=0}^{L} r(i)(2\mu\mathbb{E}\, J(\boldsymbol{w}_i) - 2\mu J(w^\star)) \leq \frac{\alpha^{L+1}(1-\alpha)}{1-\alpha^{L+1}} b(-1) \;+\; \mu^2\tau^2. \tag{7.87}$$

Appealing to the convexity property (7.83) we conclude that

$$2\mu(\mathbb{E}\, J(\bar{\boldsymbol{w}}_L) - 2\mu J(w^\star)) \leq \frac{\alpha^{L+1}(1-\alpha)}{1-\alpha^{L+1}} b(-1) \;+\; \mu^2\tau^2. \tag{7.88}$$

Taking the limit as $L\to\infty$ leads to (7.84). ■

Using α as a scaling weight in (7.80) may still be inconvenient because its value needs to be determined. The analysis however suggests that we may replace α by any parameter κ satisfying $\alpha \leq \kappa \leq 1$. The parameter κ plays a role similar to the step-size, μ: both become

parameters selected by the designer. Next, we introduce the new weighted variable:

$$\bar{\boldsymbol{w}}'_L \;\triangleq\; \sum_{j=0}^{L} r'_L(j)\boldsymbol{w}_j, \tag{7.89}$$

where now

$$r'_L(j) = \kappa^{L-j}/S'_L, \;\; j = 0, 1, \ldots, L, \tag{7.90}$$

and

$$S'_L = \sum_{j=0}^{L} \kappa^{L-j}. \tag{7.91}$$

Corollary 7.2 (Relaxed Weighted iterate) *Under the same conditions as in Theorem 7.1 and $\alpha \leq \kappa < 1$, relation (7.84) continues to hold with $\bar{\boldsymbol{w}}_L$ replaced by $\bar{\boldsymbol{w}}'_L$. Moreover, convergence now occurs at the exponential rate $O(\kappa^L)$.*

Proof: The argument requires some modification relative to what we have done before. We start from (7.64) again:

$$2\mu a(i) \leq \alpha b(i-1) - b(i) + \mu^2\tau^2. \tag{7.92}$$

But unlike the previous derivation in (7.65), now we use κ to expand the recursion from iteration $i = 0$ to L:

$$\begin{aligned}
\sum_{i=0}^{L} \kappa^{L-i}(2\mu a(i) - \mu^2\tau^2) &\leq \sum_{i=0}^{L} \kappa^{L-i}\Big(\alpha b(i-1) - b(i)\Big) \\
&= \sum_{i=-1}^{L-1} \kappa^{L-i-1}\alpha b(i) - \sum_{i=0}^{L} \kappa^{L-i} b(i) \\
&= \sum_{i=-1}^{L-1} \kappa^{L-i-1}(\alpha - \kappa) b(i) + \kappa^{L+1} b(-1) - b(L) \\
&\leq \kappa^{L+1} b(-1),
\end{aligned} \tag{7.93}$$

where in the last inequality we used the fact that $\kappa \geq \alpha$. We can now proceed from here and complete the argument as before. ■

Corollary 7.3 (Mean-Square-Deivation Performance) *Under the same conditions as in Theorem 7.1 and $\alpha \leq \kappa < 1$. It holds that*

$$\lim_{L\to\infty} \mathbb{E}\,\|\bar{\boldsymbol{w}}'_L - w^\star\|^2 \leq \mu(f^2+\sigma^2)/\eta \tag{7.94}$$

Moreover, convergence to the steady-state regime occurs at the exponential rate $O(\kappa^L)$.

Proof: Referring to equation (7.29), we get

$$\mathbb{E}\,\|\bar{\boldsymbol{w}}'_i - w^\star\|^2 \leq \frac{2}{\eta}\left(\mathbb{E}\,J(\bar{\boldsymbol{w}}'_i) - J(w^\star)\right) \tag{7.95}$$

Combining with the corollary 7.2, we arrive at (7.94). ■

It is interesting to compare result (7.84) with what happens in the case of *differentiable* risk functions. In that case, the standard stochastic gradient algorithm, using the actual gradient vector rather than sub-gradients, can be employed to seek the minimizer, $w^\star$. It was established in [11, Ch. 4] that for risk functions that are twice-differentiable, the stochastic gradient algorithm guarantees

$$\lim_{L\to\infty} \mathbb{E}\,J(\boldsymbol{w}_L) - J(w^\star) = \frac{\mu}{4}\sigma^2, \tag{7.96}$$

where the right-hand side is dependent on σ^2 alone; this factor arises from the bound (7.22) on the gradient noise process. In contrast, in the non-smooth case (7.84), we established here a similar bound that is still in the order of $O(\mu)$. However, the size of the bound is not solely dependent on σ^2 anymore but it also includes the factor f^2; this latter factor arises from condition (7.40) on the sub-gradient vectors. That is, there is some minor degradation (since μ is small) that arises from the non-smoothness of the risk function. If we set $f = 0$ in (7.84), we recover (7.96) up to a scaling factor of 2. Although the bound in this case is still $O(\mu)$, as desired, the reason why it is not as tight as the bound derived in the smooth case in [11] is because the derivation in the current paper is not requiring the risk function to be twice differentiable, as was the case in [11], and we are also discarding the term $b(L)$ in equation (7.65). The important conclusion to note is that the right-hand side of (7.84) is

also $O(\mu)$, as in the smooth case (7.96).

7.4.4 Interpretation of Results

The results derived in this section highlight several important facts that we would like to summarize:

(1) First, it has been observed in the optimization literature that sub-gradient descent iterations can perform poorly in *deterministic* problems (where $J(w)$ is known). Their convergence rate is $O(1/\sqrt{i})$ under convexity and $O(1/i)$ under strong-convexity [136] when decaying step-sizes, $\mu(i) = 1/i$, are used to ensure convergence [60]. Our arguments show that the situation is different in the context of *stochastic* optimization when true sub-gradients are approximated from streaming data. By using *constant* step-sizes to enable continuous learning and adaptation, the sub-gradient iteration is now able to achieve exponential convergence at the rate of $O(\alpha^i)$ for some $\alpha = 1 - O(\mu)$.

(2) Second, of course, this substantial improvement in convergence rate comes at a cost, but one that is acceptable and controllable. Specifically, we cannot guarantee convergence of the algorithm to the global minimum value, $J(w^\star)$, anymore but can instead approach this optimal value with high accuracy in the order of $O(\mu)$, where the size of μ is under the designer's control and can be selected as small as desired.

(3) Third, this performance level is sufficient in most cases of interest because, in practice, one rarely has an infinite amount of data and, moreover, the data is often subject to distortions not captured by any assumed models. It is increasingly recognized in the literature that it is not always necessary to ensure exact convergence towards the optimal solution, $w^\star$, or the minimum value, $J(w^\star)$, because these optimal values may not reflect accurately the true state due to modeling error. For example, it is explained in the works [18, 19, 144] that it is generally unnecessary to reduce the error measures below the statistical error level that is present in the data.

7.5 Applications: Single Agent Case

We now apply the results of the previous analysis to several cases in order to illustrate that *stochastic* sub-gradient constructions can indeed lead to good performance.

Example 5 (LASSO problem). For the LASSO problem, we choose $R = 0.5$ so that

$$f^2 = d^2 + cd = 4\delta^2 M + 2\delta M^{\frac{1}{2}}. \tag{7.97}$$

It then follows that

$$\lim_{L\to\infty} \mathbb{E}\, J^{lasso}(\bar{\boldsymbol{w}}_L) - J^{lasso}(w^\star) \leq \mu(2\delta^2 M + \delta M^{\frac{1}{2}}) + \frac{\mu}{2}\sigma_n^2 \mathrm{Tr}(R_h). \tag{7.98}$$

In order to verify this result, we run a simulation with $\mu = 0.001$, $\delta = 0.002$, and $M = 100$. Only two entries in w° are assumed to be nonzero. The regression vectors and noise process $\{\boldsymbol{h}_i, \boldsymbol{n}(i)\}$ are both generated according to zero-mean normal distributions with variances $R_h = I$ and $\sigma_n^2 = 0.01$, respectively. From the optimality condition, $0 \in \partial J(\boldsymbol{w}^\star)$, it is easy to conclude that [158]

$$w^\star = \mathcal{S}_\delta(w^\circ), \tag{7.99}$$

where the symbol $\mathcal{S}_\delta$ represents the soft-thresholding function with parameter δ, i.e.,

$$\mathcal{S}_\delta(x) = sgn(x) \cdot \max\{0, |x| - \delta\}. \tag{7.100}$$

Figure 7.1 plots the evolution of the excess-risk curve, $\mathbb{E}\, J^{lasso}(\bar{\boldsymbol{w}}_L) - J^{lasso}(w^\star)$, obtained by averaging over 50 experiments. The figure compares the performance of the standard LMS solution:

$$\boldsymbol{w}_i = \boldsymbol{w}_{i-1} + \mu \boldsymbol{h}_i(\boldsymbol{\gamma}(i) - \boldsymbol{h}_i^{\mathsf{T}} \boldsymbol{w}_{i-1}), \tag{7.101}$$

against the sparse sub-gradient version [153, 159, 160]:

$$\boldsymbol{w}_i = \boldsymbol{w}_{i-1} + \mu \boldsymbol{h}_i(\boldsymbol{\gamma}(i) - \boldsymbol{h}_i^{\mathsf{T}} \boldsymbol{w}_{i-1}) - \mu\delta \cdot sgn(\boldsymbol{w}_{i-1}). \tag{7.102}$$

It is observed that the stochastic sub-gradient implementation satisfies the bound predicted by theory.

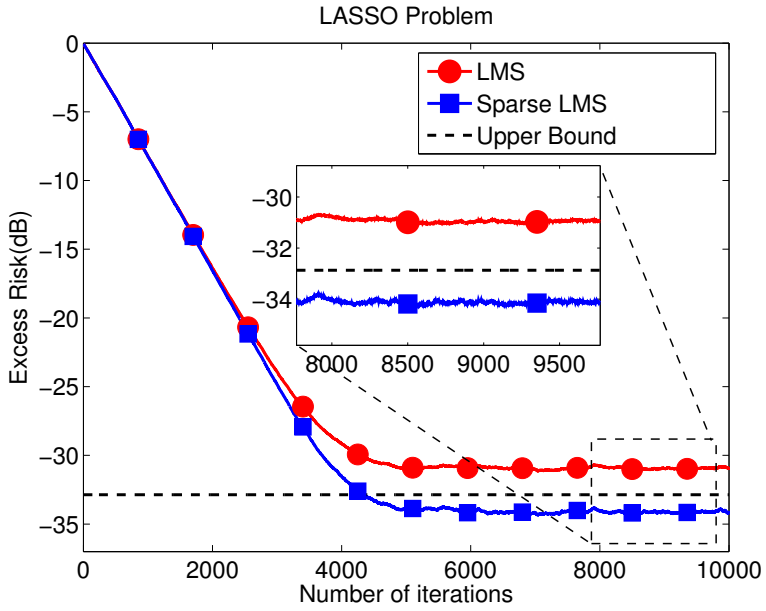

Figure 7.1: LASSO problem. The excess-risk curves, i.e. $\big(\mathbb{E}\, J(\boldsymbol{w}_L) - J(w^\star)\big)$, for LMS and for LASSO-LMS are obtained by averaging over 50 experiments.

□

Example 6 (SVM problem). For the SVM problem, we again select $R = 0.5$ and conclude that

$$\lim_{L\to\infty} \mathbb{E}\, J^{svm}(\bar{\boldsymbol{w}}_L) - J^{svm}(w^\star) \leq \frac{\mu}{2}\left(5\mathrm{Tr}(R_h) + 2\rho(\mathrm{Tr}(R_h))^{\frac{1}{2}}\right). \tag{7.103}$$

Actually, for the SVM construction, we can obtain a tighter upper bound than the one provided by Corollary 7.1; this is because we can exploit the special structure of the SVM cost to arrive at

$$\lim_{L\to\infty} \mathbb{E}\, J^{svm}(\bar{\boldsymbol{w}}_L) - J^{svm}(w^\star) \leq \mu\big(\rho^2\|w^\star\|^2 + \rho + \mathrm{Tr}(R_h)/2\big), \tag{7.104}$$

with convergence rate $\alpha = 1 - 2\mu\rho + \mu^2\rho^2$. The proof is provided in Appendix 7.A. We

compare the performance of the stochastic sub-gradient SVM implementation:

$$\begin{cases} \boldsymbol{w}_i = (1-\rho\mu)\boldsymbol{w}_{i-1} - \mu\boldsymbol{\gamma}(i)\boldsymbol{h}_i\mathbb{I}[\boldsymbol{\gamma}(i)\boldsymbol{h}_i^{\mathsf{T}}\boldsymbol{w}_{i-1} \leq 1] \\ S_i = (1-2\mu\rho+\mu^2\rho^2)S_{i-1}+1 \\ \bar{\boldsymbol{w}}_i = \left(1-\dfrac{1}{S_i}\right)\bar{\boldsymbol{w}}_{i-1} + \dfrac{1}{S_i}\boldsymbol{w}_i \end{cases}, \tag{7.105}$$

(with all variables initiate at zero) against LIBSVM (a popular SVM solver that uses dual quadratic programming) [6]. The test data is obtained from the LIBSVM website[2] and also from the UCI dataset[3]. We first use the Adult dataset after preprocessing [104] with 11,220 training data and 21,341 testing data in 123 feature dimensions. To ensure a fair comparison, we use linear LIBSVM with the exact same parameters as the sub-gradient method. Hence, we choose $C = 5 \times 10^2$ for LIBSVM, which corresponds to $\rho = \frac{1}{C} = 2 \times 10^{-3}$. We also set $\mu = 0.05$.

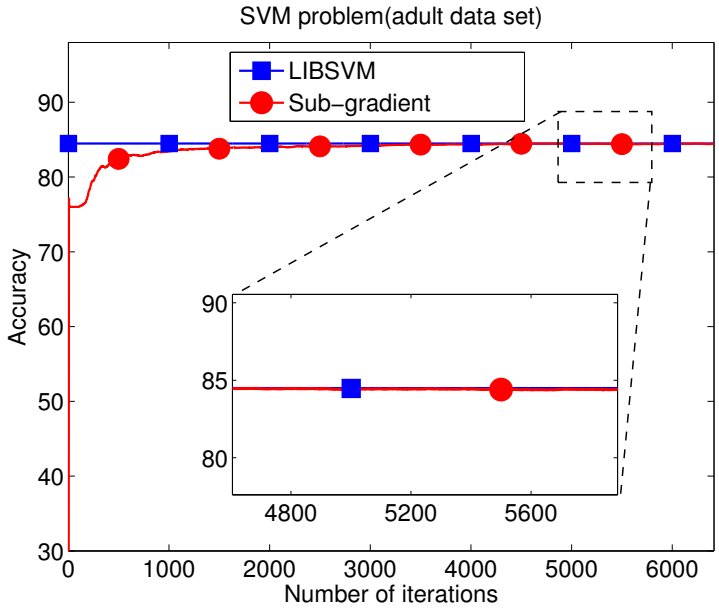

Figure 7.2: SVM solvers applied to the Adult data set. Comparison of the performance accuracy, percentage of correct prediction over test dataset, for LIBSVM [6] and a stochastic sub-gradient implementation (7.105).

[2] http://www.csie.ntu.edu.tw/~cjlin/libsvmtools/datasets/binary.html

[3] http://archive.ics.uci.edu/ml/

We can see from Fig. 7.2 that the stochastic sub-gradient algorithm is able converge to the performance of LIBSVM quickly. Since we only use each data point once, and since each iteration is computationally simpler, the sub-gradient implementation ends up being computationally more efficient. We also examine the performance of the sub-gradient SVM solver on another large-scale dataset, namely, the Reuters Corpus Volume I (RCV1) data with 20242 training data and 253843 testing data consisting of 47236 feature dimensions. The chosen parameters are $C = 1 \times 10^5, \mu = 0.2$. The performance is shown in Fig. 7.3.

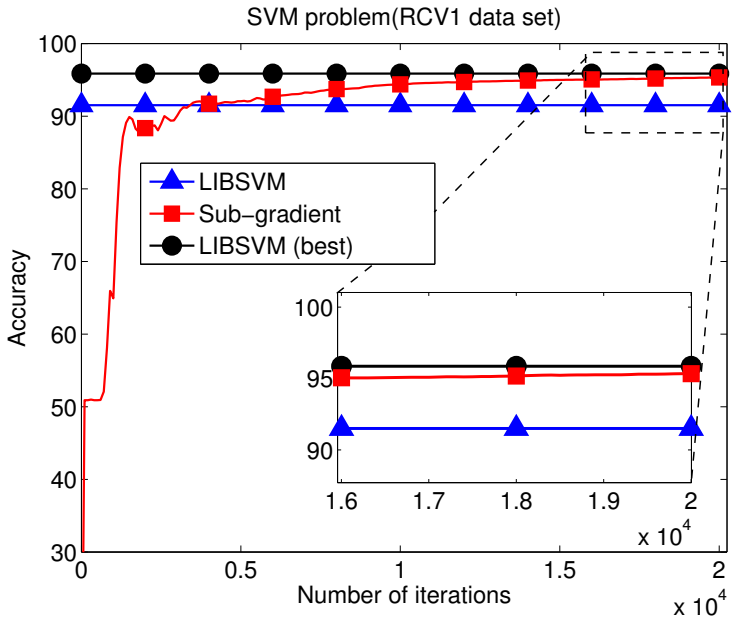

Figure 7.3: SVM solvers applied to the RCV1 data set. Comparison of the performance accuracy, percentage of correct prediction over test dataset, for LIBSVM [6] and a stochastic sub-gradient implementation (7.105). The blue line for LIBSVM is generated by using the same parameters as the sub-gradient implementation, while the black line is determined by using cross validation. The difference between both lines is because LIBSVM achieves higher accuracy when setting ρ to a large value, say, around the value of one. In comparison, from (7.104) we know that sub-gradient methods need a small ρ to achieve higher accuracy.

□

Example 7 (Image denoising problem). We next illustrate how the stochastic sub-gradient implementation can match the performance of some sophisticated techniques for image denoising, such as the FISTA algorithm. This latter technique solves the denoising problem by relying on the use of proximal projections and acceleration methods applied to

a dual problem [7, 161].

One classical formulation for the image denoising problem with total-variation regularization involves seeking an image (or matrix) that minimizes the following deterministic cost [140]:

$$\min_{\mathcal{I}} \ \frac{1}{2}\|\mathcal{I}-\mathcal{I}_{\mathrm{n}oisy}\|_F^2+\lambda\cdot \mathrm{T}V(\mathcal{I}), \tag{7.106}$$

where $\lambda > 0$ is a regularization factor. Moreover, the term $\mathcal{I}$ denotes some rectangular or square image that we wish to recover, say, of size $N \times N$, and $\mathcal{I}_{\mathrm{n}oisy}$ refers to the available noisy measurement of the true image:

$$\mathcal{I}_{\mathrm{n}oisy} \ = \ \mathcal{I}^o \ + \ \text{noise}, \tag{7.107}$$

where the noise term refers to a zero-mean perturbation. The notation $\|\cdot\|_F$ denotes the Frobenious norm of its matrix argument, and the operation $\mathrm{T}V(\cdot)$ stands for a total-variation computation, which is defined as follows:[4]

$$\mathrm{T}V(\mathcal{I}) \ \triangleq \ \sum_{m,n} |\mathcal{I}(m,n)-\mathcal{I}(m+1,n)|+|\mathcal{I}(m,n)-\mathcal{I}(m,n+1)|. \tag{7.108}$$

The total variation term essentially encourages the difference between the image and some of its shifted versions to remain nearly sparse. We may also formulate a stochastic version of the denoising problem by considering instead:

$$\min_{\mathcal{I}} \ \frac{1}{2}\mathbb{E}\,\|\mathcal{I}-\boldsymbol{\mathcal{I}}_{\mathrm{n}oisy}\|_F^2+\lambda\cdot \mathrm{T}V(\mathcal{I}), \tag{7.109}$$

where the expectation is now over the randomness in the noise used to generate the noisy image (here we only consider the synthesis case). The sub-gradient of the Total Variation term is straightforward to compute. For illustration purposes, we evaluate the sub-gradient at some arbitrary point (m_0, n_0). Expanding the summation and separating the terms related

[4]Here, we only consider the discrete ℓ_1-based anisotropic TV and neglect the boundary modification.

to point (m_0, n_0), we obtain:

$$\begin{aligned}\mathrm{TV}(\mathcal{I}) &= |\mathcal{I}(m_0,n_0)-\mathcal{I}(m_0+1,n_0)|+|\mathcal{I}(m_0,n_0)-\mathcal{I}(m_0,n_0+1)|+\\ &\quad |\mathcal{I}(m_0-1,n_0)-\mathcal{I}(m_0,n_0)|+|\mathcal{I}(m_0,n_0-1)-\mathcal{I}(m_0,n_0)|+\mathrm{rest}, \quad (7.110)\end{aligned}$$

where the *rest* variable refers to terms that do not contain the variable $\mathcal{I}(m_0, n_0)$. Computing the sub-gradient with respect to $\mathcal{I}(m_0, n_0)$ will generate four terms with the sign function as in the LASSO problem. It is then clear that stochastic sub-gradient implementation in this case is given by:

$$\boldsymbol{\mathcal{I}}_i = \boldsymbol{\mathcal{I}}_{i-1} + \mu\left(\boldsymbol{\mathcal{I}}_{i-1} - \boldsymbol{\mathcal{I}}_{\mathrm{noisy}} + \sum_{j=1}^{4} \lambda \cdot \mathrm{sgn}(\boldsymbol{\mathcal{I}}_{i-1} - \boldsymbol{\mathcal{I}}_{i-1}^{j})\right), \quad (7.111)$$

where $\boldsymbol{\mathcal{I}}_i$ represents the recovered image at iteration i, $\boldsymbol{\mathcal{I}}_i^1$ represents shifting the image to the left by one pixel, while $\boldsymbol{\mathcal{I}}_i^2, \boldsymbol{\mathcal{I}}_i^3, \boldsymbol{\mathcal{I}}_i^4$ represent shifting the image to the right, up, and down by one pixel, respectively. We observe that recursion (7.111) now iterates repeatedly over the *same* single image, $\boldsymbol{\mathcal{I}}_{\mathrm{noisy}}$. Accordingly, in this example, the stochastic gradient noise does not vary over time, i.e.,

$$s_i(\boldsymbol{\mathcal{I}}_i) \;=\; \mathcal{I}^o - \boldsymbol{\mathcal{I}}_{\mathrm{noisy}}, \quad \forall i. \quad (7.112)$$

Nevertheless, Assumption 7.1 still holds; it was not required there that the gradient noise process cannot be independent of time. Table 7.5 lists performance results using the Kodak image suite[5]. The table lists two metrics. The first metric is the PSNR defined as

$$\mathrm{P}SNR = 10 \times \log \frac{(255)^2}{\mathrm{MSE}}, \quad (7.113)$$

where MSE represents the mean-square-error, and the second metric is the execution time. For a fair comparison, we used similar un-optimized MATLAB codes[6] under the same com-

[5] `http://r0k.us/graphics/kodak/`

[6] Code for FISTA is available at `http://iew3.technion.ac.il/~becka/papers/tv_fista.zip`.

	Test Image	kodim1	kodim5	kodim7	kodim8	kodim11	kodim14
PSNR(dB)	Sub-gradient	**25.19**	**25.18**	**29.43**	**24.59**	**27.80**	**30.32**
	FISTA	24.90	24.87	29.14	24.26	27.59	30.25
Time(s)	Sub-gradient	**8.88**	**9.33**	**8.50**	8.46	**9.00**	**8.7**
	FISTA	9.16	9.78	10.15	**8.22**	10.24	9.19
	Test Image	kodim15	kodim17	kodim19	kodim21	kodim23	kodim24
PSNR(dB)	Sub-gradient	**30.32**	**29.38**	**27.53**	**27.29**	**31.68**	**26.25**
	FISTA	30.25	29.17	27.23	27.03	31.51	25.95
Time(s)	Sub-gradient	**9.07**	**9.50**	9.60	**9.45**	**9.17**	8.88
	FISTA	10.13	9.98	**9.47**	9.62	10.17	**8.51**

Table 7.1: Comparison between the stochastic sub-gradient method (7.111) and FISTA [7] over the KODIM test image set (c.f. footnote 4). All test images are subject to additive zero-mean Gaussian noise with standard variance 0.1 (with respect to image values in the range $[0, 1]$). We set $\lambda = 0.08, \mu = 0.002$ and 300 max iterations for sub-gradient methods. For different values of λ and μ, the results will be different, but the algorithms will perform similarly when μ is chosen properly. The results in the table show that the sub-gradient implementation can, in general, achieve similar or higher PSNR in shorter time.

puter environment. The table shows that the sub-gradient implementation can achieve comparable or higher PSNR values in shorter time. Clearly, if we vary the algorithm parameters, these values will change. However, in general, it was observed in these experiments that the sub-gradient implementation succeeds in matching the performance of FISTA reasonably well. □

7.6 Problem Formulation: Multi-Agent Case

We now extend the previous analysis to multi-agent networks where a collection of agents cooperate with each other to seek the minimizer of an aggregate cost of the form:

$$\min_{w} \sum_{k=1}^{N} J_k(w), \tag{7.114}$$

where k refers to the agent index. Each individual risk function continues to be expressed as the expected value of some loss function:

$$J_k(w) \triangleq \mathbb{E}\, Q_k(w; \boldsymbol{x}_k). \tag{7.115}$$

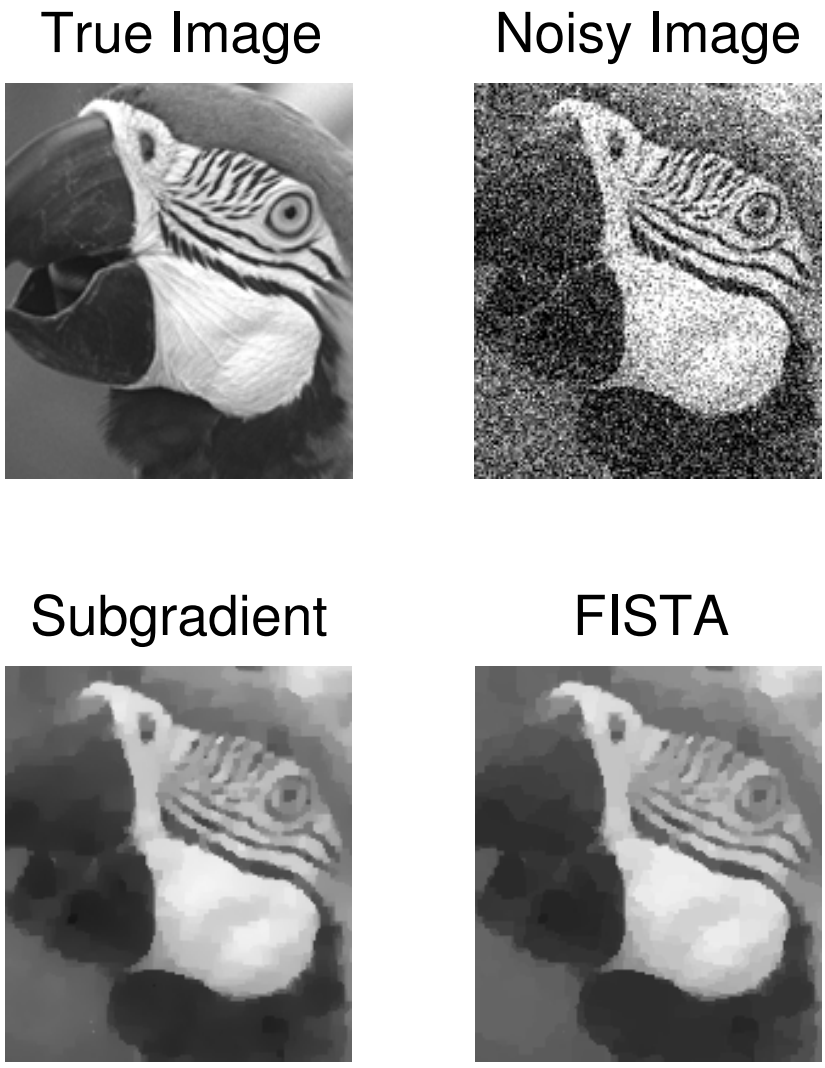

Figure 7.4: Comparison of the performance of FISTA and sub-gradient implementations on test image Kodim 23 (zoom-in) under low PSNR (13dB). The result of the sub-gradient implementation is almost indistinguishable from the result of FISTA.

The expectation is over the data at agent k. We continue to assume that the individual costs satisfy Assumptions 7.2 and 7.3, i.e., each $J_k(w)$ is strongly-convex and its sub-gradient vectors are affine-Lipschitz with parameters $\{\eta_k, c_k, d_k\}$; we are attaching a subscript k to these parameters to make them agent-dependent (alternatively, if desired, we can replace

them by agent-independent parameters by using bounds on their values). We further assume that the individual risks share a common minimizer, $w^\star$, which will therefore agree with the global minimizer for (7.114). This scenario corresponds to the important situation in which the agents have a common objective (or task), namely, that of estimating the same parameter vector, $w^\star$, in a distributed manner through *localized* interactions and cooperation.

7.6.1 Network Model

Thus, consider a network consisting of N separate agents connected by a topology. As described in [11,28], we assign a pair of nonnegative weights, $\{a_{k\ell}, a_{\ell k}\}$, to the edge connecting any two agents k and ℓ. The scalar $a_{\ell k}$ is used by agent k to scale the data it receives from agent ℓ and similarly for $a_{k\ell}$. The network is said to be *connected* if paths with nonzero scaling weights can be found linking any two distinct agents in both directions. The network is said to be *strongly–connected* if it is connected with at least one self-loop, meaning that $a_{kk} > 0$ for some agent k. Figure 7.5 shows one example of a strongly–connected network. For emphasis in this figure, each edge between two neighboring agents is represented by two directed arrows. The neighborhood of any agent k is denoted by $\mathcal{N}_k$ and it consists of all agents that are connected to k by edges; we assume by default that this set includes agent k regardless of whether agent k has a self-loop or not.

There are several strategies that the agents can employ to seek the minimizer, $w^\star$, including consensus and diffusion strategies [11,28,48,58,162]. In this chapter, we focus on the latter class since diffusion implementations have been shown to have superior stability and performance properties over consensus strategies when used in the context of adaptation and learning from streaming data (i.e., when the step-sizes are set to a constant value as opposed to a diminishing value) [11,28,163]. As explained earlier, diminishing step-sizes annihilate the gradient noise term but disable adaptation and learning in the long run. On the other hand, constant step-size updates keep adaptation alive, which permits gradient noise to seep into the operation of the algorithm. The challenge is to show that the dynamics of the algorithm over the network is such that this noise effect does not degrade performance and that the

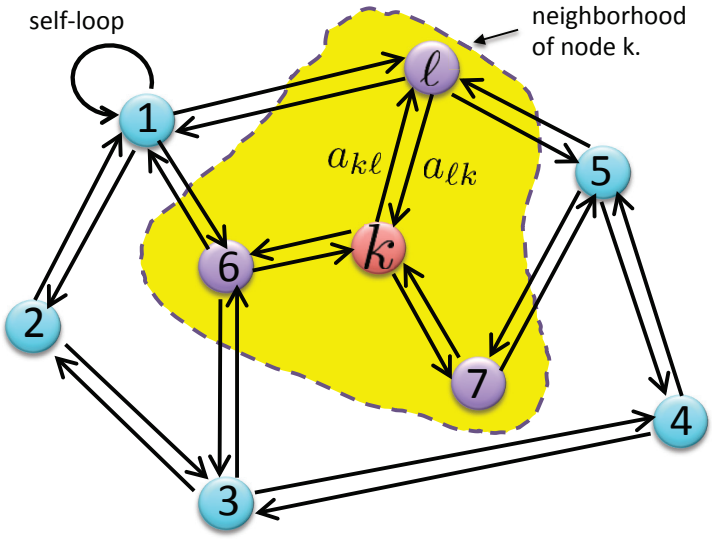

Figure 7.5: Agents that are linked by edges can share information. The neighborhood of agent k is marked by the broken line and consists of the set $\mathcal{N}_k = \{6, 7, \ell, k\}$.

network will still able to learn the unknown. This kind of analysis has been answered before in the affirmative for smooth twice-differentiable functions, $J_k(w)$ — see [11, 28, 102, 146]. In this chapter, we want to pursue the analysis more generally for possibly *non-differentiable* costs in order to encompass important applications (such as SVM learning by multi-agents or LASSO and sparsity-aware learning by similar agents [164–167]). We also want to pursue the analysis under the weaker affine-Lipschitz assumption on the sub-gradients than the stronger conditions used in the prior literature, as we already explained in the earlier sections.

7.6.2 Distributed Strategy

We therefore consider the following diffusion strategy in its adapt-then-combine (ATC) form:

$$\begin{cases} \boldsymbol{\psi}_{k,i} = \boldsymbol{w}_{k,i-1} - \mu\, \widehat{g}_k(\boldsymbol{w}_{k,i-1}) \\ \boldsymbol{w}_{k,i} = \displaystyle\sum_{\ell \in \mathcal{N}_k} a_{\ell k} \boldsymbol{\psi}_{\ell,i} \end{cases} . \tag{7.116}$$

Here, the first step involves adaptation by agent k by using a stochastic sub-gradient iteration, while the second step involves aggregation; we assume the gradient noise processes

across all agents are independent of each other. The entries $A = [a_{\ell k}]$ define a left-stochastic matrix, namely, the entries of A are non-negative and each of its columns adds up to one. Since the network is strongly-connected, the combination matrix A will be primitive [11,168]. This implies that A will admit a Jordan-decomposition of the form:

$$A = V_\epsilon J V_\epsilon^{-1} \triangleq \left[\begin{array}{c|c} p & V_R \end{array}\right] \left[\begin{array}{c|c} 1 & 0 \\ \hline 0 & J_\epsilon \end{array}\right] \left[\begin{array}{c} \mathbb{1}^{\mathsf{T}} \\ \hline V_L^{\mathsf{T}} \end{array}\right], \tag{7.117}$$

with a single eigenvalue at one and all eigenvalues strictly inside the unit circle. The matrix J_ϵ has a Jordan structure with the ones that would typically appear along its first sub-diagonal replaced by a small positive number, $\epsilon > 0$. Note that the eigenvectors of A corresponding to the eigenvalue at one are denoted by

$$Ap = p, \quad A^{\mathsf{T}}\mathbb{1} = \mathbb{1}. \tag{7.118}$$

It is further known from the Perron-Frobenius theorem [168] that the entries of p are all strictly positive and we normalize them to add up to one. We denote the individual entries of p by $\{p_k\}$:

$$p_k > 0, \quad \sum_{k=1}^{N} p_k = 1. \tag{7.119}$$

Furthermore, since $V_\epsilon V_\epsilon^{-1} = I$, it holds that

$$V_R^{\mathsf{T}}\mathbb{1} = 0, \quad V_L^{\mathsf{T}} p = 0, \quad V_L^{\mathsf{T}} V_R = I. \tag{7.120}$$

7.6.3 Network Performance

We are now ready to establish the following extension of Theorem 7.1 to the network case. The result establishes that the distributed strategy is stable and converges exponentially fast for sufficiently small step-sizes. The statement below is again in terms of a pocket variable, which we define as follows.

At every iteration i, the risk value that is attained by iterate $\boldsymbol{w}_{k,i}$ is $J_k(\boldsymbol{w}_{k,i})$. This value

is a random variable due to the randomness in the streaming data used to run the algorithm. We denote the mean risk value at agent k by $\mathbb{E}\, J_k(\boldsymbol{w}_{k,i})$. We again introduce a *best pocket* iterate, denoted by $\boldsymbol{w}_{k,i}^{best}$. At any iteration i, the value that is saved in the pocket variable is the iterate, $\boldsymbol{w}_{k,j}$, that has generated the smallest mean risk value up to time i, i.e.,

$$\boldsymbol{w}_{k,i}^{\text{best}} \;\triangleq\; \arg\min_{0\leq j\leq i} \;\mathbb{E}\; J_k(\boldsymbol{w}_{k,j}). \tag{7.121}$$

Observe that we now have N pocket values, one for each agent k.

Theorem 7.2 (Network performance) *Consider using the stochastic sub-gradient diffusion algorithm (7.116) to seek the unique minimizer, $w^\star$, of the optimization problem (7.114), where the risk functions, $J_k(w)$, are assumed to satisfy Assumptions 7.1–7.3 with parameters $\{\eta_k, \beta_k^2, \sigma_k^2, e_k^2, f_k^2\}$. Assume the step-size parameter is sufficiently small (see condition (7.178)). Then, it holds that*

$$\mathbb{E}\left(\sum_{k=1}^{N} p_k J_k(\boldsymbol{w}_{k,i}^{best}) - \sum_{k=1}^{N} p_k J_k(w^\star)\right) \leq \xi\cdot\alpha_q^i \sum_{k=1}^{N} p_k \|w_{k,0}-w^\star\| + \frac{\mu}{2}\sum_{k=1}^{N}\left(p_k f_k^2 + p_k^2\sigma_k^2 + 2p_k f_k h\right) \tag{7.122}$$

for some finite constants h and ξ. The convergence to steady-state regime occurs at an exponential rate, $O(\alpha_q^i)$, dictated by the parameter

$$\alpha_q \;\triangleq\; \max_k \left\{1 - \mu\eta_k + \mu^2 e_k^2 + \mu^2\beta_k^2 p_k + \mu^2 h\frac{e_k^2}{f_k}\right\} = 1 - O(\mu). \tag{7.123}$$

Condition (7.178) further ahead ensures $\alpha \in (0,1)$.

Proof: The argument is provided in Appendix 7.B. ■

The above theorem clarifies the performance of the network in terms of the best pocket values across the agents. However, these pocket values are not readily available because the risk values, $J_k(\boldsymbol{w}_{k,i})$, cannot be evaluated. This is due to the fact that the statistical properties of the data are not known beforehand. As was the case with the single-agent scenario, a more practical conclusion can be deduced from the statement of the theorem as

follows. We again introduce the geometric sum

$$S_L \triangleq \sum_{j=0}^{L} \alpha_q^{L-j} = \frac{1-\alpha_q^{L+1}}{1-\alpha_q}, \tag{7.124}$$

as well as the normalized and convex-combination coefficients:

$$r_L(j) \triangleq \frac{\alpha_q^{L-j}}{S_L}, \quad j = 0, 1, \ldots, L. \tag{7.125}$$

Using these coefficients, we define a weighted iterate at each agent:

$$\bar{\boldsymbol{w}}_{k,L} \triangleq \sum_{j=0}^{L} r_L(j)\boldsymbol{w}_{k,j} = \frac{1}{S_L}\left[\alpha_q^L \boldsymbol{w}_{k,0} + \ldots + \alpha_q \boldsymbol{w}_{k,L-1} + \boldsymbol{w}_{k,L}\right], \tag{7.126}$$

and observe that $\bar{\boldsymbol{w}}_{k,L}$ satisfies the recursive construction:

$$\bar{\boldsymbol{w}}_{k,L} = \left(1 - \frac{1}{S_L}\right)\bar{\boldsymbol{w}}_{k,L-1} + \frac{1}{S_L}\boldsymbol{w}_{k,L}. \tag{7.127}$$

In particular, as $L \to \infty$, we have $S_L \to 1/(1-\alpha_q)$, and the above recursion simplifies in the limit to

$$\bar{\boldsymbol{w}}_{k,L} = \alpha_q \bar{\boldsymbol{w}}_{k,L-1} + (1-\alpha_q)\boldsymbol{w}_{k,L}. \tag{7.128}$$

Corollary 7.4 (Weighted iterates) *Under the same conditions as in Theorem 7.2, it holds that*

$$\lim_{L\to\infty} \mathbb{E}\left(\sum_{k=1}^{N} p_k J_k(\bar{\boldsymbol{w}}_{k,L}) - \sum_{k=1}^{N} p_k J_k(w^\star)\right) \leq \frac{\mu}{2}\sum_{k=1}^{N}\left(p_k f_k^2 + p_k^2\sigma_k^2 + 2hp_k f_k\right) = O(\mu), \tag{7.129}$$

and convergence continues to occur at the same exponential rate, $O(\alpha_q^L)$. ■

Result (7.129) is an interesting conclusion. However, the statement is in terms of the averaged iterate $\bar{\boldsymbol{w}}_{k,L}$ whose computation requires knowledge of α_q. This latter parameter is a global information, which is not readily available to all agents. Nevertheless, result

(7.129) motivates the following useful distributed implementation with a similar guaranteed performance bound. We can replace α_q by a design parameter, κ, that is no less than α_q but still smaller than one, i.e., $\alpha_q \leq \kappa < 1$. Next, we introduce the weighted variable:

$$\bar{\boldsymbol{w}}'_{k,L} \quad \stackrel{\Delta}{=} \quad \sum_{j=0}^{L} r'_L(j)\boldsymbol{w}_{k,j}, \tag{7.130}$$

where

$$r'_L(j) = \kappa^{L-j}/S'_L, \;\; j = 0, 1, \ldots, L, \tag{7.131}$$

and

$$S'_L = \sum_{j=0}^{L} \kappa^{L-j}. \tag{7.132}$$

Corollary 7.5 (Distributed Weighted iterates) *Under the same conditions as in Theorem 7.2 and $\alpha_q \leq \kappa < 1$, relation (7.129) continues to hold with $\bar{\boldsymbol{w}}_{k,L}$ replaced by $\bar{\boldsymbol{w}}'_{k,L}$. Moreover, convergence now occurs at the exponential rate $O(\kappa^L)$.* ■

7.6.4 Interpretation of Results

Examining the bound in (7.129), and comparing it with result (7.84) for the single-agent case, we observe that the topology of the network is now reflected in the bound through the Perron entries, p_k. Recall from (7.118) that the $\{p_k\}$ are the entries of the right-eigenvector of A corresponding to the eigenvalue at one. Moreover, the bound in (7.129)involves three terms (rather than only two as in the single-agent case):

(1) $p_k f_k^2$, which arises from the non-smoothness of the risk function;

(2) $p_k^2 \sigma_k^2$, which is due to gradient noise and the approximation of the true sub-gradient vector;

(3) $2hp_k f_k$, which is an extra term in comparison to the single agent case. We explained in (7.168) that the value of h is related to how far the error at each agent is away from

the weighted average error across the network. Therefore, the term $2hp_kf_k$ is induced by the distributed cooperative strategy.

Additionally, when the risk function happens to be smooth, we have $f_k^2 = 0$. But we cannot set $R = f_k$ any more. Instead, we can set R to be a small number, say $\mu^{1/2}$. Relation (7.171) will then give:

$$\begin{aligned}\mathbb{E}\,\bar{g}(\boldsymbol{\mathcal{W}}_{i-1})^{\mathsf{T}}\bar{\boldsymbol{w}}_{i-1} \leq &\sum_{k=1}^{N} p_k\Big(J_k(w^\star) - \mathbb{E}\,J_k(\boldsymbol{w}_{k,i-1}) - \frac{\eta_k}{2}\mathbb{E}\,\|\widetilde{\boldsymbol{w}}_{k,i-1}\|^2\Big) \\ &+ \frac{h}{2}\sum_{k=1}^{N} p_k\big(\mu^{1/2}e_k^2\mathbb{E}\,\|\widetilde{\boldsymbol{w}}_{k,i-1}\|^2 + \mu^{3/2}\big). \end{aligned} \tag{7.133}$$

Hence, we can recover the tighter result:

$$\lim_{L\to\infty}\ \mathbb{E}\sum_{k=1}^{N} p_k J_k(\bar{\boldsymbol{w}}_{k,L}) - \sum_{k=1}^{N} p_k J_k(w^\star) \leq \frac{\mu}{2}\sum_{k=1}^{N} p_k^2\sigma_k^2 + o(\mu). \tag{7.134}$$

This bound is similar to the one derived in [11] except for a factor of two ($\mu/2$ instead of $\mu/4$) since the derivation in the current article does not require the cost functions to be twice-differentiable.

Example 8 (Multi-agent LASSO problem) We now consider the LASSO problem with 20 agents connected according to Fig. 7.6(left). Each agent has different regression power and noise level, as illustrated in Fig. 7.6(right). The remaining parameters, including $\boldsymbol{w}^\circ, \delta, \mu$ and M are the same as in Example 5. The parameter $h \approx 1.8$ is computed from the simulation directly. Figure. 7.7 compares several strategies including standard diffusion LMS [11,28,169]

$$\begin{cases} \boldsymbol{\psi}_{k,i} = \boldsymbol{w}_{k,i-1} + \mu\boldsymbol{h}_{k,i}(\boldsymbol{\gamma}(k,i) - \boldsymbol{h}_{k,i}^{\mathsf{T}}\boldsymbol{w}_{k,i-1}) \\ \boldsymbol{w}_{k,i} = \displaystyle\sum_{\ell\in\mathcal{N}_k} a_{\ell k}\boldsymbol{\psi}_{\ell,i} \end{cases}, \tag{7.135}$$

and sparse diffusion LMS [164–167]:

$$\begin{cases} \boldsymbol{\psi}_{k,i} = \boldsymbol{w}_{k,i-1} + \mu \boldsymbol{h}_{k,i}(\boldsymbol{\gamma}(k,i) - \boldsymbol{h}_{k,i}^{\mathsf{T}} \boldsymbol{w}_{k,i-1}) - \mu\delta \cdot sgn(\boldsymbol{w}_{k,i-1}) \\ \boldsymbol{w}_{k,i} = \sum_{\ell \in \mathcal{N}_k} a_{\ell k} \boldsymbol{\psi}_{\ell,i} \end{cases}, \tag{7.136}$$

in the cooperative and non-cooperative modes of operation (in the latter case, the combination step (7.116) is absent).

□

Example 9 (Multi-agent SVM learning) We examine the diffusion sub-gradient SVM implementation:

$$\begin{cases} \boldsymbol{\psi}_{k,i} = (1-\rho\mu)\boldsymbol{w}_{k,i-1} - \mu\boldsymbol{\gamma}(k,i)\boldsymbol{h}_i \mathbb{I}[\boldsymbol{\gamma}(k,i)\boldsymbol{h}_{k,i}^{\mathsf{T}} \boldsymbol{w}_{k,i-1} \leq 1] \\ \boldsymbol{w}_{k,i} = \sum_{\ell \in \mathcal{N}_k} a_{\ell k} \boldsymbol{\psi}_{\ell,i} \\ S_{k,i} = \kappa S_{k,i-1} + 1 \\ \bar{\boldsymbol{w}}_{k,i} = \left(1 - \frac{1}{S_{k,i}}\right) \bar{\boldsymbol{w}}_{k,i-1} + \frac{1}{S_{k,i}} \boldsymbol{w}_{k,i} \end{cases}, \tag{7.137}$$

(with all variables initiated at zero) over the Adult dataset again. We distribute 32561 training data over a network consisting of 20 agents. We set $\rho = 0.002$ and $\mu = 0.15$ for all agents. From Example 6 and Theorem 7.2, we know that for the multi-agent SVM problem:

$$\begin{aligned} \alpha_q &= \max_k \left\{1 - \mu\rho + \mu^2(h+1)e_k^2\right\} \\ &= \max_k \left\{1 - \mu\rho + \mu^2(h+1)\left(\rho^2 + \frac{2\rho d_k}{R}\right)\right\}. \end{aligned} \tag{7.138}$$

We set $\kappa = 1 - 0.9 \cdot \mu\rho$, which usually guarantees $\kappa \geq \alpha_q$. Fig. 7.8 (left) shows that cooperation among the agents outperforms the non-cooperative solution. Moreover, the distributed network can almost match the performance of the centralized LIBSVM solution. We also examined the RCV1 dataset. Here we have 20242 training data points and we distribute them over 20 agents. We set the parameters to $\rho = 1 \times 10^{-5}$ and $\mu = 0.5$(for

limited data). We now use $\kappa = 1 - 0.5 \cdot \mu\rho$ since μ is not that small. The result is shown in Fig. 7.8 (right).

□

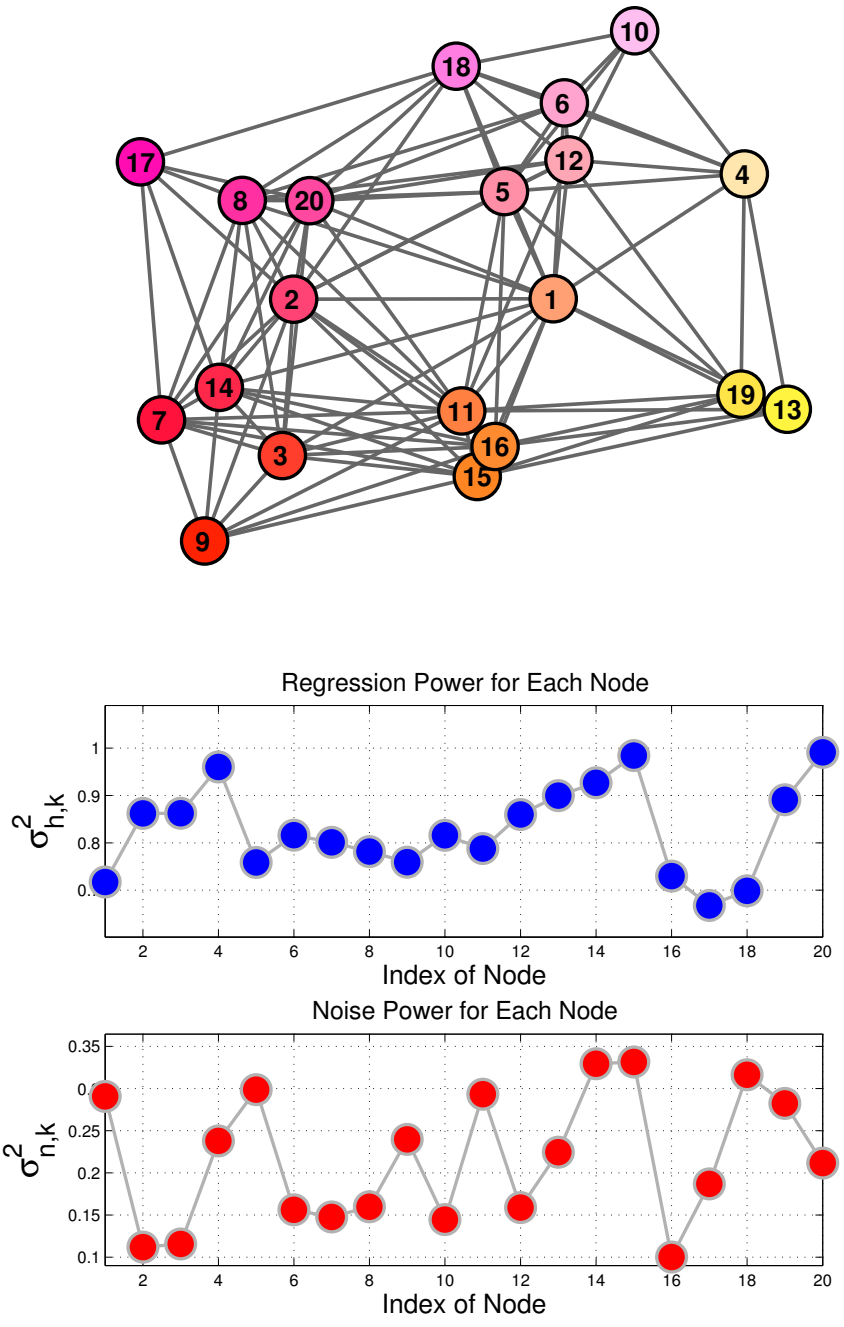

Figure 7.6: Top: Network topology linking $N = 20$ agents. Left: Feature and noise variances across the agents.

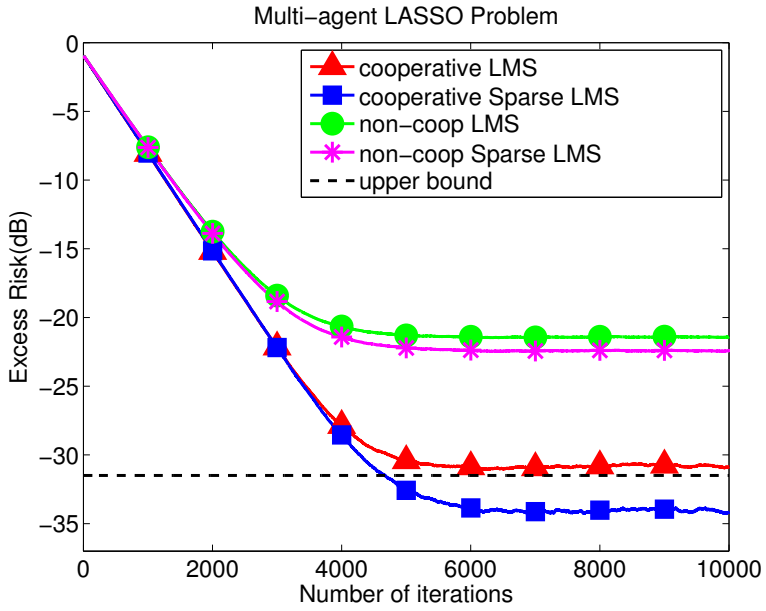

Figure 7.7: The excess-risk curves (Eq. 7.129) for several strategies. Cooperative LMS refers the implementation (7.135); Cooperative LMS refers the implementation (7.136); non-coop sparse LMS refers (7.105); and the upper bound comes from (7.122)

7.7 Conclusion

In summary, we examined the performance of stochastic sub-gradient learning strategies. We proposed a new affine-Lipschitz condition, which is quite suitable for strongly convex but non-differentiable cost functions and is applicable to several important cases including SVM, LASSO, Total-Variation denoising, etc. Under this weaker condition, the analysis establishes that sub-gradient strategies can attain exponential convergence rates, as opposed to sub-linear rates. The analysis also established that these strategies can approach the optimal solution within $O(\mu)$, for sufficiently small step-size. Both single-agent and multi-agent scenarios are studied.

7.A Derivation of the Tighter SVM Bound (7.104)

We assumed in the exposition leading to Theorem 7.1 and Corollary 7.1 that the sub-gradient vectors and the variance of the gradient noise satisfy affine-like forms separately — see (7.22)

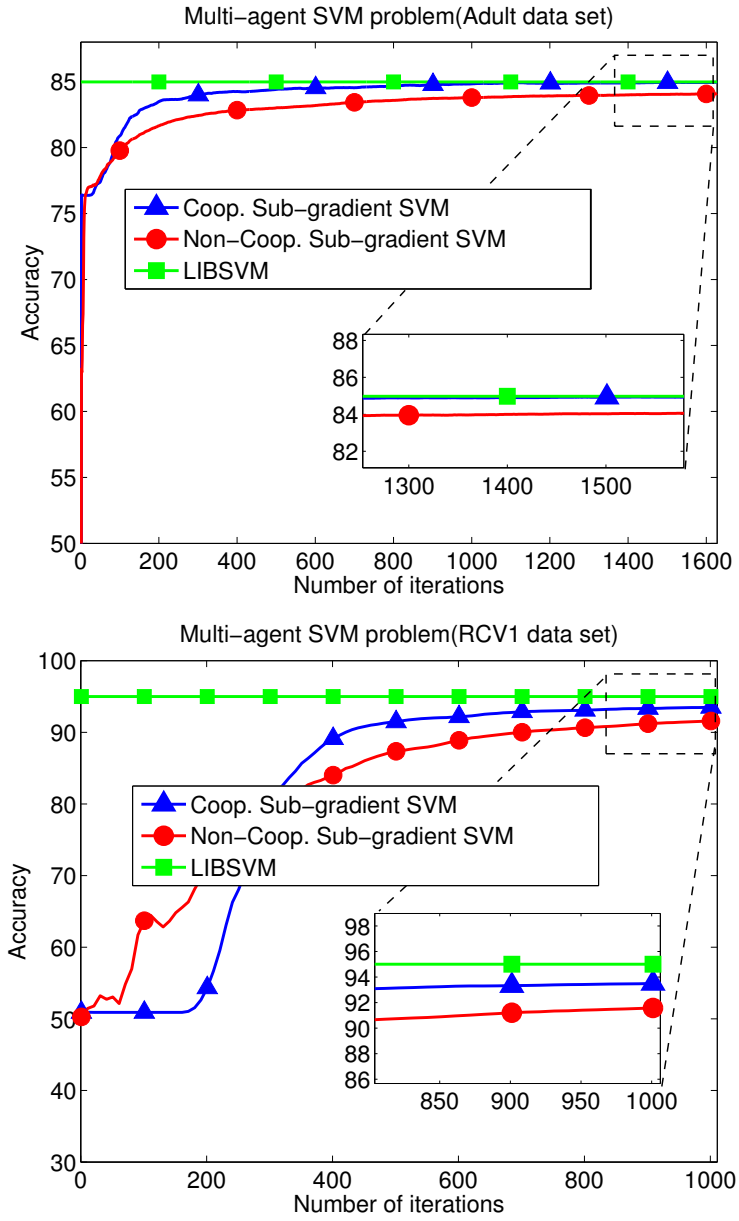

Figure 7.8: Performance of multi-agent SVM solution for the Adult dataset(Top) and RCV1 dataset(Bottom), where vertical axis measures the percentage of correct prediction over test dataset.

and (7.26). For the case of the SVM problem, a joint bound can be derived as follows. First, note that

$$\mathbb{E}\left[\,\|g^{svm}(\boldsymbol{w}_{i-1})+\boldsymbol{s}_i(\boldsymbol{w}_{i-1})\|^2\;|\;\boldsymbol{\mathcal{F}}_{i-1}\,\right]$$

$$
\begin{aligned}
&= \mathbb{E}\,\big[\,\|\widehat{g}^{\,svm}(\boldsymbol{w}_{i-1})\|^2\;|\;\boldsymbol{\mathcal{F}}_{i-1}\,\big] \\
&= \mathbb{E}\,\Big[\,\big\|\rho\boldsymbol{w}_{i-1} + \boldsymbol{\gamma}(i)\boldsymbol{h}_i\,\mathbb{I}[\boldsymbol{\gamma}(i)\boldsymbol{h}_i^{\mathsf{T}}\boldsymbol{w}_{i-1} \le 1]\big\|^2\;|\;\boldsymbol{\mathcal{F}}_{i-1}\,\Big] \\
&= \rho^2\|\boldsymbol{w}_{i-1}\|^2 + 2\rho\mathbb{E}\,\boldsymbol{\gamma}(i)\boldsymbol{h}_i^{\mathsf{T}}\boldsymbol{w}_{i-1}\,\mathbb{I}[\boldsymbol{\gamma}(i)\boldsymbol{h}_i^{\mathsf{T}}\boldsymbol{w}_{i-1} \le 1]\,|\,\boldsymbol{\mathcal{F}}_{i-1}] + \\
&\qquad \mathbb{E}\,\big[\boldsymbol{\gamma}^2(i)\boldsymbol{h}_i\boldsymbol{h}_i^{\mathsf{T}}\,\mathbb{I}[\boldsymbol{\gamma}(i)\boldsymbol{h}_i^{\mathsf{T}}\boldsymbol{w}_{i-1} \le 1]\,|\,\boldsymbol{\mathcal{F}}_{i-1}\big] \\
&\overset{(a)}{\le} \rho^2\|\boldsymbol{w}_{i-1}\|^2 + 2\rho + \mathbb{E}\,[\boldsymbol{h}_i^{\mathsf{T}}\boldsymbol{h}_i\,\mathbb{I}[\boldsymbol{\gamma}(i)\boldsymbol{h}_i^{\mathsf{T}}\boldsymbol{w}_{i-1} \le 1]\,|\,\boldsymbol{\mathcal{F}}_{i-1}] \\
&\le \rho^2\|\boldsymbol{w}_{i-1}\|^2 + 2\rho + \mathrm{Tr}(R_h) \\
&\overset{(b)}{\le} 2\rho^2\|\widetilde{\boldsymbol{w}}_{i-1}\|^2 + 2\rho^2\|w^\star\| + 2\rho + \mathrm{Tr}(R_h),
\end{aligned}
\tag{7.139}
$$

where in step (a) we used the facts that $\boldsymbol{\gamma}^2(i) = 1$ and

$$
\boldsymbol{\gamma}(i)\boldsymbol{h}_i^{\mathsf{T}}\boldsymbol{w}_{i-1}\mathbb{I}[\boldsymbol{\gamma}(i)\boldsymbol{h}_i^{\mathsf{T}}\boldsymbol{w}_{i-1} \le 1]\,|\,\boldsymbol{\mathcal{F}}_{i-1}] \le 1, \tag{7.140}
$$

while step (b) follows from Jensen's inequality by adding and subtracting $w^\star$ to $\boldsymbol{w}_{i-1}$. We therefore conclude that

$$
\mathbb{E}\,[\|g^{\,svm}(\boldsymbol{w}_{i-1}) + \boldsymbol{s}_i(\boldsymbol{w}_{i-1})\|^2\,|\,\boldsymbol{\mathcal{F}}_{i-1}] \le 2\rho^2\|\widetilde{\boldsymbol{w}}_{i-1}\|^2 + 2\rho^2\|w^\star\| + 2\rho + \mathrm{Tr}(R_h). \tag{7.141}
$$

We now use this result to expand the first line of (7.54).

$$
\begin{aligned}
\mathbb{E}\,[\|\widetilde{\boldsymbol{w}}_i\|^2\,|\,\boldsymbol{\mathcal{F}}_{i-1}\,] &= \mathbb{E}\,[\,\|\widetilde{\boldsymbol{w}}_{i-1} + \mu g^{\,svm}(\boldsymbol{w}_{i-1}) + \mu\boldsymbol{s}_i(\boldsymbol{w}_{i-1})\|^2\,|\,\boldsymbol{\mathcal{F}}_{i-1}\,] \\
&= \|\widetilde{\boldsymbol{w}}_{i-1}\|^2 + 2\mu g(\boldsymbol{w}_{i-1})^{\mathsf{T}}\widetilde{\boldsymbol{w}}_{i-1} + \mu^2\mathbb{E}\,[\|g^{\,svm}(\boldsymbol{w}_{i-1}) + \boldsymbol{s}_i(\boldsymbol{w}_{i-1})\|^2\,|\,\boldsymbol{\mathcal{F}}_{i-1}\,] \\
&\le \|\widetilde{\boldsymbol{w}}_{i-1}\|^2 + 2\mu g(\boldsymbol{w}_{i-1})^{\mathsf{T}}\widetilde{\boldsymbol{w}}_{i-1} + \mu^2\big(2\rho^2\|\widetilde{\boldsymbol{w}}_{i-1}\|^2 + 2\rho^2\|w^\star\| + 2\rho + \mathrm{Tr}(R_h)\big).
\end{aligned}
\tag{7.142}
$$

Now following the same steps after (7.55) , we arrive at the tighter bound (7.104).

7.B Proof of theorem 7.2

Introduce the error vector, $\widetilde{\boldsymbol{w}}_{k,i} = w^\star - w_{k,i}$. We collect the iterates and the respective errors from across the network into block column vectors:

$$\boldsymbol{\mathcal{W}}_i \triangleq col\{\boldsymbol{w}_{1,i}, \boldsymbol{w}_{2,i}, \ldots, \boldsymbol{w}_{N,i}\} \tag{7.143}$$

$$\widetilde{\boldsymbol{\mathcal{W}}}_i \triangleq col\{\widetilde{\boldsymbol{w}}_{1,i}, \widetilde{\boldsymbol{w}}_{2,i}, \ldots, \widetilde{\boldsymbol{w}}_{N,i}\}. \tag{7.144}$$

We also define the extended quantities:

$$\mathcal{A} \triangleq A \otimes I_M \tag{7.145}$$

$$\boldsymbol{\mathcal{G}}(\boldsymbol{\mathcal{W}}_{i-1}) \triangleq col\{g_1(\boldsymbol{w}_{1,i-1}), \ldots, g_N(\boldsymbol{w}_{N,i-1})\} \tag{7.146}$$

$$\boldsymbol{\mathcal{s}}_i(\boldsymbol{\mathcal{W}}_{i-1}) \triangleq col\{s_{1,i}(\boldsymbol{w}_{1,i-1}), \ldots, s_{N,i}(\boldsymbol{w}_{N,i-1})\}, \tag{7.147}$$

where $\otimes$ denotes the Kronecker product operation, and $\boldsymbol{s}_{k,i}(\boldsymbol{w}_{k,i-1})$ denotes the gradient noise at agent k. Then, it is straightforward to verify that the network error vector generated by the diffusion strategy evolves according to the following dynamics:

$$\widetilde{\boldsymbol{\mathcal{W}}}_i = \mathcal{A}^{\mathsf{T}} \left(\widetilde{\boldsymbol{\mathcal{W}}}_{i-1} + \mu \boldsymbol{\mathcal{G}}(\boldsymbol{\mathcal{W}}_{i-1}) + \mu \boldsymbol{\mathcal{s}}_i(\boldsymbol{\mathcal{W}}_{i-1})\right). \tag{7.148}$$

Motivated by the treatment of the smooth case in [11, 102, 146], we perform a useful change of variables. Let $\mathcal{V}_\epsilon = V_\epsilon \otimes I_M$ and $\mathcal{J}_\epsilon = J_\epsilon \otimes I_M$. Multiplying (7.148) from the left by $\mathcal{V}_\epsilon^{\mathsf{T}}$ gives

$$\mathcal{V}_\epsilon^{\mathsf{T}} \widetilde{\boldsymbol{\mathcal{W}}}_i = \mathcal{J}_\epsilon^{\mathsf{T}} \left[\mathcal{V}_\epsilon^{\mathsf{T}} \widetilde{\boldsymbol{\mathcal{W}}}_{i-1} + \mu \mathcal{V}_\epsilon^{\mathsf{T}} \boldsymbol{\mathcal{G}}(\boldsymbol{\mathcal{W}}_{i-1}) + \mu \mathcal{V}_\epsilon^{\mathsf{T}} \boldsymbol{\mathcal{s}}_i(\boldsymbol{\mathcal{W}}_{i-1})\right]. \tag{7.149}$$

We introduce the transformed quantities:

$$\mathcal{V}_\epsilon^{\mathsf{T}} \widetilde{\boldsymbol{\mathcal{W}}}_i = \begin{bmatrix} (p^{\mathsf{T}} \otimes I)\widetilde{\boldsymbol{\mathcal{W}}}_i \\ (V_R^{\mathsf{T}} \otimes I)\widetilde{\boldsymbol{\mathcal{W}}}_i \end{bmatrix} \triangleq \begin{bmatrix} \bar{\boldsymbol{w}}_i \\ \check{\boldsymbol{\mathcal{W}}}_i \end{bmatrix}, \tag{7.150}$$

$$\mathcal{V}_\epsilon^{\mathsf{T}} \boldsymbol{\mathcal{G}}(\boldsymbol{\mathcal{W}}_{i-1}) = \begin{bmatrix} (p^{\mathsf{T}} \otimes I)\boldsymbol{\mathcal{G}}(\boldsymbol{\mathcal{W}}_{i-1}) \\ (V_R^{\mathsf{T}} \otimes I)\boldsymbol{\mathcal{G}}(\boldsymbol{\mathcal{W}}_{i-1}) \end{bmatrix} \triangleq \begin{bmatrix} \bar{g}(\boldsymbol{\mathcal{W}}_{i-1}) \\ \check{\boldsymbol{\mathcal{G}}}(\boldsymbol{\mathcal{W}}_{i-1}) \end{bmatrix}, \tag{7.151}$$

$$\mathcal{V}_\epsilon^{\mathsf{T}}\boldsymbol{s}_i(\boldsymbol{\mathcal{W}}_{i-1}) = \left[\begin{array}{c} (p^{\mathsf{T}}\otimes I)\boldsymbol{s}_i(\boldsymbol{\mathcal{W}}_{i-1}) \\ (V_R^{\mathsf{T}}\otimes I)\boldsymbol{s}_i(\boldsymbol{\mathcal{W}}_{i-1}) \end{array}\right] \triangleq \left[\begin{array}{c} \bar{\boldsymbol{s}}_i(\boldsymbol{\mathcal{W}}_{i-1}) \\ \check{\boldsymbol{s}}_i(\boldsymbol{\mathcal{W}}_{i-1}) \end{array}\right]. \tag{7.152}$$

Note that the quantities $\{\bar{\boldsymbol{w}}_i, \bar{g}(\boldsymbol{\mathcal{W}}_{i-1}), \bar{\boldsymbol{s}}_i(\boldsymbol{\mathcal{W}}_{i-1})\}$ amount to the weighted averages:

$$\bar{\boldsymbol{w}}_i = \sum_{k=1}^{N} p_k \widetilde{\boldsymbol{w}}_{k,i}, \tag{7.153}$$

$$\bar{g}(\boldsymbol{\mathcal{W}}_{i-1}) = \sum_{k=1}^{N} p_k g_k(\boldsymbol{w}_{k,i-1}), \tag{7.154}$$

$$\bar{\boldsymbol{s}}_i(\boldsymbol{\mathcal{W}}_{i-1}) = \sum_{k=1}^{N} p_k \boldsymbol{s}_{k,i}(\boldsymbol{w}_{k,i-1}). \tag{7.155}$$

We can now rewrite (7.149) as

$$\left[\begin{array}{c} \bar{\boldsymbol{w}}_i \\ \check{\boldsymbol{\mathcal{W}}}_i \end{array}\right] = \left[\begin{array}{c|c} I_M & 0 \\ \hline 0 & \mathcal{J}_\epsilon^{\mathsf{T}} \end{array}\right] \left(\left[\begin{array}{c} \bar{\boldsymbol{w}}_{i-1} \\ \check{\boldsymbol{\mathcal{W}}}_{i-1} \end{array}\right] + \mu \left[\begin{array}{c} \bar{g}(\boldsymbol{\mathcal{W}}_{i-1}) \\ \check{\boldsymbol{\mathcal{G}}}(\boldsymbol{\mathcal{W}}_{i-1}) \end{array}\right] + \mu \left[\begin{array}{c} \bar{\boldsymbol{s}}_i(\boldsymbol{\mathcal{W}}_{i-1}) \\ \check{\boldsymbol{s}}_i(\boldsymbol{\mathcal{W}}_{i-1}) \end{array}\right] \right). \tag{7.156}$$

Consider the top recursion, namely,

$$\bar{\boldsymbol{w}}_i = \bar{\boldsymbol{w}}_{i-1} + \mu\bar{g}(\boldsymbol{\mathcal{W}}_{i-1}) + \mu\bar{\boldsymbol{s}}_i(\boldsymbol{\mathcal{W}}_{i-1}). \tag{7.157}$$

Squaring and taking expectations we have

$$\begin{aligned} \mathbb{E}\,[\|\bar{\boldsymbol{w}}_i\|^2 \,|\, \boldsymbol{\mathcal{F}}_{i-1}] &= \mathbb{E}\,[\|\bar{\boldsymbol{w}}_{i-1} + \mu\bar{g}(\boldsymbol{\mathcal{W}}_{i-1}) + \mu\bar{\boldsymbol{s}}_i(\boldsymbol{\mathcal{W}}_{i-1})\|^2 \,|\, \boldsymbol{\mathcal{F}}_{i-1}] \\ &= \|\bar{\boldsymbol{w}}_{i-1}\|^2 + 2\mu\bar{g}(\boldsymbol{\mathcal{W}}_{i-1})^{\mathsf{T}}\bar{\boldsymbol{w}}_{i-1} \;+ \mu^2\|\bar{g}(\boldsymbol{\mathcal{W}}_{i-1})\|^2 \;+\; \mu^2\mathbb{E}\,[\|\bar{\boldsymbol{s}}_i(\boldsymbol{\mathcal{W}}_{i-1})\|^2 \,|\, \boldsymbol{\mathcal{F}}_{i-1}]. \end{aligned} \tag{7.158}$$

We examine the terms on the right-hand side one by one. First note that, using Jensen's inequality,

$$\|\bar{g}(\boldsymbol{\mathcal{W}}_{i-1})\|^2 \quad = \quad \left\| \sum_{k=1}^{N} p_k g_k(\boldsymbol{w}_{k,i-1}) \right\|^2$$

$$
\begin{aligned}
&\leq \sum_{k=1}^{N} p_k \|g_k(\boldsymbol{w}_{k,i-1})\|^2 \\
&\overset{(7.40)}{\leq} \sum_{k=1}^{N} p_k \Big(e_k^2 \|\widetilde{\boldsymbol{w}}_{k,i-1}\|^2 + f_k^2 \Big). \qquad (7.159)
\end{aligned}
$$

Next, using the assumed independence of the stochastic gradient noises across the agents, we have

$$
\begin{aligned}
\mathbb{E}\,[\|\bar{\boldsymbol{s}}_i(\boldsymbol{\mathcal{W}}_{i-1})\|^2 \,|\, \boldsymbol{\mathcal{F}}_{i-1}] &= \mathbb{E}\,\left[\left\| \sum_{k=1}^{N} p_k \boldsymbol{s}_k(\boldsymbol{w}_{k,i-1}) \right\|^2 \,|\, \boldsymbol{\mathcal{F}}_{i-1} \right] \\
&= \sum_{k=1}^{N} p_k^2 \mathbb{E}\,[\|\boldsymbol{s}_k(\boldsymbol{w}_{k,i-1})\|^2 \,|\, \boldsymbol{\mathcal{F}}_{i-1}] \\
&\leq \sum_{k=1}^{N} p_k^2 \Big(\beta_k^2 \|\widetilde{\boldsymbol{w}}_{k,i-1}\|^2 + \sigma_k^2 \Big). \qquad (7.160)
\end{aligned}
$$

Finally, with regards to the cross term in (7.158), we adapt an argument from [58] to note that:

$$
\begin{aligned}
\bar{g}(\boldsymbol{\mathcal{W}}_{i-1})^{\mathsf{T}} \bar{\boldsymbol{w}}_{i-1} &= \sum_{k=1}^{N} p_k g_k^{\mathsf{T}}(\boldsymbol{w}_{k,i-1}) \big(\widetilde{\boldsymbol{w}}_{k,i-1} + \bar{\boldsymbol{w}}_{i-1} - \widetilde{\boldsymbol{w}}_{k,i-1} \big) \\
&= \sum_{k=1}^{N} p_k g_k^{\mathsf{T}}(\boldsymbol{w}_{k,i-1}) \widetilde{\boldsymbol{w}}_{k,i-1} + \sum_{k=1}^{N} p_k g_k^{\mathsf{T}}(\boldsymbol{w}_{k,i-1}) \big(\bar{\boldsymbol{w}}_{i-1} - \widetilde{\boldsymbol{w}}_{k,i-1} \big). \qquad (7.161)
\end{aligned}
$$

Using the strong-convexity property, namely,

$$
g_k(\boldsymbol{w}_{k,i-1})^{\mathsf{T}} \widetilde{\boldsymbol{w}}_{k,i-1} \leq J_k(w^\star) - J_k(\boldsymbol{w}_{k,i-1}) - \frac{\eta_k}{2} \|\widetilde{\boldsymbol{w}}_{k,i-1}\|^2, \qquad (7.162)
$$

we get

$$
\begin{aligned}
\bar{g}(\boldsymbol{\mathcal{W}}_{i-1})^{\mathsf{T}} \bar{\boldsymbol{w}}_{i-1} \;\leq\; & \sum_{k=1}^{N} p_k \Big(J_k(w^\star) - J_k(\boldsymbol{w}_{k,i-1}) - \frac{\eta_k}{2} \|\widetilde{\boldsymbol{w}}_{k,i-1}\|^2 \Big) + \\
& \sum_{k=1}^{N} p_k g_k^{\mathsf{T}}(\boldsymbol{w}_{k,i-1}) \big(\bar{\boldsymbol{w}}_{i-1} - \widetilde{\boldsymbol{w}}_{k,i-1} \big) \qquad (7.163)
\end{aligned}
$$

$$\leq \sum_{k=1}^{N} p_k\Big(J_k(w^\star) - J_k(\boldsymbol{w}_{k,i-1}) - \frac{\eta_k}{2}\|\widetilde{\boldsymbol{w}}_{k,i-1}\|^2\Big) + \sum_{k=1}^{N} p_k\|g_k(\boldsymbol{w}_{k,i-1})\|\|\bar{\boldsymbol{w}}_{i-1} - \widetilde{\boldsymbol{w}}_{k,i-1}\|. \tag{7.164}$$

It follows, under expectation, that

$$\mathbb{E}\,\bar{g}(\boldsymbol{\mathcal{W}}_{i-1})^{\mathsf{T}}\bar{\boldsymbol{w}}_{i-1} \leq \sum_{k=1}^{N} p_k\Big(J_k(w^\star) - \mathbb{E}\,J_k(\boldsymbol{w}_{k,i-1}) - \frac{\eta_k}{2}\mathbb{E}\,\|\widetilde{\boldsymbol{w}}_{k,i-1}\|^2\Big) + \sum_{k=1}^{N} p_k\mathbb{E}\,\left(\|g_k(\boldsymbol{w}_{k,i-1})\|\|\bar{\boldsymbol{w}}_{i-1} - \widetilde{\boldsymbol{w}}_{k,i-1}\|\right). \tag{7.165}$$

Now, using the Cauchy-Schwartz inequality, we can bound the last expectation as

$$\mathbb{E}\left(\|g_k(\boldsymbol{w}_{k,i-1})\|\|\bar{\boldsymbol{w}}_{i-1} - \widetilde{\boldsymbol{w}}_{k,i-1}\|\right) \leq \sqrt{\mathbb{E}\,\|g_k(\boldsymbol{w}_{k,i-1})\|^2\mathbb{E}\,\|\bar{\boldsymbol{w}}_{i-1} - \widetilde{\boldsymbol{w}}_{k,i-1}\|^2}. \tag{7.166}$$

After sufficient iterations, it will hold that (see Appendix 7.C for the proof):

$$\mathbb{E}\,\|\bar{\boldsymbol{w}}_{i-1} - \widetilde{\boldsymbol{w}}_{k,i-1}\|^2 = O(\mu^2). \tag{7.167}$$

This means that there exists an I_o large enough and a constant h such that for all $i \geq I_o$:

$$\mathbb{E}\,\|\bar{\boldsymbol{w}}_{i-1} - \widetilde{\boldsymbol{w}}_{k,i-1}\|^2 \leq h^2\mu^2. \tag{7.168}$$

Therefore, we find that

$$\begin{aligned}\mathbb{E}\left(\|g_k(\boldsymbol{w}_{k,i-1})\|\|\bar{\boldsymbol{w}}_{i-1} - \widetilde{\boldsymbol{w}}_{k,i-1}\|\right) &\leq h\mu\,\left(\sqrt{\mathbb{E}\,\|g_k(\boldsymbol{w}_{k,i-1})\|^2}\right)\\ &\leq h\mu\,\left(\sqrt{e_k^2\mathbb{E}\,\|\widetilde{\boldsymbol{w}}_{k,i-1}\|^2 + f_k^2}\right)\\ &\leq \mu\frac{h}{2}\left(\frac{e_k^2\mathbb{E}\,\|\widetilde{\boldsymbol{w}}_{k,i-1}\|^2 + f_k^2}{R} + R\right),\end{aligned} \tag{7.169}$$

where the last inequality follows from using

$$\sqrt{x} \leq \frac{1}{2}\Big(\frac{x}{R} + R\Big), \;\; x \geq 0, \tag{7.170}$$

for any positive R, e.g., $R = f_k$, which allows us to conclude that, as $i \to \infty$:

$$\begin{aligned}\mathbb{E}\,\bar{g}(\boldsymbol{\mathcal{w}}_{i-1})^{\mathsf{T}}\bar{\boldsymbol{w}}_{i-1} \leq & \sum_{k=1}^{N} p_k\Big(J_k(w^\star) - \mathbb{E}\,J_k(\boldsymbol{w}_{k,i-1}) - \frac{\eta_k}{2}\mathbb{E}\,\|\widetilde{\boldsymbol{w}}_{k,i-1}\|^2\Big) \\ & + \mu\frac{h}{2}\sum_{k=1}^{N} p_k\left(\frac{e_k^2}{f_k}\mathbb{E}\,\|\widetilde{\boldsymbol{w}}_{k,i-1}\|^2 + 2f_k\right).\end{aligned} \tag{7.171}$$

Taking expectation of (7.158) over the filtration and substituting (7.159), (7.160), and (7.171), we obtain asymptotically that:

$$\begin{aligned}\mathbb{E}\,\|\bar{\boldsymbol{w}}_i\|^2 \;\; &\leq \;\; \mathbb{E}\,\|\bar{\boldsymbol{w}}_{i-1}\|^2 + 2\mu\sum_{k=1}^{N} p_k\Big(J_k(w^\star) - \mathbb{E}\,J_k(\boldsymbol{w}_{k,i-1})\Big) - \mu\sum_{k=1}^{N} p_k\eta_k\mathbb{E}\,\|\widetilde{\boldsymbol{w}}_{k,i-1}\|^2 + \\ &\quad \mu^2\sum_{k=1}^{N} p_k\Big(e_k^2\mathbb{E}\,\|\widetilde{\boldsymbol{w}}_{k,i-1}\|^2 + f_k^2\Big) + \mu^2\sum_{k=1}^{N} p_k^2\Big(\beta_k^2\mathbb{E}\,\|\widetilde{\boldsymbol{w}}_{k,i-1}\|^2 + \sigma_k^2\Big) + \\ &\quad \mu^2 h\sum_{k=1}^{N} p_k\Big(\frac{e_k^2}{f_k}\mathbb{E}\,\|\boldsymbol{w}_{k,i-1}\|^2 + 2f_k\Big) \\ &\leq \;\; \mathbb{E}\,\|\bar{\boldsymbol{w}}_{i-1}\|^2 + 2\mu\sum_{k=1}^{N} p_k\Big(J_k(w^\star) - \mathbb{E}\,J_k(\boldsymbol{w}_{k,i-1})\Big) - \\ &\quad \sum_{k=1}^{N}(1-\alpha_k)\,p_k\mathbb{E}\,\|\widetilde{\boldsymbol{w}}_{k,i-1}\|^2 + \mu^2\sum_{k=1}^{N}\Big(p_k f_k^2 + p_k^2\sigma_k^2 + 2hp_k f_k\Big),\end{aligned} \tag{7.172}$$

where we defined α_k in the second inequality as follows:

$$1 - \alpha_k \;\triangleq\; \mu\eta_k - \mu^2 e_k^2 - \mu^2 p_k\beta_k^2 - \mu^2 h\frac{e_k^2}{f_k}. \tag{7.173}$$

Let q denote the index of the agent that has the smallest $1 - \alpha_k$:

$$q \;=\; \arg\min_{1\leq k\leq N}\,\{1-\alpha_k\}. \tag{7.174}$$

Then, it holds that

$$\sum_{k=1}^{N}(1-\alpha_k)\,p_k\mathbb{E}\,\|\widetilde{\boldsymbol{w}}_{k,i-1}\|^2 \geq (1-\alpha_q)\sum_{k=1}^{N}p_k\mathbb{E}\,\|\widetilde{\boldsymbol{w}}_{k,i-1}\|^2$$
$$\geq (1-\alpha_q)\mathbb{E}\,\|\bar{\boldsymbol{w}}_{i-1}\|^2, \tag{7.175}$$

where we used Jensen's inequality to deduce that

$$\|\bar{\boldsymbol{w}}_{i-1}\|^2 = \left\|\sum_{k=1}^{N}p_k\widetilde{\boldsymbol{w}}_{k,i-1}\right\|^2 \leq \sum_{k=1}^{N}p_k\|\widetilde{\boldsymbol{w}}_{k,i-1}\|^2. \tag{7.176}$$

It follows from (7.172) that

$$2\mu\Big(\sum_{k=1}^{N}p_k\left(\mathbb{E}\;J_k(\boldsymbol{w}_{k,i-1})-J_k(w^\star)\right)\Big) \leq \;\alpha_q\mathbb{E}\,\|\bar{\boldsymbol{w}}_{i-1}\|^2-\mathbb{E}\,\|\bar{\boldsymbol{w}}_i\|^2+\mu^2\sum_{k=1}^{N}(p_kf_k^2+p_k^2\sigma_k^2+2hp_kf_k). \tag{7.177}$$

This inequality recursion has a form similar to the one we encountered in (7.59) in the single agent case. The argument can now be continued similarly to arrive at the conclusions in the statement of the theorem and that stability is ensured for sufficiently small step-sizes satisfying

$$\mu < \frac{\eta_q}{e_q^2+p_q\beta_q^2+he_q^2/f_q}. \tag{7.178}$$

This bound ensures that $\alpha_q \in (0,1)$.

7.C Proof of (7.167)

We now establish the asymptotic result (7.167). Let

$$\bar{\boldsymbol{\mathcal{W}}}_i = col\{\bar{\boldsymbol{w}}_i,\ldots,\bar{\boldsymbol{w}}_i\}, \tag{7.179}$$

where the vector $\bar{\boldsymbol{w}}_i$ is stacked N times to match the dimension of $\widetilde{\boldsymbol{\mathcal{W}}}_i$. We start from the second relation in the error recursion (7.156):

$$\check{\boldsymbol{\mathcal{W}}}_i = \mathcal{J}_\epsilon^{\mathsf{T}}\Big(\check{\boldsymbol{\mathcal{W}}}_{i-1} + \mu\check{\boldsymbol{\mathcal{G}}}(\boldsymbol{\mathcal{W}}_{i-1}) + \mu\check{\boldsymbol{\mathcal{S}}}_i(\boldsymbol{\mathcal{W}}_{i-1})\Big), \tag{7.180}$$

and first explain how to recover $\widetilde{\boldsymbol{\mathcal{W}}}_i - \bar{\boldsymbol{\mathcal{W}}}_i$ from $\check{\boldsymbol{\mathcal{W}}}_i$. Consider the linear system of equations $\check{\boldsymbol{\mathcal{W}}}_i = \mathcal{V}_R^{\mathsf{T}}\widetilde{\boldsymbol{\mathcal{W}}}_i$, with coefficient matrix $\mathcal{V}_R^{\mathsf{T}}$. Noting from (7.120) that $V_L^{\mathsf{T}}V_R = I$, we conclude from the relationship between the nullspace and range space of a matrix that [170]:

$$\widetilde{\boldsymbol{\mathcal{W}}}_i - \mathcal{V}_L\check{\boldsymbol{\mathcal{W}}}_i \in \mathcal{N}(\mathcal{V}_R^{\mathsf{T}}) \iff \widetilde{\boldsymbol{\mathcal{W}}}_i = \mathcal{V}_L\check{\boldsymbol{\mathcal{W}}}_i + \boldsymbol{\mathcal{Z}}, \tag{7.181}$$

for some $\boldsymbol{\mathcal{Z}} \in \mathcal{N}(\mathcal{V}_R^{\mathsf{T}})$. We further know that $\mathcal{N}(V_R^{\mathsf{T}})$ is the linear space generated by the vector $\mathbb{1}$, so that $\boldsymbol{\mathcal{Z}}$ is parallel to the vector $\mathbb{1}$. Therefore, we can write $\boldsymbol{\mathcal{Z}} = \mathbb{1} \otimes z$, for some vector z. Moreover, it holds that

$$\begin{aligned}
\bar{\boldsymbol{w}}_i &= (p^{\mathsf{T}} \otimes I)\widetilde{\boldsymbol{\mathcal{W}}}_i \\
&= (p^{\mathsf{T}} \otimes I)\mathcal{V}_L\check{\boldsymbol{\mathcal{W}}}_i + (p^{\mathsf{T}} \otimes I)\boldsymbol{\mathcal{Z}} \\
&\overset{(a)}{=} (p^{\mathsf{T}} \otimes I)(\mathbb{1} \otimes z) \\
&\overset{(b)}{=} z,
\end{aligned} \tag{7.182}$$

where step (a) is because $(p^{\mathsf{T}} \otimes I)\mathcal{V}_L = (p^{\mathsf{T}}V_L) \otimes I = 0$, and step (b) is because of (7.119). Hence, combining the above two results, we find that

$$\widetilde{\boldsymbol{\mathcal{W}}}_i - \bar{\boldsymbol{\mathcal{W}}}_i = \mathcal{V}_L\check{\boldsymbol{\mathcal{W}}}_i, \tag{7.183}$$

which shows how we can recover $\widetilde{\boldsymbol{\mathcal{W}}}_i - \bar{\boldsymbol{\mathcal{W}}}_i$ from $\check{\boldsymbol{\mathcal{W}}}_i$.

Now returning to the error recursion (7.180), and computing the expected squared norm, we obtain:

$$\mathbb{E}\,[\|\check{\boldsymbol{\mathcal{W}}}_i\|^2 \,|\, \boldsymbol{\mathcal{F}}_{i-1}] = \Big\|\mathcal{J}_\epsilon^{\mathsf{T}}\Big(\check{\boldsymbol{\mathcal{W}}}_{i-1} + \mu\check{\boldsymbol{\mathcal{G}}}(\boldsymbol{\mathcal{W}}_{i-1})\Big)\Big\|^2 + \mu^2\mathbb{E}\,[\|\mathcal{J}_\epsilon^{\mathsf{T}}\check{\boldsymbol{\mathcal{S}}}_i(\boldsymbol{\mathcal{W}}_{i-1})\|^2 \,|\, \boldsymbol{\mathcal{F}}_{i-1}]$$

$$\leq \quad \rho(\mathcal{J}_\epsilon\mathcal{J}_\epsilon^{\mathsf{T}})\|\check{\boldsymbol{\mathcal{W}}}_{i-1}+\mu\check{\boldsymbol{\mathcal{G}}}(\boldsymbol{\mathcal{W}}_{i-1})\|^2+\mu^2\rho(\mathcal{J}_\epsilon\mathcal{J}_\epsilon^{\mathsf{T}})\mathbb{E}\,[\|\check{\boldsymbol{\mathcal{S}}}_i(\boldsymbol{\mathcal{W}}_{i-1})\|^2\,|\,\boldsymbol{\mathcal{F}}_{i-1}], \tag{7.184}$$

where, from [11, Ch. 9], we know that

$$\rho(\mathcal{J}_\epsilon\mathcal{J}_\epsilon^{\mathsf{T}})\leq(\rho(J_\epsilon)+\epsilon)^2<1. \tag{7.185}$$

Let us examine the terms in (7.184). To begin with, note that

$$\begin{aligned}
\rho(\mathcal{J}_\epsilon\mathcal{J}_\epsilon^{\mathsf{T}})\|\check{\boldsymbol{\mathcal{W}}}_{i-1}+\mu\check{\boldsymbol{\mathcal{G}}}(\boldsymbol{\mathcal{W}}_{i-1})\|^2 &\leq (\rho(J_\epsilon)+\epsilon)^2\left\|t\frac{1}{t}\check{\boldsymbol{\mathcal{W}}}_{i-1}+\frac{1-t}{1-t}\mu\check{\boldsymbol{\mathcal{G}}}(\boldsymbol{\mathcal{W}}_{i-1})\right\|^2\\
&\overset{(a)}{\leq} \frac{(\rho(J_\epsilon)+\epsilon)^2}{t}\|\check{\boldsymbol{\mathcal{W}}}_{i-1}\|^2+\mu^2\frac{(\rho(J_\epsilon)+\epsilon)^2}{1-t}\|\check{\boldsymbol{\mathcal{G}}}(\boldsymbol{\mathcal{W}}_{i-1})\|^2\\
&\overset{(b)}{\leq} (\rho(J_\epsilon)+\epsilon)\|\check{\boldsymbol{\mathcal{W}}}_{i-1}\|^2+\mu^2\frac{(\rho(J_\epsilon)+\epsilon)^2}{1-\rho(J_\epsilon)-\epsilon}\|\check{\boldsymbol{\mathcal{G}}}(\boldsymbol{\mathcal{W}}_{i-1})\|^2,
\end{aligned} \tag{7.186}$$

where step (a) is because of Jensen's inequality and in step (b) we set $t=\rho(J_\epsilon)+\epsilon<1$. Next, we bound the square of the sub-gradient term:

$$\begin{aligned}
\|\check{\boldsymbol{\mathcal{G}}}(\boldsymbol{\mathcal{W}}_{i-1})\|^2 &= \|\mathcal{V}_R^{\mathsf{T}}\boldsymbol{\mathcal{G}}(\boldsymbol{\mathcal{W}}_{i-1})\|^2\\
&\leq \|V_R\|^2\Big(\sum_{k=1}^N\|g_k(\boldsymbol{w}_{k,i-1})\|^2\Big)\\
&\leq \|V_R\|^2\Big(\sum_{k=1}^N e_k^2\|\widetilde{\boldsymbol{w}}_{k,i-1}\|^2+f_k^2\Big)\\
&\overset{(a)}{\leq} \|V_R\|^2\Big(e_{\max}^2\|\widetilde{\boldsymbol{\mathcal{W}}}_{i-1}\|^2+\sum_{k=1}^N f_k^2\Big),
\end{aligned} \tag{7.187}$$

where in step (a) we let $e_{\max}^2=\max_k e_k^2$. We can then bound (7.186) by

$$\begin{aligned}
\rho(\mathcal{J}_\epsilon\mathcal{J}_\epsilon^{\mathsf{T}})\|\check{\boldsymbol{\mathcal{W}}}_{i-1}+\mu\check{\boldsymbol{\mathcal{G}}}(\boldsymbol{\mathcal{W}}_{i-1})\|^2 &\leq (\rho(J_\epsilon)+\epsilon)\|\check{\boldsymbol{\mathcal{W}}}_{i-1}\|^2+\mu^2\frac{(\rho(J_\epsilon)+\epsilon)^2}{1-\rho(J_\epsilon)-\epsilon}\|V_R\|^2\sum_{k=1}^N f_k^2\\
&\quad+\mu^2\frac{(\rho(J_\epsilon)+\epsilon)^2}{1-\rho(J_\epsilon)-\epsilon}\|V_R\|^2e_{\max}^2\|\widetilde{\boldsymbol{\mathcal{W}}}_{i-1}\|^2.
\end{aligned} \tag{7.188}$$

Finally, we consider the last term involving the gradient noise in (7.184):

$$\begin{aligned}\mathbb{E}\,[\|\check{\boldsymbol{s}}_i(\boldsymbol{\mathcal{W}}_{i-1})\|^2\,|\,\boldsymbol{\mathcal{F}}_{i-1}] &\leq \|V_R\|^2\left(\sum_{k=1}^{N}\beta_k^2\|\widetilde{\boldsymbol{w}}_{k,i-1}\|^2+\sigma_k^2\right)\\ &\leq \|V_R\|^2\beta_{\max}\|\widetilde{\boldsymbol{\mathcal{W}}}_{i-1}\|^2+\|V_R\|^2\sum_{k=1}^{N}\sigma_k^2. \end{aligned}\tag{7.189}$$

Now introduce the constants:

$$a \triangleq \frac{(\rho(J_\epsilon)+\epsilon)^2}{1-\rho(J_\epsilon)-\epsilon}\|V_R\|^2e_{\max}^2+\rho(J_\epsilon J_\epsilon^{\mathsf{T}})\|V_R\|^2\beta_{\max},\tag{7.190}$$

$$b \triangleq \frac{(\rho(J_\epsilon)+\epsilon)^2}{1-\rho(J_\epsilon)-\epsilon}\|V_R\|^2\sum_{k=1}^{N}f_k^2+\rho(J_\epsilon J_\epsilon^{\mathsf{T}})\|V_R\|^2\sum_{k=1}^{N}\sigma_k^2.\tag{7.191}$$

Then, substituting the previous results into (7.184), we arrive at

$$\mathbb{E}\,\|\check{\boldsymbol{\mathcal{W}}}_i\|^2\leq(\rho(J_\epsilon)+\epsilon)\mathbb{E}\,\|\check{\boldsymbol{\mathcal{W}}}_{i-1}\|^2+\mu^2a\mathbb{E}\,\|\widetilde{\boldsymbol{\mathcal{W}}}_{i-1}\|^2+\mu^2b,\tag{7.192}$$

In Appendix 7.D we show that $\mathbb{E}\,\|\widetilde{\boldsymbol{\mathcal{W}}}_{i-1}\|^2$, for any iteration i, is bounded by a constant value for sufficient small step-sizes. In this case, we can conclude that

$$\mathbb{E}\,\|\check{\boldsymbol{\mathcal{W}}}_i\|^2\leq(\rho(J_\epsilon)+\epsilon)\mathbb{E}\,\|\check{\boldsymbol{\mathcal{W}}}_{i-1}\|^2+\mu^2b',\tag{7.193}$$

for some constant b', so that at steady state:

$$\limsup_{i\to\infty}\mathbb{E}\,\|\check{\boldsymbol{\mathcal{W}}}_i\|^2\leq\frac{\mu^2b'}{1-\rho(\mathcal{J}_\epsilon)-\epsilon}=O(\mu^2).\tag{7.194}$$

Using relation (7.183), it then follows asymptotically that

$$\mathbb{E}\,\|\widetilde{\boldsymbol{\mathcal{W}}}_i-\bar{\boldsymbol{\mathcal{W}}}_i\|^2 \leq \|V_L\|^2\cdot\mathbb{E}\,\|\check{\boldsymbol{\mathcal{W}}}_i\|^2=O(\mu^2),\tag{7.195}$$

and, consequently,

$$\mathbb{E}\,\|\widetilde{\boldsymbol{w}}_{k,i}-\bar{\boldsymbol{w}}_i\|^2 \;\;\leq\;\; \mathbb{E}\,\|\widetilde{\boldsymbol{\mathcal{W}}}_i-\bar{\boldsymbol{\mathcal{W}}}_i\|^2 = O(\mu^2). \tag{7.196}$$

7.D Proof that $\mathbb{E}\,\|\widetilde{\boldsymbol{\mathcal{W}}}_i\|^2$ is uniformly bounded

We follow mathematical induction to establish that $\mathbb{E}\,\|\widetilde{\boldsymbol{\mathcal{W}}}_i\|^2$ is uniformly bounded by a constant value, for all i. Assume, at the initial time instant we have $\mathbb{E}\,\|\widetilde{\boldsymbol{w}}_{k,-1}\|^2 < \mathsf{c}$ for all k and for some constant value c, such that

$$c \geq \eta_k(J_k(w^\star)-J_k(w_k^\star))/2 \quad ,\forall k \tag{7.197}$$

Then, assuming this bound holds at iteration $i-1$, namely,

$$\mathbb{E}\,\|\widetilde{\boldsymbol{w}}_{k,i-1}\|^2 \leq \mathsf{c} \quad ,\forall k \tag{7.198}$$

where $w_k^\star$ is the minimizer of $J_k(w)$. We would like to show that it also holds at iteration i. Recall from (7.116) that the diffusion strategy consists of two steps: an adaptation step followed by a combination step. The adaptation step has a similar structure to the single-agent case. Hence, the same derivation that was used to establish (7.59) would show that for agent k:

$$2\mu\,(\mathbb{E}\;J_k(\boldsymbol{w}_{k,i-1})-J_k(w^\star)) \leq\; \alpha_k\mathbb{E}\,\|\widetilde{\boldsymbol{w}}_{k,i-1}\|^2 - \mathbb{E}\,\|\widetilde{\boldsymbol{\psi}}_{k,i}\|^2 + \mu^2(f_k^2+\sigma_k^2), \tag{7.199}$$

where

$$\alpha_k = 1-\mu\eta_k+\mu^2(e_k^2+\beta_k^2) = 1-O(\mu). \tag{7.200}$$

Now, since $\mathbb{E}\;J_k(\boldsymbol{w}_{k,i-1}) \geq J_k(w_k^\star)$, we conclude that

$$\begin{aligned}\mathbb{E}\,\|\widetilde{\boldsymbol{\psi}}_{k,i}\|^2 \quad &\leq \quad \alpha_k\mathbb{E}\,\|\widetilde{\boldsymbol{w}}_{k,i-1}\|^2+\mu^2(f_k^2+\sigma_k^2)+\mu(J_k(w^\star)-J_k(w_k^\star)) \\ &\overset{(7.198)}{\leq} \quad \alpha_k'\mathsf{c}+\mu^2(f_k^2+\sigma_k^2),\end{aligned} \tag{7.201}$$

where

$$\alpha_k' = 1 - \mu\eta_k/2 + \mu^2(e_k^2 + \beta_k^2) = 1 - O(\mu). \tag{7.202}$$

and the step-size can be chosen small enough to ensure $\alpha_k' \in (0,1)$. Now, it is also clear that there exist sufficiently small values for μ to ensure that, for all agents k:

$$\alpha_k'\mathsf{c} + \mu^2(f_k^2 + \sigma_k^2) \leq \mathsf{c}, \tag{7.203}$$

which then guarantees that

$$\mathbb{E}\,\|\widetilde{\boldsymbol{\psi}}_{k,i}\|^2 \leq \mathsf{c}. \tag{7.204}$$

It then follows from the combination step (7.116) that

$$\begin{aligned}
\mathbb{E}\,\|\widetilde{\boldsymbol{w}}_{k,i}\|^2 &= \mathbb{E}\,\left\|\sum_{\ell\in\mathcal{N}_k} a_{\ell k}\widetilde{\boldsymbol{\psi}}_{\ell,i}\right\|^2 \\
&\leq \sum_{\ell\in\mathcal{N}_k} a_{\ell k}\mathbb{E}\,\left\|\widetilde{\boldsymbol{\psi}}_{\ell,i}\right\|^2 \\
&\leq \sum_{\ell\in\mathcal{N}_k} a_{k\ell}\mathsf{c} \\
&= \mathsf{c}, \quad \forall k.
\end{aligned} \tag{7.205}$$

Therefore, starting from (7.198), we conclude that $\mathbb{E}\,\|\widetilde{\boldsymbol{w}}_{k,i}\|^2 < \mathsf{c}$ as well, as desired. Finally, since $\mathbb{E}\,\|\widetilde{\boldsymbol{\mathcal{W}}}_i\|^2 = \sum_{k=1}^N \mathbb{E}\,\|\widetilde{\boldsymbol{w}}_{k,i}\|^2$, we conclude that $\mathbb{E}\,\|\widetilde{\boldsymbol{\mathcal{W}}}_i\|^2$ is also uniformly bounded over time.

CHAPTER 8

Issues for Future Consideration

In this dissertation, we addressed several important aspects pertaining to the implementation of distributed learning algorithms under random reshuffling. We examined the performance of random reshuffling for both vanilla stochastic gradient and variance-reduction cases. We showed that for vanilla stochastic gradient case, random reshuffling setting improves the convergence region from $O(\mu)$ to $O(\mu^2)$; while variance-reduced algorithms converge to the exact solution at a linear rate. We further examined learning under large feature spaces, where the feature information is assumed to be spread across agents in a network. We proposed two solution methods: one operates in the dual domain and another operates in the primal domain. The dual domain solution builds on gradient boosting techniques while the primal domain solution is achieved by combining a dynamic diffusion construction, a pipeline strategy, and variance-reduced techniques. The primal solution does not require separate timescales and convergence towards the exact minimizer occurs at a linear rate. There are several open issues that deserve further investigation:

1. Establish that random reshuffling techniques can lead to similar performance improvement for sub-gradient implementations with constant step-size when the loss functions are not necessarily smooth. Such improvements have been observed in simulation but the analysis is still lacking.

2. Establish analytically that variance-reduced algorithms under random reshuffling are indeed faster than under uniform sampling. Simulations suggest that this appears to be the case.

3. Show that the PVRD^2 algorithm still converges linearly under random reshuffling. The

PVRD^2 algorithm proposed in this dissertation requires all agents to select the same random indices. The difficulty can be addressed if we appeal to random reshuffling.

4. Design an asynchronous PVRD^2 algorithm. The limitation that all agents select the same random indices can be completely avoided in the asynchronous setting. Moreover, this possibility has great potential for the distributed feature problem.

REFERENCES

[1] D. Needell, R. Ward, and N. Srebro, "Stochastic gradient descent, weighted sampling, and the randomized kaczmarz algorithm," in *Proc. Advances in Neural Information Processing Systems (NIPS)*, Montreal, Canada, 2014, pp. 1017–1025.

[2] P. Zhao and T. Zhang, "Stochastic optimization with importance sampling for regularized loss minimization," in *Proc. International Conference on Machine Learning (ICML)*, Lille, France, 2015, pp. 1355–1363.

[3] M. Gürbüzbalaban, A. Ozdaglar, and P. Parrilo, "Convergence rate of incremental gradient and newton methods," *arXiv:1510.08562*, Oct. 2015.

[4] N. D. Vanli, M. Gurbuzbalaban, and A. Ozdaglar, "Global convergence rate of proximal incremental aggregated gradient methods," *SIAM Journal on Optimization*, vol. 28, no. 2, pp. 1282–1300, 2018.

[5] S. Sundhar, A. Nedić, and V. V. Veeravalli, "A new class of distributed optimization algorithms: Application to regression of distributed data," *Optimization Methods and Software*, vol. 27, no. 1, pp. 71–88, 2012.

[6] C.-C. Chang and C.-J. Lin, "LIBSVM: A library for support vector machines," *ACM Trans. on Intelligent Systems and Tech.*, vol. 2, pp. 27:1–27:27, 2011.

[7] A. Beck and M. Teboulle, "Fast gradient-based algorithms for constrained total variation image denoising and deblurring problems," *IEEE Trans. Image Process.*, vol. 18, no. 11, pp. 2419–2434, 2009.

[8] V. Vapnik, *Statistical Learning Theory*, Wiley NY, 1998.

[9] T. Hastie, R. Tibshirani, and J. Friedman, *The Elements of Statistical Learning*, Springer, 2009.

[10] C. M. Bishop, *Pattern Recognition and Machine Learning*, Springer, 2006.

[11] A. H. Sayed, "Adaptation, learning, and optimization over networks," *Foundations and Trends in Machine Learning*, vol. 7, no. 4-5, pp. 311–801, 2014.

[12] A. Nedić and D. P. Bertsekas, "Convergence rate of incremental subgradient algorithms," in *Stochastic Optimization: Algorithms and Applications*, pp. 223–264. Springer, 2001.

[13] K. Yuan, B. Ying, S. Vlaski, and A. H. Sayed, "Stochastic gradient descent with finite samples sizes," in *Proc. IEEE International Workshop on Machine Learning for Signal Processing*. IEEE, 2016, pp. 1–6.

[14] H. Robbins and S. Monro, "A stochastic approximation method," *The Annals of Mathematical Statistics*, pp. 400–407, 1951.

[15] D. P. Bertsekas and J. N. Tsitsiklis, *Parallel and Distributed Computation: Numerical Methods*, Prentice Hall, NJ, 1989.

[16] B. T. Polyak and A. B. Juditsky, "Acceleration of stochastic approximation by averaging," *SIAM Journal on Control and Optimization*, vol. 30, no. 4, pp. 838–855, 1992.

[17] B. T. Polyak, *Introduction to Optimization*, Optimization Software New York, 1987.

[18] L. Bottou, "Large-scale machine learning with stochastic gradient descent," in *Proc. of COMPSTAT*, pp. 177–186. Springer, 2010.

[19] O. Bousquet and L. Bottou, "The tradeoffs of large scale learning," in *Advances in Neural Information Processing Systems(NIPS), 20*, pp. 161–168. 2008.

[20] E. Moulines and F. Bach, "Non-asymptotic analysis of stochastic approximation algorithms for machine learning," in *Advances in Neural Information Processing Systems(NIPS), 24*, pp. 451–459. Curran Associates, Inc., 2011.

[21] T. Zhang, "Solving large scale linear prediction problems using stochastic gradient descent algorithms," in *Proc. International Conference on Machine Learning (ICML)*, Canada, 2004, pp. 116–124.

[22] A. Nemirovski, A. Juditsky, G. Lan, and A. Shapiro, "Robust stochastic approximation approach to stochastic programming," *SIAM J. Optm.*, vol. 19, no. 4, pp. 1574–1609, 2009.

[23] S. S. Haykin, *Adaptive Filter Theory*, Fourth Edition, Prentice-Hall, NJ, 2008.

[24] A. H. Sayed, *Adaptive Filters*, John Wiley & Sons, 2008.

[25] M. V. Solodov, "Incremental gradient algorithms with stepsizes bounded away from zero," *Computational Optimization and Applications*, vol. 11, no. 1, pp. 23–35, 1998.

[26] Y. Koren, "Factorization meets the neighborhood: a multifaceted collaborative filtering model," in *Proc. International Conference on Knowledge Discovery and Data Mining (SIGKDD)*, Las Vegas, USA, 2008, pp. 426–434.

[27] I. Sutskever, J. Martens, G. Dahl, and G. Hinton, "On the importance of initialization and momentum in deep learning," in *Proc. International Conference on Machine Learning (ICML)*, Atlanta, USA, 2013, pp. 1139–1147.

[28] A. H. Sayed, "Adaptive networks," *Proceedings of the IEEE*, vol. 102, no. 4, pp. 460–497, 2014.

[29] M. Rupp and A. H. Sayed, "A time-domain feedback analysis of filtered-error adaptive gradient algorithms," *IEEE Transactions on Signal Processing*, vol. 44, no. 6, pp. 1428–1439, 1996.

[30] A. H. Sayed and M. Rupp, "An ℓ_2 stable feedback structure for nonlinear adaptive filtering and identification," *Automatica*, vol. 33, no. 1, pp. 13–30, Jan. 1997.

[31] B. Ying, K. Yuan, and A. H. Sayed, "Variance-reduced stochastic learning under random reshuffling," *submitted for publication* and available as *arXiv:1708.01383*, Aug. 2017.

[32] D. P. Bertsekas, A. Nedi, and A. E Ozdaglar, *Convex Analysis and Optimization*, Athena Scientific, 2003.

[33] D. G. Horvitz and D. J. Thompson, "A generalization of sampling without replacement from a finite universe," *Journal of the American statistical Association*, vol. 47, no. 260, pp. 663–685, 1952.

[34] B. Recht and C. Ré, "Toward a noncommutative arithmetic-geometric mean inequality: Conjectures, case-studies, and consequences," in *Proc. Conference On Learning Theory (COLT)*, 2012, pp. 1–11.

[35] M. Gürbüzbalaban, A. Ozdaglar, and P. Parrilo, "Why random reshuffling beats stochastic gradient descent," *arXiv:1510.08560*, Oct. 2015.

[36] O. Shamir, "Without-replacement sampling for stochastic gradient methods: Convergence results and application to distributed optimization," *arXiv:1603.00570*, Mar. 2016.

[37] A. Agarwal, M. J. Wainwright, P. L. Bartlett, and P. K. Ravikumar, "Information-theoretic lower bounds on the oracle complexity of convex optimization," in *Proc. Advances in Neural Information Processing Systems (NIPS)*, Vancouver, Canada, 2009, pp. 1–9.

[38] Rie Johnson and Tong Zhang, "Accelerating stochastic gradient descent using predictive variance reduction," in *Proc. Advances in Neural Information Processing Systems (NIPS)*, Lake Tahoe, Navada, 2013, pp. 315–323.

[39] A. Defazio, F. Bach, and S. Lacoste-Julien, "SAGA: A fast incremental gradient method with support for non-strongly convex composite objectives," in *Proc. Advances in Neural Information Processing Systems (NIPS)*, Montreal, Canada, 2014, pp. 1646–1654.

[40] A. Defazio and J. Domke, "Finito: A faster, permutable incremental gradient method for big data problems," in *International Conference on Machine Learning*, 2014, pp. 1125–1133.

[41] Shai Shalev-Shwartz and Tong Zhang, "Stochastic dual coordinate ascent methods for regularized loss," *The Journal of Machine Learning Research*, vol. 14, no. 1, pp. 567–599, 2013.

[42] Nicolas L Roux, Mark Schmidt, and Francis R Bach, "A stochastic gradient method with an exponential convergence rate for finite training sets," in *Proc. Advances in Neural Information Processing Systems (NIPS)*, Lake Tahoe, Navada, 2012, pp. 2663–2671.

[43] S. De and T. Goldstein, "Efficient distributed SGD with variance reduction," in *IEEE 16th International Conference on Data Mining (ICDM)*, 2016, pp. 111–120.

[44] D. P. Bertsekas, *Nonlinear Programming*, Athena scientific Belmont, 1999.

[45] L. Bottou and O. Bousquet, "The tradeoffs of large scale learning," in *Proc. Advances in Neural Information Processing Systems (NIPS)*, 2008, pp. 161–168.

[46] S. Boyd and L. Vandenberghe, *Convex Optimization*, Cambridge University Press, 2004.

[47] M. Rabbat and R. Nowak, "Distributed optimization in sensor networks," in *Proceedings of the 3rd international symposium on Information processing in sensor networks*. ACM, 2004, pp. 20–27.

[48] S. Kar and J. M. F. Moura, "Distributed consensus algorithms in sensor networks with imperfect communication: Link failures and channel noise," *IEEE Trans. Signal Process.*, vol. 57, no. 1, pp. 355–369, 2009.

[49] S. Sun, "A survey of multi-view machine learning," *Neural Computing and Applications*, vol. 23, no. 7-8, pp. 2031–2038, 2013.

[50] C. Xu, D. Tao, and C. Xu, "A survey on multi-view learning," *available at arXiv:1304.5634*, 2013.

[51] K. Bimpikis, S. Ehsani, and R. Ilkilic, "Cournot competition in networked markets," in *Proc. of ACM Conf. on Economics and Computation*, Stanford, CA,, Jun. 2014, p. 733.

[52] C. Metzler, B. F. Hobbs, and J.-S. Pang, "Nash-cournot equilibria in power markets on a linearized dc network with arbitrage: Formulations and properties," *Networks and Spatial Economics*, vol. 3, no. 2, pp. 123–150, 2003.

[53] Chung-Kai Yu, Mihaela van der Schaar, and Ali H Sayed, "Distributed learning for stochastic generalized nash equilibrium problems," *IEEE Transactions on Signal Processing*, vol. 65, no. 15, pp. 3893–3908, 2017.

[54] J. Chen, Z. J. Towfic, and A. H. Sayed, "Dictionary learning over distributed models," *IEEE Transactions on Signal Processing*, vol. 63, no. 4, pp. 1001–1016, 2015.

[55] J. Duchi, E. Hazan, and Y. Singer, "Adaptive subgradient methods for online learning and stochastic optimization," *Journal of Machine Learning Research*, vol. 12, no. Jul, pp. 2121–2159, 2011.

[56] A. Mokhtari and A. Ribeiro, "DSA: decentralized double stochastic averaging gradient algorithm," *Journal of Machine Learning Research*, vol. 17, no. 61, pp. 1–35, Mar. 2016.

[57] K. Yuan, B. Ying, X. Zhao, and A. H. Sayed, "Exact dffusion for distributed optimization and learning – Part I: Algorithm development," *Submitted for publication, Also available on arXiv*, Feb. 2017.

[58] A. Nedic and A. Ozdaglar, "Distributed subgradient methods for multi-agent optimization," *IEEE Trans. Autom. Control*, vol. 54, no. 1, pp. 48–61, 2009.

[59] J. Chen and A. H. Sayed, "Distributed pareto optimization via diffusion strategies," *IEEE Journal of Selected Topics in Signal Processing*, vol. 7, no. 2, pp. 205–220, 2013.

[60] S. Shalev-Shwartz, Y. Singer, N. Srebro, and A. Cotter, "Pegasos: Primal estimated sub-gradient solver for SVM," *Mathematical Programming*, vol. 127, no. 1, pp. 3–30, 2011.

[61] A. Cotter, O. Shamir, N. Srebro, and K. Sridharan, "Better mini-batch algorithms via accelerated gradient methods," in *Proc. Advances in Neural Information Processing Systems (NIPS)*, Granada, Spain, 2011, pp. 1647–1655.

[62] L. Bottou, "Curiously fast convergence of some stochastic gradient descent algorithms," in *Proc. Symposium on Learning and Data Science*, Paris, 2009.

[63] T. Zhang, "A note on the non-commutative arithmetic-geometric mean inequality," *Available at arXiv:1411.5058*, 2014.

[64] R. Johnson and T. Zhang, "Accelerating stochastic gradient descent using predictive variance reduction," in *Advances in Neural Information Processing Systems (NIPS)*, 2013, pp. 315–323.

[65] K. Lee, M. Lam, R. Pedarsani, D. Papailiopoulos, and K. Ramchandran, "Speeding up distributed machine learning using codes," *IEEE Transactions on Information Theory*, pp. 1514 – 1529, August 2017.

[66] D. P. Bertsekas, "A new class of incremental gradient methods for least squares problems," *SIAM Journal on Optimization*, vol. 7, no. 4, pp. 913–926, 1997.

[67] S. Theodoridis and K. Koutroumbas, *Pattern Recognition*, Academic Press, 4th edition, 2008.

[68] Y. Nesterov, *Introductory Lectures on Convex Optimization: A basic course*, vol. 87, Springer, 2013.

[69] B. Ying, K. Yuan, S. Vlaski, and A. H. Sayed, "On the performance of random reshuffling in stochastic learning," in *Proc. Information Theory and Applications Workshop (ITA)*, San Diego, CA, Feb. 2017, pp. 1–5.

[70] R. Durrett, *Probability: Theory and Examples*, Cambridge University Press, 2010.

[71] R. A. Horn and C. R. Johnson, *Matrix Analysis*, Cambridge University Press, 2003.

[72] M. Gürbüzbalaban, A. Ozdaglar, and P. Parrilo, "Why random reshuffling beats stochastic gradient descent," *arXiv:1510.08560*, Oct. 2015.

[73] B. Ying, K. Yuan, S. Vlaski, and A. H. Sayed, "Stochastic learning under random reshuffling," *available arXiv:1803.07964*, 2018.

[74] D. Blatt, A. O. Hero, and H. Gauchman, "A convergent incremental gradient method with a constant step size," *SIAM Journal on Optimization*, vol. 18, no. 1, pp. 29–51, 2007.

[75] M. Gurbuzbalaban, A. Ozdaglar, and P. Parrilo, "On the convergence rate of incremental aggregated gradient algorithms," *arXiv:1506.02081*, 2015.

[76] Hui Zhang, "Linear convergence of the proximal incremental aggregated gradient method under quadratic growth condition," *available on arXiv:1702.08166*, 2017.

[77] N. D. Vanli, M. Gurbuzbalaban, and A. Ozdaglar, "A stronger convergence result on the proximal incremental aggregated gradient method," *available on arXiv:1611.08022*, 2016.

[78] S. J. Reddi, A. Hefny, S. Sra, B. Poczos, and A. J. Smola, "On variance reduction in stochastic gradient descent and its asynchronous variants," in *Advances in Neural Information Processing (NIPS)*, pp. 2647–2655. MontrÃ©al, Canada, 2015.

[79] J. Konecny, H.B. McMahan, D. Ramage, and P. Richtárik, "Federated optimization: distributed machine learning for on-device intelligence," *available on arXiv:1610.02527*, Oct. 2016.

[80] K. Yuan, B. Ying, and A. H. Sayed, "Efficient variance-reduced learning for fully decentralized on-device intelligence," *available on arXiv:1708.01384*, August 2017.

[81] R. Harikandeh, M. O. Ahmed, A. Virani, M. Schmidt, J. Konecny, and S. Sallinen, "Stop wasting my gradients: Practical SVRG," in *Advances in Neural Information Processing (NIPS)*, pp. 2251–2259. MontrÃ©al, Canada, 2015.

[82] J. Xie, H. Qian, Z. Shen, and C. Zhang, "Towards memory-friendly deterministic incremental gradient method," in *Proc. International Conference on Artificial Intelligence and Statistics*, 2018, pp. 1147–1156.

[83] P. Tseng and S. Yun, "Incrementally updated gradient methods for constrained and regularized optimization," *Journal of Optimization Theory and Applications*, vol. 160, no. 3, pp. 832–853, 2014.

[84] Lin Xiao and Tong Zhang, "A proximal stochastic gradient method with progressive variance reduction," *SIAM Journal on Optimization*, vol. 24, no. 4, pp. 2057–2075, 2014.

[85] Y. Freund and R. E. Schapire, "A decision-theoretic generalization of on-line learning and an application to boosting," *Journal of Computer and System Sciences*, vol. 55, no. 1, pp. 119–139, 1997.

[86] T. Hastie and R. Tibshirani, *Generalized Additive Models*, CRC Press, 1990.

[87] Y. Freund and R. E. Schapire, "Experiments with a new boosting algorithm," in *Proc. ICML*, Bari, Italy, 1996, vol. 96, pp. 148–156.

[88] P. Viola and M. Jones, "Rapid object detection using a boosted cascade of simple features," in *Proc. CVPR*, HI, USA, 2001, vol. 1, pp. I–511.

[89] L. Breiman, "Arcing classifier," *The Annals of Statistics*, vol. 26, no. 3, pp. 801–849, 1998.

[90] Jerome H Friedman, "Greedy function approximation: a gradient boosting machine," *Annals of statistics*, pp. 1189–1232, 2001.

[91] L. Mason, J. Baxter, P. Bartlett, and M. Frean, "Boosting algorithms as gradient descent in function space," in *Proc. NIPS*, 1999.

[92] A. Lazarevic and Z. Obradovic, "Boosting algorithms for parallel and distributed learning," *Distributed and Parallel Databases*, vol. 11, no. 2, pp. 203–229, 2002.

[93] A. Lazarevic and Z. Obradovic, "The distributed boosting algorithm," in *Proc. ACM SIGKDD*, San Francisco, CA, USA, 2001, pp. 311–316.

[94] W. Fan, S. J. Stolfo, and J. Zhang, "The application of adaboost for distributed, scalable and on-line learning," in *Proc. ACM SIGKDD*, San Diego, CA, USA, 1999, pp. 362–366.

[95] L. Breiman, "Pasting small votes for classification in large databases and on-line," *Machine Learning*, vol. 36, no. 1-2, pp. 85–103, 1999.

[96] N. V. Chawla, L. O. Hall, K. W. Bowyer, T.E. Moore Jr., and W. P. Kegelmeyer, "Distributed pasting of small votes," in *Multiple Classifier Systems*, F. Roli and J. Kittler, Eds., pp. 52–61. Springer, June 2002.

[97] N. V. V Chawla, L. O. O Hall, K. W. Bowyer, and W. P. Kegelmeyer, "Learning ensembles from bites: A scalable and accurate approach," *The Journal of Machine Learning Research*, vol. 5, pp. 421–451, 2004.

[98] J. H. Friedman, "Stochastic gradient boosting," *Computational Statistics and Data Analysis*, vol. 38, no. 4, pp. 367–378, 2002.

[99] J. Friedman and R. Hastie, T.and Tibshirani, "Additive logistic regression: a statistical view of boosting (with discussion and a rejoinder by the authors)," *The Annals of Statistics*, vol. 28, no. 2, pp. 337–407, 2000.

[100] L. Breiman, J. Friedman, C. Stone, and R. Olshen, *Classification and Regression Trees*, CRC press, 1984.

[101] J. M. Borwein and A. S. Lewis, *Convex Analysis and Nonlinear Optimization: Theory and Examples*, Springer, 2010.

[102] J. Chen and A. H. Sayed, "On the learning behavior of adaptive networks—Part I: Transient analysis," *IEEE Trans. Inf. Thy.*, vol. 61, no. 6, pp. 3487–3517, June 2015.

[103] M. H. DeGroot, "Reaching a consensus," *Journal of the American Statistical Association*, vol. 69, no. 345, pp. 118–121, 1974.

[104] J. Platt, "Fast training of support vector machines using sequential minimal optimization," *Advances in Kernel Methods—Support Vector Learning*, vol. 3, 1999.

[105] T.-H. Chang, M. Hong, and X. Wang, "Multi-agent distributed optimization via inexact consensus ADMM," *IEEE Transactions on Signal Processing*, vol. 63, no. 2, pp. 482–497, 2015.

[106] B. Ying and A. H. Sayed, "Diffusion gradient boosting for networked learning," in *Proc. International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2017, pp. 2512–2516.

[107] A. Falsone, K. Margellos, S. Garatti, and M. Prandini, "Dual decomposition for multi-agent distributed optimization with coupling constraints," *Automatica*, vol. 84, pp. 149–158, 2017.

[108] W. Shi, Q. Ling, G. Wu, and W. Yin, "EXTRA: An exact first-order algorithm for decentralized consensus optimization," *SIAM Journal on Optimization*, vol. 25, no. 2, pp. 944–966, 2015.

[109] S. Kar and J. M. F. Moura, "Convergence rate analysis of distributed gossip (linear parameter) estimation: Fundamental limits and tradeoffs," *IEEE Journal of Selected Topics in Signal Processing*, vol. 5, no. 4, pp. 674–690, 2011.

[110] Z.-Q. Luo and P. Tseng, "On the convergence of the coordinate descent method for convex differentiable minimization," *Journal of Optimization Theory and Applications*, vol. 72, no. 1, pp. 7–35, 1992.

[111] P. Tseng, "Convergence of a block coordinate descent method for nondifferentiable minimization," *Journal of optimization theory and applications*, vol. 109, no. 3, pp. 475–494, 2001.

[112] A. Mokhtari, A. Koppel, and A. Ribeiro, "Doubly random parallel stochastic methods for large scale learning," in *Proc. American Control Conference (ACC)*, 2016, pp. 4847–4852.

[113] I. Notarnicola, Y. Sun, G. Scutari, and G. Notarstefano, "Distributed big-data optimization via block-iterative convexification and averaging," in *Proc. Decision and Control (CDC)*, 2017, pp. 2281–2288.

[114] C. Wang, Y. Zhang, B. Ying, and A. H. Sayed, "Coordinate-descent diffusion learning by networked agents," *IEEE Transactions on Signal Processing*, vol. 66, no. 2, pp. 352–367, Jan 2018.

[115] S. Shalev-Shwartz and T. Zhang, "Stochastic dual coordinate ascent methods for regularized loss minimization," *Journal of Machine Learning Research*, vol. 14, no. Feb, pp. 567–599, 2013.

[116] S. Shalev-Shwartz, "SDCA without duality, regularization, and individual convexity," in *International Conference on Machine Learning*, 2016, pp. 747–754.

[117] S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein, "Distributed optimization and statistical learning via the alternating direction method of multipliers," *Foundations and Trends in Machine Learning*, vol. 3, no. 1, pp. 1–122, 2011.

[118] W. Shi, Q. Ling, K. Yuan, G. Wu, and W. Yin, "On the linear convergence of the admm in decentralized consensus optimization.," *IEEE Trans. Signal Processing*, vol. 62, no. 7, pp. 1750–1761, 2014.

[119] T.-H. Chang, A. Nedic, and A. Scaglione, "Distributed constrained optimization by consensus-based primal-dual perturbation method," *IEEE Transactions on Automatic Control*, vol. 59, no. 6, pp. 1524–1538, 2014.

[120] J. F. Mota, J. M. Xavier, P. M. Aguiar, and M. Puschel, "Distributed basis pursuit," *IEEE Transactions on Signal Processing*, vol. 60, no. 4, pp. 1942–1956, 2012.

[121] M. Zhu and S. Martinez, "Discrete-time dynamic average consensus," *Automatica*, vol. 46, no. 2, pp. 322–329, 2010.

[122] R. A. Freeman, P. Yang, and K. M. Lynch, "Stability and convergence properties of dynamic average consensus estimators," in *Proc. Conference on Decision and Control*, 2006, pp. 338–343.

[123] P. D. Lorenzo and G. Scutari, "Next: In-network nonconvex optimization," *IEEE Transactions on Signal and Information Processing over Networks*, vol. 2, no. 2, pp. 120–136, 2016.

[124] A. Nedic, A. Olshevsky, and W. Shi, "Achieving geometric convergence for distributed optimization over time-varying graphs," *SIAM Journal on Optimization*, vol. 27, no. 4, pp. 2597–2633, 2017.

[125] S. Pu, W. Shi, J. Xu, and A. Nedich, "A push-pull gradient method for distributed optimization in networks," *Available at arXiv:1803.07588*, 2018.

[126] R. Xin and U. A. Khan, "A linear algorithm for optimization over directed graphs with geometric convergence," *Available at arXiv:1803.02503*, 2018.

[127] J. L. Hennessy and D. A. Patterson, *Computer architecture: a quantitative approach*, Elsevier, 2011.

[128] S. Boyd, A. Ghosh, B. Prabhakar, and D. Shah, "Randomized gossip algorithms," *IEEE Trans. Inf. Theory*, vol. 52, no. 6, pp. 2508–2530, 2006.

[129] R. A. Horn and C. R. Johnson, *Matrix analysis*, Cambridge university press, 1990.

[130] S. U. Pillai, T. Suel, and S. Cha, "The perron-frobenius theorem: some of its applications," *IEEE Signal Processing Magazine*, vol. 22, no. 2, pp. 62–75, 2005.

[131] K. Yuan, B. Ying, X. Zhao, and A. H. Sayed, "Exact dffusion for distributed optimization and learning – Part II: Convergence analysis," *Submitted for publication, Also available on arXiv*, Feb. 2017.

[132] A. Agarwal and J. C. Duchi, "Distributed delayed stochastic optimization," in *Advances in Neural Information Processing Systems (NIPS)*, 2011, pp. 873–881.

[133] Z. Peng, Y. Xu, M. Yan, and W. Yin, "ARock: an algorithmic framework for asynchronous parallel coordinate updates," *SIAM Journal on Scientific Computing*, vol. 38, no. 5, pp. A2851–A2879, 2016.

[134] T. Wu, K. Yuan, Q. Ling, W. Yin, and A. H. Sayed, "Decentralized consensus optimization with asynchrony and delays," *IEEE Transactions on Signal and Information Processing over Networks, To appear*, 2018.

[135] M. Penrose, *Random geometric graphs*, Number 5. Oxford university press, 2003.

[136] Y. Nesterov, *Introductory Lectures on Convex Optimization*, Springer, 2004.

[137] R. Tibshirani, "Regression shrinkage and selection via the lasso," *Journal of the Royal Statistical Society, Series B (Methodological)*, pp. 267–288, 1996.

[138] C. Cortes and V. Vapnik, "Support-vector networks," *Machine learning*, vol. 20, no. 3, pp. 273–297, 1995.

[139] S. Theodoridis and K. Koutroumbas, *Pattern Recognition*, Academic Press, 4th edition, 2008.

[140] L. I. Rudin, S. Osher, and E. Fatemi, "Nonlinear total variation based noise removal algorithms," *Physica D: Nonlinear Phenomena*, vol. 60, no. 1, pp. 259–268, 1992.

[141] J. E. Kelley, "The cutting-plane method for solving convex programs," *Journal of the Society for Industrial and Applied Mathematics*, pp. 703–712, 1960.

[142] N. Parikh and S. Boyd, "Proximal algorithms," *Foundations and Trends in optimization*, vol. 1, no. 3, pp. 123–231, 2013.

[143] F. Bach, R. Jenatton, J. Mairal, and G. Obozinski, "Optimization with sparsity-inducing penalties," *Foundations and Trends in Machine Learning*, vol. 4, no. 1, pp. 1–106, 2012.

[144] L. Bottou, "Stochastic gradient tricks," in *Neural Networks, Tricks of the Trade, Reloaded*, G. Montavon, G. B. Orr, and K. Müller, Eds., Lecture Notes in Computer Science (LNCS 7700), pp. 430–445. Springer, 2012.

[145] Z. J. Towfic and A. H. Sayed, "Stability and performance limits of adaptive primal-dual networks," *IEEE Trans. Signal Process.*, vol. 63, no. 11, pp. 2888–2903, June 2015.

[146] J. Chen and A. H. Sayed, "On the learning behavior of adaptive networks—Part II: Performance analysis," *IEEE Trans. Inf. Thy.*, vol. 61, no. 6, pp. 3518–3548, June 2015.

[147] S. S. Ram, A. Nedić, and V. V. Veeravalli, "Distributed stochastic subgradient projection algorithms for convex optimization," *J. Optm. Theory and Appl.*, vol. 147, no. 3, pp. 516–545, 2010.

[148] S. Boyd and A. Mutapcic, "Stochastic subgradient methods," *Lecture Notes for EE364b, Stanford University,*, 2008.

[149] A. Agarwal, P. L. Bartlett, P. Ravikumar, and M. J. Wainwright, "Information-theoretic lower bounds on the oracle complexity of stochastic convex optimization," *IEEE Trans. Inf. Theory*, vol. 58, no. 5, pp. 3235–3249, May 2012.

[150] A. H. Sayed, *Adaptive Filters*, Wiley, NY, 2008.

[151] N. Z. Shor, *Minimization Methods for Non-differentiable Functions*, vol. 3, Springer-Verlag, 2012.

[152] K. Kiwiel, *Methods of Descent for Non-differentiable Optimization*, Springer-Verlag, 1985.

[153] Y. Chen, Y. Gu, and A. O. Hero, "Sparse LMS for system identification," in *Proc. IEEE ICASSP*, Taipei, Taiwan, 2009, pp. 3125–3128.

[154] S. S. Chen, D. L. Donoho, and M. A. Saunders, "Atomic decomposition by basis pursuit," *SIAM journal on scientific computing*, vol. 20, no. 1, pp. 33–61, 1998.

[155] D. L. Donoho, M. Elad, and V. N. Temlyakov, "Stable recovery of sparse overcomplete representations in the presence of noise," *IEEE Trans. Inf. Theory*, vol. 52, no. 1, pp. 6–18, 2006.

[156] Y. Murakami, M. Yamagishi, M. Yukawa, and I. Yamada, "A sparse adaptive filtering using time-varying soft-thresholding techniques," in *Proc. IEEE Int. Conf. Acoust. Speech Signal Process. (ICASSP)*, Dallas, TX, 2010, pp. 3734–3737.

[157] A. Nedic, "Subgradient methods for convex minimization," May 2002.

[158] D. L. Donoho and J. M. Johnstone, "Ideal spatial adaptation by wavelet shrinkage," *Biometrika*, vol. 81, no. 3, pp. 425–455, 1994.

[159] D. L. Duttweiler, "Proportionate normalized least-mean-squares adaptation in echo cancelers," *IEEE Trans. Speech and Audio Process.*, vol. 8, no. 5, pp. 508–518, Sep. 2000.

[160] Y. Kopsinis, K. Slavakis, and S. Theodoridis, "Online sparse system identification and signal reconstruction using projections onto weighted balls," *IEEE Trans. Signal Process.*, vol. 59, no. 3, pp. 936–952, Mar. 2011.

[161] A. Beck and M. Teboulle, "A fast iterative shrinkage-thresholding algorithm for linear inverse problems," *SIAM J. Imag. Sci.*, vol. 2, no. 1, pp. 183–202, 2009.

[162] W. Yu, G. Chen, Z. Wang, and W. Yang, "Distributed consensus filtering in sensor networks," *IEEE Trans. on Systems, Man, and Cybernetics, Part B: Cybernetics*, vol. 39, no. 6, pp. 1568–1577, 2009.

[163] S.-Y. Tu and A. H. Sayed, "Diffusion strategies outperform consensus strategies for distributed estimation over adaptive networks," *IEEE Trans. Signal Process.*, vol. 60, no. 12, pp. 6217–6234, 2012.

[164] P. Di Lorenzo, S. Barbarossa, and A. H. Sayed, "Sparse diffusion LMS for distributed adaptive estimation," in *Proc. IEEE Int. Conf. Acoust. Speech Signal Process. (ICASSP)*, Kyoto, Japan, Mar. 2012, pp. 3281–3284.

[165] P. Di Lorenzo and A. H. Sayed, "Sparse distributed learning based on diffusion adaptation," *IEEE Trans. Signal Process.*, vol. 61, no. 6, pp. 1419–1433, March 2013.

[166] Y. Liu, C. Li, and Z. Zhang, "Diffusion sparse least-mean squares over networks," *IEEE Trans. Signal Process.*, vol. 60, no. 8, pp. 4480–4485, Aug. 2012.

[167] S. Chouvardas, K. Slavakis, Y. Kopsinis, and S. Theodoridis, "A sparsity promoting adaptive algorithm for distributed learning," *IEEE Trans. Signal Process.*, vol. 60, no. 10, pp. 5412–5425, Oct. 2012.

[168] C. D. Meyer, *Matrix Analysis and Applied Linear Algebra*, SIAM, 2000.

[169] F. S. Cattivelli and A. H. Sayed, "Diffusion LMS strategies for distributed estimation," *IEEE Trans. Signal Process.*, vol. 58, no. 3, pp. 1035–1048, 2010.

[170] A. J. Laub, *Matrix Analysis for Scientists and Engineers*, SIAM, 2005.