Syracuse University

SURFACE at Syracuse University

Dissertations - ALL

SURFACE at Syracuse University

Summer 8-27-2021

Some New Results in Distributed Tracking and Optimization

Pranay Sharma
Syracuse University

Follow this and additional works at: https://surface.syr.edu/etd

Part of the Electrical and Computer Engineering Commons

Recommended Citation

Sharma, Pranay, "Some New Results in Distributed Tracking and Optimization" (2021). *Dissertations - ALL*. 1375.
https://surface.syr.edu/etd/1375

This Dissertation is brought to you for free and open access by the SURFACE at Syracuse University at SURFACE at Syracuse University. It has been accepted for inclusion in Dissertations - ALL by an authorized administrator of SURFACE at Syracuse University. For more information, please contact surface@syr.edu.

ABSTRACT

The current age of Big Data is built on the foundation of distributed systems, and efficient distributed algorithms to run on these systems. With the rapid increase in the volume of the data being fed into these systems, storing and processing all this data at a central location becomes infeasible. Such a central *server* requires a gigantic amount of computational and storage resources. Even when it is possible to have central servers, it is not always desirable, due to privacy concerns. Also, sending huge amounts of data to such servers incur often infeasible bandwidth requirements.

In this dissertation, we consider two kinds of distributed architectures: 1) star-shaped topology, where multiple worker nodes are connected to, and communicate with a server, but the workers do not communicate with each other; and 2) mesh topology or network of interconnected workers, where each worker can communicate with a small number of neighboring workers.

In the first half of this dissertation (Chapters 2 and 3), we consider distributed systems with mesh topology. We study two different problems in this context. First, we study the problem of simultaneous localization and multi-target tracking. Multiple mobile agents localize themselves cooperatively, while also tracking multiple, unknown number of mobile targets, in the presence of measurement-origin uncertainty. In situations with limited GPS signal availability, agents (like self-driving cars in urban canyons, or autonomous vehicles in hazardous environments) need to rely on inter-agent measurements for localization. The agents perform the additional task of tracking multiple targets (pedestrians and road-signs for self-driving cars). We propose a decentralized algorithm for this problem. To be effective in real-time applications, we propose efficient Gaussian and Gaussian-mixture based filters, rather than the computationally expensive particle-based methods in the existing literature. Our novel factor-graph based approach gives better performance, in terms of both agent localization errors, and target-location and cardinality errors.

Next, we study an online convex optimization problem, where a network of agents cooperate to minimize a global time-varying objective function. Only the local functions are revealed to

individual agents. The agents also need to satisfy their individual constraints. We propose a primal-dual update based decentralized algorithm for this problem. Under standard assumptions, we prove that the proposed algorithm achieves sublinear regret and constraint violation across the network. In other words, over a long enough time horizon, the decisions taken by the agents are, on average, as good as if all the information was revealed ahead of time. In addition, the individual constraint violations of the agents, averaged over time, are zero.

In the next part of the dissertation (Chapters 4), we study distributed systems with a star-shaped topology. The problem we study is distributed nonconvex optimization. With the recent success of deep learning, coupled with the use of distributed systems to solve large-scale problems, this problem has gained prominence over the past decade. The recently proposed paradigm of Federated Learning (which has already been deployed by Google/Apple in Android/iOS phones) has further catalyzed research in this direction. The problem we consider is minimizing the average of local smooth, nonconvex functions. Each node has access only to its own loss function, but can communicate with the server, which aggregates updates from all the nodes, before distributing them to all the nodes. With the advent of more and more complex neural network architectures, these updates can be high dimensional. To save resources, the problem needs to be solved via communication-efficient approaches. We propose a novel algorithm, which combines the idea of variance-reduction, with the paradigm of carrying out multiple local updates at each node before averaging. We prove the convergence of the approach to a first-order stationary point. Our algorithm is optimal in terms of computation, and state-of-the-art in terms of the communication requirements.

Lastly in Chapter 5, we consider the situation when the nodes do not have access to function gradients, and need to minimize the loss function using only function values. This problem lies in the domain of zeroth-order optimization. For simplicity of analysis, we study this problem only in the single-node case. This problem finds application in simulation-based optimization, and adversarial example generation for attacking deep neural networks. We propose a novel function value based gradient estimator, which has better variance, and better query-efficiency compared to

existing estimators. The proposed estimator covers the most commonly used existing estimators as special cases. We conduct a comprehensive convergence analysis under different conditions. We also demonstrate its effectiveness through a real-world application to generating adversarial examples from a black-box deep neural network.

To my parents

and Shweta

SOME NEW RESULTS IN DISTRIBUTED TRACKING AND OPTIMIZATION

By

Pranay Sharma
B.Tech-M.Tech, Indian Institute of Technology, Kanpur, 2013

DISSERTATION

Submitted in partial fulfillment of the requirements for the degree of
Doctor of Philosophy in Electrical and Computer Engineering

Syracuse University
August 2021

Copyright © Pranay Sharma, 2021

All Rights Reserved

ACKNOWLEDGMENTS

"He who would learn to fly one day must first learn to stand and walk and run and climb and dance; one cannot fly into flying." Friedrich Nietzsche

"A human being should be able to change a diaper, plan an invasion, butcher a hog, conn a ship, design a building, write a sonnet, balance accounts, build a wall, set a bone, comfort the dying, take orders, give orders, cooperate, act alone, solve equations, analyze a new problem, pitch manure, program a computer, cook a tasty meal, fight efficiently, die gallantly.

Specialization is for insects." Robert Anson Heinlein

I would like to express my profound gratitude to my advisor Prof. Pramod K. Varshney for his invaluable guidance along with the continued intellectual and motivational support throughout my doctoral studies. With his patience, wisdom, and consistent encouragement, he has played the role the Jambavanta (this reference comes from Hindu mythology) for me these past 6 years. I would like to express my gratitude to Prof. Sijia Liu (Michigan State University), Prof. Lixin Shen, Prof. Ketan Rajawat (Indian Institute of Technology, Kanpur, India), Dr. Alexandru-Augustin Saucan (Massachussets Institute of Technology), Dr. Donald J. Bucci Jr. (Lockheed Martin, Advanced Technologies Lab), and Prof. Swastik Brahma (Tennessee State University) for mentoring me during the course of my research work. I would also like to thank my defense committee members Prof. Lixin Shen, Prof. Biao Chen, Prof. Makan Fardad, Prof. Venkata S. S. Gandikota, and Prof. Ferdinando Fioretto for their valuable suggestions.

I would like to extend sincere thanks to my current and past "Sensor Fusion Lab" members and friends, including Aditya, Arun, Bao, Bhavya, Chen, Hao, Nandan, Nianxia, Prashant, Qunwei, Raghed, Sai, Sid, Sijia, Shan, Swatantra, and Xianchang for all the helpful technical and philosophical discussions we had during the course of this dissertation. Times spent with Manish, Prashant, Rajesh, Sai, and Swatantra, will forever be cherished. I would like to give special thanks to Aunty for her motherly love and making my stay in Syracuse feel like home.

I am grateful to all the professors who taught me during my undergraduate and masters days at IIT Kanpur. Special thanks to Santosh Nigam, my high-school Physics teacher, to whom I owe a greater debt than any other teacher in my life. A heartfelt thanks to all my friends outside my academic circle including my high-school and college friends for their unwavering support throughout this journey. Animesh, Sagar, Vishal, Akash, Mohit, Piyush, and the entire Y8 2-top group deserve special mention.

This past year and half since covid-19 hit, has definitely shown how fortunate many of us are in life. I see the world around me full of people, who are far more deserving of the gifts life has offered me, but are less fortunate. We are surrounded by people, who make our lives better, and never get any credit for it. These are the janitors in our offices, the administrative staff in our buildings, the shuttle/bus drivers, the maintenance crew, the security officers, and most significantly in the current times, the frontline health workers. I thank them all.

Lastly and most importantly, I am grateful to my parents for their unconditional love and selfless care; to my brother and sister-in-law for their consistent support throughout this journey; and to my nephew Avyaan, who just turned one, well, for being. My mother is the most hardworking person I know. I guess she passed some of that on to the next generation. Even post retirement, my parents have an amazing zeal to learn something new everyday. I hope their enthusiasm continues to inspire me as well. Finally, I would like to thank my wife, Shweta, whose mere presence makes the world a better place. "You make me wanna be a better man."

TABLE OF CONTENTS

Acknowledgments **vii**

List of Tables **xiv**

List of Figures **xv**

1 Introduction **1**

1.1 Mesh Networks 3

1.1.1 Self-localization and Target Tracking using Network of Agents 3

1.1.2 Online Convex Optimization over Network of Agents 5

1.2 Stochastic Nonconvex Optimization over Star-Topology 7

1.3 Zeroth-Order Stochastic Nonconvex Optimization 9

1.4 Organization of the Dissertation 10

1.5 Bibliographic Note 10

2 Cooperative Self-localization and Multi-target Tracking **13**

2.1 Introduction 13

2.1.1 Related Work 14

2.1.2 Our Contributions 15

2.2 System Model and notation 16

2.2.1 Notation 16

2.2.2 Assumptions 17

2.2.3 System Model 19

2.3 The SCS-MTT filter 22
    2.3.1 The SCS-MTT filter: the BP message passing scheme 23
    2.3.2 Agent and target inference 29
2.4 Decentralized Gaussian Mixture SCS-MTT filter 30
    2.4.1 GM prediction and likelihood messages 33
    2.4.2 Agent belief via centralized GM product 36
    2.4.3 Target belief via decentralized GM product 40
2.5 Decentralized Gaussian SCS-MTT filter 46
2.6 Simulation Results 48
    2.6.1 SCS-MTT vs SPAWN 51
    2.6.2 Single Gaussian vs Gaussian mixture SCS-MTT filters 53
    2.6.3 Centralized vs decentralized Gaussian mixture 54
2.7 Summary 55

3 Distributed Online Convex Optimization **56**
3.1 Introduction 56
    3.1.1 Related Work 57
    3.1.2 Our Contributions 59
3.2 Problem Formulation 60
    3.2.1 Performance Metrics - Dynamic Regret and Fit 60
3.3 Background and Assumptions 61
    3.3.1 Network 61
    3.3.2 Local Objective Functions and Constraints 62
    3.3.3 Bregman Divergence 63
    3.3.4 Projection 64
3.4 Distributed Primal-Dual Mirror Descent based Algorithm 65
3.5 Dynamic Regret and Fit Bounds 66
    3.5.1 Some Intermediate Results 66

3.5.2 Dynamic Regret and Fit Bounds 70
3.6 Conclusion 73

4 Distributed Stochastic First-order Optimization: PR-SPIDER 74

4.1 Introduction 74
4.1.1 Problem 74
4.1.2 Related Work 76
4.1.3 Our Contributions 81
4.1.4 Notations and Assumptions 82
4.2 Parallel Restarted SPIDER - Finite Sum Case 83
4.2.1 Proposed Algorithm 83
4.2.2 Convergence Result 85
4.2.3 Convergence Analysis 86
4.3 Parallel Restarted SPIDER - Online Case 94
4.3.1 Proposed Algorithm 94
4.3.2 Convergence Result 95
4.3.3 Convergence Analysis 97
4.4 Simulation Results 101
4.5 Drawbacks of PR-SPIDER and A Novel Algorithm to Fix Them 104
4.5.1 Drawbacks of PR-SPIDER 104
4.5.2 STEM Algorithm 105
4.5.3 Main results: convergence guarantees for STEM 106
4.5.4 Simulation Results for STEM 108
4.6 Summary 110

5 Stochastic Zeroth-order Optimization: Hybrid Gradient Descent 111

5.1 Introduction 111
5.1.1 Motivating Application: Adversarial Example Generation 112

5.1.2 Related Work 112
5.1.3 Our Contributions 113
5.2 Preliminaries 114
5.3 Hybrid Gradient Estimator 116
5.4 ZO Hybrid Gradient Descent 123
5.4.1 The Algorithm 123
5.4.2 Technical challenges of ZO-HGD. 124
5.4.3 Convergence analysis. 124
5.4.4 Comparison with Other Methods 131
5.4.5 Tradeoff between RGE and CGE in ZO-HGD. 132
5.5 ZO-HGD for Convex Optimization 135
5.6 Simulation Results 136
5.7 Summary 139

6 Conclusion **140**

6.1 Summary 140
6.2 Future Directions 141
6.2.1 Federated Learning 141
6.2.2 Online Nonconvex Optimization 143
6.2.3 Stochastic Compositional Optimization 144

A Appendix **145**

A.1 Cooperative Self-localization and Multi-Target Tracking 145
A.2 Distributed Online Convex Optimization 148
A.2.1 Proof of Lemma 3.5.1 148
A.2.2 Proof of Lemma 3.5.2 149
A.2.3 Proof of Lemma 3.5.3 152
A.3 PR-SPIDER: Convergence of the Finite-sum case 154

A.3.1 Proof of Lemma 4.2.2 154

A.3.2 Proof of Lemma 4.2.3 156

A.3.3 Intermediate results for Lemma 4.2.4 157

A.3.4 Choice of γ 161

A.4 Zeroth-order Hybrid Gradient Descent 162

A.4.1 Mathematical Notations 162

A.4.2 Proof of Proposition 5.3.1 162

A.4.3 Proof of Proposition 5.3.2 165

A.4.4 Choice of α 167

A.4.5 Special Cases of Theorem 5.4.3 169

A.4.6 Convex Case 170

A.4.7 Proof of Theorem 5.5.1 (Convex Case) 171

A.4.8 Special Cases of Theorem 5.5.1 (Convex Case) 174

A.4.9 Proof of Theorem 5.5.1 (Strongly Convex Case) 176

References **181**

LIST OF TABLES

2.1 State transition kernels for different values of target existence indicators. The function $f_D(\cdot)$ is a dummy pdf [126]. 19

2.2 Gaussian mixture parameters for the message passing scheme of Algorithm 1. . . . 34

2.3 Conditional sampling of a new label vector $\boldsymbol{i}'$ given previous labels $\boldsymbol{i}$ in synchronous and Hogwild! Gibbs. 42

4.1 IFO complexity of different algorithms to reach an ϵ-stationary point, for the stochastic smooth non-convex optimization problem. $\tilde{O}(\cdot)$ hides logarithmic factors. . . . 78

4.2 Communication and computation complexities of different algorithms to reach an ϵ-FoS point, for the stochastic smooth non-convex optimization problem. 80

5.1 Comparison of different ZO algorithms, for solving smooth, nonconvex optimization problems. 133

5.2 Randomly selected 5 over 10 images in one trial from our universal black-box attack generation using ZO-SGD, ZO-SCD, ZO-signSGD and ZO-HGD. 139

A1 Comparison of smoothing parameters and convergence rates, for difference regimes of RGE and CGE query budgets 174

LIST OF FIGURES

1.1 Two possible topologies for distributed systems 2

2.1 Factor graph representing the factorization of (2.11), for one time step. 24

2.2 GM processing flowchart of the DGM-SCS-MTT filter at time n. 32

2.3 Ground truth trajectories of agents and targets plotted over time (top), along with the true target cardinality over time (bottom). 48

2.4 Comparison of the average agent RMSE and target OSPA error for SPAWN and the proposed SCS-MTT filter. 51

2.5 Comparison of the average agent RMSE and target OSPA error, for the single Gaussian (G) and Gaussian mixture (GM) filters. 52

2.6 Comparison of cardinality estimates for the different GM filters. 53

2.7 Comparison of average agent RMSE and target OSPA error for the DGM-SCS-MTT and CGM-SCS-MTT filters. 54

4.1 Convergence of PR-SPIDER for over MNIST dataset. 102

4.2 Convergence of PR-SPIDER for ResNet20 over CIFAR-10 dataset. 103

4.3 Training loss and testing accuracy in the moderate heterogeneity setting. $b = 8$ and $I = 61$. 109

4.4 Training loss and testing accuracy in the moderate heterogeneity setting. $b = 64$ and $I = 7$. 109

4.5 Training loss and testing accuracy in the high heterogeneity setting. $b = 8$ and $I = 61$. 109

5.1 Averaged objective value when solving problem (5.37) over 50 random trials versus the number of function queries. 137

5.2 Comparison of ZO-HGD with other methods for generating universal adversarial perturbations. 138

A1 Ground truth trajectories of agents and targets, plotted over time. 145

A2 Average RMSE in agent location estimates over time. 146

A3 Mean OSPA error, position error, and cardinality error for targets plotted over time. 147

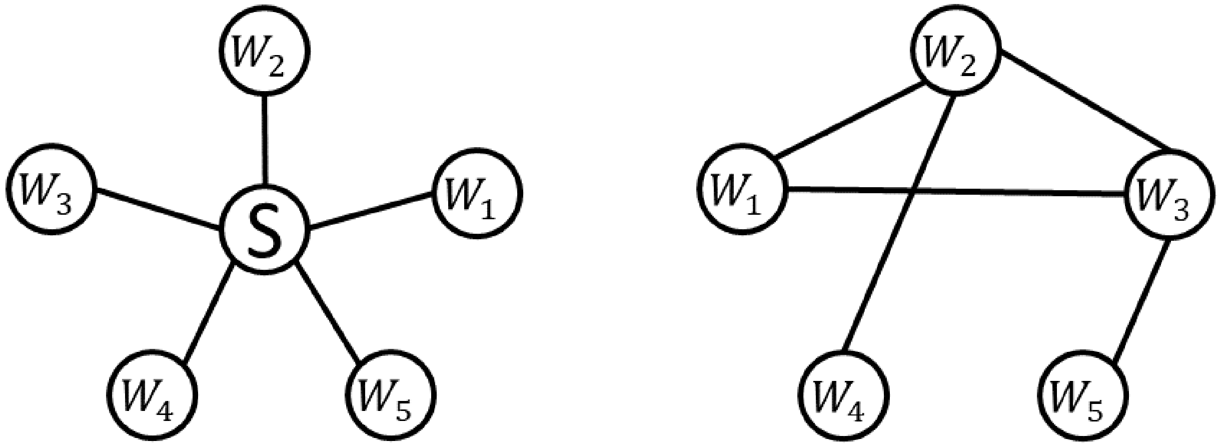
CHAPTER 1

INTRODUCTION

The current age of Big Data is built on the foundation of distributed systems, and efficient distributed algorithms to run on these systems [195]. With the rapid increase in the volume of the data being fed into these systems, storing and processing all this data at a central location becomes infeasible. Such a central *server* requires a gigantic amount of computational and storage resources [194]. Also, the input data is usually collected from a myriad of sources, which might inherently be distributed [98]. Transferring all this data to a single location might be expensive. Depending on the sensitivity of the underlying data, this might also lead to concerns about maintaining the privacy and anonymity of this data [103]. Therefore, even in situations where it is possible to have central servers, it is not always desirable.

Distributed systems might have 1) star-shaped topology (Figure 1.1a), where each *worker node* (denoted by $W_i, i = 1, \ldots, 5$) is connected to a server (denoted by S), but the workers do not get to communicate with each other; or 2) mesh topology or a network of interconnected workers (Figure 1.1b), where each worker can communicate with a small number of workers, often in its geographical vicinity.

In this dissertation, we consider both these kinds of distributed systems to solve different, but somewhat related problems.



(a) Star Topology **(b)** Mesh Topology

Fig. 1.1: Two possible topologies for distributed systems

1. For systems with workers[1] connected in a mesh-topology, we study the following problems.
   - A simultaneous localization and multi-target tracking problem. Multiple mobile agents localize themselves cooperatively, while also tracking multiple, unknown number of mobile targets, in the presence of measurement-origin uncertainty.
   - An online convex optimization problem, where a network of agents cooperate to minimize a global time-varying objective function. However, only the local functions are revealed to individual agents. The agents also need to satisfy their individual constraints.
2. For distributed systems with a star-shaped topology, we study two related distributed stochastic nonconvex optimization problems. The proposed first-order algorithm is optimal in terms of computation, and state-of-the-art in terms of the communication requirements.
3. Finally, we study a single-node[2] zeroth-order stochastic non-convex optimization problem. This problem finds application in adversarial example generation for attacking deep neural networks.

---

[1] In this dissertation, we use the terms devices, nodes, agents to mean the same thing

[2] As a logical next step to the previous problem, we study the scenario where the gradient of the cost function is no longer available. However, to simplify the analysis, we focus only on the single-node case.

In the following sections, we describe all these problems in some detail.

1.1 Mesh Networks

Networks consisting of mobile interconnected agents with different sensing capabilities have found numerous applications in a multitude of applications including surveillance [5], target tracking [13], intelligent transportation systems [163, 164], and robotics [24]. In Chapters 2, we address the problem of using such a network of multiple mobile agents to track multiple (unknown, time-varying number of) mobile targets. In addition to tracking the targets, the agents are supposed to cooperatively estimate their own locations (states) as well. In Chapter 3, we consider a more abstract mathematical formulation of online convex optimization. A network of agents cooperatively minimize a global time-varying objective function, while also satisfying their individual time-varying constraints. Next, we discuss both the problems in more detail.

1.1.1 Self-localization and Target Tracking using Network of Agents

In GPS-denied environments and for agents with limited power, cooperative self-localization (CS) schemes that rely on inter-agent measurements become necessary. The objective of multi-target tracking (MTT) is the estimation of the states of an unknown and time-varying number of targets. At any time instant, the sensors of an agent produce two kinds of measurements:

1. Inter-agent measurements - by observing other agents in proximity, and
2. Target measurements - by observing the targets that are within the measurement range of the agent.

Due to the collaborative nature of CS, the inter-agent measurements are assumed unambiguous, i.e., the identity of the neighboring agent is known for each inter-agent measurement. On the other hand, targets are non-cooperative and the measurement-to-target associations are not known. Clutter and missed detections also affect the target measurement set. We refer to the problem as

(S)imultaenous CS-MTT. At each time instant, each agent is supposed to estimate its own state, as well as the states of all the targets, by communicating only with its neighboring agents. Unlike some existing work [74], the agents do not exchange their raw measurements, since this incurs huge bandwidth overhead, especially with large image or video measurements. Instead, agents maintain state estimates (for themselves, and for the targets) using marginal posterior densities.

Past work has focused on bits and pieces of this problem. In [192], CS is achieved via the SPAWN (Sum-Product Algorithm over a Wireless Network) method which relies on Belief Propagation (BP) [101]. In [127], simultaneous cooperative self-localization and target tracking is performed. However, the number of targets is assumed fixed and known. In addition, perfect association between measurements and targets is assumed known at each agent. These two assumptions are relaxed in [126]. However, the algorithm is centralized and only solves the tracking problem, without sensor self-localization. The problem of SCS-MTT, in its full generality that we consider here, has not been tackled so far in the literature.

Also, existing methods rely on particle representations of agent and target probability densities and the BP messages. Particle filters (PF) [47] are methods for sequential estimation of the state vector in highly non-linear and/or non-Gaussian state systems. However, the computational and communication requirements of PF-based methods can be quite high. This creates a bottleneck from the computational standpoint in real-time applications. Gaussian densities, on the other hand, are simpler, yet limited in their expressive power. Gaussian mixture densities strike a balance between the two extremes, and can be tweaked towards either end, by varying the number of modes.

Our Contributions

- We propose an efficient, decentralized BP message passing based algorithm for simultaneous cooperative self-localization (of mobile agents) and multi-target tracking (SCS-MTT), under measurement-to-target association uncertainty. The factor graph of the joint posterior over agent and target states has cycles and several message orderings are possible. The novelty of our contribution also lies in the ordering of messages that ensures a reduced amount of data exchange over the network.

- To save on the computational and communication costs of particle filters, we model the state densities using single Gaussians and Gaussian mixtures (GMs). The Gaussian-based and GM-based filters achieve network-wide consensus over the target beliefs, i.e., over the means, covariance matrices and component weights of the Gaussian Mixtures (GM). Our GM-based approach allows a robust trade-off between computational efficiency (GMs with fewer modes) and modeling accuracy (higher number of modes).
- In case of GMs, the decentralized computation of target beliefs involves a product of GM likelihood messages (stored at different agents) and a GM prior. The number of components in the complete GM product is exponential in the number of agents. To overcome this high complexity, we propose a novel decentralized Gibbs mechanism, extending the centralized Gibbs approach proposed in [168], to sample only the components of the GM product with the highest weights, and thus approximate the entire product. This approach of efficiently computing product of GMs will be of independent interest to the research community.
- Our numerical results show that the performance of the proposed decentralized algorithm is close to its centralized counterpart. Numerical experiments also exhibit performance improvement compared to a separate SPAWN (for localization) and MTT approach (see [127]).

1.1.2 Online Convex Optimization over Network of Agents

Next, in Chapter 3, we discuss a more abstract, mathematical problem, under the paradigm of online convex optimization (OCO) [76, 157]. In this paradigm, the agent makes decisions iteratively. After taking the decision/action, the agent incurs a loss/cost. Owing to the dynamic nature of the problem, the loss function might vary across time. Also, the agent might need to satisfy some individual constraints, which might be fixed across time, or might themselves be time-varying. We consider this problem in the context of agents, connected in a network.

A set of nodes, jointly aim to minimize a global objective function, which is the sum of local convex functions. The objective and constraint functions are revealed locally to the nodes, at each time, after taking an action. To quantify the performance of the proposed algorithm, we utilize the metrics of *dynamic regret* [20] and *fit* [144]. Dynamic regret measures the cumulative (over time) loss incurred by the algorithm, compared to the best possible dynamic strategy. On the other hand, fit measures the long term cumulative constraint violations. The aim of OCO algorithms is to prove that the regret (and fit) are sublinear in the time horizon (usually denoted by T). If the regret is sublinear, on average (averaged over T), the decisions taken by the agent are *as good as* those of an agent with complete knowledge of the loss functions ahead of time. Similarly, a sublinear fit implies, on average, no constraint violation.

Our Contributions

- We consider the distributed online convex optimization problem, with nodes arranged in a mesh topology. The cost functions and the time-varying inequality constraints are revealed locally to the individual nodes. With a dynamic benchmark, this problem has so far not been considered in the literature even with static non-coupled inequality constraints.
- We propose a distributed primal-dual mirror descent based approach, in which the primal and dual updates are carried out locally at all the nodes. This is followed by sharing and mixing of the primal variables by the local nodes via communication with the immediate neighbors.
- We prove that the proposed algorithm has sublinear dynamic regret. In other words, over time, the proposed algorithm performs as well as a clairvoyant algorithm, which has all the information of the network, ahead of time. We also prove the dynamic fit to be sublinear, i.e., on average, the cumulative constraint violation over time, averaged across all the nodes in the network is zero.

A special case of the online optimization problem described above is when the cost function is not time-varying. In that case, the problem *reduces* to that of stochastic optimization. This special case enables rich mathematical analysis, which has so far not been done for the more general online case. For example, other than some notable exceptions [197, 4, 59, 169], the online optimization literature is limited to the convex case. On the other hand, over the past decade, much progress has been made in the stochastic optimization literature on nonconvex problems. In the next two sections, we outline our work on stochastic nonconvex optimization.

1.2 Stochastic Nonconvex Optimization over Star-Topology

One of the factors which has made this Big Data explosion possible is the meteoric rise in the capabilities, and the consequent proliferation, of end-user devices [98]. These *worker* nodes or machines have significant storage and computational resources. One simple solution to the potential central server bottleneck problem faced by distributed systems is for the server to offload some of its conventional tasks to these workers [194]. More precisely, instead of sharing its entire data with the server, which is expensive communication-wise, each worker node carries computations on its data itself. The results of these "local" computations are then shared with the server. The server aggregates these local updates from multiple workers to arrive at a "global" estimate, which is then shared with all the workers.

In Chapter 4, we solve a distributed optimization problem on worker-server or star-shaped architecture. This problem has application in several domains including robust regression [179], sparse regression [132], image denoising [1], and community detection in social networks [82] to name a few. The optimization problem we solve is as follows:

$$\min_{x \in \mathbb{R}^d} f(\mathbf{x}) \triangleq \frac{1}{N} \sum\nolimits_{i=1}^{N} f_i(\mathbf{x}),$$

where N is the number of worker nodes. The local function corresponding to node i, $f_i(\mathbf{x})$ is a smooth, potentially non-convex function. This problem has been extensively studied in the literature [159, 182, 172]. See the survey [116] and the reference therein. The ideal solution for this problem should have two properties: 1) optimal computation cost, i.e., the algorithm should achieve the solution (Definition 4.1.1) in minimum number of iterations, and 2) optimal communication cost, i.e, to reach the solution, the nodes should require the minimum amount of communication with the server. Our work in Chapter 4 satisfies both these requirements.

Our Contributions

- We propose a distributed first-order algorithm for the smooth, non-convex optimization problem described above. The algorithm is a non-trivial extension of the single-node SPIDER algorithm [52, 187] to the distributed case. We combine elements of periodic averaging [203] with SPIDER, while carefully choosing the update frequency. The proposed approach achieves the *optimal* computation complexity, while also achieving *linear speedup* with respect to the number of nodes N.
- The communication complexity achieved is also the best known in the literature. To the best of our knowledge, the only other work to achieve the same communication complexity is [200] (but with a higher computation cost).
- Compared to several existing approaches which require the samples across nodes to follow the same distribution, our approach is more general in the sense that the data distribution across nodes may be different (hence, our approach is applicable to the more general federated learning problem [98]).

In Chapter 4, the method proposed is a first-order algorithm. Therefore, we assume access to the IFO oracle (Definition 4.1.2), which returns the loss function value and the gradient vector at the specified point. However, in many situations, the analytical expressions of the objective functions are either expensive or infeasible to obtain. In such cases, we only have access to function values. This precludes using any first-order methods. As discussed next, in Chapter 5, we describe a method for stochastic nonconvex optimization, which only uses function values.

1.3 Zeroth-Order Stochastic Nonconvex Optimization

Derivative-free optimization (DFO) methods [37, 102] have become increasingly popular in recent years, owing to the advent of several machine learning applications where the analytical expressions of the objective functions are either expensive or infeasible to obtain. Some examples of such applications are black-box adversarial example generation in deep neural networks (DNNs), reinforcement learning, and control and management of time-varying networks with limited computational resources [118].

Zeroth-order (ZO) methods form a special class of DFO methods which can be seen as gradient-less versions of first-order (gradient-based) optimization methods [152, 136, 61]. ZO optimization involves approximating the full/stochastic gradient of the function using only the function values, and using this gradient estimator in the first-order (FO) optimization framework [137]. Advantages of ZO-methods, over conventional DFO methods like direct-search based methods [22, 18], and trust-region methods [36], are the ease of implementation and the convergence properties of these methods.

ZO algorithms often suffer from the high variance of gradient estimates. The existing estimators involve choosing between saving on the function query cost [61], and achieving higher accuracy of the gradient estimates [113]. A principled approach, which allows a trade-off between accuracy and function query cost is so far missing in the literature.

Our Contributions

We summarize our contributions below:

- We propose a novel function value based gradient estimator (we call HGE, hybrid gradient estimator), which takes advantage of both the query-efficient random gradient estimate and the variance-reduced coordinate-wise gradient estimate. We also develop a coordinate importance sampling method to further improve the variance of HGE under a fixed number of function queries.

- We propose a ZO hybrid gradient descent (ZO-HGD) optimization method with the aid of HGE, to solve a stochastic optimization problem, in the absence of a gradient oracle. We show that ZO-HGD is general since it covers ZO stochastic gradient descent (ZO-SGD) [61] and ZO stochastic coordinate descent (ZO-SCD) [113] as special cases. We provide a comprehensive theoretical analysis for the convergence of ZO-HGD across different optimization domains, showing that ZO-HGD is efficient in both iteration and function query complexities.
- We demonstrate the effectiveness of ZO-HGD through a real-world application to generating adversarial examples from a black-box deep neural network [28, 80]. We show that ZO-HGD outperforms ZO-SGD, ZO-SCD and ZO sign-based SGD methods in striking a graceful balance between query efficiency and attack success rate.

1.4 Organization of the Dissertation

The rest of this dissertation is organized as follows. In Chapter 2, we discuss the simultaneous self-localization and multi-target tracking (SCS-MTT) problem with passive sensing. In Chapter 3, we abstract the active sensing problem to a distributed online convex optimization problem, with time-varying (adversarial) constraints. In Chapter 4, we discuss the distributed stochastic nonconvex optimization problem, and the proposed first-order method. This is followed by the zeroth-order method to solve the stochastic nonconvex optimization problem in Chapter 5. Finally, in Chapter 6, we conclude with a summary of the major contributions and a discussion of some possible future directions we intend to pursue.

1.5 Bibliographic Note

The research work appearing in this dissertation has either already been published at various venues or is currently under review. Following is a list of the published/submitted papers, based on the work during the course of my research work at Syracuse University.

Work Included in the Dissertation

Journal Papers:

- **P. Sharma**, A. A. Saucan, D. J. Bucci Jr., and P. K. Varshney, "Decentralized Gaussian Filters for Cooperative Self-Localization and Multi-Target Tracking," *IEEE Trans. Signal Process.*, vol. 67, no. 22, pp. 5896–5911, 2019.

Conference Papers:

- **P. Sharma**, P. Khanduri, L. Shen, D. J. Bucci Jr., P. K. Varshney, "On Distributed Online Convex Optimization with Sublinear Dynamic Regret and Fit," accepted in *55th Asilomar Conference on Signals, Systems, and Computers*, 2021.
- **P. Sharma**, A. A. Saucan, D. J. Bucci Jr., and P. K. Varshney, "On Decentralized Self-localization and Tracking Under Measurement Origin Uncertainty," *22th International Conference on Information Fusion (FUSION)*, 2019.
- **P. Sharma**, A. A. Saucan, D. J. Bucci Jr., and P. K. Varshney, "On Self-Localization and Tracking with an Unknown Number of Targets," *52nd Asilomar Conference on Signals, Systems, and Computers*, 2018.

Under Review:

- P. Khanduri, **P. Sharma**, H. Yang, M. Hong, J. Liu, K. Rajawat, P. K. Varshney, "STEM: A Stochastic Two-Sided Momentum Algorithm Achieving Near-Optimal Sample and Communication Complexities for Federated Learning," (submitted to *NeurIPS*, 2021).
- **P. Sharma**, K. Xu, S. Liu, P.Y. Chen, X. Lin, P. K. Varshney, "Zeroth-Order Hybrid Gradient Descent: Towards A Principled Black-Box Optimization Framework," *arXiv:2012.11518.* (in revision with *IEEE Trans. Signal Process.*)

- **P. Sharma**, P. Khanduri, S. Bulusu, K. Rajawat, P. K. Varshney, “Parallel Restarted SPIDER: Communication Efficient Distributed Nonconvex Optimization with Optimal Computation Complexity,” *arXiv:1912.06036.* (submitted to *IEEE Trans. Signal Process.*)

Work not Included in the Dissertation

Journal Papers:

- **P. Sharma**, D. J. Bucci Jr., S. K. Brahma, and P. K. Varshney, “Communication Network Topology Inference via Transfer Entropy,” *IEEE Transactions on Network Science and Engineering*, vol. 7, no. 1, pp. 562–575, 2019.

Conference Papers:

- P. Khanduri, S. Bulusu, **P. Sharma**, and P K Varshney, “Byzantine SVRG with Distributed Batch Gradient Computations,” *11th OPT Workshop on Optimization for Machine Learning*, 2019.
- K. R. Varshney, P. Khanduri, **P. Sharma**, S. Zhang, and P. K. Varshney, “Why Interpretability in Machine Learning? An Answer Using Distributed Detection and Data Fusion Theory", *Third Annual Workshop on Human Interpretability in Machine Learning (WHI)* 2018.

Under Preparation:

- P. Khanduri, **P. Sharma**, S. Kafle, S. Bulusu, K. Rajawat, P. K. Varshney, “Distributed Stochastic Non-Convex Optimization: Momentum-Based Variance Reduction,” *arXiv preprint arXiv:2005.00224*.

CHAPTER 2

COOPERATIVE SELF-LOCALIZATION AND MULTI-TARGET TRACKING

In this chapter, we address the problem of Simultaneous Cooperative Self-localization and Multi-Target Tracking (SCS-MTT) under target data association uncertainty, i.e., the associations between measurements and target tracks are unknown.

2.1 Introduction

In GPS-denied environments and for agents with limited power, cooperative self-localization (CS) schemes that rely on inter-agent measurements become necessary. The objective of multi-target tracking (MTT) is the estimation of the trajectories of an unknown and time-varying number of targets. At any time instant, the sensors of an agent produce two kinds of measurements: inter-agent measurements - by observing other agents in proximity, and target measurements - by observing the targets that are within the measurement range of the agent. Due to the collaborative nature of CS, the inter-agent measurements are unambiguous, i.e., the identity of the neighboring agent is known for each inter-agent measurement. On the other hand, targets are non-cooperative and the measurement-to-target associations are not known. Clutter and missed detections also affect the target measurement set.

2.1.1 Related Work

In [192], CS is achieved via the SPAWN (Sum-Product Algorithm over a Wireless Network) method which relies on Belief Propagation (BP) [101, 198] for an efficient evaluation of marginal agent posterior densities. The factorization of a joint posterior density is leveraged by BP to efficiently compute marginals. Techniques that address MTT under association uncertainty can be classified as hard (finding the most likely association map) [154], [178] and soft or marginal-based (computing the target state marginal distribution over all measurement-to-target associations) [12]. MTT with multiple static agents is addressed in [135, 45, 134, 126] and [153].

In [127], an iterative BP message-passing method is proposed for simultaneous cooperative self-localization and target tracking. That is, the target measurements are used for CS in addition to the inter-agent measurements. State inference for both the agents and targets benefits from the exchange of probabilistic information between the CS and tracking tasks. However, the number of targets is assumed fixed and known in [127]. In addition, perfect association between measurements and targets is assumed known at each agent. These two assumptions are relaxed in [126], which employs the BP message passing approach of [190, 191] to compute marginal measurement-to-target association probabilities followed by marginal target densities. However, the algorithm is centralized and without sensor self-localization. For a general overview of BP-based methods for MTT, we refer the reader to [124]. Methods in both [127] and [126] rely on particle representations of agent and target probability densities and the BP messages. Particle filters (PF) [47] are methods for sequential estimation of the state vector in highly non-linear and/or non-Gaussian state systems. However, the computational and communication requirements of PF-based methods can be quite high.

A simultaneous CS-MTT (SCS-MTT) method for intelligent transportation systems was proposed in [23], where a MAP rule is employed to select the measurement-to-target associations with the highest marginal probabilities. Additionally, a decentralized single Gaussian implementation is given. In [57], a centralized BP method for agent localization is proposed, which only uses target measurements. The number of targets is assumed known. This is extended in [129], where the number of targets is unknown. The agents can exchange their location information as well as the target measurements to assist each other. A BP-based method for CS is proposed in [128] where association uncertainty is considered for the inter-agent measurements. BP-based methods for SCS-MTT under measurement and/or dynamic model uncertainties were proposed in [125] and [165].

2.1.2 Our Contributions

We propose an efficient, decentralized BP message passing based algorithm for simultaneous cooperative self-localization (of mobile agents) and multi-target tracking (SCS-MTT), under measurement-to-target association uncertainty, extending the work in [126] and [127]. As in [126], the data association problem is solved using an iterative BP-based approach [190]. Unlike [129], target measurements are not shared across agents.

- We propose a novel factoring of the joint posterior over agent and target states. Since the resulting factor graph has cycles, several message orderings are possible. The novelty of our contribution also lies in the ordering of messages that ensures a reduced amount of data exchange over the network.

- We compare the performance of a centralized, PF-based implementation of this approach, with a separate SPAWN and MTT method which first localizes the agents using SPAWN and subsequently performs MTT, similar to the method of [126].

- Next, we propose the more computationally efficient decentralized Gaussian-based (DG) and decentralized Gaussian-Mixture based (DGM) approaches, which we call DG-SCS-MTT and DGM-SCS-MTT filters respectively. For most kinematic tracking applications, the communication loads of DG-SCS-MTT and DGM-SCS-MTT are significantly smaller than the PF implementations.

- Our numerical results show that the performance of the decentralized algorithm that employs these techniques is similar to its centralized counterpart. Numerical experiments exhibit improved performance of both DG-SCS-MTT and DGM-SCS-MTT filters when compared to a separate SPAWN [192] (for localization) and MTT [126] approach.

The chapter is organized as follows. The system model and notation is discussed in Section 2.2, followed by the proposed SCS-MTT filter in Section 2.3. The decentralized Gaussian-mixture and single Gaussian filters are discussed in Sections 2.4 and 2.5 respectively. We present the simulation results in Section 2.6, followed by conclusion in Section 2.7.

2.2 System Model and notation

The notations, assumptions, and the resulting system model are presented in the following sections. The system model is essentially a combination of the system models in [126, 127]. Therefore, a lot of notation is also borrowed from [126, 127].

2.2.1 Notation

Agent and target states

For $a, b \in \mathbb{N}$, we denote with $[a : b]$ the set of positive integers $\{a, a+1, \cdots, b\}$. We denote the set of agents by $\mathcal{A} \triangleq [1 : S]$, and the set of Potential Targets (PTs) by $\mathcal{T} \triangleq [1 : K]$, where K is the maximum possible number of PTs present. The state of agent s at time n is denoted by $\mathbf{y}_{n,s} \in \mathbb{R}^{d_a}$.

PT $k \in \mathcal{T}$ is described at time n, by state $\mathbf{x}_{n,k} \in \mathbb{R}^{d_t}$ alongside a binary variable, $r_{n,k}$, that indicates its existence at time n ($r_{n,k} = 1$ for presence, 0 for absence). The time-varying number of targets is accounted for via the variables $\{r_{n,k}\}$ while target existence can be inferred from the probability of existence $\Pr(r_{n,k} = 1)$. We further define the joint state vector of all the PTs at time n, $\mathbf{x}_n \triangleq \left[\mathbf{x}_{n,1}^T, \cdots, \mathbf{x}_{n,K}^T\right]^T$, and the across-time vector, $\mathbf{x} \triangleq \left[\mathbf{x}_0^T, \cdots, \mathbf{x}_n^T\right]^T$. In an analogous manner, we introduce the joint vectors at time n, $\mathbf{r}_n$ and $\mathbf{y}_n$, and across-time vectors $\mathbf{r}$ and $\mathbf{y}$. Let

$\tilde{\mathbf{x}}_{n,k} = [\mathbf{x}_{n,k}^T, r_{n,k}]^T$ be the *augmented* state vector for PT k at time n. We also define $\tilde{\mathbf{x}}_n = [\mathbf{x}_n^T, \mathbf{r}_n^T]^T$ and $\tilde{\mathbf{x}} = [\mathbf{x}^T, \mathbf{r}^T]^T$. In addition, we introduce the notation $\int(\cdot)d\tilde{\mathbf{x}}_{n,k} \triangleq \sum_{r_{n,k}\in\{0,1\}} \int(\cdot)d\mathbf{x}_{n,k}$. If $f(\tilde{\mathbf{x}}_{n,k}) \equiv f(\mathbf{x}_{n,k}, r_{n,k})$ is the augmented state probability density for PT k, then the probability of existence at time n is $P_{n,k}^e \triangleq \Pr(r_{n,k} = 1) = \int f(\mathbf{x}_{n,k}, 1)d\mathbf{x}_{n,k}$.

Inter-agent Measurements

For agent s, let $\mathcal{A}_{n,s} \subseteq \mathcal{A}$ denote the set of its neighboring agents, i.e., agents that are within its inter-agent measurement range, at time n. Let $\mathbf{w}_{s,\ell;n} \in \mathbb{R}^{d_w}$ be the measurement that agent s makes with respect to the neighboring agent $\ell \in \mathcal{A}_{n,s}$, at time n. The inter-agent measurement likelihood is denoted as $f(\mathbf{w}_{s,\ell;n}|\mathbf{y}_{s,n}, \mathbf{y}_{\ell,n})$. The stacked vector of the inter-agent measurements at agent s at time n is denoted by $\mathbf{w}_{s;n}$. Let $\mathbf{w}_n \triangleq \left[\mathbf{w}_{1;n}^T, \cdots, \mathbf{w}_{S;n}^T\right]^T$ and $\mathbf{w} \triangleq \left[\mathbf{w}_1^T, \cdots, \mathbf{w}_n^T\right]^T$.

Target Measurements

Agent s observes a subset $\mathcal{T}_{n,s} \subset \mathcal{T}$ of PTs that are within its target-measurement range. We also define the set of agents observing PT k at time n as $\mathcal{A}_{n,k} = \{s \in \mathcal{A} \colon k \in \mathcal{T}_{n,s}\}$. Since targets are non-cooperative, the collection of target measurements suffers from missed detections, clutter and association uncertainty. Let M_n^s be the number of target measurements gathered by agent s at time n. Let $\mathbf{z}_n^s \triangleq [(\mathbf{z}_{n,1}^s)^T, \cdots, (\mathbf{z}_{n,M_n^s}^s)^T]^T$ be an arbitrarily ordered collection of these measurements, with $\mathbf{z}_{n,m}^s \in \mathcal{R}^{d_z}\ \forall\ m \in \mathcal{M}_{n,s} \triangleq [1 : M_n^s]$. Furthermore, let $\mathbf{z}_n \triangleq [(\mathbf{z}_n^1)^T, \cdots, (\mathbf{z}_n^S)^T]^T$, $\mathbf{z} \triangleq [(\mathbf{z}_1)^T, \cdots, (\mathbf{z}_n)^T]^T$, $\mathbf{m}_n \triangleq [M_n^1 \cdots M_n^s]^T$ and $\mathbf{m} \triangleq [\mathbf{m}_1^T, \cdots, \mathbf{m}_n^T]^T$. The likelihood of measurement $\mathbf{z}_{n,m}^s$ made by agent s, if it corresponds to PT k is $f(\mathbf{z}_{n,m}^s|\mathbf{y}_{n,s}, \mathbf{x}_{n,k})$. A PT $\mathbf{x}_{n,k}$ is detected by agent s with probability $P_D^s(\mathbf{x}_{n,k})$. Finally, $\mathbf{z}_n^s$ also contains clutter measurements, independently sampled from a Poisson point process. The rate of clutter points is λ_n^s and their probability distribution is $f_{n,s}^{FA}(\mathbf{z})$, for measurement $\mathbf{z}$.

2.2.2 Assumptions

Our assumptions in this work stem from [126, 127] and are provided in the following:

(A1) Agent and target states are *a priori* independent and evolve independently in time according to Markov processes.

(A2) The communication graph $\mathcal{G}_n$ that spans the decentralized network of agents is connected at all times and the communication links between the agents are bidirectional.

(A3) Given the current agent states $\mathbf{y}_n$ and augmented target states $\tilde{\mathbf{x}}_n$, the measurements $\mathbf{w}_n$ and $\mathbf{z}_n$ are conditionally independent of past $(\mathbf{w}_{1:n-1}, \mathbf{z}_{1:n-1})$ and future $(\mathbf{w}_{n+1:\infty}, \mathbf{z}_{n+1:\infty})$ measurements, i.e., $f(\mathbf{w}_n, \mathbf{z}_n | \mathbf{w}_{1:n-1}, \mathbf{z}_{1:n-1}, \mathbf{w}_{n+1:\infty}, \mathbf{z}_{n+1:\infty}, \mathbf{y}_n, \tilde{\mathbf{x}}_n) = f(\mathbf{w}_n, \mathbf{z}_n | \mathbf{y}_n, \tilde{\mathbf{x}}_n)$.

(A4) Current agent and target states $\mathbf{y}_n$, $\tilde{\mathbf{x}}_n$, are conditionally independent of all the past measurements $\mathbf{w}_{0:n-1}, \mathbf{z}_{0:n-1}$ given the previous states $\mathbf{y}_{n-1}$ and $\tilde{\mathbf{x}}_{n-1}$.

(A5) Given $\mathbf{y}_n$, the inter-agent measurements $\mathbf{w}_{s,\ell;n}$ and $\mathbf{w}_{s',\ell';n}$ are conditionally independent if $(s, \ell) \neq (s', \ell')$.

(A6) Given $\mathbf{y}_n$ and $\tilde{\mathbf{x}}_n$, the target and agent measurements $\mathbf{z}_n$ and $\mathbf{w}_n$ are conditionally independent and furthermore $f(\mathbf{w}_n, \mathbf{z}_n | \mathbf{y}_n, \tilde{\mathbf{x}}_n) = f(\mathbf{w}_n | \mathbf{y}_n) f(\mathbf{z}_n | \mathbf{y}_n, \tilde{\mathbf{x}}_n)$.

(A7) At any time n, an existing target can generate at most one measurement at any agent, and any target measurement at an agent is generated by at most one existing target [13, 122]. The detection process is independent for different targets and across different agents.

(A8) Target measurements $\mathbf{z}_n$ suffer from origin uncertainty, i.e., the associations between the individual measurements of $\mathbf{z}_n$ and the PT $\tilde{\mathbf{x}}_n$ are unknown. Some measurements are due to clutter and some PTs are not detected.

(A9) Inter-agent measurements do not suffer from origin uncertainty. Agent s knows that measurement $\mathbf{w}_{s,\ell;n}$ originates from agent $\ell \in \mathcal{A}_{n,s}$ and the inter-agent measurement links are bidirectional, i.e., $\ell \in \mathcal{A}_{n,s} \Leftrightarrow s \in \mathcal{A}_{n,\ell}$ for $s, \ell \in \mathcal{S}$.

(A10) Each agent knows its own prior and dynamic model and the priors and dynamic models of all PTs. All agents have synchronized internal clocks.

$r_{n-1,k}$	$r_{n,k}$	$f\left(\mathbf{x}_{n,k}, r_{n,k} \mid \mathbf{x}_{n-1,k}, r_{n-1,k}\right)$
0	1	$P^B_{n,k} f_b\left(\mathbf{x}_{n,k}\right)$
0	0	$\left(1 - P^B_{n,k}\right) f_D\left(\mathbf{x}_{n,k}\right)$
1	0	$\left(1 - P^S_{n,k}\right) f_D\left(\mathbf{x}_{n,k}\right)$
1	1	$P^S_{n,k}(\mathbf{x}_{n,k}) f\left(\mathbf{x}_{n,k} \mid \mathbf{x}_{n-1,k}\right)$

Table 2.1: State transition kernels for different values of target existence indicators. The function $f_D(\cdot)$ is a dummy pdf [126].

The SPAWN approach [192] addresses the problem of self-localization without MTT. In [127], a perfect knowledge of the target-to-measurement associations is assumed. Also, the number of targets is known and time-invariant. In [126], these assumptions are removed, but the agents have perfect knowledge of their positions, i.e., fixed sensors case. In this work, we extend [127] by relaxing the assumption of known origins of target measurements, and accommodate an unknown, time-varying number of targets as in [126].

2.2.3 System Model

Under assumption (A1), we denote the agent transition densities with $f\left(\mathbf{y}_{n,s} \mid \mathbf{y}_{n-1,s}\right) \forall\, s$. For PT k, the transition kernel $f(\tilde{\mathbf{x}}_{n,k} \mid \tilde{\mathbf{x}}_{n-1,k}) \equiv f(\mathbf{x}_{n,k}, r_{n,k} \mid \mathbf{x}_{n-1,k}, r_{n-1,k})$ accounts for target birth, death and evolution (in case of survival) as listed in Table 2.1. The dynamic kernel is a function of the indicator variable $r_{n,k}$. Here, $P^B_{n,k}$ is the birth probability, $f_b\left(\mathbf{x}_{n,k}\right)$ is the birth pdf, $P^S_{n,k}(\cdot)$ is the survival probability, and $f\left(\mathbf{x}_{n,k} \mid \mathbf{x}_{n-1,k}\right)$ is the state transition pdf. Under assumption (A1), the joint pdf of $[\mathbf{y}^T, \mathbf{x}^T, \mathbf{r}^T]^T$ given by

$$f(\mathbf{y}, \underbrace{\mathbf{x}, \mathbf{r}}_{\tilde{\mathbf{x}}}) = \prod_{s=1}^{S} f\left(\mathbf{y}_{0,s}\right) \prod_{n'=1}^{n} f\left(\mathbf{y}_{n',s} \mid \mathbf{y}_{n'-1,s}\right)$$
$$\times \prod_{k=1}^{K} f\left(\tilde{\mathbf{x}}_{0,k}\right) \prod_{n'=1}^{n} f\left(\tilde{\mathbf{x}}_{n',k} \mid \tilde{\mathbf{x}}_{n'-1,k}\right). \tag{2.1}$$

To solve the data association problem of assumption (A8), i.e., finding the associations between measurements and PTs, we use the redundant formulation of association variables proposed in [190]. *Target oriented association variables* define the PT-measurement associations at sensor s at time n:

$$a_{n,k}^s \triangleq \begin{cases} m \in \mathcal{M}_{n,s} & \text{PT } k \text{ generated } \mathbf{z}_{n,m}^s \text{ at time } n, \\ 0 & \text{PT } k \text{ is not detected at time } n. \end{cases} \tag{2.2}$$

The *measurement-oriented association variables* are

$$b_{n,m}^s \triangleq \begin{cases} k \in \mathcal{K} & \text{PT } k \text{ generated } \mathbf{z}_{n,m}^s \text{ at time } n, \\ 0 & \mathbf{z}_{n,m}^s \text{ is a clutter measurement.} \end{cases} \tag{2.3}$$

Further, we define stacked vectors of association variables: $\mathbf{a}_n^s \triangleq [a_{n,1}^s, \cdots, a_{n,K}^s]^T$, $\mathbf{a}_n \triangleq [(\mathbf{a}_n^1)^T, \cdots, (\mathbf{a}_n^S)^T]^T$, $\mathbf{a} \triangleq [\mathbf{a}_1^T, \cdots, \mathbf{a}_n^T]^T$, and $\mathbf{b}_n^s \triangleq [b_{n,1}^s, \cdots, b_{n,M_n^s}^s]^T$, $\mathbf{b}_n \triangleq [(\mathbf{b}_n^1)^T, \cdots, (\mathbf{b}_n^S)^T]^T$, $\mathbf{b} \triangleq [\mathbf{b}_1^T, \cdots, \mathbf{b}_n^T]^T$. Note that $\mathbf{a}_n^s$ and $\mathbf{b}_n^s$ are redundant, meaning one can be derived from the other. We define the indicator function $\Psi(a_{n,k}^s, b_{n,m}^s)$

$$\Psi\left(a_{n,k}^s, b_{n,m}^s\right) \triangleq \begin{cases} 0, & a_{n,k}^s = m \text{ and } b_{n,m}^s \neq k \text{ OR } a_{n,k}^s \neq m \text{ and } b_{n,m}^s = k \\ 1, & \text{otherwise,} \end{cases} \tag{2.4}$$

where $\{\Psi(a_{n,k}^s, b_{n,m}^s)\}_{k,m}$ collectively enforce the association variables $\mathbf{a}_n^s$ and $\mathbf{b}_n^s$ to be consistent [190]. Under the assumptions (A3-A9), the joint measurement likelihood becomes

$$\begin{aligned} f\left(\mathbf{z}, \mathbf{w} | \mathbf{y}, \tilde{\mathbf{x}}, \mathbf{a}, \mathbf{m}\right) &= f\left(\mathbf{w} | \mathbf{y}\right) f\left(\mathbf{z} | \mathbf{y}, \tilde{\mathbf{x}}, \mathbf{a}, \mathbf{m}\right) \qquad (2.5) \\ &= \prod_{n'} \prod_{s} f\left(\mathbf{z}_{n'}^s \middle| \mathbf{y}_{n',s}, \tilde{\mathbf{x}}_{n'}, \mathbf{a}_{n'}^s, M_{n'}^s\right) \prod_{\ell \in \mathcal{A}_{n',s}} f\left(\mathbf{w}_{s,\ell;n'} \middle| \mathbf{y}_{n',s}, \mathbf{y}_{n',\ell}\right). \end{aligned}$$

Since under assumption (A7) each measurement is caused by a target or clutter, the target measurement likelihood further factorizes as

$$f(\mathbf{z}_n^s | \mathbf{y}_{n,s}, \tilde{\mathbf{x}}_n, \mathbf{a}_n^s, M_n^s) = \underbrace{\prod\nolimits_{m: b_{n,m}^s = 0} f_{n,s}^{FA}\left(\mathbf{z}_{n,m}^s\right)}_{\text{clutter measurements}} \times \underbrace{\prod\nolimits_{\substack{m: b_{n,m}^s = k \\ \text{and } r_{n,k} = 1}} f\left(\mathbf{z}_{n,m}^s \middle| \mathbf{x}_{n,k}, \mathbf{y}_{n,s}\right)}_{\text{Measurements from existing targets}} \tag{2.6}$$

which we can rewrite as follows

$$f(\mathbf{z}_n^s|\mathbf{y}_{n,s},\tilde{\mathbf{x}}_n,\mathbf{a}_n^s,M_n^s)\propto\prod_{k\in\mathcal{K}}g_k(\tilde{\mathbf{x}}_{n,k},\mathbf{y}_{n,s},a_{n,k}^s;\mathbf{z}_n^s)\tag{2.7}$$

where the normalization factor depends only on $f_{n,s}^{FA}(\cdot)$, hence only on the measurements $\mathbf{z}_n^s$. For $m\in\ \mathcal{M}_n^s$

$$g_k\left(\mathbf{x}_{n,k},r_{n,k}=1,\mathbf{y}_{n,s},a_{n,k}^s=m;\mathbf{z}_n^s\right)=\frac{f\left(\mathbf{z}_{n,m}^s|\mathbf{x}_{n,k},\mathbf{y}_{n,s}\right)}{f_{n,s}^{FA}\left(\mathbf{z}_{n,m}^s\right)}\tag{2.8}$$

while $g_k(\mathbf{x}_{n,k},r_{n,k}=1,\mathbf{y}_{n,s},a_{n,k}^s=0;\mathbf{z}_n^s)=1$. For absent targets ($r_{n,k}=0$), $g_k(\mathbf{x}_{n,k},r_{n,k}=0,\mathbf{y}_{n,s},a_{n,k}^s=m;\mathbf{z}_n^s)=1,\ \forall\, m=0,\ldots,M_n^s$.

The association variables $\mathbf{a}$, $\mathbf{b}$ and the number of measurements $\mathbf{m}$ are assumed conditionally independent across time and across agents, given the states of agents and targets. Thus, the joint distribution of association variables and the number of measurements, factorizes as

$$\begin{aligned}p(\mathbf{a},\mathbf{b},\mathbf{m}|\mathbf{y},\tilde{\mathbf{x}})&=\prod_{n'=1}^{n}\prod_{s=1}^{S}p(a_{n'}^s,b_{n'}^s,M_{n'}^s|\mathbf{y}_{n',s},\tilde{\mathbf{x}}_{n'})\\&\propto\prod_{n'=1}^{n}\prod_{s=1}^{S}\prod_{k=1}^{K}h_k(\tilde{\mathbf{x}}_{n',k},a_{n',k}^s,\mathbf{y}_{n',s})\prod_{m=1}^{M_{n'}^s}\Psi\left(a_{n',k}^s,b_{n',m}^s\right),\end{aligned}\tag{2.9}$$

where the normalization constant depends only on the clutter rate λ_n^s and the number of measurements $\mathbf{m}$ [79]. The term $h_k(\cdot)$ is defined as

$$h_k(\mathbf{x}_{n,k},1,a_{n,k}^s,\mathbf{y}_{n,s})=\begin{cases}\frac{P_D^s\left(\mathbf{x}_{n,k}\right)}{\lambda_n^s}, & \text{if } a_{n,k}^s\in\mathcal{M}_n^s\\ 1-P_D^s\left(\mathbf{x}_{n,k}\right), & \text{if } a_{n,k}^s=0,\end{cases}\tag{2.10}$$

and $h_k(\mathbf{x}_{n,k},r_{n,k}=0,a_{n,k}^s,\mathbf{y}_{n,s})=1(a_{n,k}^s)$ where $1(a)=1$ if $a=0$ and $1(a)=0$ otherwise. $\Psi\left(\cdot,\cdot\right)$ is defined in (2.4).

2.3 The SCS-MTT filter

We perform agent and target state inference using the marginal posterior densities. These are obtained from the joint posterior density using the following factorization, which is derived by extending analogous results in [126] and [127].

Lemma 2.3.1. *The joint posterior density of all the agent and PT states, given inter-agent and target measurements, up to time n, admits the factorization*

$$\begin{aligned} f(\mathbf{y}, \tilde{\mathbf{x}}, \mathbf{a}, \mathbf{b}|\mathbf{z}, \mathbf{w}) \propto & \left[\prod_{k=1}^{K} f\left(\tilde{\mathbf{x}}_{0,k}\right) \prod_{n'=1}^{n} f\left(\tilde{\mathbf{x}}_{n',k}|\tilde{\mathbf{x}}_{n'-1,k}\right)\right] \\ & \times \prod_{s=1}^{S} \Bigg\{ f\left(\mathbf{y}_{0,s}\right) \prod_{n'=1}^{n} \Bigg[f\left(\mathbf{y}_{n',s}|\mathbf{y}_{n'-1,s}\right) \Bigg(\prod_{\ell \in \mathcal{A}_{n',s}} f\left(\mathbf{w}_{s,\ell;n'}|\mathbf{y}_{n',s}, \mathbf{y}_{n',\ell}\right) \Bigg) \\ & \times \prod_{k=1}^{K} \Bigg(v_k^s(\tilde{\mathbf{x}}_{n',k}, a_{n',k}^s, \mathbf{y}_{n',s}; \mathbf{z}_{n'}^s) \prod_{m=1}^{M_{n'}^s} \Psi\left(a_{n',k}^s, b_{n',m}^s\right) \Bigg) \Bigg] \Bigg\} \end{aligned} \tag{2.11}$$

where, $v_k^s(\tilde{\mathbf{x}}_{n',k}, a_{n',k}^s, \mathbf{y}_{n',s}; \mathbf{z}_{n'}^s)$

$$\triangleq \begin{cases} \frac{P_D^s(\mathbf{x}_{n',k}) f\left(\mathbf{z}_{n',m}^s | \mathbf{x}_{n',k}, \mathbf{y}_{n',s}\right)}{\lambda_{n'}^s f_{n',s}^{FA}\left(\mathbf{z}_{n',m}^s\right)}, & \text{if } a_{n',k}^s = m \neq 0 \ \text{ AND } \ r_{n',k} = 1 \\ 1 - P_D^s(\mathbf{x}_{n',k}), & \text{if } a_{n',k}^s = 0, \ r_{n',k} = 1 \\ 1, & \text{if } a_{n',k}^s = 0, \ r_{n',k} = 0 \\ 0, & \text{otherwise.} \end{cases}$$

PROOF: Note that the number of target measurements M_n^s becomes fixed when conditioning on $\mathbf{z}_n^s$. Applying Bayes' rule

$$\begin{aligned} f\left(\mathbf{y}, \mathbf{x}, \mathbf{r}, \mathbf{a}, \mathbf{b}|\mathbf{z}, \mathbf{w}\right) &= f\left(\mathbf{y}, \mathbf{x}, \mathbf{r}, \mathbf{a}, \mathbf{b}, \mathbf{m}|\mathbf{z}, \mathbf{w}\right) \\ &\propto \underbrace{f\left(\mathbf{y}, \mathbf{x}, \mathbf{r}\right)}_{(i)} \cdot \underbrace{p\left(\mathbf{a}, \mathbf{b}, \mathbf{m}|\mathbf{y}, \mathbf{x}, \mathbf{r}\right)}_{(ii)} \cdot \underbrace{f\left(\mathbf{z}, \mathbf{w}|\mathbf{y}, \mathbf{x}, \mathbf{r}, \mathbf{a}, \mathbf{b}, \mathbf{m}\right)}_{(iii)}. \end{aligned} \tag{2.12}$$

where (i) represents the joint distribution (2.1) of the agent and target states up to time n; (ii)

represents the data association and detection of the targets given the agent and augmented target states (2.9)-(2.10); and (iii) represents the joint measurement likelihood, given the states of all the agents and targets, and their data association relationships (2.5)-(2.8). Substituting the expressions for $(i) - (iii)$ into (2.12), and defining $v(\tilde{\mathbf{x}}_{n,k}, a^s_{n,k}, \mathbf{y}_{n,s}; \mathbf{z}^s_n) \triangleq h_k(\tilde{\mathbf{x}}_{n,k}, a^s_{n,k}, \mathbf{y}_{n,s}) \times g_k(\tilde{\mathbf{x}}_{n,k}, a^s_{n,k}, \mathbf{y}_{n,s}; \mathbf{z}^s_n)$, we obtain (2.11). ■

The marginals associated with (2.11) can be efficiently computed via BP algorithms that exploit the structure embedded in its factorization. The factor graph corresponding to (2.11) for a fixed time step n is shown in Figure 2.1[1]. The factor nodes are shown as rectangles, while the variable nodes are shown as ovals. Time index n has been omitted from notations and messages passed between nodes are represented as annotations on each link. Following are the factor nodes: for PT k, $f_k \triangleq f(\tilde{\mathbf{x}}_{n,k}|\tilde{\mathbf{x}}_{n-1,k})$; for agent s, $g_s \triangleq f(\mathbf{y}_{n,s}|\mathbf{y}_{n-1,s})$, $v_k \triangleq v(\mathbf{x}_{n,k}, r_{n,k}, a^s_{n,k}, \mathbf{y}_{n,s}; \mathbf{z}^s_n)$; for agent s measuring another agent ℓ, $f_{s,\ell} \triangleq f(\mathbf{w}_{s,\ell;n}|\mathbf{y}_{n,s}, \mathbf{y}_{n,\ell})$, $\Psi_{k,m} \triangleq \Psi(a^s_{n,k}, b^s_{n,m})$. The numbered circles $1 - 6$ in the block corresponding to sensor s demonstrate the order in which messages are computed. The beliefs broadcast by the agents at the beginning of each outer loop are shown by arrows b_s and b_ℓ coming out of agent state nodes $\mathbf{y}_s$ and $\mathbf{y}_\ell$ respectively.

2.3.1 The SCS-MTT filter: the BP message passing scheme

In this section, we describe our proposed message passing algorithm for inferring the marginal densities of targets $b(\tilde{\mathbf{x}}_{n,k})$ and agents $b(\mathbf{y}_{n,s})$ at time n corresponding to the joint density of (2.11). For an introduction to BP, the reader is directed to [198]. Since the factor graph of Figure 2.1 has cycles, multiple message ordering schemes exist. Similar to [126], we assume that: (i) messages are not sent backward in time, and (ii) marginal association probabilities are evaluated via BP at each agent.

At each beginning of time step n, using the agent belief $b(\mathbf{y}_{n-1,s})$ from the previous time step,

---

[1] This factor graph represents a combination of the factor graph containing the agent and target states from [127, Figure 2] and the factor graph corresponding to target measurement uncertainty of [126].

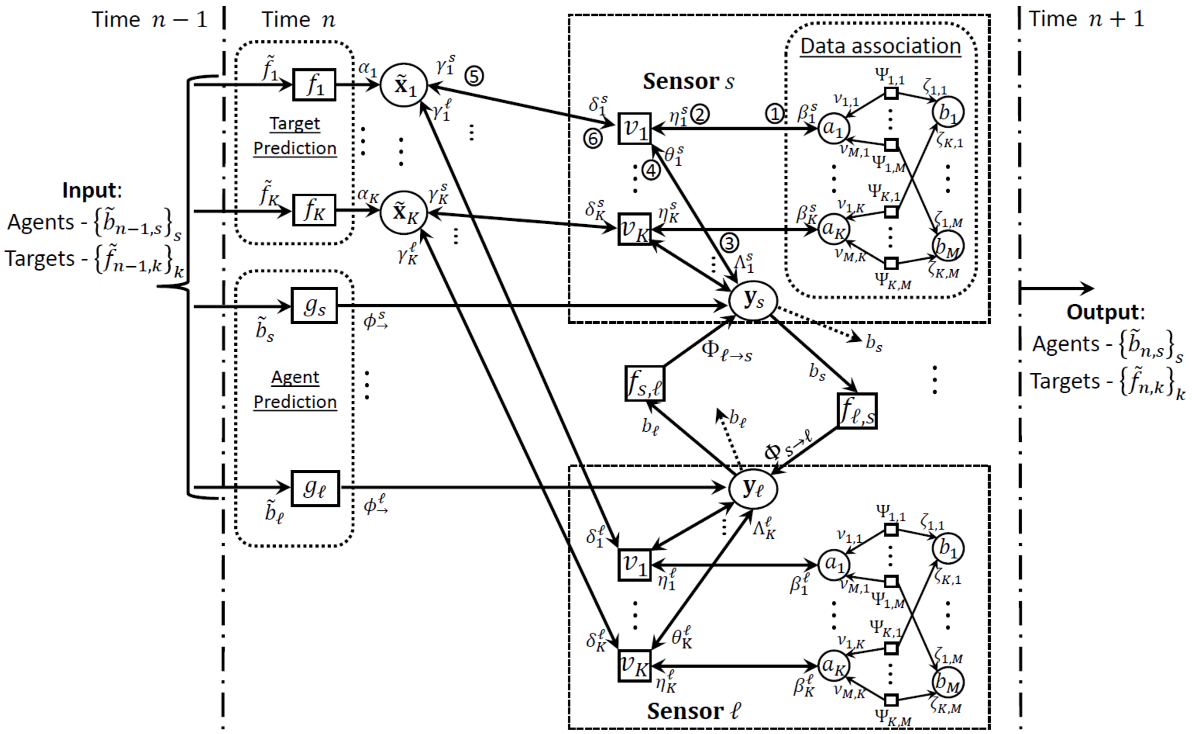


Fig. 2.1: Factor graph representing the factorization of (2.11), for one time step.

agent s computes the *prediction message* $\phi_{\rightarrow n}(\mathbf{y}_{n,s})$ given by

$$\phi_{\rightarrow n}(\mathbf{y}_{n,s}) = \int f(\mathbf{y}_{n,s}|\mathbf{y}_{n-1,s})b(\mathbf{y}_{n-1,s})d\mathbf{y}_{n-1,s}. \tag{2.13}$$

Additionally, each agent also computes locally, the predicted messages $\alpha_{\rightarrow n}(\mathbf{x}_{n,k}, r_{n,k})$ for all PTs $k \in \mathcal{T}$, using the target state beliefs at the previous time step $\tilde{b}(\tilde{\mathbf{x}}_{n-1,k})$

$$\alpha_{\rightarrow n}(\tilde{\mathbf{x}}_{n,k}) = \int f(\tilde{\mathbf{x}}_{n,k}|\tilde{\mathbf{x}}_{n-1,k})b(\tilde{\mathbf{x}}_{n-1,k})\, d\tilde{\mathbf{x}}_{n-1,k}, \tag{2.14}$$

where the transition density $f(\tilde{\mathbf{x}}_{n,k}|\tilde{\mathbf{x}}_{n-1,k})$ (Table 2.1) incorporates target birth and death in addition to its kinematic model. Note that (2.13)-(2.14) correspond to the Chapman–Kolmogorov equations in the prediction step of the recursive Bayesian filters and incorporate the agent and target dynamic models.

Before the start of the message passing scheme, the agent beliefs at the current time-step $b(\mathbf{y}_{n,s})$ are initialized with the predicted beliefs $\phi_{\rightarrow n}(\mathbf{y}_{n,s})$, $\forall\ s \in \mathcal{S}$. Synchronously and in parallel, the agents run the iterative message passing scheme, referred to as the outer BP loop in Algorithm 1. Each agent s executes the loop P times. Subsequently, we present the BP outer-loop messages in the order in which they are evaluated in Algorithm 1 while also indicating the corresponding nodes and messages in the factor graph of Figure 2.1.

At the beginning of an outer loop, each agent broadcasts its belief $b(\mathbf{y}_{n,s})$ to its neighboring agents $\ell \in \mathcal{A}_{n,s}$ (see the arrows coming out of $\mathbf{y}_s, \mathbf{y}_\ell$ in Figure 2.1). The time subscript will be dropped for the rest of this section since all subsequent messages only involve variables at time n. We shall use the term "target" generically, and "PT" when referring to a specific potential target k. We present the expressions of different BP messages involving agent s.

The current beliefs $b(\mathbf{y}_\ell)$ of neighboring agents $\ell \in \mathcal{A}_{n,s}$ are broadcast, and received at agent s. Next, agent s computes the *likelihood* messages $\Phi_{\ell\to s}$ (line 6, Algorithm 1) using

$$\Phi_{\ell\to s}(\mathbf{y}_s) = \int f(\mathbf{w}_{s,\ell}|\mathbf{y}_s, \mathbf{y}_\ell) b(\mathbf{y}_\ell) d\mathbf{y}_\ell. \tag{2.15}$$

By marginalizing over the state of agent ℓ, the message $\Phi_{\ell\to s}$ represents the likelihood of agent s for the measurement $\mathbf{w}_{s,\ell}$ taken by agent s with respect to agent ℓ. This is followed by locally computing the single-target association weights $\beta_k^s(a_k^s = m)$ between the local measurements $\mathbf{z}_n^s$ (at agent s) and the target set $\mathcal{K}$. For all the PTs $k \in \mathcal{K}$ and $m \in \{0, \cdots, M^s\}$, these weights $\beta_k^s(a_k^s = m)$ (line 9 in Algorithm 1 and ① in Figure 2.1) are given as

$$\begin{aligned}\beta_k^s(a_k^s = m) &= \int v_k^s(\tilde{\mathbf{x}}_k, a_k^s, \mathbf{y}_s; \mathbf{z}^s)\delta_k^s(\tilde{\mathbf{x}}_k)\theta_k^s(\mathbf{y}_s) d\tilde{\mathbf{x}}_k d\mathbf{y}_s \\ &= \begin{cases} \frac{\int P_D^s(\mathbf{x}_k) f(\mathbf{z}_m^s|\mathbf{y}_s, \mathbf{x}_k)\delta_k^s(\mathbf{x}_k, 1)\theta_k^s(\mathbf{y}_s) d\mathbf{y}_s d\mathbf{x}_k}{\lambda^s f_s^{\mathrm{FA}}(\mathbf{z}_m^s)}, & \text{if } m \neq 0 \\ 1 - \int P_D^s(\mathbf{x}_k)\delta_k^s(\mathbf{x}_k, 1) d\mathbf{x}_k, & \text{if } m = 0, \end{cases}\end{aligned} \tag{2.16}$$

for $k \in \mathcal{T}_s$, and $\beta_k^s(m) = 0$ for $k \notin \mathcal{T}_s$ and $\forall\ m$. At the first iteration of the outer BP-loop, we initialize the messages as $\delta_k^s(\tilde{\mathbf{x}}_k) = \alpha_{\to n}(\tilde{\mathbf{x}}_k)$ and $\theta_k^s(\mathbf{y}_s) = \phi_{\to n}(\mathbf{y}_s)$. In other words, the association weights in the first outer loop iteration are estimated by marginalizing with respect to the predicted agent and target densities.

Algorithm 1 SCS-MTT outer BP-loop - in parallel $\forall\, s \in \mathcal{S}$

```
 1: Input: Predicted beliefs $\phi_{\to n}(\mathbf{y}_s)$, $\{\alpha_{\to n}(\tilde{\mathbf{x}}_k)\}_{k\in\mathcal{T}}$
 2: Initialize: $b(\mathbf{y}_s) \leftarrow \phi_{\to n}(\mathbf{y}_s)$, $\theta_k^s(\mathbf{y}_s) \leftarrow b(\mathbf{y}_s)$ and $\delta_k^s(\tilde{\mathbf{x}}_k) \leftarrow \alpha_{\to n}(\tilde{\mathbf{x}}_k)$, $\forall\, k \in \mathcal{T}_s$
 3: for $p \leftarrow 1$ to $P$ do                                   (Outer BP iterations)
 4:     Broadcast $b(\mathbf{y_s})$ and receive $b(\mathbf{y}_\ell)$ $\forall\, \ell \in \mathcal{A}_s$
 5:     for all $\ell \in \mathcal{A}_s$ do
 6:         Compute $\Phi_{\ell\to s}(\mathbf{y}_s)$ via (2.15)
 7:     end for
 8:     for all $k \in \mathcal{T}_s$ do
 9:         Compute $\beta_k^s(a_k^s)$ via (2.16)
10:         Compute $\eta_k^s(a_k^s)$ as in [190]                       (Data Association)
11:         Compute $\Lambda_k^s(\mathbf{y}_s)$ via (2.17)
12:     end for
13:     Update agent belief $b_s(\mathbf{y}_s)$ via (2.18)
14:     for all $k \in \mathcal{T}_s$ do
15:         Compute $\theta_k^s(\mathbf{y}_s)$ via (2.19)
16:     end for
17:     for all $k \in \mathcal{T}$ do
18:         Compute $\gamma_k^s(\mathbf{x}_k, r_k)$ via (2.20) if $k \in \mathcal{T}_s$
19:         Set $\gamma_k^s(\mathbf{x}_k, r_k) = 1$ if $k \notin \mathcal{T}_s$
20:         Network consensus to update $b(\mathbf{x}_k, r_k)$ via (2.21)
21:         Compute $\delta_k^s(\tilde{\mathbf{x}}_k)$ via (2.22) if $k \in \mathcal{T}_s$ and $p \neq P$
22:     end for
23: end for
24: Return $b(\mathbf{y}_s)$ and $b(\tilde{\mathbf{x}}_k)$ $\forall\, k \in \mathcal{T}$.
```

Next, these weights $\{\beta_k^s(m)\}$ are used to evaluate the messages $\{\eta_k^s(m)\}$ (line 10 in Algorithm 1 and ② in Figure 2.1). This is achieved by a second, inner BP loop which involves message exchanges between the local association variables $\mathbf{a}_n^s$ and $\mathbf{b}_n^s$ of agent s [190]. Similar to other track-oriented marginal filters such as the JPDAF [12], the SCS-MTT filter evaluates the single–target association weights β_k^s followed by an efficient BP evaluation of the marginal association probabilities. Additionally in SCS-MTT, the uncertainty in the position of agent s is accounted for in β_k^s by marginalizing over the message θ_k^s.

The η_k^s messages are subsequently used to evaluate the likelihood messages $\Lambda_k^s(\mathbf{y}_s)$ (line 11 in Algorithm 1 and ③ in Figure 2.1), sent from the factor node v_k^s of each PT $k \in \mathcal{T}_s$, to the agent state node $\mathbf{y}_s$.

$$\begin{aligned}\Lambda_k^s(\mathbf{y}_s) &= \sum_{m=0}^{M^s} \int v_k^s(\tilde{\mathbf{x}}_k, m, \mathbf{y}_s; \mathbf{z}_m^s)\eta_k^s(m)\delta_k^s(\tilde{\mathbf{x}}_k)d\tilde{\mathbf{x}}_k \\ &= \sum_{m=1}^{M^s} \frac{\eta_k^s(m)}{\lambda^s f_s^{\mathrm{FA}}(\mathbf{z}_m^s)} \int P_D^s(\mathbf{x}_k)\delta_k^s(\mathbf{x}_k, 1)f(\mathbf{z}_m^s|\mathbf{x}_k, \mathbf{y}_s)d\mathbf{x}_k \\ &\quad + \eta_k^s(0) \cdot \Big[1 - \int P_D^s(\mathbf{x}_k)\delta_k^s(\mathbf{x}_k, 1)d\mathbf{x}_k\Big]. \end{aligned} \tag{2.17}$$

The message $\Lambda_k^s(\mathbf{y}_s)$ can be seen as a likelihood function for the target measurements made by sensor s which also incorporates the target position uncertainty, via $\delta_k^s(\tilde{\mathbf{x}}_k)$, and association uncertainty, via $\eta_k^s(a_k^s)$. The updated belief for agent s can now be evaluated in a Bayesian manner, that is, by multiplying the predicted message $\alpha_{\rightarrow n}(\mathbf{y}_s)$ (i.e., prior) with the inter-agent likelihood messages $\Phi_{\ell\rightarrow s}\ \forall\ \ell \in \mathcal{A}_s$ and agent-to-target likelihood messages $\Lambda_k^s\ \forall\ k \in \mathcal{T}_s$. More specifically the updated agent belief (line 13 in Algorithm 1) is given as

$$b(\mathbf{y}_s) \propto \phi_{\rightarrow n}(\mathbf{y}_s) \prod\nolimits_{\ell\in\mathcal{A}_s} \Phi_{\ell\rightarrow s}(\mathbf{y}_s) \prod\nolimits_{k\in\mathcal{T}_s} \Lambda_k^s(\mathbf{y}_s), \tag{2.18}$$

and normalized as $\int b(\mathbf{y}_s)d\mathbf{y}_s = 1$ in order to represent an approximation to the agent posterior probability density. Note that the product of agent-to-target likelihood messages $\Lambda_k^s\ \forall\ k \in \mathcal{T}_s$ in (2.18) represents the probabilistic transfer of information from target tracking to agent localization. In contrast, for separate localization and MTT algorithms, there are no agent-to-target likelihood messages Λ_k^s in the agent belief as probabilistic information is only passed down from the agents to the targets. Thus, the messages $\Lambda_k^s\ \forall\ k \in \mathcal{T}_s$ lead SCS-MTT methods to improved agent localization as compared to separate localization and MTT methods.

Next, using the updated agent belief computed in (2.18), the message $\theta_k^s(\mathbf{y}_s)$ (line 15 in Algorithm 1 and ④ in Figure 2.1), sent from $\mathbf{y}_s$ to factor node v_k^s, $\forall\ k \in \mathcal{T}_s$ is computed as

$$\theta_k^s(\mathbf{y}_s) \propto \phi_{\rightarrow n}(\mathbf{y}_s) \prod_{\ell\in\mathcal{A}_s} \Phi_{\ell\rightarrow s}(\mathbf{y}_s) \prod_{k'\in\mathcal{T}_s\setminus\{k\}} \Lambda_{k'}^s(\mathbf{y}_s) \tag{2.19}$$

and normalized, i.e., $\int \theta_k^s(\mathbf{y}_s) d\mathbf{y}_s = 1$. The message θ_k^s represents the belief in the localization of agent s without the benefit of PT k (i.e., $\theta_k^s(\mathbf{y}_s) \propto b(\mathbf{y}_s)/\Lambda_k^s(\mathbf{y}_s)$) and is referred to as the *extrinsic information* [127] on agent s, seen by PT k.

Next, for each PT $k \in \mathcal{K}$, the likelihood message $\gamma_k^s(\mathbf{x}_k, r_k)$ (line 18 in Algorithm 1 and ⑤ in Figure 2.1) from factor node v_k^s to the variable node $\tilde{\mathbf{x}}_k$ is computed as

$$\begin{aligned} \gamma_k^s(\mathbf{x}_k, r_k) &= \sum\nolimits_{m=0}^{M^s} \int v_k^s(\mathbf{x}_k, r_k, \mathbf{y}_s; \mathbf{z}_m^s) \eta_k^s(m) \theta_k^s(\mathbf{y}_s) d\mathbf{y}_s \\ &= \begin{cases} \sum\nolimits_{m=1}^{M^s} \frac{\eta_k^s(m) P_D^s(\mathbf{x}_k)}{\lambda^s f_s^{\mathrm{FA}}(\mathbf{z}_m^s)} \int f(\mathbf{z}_m^s | \mathbf{x}_k, \mathbf{y}_s) \theta_k^s(\mathbf{y}_s) d\mathbf{y}_s \; + \; \eta_k^s(0) \left(1 - P_D^s(\mathbf{x}_k)\right), & \text{for } r_k = 1 \\ \eta_k^s(0), & \text{for } r_k = 0 \end{cases} \end{aligned} \tag{2.20}$$

if $k \in \mathcal{T}_s$ and $\gamma_k^s(\mathbf{x}_k, r_k) = 1$ otherwise. The message γ_k^s represents a likelihood function for PT k with respect to the measurements made by agent s. It accounts for the uncertainty in the position of agent s, via θ_k^s, and the uncertainty in the association of the measurements to PT k, via $\eta_k^s(a_k^s)$. Given the messages γ_k^s from all the agents $s \in \mathcal{S}$, we update the target beliefs (line 20 in Algorithm 1) in a decentralized way as

$$b(\mathbf{x}_k, r_k) \propto \alpha_{\rightarrow n}(\mathbf{x}_k, r_k) \prod_{s \in \mathcal{S}} \gamma_k^s(\mathbf{x}_k, r_k). \tag{2.21}$$

Note that (2.21) involves network consensus, i.e., agent s obtains $b(\tilde{\mathbf{x}}_k)$ even if PT k is not observed by agent s. Network consensus is implementation dependent, i.e., it depends on the representation of the messages as discrete particle sets or Gaussian mixtures. In Section 2.4.3 and Section 2.5, we provide algorithms for GM and single Gaussian implementations. Furthermore, the target belief is normalized as $\sum_{r_k \in \{0,1\}} \int b(\mathbf{x}_k, r_k) d\mathbf{x}_k = 1$. Note that (2.21) is reminiscent of the Bayesian multi-sensor update of a target with prior density $\alpha_{\rightarrow n}(\tilde{\mathbf{x}}_k)$ and sensor likelihood functions $\gamma_k^s(\tilde{\mathbf{x}}_k)$.

Finally, for $k \in \mathcal{T}_s$, we compute the $\delta_k^s(\mathbf{x}_k, r_k)$ messages (⑥ in Figure 2.1, sent from $\tilde{\mathbf{x}}_k$ to the factor node v_k^s) as

$$\delta_k^s(\mathbf{x}_k, r_k) \propto \alpha_{\rightarrow n}(\mathbf{x}_k, r_k) \prod_{s' \in \mathcal{S} \setminus \{s\}} \gamma_k^{s'}(\mathbf{x}_k, r_k). \tag{2.22}$$

The message δ_k^s can be seen as the *extrinsic information* on the state of PT k as seen by agent s ($\delta_k^s(\mathbf{x}_k, r_k) \propto b(\mathbf{x}_k, r_k)/\gamma_k^s(\mathbf{x}_k, r_k)$). Note that (2.22) can be efficiently evaluated (or approximated) from the belief $b(\tilde{\mathbf{x}}_k)$, hence avoiding additional network-consensus processes, as presented in Section 2.4.3 and Section 2.5 for the case of Gaussian mixture and single Gaussian implementations. Furthermore, the message $\delta_k^s(\mathbf{x}_k, r_k)$ is only computed if $p \neq P$. At the end of the outer iterations, i.e., when $p = P$, the agent $b(\mathbf{y}_s)$ and target $b(\mathbf{x}_k, r_k)$ beliefs represent estimates of their marginal probability densities for the n-th time step and are used as inputs for the next time step.

Note the similarities between Algorithm 1 and that of [127], with the exception that Algorithm 1 also considers association uncertainty for target measurements which requires the computation of single-target association weights β_k^s and the execution of the inner-BP loop, as done in [126]. The inner-BP loop, as shown in [190], converges to a unique fixed point. In contrast, the convergence of the overall message passing scheme (outer and inner BP loops) is not guaranteed due to the presence of loops in the factor graph of Figure 2.1. This can lead to overconfident beliefs, as also shown in [127], which in practice are countered by performing the outer-BP loop only once per time-step (i.e., $P = 1$). The proposed message passing scheme with $P = 1$ is shown in Section 2.6 and in [127] to accurately localize agents and targets.

2.3.2 Agent and target inference

An MMSE estimate of the state of agent s is obtained via $\hat{\mathbf{y}}_{n,s} = \int \mathbf{y}_s b(\mathbf{y}_s) d\mathbf{y}_s$, where $b(\mathbf{y}_s)$ is the agent marginal density estimated via Algorithm 1. Based on the estimated marginal density $b(\mathbf{x}_k, r_k)$, PT k is declared a valid target if the estimated probability of existence $P_{n,k}^e \triangleq \text{Pr}(r_{n,k} = 1) = \int b(\mathbf{x}_k, 1) d\mathbf{x}_k$ is greater than a specified threshold $P_{n,k}^e \geq \tau$ (in this work $\tau = 0.5$). Subsequently an MMSE state estimate is given as $\hat{\mathbf{x}}_{n,k} = \frac{1}{P_{n,k}^e} \int \mathbf{x}_k b(\mathbf{x}_k, 1) d\mathbf{x}_k$.

We compare the performance of a centralized, PF-based implementation of Algorithm 1, with a separate SPAWN and MTT method which first localizes the agents using SPAWN and subsequently performs MTT, similar to the method of [126]. Since the discussion in this chapter is focused more on the decentralized localization and tracking problem, we have relegated the results of this comparison between two centralized algorithms to Appendix A.1.

In the next two sections, we discuss the Gaussian and Gaussian-mixture based implementation of the messages derived in this section.

2.4 Decentralized Gaussian Mixture SCS-MTT filter

In this section, we present the Gaussian Mixture (GM) implementation of the messages of Section 2.3.1. The filters achieve network-wide consensus over the target beliefs, i.e., over the means, covariance matrices and component weights of the Gaussian Mixture (GM). For most kinematic tracking applications, the communication load of the resulting DGM-SCS-MTT filter is significantly smaller than the PF implementations. In case of GMs, the decentralized computation of target beliefs involves a product of GM likelihood messages (stored at different agents) and a GM prior. The number of components in the complete GM product is exponential in the number of agents. To reduce this high complexity, we propose a novel decentralized Gibbs mechanism, extending the centralized Gibbs approach proposed in [168], to sample only the components of the GM product with the highest weights, and thus approximate the entire product. In parallel, the agents sample local Gaussian components followed by a synchronization step where a consensus is reached among the agents regarding the parameters of the resulting product component.

We denote a Gaussian pdf over $\mathbf{x} \in \mathbb{R}^d$, with mean $\mathbf{m}$ and covariance matrix $\mathbf{P}$ as $\mathcal{N}(\mathbf{x}; \mathbf{m}, \mathbf{P})$. A GM density function $\sum_{j=1}^{J} w^{(j)} \mathcal{N}(\mathbf{x}; \mathbf{m}^{(j)}, \mathbf{P}^{(j)})$ is compactly denoted as $\mathrm{GM}\big(\mathbf{x}; \{(w^{(j)}, \mathbf{m}^{(j)}, \mathbf{P}^{(j)})\}_{j=1}^{J}\big)$. Except for likelihood messages, GM messages are normalized $\sum_{j=1}^{J} w^{(j)} = 1$ for agents while for targets $\sum_{j=1}^{J} w^{(j)} \leq 1$. Throughout this section, we employ the following GM assumptions:

(G1) The agent dynamic model is Gaussian with transition kernel $f(\mathbf{y}_{n,s}|\mathbf{y}_{n-1,s}) = \mathcal{N}(\mathbf{y}_{n,s}; \mathbf{A}_{n,s}\mathbf{y}_{n-1,s}, \mathbf{Q}_{n,s})$.

(G2) The target dynamic model (Table 2.1) involves a constant probability of survival $P^{S}_{n,k}(\mathbf{x}) = P^{S}_{n,k}$, a dynamic model $f(\mathbf{x}_{n,k}, 1|\mathbf{x}_{n-1,k}, 1) = P^{S}_{n,k}\mathcal{N}(\mathbf{x}_{n,k}; \mathbf{B}_{n,k}\mathbf{x}_{n-1,k}, \mathbf{\Sigma}_{n,k})$. We also assume a GM birth density $f_b(\mathbf{x}_{n,k}) = \mathrm{GM}(\mathbf{x}_{n,k}; \{(\omega^{B,(j)}_{n,k}, \boldsymbol{\mu}^{B,(j)}_{n,k}, \mathbf{\Omega}^{B,(j)}_{n,k})\}_{j=1}^{J^B_{n,k}})$ with probability of birth $P^{B}_{n,k} = \sum_{j=1}^{J^B_{n,k}} \omega^{B,(j)}_{n,k}$.

(G3) The inter-agent measurement model of agent s measuring agent ℓ is linear with Gaussian noise: $f(\mathbf{w}_{s,\ell}|\mathbf{y}_s,\mathbf{y}_\ell)=\mathcal{N}\left(\mathbf{w}_{s,\ell};\mathbf{D}_s\mathbf{y}_s+\mathbf{F}_\ell\mathbf{y}_\ell,\mathbf{W}_s\right)$.

(G4) The target measurement model of agent s is linear with Gaussian noise: $f(\mathbf{z}|\mathbf{y}_s,\mathbf{x}_k) = \mathcal{N}(\mathbf{z};\mathbf{G}_s\mathbf{y}_s+\mathbf{E}_s\mathbf{x}_k,\mathbf{R}_s)$, and a constant probability of detection $p^D_{n,s}(\mathbf{x})=P^D_{n,s}$.

(G5) The initial marginal densities of agents and PTs are assumed GM.

The constant probability of survival and of detection is a common requirement in GM implementations of MTT filters (e.g., [181, 180]). Note that the proposed GM-SCS-MTT filter can easily accommodate GM dynamic kernels for both agents (G1) and targets (G2), and GM likelihood functions for both inter-agent (G3) and target (G4) measurements. For compactness, we present the GM expressions for the BP messages of Algorithm 1 under assumptions G1-G5, which, as we will show further, lead to the following generic GM expressions, for the agent and PT beliefs

$$b(\mathbf{y}_{n,s})=\sum\nolimits_{j=1}^{J_{n,s}} w^{(j)}_{n,s}\mathcal{N}(\mathbf{y}_{n,s};\mathbf{m}^{(j)}_{n,s},\mathbf{P}^{(j)}_{n,s}), \tag{2.23}$$

$$b(\mathbf{x}_{n,k},1)=\sum\nolimits_{j=1}^{J_{n,k}} \omega^{(j)}_{n,k}\mathcal{N}(\mathbf{x}_{n,k};\boldsymbol{\mu}^{(j)}_{n,k},\boldsymbol{\Omega}^{(j)}_{n,k}), \tag{2.24}$$

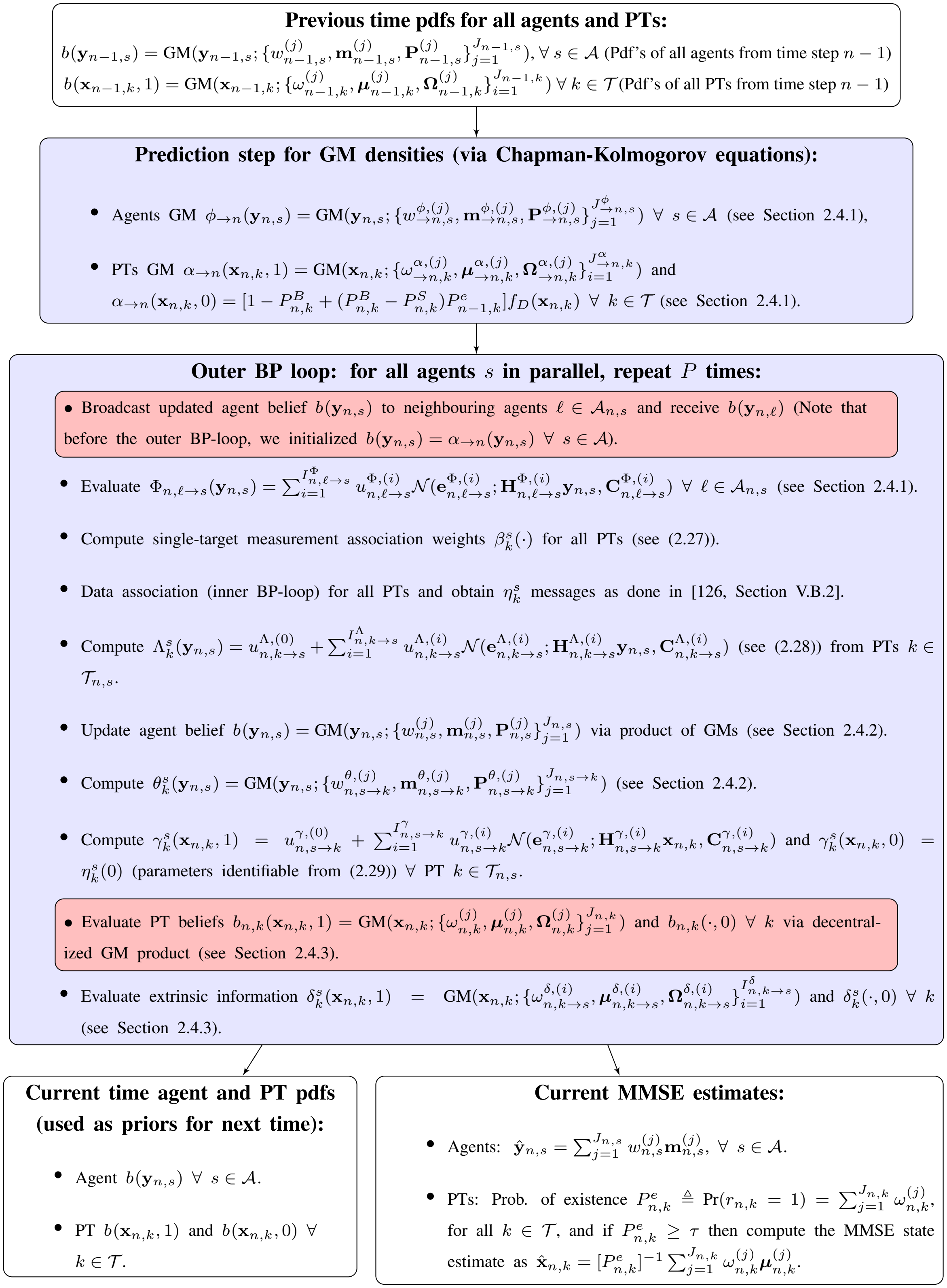
and for the extrinsic information messages

$$\theta^s_{n,k}(\mathbf{y}_{n,s})=\sum\nolimits_{j=1}^{J^\theta_{n,s\to k}} w^{\theta,(j)}_{n,s\to k}\mathcal{N}(\mathbf{y}_{n,s};\mathbf{m}^{\theta,(j)}_{n,s\to k},\mathbf{P}^{\theta,(j)}_{n,s\to k}), \tag{2.25}$$

$$\delta^s_{n,k}(\mathbf{x}_{n,k},1)=\sum\nolimits_{j=1}^{J^\delta_{n,k\to s}} \omega^{\delta,(j)}_{n,k\to s}\mathcal{N}(\mathbf{x}_{n,k};\boldsymbol{\mu}^{\delta,(j)}_{n,k\to s},\boldsymbol{\Omega}^{\delta,(j)}_{n,k\to s}). \tag{2.26}$$

Remark. In practice, as well as in Section 2.6, the nonlinear measurement models are often linearized (for example, using the extended Kalman filter [150, Ch. 2.1]).

Such generic forms for all GM messages are shown in the flowchart of Figure 2.2. Decentralized and local computations are colored in red and blue respectively. Detailed expressions for the GM parameters are presented in the following section. The properties of Gaussian functions [150, Ch. 3.8] and G1-G4 allow the derivation of closed form GM expressions for the GM-SCS-MTT messages.



Previous time pdfs for all agents and PTs:

$b(\mathbf{y}_{n-1,s}) = \mathrm{GM}(\mathbf{y}_{n-1,s};\{w^{(j)}_{n-1,s},\mathbf{m}^{(j)}_{n-1,s},\mathbf{P}^{(j)}_{n-1,s}\}_{j=1}^{J_{n-1,s}}), \forall\, s \in \mathcal{A}$ (Pdf's of all agents from time step $n-1$)

$b(\mathbf{x}_{n-1,k},1) = \mathrm{GM}(\mathbf{x}_{n-1,k};\{\omega^{(j)}_{n-1,k},\boldsymbol{\mu}^{(j)}_{n-1,k},\boldsymbol{\Omega}^{(j)}_{n-1,k}\}_{i=1}^{J_{n-1,k}})\ \forall\, k \in \mathcal{T}$ (Pdf's of all PTs from time step $n-1$)

Prediction step for GM densities (via Chapman-Kolmogorov equations):

- Agents GM $\phi_{\rightarrow n}(\mathbf{y}_{n,s}) = \mathrm{GM}(\mathbf{y}_{n,s};\{w^{\phi,(j)}_{\rightarrow n,s},\mathbf{m}^{\phi,(j)}_{\rightarrow n,s},\mathbf{P}^{\phi,(j)}_{\rightarrow n,s}\}_{j=1}^{J^{\phi}_{\rightarrow n,s}})\ \forall\ s \in \mathcal{A}$ (see Section 2.4.1),
- PTs GM $\alpha_{\rightarrow n}(\mathbf{x}_{n,k},1) = \mathrm{GM}(\mathbf{x}_{n,k};\{\omega^{\alpha,(j)}_{\rightarrow n,k},\boldsymbol{\mu}^{\alpha,(j)}_{\rightarrow n,k},\boldsymbol{\Omega}^{\alpha,(j)}_{\rightarrow n,k}\}_{i=1}^{J^{\alpha}_{\rightarrow n,k}})$ and $\alpha_{\rightarrow n}(\mathbf{x}_{n,k},0) = [1 - P^{B}_{n,k} + (P^{B}_{n,k} - P^{S}_{n,k})P^{e}_{n-1,k}]f_{D}(\mathbf{x}_{n,k})\ \forall\ k \in \mathcal{T}$ (see Section 2.4.1).

Outer BP loop: for all agents s in parallel, repeat P times:

- Broadcast updated agent belief $b(\mathbf{y}_{n,s})$ to neighbouring agents $\ell \in \mathcal{A}_{n,s}$ and receive $b(\mathbf{y}_{n,\ell})$ (Note that before the outer BP-loop, we initialized $b(\mathbf{y}_{n,s}) = \alpha_{\rightarrow n}(\mathbf{y}_{n,s})\ \forall\ s \in \mathcal{A}$).
- Evaluate $\Phi_{n,\ell\rightarrow s}(\mathbf{y}_{n,s}) = \sum_{i=1}^{I^{\Phi}_{n,\ell\rightarrow s}} u^{\Phi,(i)}_{n,\ell\rightarrow s}\mathcal{N}(\mathbf{e}^{\Phi,(i)}_{n,\ell\rightarrow s};\mathbf{H}^{\Phi,(i)}_{n,\ell\rightarrow s}\mathbf{y}_{n,s},\mathbf{C}^{\Phi,(i)}_{n,\ell\rightarrow s})\ \forall\ \ell \in \mathcal{A}_{n,s}$ (see Section 2.4.1).
- Compute single-target measurement association weights $\beta^{s}_{k}(\cdot)$ for all PTs (see (2.27)).
- Data association (inner BP-loop) for all PTs and obtain η^{s}_{k} messages as done in [126, Section V.B.2].
- Compute $\Lambda^{s}_{k}(\mathbf{y}_{n,s}) = u^{\Lambda,(0)}_{n,k\rightarrow s} + \sum_{i=1}^{I^{\Lambda}_{n,k\rightarrow s}} u^{\Lambda,(i)}_{n,k\rightarrow s}\mathcal{N}(\mathbf{e}^{\Lambda,(i)}_{n,k\rightarrow s};\mathbf{H}^{\Lambda,(i)}_{n,k\rightarrow s}\mathbf{y}_{n,s},\mathbf{C}^{\Lambda,(i)}_{n,k\rightarrow s})$ (see (2.28)) from PTs $k \in \mathcal{T}_{n,s}$.
- Update agent belief $b(\mathbf{y}_{n,s}) = \mathrm{GM}(\mathbf{y}_{n,s};\{w^{(j)}_{n,s},\mathbf{m}^{(j)}_{n,s},\mathbf{P}^{(j)}_{n,s}\}_{j=1}^{J_{n,s}})$ via product of GMs (see Section 2.4.2).
- Compute $\theta^{s}_{k}(\mathbf{y}_{n,s}) = \mathrm{GM}(\mathbf{y}_{n,s};\{w^{\theta,(j)}_{n,s\rightarrow k},\mathbf{m}^{\theta,(j)}_{n,s\rightarrow k},\mathbf{P}^{\theta,(j)}_{n,s\rightarrow k}\}_{j=1}^{J_{n,s\rightarrow k}})$ (see Section 2.4.2).
- Compute $\gamma^{s}_{k}(\mathbf{x}_{n,k},1) = u^{\gamma,(0)}_{n,s\rightarrow k} + \sum_{i=1}^{I^{\gamma}_{n,s\rightarrow k}} u^{\gamma,(i)}_{n,s\rightarrow k}\mathcal{N}(\mathbf{e}^{\gamma,(i)}_{n,s\rightarrow k};\mathbf{H}^{\gamma,(i)}_{n,s\rightarrow k}\mathbf{x}_{n,k},\mathbf{C}^{\gamma,(i)}_{n,s\rightarrow k})$ and $\gamma^{s}_{k}(\mathbf{x}_{n,k},0) = \eta^{s}_{k}(0)$ (parameters identifiable from (2.29)) $\forall$ PT $k \in \mathcal{T}_{n,s}$.
- Evaluate PT beliefs $b_{n,k}(\mathbf{x}_{n,k},1) = \mathrm{GM}(\mathbf{x}_{n,k};\{\omega^{(j)}_{n,k},\boldsymbol{\mu}^{(j)}_{n,k},\boldsymbol{\Omega}^{(j)}_{n,k}\}_{j=1}^{J_{n,k}})$ and $b_{n,k}(\cdot,0)\ \forall\ k$ via decentralized GM product (see Section 2.4.3).
- Evaluate extrinsic information $\delta^{s}_{k}(\mathbf{x}_{n,k},1) = \mathrm{GM}(\mathbf{x}_{n,k};\{\omega^{\delta,(i)}_{n,k\rightarrow s},\boldsymbol{\mu}^{\delta,(i)}_{n,k\rightarrow s},\boldsymbol{\Omega}^{\delta,(i)}_{n,k\rightarrow s}\}_{i=1}^{I^{\delta}_{n,k\rightarrow s}})$ and $\delta^{s}_{k}(\cdot,0)\ \forall\ k$ (see Section 2.4.3).

Current time agent and PT pdfs (used as priors for next time):

- Agent $b(\mathbf{y}_{n,s})\ \forall\ s \in \mathcal{A}$.
- PT $b(\mathbf{x}_{n,k},1)$ and $b(\mathbf{x}_{n,k},0)\ \forall\ k \in \mathcal{T}$.

Current MMSE estimates:

- Agents: $\hat{\mathbf{y}}_{n,s} = \sum_{j=1}^{J_{n,s}} w^{(j)}_{n,s}\mathbf{m}^{(j)}_{n,s},\ \forall\ s \in \mathcal{A}$.
- PTs: Prob. of existence $P^{e}_{n,k} \triangleq \Pr(r_{n,k} = 1) = \sum_{j=1}^{J_{n,k}} \omega^{(j)}_{n,k}$, for all $k \in \mathcal{T}$, and if $P^{e}_{n,k} \geq \tau$ then compute the MMSE state estimate as $\hat{\mathbf{x}}_{n,k} = [P^{e}_{n,k}]^{-1}\sum_{j=1}^{J_{n,k}} \omega^{(j)}_{n,k}\boldsymbol{\mu}^{(j)}_{n,k}$.

Fig. 2.2: GM processing flowchart of the DGM-SCS-MTT filter at time n.

In Section 2.4.1, the GM parameters of the prediction and likelihood messages are given. The computation of the GM beliefs (2.23)-(2.24) and the extrinsic information (2.25)-(2.26) requires the product of several GM terms. Exact computation of these is computationally prohibitive and incurs a high communication cost. Therefore in Section 2.4.2, we propose a centralized and efficient algorithm to select high-weight Gaussian components from the GM product based on Gibbs sampling [168]. In Section 2.4.3 a decentralized Gibbs algorithm is proposed for efficiently evaluating the target beliefs. The special case of this algorithm for a single Gaussian implementation is discussed in Section 2.5.

2.4.1 GM prediction and likelihood messages

Agent Prediction Messages

We start with the belief $b(\mathbf{y}_{n-1,s})$ of agent s computed at the previous time $n-1$ and with parameters similar to (2.23). Assuming G1 and substituting the GM representation of $b(\mathbf{y}_{n-1,s})$ in (2.13), we obtain the agent predicted message $\phi_{\rightarrow n}(\mathbf{y}_{n,s}) = \mathrm{GM}\big(\mathbf{y}_{n,s}; \{(w^{\phi,(j)}_{\rightarrow n,s}, \mathbf{m}^{\phi,(j)}_{\rightarrow n,s}, \mathbf{P}^{\phi,(j)}_{\rightarrow n,s})\}_{j=1}^{J^{\phi}_{\rightarrow n,s}}\big)$ with $J^{\phi}_{\rightarrow n,s} = J_{n-1,s}$ Gaussian components with parameters given in Table 2.2a. As seen in Figure 2.2 and discussed in Section 2.3.1, before the BP iterations begin, the current agent belief $b(\mathbf{y}_{n,s})$ is initialized with $\phi_{\rightarrow n}(\mathbf{y}_{n,s})$. Also, we initialize $\theta^{s}_{n,k}(\mathbf{y}_s)$ with $\phi_{\rightarrow n}(\mathbf{y}_{n,s})$.

Target Prediction Messages

Similarly, assuming a GM belief such as (2.24) for PT k at $n-1$, under assumption G2 and from (2.14) we obtain the predicted GM message $\alpha_{\rightarrow n}(\mathbf{x}_{n,k}, 1) = \mathrm{GM}\big(\mathbf{x}_{n,k}; \{(\omega^{\alpha,(j)}_{\rightarrow n,k}, \boldsymbol{\mu}^{\alpha,(j)}_{\rightarrow n,k}, \boldsymbol{\Omega}^{\alpha,(j)}_{\rightarrow n,k})\}_{j=1}^{J^{\alpha}_{\rightarrow n,k}}\big)$. The $J^{\alpha}_{\rightarrow n,k} = J_{n-1,k} + J^{B}_{n,k}$ Gaussian components of $\alpha_{\rightarrow n}(\mathbf{x}_{n,k}, 1)$ are the union of surviving and birthed tracks,

$$\underbrace{\{(\omega^{S,(j)}_{\rightarrow n,k}, \boldsymbol{\mu}^{S,(j)}_{\rightarrow n,k}, \boldsymbol{\Omega}^{S,(j)}_{\rightarrow n,k})\}_j}_{J_{n-1,k}\text{ surviving components}} \bigcup \underbrace{\{(\omega^{B,(j)}_{\rightarrow n,k}, \boldsymbol{\mu}^{B,(j)}_{\rightarrow n,k}, \boldsymbol{\Omega}^{B,(j)}_{\rightarrow n,k})\}_j}_{J^{B}_{n,k}\text{ new birthed components}}$$

GM Message	Weights	Means	Covariance Matrices
$\phi_{\rightarrow n}(\mathbf{y}_{n,s})$	$w^{\phi,(j)}_{\rightarrow n,s} = w^{(j)}_{n-1,s}$	$\mathbf{m}^{\phi,(j)}_{\rightarrow n,s} = \mathbf{A}_{n,s}\mathbf{m}^{(j)}_{n-1,s}$	$\mathbf{P}^{\phi,(j)}_{\rightarrow n,s} = \mathbf{Q}_{n,s} + \mathbf{A}_{n,s}\mathbf{P}^{(j)}_{n-1,s}\mathbf{A}^{T}_{n,s}$
$\alpha_{\rightarrow n}(\mathbf{x}_{n,k}, 1)$	$\omega^{S,(j)}_{\rightarrow n,k} = P^{S}_{n,k}\omega^{(j)}_{n-1,k}$	$\boldsymbol{\mu}^{S,(j)}_{\rightarrow n,k} = \mathbf{B}_{n,k}\boldsymbol{\mu}^{(j)}_{n-1,k}$	$\mathbf{\Omega}^{S,(j)}_{\rightarrow n,s} = \mathbf{\Sigma}_{n,k} + \mathbf{B}_{n,k}\mathbf{\Omega}^{(j)}_{n-1,k}\mathbf{B}^{T}_{n,k}$
	$\omega^{B,(j)}_{\rightarrow n,k} = \omega^{B,(j)}_{n,k}(1 - P^{e}_{n-1,k})$	$\boldsymbol{\mu}^{B,(j)}_{\rightarrow n,k} = \boldsymbol{\mu}^{B,(j)}_{n,k}$	$\mathbf{\Omega}^{B,(j)}_{\rightarrow n,k} = \mathbf{\Omega}^{B,(j)}_{n,k}$

(a) GM parameters of prediction messages for agents (2.13) and PTs (2.14).

Message	Weights	Residuals	Obs. matrix	Covariance Matrices
$\Phi_{\ell\rightarrow s}(\mathbf{y}_s)$	$u^{\Phi,(i)}_{\ell\rightarrow s} = w^{(i)}_{\ell}$	$\mathbf{e}^{\Phi,(i)}_{\ell\rightarrow s} = \mathbf{w}_{s,\ell} - \mathbf{F}_{\ell}\mathbf{m}^{(i)}_{\ell}$	$\mathbf{H}^{\Phi,(i)}_{\ell\rightarrow s} = \mathbf{D}_s$	$\mathbf{C}^{\Phi,(i)}_{\ell\rightarrow s} = \mathbf{W}_s + \mathbf{F}_{\ell}\mathbf{P}^{(i)}_{\ell}\mathbf{F}^{T}_{\ell}$
$\Lambda^{s}_{k}(\mathbf{y}_s)$	$u^{\Lambda,(m,j)}_{k\rightarrow s} = \frac{\eta^{s}_{k}(m)P^{s}_{D}\omega^{\delta,(j)}_{k\rightarrow s}}{\lambda^{s} f^{\mathrm{FA}}_{s}(\mathbf{z}^{s}_{m})}$ $u^{\Lambda,(0)}_{k\rightarrow s} = \eta^{s}_{k}(0)[1 - P^{s}_{D}\sum_{j}\omega^{\delta,(j)}_{k\rightarrow s}]$	$\mathbf{e}^{\Lambda,(m,j)}_{k\rightarrow s} = \mathbf{z}^{s}_{m} - \mathbf{E}_s\boldsymbol{\mu}^{\delta,(j)}_{k\rightarrow s}$	$\mathbf{H}^{\Lambda,(m,j)}_{k\rightarrow s} = \mathbf{G}_s$	$\mathbf{C}^{\Lambda,(m,j)}_{k\rightarrow s} = \mathbf{R}_s + \mathbf{E}_s\mathbf{\Omega}^{\delta,(j)}_{k\rightarrow s}\mathbf{E}^{T}_{s}$ (independent of m)
$\gamma^{s}_{k}(\mathbf{x}_k, 1)$	$u^{\gamma,(m,j)}_{s\rightarrow k} = \frac{\eta^{s}_{k}(m)P^{s}_{D}w^{\theta,(j)}_{s\rightarrow k}}{\lambda^{s} f^{\mathrm{FA}}_{s}(\mathbf{z}^{s}_{m})}$ $u^{\gamma,(0)}_{s\rightarrow k} = \eta^{s}_{k}(m)[1 - P^{s}_{D}]$	$\mathbf{e}^{\gamma,(m,j)}_{s\rightarrow k} = \mathbf{z}^{s}_{m} - \mathbf{G}_s\mathbf{m}^{\theta,(j)}_{s\rightarrow k}$	$\mathbf{H}^{\gamma,(m,j)}_{s\rightarrow k} = \mathbf{E}_s$	$\mathbf{C}^{\gamma,(m,j)}_{s\rightarrow k} = \mathbf{R}_s + \mathbf{G}_s\mathbf{P}^{\theta,(j)}_{s\rightarrow k}\mathbf{G}^{T}_{s}$ (independent of m)

(b) GM parameters of likelihood messages for agents (2.15), (2.17) and PTs (2.20).

Table 2.2: Gaussian mixture parameters for the message passing scheme of Algorithm 1.

where the component parameters are given in Table 2.2a. Similarly, $\alpha_{\rightarrow n}(\mathbf{x}_{n,k}, 0) = [1 - P^{B}_{n,k} + (P^{B}_{n,k} - P^{S}_{n,k})P^{e}_{n-1,k}]f_D(\mathbf{x}_{n,k})$. Also, we initialize $\delta^{s}_{n,k}(\tilde{\mathbf{x}}_{n,k}) = \alpha_{\rightarrow n}(\tilde{\mathbf{x}}_{n,k})$. Henceforth, we drop the time index n since all the following messages correspond to the current time instant.

Single-target association weights

Using the generic GM representations for θ^{s}_{k} (2.25) and δ^{s}_{k} (2.26), the single-target association weight $\beta^{s}_{k}(m)$ in (2.16), for $m \in [1 : M^{s}_{n}]$ becomes

$$\beta^{s}_{k}(m) = \sum_{j=1}^{J^{\theta}_{s\rightarrow k}} \sum_{i=1}^{J^{\delta}_{k\rightarrow s}} \frac{P^{s}_{D}\omega^{\delta,(i)}_{k\rightarrow s} w^{\theta,(j)}_{s\rightarrow k}}{\lambda^{s} f^{\mathrm{FA}}_{s}(\mathbf{z}^{s}_{m})} \mathcal{N}(\mathbf{z}^{s}_{m}; \mathbf{m}^{(i,j)}_{s,k}, \mathbf{P}^{(i,j)}_{s,k}),$$

where

$$\begin{aligned} \mathbf{m}^{(i,j)}_{s,k} &= \mathbf{E}_s\boldsymbol{\mu}^{\delta,(i)}_{k\rightarrow s} + \mathbf{G}_s\mathbf{m}^{\theta,(j)}_{s\rightarrow k} \\ \mathbf{P}^{(i,j)}_{s,k} &= \mathbf{R}_s + \mathbf{E}_s\mathbf{\Omega}^{\delta,(i)}_{k\rightarrow s}\mathbf{E}^{T}_{s} + \mathbf{G}_s\mathbf{P}^{\theta,(j)}_{s\rightarrow k}\mathbf{G}^{T}_{s}. \end{aligned}$$

For $m = 0$, $\beta^{s}_{k}(0) = 1 - P^{s}_{D}\sum_{i=1}^{J^{\delta}_{k\rightarrow s}}\omega^{\delta,(i)}_{k\rightarrow s}$. These weights are then used to compute the messages $\{\eta^{s}_{k}(m)\}$ using the inner BP loop [190]. The η^{s}_{k} messages are subsequently used in the computations of the following likelihood messages.

Agent likelihood messages

During an outer-BP loop, using the generic GM form for the belief of agent ℓ (2.23) and under G3, the likelihood message $\Phi_{\ell\to s}(\mathbf{y}_s)$ in (2.15) becomes

$$\Phi_{\ell\to s}(\mathbf{y}_s) = \sum_{i=1}^{I^{\Phi}_{\ell\to s}} u^{\Phi,(i)}_{\ell\to s}\mathcal{N}\big(\mathbf{e}^{\Phi,(i)}_{\ell\to s}; \mathbf{H}^{\Phi,(i)}_{\ell\to s}\mathbf{y}_s, \mathbf{C}^{\Phi,(i)}_{\ell\to s}\big), \tag{2.27}$$

where $I^{\Phi}_{\ell\to s} = J_\ell$, the weights $u^{\Phi,(i)}_{\ell\to s}$, residuals $\mathbf{e}^{\Phi,(i)}_{\ell\to s}$, observation matrices $\mathbf{H}^{\Phi,(i)}_{\ell\to s}$ and covariance matrices $\mathbf{C}^{\Phi,(i)}_{\ell\to s}$ are given in Table 2.2b. Similarly, using G4 and the GM expression for δ^s_k of (2.26), the message $\Lambda^s_k(\mathbf{y}_s)$ in (2.17) becomes

$$\Lambda^s_k(\mathbf{y}_s) = u^{\Lambda,(0)}_{k\to s} + \sum_{m=1}^{M^s}\sum_{j=1}^{J^{\delta}_{k,s}} u^{\Lambda,(m,j)}_{k\to s}\mathcal{N}(\mathbf{e}^{\Lambda,(m,j)}_{k\to s}; \mathbf{H}^{\Lambda,(m,j)}_{k\to s}\mathbf{y}_s, \mathbf{C}^{\Lambda,(m,j)}_{k\to s}), \tag{2.28}$$

with parameters given in Table 2.2b.

Target likelihood messages

From (2.20) and assuming G4 and the GM form (2.25) for θ^s_k, the γ^s_k message becomes

$$\gamma^s_k(\mathbf{x}_k, 1) = u^{\gamma,(0)}_{s\to k} + \sum_{m=1}^{M^s}\sum_{j=1}^{J^{\theta}_{s,k}} u^{\gamma,(m,j)}_{s\to k}\mathcal{N}(\mathbf{e}^{\gamma,(m,j)}_{s\to k}; \mathbf{H}^{\gamma,(m,j)}_{s\to k}\mathbf{x}_k, \mathbf{C}^{\gamma,(m,j)}_{s\to k}), \tag{2.29}$$

with parameters given in Table 2.2b.

In the flowchart of Figure 2.2, for the GM likelihood messages Λ^s_k (2.28) and γ^s_k (2.29), for compactness, we employ a notation using a single summation. The correspondence between the double and single summation parameters for Λ^s_k is given by any one-to-one mapping from $[1:M^s]\times[1:J^{\delta}_{k,s}]$ to $[1:I^{\Lambda}_{k\to s}]$, where $I^{\Lambda}_{k\to s} = M^s\cdot J^{\delta}_{k,s}$ An analogous one-to-one mapping yields the correspondence of parameters for γ^s_k.

2.4.2 Agent belief via centralized GM product

The belief (2.18) of agent s, under the assumptions of the previous section, is given by the product of locally-available GM likelihood messages and has the generic form

$$b(\mathbf{y}_s) = \sum\nolimits_{j=1}^{J_{\to n,s}} w^{(j)}_{\to n,s} \mathcal{N}\big(\mathbf{y}_s; \mathbf{m}^{(j)}_{\to n,s}, \mathbf{P}^{(j)}_{\to n,s}\big) \tag{2.30}$$
$$\times \prod\nolimits_{l\in\mathcal{N}_s} \Big(u^{(0)}_{l\to s} + \sum\nolimits_{i_l=1}^{I_{l\to s}} u^{(i_l)}_{l\to s} \mathcal{N}\big(\mathbf{e}^{(i_l)}_{l\to s}; \mathbf{H}^{(i_l)}_{l\to s}\mathbf{y}_s, \mathbf{C}^{(i_l)}_{l\to s}\big)\Big),$$

where $\mathcal{N}_s = \mathcal{A}_s \cup \mathcal{T}_s$ is the set of neighboring agents and the targets observed by agent s at time n. The GM in the first line represents the predicted message $\phi_{\to n}$ (2.18), where the superscript ϕ is dropped for clarity. The $L \triangleq |\mathcal{N}_s|$ GM likelihood terms in the second line represent the various inter-agent and target measurement likelihood terms ($\Phi_{\ell\to s}$ and Λ^s_k respectively). Since both Φ and Λ share the same GM likelihood structure, the superscripts Φ and Λ are dropped for clarity and solely the index l identifies each likelihood term as a $\Phi_{l\to s}$ (if $l \in \mathcal{A}_s$) or a Λ^s_l (if $l \in \mathcal{T}_s$) message. The corresponding parameters for each likelihood term $u^{(i_l)}_{l\to s}, \mathbf{e}^{(i_l)}_{l\to s}, \mathbf{H}^{(i_l)}_{l\to s}, \mathbf{C}^{(i_l)}_{l\to s}$ are defined in Table 2.2b. As already stated above, we replace the double superscript (m, j) in parameters of Λ^s_l with the single superscript (i). Comparing (2.27) with (2.28), each Λ^s_l message has a constant term $u^{(0)}_{l\to s} \neq 0$, whereas for $\Phi_{l\to s}$, $u^{(0)}_{l\to s} = 0$. For $i_l \neq 0$ let

$$\begin{aligned}
\tilde{\mathbf{e}}^{(i_l)}_{l\to s} &\triangleq \big[\mathbf{H}^{(i_l)}_{l\to s}\big]^T \big[\mathbf{C}^{(i_l)}_{l\to s}\big]^{-1} \mathbf{e}^{(i_l)}_{l\to s},\\
\tilde{\mathbf{C}}^{(i_l)}_{l\to s} &\triangleq \big[\mathbf{H}^{(i_l)}_{l\to s}\big]^T \big[\mathbf{C}^{(i_l)}_{l\to s}\big]^{-1} \mathbf{H}^{(i_l)}_{l\to s},\\
c^{(i_l)}_{l\to s} &\triangleq \log\left(\frac{u^{(i_l)}_{l\to s}}{\sqrt{\det(2\pi\mathbf{C}^{(i_l)}_{l\to s})}}\right) - \frac{1}{2}\big[\big(\mathbf{e}^{(i_l)}_{l\to s}\big)^T \big(\mathbf{C}^{(i_l)}_{l\to s}\big)^{-1} \mathbf{e}^{(i_l)}_{l\to s}\big].
\end{aligned} \tag{2.31}$$

For $i_l = 0$, let $\tilde{\mathbf{e}}^{(0)}_{l\to s} = \mathbf{0}_{d_a}$, $\tilde{\mathbf{C}}^{(0)}_{l\to s} = \mathbf{0}_{d_a\times d_a}$ and $c^{(0)}_{l\to s} = \log(u^{(0)}_{l\to s})$. We also define the L-length vector $\boldsymbol{i} \triangleq [i_1, \cdots, i_L]$, where $i_l \in [0 : I_{l\to s}]$, and the product space $\boldsymbol{I}_L \triangleq \bigtimes_{l=1}^{L} [0 : I_{l\to s}]$. Furthermore, let

$$\tilde{\mathbf{C}}^{(\boldsymbol{i})} \triangleq \sum\nolimits_{l=1}^{L} \tilde{\mathbf{C}}^{(i_l)}_{l\to s}, \tilde{\mathbf{e}}^{(\boldsymbol{i})} \triangleq \sum\nolimits_{l=1}^{L} \tilde{\mathbf{e}}^{(i_l)}_{l\to s}, c^{(\boldsymbol{i})} \triangleq \sum\nolimits_{l=1}^{L} c^{(i_l)}_{l\to s}. \tag{2.32}$$

Then by the property of the product of Gaussian functions [150, Ch. 3.8], the result of (2.30) is the $\mathrm{GM}(\mathbf{y}_s; \{w_s^{(j,\boldsymbol{i})}, \mathbf{m}_s^{(j,\boldsymbol{i})}, \mathbf{P}_s^{(j,\boldsymbol{i})}\}_{j\in[1:J_{\to n,s}],\boldsymbol{i}\in\boldsymbol{I}_L})$ where

$$\mathbf{P}_s^{(j,\boldsymbol{i})} = \Big[\big(\mathbf{P}_{\to n,s}^{(j)}\big)^{-1} + \tilde{\mathbf{C}}^{(\boldsymbol{i})}\Big]^{-1} \tag{2.33}$$

$$\mathbf{m}_s^{(j,\boldsymbol{i})} = \mathbf{P}_s^{(j,\boldsymbol{i})}\Big[\big(\mathbf{P}_{\to n,s}^{(j)}\big)^{-1}\mathbf{m}_{\to n,s}^{(j)} + \tilde{\mathbf{e}}^{(\boldsymbol{i})}\Big] \tag{2.34}$$

$$\begin{aligned} w_s^{(j,\boldsymbol{i})} &= w_{\to n,s}^{(j)} \exp\Big(c^{(\boldsymbol{i})} - \frac{1}{2}\big(\mathbf{m}_{\to n,s}^{(j)}\big)^T\big(\mathbf{P}_{\to n,s}^{(j)}\big)^{-1}\mathbf{m}_{\to n,s}^{(j)}\Big) \\ &\quad\times \exp\Big(\frac{1}{2}\big(\mathbf{m}_s^{(j,\boldsymbol{i})}\big)^T\big(\mathbf{P}_s^{(j,\boldsymbol{i})}\big)^{-1}\mathbf{m}_s^{(j,\boldsymbol{i})}\Big)\sqrt{\frac{\det(\mathbf{P}_s^{(j,\boldsymbol{i})})}{\det(\mathbf{P}_{\to n,s}^{(j)})}}. \end{aligned} \tag{2.35}$$

Although the computation of (2.33)-(2.35) involves parameters that are locally available at each agent (2.18), it has computational complexity $O(J_{\to n,s}\prod_{l=1}^{L} I_{l\to s})$, i.e., exponential in the number L of likelihood terms. In the following, we present a Gibbs-sampling based method that efficiently constructs a truncated GM approximation of (2.30) where only the T highest scoring mixture components are retained.

GM product via Gibbs sampling

The Gibbs sampling approach borrows from the method in [168] which involves the product of GM probability densities whereas (2.30) involves the product of a GM density with L GM likelihood terms. The Gibbs procedure for the GM product of (2.30) is given in Algorithm 2 and referred to as Centralized Gibbs, since all the required messages are locally available.

To address the challenge of the high number $\prod_{l=1}^{L} I_{l\to s}$ of components of the likelihood product in (2.30), we aim to select component labels $\boldsymbol{i}$ from the product space $\boldsymbol{I}_L$ of likelihood components that lead to Gaussian components (2.33)-(2.35) with high weights $w_s^{(j,\boldsymbol{i})}$. Ideally, this can be achieved by sampling independently with probability

$$\Pr(\boldsymbol{i} = [i_1, \ldots, i_L]^T) = \sum\nolimits_{j=1}^{J_{\to n,s}} \Pr(j, \boldsymbol{i} = [i_1, \ldots, i_L]^T) \propto \sum\nolimits_{j=1}^{J_{\to n,s}} w_s^{(j,\boldsymbol{i})}. \tag{2.36}$$

According to (2.36), vectors $\boldsymbol{i}$ that lead to higher weights (2.35) are selected with higher probabil-

Algorithm 2 Centralized Gibbs GM product for belief $b(\mathbf{y}_s)$

```
1:  **Input** parameters of (2.30).
2:  Sample $i_l \,\forall\, l$ s.t. $\Pr(i_l) \propto \varrho(i_l)$ where $\varrho(0) = u^{(0)}_{l\to s}$ and for $i_l \neq 0$, $\varrho(i_l)$ in (2.37).
3:  Compute $c^{(\boldsymbol{i})} = \sum_{l=1}^{L} c^{i_l}_{l\to s}$, $\tilde{\mathbf{C}}^{(\boldsymbol{i})} = \sum_{l=1}^{L} \tilde{\mathbf{C}}^{(i_l)}_{l\to s}$ and $\tilde{\mathbf{e}}^{(\boldsymbol{i})} = \sum_{l=1}^{L} \tilde{\mathbf{e}}^{(i_l)}_{l\to s}$.
4:  **for** $l \leftarrow 1$ to $L$ **do**
5:      Compute $c^{(\boldsymbol{i}_{\neg l})} = c^{(\boldsymbol{i})} - c^{(i_l)}_{l\to s}$, $\tilde{\mathbf{C}}^{(\boldsymbol{i}_{\neg l})} = \tilde{\mathbf{C}}^{(\boldsymbol{i})} - \tilde{\mathbf{C}}^{(i_l)}_{l\to s}$ and $\tilde{\mathbf{e}}^{(\boldsymbol{i}_{\neg l})} = \tilde{\mathbf{e}}^{(\boldsymbol{i})} - \tilde{\mathbf{e}}^{(i_l)}_{l\to s}$.
6:      **for** $q \leftarrow 0$ to $I_{l\to s}$ **do**
7:          Let $\boldsymbol{i}_{*q} \triangleq [i_1, \ldots, i_{l-1}, q, i_{l+1}, \ldots, i_L]$.
8:          Set $c^{(\boldsymbol{i}_{*q})} = c^{(\boldsymbol{i}_{\neg l})} + c^{(q)}_{l\to s}$, $\tilde{\mathbf{C}}^{(\boldsymbol{i}_{*q})} = \tilde{\mathbf{C}}^{(\boldsymbol{i}_{\neg l})} + \tilde{\mathbf{C}}^{(q)}_{l\to s}$, and $\tilde{\mathbf{e}}^{(\boldsymbol{i}_{*q})} = \tilde{\mathbf{e}}^{(\boldsymbol{i}_{\neg l})} + \tilde{\mathbf{e}}^{(q)}_{l\to s}$.
9:          **for** $j \leftarrow 1$ to $J_{\to n,s}$ **do**
10:             Compute $\mathbf{P}_s^{(j,\boldsymbol{i}_{*q})}, \mathbf{m}_s^{(j,\boldsymbol{i}_{*q})}, w_s^{(j,\boldsymbol{i}_{*q})}$ (2.33)-(2.35).
11:         **end for**
12:         Compute $\pi_l(q|\boldsymbol{i}_{\neg l}) \propto \sum_{j=1}^{J_{\to n,s}} w_s^{(j,\boldsymbol{i}_{*q})}$.
13:     **end for**
14:     Sample new label $q' \sim \pi_l(q|\boldsymbol{i}_{\neg l})$ and set $\boldsymbol{i} \leftarrow \boldsymbol{i}_{*q'}$,
15:     $\tilde{\mathbf{C}}^{(\boldsymbol{i})} \leftarrow \tilde{\mathbf{C}}^{(\boldsymbol{i}_{*q'})}$, $\tilde{\mathbf{e}}^{(\boldsymbol{i})} \leftarrow \tilde{\mathbf{e}}^{(\boldsymbol{i}_{*q'})}$, $c^{(\boldsymbol{i})} \leftarrow c^{(\boldsymbol{i}_{*q'})}$.
16: **end for**
17: Repeat the steps 4-16 for $T$ iterations.
18: **Return** $c^{(\boldsymbol{i})}$, $\tilde{\mathbf{C}}^{(\boldsymbol{i})}$, and $\tilde{\mathbf{e}}^{(\boldsymbol{i})}$ for the distinct samples $\boldsymbol{i} \in \boldsymbol{I}_L$.
```

ity. However, sampling from (2.36) is difficult as it requires the computation of all $\{w_s^{(j,\boldsymbol{i})}\}$ which is again $O(J_{\to n,s} \prod_{l=1}^{L} I_{l\to s})$. The Gibbs sampler constructs a finite Markov chain with stationary distribution (2.36) by iteratively sampling from conditional densities that are easily constructed. The proposed Gibbs method starts by sampling an initial label vector $\boldsymbol{i}$ (line 2, Algorithm 2), with probabilities $\Pr(i_l) \propto \varrho(i_l)$ where $\varrho(0) = u^{(0)}_{l\to s}$ and for $i_l \neq 0$

$$\begin{aligned}
\varrho(i_l) &= u^{(i_l)}_{l\to s} \int \phi_{\to n}(\mathbf{y}_s) \mathcal{N}(\mathbf{e}^{(i_l)}_{l\to s}; \mathbf{H}^{(i_l)}_{l\to s}\mathbf{y}_s, \mathbf{C}^{(i_l)}_{l\to s}) d\mathbf{y}_s \\
&= u^{(i_l)}_{l\to s} \sum_{j=1}^{J_{\to n,s}} w^{(j)}_{\to n,s} \times \mathcal{N}\big(\mathbf{e}^{(i_l)}_{l\to s}; \mathbf{H}^{(i_l)}_{l\to s}\mathbf{m}^{(j)}_{\to n,s}, \mathbf{C}^{(i_l)}_{l\to s} + \mathbf{H}^{(i_l)}_{l\to s}\mathbf{P}^{(j)}_{\to n,s}\big[\mathbf{H}^{(i_l)}_{l\to s}\big]^T\big).
\end{aligned} \tag{2.37}$$

These initial weights are based on the intuition that if $\mathcal{N}_s = \{l\}$, i.e., agent s has only one neighbor, (2.37) would give the weight contributed to by the i_l-th component of the likelihood, in the resulting GM in (2.30). This is followed by sequentially sampling new labels for each of the L likelihood messages (lines 5-15, Algorithm 2), from the conditional distributions of (2.36), i.e.,

$$\pi_l(i_l|\boldsymbol{i}_{\neg l}) \triangleq \text{Pr}(i_l|\boldsymbol{i}_{\neg l}) = \frac{\text{Pr}(\boldsymbol{i})}{\text{Pr}(\boldsymbol{i}_{\neg l})} = \frac{\sum_{j=1}^{J_{\rightarrow n,s}} \text{Pr}(j,\boldsymbol{i})}{\sum_{q=1}^{I_{l\rightarrow s}} \sum_{j=1}^{J_{\rightarrow n,s}} \text{Pr}(j,\boldsymbol{i}_{*q})} = \frac{\sum_{j=1}^{J_{\rightarrow n,s}} w_s^{(j,\boldsymbol{i})}}{\sum_{q=1}^{I_{l\rightarrow s}} \sum_{j=1}^{J_{\rightarrow n,s}} w_s^{(j,\boldsymbol{i}_{*q})}}, \tag{2.38}$$

where $\boldsymbol{i}_{\neg l} \triangleq [i_1, \ldots, i_{l-1}, i_{l+1}, \ldots, i_L]^T$ and $\boldsymbol{i}_{*q} \triangleq [i_1, \ldots, i_{l-1}, q, i_{l+1}, \ldots, i_L]^T$. Each cycle (lines 5-15) of the Gibbs sampler involves sampling a new component $q \in [0 : I_{l\rightarrow s}]$ for each of the likelihood messages $l \in [1 : L]$. Holding fixed the labels for messages $[1 : L] \setminus \{l\}$, the parameters (2.32) are computed by first removing the contribution of the old label i_l (line 5) and adding the contribution of each $q \in [0 : I_{l\rightarrow s}]$ (line 8). Next, the resulting components are used to update the predicted agent message (line 10) and the conditional distribution (2.38) is obtained by marginalizing out the prediction message labels $j \in [1 : J_{\rightarrow n,s}]$ (line 12). A new label i_l is sampled (line 14) and the corresponding parameters (2.32) are updated before continuing the Gibbs cycle for the next likelihood message. The entire sampling procedure is repeated T times and the parameters $(c^{(\boldsymbol{i})}, \tilde{\mathbf{C}}^{(\boldsymbol{i})}, \tilde{\mathrm{e}}^{(\boldsymbol{i})})$ corresponding to all distinct vectors $\boldsymbol{i}$ (i.e., two vectors differ in at least one entry) are returned. The T highest scoring components according to $c^{(\boldsymbol{i})}$ are used to construct the truncated agent belief via (2.33)-(2.35). The convergence of Algorithm 2 and the uniqueness of the stationary distribution follow from the regularity of the transition matrix $P_{\boldsymbol{i},\boldsymbol{i}'} = \pi(\boldsymbol{i}|\boldsymbol{i}') > 0$ (as $w_s^{(j,\boldsymbol{i})} > 0 \;\forall\; (j, \boldsymbol{i})$ from (2.35)). The convergence rate is geometrically fast [58, Section 4.3.3], i.e., $|[P^n]_{\boldsymbol{i},\boldsymbol{i}'} - \text{Pr}(\boldsymbol{i})| \leq (1 - 2\vartheta)^n \;\forall\; \boldsymbol{i}, \boldsymbol{i}'$ and where $\vartheta = \min_{\boldsymbol{i},\boldsymbol{i}'} P_{\boldsymbol{i},\boldsymbol{i}'}$ is the least likely 1-step transition probability. All resulting samples are used since every distinct sample $\boldsymbol{i}$ contributes to an improved approximation of (2.30). Hence, no burn-in period is required. Due to the pre-computations at lines 5-8 in Algorithm 2, the evaluation of (2.33)-(2.35) at lines 9-11 for all $l \in [1 : L]$ is $O(J_{\rightarrow n,s} d_a^3)$, leading to a time complexity for Algorithm 2 of $O(T J_{\rightarrow n,s} d_a^3 \sum_l^L I_{l\rightarrow s})$.

Computation of extrinsic information $\theta_k^s(\cdot)$

The θ_k^s message (2.19) represents the extrinsic information sent from agent s to the PT k. If $l \in \mathcal{T}_s$ in the generic product of (2.30), then $\theta_l^s(\mathbf{y}_s) \propto b(\mathbf{y}_s)/\Lambda_l^s(\mathbf{y}_s)$ appears to be a ratio of GMs (which is not a GM in general). An alternate procedure based on (2.19) is described next. First note from line 5 of Algorithm 2 that the parameters $c^{(\boldsymbol{i}_{\neg l})}$, $\tilde{\mathbf{C}}^{(\boldsymbol{i}_{\neg l})}$ and $\tilde{\mathbf{e}}^{(\boldsymbol{i}_{\neg l})}$ characterize the product $\prod_{\ell \neq l} u^{(i_\ell)}_{l\to s}\mathcal{N}(\mathbf{e}^{(i_\ell)}_{l\to s}; \mathbf{H}^{(i_\ell)}_{l\to s}\mathbf{y}_s, \mathbf{C}^{(i_\ell)}_{l\to s})$ of likelihood terms identified by the labels $\boldsymbol{i}_{\neg l}$. The resulting components, after multiplication with the prior $\phi_{\to n}(\mathbf{y}_s)$, lead to an efficient GM approximation of θ^s_k, without requiring a dedicated separate procedure like Algorithm 2 to compute θ^s_k.

2.4.3 Target belief via decentralized GM product

Decentralized SCS-MTT algorithms require a distributed evaluation of the target beliefs across the entire network. The computed target belief for a PT k needs to be identical across all the agents, including the agents that do not observe the PT k at time n. In this section, we propose an efficient method based on Gibbs sampling and average consensus for GM beliefs. As we shall see, much of the discussion in this section follows Section 2.4.2 closely, with the difference that not all the messages are locally available at any single agent. The belief (2.21) for a PT k can be expressed as

$$\begin{aligned} b(\mathbf{x}_k, 1) &= \sum\nolimits_{j=1}^{J_{\to n,k}} \omega^{(j)}_{\to n,k}\mathcal{N}(\mathbf{x}_k; \boldsymbol{\mu}^{(j)}_{\to n,k}, \boldsymbol{\Omega}^{(j)}_{\to n,k}) & (2.39)\\ &\quad\times \prod\nolimits_{s=1}^{S}\Big[u^{(0)}_{s\to k} + \sum\nolimits_{i_s=1}^{I_{s\to k}} u^{(i_s)}_{s\to k}\mathcal{N}(\mathbf{e}^{(i_s)}_{s\to k}; \mathbf{H}^{(i_s)}_{s\to k}\mathbf{x}_k, \mathbf{C}^{(i_s)}_{s\to k})\Big], & \\ b(\mathbf{x}_k, 0) &= \alpha_{\to n}(\mathbf{x}_k, 0)\prod\nolimits_{s=1}^{S}\eta^s_k(0), & (2.40) \end{aligned}$$

where (2.39) is analogous to (2.30) in its generic form. The GM in the first line represents the predicted message $\alpha_{\to n}(\mathbf{x}_k, 1)$ in (2.14). The likelihood terms in the second line represent the $\gamma^s_k(\cdot, 1)$ messages. Note that each agent s only has access to its local message γ^s_k. We use the generic forms of $\alpha_{\to n}$ and $\gamma^s_k(\cdot, 1)$ from Figure 2.2. For clarity, the respective superscripts α, γ are dropped from the parameters. Note that if a target is not observed by an agent s, i.e., if $k \notin \mathcal{T}_{n,s}$, then $\gamma^s_k(\mathbf{x}_k, 1) = \gamma^s_k(\mathbf{x}_k, 0) = 1$ which, for $r_k = 1$, is represented as $u^{(0)}_{s\to k} = 1$ with $I_{s\to k} = 0$ and $\log(\eta^s_k(0)) = 0$. As a consequence of assumption (A10) from Section 2.2.2, each agent k has access to its local parameters and the predicted message $\alpha_{\to n}(\mathbf{x}_k)$ (Section 2.4.1), the latter being identical across all agents. For compactness, $\forall\ s \in \mathcal{A}$ and $i_s \neq 0$ we define the parameters of the local $\gamma_k^s(\cdot, 1)$ messages as

$$\begin{aligned}
\tilde{\mathbf{e}}^{(i_s)}_{s\to k} &\triangleq \big[\boldsymbol{H}^{(i_s)}_{s\to k}\big]^T \big[\mathbf{C}^{(i_s)}_{s\to k}\big]^{-1} \mathbf{e}^{(i_s)}_{s\to k}, \\
\tilde{\mathbf{C}}^{(i_s)}_{s\to k} &\triangleq \big[\mathbf{H}^{(i_s)}_{s\to k}\big]^T \big[\mathbf{C}^{(i_s)}_{s\to k}\big]^{-1} \mathbf{H}^{(i_s)}_{s\to k}, \\
c^{(i_s)}_{s\to k} &\triangleq \log\left(\frac{u^{(i_s)}_{s\to k}}{\sqrt{\det(2\pi\mathbf{C}^{(i_s)}_{s\to k})}}\right) - \frac{1}{2}\big[\big(\mathbf{e}^{(i_s)}_{s\to k}\big)^T\big(\mathbf{C}^{(i_s)}_{s\to k}\big)^{-1}\mathbf{e}^{(i_s)}_{s\to k}\big].
\end{aligned} \tag{2.41}$$

Furthermore, for $i_s = 0$, let $\tilde{\mathbf{e}}^{(0)}_{s\to k} = \mathbf{0}_{d_t}$, $\tilde{\mathbf{C}}^{(0)}_{s\to k} = \mathbf{0}_{d_t\times d_t}$ and $c^{(0)}_{s\to k} = \log(u^{(0)}_{s\to k})$. Note that (2.41) is analogous to (2.31). In (2.39), for each likelihood product term denoted by the S-length vector $\boldsymbol{i} = [i_1, \ldots, i_S] \in \boldsymbol{I}_S \triangleq \bigtimes_{s=1}^{S}[0 : I_{s\to k}]$, the quantities

$$\boldsymbol{\Xi}^{(\boldsymbol{i})} = \sum_{s=1}^{S} \tilde{\mathbf{C}}^{(i_s)}_{s\to k},\ \boldsymbol{\xi}^{(\boldsymbol{i})} = \sum_{s=1}^{S} \tilde{\mathbf{e}}^{(i_s)}_{s\to k},\ \xi^{(\boldsymbol{i})} = \sum_{s=1}^{S} c^{(i_s)}_{s\to k} \tag{2.42}$$

require information from across the network and are referred to as global information (this is in contrast to Section 2.4.2 where the analogous quantities (2.32) are locally available). As a result, $b(\mathbf{x}_k, 1)$ is GM($\mathbf{x}_k$; $\{\omega_k^{(j,\boldsymbol{i})}, \boldsymbol{\mu}_k^{(j,\boldsymbol{i})}, \boldsymbol{\Omega}_k^{(j,\boldsymbol{i})}\}_{j\in[1:J_{\to n,k}],\boldsymbol{i}\in\boldsymbol{I}_S}$) with parameters

$$\boldsymbol{\Omega}_k^{(j,\boldsymbol{i})} = \left[\big(\boldsymbol{\Omega}^{(j)}_{\to n,k}\big)^{-1} + \boldsymbol{\Xi}^{(\boldsymbol{i})}\right]^{-1} \tag{2.43}$$

$$\boldsymbol{\mu}_k^{(j,\boldsymbol{i})} = \boldsymbol{\Omega}_k^{(j,\boldsymbol{i})}\left[\big(\boldsymbol{\Omega}^{(j)}_{\to n,k}\big)^{-1}\boldsymbol{\mu}^{(j)}_{\to n,k} + \boldsymbol{\xi}^{(\boldsymbol{i})}\right] \tag{2.44}$$

$$\begin{aligned}
\omega_k^{(j,\boldsymbol{i})} &= \omega^{(j)}_{\to n,k} \exp\left(\xi^{(\boldsymbol{i})} - \frac{1}{2}\big(\boldsymbol{\mu}^{(j)}_{\to n,k}\big)^T\big(\boldsymbol{\Omega}^{(j)}_{\to n,k}\big)^{-1}\boldsymbol{\mu}^{(j)}_{s\to k}\right) \\
&\quad\times \exp\left(\frac{1}{2}\big(\boldsymbol{\mu}_k^{(j,\boldsymbol{i})}\big)^T\big(\boldsymbol{\Omega}_k^{(j,\boldsymbol{i})}\big)^{-1}\boldsymbol{\mu}_k^{(j,\boldsymbol{i})}\right)\sqrt{\frac{\det(\boldsymbol{\Omega}_k^{(j,\boldsymbol{i})})}{\det(\boldsymbol{\Omega}^{(j)}_{\to n,k})}}.
\end{aligned} \tag{2.45}$$

Again, (2.43)-(2.45) are analogous to (2.33)-(2.35) in Section 2.4.2. However, directly applying here the sequential Gibbs approach (Algorithm 2) becomes impractical. This is because the evaluation of the parameters of selected Gaussian components (indexed by $\boldsymbol{i}$) in (2.42) requires the aggregation of parameters from across the entire network. This process happens sequentially for each new label (line 15 of Algorithm 2). In a decentralized algorithm, this incurs a high communication cost and latency. Thus, a parallel sampling mechanism is favored, where agents sample local

Sequential Gibbs	Hogwild! Gibbs
$\pi_1(i'_1 \mid \boldsymbol{i}_{2:S})$ $\vdots$ $\pi_s(i'_s \mid \boldsymbol{i}'_{1:s-1}, \boldsymbol{i}_{s+1:S})$ $\vdots$ $\pi_S(i'_S \mid \boldsymbol{i}'_{1:S-1})$	In parallel: $\pi_1(i'_1 \mid \boldsymbol{i}_{\neg 1}), \cdots \ \pi_s(i'_s \mid \boldsymbol{i}_{\neg s}),$ $\cdots, \pi_S(i'_S \mid \boldsymbol{i}_{\neg S})$

Table 2.3: Conditional sampling of a new label vector $\boldsymbol{i}'$ given previous labels $\boldsymbol{i}$ in synchronous and Hogwild! Gibbs.

labels $i_s \in [0 : I_{s\to k}]$ in parallel to form a new label vector $\boldsymbol{i} = [i_1 \ldots, i_S]^T$. This is followed by synchronization, that is, the computation of the parameters (2.42) corresponding to $\boldsymbol{i}$ via average consensus.

Such sampling schemes are referred to as partially synchronous Gibbs sampling [174] or Hogwild! Gibbs [89]. In general, in Hogwild [89] or asynchronous methods, the agents perform sampling/updating as fast as they can, while periodic global synchronization is achieved across the network. The difference between the sequential Gibbs of Algorithm 2 and the Hogwild! Gibbs employed here is shown in Table 2.3. Starting from a label vector $\boldsymbol{i}$, the sequential Gibbs effectively samples new labels $\boldsymbol{i}'$ sequentially according to the marginal densities (2.38). The Hogwild! Gibbs method samples, in parallel at each agent $s \in [1, S]$, a local label i'_s conditioned on the previous labels $\boldsymbol{i}_{\neg s}$. In contrast to the sequential Gibbs, Hogwild! Gibbs requires the computation of global parameters only after all the agents have locally sampled a new index $i'_s \ \forall \ s$.

Hogwild! Gibbs for GM product

The proposed Hogwild! Gibbs algorithm produces a set of high-weight Gaussian components. The resulting GM approximates the target belief (2.39) and is presented in Algorithm 3, which is executed synchronously and in parallel at all agents for each PT. The main steps of Algorithm 3 are detailed in the following:

a) *Initialization* (line 2). Each agent s samples an initial label i_s from the local labels $[0 : I_s]$ with probability $\Pr(i_s) \propto \varrho(i_s)$, where $\varrho(0) = u^{(0)}_{s\to k} \sum_{j=1}^{J_{\to n,k}} \omega^{(j)}_{\to n,k}$, and for $i_s \neq 0$

$$\varrho(i_s) = u^{(i_s)}_{s\to k} \sum_{j=1}^{J_{\to n,k}} \omega^{(j)}_{\to n,k} \mathcal{N}\big(\mathbf{e}^{(i_s)}_{s\to k}; \mathbf{H}^{(i_s)}_{s\to k}\boldsymbol{\mu}^{(j)}_{\to n,k}, \mathbf{C}^{(i_s)}_{s\to k} + \mathbf{H}^{(i_s)}_{s\to k}\boldsymbol{\Omega}^{(j)}_{\to n,k}\big[\mathbf{H}^{(i_s)}_{s\to k}\big]^T\big). \qquad (2.46)$$

In particular, the weight $\varrho(i_s)$ of the i_s-th likelihood component from $\gamma^s_k(\cdot, 1)$, given in (2.46), is high if it leads to high-weight Gaussian components after updating the prior $\alpha_{\to n}(\mathbf{x}_k)$. Note that (2.46) is analogous to (2.37).

b) *Global parameter evaluation* (line 3). Corresponding to the selected labels $\boldsymbol{i}$, the global parameters of (2.42) are evaluated via average and max consensus [142, 46]. The weights we use are Metropolis weights [193]. Convergence is guaranteed as long as the communication graph spanning the agents is connected [46]. In practice, we stop after a sufficiently large number Q of consensus iterations. An additional max-consensus is carried out to ensure identical values for all agents. The consensus is reached across the entire network, even for agents s that do not observe the target k, i.e, for which $k \notin \mathcal{T}_{n,s}$. Note that this is a decentralized implementation of the analogous step (line 3) in Algorithm (2).

c) *Computing local Gaussian components* (lines 4-10). Given the globally computed parameters of the product indexed by $\boldsymbol{i}$ (2.42), each agent s constructs the Gaussian indexed by $(j, \boldsymbol{i}_{*q})$, with parameters given by (2.43)-(2.45). This is achieved by first replacing the i_s-th component of $\gamma^s_k(\cdot, 1)$ with the q-th component of $\gamma^s_k(\cdot, 1)$. This is done locally, since the previous label i_s and the parameters of all the $q \in [0 : I_{s\to k}]$ components of $\gamma^s_k(\cdot, 1)$ are available locally.

d) *Computing conditional probabilities and sample* (line 11). The weights of the Gaussian components indexed by $(j, \boldsymbol{i}_{*q})$, $\omega^{(j,\boldsymbol{i}_{*q})}_k$, are employed to compute the conditional density $\pi_s(q|\boldsymbol{i}_{\neg s}) \propto \sum_{j=1}^{J_{\to n,k}} \omega^{(j,\boldsymbol{i}_{*q})}_k$ from which a new local label is sampled $i_s \sim \pi_s(q|\boldsymbol{i}_{\neg s})$. This follows from (2.36) and is analogous to line 12 in Algorithm 2.

e) *Repeat for T iterations* the steps b)-d) and return the Gaussian components with distinct labels $(j, \boldsymbol{i})$ for $b(\cdot, 1)$. An additional network consensus (line 13) is required for the non-existence

Algorithm 3 Dcent. Gibbs–PT k at agent s (in parallel $\forall\, s$)

```
1:  **Input** parameters of (2.39)-(2.40).
2:  Sample $i_s \in [0 : I_{s\to k}]$ as described at Section 2.4.3-a).
3:  **Network consensus** for $\Xi^{(\boldsymbol{i})}$, $\boldsymbol{\xi}^{(\boldsymbol{i})}$ and $\xi^{(\boldsymbol{i})}$ of (2.42).
4:  **for** $q \leftarrow 0$ to $I_{s\to k}$ **do**
5:      Set $\boldsymbol{i}_{*q} \leftarrow [i_1, \cdots, i_{s-1}, q, i_{s+1}, \cdots, i_S]$.
6:      Compute $\Xi^{(\boldsymbol{i}_{*q})} = \Xi^{(\boldsymbol{i})} - \left(\mathbf{C}_{s\to k}^{(i_s)}\right)^{-1} + \left(\mathbf{C}_{s\to k}^{(q)}\right)^{-1}$, $\boldsymbol{\xi}^{(\boldsymbol{i}_{*q})} = \boldsymbol{\xi}^{(\boldsymbol{i})} - \tilde{\mathbf{e}}_{s\to k}^{(i_s)} + \tilde{\mathbf{e}}_{s\to k}^{(q)}$,
    $c^{(\boldsymbol{i}_{*q})} = \xi^{(\boldsymbol{i})} - c_{s\to k}^{(i_s)} + c_{s\to k}^{(q)}$.
7:      **for** $j \leftarrow 1$ to $J_{\to n,k}$ **do**
8:          Compute $\omega_k^{(j,\boldsymbol{i}_{*q})}$, $\boldsymbol{\mu}_k^{(j,\boldsymbol{i}_{*q})}$, $\boldsymbol{\Omega}_k^{(j,\boldsymbol{i}_{*q})}$ as in (2.43)-(2.45).
9:      **end for**
10: **end for**
11: Set $\pi_s(q|\boldsymbol{i}_{\neg s}) \propto \sum_{j=1}^{J_{\to n,k}} \omega_k^{(j,\boldsymbol{i}_{*q})}$, sample $\boldsymbol{i}_s \sim \pi_s(q|\boldsymbol{i}_{\neg s})$.
12: Repeat the steps 3-11 for $T$ iterations.
13: **Network consensus** for $\log(b_0) \triangleq \sum_{s=1}^{S} \log(\eta_k^s(0))$.
14: **Return** $\{(w_k^{(j,\boldsymbol{i})}, \boldsymbol{\mu}_k^{(j,\boldsymbol{i})}, \boldsymbol{\Omega}_k^{(j,\boldsymbol{i})})\}$ $\forall$ distinct $(S+1)$-tuples $(j, \boldsymbol{i})$ and $b_0$ to provide a GM approx-
    imation for $b(\mathbf{x}_k, r_k)$.
```

case $r_k = 0$ of (2.40), where the scalar value $\log(b_0) \triangleq \sum_{s=1}^{S} \log(\eta_k^s(0))$ is evaluated.

f) *Normalization of PT belief* (not shown in Algorithm 3) is necessary in order to obtain an approximate pdf for PT k. The pdf of PT k is given as $f_k(\mathbf{x}_k, 1) = \frac{1}{N}\sum_{(j,\boldsymbol{i})} \omega_k^{(j,\boldsymbol{i})} \mathcal{N}(\mathbf{x}_k; \boldsymbol{\mu}_k^{(j,\boldsymbol{i})}, \boldsymbol{\Omega}_k^{(j,\boldsymbol{i})})$ and $f_k(\mathbf{x}_k, 0) = \frac{b_0}{N}\alpha_k(\mathbf{x}_k, 0)$ where $N = b_0\left[1 - \sum_{j=1}^{J_{\to n,k}} \omega_{\to n,k}^{(j)}\right] + \sum_{(j,\boldsymbol{i})} \omega_k^{(j,\boldsymbol{i})}$ is the normalization constant.

During the consensus step in line 3 of Algorithm 3, it is assumed that agent s learns the labels $\boldsymbol{i}_l$ for all $l \in \mathcal{A} \setminus \{s\}$. This can be achieved by diffusing the scalars $\boldsymbol{i}_l$ throughout the network and which involves only a mild increase in communication load as compared to the average consensus communication requirements. Note however that only the values taken by the global parameters $\Xi^{(\boldsymbol{i})}$, $\boldsymbol{\xi}^{(\boldsymbol{i})}$, $\xi^{(\boldsymbol{i})}$ are necessary for the computation of local Gaussian components, the conditional probabilities and the ensuing sampling. The label values are only necessary for returning the distinct Gaussian components, i.e., for district $(S+1)$-tuples $(j, \boldsymbol{i})$. An alternative decentralized algorithm that avoids the diffusion of the label values i_s is possible by modifying Algorithm 3 to return only the Gaussian components with distinct weights $\omega_k^{(j,\boldsymbol{i})}$, as these form a subset of the set of Gaussian components returned by Algorithm 3.

Complexity and Convergence

The time complexity of Algorithm 3 is $O(TJ_{\rightarrow n,k}I_{s\rightarrow k}d_t^3)$. Assuming Q average consensus iterations and denoting with $D_{\mathcal{G}}$ the diameter of the communication graph, the communication load of the consensus step of Algorithm 3 is $(Q + D_{\mathcal{G}})[T(d_t^2 + d_t + 1) + 1]$ real values and also incurs a latency of $(Q + D_{\mathcal{G}})(T + 1)$ communication slots. Note that the Gibbs method of Algorithm 3 does not represent a Markov chain as the agents sample in parallel and not sequentially as in Algorithm 2. The existence of a stationary distribution as well as the convergence of the samples drawn with Algorithm 3 to such a stationary distribution is not guaranteed outside of special cases [89]. Nonetheless, Hogwild! Gibbs methods have been successfully employed in latent Dirichlet Allocation [162]. In Section 2.6, we numerically show the performance of the SCS-MTT filter with Algorithm 3 to be close to that of the centralized filter, where a fusion center has access to all the measurements, and carries out all the computations.

Computation of the GM extrinsic information $\delta_k^s(\cdot)$

The computation of the δ_k^s messages in (2.22) requires again the product of several GM likelihood terms available at different agents in the network. Note that since both $b_k(\mathbf{x}_k, 1)$ (obtained via Algorithm 3) and $\gamma_k^s(\mathbf{x}_k, 1)$ are available as GMs at agent s, a GM approximation for the extrinsic information δ_k^s can be constructed in the following manner. First note from line 6 of Algorithm 3, that the parameters $\mathbf{\Xi}^{(\boldsymbol{i})} - (\mathbf{C}_{s\rightarrow k}^{(i_s)})^{-1}$, $\boldsymbol{\xi}^{(\boldsymbol{i})} - \tilde{\mathbf{e}}_{s\rightarrow k}^{(i_s)}$ and $\xi^{(\boldsymbol{i})} - c_{s\rightarrow k}^{(i_s)}$ characterize the product $\prod_{\ell\neq s} u_{\ell\rightarrow k}^{(i_\ell)}\mathcal{N}(\mathbf{e}_{\ell\rightarrow k}^{(i_\ell)}; \mathbf{H}_{\ell\rightarrow k}^{(i_\ell)}\mathbf{x}_k, \mathbf{C}_{\ell\rightarrow k}^{(i_\ell)})$ for a label vector $\boldsymbol{i}$. Thus, at each iteration of Algorithm 3, we can construct the following Gaussian components $\{(\omega_{k\rightarrow s}^{\delta,(j)}, \boldsymbol{\mu}_{k\rightarrow s}^{\delta,(j)}, \mathbf{\Omega}_{k\rightarrow s}^{\delta,(j)})\}_{j=1}^{J_{k\rightarrow s}}$ of $\delta_k^s(\mathbf{x}_k, 1)$ (with the same notations as in Figure 2.2) as

$$
\begin{aligned}
\mathbf{\Omega}_{k\rightarrow s}^{\delta,(j)} &= \big[(\mathbf{\Omega}_{\rightarrow n,k}^{(j)})^{-1} + \mathbf{\Xi}^{(\boldsymbol{i})} - (\mathbf{C}_{s\rightarrow k}^{(i_s)})^{-1}\big]^{-1},\\
\boldsymbol{\mu}_{k\rightarrow s}^{\delta,(j)} &= \mathbf{\Omega}_{k\rightarrow s}^{\delta,(j)}\big[(\mathbf{\Omega}_{\rightarrow n,k}^{(j)})^{-1}\boldsymbol{\mu}_{\rightarrow n,k}^{(j)} + \boldsymbol{\xi}^{(\boldsymbol{i})} - \tilde{\mathbf{e}}_{s\rightarrow k}^{(i_s)}\big],\\
\omega_{k\rightarrow s}^{\delta,(j)} &= \omega_{\rightarrow n,k}^{(j)} \exp\Big(\xi^{(\boldsymbol{i})} - \frac{1}{2}\big(\boldsymbol{\mu}_{\rightarrow n,k}^{(j)}\big)^T\big(\mathbf{\Omega}_{\rightarrow n,k}^{(j)}\big)^{-1}\boldsymbol{\mu}_{\rightarrow n,k}^{(j)}\Big)
\end{aligned}
$$

$$\times \exp\left(\frac{1}{2}(\boldsymbol{\mu}_{k\to s}^{\delta,(j)})^T(\boldsymbol{\Omega}_{k\to s}^{\delta,(j)})^{-1}\boldsymbol{\mu}_{k\to s}^{\delta,(j)} - c_{s\to k}^{(i_s)}\right)\sqrt{\frac{\det(\boldsymbol{\Omega}_{k\to s}^{\delta,(j)})}{\det(\boldsymbol{\Omega}_{\to n,k}^{(j)})}}.$$

Note that the expressions above are analogous to the GM parameters for $b(\mathbf{x}_k, 1)$ in (2.43)-(2.45). The only difference being the absence of the terms (2.41) corresponding to $\gamma_k^s(\mathbf{x}_k, 1)$ After the T iterations of Algorithm 3, the Gaussian components with highest distinct weights $\omega_{k\to s}^{\delta,(j)}$ are retained to form an approximation of $\delta_k^s(\mathbf{x}_k, 1)$ while $\delta_k^s(\mathbf{x}_k, 0)$ is obtained as $b(\mathbf{x}_k, 0)/\eta_k^s(0)$. This procedure allows for the local computation of an approximate GM representation for $\delta_k^s(\mathbf{x}_k, r_k)$ without additional network consensus operations.

2.5 Decentralized Gaussian SCS-MTT filter

A special case of the GM SCS-MTT filter of the previous section is obtained when all the agent and target densities are represented as single Gaussians. For most kinematic tracking applications, the communication loads of the resulting DG-SCS-MTT filter is significantly smaller than the PF implementations. Single Gaussian expressions for the messages exchanged by the SCS-MTT filter can be obtained by specializing the expressions in Section 2.4. However, due to the measurement-to-target association uncertainty (see assumption (A8) of Section 2.2.2 and the summation over m in (2.17), (2.20)), the agent and target beliefs become GMs even if their predicted messages are single Gaussians. The Probabilistic Data Association filter [12], addresses this by performing a single Gaussian approximation of the resulting GM via first and second order moment matching. This is applied straightforwardly to the case of the agent beliefs $b_s(\cdot)$ and their extrinsic information $\theta_k^s(\cdot)$ as their GM computation is done locally as shown in Section 2.4.2.

The DG-SCS-MTT filter achieves a single Gaussian representation of the target beliefs with a lower communication load than the Hogwild! Gibbs of Algorithm 3. Suppose $b_s(\mathbf{x}_k, 1) \triangleq \sqrt[S]{\alpha_{\to n}(\mathbf{x}_k, 1)}\gamma_k^s(\mathbf{x}_k, 1)$. Observe that (2.21) becomes $b(\mathbf{x}_k, 1) = \prod_{s\in\mathcal{S}} b_s(\mathbf{x}_k, 1)$. If $\alpha_{\to n}(\mathbf{x}_k, 1) = c_k\mathcal{N}(\mathbf{x}_k; \mathbf{m}, \mathbf{P})$ then $\sqrt[S]{\alpha_{\to n}(\mathbf{x}_k, 1)} = c'_k\mathcal{N}(\mathbf{x}_k; \mathbf{m}, S\mathbf{P})$ is a scaled Gaussian [15, eq. 36], where $c'_k = \sqrt[S]{c_k}\frac{(\det(2\pi S\mathbf{P}))^{1/2}}{\sqrt[2S]{\det(2\pi\mathbf{P})}}$. Furthermore, let $\gamma_k^s(\mathbf{x}_k, 1)$ be a GM of the form (2.29). Then, a locally computed GM $b_s(\mathbf{x}_k, 1) = \sum_{i=1}^{I_s} w_s^{(i)} \mathcal{N}(\mathbf{x}_k; \mathbf{m}_s^{(i)}, \mathbf{P}_s^{(i)})$ is given as a special case of (2.39) with $J_{\to n,k} = S = 1$. The scaled single Gaussian $\hat{b}_s(\mathbf{x}_k, 1) = \hat{c}_s \mathcal{N}(\mathbf{x}_k; \hat{\mathbf{m}}_s, \hat{\mathbf{P}}_s)$ that matches the first and second order moments of the GM $b_s(\mathbf{x}_k, 1)$ has parameters [143]

$$\hat{c}_s = \sum\nolimits_{i=1}^{I_s} w_s^{(i)}, \qquad \hat{\mathbf{m}}_s = \frac{1}{\hat{c}_s} \sum\nolimits_{i=1}^{I_s} w_s^{(i)} \mathbf{m}_s^{(i)}, \tag{2.47}$$

$$\hat{\mathbf{P}}_s = \frac{1}{\hat{c}_s} \sum\nolimits_{i=1}^{I_s} w_s^{(i)} \big[\mathbf{P}_s^{(i)} + (\mathbf{m}_s^{(i)} - \hat{\mathbf{m}}_s)(\mathbf{m}_s^{(i)} - \hat{\mathbf{m}}_s)^T\big]. \tag{2.48}$$

Note that (2.48) also accounts for the spread of the means of the initial GM. A global $\hat{b}(\mathbf{x}_k, 1) = \hat{c}\mathcal{N}(\mathbf{x}_k; \hat{\mathbf{m}}, \hat{\mathbf{P}})$, as a single Gaussian approximation of $b(\mathbf{x}_k, 1)$, is obtained via network (average and max) consensus over the weights $\log(\hat{c}) = \sum_{s=1}^{S} \log(\hat{c}_s)$, matrices $\hat{\mathbf{P}}^{-1} = \sum_{s=1}^{S} \hat{\mathbf{P}}_s^{-1}$ and vectors $\hat{\mathbf{m}} = \hat{\mathbf{P}} \sum_{s=1}^{S} \hat{\mathbf{P}}_s^{-1} \hat{\mathbf{m}}_s$. The computation of $b(\mathbf{x}_k, 0)$ remains unchanged from Section 2.4.3. In contrast to the Hogwild! Gibbs of Algorithm 3, the DG-SCS-MTT filter only performs network consensus once for each PT which involves an exchange of $(Q + D_{\mathcal{G}_k})(d_t^2 + d_t + 2)$ real values at each outer-BP loop. Furthermore, we note that computing the local GM belief $b_s(\cdot, 1)$ takes $O(I_s d_t^3)$ operations; single Gaussian compression is $O(I_{s\to k} d_t^2)$; and the computations required for average consensus are $O(Q d_t^2)$ (assuming the number of neighbors of an agent is small compared to Q). Hence, the overall computational complexity of the DG-SCS-MTT filter is $O(I_{s\to k} d_t^3)$ for each PT, at each outer loop iteration.

The DG-SCS-MTT filter also achieves a single Gaussian approximation for the extrinsic information message $\delta_k^s(\cdot)$ without the need of additional network consensus operations. Based on the parameters of $b(\mathbf{x}_k, 1)$, we evaluate $b_{\neg s}(\mathbf{x}_k, 1) \triangleq \hat{c}_{\neg s} \mathcal{N}(\mathbf{x}_k; \hat{\mathbf{m}}_{\neg s}, \hat{\mathbf{P}}_{\neg s})$ where the parameters $\hat{c}_{\neg s} = \hat{c}/\hat{c}_s$, $\hat{\mathbf{P}}_{\neg s}^{-1} = \hat{\mathbf{P}}^{-1} - \hat{\mathbf{P}}_s^{-1}$, and $\hat{\mathbf{m}}_{\neg s} = \hat{\mathbf{P}}_{\neg s} \left[\hat{\mathbf{P}}^{-1} \hat{\mathbf{m}} - \hat{\mathbf{P}}_s^{-1} \hat{\mathbf{m}}_s\right]$ are computed locally. Finally, we evaluate $\delta_k^s(\mathbf{x}_k, 1) = b_{\neg s}(\mathbf{x}_k, 1) \sqrt[S]{\alpha(\mathbf{x}_k, 1)}$, which has a scaled single Gaussian form, and $\delta_k^s(\mathbf{x}_k, 0) = b(\mathbf{x}_k, 0)/\eta_k^s(0)$.

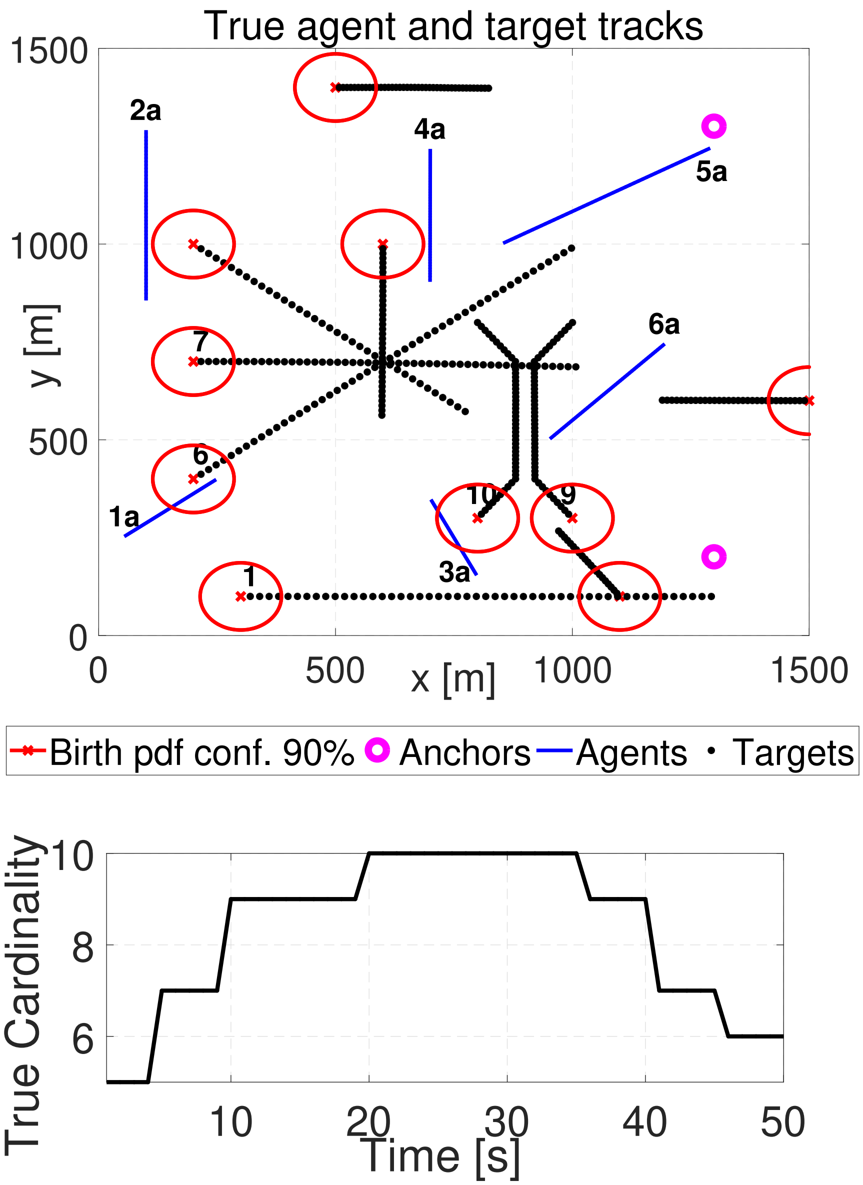


Fig. 2.3: Ground truth trajectories of agents and targets plotted over time (top), along with the true target cardinality over time (bottom).

2.6 Simulation Results

In this section, we numerically evaluate the performance of our proposed GM-SCS-MTT and G-SCS-MTT filters for both decentralized and centralized versions[2]

The centralized GM (CGM-SCS-MTT) version employs the centralized Gibbs Algorithm 2

---

[2] We have relegated the simulation results for a PF-based implementation of the centralized algorithm (based on the discussion in Section 2.3, and assuming all the observations are available at a single location) to Appendix A.1.

for both agent and PT beliefs while the decentralized GM (DGM-SCS-MTT) filter employs the Hogwild! Gibbs method of Algorithm 3 for PT beliefs. We also consider a reference method, named here SPAWN, which consists of the agent self-localization method of [192] followed by the MTT method of [126]. Centralized GM (CGM-SPAWN) and single Gaussian (CG-SPAWN) versions of SPAWN are employed, where the GM product of messages is computed using the centralized Gibbs Algorithm 2.

Figure 2.3 shows the ground truth tracks of all the agents and targets, over a span of 50 time steps and the true target cardinality as a function of time. The uncertainty in target births is shown as red ellipses that delineate the area containing 90% of the mass of $f_b(\mathbf{x}_{n,k})$. Our network has two stationary agents (called anchors), 6 mobile agents, and a maximum of 10 targets over a $[0, 1500\text{m}] \times [0, 1500\text{m}]$ region of interest (ROI). Each mobile agent has a measurement and communication range of 1000m. The anchors have a communication range of 1000m and a measurement range of 1500m.

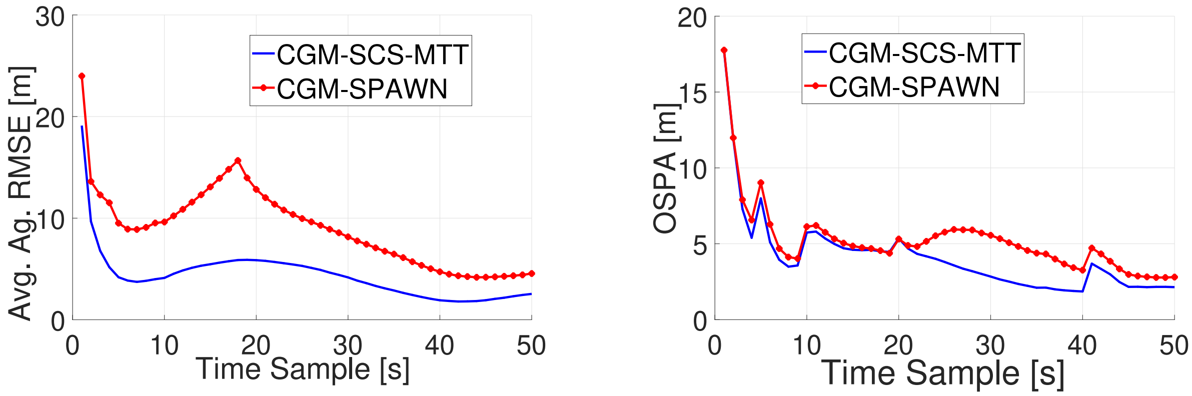
Agent and target state vectors are constructed as $\mathbf{x} = [p_x,\, p_y,\, \dot{p}_x,\, \dot{p}_y]^T$, where p_x and p_y represent the x-y target coordinates and $\dot{p}_x$ and $\dot{p}_y$ are its velocity components along the two axes. All targets have the same dynamical model $f(\mathbf{x}_n|\mathbf{x}_{n-1}) = \mathcal{N}(\mathbf{x}_n; \mathbf{B}_n\mathbf{x}_{n-1}, \mathbf{\Sigma}_n)$, where the state transition matrix is $\mathbf{B}_n = \left[\begin{smallmatrix} \mathbf{I}_2 & T_S\mathbf{I}_2 \\ \mathbf{0}_2 & \mathbf{I}_2 \end{smallmatrix}\right]$ with a sampling period of $T_s = 1$s and $\mathbf{0}_n$ and $\mathbf{I}_n$ are the zero and identity matrices of size $n \times n$. The covariance matrix is $\mathbf{\Sigma}_n = \sigma_q^2 \left[\begin{smallmatrix} 0.25T_S^4\mathbf{I}_2 & 0.5T_S^3\mathbf{I}_2 \\ 0.5T_S^3\mathbf{I}_2 & T_S^2\mathbf{I}_2 \end{smallmatrix}\right]$, with $\sigma_q = 0.5$. Similarly, all agents have the same linear-Gaussian kinematic model with $\sigma_q = 0.1$. Each agent s, with coordinates (p_x^s, p_y^s), observes with probability P_D^s a target with state vector $\mathbf{x}$ through a range-bearing model:

$$\mathbf{z}_n^s = \begin{bmatrix} \sqrt{(p_x - p_x^s)^2 + (p_y - p_y^s)^2} \\ \tan^{-1}(\frac{p_y - p_y^s}{p_x - p_x^s}) \end{bmatrix} + \mathbf{n}_n^s, \tag{2.49}$$

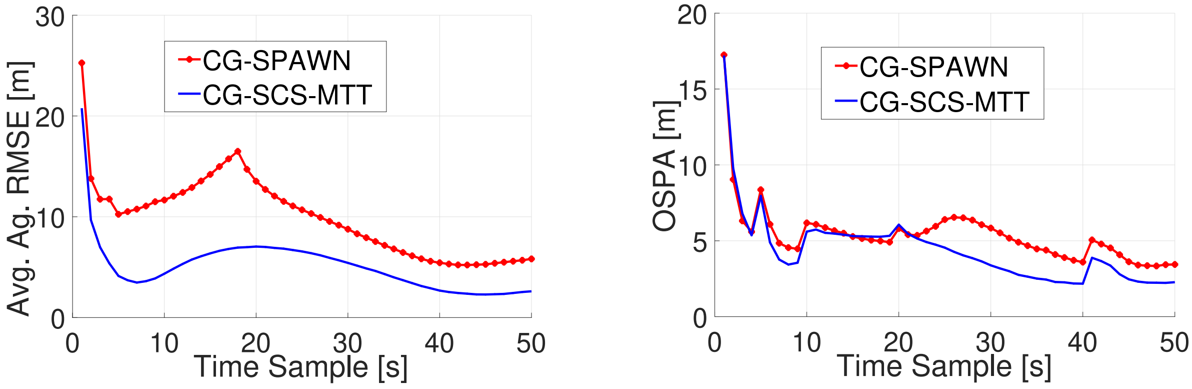
where the measurement noise is $\mathbf{n}_n^s \sim \mathcal{N}(0, \mathbf{R}_n)$. The same range-bearing measurement model (with potentially different parameter values) is employed for inter-agent measurements. Linearization of the non-linear range-bearing observation model is performed before applying the GM or Gaussian SCS-MTT filter. Similar to the extended Kalman filter [150, Ch. 2.1], this is achieved locally at each agent by evaluating the Jacobian of the transformation (2.49) at the weighted mean of the PT prediction message $\alpha_{\to n}(\cdot)$. Analogously, the inter-agent range-bearing measurement model is linearized with the Jacobian being evaluated at the mean of the agent prediction message $\phi_{\to n}(\cdot)$. Birthed PTs are appended to the existing PTs in the prediction step of the filters. The birth locations are shown in Figure 2.3. Unless stated otherwise, the birth probabilities of existence are set to 0.25, the probability of target survival $P_s = 0.99$, the probability of target detection $P_D^s = 0.95$, and the measurement noise covariance $\mathbf{R}_n = \text{diag}(10, 100)$. At each frame, the clutter process for agent s follows a Poisson distribution with rate $\lambda^s = 25$, and the clutter points are distributed uniformly over the ROI. Target inference is achieved as indicated in Section 2.3.2. The number of outer BP iterations is fixed to $P = 1$, to avoid over-confident beliefs [127].

In the GM filters, the initial positions of the agents are modeled using Gaussian mixture densities. More precisely, each agent track is initialized with 4 equal-weighted GM components, with means at a distance of $R_a = 50$m along the x and y directions, from the position shown in Figure 2.3. All the 4 GM components have the same covariance diag$(1600, 1600, 40, 40)$. In the single Gaussian filters, the mean and covariance of the agent tracks are initialized by the respective values achieved via moment matching [143]. Target tracks are initialized with single Gaussian densities in all filters, with means given by the birth locations and covariance matrices diag$(1600, 1600, 16, 16)$.

Keeping the agent and target tracks fixed, 100 independent Monte Carlo (MC) simulation runs are carried out by regenerating the measurement sets. In Figures 2.4, 2.5, 2.7, we have plotted: (i) the average root mean squared errors (RMSE) in the agent location estimates (averaged across the MC runs and across all the mobile agents); and, (ii) the average target tracking performance via the Optimum Sub-Pattern Assignment (OSPA) error [155]. The OSPA metric is capable of taking into account errors in estimating both the number of targets (i.e., cardinality) and their tracks. The OSPA employs two parameters: cut-off, set to 20m and order, set to 1. In Figure 2.6, we have explicitly plotted the average estimated cardinality over time.



(a) CGM-SCS-MTT vs CGM-SPAWN filters

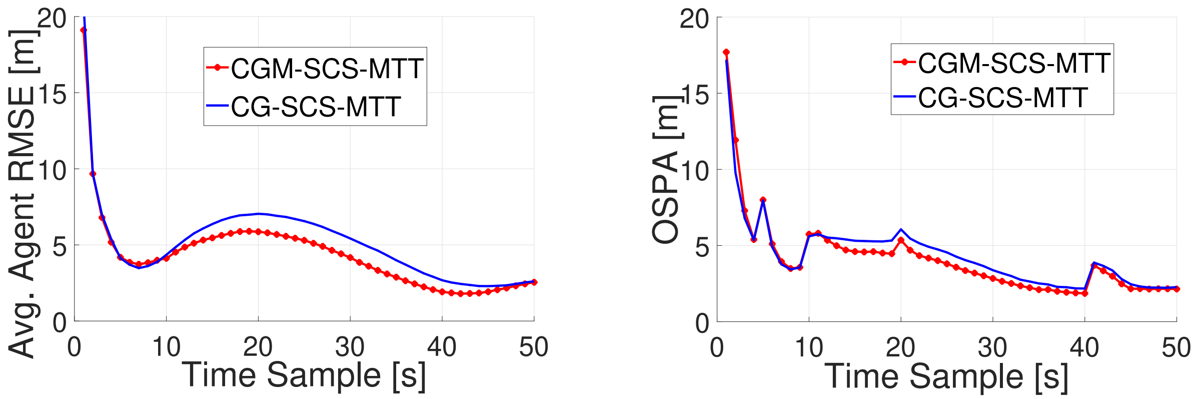


(b) CG-SCS-MTT vs CG-SPAWN filters

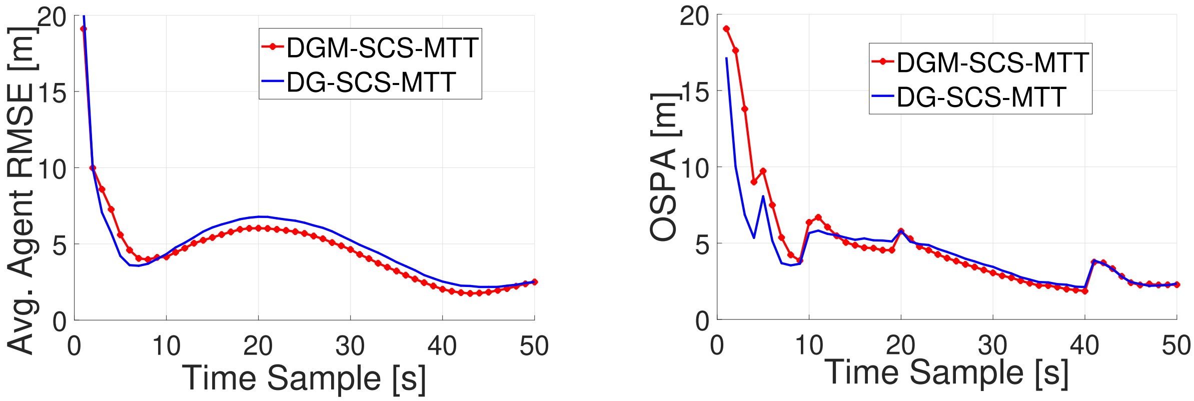
Fig. 2.4: Comparison of the average agent RMSE and target OSPA error for SPAWN and the proposed SCS-MTT filter.

2.6.1 SCS-MTT vs SPAWN

In Figure 2.4, we compare the average agent localization error, and the average OSPA error for targets, of our approach (SCS-MTT) against SPAWN. Figure 2.4a shows the comparison when the agent and target densities are modeled as Gaussian mixtures. Figure 2.4b showcases the same comparison for single Gaussian densities. Our approach significantly improves the localization performance by taking into account the contribution of the Λ_k^s messages from the within-range targets to the agents. This would be especially beneficial for agents which are not in range of the anchor nodes, and have few neighboring agents (e.g., agents $1a, 2a$). Due to the presence of anchors, the agent localization improvements of the SCS-MTT algorithms transpire to a lesser extent into improvements on target tracking performance. The spikes in OSPA error correspond to



(a) CGM-SCS-MTT vs CG-SCS-MTT (centralized filters)

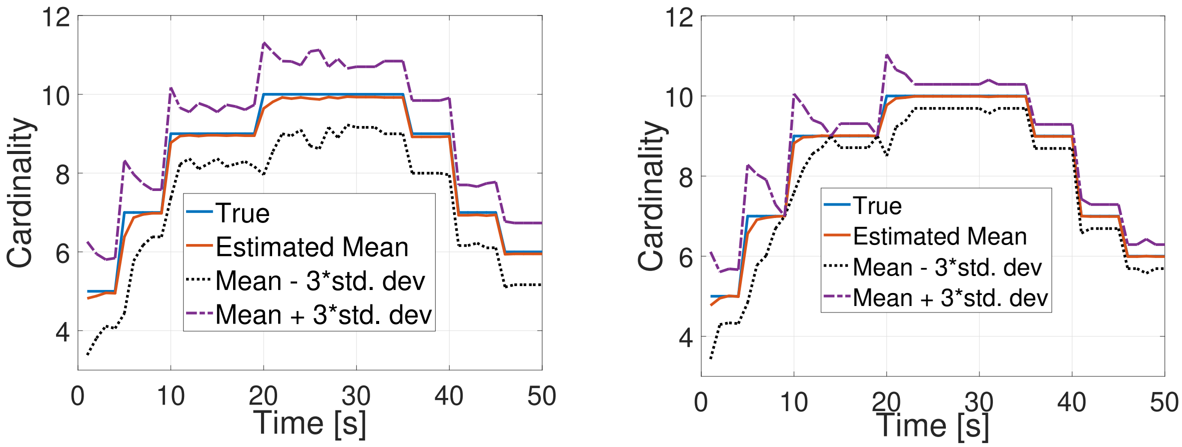


(b) DGM-SCS-MTT vs DG-SCS-MTT (decentralized filters)

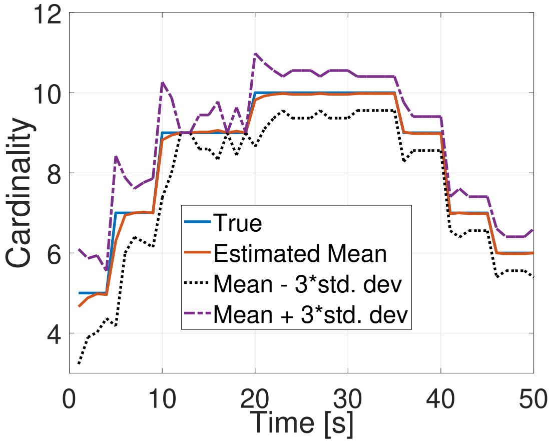
Fig. 2.5: Comparison of the average agent RMSE and target OSPA error, for the single Gaussian (G) and Gaussian mixture (GM) filters.

the time instants of target births ($t = 5, 10, 20$s) and deaths ($t = 40$). Note that due to the dynamic nature of the network (frequent target births and deaths), the localization performance cannot be expected to converge over time. This is also evident from the slight increase in the localization error after $t = 40$s, in Figures 2.4 and 2.5.

Figure 2.6 shows the true cardinality, the mean estimated cardinality, and mean $\pm 3\times$ standard deviation curves for the CGM-SPAWN (Figure 2.6a), CGM-SCS-MTT (Figure 2.6b) and DGM-SCS-MTT (Figure 2.6c) filters. In all cases, the mean estimated cardinality is close to the true cardinality with the CGM-SPAWN filter having higher cardinality variance. Both the centralized and decentralized GM-SCS-MTT filters have smaller cardinality variance than CGM-SPAWN, while the DGM-SCS-MTT filter has a slightly higher variance than the CGM-SCS-MTT filter. This is attributed to the differences between the sequential Gibbs and the parallel Hogwild! Gibbs



(a) CGM-SPAWN filter **(b)** CGM-SCS-MTT filter



(c) DGM-SCS-MTT filter

Fig. 2.6: Comparison of cardinality estimates for the different GM filters.

samplers and to the network consensus process.

2.6.2 Single Gaussian vs Gaussian mixture SCS-MTT filters

In Figure 2.5, we present the performance of the GM-SCS-MTT, which employs GM representations for both target and agent beliefs, with respect to the single Gaussian G-SCS-MTT filter. Figure 2.5a shows this comparison for the centralized SCS-MTT filters. Figure 2.5b showcases the same comparison for the decentralized SCS-MTT filters. The GM-SCS-MTT filters exhibit only a slight gain in terms of localization and tracking performance as compared to the G-SCS-MTT filters. This is because the agent and target dynamic models, as discussed in Section 2.4, are linear with additive Gaussian noise. Additionally, the considered measurement model is only moderately

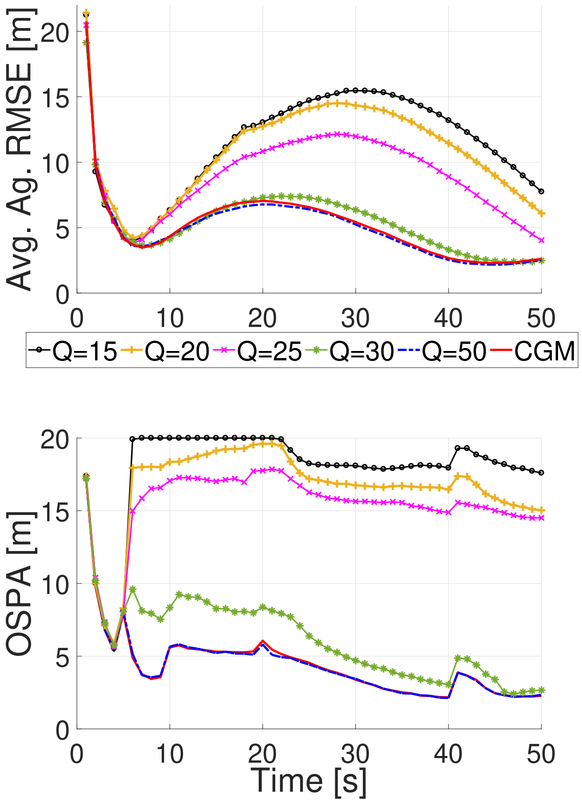


Fig. 2.7: Comparison of average agent RMSE and target OSPA error for the DGM-SCS-MTT and CGM-SCS-MTT filters.

nonlinear. Applying our GM-SCS-MTT filter to a highly nonlinear and/or non-Gaussian setting is one of the directions we wish to pursue in a subsequent study.

2.6.3 Centralized vs decentralized Gaussian mixture

In Figure 2.7, we compare the performance of the centralized (CGM-SCS-MTT) and decentralized (DGM-SCS-MTT) filters. The decentralized method achieves performance similar to the centralized filter, given a sufficient number of consensus iterations Q. Compared to the centralized approach, the decentralized approach does not have to rely on a central fusion center and is scalable with the number of sensors. We have plotted the average localization and tracking error for different values of Q, namely $Q = 15, 20, 25, 30, 50$. For $Q = 50$, the DGM filter performance is almost identical to the CGM filter. For small Q, since the network is sparsely connected (some agents have only one neighboring agent), the target belief product of (2.39)-(2.40) is not accurately evaluated. This leads to poor tracking of targets which subsequently leads to poor localization of agents, due to the interdependence of localization and tracking for SCS-MTT algorithms. Similar results and conclusions hold for the single Gaussian case, which is omitted due to space constraints.

2.7 Summary

In this chapter, we proposed a novel decentralized method for simultaneous agent localization and multi-target tracking, for an unknown number of targets, under measurement-origin uncertainty. We proposed decentralized single-Gaussian as well as Gaussian-mixture implementations for our proposed filter. The two cases capture the trade-off between computational and communication efficiency (single Gaussian) and modeling accuracy (Gaussian mixtures). For the Gaussian-mixture case, we proposed a novel decentralized Gibbs method for efficiently computing products of Gaussian mixtures. We have demonstrated the robustness of our approach using a challenging range-bearing measurement model, which showcases the improved performance of the proposed methods with respect to the SPAWN method that performs agent localization and target tracking separately.

CHAPTER 3

DISTRIBUTED ONLINE CONVEX OPTIMIZATION

3.1 Introduction

Many problems of practical interest, including network resource allocation [30], target tracking [156], network routing [209], online regression [157], and spam filtering [76] can be framed in an Online Convex Optimization (OCO) framework. The OCO framework first introduced in [209] aims to minimize a time varying convex objective function which is revealed to the observer in a sequential manner. For a detailed review of OCO, please see [76, 157]. In this chapter, we consider a distributed constrained OCO problem, with time-varying (potentially adversarial) constraints.

Recently, distributed OCO frameworks have gained popularity as they distribute the computational and memory resources across multiple nodes rather than having a central node perform all the operations [99, 156, 144, 111, 199]. Here we consider the constrained OCO problem in a distributed framework, where the convex objective is assumed to be decomposed and distributed across a set of multiple communicating agents. Each agent takes its own action with the goal of minimizing the dynamically varying global function while satisfying its individual constraints. Next, we discuss the related work along with the performance metrics we use to evaluate the performance of the proposed algorithm.

3.1.1 Related Work

Regret: The performance in OCO problems is quantified in terms of how well the agent does as compared to an offline system, over time. In other words, how much the agent "regrets" not having the information, which was revealed to it post-hoc, to begin with. Since regret is cumulative over time, an algorithm that achieves sub-linear increase in regret with time, asymptotically achieves zero average loss. It is naturally desirable to compare against an offline system, the action(s) of which are "optimal" in some sense.

Static Regret: The initial work on OCO, starting with [209, 157, 76], almost exclusively focused on *static regret* $\mathbf{Reg}_T^s$, which uses an optimal *static* solution, in hindsight, as the benchmark. In other words, the fictitious offline adversary w.r.t. which the online system measures its regret, chooses the best fixed strategy, assuming it has access to the entire information, which is revealed to the online system over time horizon T.

$$\mathbf{Reg}_T^s \triangleq \sum_{t=1}^{T} f_t(\mathbf{x}_t) - \min_{\mathbf{x}} \sum_{t=1}^{T} f_t(\mathbf{x}).$$

Under standard regularity conditions, for general OCO problems, a tight upper bound of $O(\sqrt{T})$ has been shown for static regret [209, 75]. However, for applications such as online parameter estimation or tracking moving targets, where the quantity of interest also evolves over time, comparison with a static benchmark is not sufficient.

This deficiency led to the development of *dynamic regret* $\mathbf{Reg}_T^d$ [73, 20]. Rather than comparing the performance relative to a fixed optimal strategy, a more demanding benchmark is used. More precisely, at each time instant, our fictitious adversary utilizes one-step look-ahead information to adopt the optimal strategy at the current time instant.

$$\mathbf{Reg}_T^d \triangleq \sum_{t=1}^{T} f_t(\mathbf{x}_t) - \sum_{t=1}^{T} \min_{\mathbf{x}} f_t(\mathbf{x}).$$

In this work, we adopt the notion of dynamic regret as the performance metric. It must, however, be noted that, in the worst case, it is impossible to achieve sublinear dynamic regret [209]. For such problems, the growth of dynamic regret is captured by the regularity measure which measures variations of the minimizer sequence over time (see C_T^* in Theorem 3.5.5).

Constraints: The conventional approaches for OCO are based on projection-based gradient descent-like algorithms. However, when working with functional inequality constraints $g_t(\mathbf{x}) \leq \mathbf{0}$ (as opposed to simple convex feasible set constraints), the projection step in itself is computationally intensive. This led to the development of primal-dual algorithms for OCO [121, 85, 204]. Instead of attempting to satisfy the constraints at each time instant, the constraints are satisfied in the long run. In other words, the cumulative accumulation of instantaneous constraint violations (often simply called **fit**) $\|[\sum_{t=1}^{T} g_t(\mathbf{x}_t)]_+\|$ is shown to be sublinear in T. This formulation allows constraint violations at some instants to be "taken-care-of" by strictly feasible actions at other times.[1]

Initially the constraints were assumed to be static across time [121, 85]. However, subsequent literature [171, 30] demonstrated that the analysis for primal-dual methods can be generalized to even handle time-varying inequality constraints. Minor variations of primal-dual methods, which replace the dual update step with virtual-queue (modified Lagrange multiplier) updates have also been proposed to handle time-varying [25] and stochastic constraints [202].

Distributed OCO Problems: So far we have only discussed centralized problems. Suppose the OCO system has a network of agents, and local cost (and possibly constraint) functions are revealed to each agent over time. The global objective is to minimize the total cost function, while also satisfying all the constraints. And each agent can only communicate with those agents that are in its immediate neighborhood. This distributed OCO problem is more challenging and much less studied in the literature than the centralized problem.

Distributed OCO problems with static set constraints have been widely studied in recent years [99, 156, 144, 111, 199]. Again here, the literature on distributed OCO with dynamic regret is

---

[1] Some more recent works [204] have considered the more stringent constraint violation metric $\sum_{t=1}^{T}([g_t(\mathbf{x}_t)]_+)^2$.

much sparser than for static regret. The authors in [156] have proposed a dynamic mirror descent based algorithm, where primal update steps are alternated with local consensus steps. The authors in [111] have proposed a distributed primal-dual algorithm for the OCO problem with coupled inequality constraints. The constraint functions are static over time. This has been generalized for time-varying coupled constraints in [199], where the authors have shown sublinearity of regret and fit, both w.r.t. dynamic and static benchmarks. However, to the best of our knowledge, the distributed OCO problem with a dynamic benchmark, even with static non-coupled inequality constraints has so far not been considered in the literature.

3.1.2 Our Contributions

In this work, we consider a distributed online convex optimization problem, where both the cost functions and the time-varying inequality constraints are revealed locally to the individual nodes. We propose a primal-dual mirror-descent based algorithm, which alternates between the local primal and dual update steps and the consensus steps to mix the local primal variables with the immediate neighbors. Importantly, we show that the proposed algorithm achieves sublinear dynamic regret and fit.

The chapter is organized as follows: the problem formulation is discussed in Section 3.2, along with the definitions of the performance metrics. In Section 3.3, we provide some background results and the assumptions required for providing theoretical guarantees. We propose our primal-dual mirror descent based algorithm in Section 3.4, followed by the theoretical results in Section 3.5. Finally, we conclude in Section 3.6.

Notations: Vectors are denoted with lowercase bold letters, e.g., $\mathbf{x}$, while matrices are denoted using uppercase bold letters, e.g., $\mathbf{X}$. The set of positive integers is represented by $\mathbb{N}_+$. We use $\mathbb{R}^n_+$ to denote the n-dimensional non-negative orthant. For $n \in \mathbb{N}_+$, the set $\{1, \dots, n\}$ is denoted by $[n]$. We denote by $\|\cdot\|$ the Euclidean norm for vectors, and the induced 2-norm for matrices. $\mathbf{0}$ denotes a zero vector, where the dimension is clear from the context. $[\mathbf{x}]_+$ denotes the projection onto $\mathbb{R}^n_+$.

3.2 Problem Formulation

We consider a network of n agents. At each time instant t, each agent i takes an action $\mathbf{x}_{i,t} \in \mathcal{X} \subseteq \mathbb{R}^d$, where the set $\mathcal{X}$ is fixed across time, across all the nodes. Then, a set of local loss functions $\{f_{i,t}(\cdot)\}_{i=1}^n$ with $f_{i,t} : \mathcal{X} \to \mathbb{R}$ are revealed to the individual nodes, leading to individual loss $f_{i,t}(\mathbf{x}_{i,t})$ at node i. Additionally, another set of local functions $\{g_{i,t}(\cdot)\}_{i=1}^n$ with $g_{i,t} : \mathcal{X} \to \mathbb{R}^m$ are revealed, corresponding to local constraints $g_{i,t}(\mathbf{x}_{i,t}) \leq \mathbf{0}$. The network objective is to minimize the global average of the local cost functions $f_t(\mathbf{x}) \triangleq \frac{1}{n}\sum_{i=1}^n f_{i,t}(\mathbf{x})$, while also satisfying all the local constraint functions $\{g_{i,t}(\cdot)\}_{i=1}^n$.

$$\min_{\mathbf{x}_t \in \mathcal{X}} \; f_t(\mathbf{x}_t) \triangleq \sum_{i=1}^n f_{i,t}(\mathbf{x}_t) \qquad \text{subject to } g_{i,t}(\mathbf{x}_t) \leq \mathbf{0}_m, \forall\, i \in [n]. \tag{3.1}$$

Since the objective is to minimize the global function $f_t(\cdot)$, the nodes need to communicate among themselves. We next define the metrics used to measure the performance of the proposed approach.

3.2.1 Performance Metrics - Dynamic Regret and Fit

We use the recently defined notion of dynamic regret [20, 73] to measure the performance relative to a time-varying benchmark.

$$\mathbf{Reg}_T^d \triangleq \frac{1}{n}\sum_{i=1}^n \sum_{t=1}^T f_t(\mathbf{x}_{i,t}) - \sum_{t=1}^T f_t(\mathbf{x}_t^*), \tag{3.2}$$

where $\mathbf{x}_{i,t}$ is the local action of agent i at time t, while $\mathbf{x}_t^*$ is the solution of the following problem.

$$\mathbf{x}_t^* \in \operatorname*{argmin}_{\mathbf{x} \in \mathcal{X}} \left\{ f_t(\mathbf{x}) \middle| g_{i,t}(\mathbf{x}) \leq \mathbf{0}, \forall\, i \in [n] \right\}, \tag{3.3}$$

As pointed out earlier, it is impossible to satisfy the time-varying constraints instantaneously, since they are revealed post-hoc. As a surrogate, to ensure the local constraints are satisfied in the *long run*, we use the distributed extension of *fit* as the performance metric. Fit has been used in the context of both time-invariant [121], as well as time-varying constraints [30, 145], for single node problems. Our definition is motivated by the one given in [144] for continuous time problems. It measures the average accumulation of constraint violations over time.

$$\mathbf{Fit}_T^d \triangleq \frac{1}{n}\sum_{i=1}^{n}\frac{1}{n}\sum_{j=1}^{n}\left\|\left[\sum_{t=1}^{T} g_{i,t}(\mathbf{x}_{j,t})\right]_+\right\|. \tag{3.4}$$

Here, $\sum_{t=1}^{T} g_{i,t}(\mathbf{x}_{j,t})$ is the constraint violation at agent i, if it adopts the actions of agent j. Note that $\sum_{t=1}^{T} g_{i,t}(\mathbf{x}_{j,t}) \leq \mathbf{0}$ is different from requiring the constraint to be met at every time instant $g_{i,t}(\mathbf{x}_{j,t}) \leq \mathbf{0}$.

Next, we discuss the assumptions and some background required for the analysis of the proposed OCO framework. Note that the following assumptions are standard for decentralized OCO problems [156, 199].

3.3 Background and Assumptions

3.3.1 Network

We assume the n agents are connected together via an undirected graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$. $\mathcal{V} = \{1, \ldots, n\}$ denotes the set of nodes of the graph, each of which represents an agent. $\mathcal{E}$ is the set of edges between the nodes. $(i, j) \in \mathcal{E}$ implies that nodes i and j are connected in the graph. The set of edges has an associated weight matrix $\mathbf{W}$, such that

$$\mathbf{W} = \begin{cases} [\mathbf{W}]_{ij} > 0 & \text{if } (i,j) \in \mathcal{E} \\ [\mathbf{W}]_{ij} = 0 & \text{else.} \end{cases} \tag{3.5}$$

The set of neighbors of node i is, therefore, defined as $\mathcal{N}_i \triangleq \{j : [\mathbf{W}]_{ij} > 0\}$. Note that $j \in \mathcal{N}_i \Leftrightarrow i \in \mathcal{N}_j$.

Assumption 1. The network is connected. The weight matrix $\mathbf{W}$ is symmetric, doubly stochastic, such that

$$\sum_{i=1}^{n}[\mathbf{W}]_{ij} = \sum_{j=1}^{n}[\mathbf{W}]_{ij} = 1. \tag{3.6}$$

Next, we discuss the properties of the local cost functions and constraints.

3.3.2 Local Objective Functions and Constraints

Assumption 2. We assume the following conditions on the set $\mathcal{X}$, the objective and constraint functions.

1. The set $\mathcal{X} \subseteq \mathbb{R}^d$ is convex and compact. Therefore, there exists a positive constant $d(\mathcal{X})$ such that

$$\|\mathbf{x} - \mathbf{y}\| \leq d(\mathcal{X}),\ \forall\, \mathbf{x}, \mathbf{y} \in \mathcal{X}. \tag{3.7}$$

2. The local node functions $f_{i,t}(\cdot), g_{i,t}(\cdot)$ are Lipschitz continuous on $\mathcal{X}$, $\forall\, i \in [n], \forall\, t \in \mathbb{N}_+$ i.e.,

$$\begin{aligned} \|f_{i,t}(\mathbf{x}) - f_{i,t}(\mathbf{y})\| &\leq L\|\mathbf{x} - \mathbf{y}\| \\ \|g_{i,t}(\mathbf{x}) - g_{i,t}(\mathbf{y})\| &\leq L\|\mathbf{x} - \mathbf{y}\| \end{aligned} \tag{3.8}$$

for any $\mathbf{x}, \mathbf{y} \in \mathcal{X}$.

3. The functions $\{f_{i,t}\}, \{g_{i,t}\}$ are convex and uniformly bounded on the set $\mathcal{X}$, i.e., there exists a constant $F > 0$ such that

$$\|f_{i,t}(\mathbf{x})\| \leq F, \|g_{i,t}(\mathbf{x})\| \leq F, \tag{3.9}$$

$\forall\, t \in \mathbb{N}_+, \forall\, i \in [n], \forall\, \mathbf{x} \in \mathcal{X}$.

4. $\{\nabla f_{i,t}\}, \{\nabla g_{i,t}\}$ exist and are uniformly bounded on $\mathcal{X}$, i.e., there exists a constant $G > 0$ such that

$$\|\nabla f_{i,t}(\mathbf{x})\| \leq G, \|\nabla g_{i,t}(\mathbf{x})\| \leq G, \tag{3.10}$$

$\forall\, t \in \mathbb{N}_+, \forall\, i \in [n], \forall\, \mathbf{x} \in \mathcal{X}$.

Next, we briefly discuss the Bregman Divergence measure, which is crucial to the proposed mirror descent based approach.

3.3.3 Bregman Divergence

Suppose we are given a μ-strongly convex function $\mathcal{R} : \mathcal{X} \to \mathbb{R}$, i.e. $\mathcal{R}(\mathbf{x}) \geq \mathcal{R}(\mathbf{y}) + \langle \nabla \mathcal{R}(\mathbf{y}), \mathbf{x} - \mathbf{y} \rangle + \frac{\mu}{2} \|\mathbf{x} - \mathbf{y}\|^2, \forall\, \mathbf{x}, \mathbf{y} \in \mathcal{X}$. The Bregman Divergence w.r.t. $\mathcal{R}$ is defined as

$$\mathcal{D}_{\mathcal{R}}(\mathbf{x}, \mathbf{y}) \triangleq \mathcal{R}(\mathbf{x}) - \mathcal{R}(\mathbf{y}) - \langle \mathbf{x} - \mathbf{y}, \nabla \mathcal{R}(\mathbf{y}) \rangle . \tag{3.11}$$

Since $\mathcal{R}(\cdot)$ is μ-strongly convex, for any $\mathbf{x}, \mathbf{y} \in \mathcal{X}$

$$\mathcal{D}_{\mathcal{R}}(\mathbf{x}, \mathbf{y}) \geq \frac{\mu}{2} \|\mathbf{y} - \mathbf{x}\|^2. \tag{3.12}$$

We assume the following conditions on $\mathcal{D}_{\mathcal{R}}(\cdot, \cdot)$.

Assumption 3. The Bregman Divergence $\mathcal{D}_{\mathcal{R}}(\cdot, \cdot)$ satisfies

1. Separate Convexity property [16]: Given $\mathbf{x}, \{\mathbf{y}_i\}_{i=1}^m \in \mathbb{R}^d$ and scalars $\{\alpha_i\}_{i=1}^m$ on the m-dimensional probability simplex, the Bregman Divergence satisfies

$$\mathcal{D}_{\mathcal{R}} \left(\mathbf{x}, \sum_{i=1}^{m} \alpha_i \mathbf{y}_i \right) \leq \sum_{i=1}^{m} \alpha_i \mathcal{D}_{\mathcal{R}} \left(\mathbf{x}, \mathbf{y}_i \right) . \tag{3.13}$$

2. The Bregman divergence satisfies the following Lipschitz continuity condition [83]

$$\left| \mathcal{D}_{\mathcal{R}} \left(\mathbf{x}, \mathbf{y} \right) - \mathcal{D}_{\mathcal{R}} \left(\mathbf{z}, \mathbf{y} \right) \right| \leq K \left\| \mathbf{x} - \mathbf{z} \right\| \tag{3.14}$$

for any $\mathbf{x}, \mathbf{y}, \mathbf{z} \in \mathcal{X}$. This condition is satisfied if $\mathcal{R}(\cdot)$ is Lipschitz continuous on $\mathcal{X}$. Consequently,

$$\mathcal{D}_{\mathcal{R}} \left(\mathbf{x}, \mathbf{y} \right) \leq K d((X)),\ \forall\, \mathbf{x}, \mathbf{y} \in \mathcal{X}, \tag{3.15}$$

where $d((X))$ is defined in (3.7).

We next give a result on Bregman divergence from [199] which is crucial to our analysis.

Lemma 3.3.1. *Let $\mathcal{R} : \mathbb{R}^d \to \mathbb{R}$ be a μ-strongly convex function. Also, assume $\mathcal{X}$ is a closed, convex set in $\mathbb{R}^p$ and $h : \mathcal{X} \to \mathcal{X}$ is a convex function. Assume that $\nabla h(\mathbf{x})$ exists $\forall\, \mathbf{x} \in \mathcal{X}$. Then, given $\mathbf{z} \in \mathcal{X}$, the regularized Bregman projection*

$$\mathbf{y} = \operatorname*{argmin}_{\mathbf{x}\in\mathcal{X}} \left\{h(\mathbf{x}) + \mathcal{D}_{\mathcal{R}}(\mathbf{x}, \mathbf{z})\right\}, \tag{3.16}$$

satisfies the following inequality

$$\langle \mathbf{y} - \mathbf{x}, \nabla h(\mathbf{y})\rangle \leq \mathcal{D}_{\mathcal{R}}(\mathbf{x}, \mathbf{z}) - \mathcal{D}_{\mathcal{R}}(\mathbf{x}, \mathbf{y}) - \mathcal{D}_{\mathcal{R}}(\mathbf{y}, \mathbf{z}) \tag{3.17}$$

$\forall\, \mathbf{x} \in \mathcal{X}$.

3.3.4 Projection

For a set $\mathcal{A} \subseteq \mathbb{R}^d$, the projection operator is defined as

$$\mathcal{P}_{\mathcal{A}}(\mathbf{y}) \triangleq \operatorname*{argmin}_{\mathbf{x}\in\mathcal{A}} \|\mathbf{x} - \mathbf{y}\|^2, \tag{3.18}$$

$\forall\, \mathbf{y} \in \mathbb{R}^d$. For closed and convex $\mathcal{A}$, projection always exists and is unique. If $\mathcal{A} = \mathbb{R}^d_+$, projection is denoted by $[\cdot]_+$ and it satisfies

$$\left\|[\mathbf{x}]_+ - [\mathbf{y}]_+\right\| \leq \|\mathbf{x} - \mathbf{y}\|, \forall\, \mathbf{x}, \mathbf{y} \in \mathbb{R}^d. \tag{3.19}$$

Algorithm 4 Distributed Primal-Dual Mirror Descent

```
1:  Input: Non-increasing sequences $\{\alpha_t > 0\}, \{\beta_t > 0\}, \{\gamma_t > 0\}$; Differentiable and strongly-convex $\mathcal{R}$
2:  Initialize: $\mathbf{x}_{i,0} = \mathbf{0}_d \in \mathcal{X}$, $f_{i,0}(\cdot) \equiv 0, g_{i,0}(\cdot) \equiv \mathbf{0}_m, \mathbf{q}_{i,0} = \mathbf{0}_m, \forall\, i \in [n]$.
3:  for $t = 1$ to $T$ do
4:      for $i = 1$ to $n$ do
5:          Observe $\nabla f_{i,t-1}(\mathbf{x}_{i,t-1}), \nabla g_{i,t-1}(\mathbf{x}_{i,t-1}), g_{i,t-1}(\mathbf{x}_{i,t-1})$
6:          $\mathbf{a}_{i,t} = \nabla f_{i,t-1}(\mathbf{x}_{i,t-1}) + [\nabla g_{i,t-1}(\mathbf{x}_{i,t-1})]^T \mathbf{q}_{i,t-1}$
7:          $\mathbf{y}_{i,t} = \text{argmin}_{\mathbf{x}\in\mathcal{X}} \{\alpha_t \langle \mathbf{x}, \mathbf{a}_{i,t}\rangle + \mathcal{D}_{\mathcal{R}}(\mathbf{x}, \mathbf{x}_{i,t-1})\}$
8:          $\mathbf{b}_{i,t} = [\nabla g_{i,t-1}(\mathbf{x}_{i,t-1})]\,(\mathbf{y}_{i,t} - \mathbf{x}_{i,t-1}) + g_{i,t-1}(\mathbf{x}_{i,t-1})$
9:          $\mathbf{q}_{i,t} = [\mathbf{q}_{i,t-1} + \gamma_t(b_{i,t} - \beta_t \mathbf{q}_{i,t-1})]_+$
10:         Broadcast $\mathbf{y}_{i,t}$ to out-neighbors $j \in \mathcal{N}_i$
11:         Obtain $\mathbf{y}_{j,t}$ from in-neighbors $j \in \mathcal{N}_i$
12:         $\mathbf{x}_{i,t} = \sum_{j=1}^{n} [\mathbf{W}]_{ij} \mathbf{y}_{j,t}$
13:     end for
14: end for
```

3.4 Distributed Primal-Dual Mirror Descent based Algorithm

We next discuss the proposed distributed primal-dual mirror descent based algorithm for online convex optimization with time-varying constraints. The pseudo-code is outlined in Algorithm 4. The algorithm runs in parallel at all the nodes. At the end of time $t-1$, $\mathbf{x}_{i,t-1}$ is the action (primal variable) at node i. Following this, the local functions $f_{i,t-1}, g_{i,t-1}$ are revealed to the agent. The corresponding function values and gradients are utilized to carry-out the updates in the next time step t. First, each agent performs the primal update locally (Step 7). This is followed by the dual update (Step 9). Note that the projection $[\cdot]_+$ ensures that the dual variable lies in the non-negative orthant $\mathbb{R}_+^m$. At the end of each time step, an average consensus step is taken across the nodes, where the local updated primal variables $\mathbf{y}_{i,t-1}$ are received from the neighbors, to compute the action $\mathbf{x}_{i,t}$.

Remark. Note that the primal and dual update steps employ different step-sizes, α_t and γ_t, respectively. This idea originated in [85] and leads to flexibility in terms of the trade-off between the bounds on dynamic regret and fit.

In the next section, we bound the dynamic regret and fit which result from Algorithm 4, and show them to be sublinear in the time-horizon T.

3.5 Dynamic Regret and Fit Bounds

First, we discuss some intermediate results required to show the sublinearity of dynamic regret and fit. We have relegated the proofs to the Appendix. Our analysis follows closely the work in [156] and [199].

3.5.1 Some Intermediate Results

Lemma 3.5.1. *Suppose Assumption 2 holds.* $\forall\, i \in [n]$, $\forall\, t \in \mathbb{N}_+$, $\mathbf{q}_{i,t}$ *generated by Algorithm 4 satisfy*

$$\|\mathbf{q}_{i,t}\| \leq \frac{F}{\beta_t}, \tag{3.20}$$

$$\begin{aligned}\frac{\Delta_{t+1}}{2\gamma_{t+1}} \leq{}& \frac{nB_1^2}{2}\gamma_{t+1} + \sum_{i=1}^{n} \mathbf{q}_{i,t}^T[\nabla g_{i,t}(\mathbf{x}_{i,t})](\mathbf{y}_{i,t+1} - \mathbf{x}_{i,t}) + \frac{\mu}{4\alpha_{t+1}} \sum_{i=1}^{n} \|\mathbf{y}_{i,t+1} - \mathbf{x}_{i,t}\|^2 \\ &+ \left(\frac{G^2\alpha_{t+1}}{\mu} + \frac{\beta_{t+1}}{2}\right) \sum_{i=1}^{n} \|\mathbf{q}_i\|^2 + \sum_{i=1}^{n} (\mathbf{q}_{i,t} - \mathbf{q}_i)^T g_{i,t}(\mathbf{x}_{i,t})\end{aligned} \tag{3.21}$$

where $B_1 = 2F + Gd(\mathcal{X})$*,*

$$\Delta_{t+1} \triangleq \sum_{i=1}^{n} \left[\|\mathbf{q}_{i,t+1} - \mathbf{q}_i\|^2 - (1 - \gamma_{t+1}\beta_{t+1})\|\mathbf{q}_{i,t} - \mathbf{q}_i\|^2\right],$$

and $\{\mathbf{q}_i\}_i$ *are arbitrary vectors in* $\mathbb{R}_+^m$*.*

PROOF: See Appendix A.2.1. ■

Remark. The penalty term $-\beta_t \mathbf{q}_{i,t-1}$ in the dual update (step 9, Algorithm 4) helps in upper bounding the local dual variables. This idea was initially used in [121] and helps get rid of the requirement of Slater's condition. Δ_{t+1} measures the regularized drift of the local dual variables. See [73] and [199] for similar results, respectively in centralized and distributed contexts.

Next, we sum the left hand side of (3.21) over t to get

$$\sum_{t=1}^{T} \frac{\Delta_{t+1}}{2\gamma_{t+1}} = \frac{1}{2} \sum_{t=1}^{T} \left(\frac{1}{\gamma_t} - \frac{1}{\gamma_{t+1}} + \beta_{t+1} \right) \sum_{i=1}^{n} \|\mathbf{q}_{i,t} - \mathbf{q}_i\|^2$$
$$- \frac{1}{2} \sum_{i=1}^{n} \left[\frac{1}{\gamma_1} \|\mathbf{q}_{i,1} - \mathbf{q}_i\|^2 - \frac{1}{\gamma_{T+1}} \|\mathbf{q}_{i,T+1} - \mathbf{q}_i\|^2 \right]. \tag{3.22}$$

Recall that $\mathbf{q}_{i,1} = \mathbf{0}$, $\forall\, i \in [n]$. We combine (3.21) and (3.22), and define $g_c(\cdot)$ such that

$$g_c(\mathbf{q}_1, \dots, \mathbf{q}_n) \triangleq \sum_{i=1}^{n} \mathbf{q}_i^T \Big(\sum_{t=1}^{T} g_{i,t}(\mathbf{x}_{i,t}) \Big) - \left[\frac{1}{2\gamma_1} + \sum_{t=1}^{T} \left(\frac{G^2 \alpha_{t+1}}{\mu} + \frac{\beta_{t+1}}{2} \right) \right] \sum_{i=1}^{n} \|\mathbf{q}_i\|^2$$
$$\leq \frac{nB_1^2}{2} \sum_{t=1}^{T} \gamma_{t+1} + \sum_{t=1}^{T} \sum_{i=1}^{n} \mathbf{q}_{i,t}^T [\nabla g_{i,t}(\mathbf{x}_{i,t})](\mathbf{y}_{i,t+1} - \mathbf{x}_{i,t}) + \sum_{t=1}^{T} \sum_{i=1}^{n} \mathbf{q}_{i,t}^T g_{i,t}(\mathbf{x}_{i,t})$$
$$+ \sum_{t=1}^{T} \frac{\mu}{4\alpha_{t+1}} \sum_{i=1}^{n} \|\mathbf{y}_{i,t+1} - \mathbf{x}_{i,t}\|^2 - \frac{1}{2} \sum_{t=1}^{T} \left(\frac{1}{\gamma_t} - \frac{1}{\gamma_{t+1}} + \beta_{t+1} \right) \sum_{i=1}^{n} \|\mathbf{q}_{i,t} - \mathbf{q}_i\|^2. \tag{3.23}$$

The function $g_c(\mathbf{q}_1, \dots, \mathbf{q}_n)$ will be used later in Lemma 3.5.4 to upper bound both the dynamic regret and fit, by appropriately choosing $\mathbf{q}_i$, $\forall\, i \in [n]$.

Before looking at the primal updates, we first consider one of the constituent terms in (3.2).

$$f_t(\mathbf{x}_{i,t}) - f_t(\mathbf{x}_t^*) = f_t(\mathbf{x}_{i,t}) - f_t(\bar{\mathbf{x}}_t) + f_t(\bar{\mathbf{x}}_t) - f_t(\mathbf{x}_t^*)$$
$$\leq L \|\mathbf{x}_{i,t} - \bar{\mathbf{x}}_t\| + f_t(\bar{\mathbf{x}}_t) - f_t(\mathbf{x}_t^*) \tag{3.24}$$
$$= \frac{1}{n} \sum_{j=1}^{n} \{ f_{j,t}(\bar{\mathbf{x}}_t) - f_{j,t}(\mathbf{x}_t^*) + f_{j,t}(\mathbf{x}_{j,t}) - f_{j,t}(\mathbf{x}_{j,t}) \} + L \|\mathbf{x}_{i,t} - \bar{\mathbf{x}}_t\|$$
$$\leq \frac{1}{n} \sum_{j=1}^{n} \{ f_{j,t}(\mathbf{x}_{j,t}) - f_{j,t}(\mathbf{x}_t^*) \} + L \|\mathbf{x}_{i,t} - \bar{\mathbf{x}}_t\| + \frac{L}{n} \sum_{j=1}^{n} \|\mathbf{x}_{j,t} - \bar{\mathbf{x}}_t\|. \tag{3.25}$$

We use Assumption 2 to obtain both (3.24), (3.25). Now, from the definition of dynamic regret

(3.2), we get

$$\mathbf{Reg}_T^d \leq \frac{1}{n}\sum_{i=1}^{n}\sum_{t=1}^{T}\frac{1}{n}\sum_{j=1}^{n}\{f_{j,t}(\mathbf{x}_{j,t}) - f_{j,t}(\mathbf{x}_t^*)\} + \frac{2L}{n}\sum_{i=1}^{n}\sum_{t=1}^{T}\|\mathbf{x}_{i,t} - \bar{\mathbf{x}}_t\|. \tag{3.26}$$

Next, we upper bound both the terms in (3.26). First, we upper bound the first term in the following lemma.

Lemma 3.5.2. *Suppose Assumptions 1-3 hold.* $\forall\ i \in [n]$*,* $\forall\ t \in \mathbb{N}_+$*, if* $\{\mathbf{x}_{i,t}\}$ *is the sequence generated by Algorithm 4. Then,*

$$\begin{aligned}
&\sum_{t=1}^{T}\sum_{i=1}^{n}[f_{i,t}(\mathbf{x}_{i,t}) - f_{i,t}(\mathbf{x}_t^*)] \\
&\leq \frac{nG^2}{\mu}\sum_{t=1}^{T}\alpha_{t+1} - \sum_{t=1}^{T}\sum_{i=1}^{n}\frac{\mu}{4\alpha_{t+1}}\|\mathbf{y}_{i,t+1} - \mathbf{x}_{i,t}\|^2 - \sum_{t=1}^{T}\sum_{i=1}^{n}\mathbf{q}_{i,t}^T\Big[g_{i,t}(\mathbf{x}_{i,t}) + \nabla g_{i,t}(\mathbf{x}_{i,t})\left(\mathbf{y}_{i,t+1} - \mathbf{x}_{i,t}\right)\Big] \\
&+ \sum_{i=1}^{n}\left[\frac{1}{\alpha_2}\mathcal{D}_{\mathcal{R}}(\mathbf{x}_1^*, \mathbf{x}_{i,1}) - \frac{1}{\alpha_{T+2}}\mathcal{D}_{\mathcal{R}}(\mathbf{x}_{T+1}^*, \mathbf{x}_{i,T+1})\right] + \frac{nK}{\alpha_{T+2}}\sum_{t=1}^{T}\|\mathbf{x}_{t+1}^* - \mathbf{x}_t^*\| + \frac{nKd((X))}{\alpha_{T+2}}.
\end{aligned} \tag{3.27}$$

PROOF: See Appendix A.2.2. ■

Next, we upper bound the second term in (3.26). This is the consensus error of the primal variables.

Lemma 3.5.3. *(Network Error): Suppose Assumptions 1-3 hold. Then, the local estimates* $\{\mathbf{x}_{i,t}\}$ *generated by Algorithm 4 satisfy*

$$\|\mathbf{x}_{i,t} - \bar{\mathbf{x}}_t\| \leq \sum_{\tau=0}^{t-1}\sqrt{n}\sigma_2^{t-\tau}(\mathbf{W})\frac{G\alpha_{\tau+1}}{\mu}\left(1 + \frac{F}{\beta_{\tau+1}}\right) \tag{3.28}$$

$\forall\ i \in [n]$*, where* $\bar{\mathbf{x}}_t = \frac{1}{n}\sum_{i=1}^{n}\mathbf{x}_{i,t}$*.* $\sigma_2(\mathbf{W})$ *is the second largest eigenvalue of* $\mathbf{W}$ *in magnitude.*

PROOF: See Appendix A.2.3. ■

Remark. The network error bound is (3.28) is independent of the node index i. The dependence on $\sigma_2(\mathbf{W})$ captures the speed with which mixing of iterates happens. The smaller the value of

$\sigma_2(\mathbf{W})$, the faster the network error diminishes. Moreover, the choice of the primal update step sizes $\{\alpha_t\}$ and the dual update regularization parameters $\{\beta_t\}$ has a crucial role to play in bounding the network error. As we shall see in Theorem 3.5.5, carefully choosing these leads to sublinear regret and fit.

Next, we combine (3.23) and Lemma 3.5.2 resulting in two intermediate bounds, which shall be needed to subsequently bound the dynamic regret and fit respectively.

Lemma 3.5.4. *Suppose Assumptions 1-3 hold. Then, the sequences* $\{\mathbf{x}_{i,t}, \mathbf{q}_{i,t}\}$ *generated by Algorithm 4 satisfy*

$$\begin{aligned}\sum_{t=1}^{T}\sum_{i=1}^{n}(f_{i,t}(\mathbf{x}_{i,t}) - f_{i,t}(\mathbf{x}_t^*)) \leq & \frac{nB_1^2}{2}\sum_{t=1}^{T}\gamma_{t+1} + \frac{nG^2}{\mu}\sum_{t=1}^{T}\alpha_{t+1} \\ & + \sum_{i=1}^{n}\left[\frac{1}{\alpha_2}\mathcal{D}_{\mathcal{R}}(\mathbf{x}_1^*, \mathbf{x}_{i,1}) - \frac{1}{\alpha_{T+2}}\mathcal{D}_{\mathcal{R}}(\mathbf{x}_{T+1}^*, \mathbf{x}_{i,T+1})\right] + \frac{nK}{\alpha_{T+2}}\sum_{t=1}^{T}\|\mathbf{x}_{t+1}^* - \mathbf{x}_t^*\| \\ & + \frac{nKd((X))}{\alpha_{T+2}} - \frac{1}{2}\sum_{t=1}^{T}\left(\frac{1}{\gamma_t} - \frac{1}{\gamma_{t+1}} + \beta_{t+1}\right)\sum_{i=1}^{n}\|\mathbf{q}_{i,t}\|^2, \end{aligned} \tag{3.29}$$

and

$$\begin{aligned}\sum_{i=1}^{n}\left\|\left[\sum_{t=1}^{T}g_{i,t}(\mathbf{x}_{i,t})\right]_+\right\|^2 \leq & 4\left[\frac{1}{2\gamma_1} + \sum_{t=1}^{T}\left(\frac{G^2\alpha_{t+1}}{\mu} + \frac{\beta_{t+1}}{2}\right)\right]\Bigg\{2nFT + \frac{nB_1^2}{2}\sum_{t=1}^{T}\gamma_{t+1} \\ & + \frac{nG^2}{\mu}\sum_{t=1}^{T}\alpha_{t+1} + \sum_{i=1}^{n}\left[\frac{1}{\alpha_2}\mathcal{D}_{\mathcal{R}}(\mathbf{x}_1^*, \mathbf{x}_{i,1}) - \frac{1}{\alpha_{T+2}}\mathcal{D}_{\mathcal{R}}(\mathbf{x}_{T+1}^*, \mathbf{x}_{i,T+1})\right] + \frac{nKd((X))}{\alpha_{T+2}} \\ & + \frac{nK}{\alpha_{T+2}}\sum_{t=1}^{T}\|\mathbf{x}_{t+1}^* - \mathbf{x}_t^*\| - \frac{1}{2}\sum_{t=1}^{T}\left(\frac{1}{\gamma_t} - \frac{1}{\gamma_{t+1}} + \beta_{t+1}\right)\sum_{i=1}^{n}\|\mathbf{q}_{i,t} - \bar{\mathbf{q}}_i\|^2\Bigg\}. \end{aligned} \tag{3.30}$$

Remark. (3.29) follows by adding (3.23) and (3.27), and substituting $\mathbf{q}_i = \mathbf{0}_m, \ \forall\, i \in [n]$. Similarly, (3.30) is obtained by adding (3.23) and (3.27), and substituting

$$\bar{\mathbf{q}}_i = \frac{\left[\sum_{t=1}^{T} g_{i,t}(\mathbf{x}_{i,t})\right]_+}{2\left[\frac{1}{2\gamma_1} + \sum_{t=1}^{T}\left(\frac{G^2\alpha_{t+1}}{\mu} + \frac{\beta_{t+1}}{2}\right)\right]}, \ \forall\, i \in [n]. \tag{3.31}$$

Before presenting out final result, we need to use the following upper bound to bound the fit.

$$\frac{1}{n}\sum_{i=1}^{n}\frac{1}{n}\sum_{j=1}^{n}\left\|\left[\sum_{t=1}^{T} g_{i,t}(\mathbf{x}_{j,t})\right]_+\right\|^2 \leq 2\left[2L\sum_{t=1}^{T}\|\mathbf{x}_{i,t}-\bar{\mathbf{x}}_t\|\right]^2 + \frac{2}{n}\sum_{i=1}^{n}\left\|\left[\sum_{t=1}^{T} g_{i,t}(\mathbf{x}_{i,t})\right]_+\right\|^2. \tag{3.32}$$

This follows from Lipschitz continuity of the constraint functions (Assumption 2). Since, we have bounded both the terms in (3.32) (the first term in Lemma 3.5.3, and the second term in Lemma 3.5.4), we are now ready to present our final result on the sublinearity of both dynamic regret and fit.

3.5.2 Dynamic Regret and Fit Bounds

Theorem 3.5.5. *Suppose Assumptions 1-3 hold, and $\{\mathbf{x}_{i,t}\}$ be the sequence of local estimates generated by Algorithm 4. We choose the step sizes*

$$\alpha_t = \frac{1}{t^a},\ \beta_t = \frac{1}{t^b},\ \gamma_t = \frac{1}{t^{1-b}},\quad \forall\, t \in \mathbb{N}_+ \tag{3.33}$$

where, $a, b \in (0,1)$ and $a > b$. Then for any $T \in \mathbb{N}_+$.

$$\mathbf{Reg}_T^d \leq R_1 T^{\max\{a,1-a+b\}} + 2KT^a C_T^*, \tag{3.34}$$

$$\frac{1}{n}\sum_{i=1}^{n}\frac{1}{n}\sum_{j=1}^{n}\left\|\left[\sum_{t=1}^{T} g_{i,t}(\mathbf{x}_{j,t})\right]_+\right\|^2 \leq D_1 T^{2-b} + D_2 T^{1+a-b} C_T^* + D_3 T^{2+2b-2a}. \tag{3.35}$$

Here, $R = \frac{4FLG\sqrt{n}\sigma_2(\mathbf{W})}{\mu(1-a)(1-\sigma_2(\mathbf{W}))}, R_1 = R + \frac{B_1^2}{2b} + \frac{G^2}{\mu(1-a)} + 2Kd((X)), D = 2 + \frac{4G^2}{\mu(1-a)} + \frac{2}{1-b}, D_1 = 2D(2F + 2Kd((X)) + \frac{B_1^2}{2b} + \frac{G^2}{\mu(1-a)} + 2Kd((X)))$, $D_2 = 4KD$ and $D_3 = 16L^2R^2$ are constants independent of T, and

$$C_T^* \triangleq \sum_{t=1}^{T}\|\mathbf{x}_{t+1}^* - \mathbf{x}_t^*\| \tag{3.36}$$

is the accumulated dynamic variation of the comparator sequence $\{\mathbf{x}_t^*\}$.

PROOF: We begin with one of the constituent terms in (3.2).

$$\begin{aligned} f_t(\mathbf{x}_{i,t}) - f_t(\mathbf{x}_t^*) &= f_t(\mathbf{x}_{i,t}) - f_t(\bar{\mathbf{x}}_t) + f_t(\bar{\mathbf{x}}_t) - f_t(\mathbf{x}_t^*) \\ &\leq L \left\| \mathbf{x}_{i,t} - \bar{\mathbf{x}}_t \right\| + f_t(\bar{\mathbf{x}}_t) - f_t(\mathbf{x}_t^*) && (3.37) \\ &= \frac{1}{n} \sum_{j=1}^{n} \left\{ f_{j,t}(\bar{\mathbf{x}}_t) - f_{j,t}(\mathbf{x}_t^*) + f_{j,t}(\mathbf{x}_{j,t}) - f_{j,t}(\mathbf{x}_{j,t}) \right\} + L \left\| \mathbf{x}_{i,t} - \bar{\mathbf{x}}_t \right\| \\ &\leq \frac{1}{n} \sum_{j=1}^{n} \left\{ f_{j,t}(\mathbf{x}_{j,t}) - f_{j,t}(\mathbf{x}_t^*) \right\} + L \left\| \mathbf{x}_{i,t} - \bar{\mathbf{x}}_t \right\| + \frac{L}{n} \sum_{j=1}^{n} \left\| \mathbf{x}_{j,t} - \bar{\mathbf{x}}_t \right\|. && (3.38) \end{aligned}$$

We use Assumption 2 to get both (3.24), (3.25). Now, from the definition of dynamic regret (3.2), we get

$$\mathbf{Reg}_T^d \leq \frac{1}{n} \sum_{i=1}^{n} \sum_{t=1}^{T} \frac{1}{n} \sum_{j=1}^{n} \left\{ f_{j,t}(\mathbf{x}_{j,t}) - f_{j,t}(\mathbf{x}_t^*) \right\} + \frac{2L}{n} \sum_{i=1}^{n} \sum_{t=1}^{T} \left\| \mathbf{x}_{i,t} - \bar{\mathbf{x}}_t \right\|. \tag{3.39}$$

We use Lemma 3.5.4 to bound the first term, and Lemma 3.5.3 to bound the second term in (3.39). But first, we bound all the terms in the upper bounds in Lemma 3.5.3 and 3.5.4, using the step-sizes in (3.33).

For a constant $p < 1$ and $T \in \mathbb{N}_+$

$$\sum_{t=1}^{T} \frac{1}{t^p} \leq 1 + \int_1^T \frac{1}{t^p} dt = \frac{T^{1-p} - 1}{1-p} + 1 \leq \frac{T^{1-p}}{1-p}. \tag{3.40}$$

Hence,

$$\begin{aligned} &\sum_{t=1}^{T} \gamma_{t+1} \leq \frac{T^b}{b}, \sum_{t=1}^{T} \alpha_{t+1} \leq \frac{T^{1-a}}{1-a}, \sum_{t+1}^{T} \beta_{t+1} \leq \frac{T^{1-b}}{1-b}, && (3.41) \\ &\frac{1}{\alpha_2} \mathcal{D}_{\mathcal{R}}(\mathbf{x}_1^*, \mathbf{x}_{i,1}) - \frac{1}{\alpha_{T+2}} \mathcal{D}_{\mathcal{R}}(\mathbf{x}_{T+1}^*, \mathbf{x}_{i,T+1}) \leq 2Kd((X)), && (3.42) \\ &\frac{1}{\alpha_{T+2}} = (T+2)^a \leq 2T^a, \qquad \forall\, T \geq 2. && (3.43) \end{aligned}$$

(3.42) follows from $\mathcal{D}_{\mathcal{R}}(\cdot, \cdot) \geq 0$, (3.15) and $\frac{1}{\alpha_2} = 2^a < 2$. Also, $\left(\frac{1}{\gamma_t} - \frac{1}{\gamma_{t+1}} + \beta_{t+1} \right) > 0$ follows from (3.33). Therefore, we ignore the last term in both (3.29) and (3.30). Also, substituting (3.28) in the second term in (3.39), we get

$$\begin{aligned}
&2L\sum_{t=1}^{T}\sum_{\tau=0}^{t-1}\sqrt{n}\sigma_2^{t-\tau}(\mathbf{W})\frac{G\alpha_{\tau+1}}{\mu}\left(1+\frac{F}{\beta_{\tau+1}}\right)\\
&=\frac{2LG\sqrt{n}}{\mu}\sum_{t=1}^{T}\sum_{\tau=0}^{t-1}\left(\frac{\sigma_2^{t-\tau}}{(\tau+1)^a}+F\frac{\sigma_2^{t-\tau}}{(\tau+1)^{a-b}}\right)\\
&=\frac{2LG\sqrt{n}}{\mu}\sum_{t=1}^{T}\sum_{\tau=1}^{t}\left(\frac{\sigma_2^{t+1-\tau}}{\tau^a}+F\frac{\sigma_2^{t+1-\tau}}{\tau^{a-b}}\right)\\
&\leq\frac{2LG\sqrt{n}\sigma_2(\mathbf{W})}{\mu(1-\sigma_2(\mathbf{W}))}\sum_{t=1}^{T}\left(\frac{1}{\tau^a}+F\frac{1}{\tau^{a-b}}\right) &\text{(3.44)}\\
&=\frac{2LG\sqrt{n}\sigma_2(\mathbf{W})}{\mu(1-\sigma_2(\mathbf{W}))}\left(\frac{T^{1-a}}{1-a}+\frac{FT^{1+b-a}}{1-a+b}\right). &\text{(3.45)}
\end{aligned}$$

where in (3.44), we use the following inequality. Given a non-negative sequence $\{\delta_t\}$, and $0 < \lambda < 1$, it holds

$$\sum_{t=1}^{T}\sum_{\tau=1}^{t}\delta_{\tau+1}\lambda^{t-\tau}=\sum_{t=1}^{T}\delta_{t+1}\sum_{\tau=0}^{T-\tau}\lambda^{\tau}\leq\frac{1}{1-\lambda}\sum_{t=1}^{T}\delta_{t+1}. \tag{3.46}$$

Using (3.41)-(3.43), (3.45), and Lemma 3.5.3, 3.5.4 we upper bound the dynamic regret (3.39) as

$$\begin{aligned}
\mathbf{Reg}_T^d \leq{}& \frac{B_1^2}{2}\frac{T^b}{b}+\frac{G^2}{\mu}\frac{T^{1-a}}{1-a}+2T^aKd((X))+2KT^aC_T^*\\
&+2Kd((X))+\frac{2LG\sqrt{n}\sigma_2(\mathbf{W})}{\mu(1-\sigma_2(\mathbf{W}))}\left(\frac{T^{1-a}}{1-a}+\frac{FT^{1+b-a}}{1-a+b}\right) &\text{(3.47)}
\end{aligned}$$

Since $a > b$, for large enough T, T^a dominates T^b, while, T^{1-a+b} will dominate T^{1-a}.

To upper bound the fit, we use (3.41)-(3.43), and Lemma 3.5.4.

$$\begin{aligned}
&\frac{1}{n}\sum_{i=1}^{n}\left\|\left[\sum_{t=1}^{T}g_{i,t}(\mathbf{x}_{i,t})\right]_+\right\|^2\\
&\leq 4\left[\frac{1}{2\gamma_1}+\frac{G^2}{\mu}\frac{T^{1-a}}{1-a}+\frac{T^{1-b}}{2(1-b)}\right]\big\{2FT+2Kd((X))\\
&+\frac{B_1^2}{2}\frac{T^b}{b}+\frac{G^2}{\mu}\frac{T^{1-a}}{1-a}+2T^aKd((X))+2KT^aC_T^*\big\}. &\text{(3.48)}
\end{aligned}$$

For large enough T, T^{2-b} dominates T, T^{2-a}, T^{2-a-b} and T^{1+a-b}. Also, T^{1+a-b} dominates T. ■

Remark. The dynamic regret $\mathbf{Reg}_T^d$ is sublinear as long as the cumulative consecutive variations of the dynamic comparators C_T^* is sublinear. This is the standard requirement for sublinearity of dynamic regret [73, 156, 199].

Remark. A similar argument as above holds for (3.35). As long as C_T^* is sublinear, we have

$$\frac{1}{n}\sum_{i=1}^{n}\frac{1}{n}\sum_{j=1}^{n}\left\|\left[\sum_{t=1}^{T} g_{i,t}(\mathbf{x}_{j,t})\right]_+\right\|^2 = o(T^2). \tag{3.49}$$

Note that (3.35) has $\|[\sum_{t=1}^{T} g_{i,t}(\mathbf{x}_{j,t})]_+\|^2$, while fit (3.4) is defined with $\|[\sum_{t=1}^{T} g_{i,t}(\mathbf{x}_{j,t})]_+\|$. However, for large enough T, each of the constituent terms in (3.49) are $o(T^2)$. Consequently, $\|[\sum_{t=1}^{T} g_{i,t}(\mathbf{x}_{j,t})]_+\|^2 = o(T), \forall\, i, j \in [n]$. Therefore, we get a sublinear fit

$$\frac{1}{n}\sum_{i=1}^{n}\frac{1}{n}\sum_{j=1}^{n}\left\|\left[\sum_{t=1}^{T} g_{i,t}(\mathbf{x}_{j,t})\right]_+\right\| = o(T). \tag{3.50}$$

3.6 Conclusion

In this work, we considered a distributed OCO problem, with time-varying (potentially adversarial) constraints. We proposed a distributed primal-dual mirror descent based approach, in which the primal and dual updates are carried out locally at all the nodes. We utilized the challenging, but more realistic metric of dynamic regret and fit. Without assuming the more restrictive Slater's conditions, we achieved sublinear regret and fit under mild, commonly used assumptions. To the best of our knowledge, this is the first work to consider distributed OCO problem with non-coupled local time-varying constraints, and achieve sublinear dynamic regret and fit.

CHAPTER 4

DISTRIBUTED STOCHASTIC FIRST-ORDER OPTIMIZATION: PR-SPIDER

4.1 Introduction

In this chapter, we study a distributed optimization problem in the worker-server architecture. The *worker* nodes or machines are assumed have significant storage and computational resources. Therefore, the server offloads some of its conventional tasks to these workers [194].

4.1.1 Problem

The optimization problem we solve is as follows:

$$\min_{x \in \mathbb{R}^d} f(\mathbf{x}) \triangleq \frac{1}{N} \sum\nolimits_{i=1}^{N} f_i(\mathbf{x}), \tag{4.1}$$

where N is the number of worker nodes. The local function corresponding to node i, $f_i(\mathbf{x})$ is a smooth, potentially non-convex function. We consider two variants of this problem.

- **Finite-Sum Case:** Each individual node function f_i in (4.1) is an empirical mean of function values corresponding to n samples. Therefore, the problem is of the form

$$\min_{x \in \mathbb{R}^d} f(\mathbf{x}) \triangleq \frac{1}{N} \sum_{i=1}^{N} \frac{1}{n} \sum_{j=1}^{n} f_i(\mathbf{x}; \xi_j), \tag{4.2}$$

where ξ_j denotes the j-th sample and $f_i(\mathbf{x}, \xi_j)$ is the cost corresponding to the j-th sample.

- **Online Case:** Each individual node function f_i in (4.1) is an expected value. Hence, the problem has the form

  $$\min_{x \in \mathbb{R}^d} f(\mathbf{x}) \triangleq \frac{1}{N} \sum_{i=1}^{N} \mathbb{E}_{\xi \in \mathcal{D}_i} f_i(\mathbf{x}; \xi), \tag{4.3}$$

  where $\mathcal{D}_i$ denotes the distribution of samples at the i-th node. Note that $\mathcal{D}_i$ can be different across different workers. This scenario is the popular **federated learning** [98] model.

Throughout the chapter, we assume that given a point $\mathbf{y}$, each node can choose samples ξ independently, and the stochastic gradients of these sample functions are unbiased estimators of the actual gradient. Finding the global minimizer of a nonconvex function is NP-hard in general [133]. Therefore, surrogate objectives are used in the literature to approximate the solution of the original problem. One such objective is finding the stationary point (x such that $\nabla f(\mathbf{x}) = 0$) of the objective function. An approximate stationary point or ϵ-stationary point ($\epsilon > 0$) is defined as a point $\mathbf{x}$ such that $\|\nabla f(\mathbf{x})\|^2 \leq \epsilon$. For stochastic algorithms, this definition is slightly modified to $\mathbb{E} \|\nabla f(\mathbf{x})\|^2 \leq \epsilon$. The expectation accounts for the randomness introduced by the stochastic nature of the algorithms. The point $\mathbf{x}$ is referred to as a first-order stationary (**FoS**) point of (4.1).

While solving a distributed optimization problem, another objective, in addition to minimizing f, is to ensure that the individual local iterates at the nodes do not drift too far. Therefore, it is appropriate to include a consensus error term in the global objective [170]. For this purpose, we modify the definition of the **FoS** point and include a consensus error term in the definition as follows.

Definition 4.1.1. **ϵ-First-order stationary (ϵ-FoS) point** [170] Let $\{\mathbf{x}_i\}_{i=1}^{N}$ with $\mathbf{x}_i \in \mathbb{R}^d$ be the local iterates at N nodes and $\bar{\mathbf{x}} = \frac{1}{N}\sum_{i=1}^{N} \mathbf{x}_i$, then we define the ϵ-**FoS** point $\bar{\mathbf{x}}$ as

$$\mathbb{E}\left\|\nabla f\left(\bar{\mathbf{x}}\right)\right\|^2 + \frac{L^2}{N}\sum\nolimits_{i=1}^{N} \mathbb{E}\left\|\mathbf{x}_i - \bar{\mathbf{x}}\right\|^2 \leq \epsilon.$$

Note that the expectation is over the stochasticity of the algorithm. All the results in this chapter are in terms of convergence to an ϵ-**FoS** point.

Note that Definition 4.1.1 is a sufficient condition to ensure that $\mathbb{E}\left\|\nabla f\left(\bar{\mathbf{x}}\right)\right\|^2 \leq \epsilon$. The complexity of the algorithm is measured in terms of the amount of computations done at the worker nodes, and the communication between the worker and the server nodes. These notions are defined more precisely next.

Definition 4.1.2. Computation Complexity: In the Incremental First-order Oracle (IFO) framework [2], given a sample ξ at node i and a point $\mathbf{x}$, the oracle returns $(f_i(\mathbf{x};\xi), \nabla f_i(\mathbf{x};\xi))$, i.e., the function value and the gradient of the local stochastic function $f_i(\cdot;\xi)$ at $\mathbf{x}$. Each access to the oracle is counted as a single IFO operation. The sample or computation complexity of the algorithm is hence, the total (aggregated across all the workers) number of IFO operations to reach an ϵ-FoS solution.

Definition 4.1.3. Communication Complexity: In one round of communication between the workers and the server, each worker sends its local vector $\in \mathbb{R}^d$ to the server, and receives an aggregated vector of the same dimension in return. The communication complexity of the algorithm is the number of such communication rounds required to reach an ϵ-FoS solution.

4.1.2 Related Work

Past few years have seen a meteoric rise in the amount of research in distributed optimization methods. Next, we review a sliver of the vast literature on this topic that is most relevant to the work presented in this chapter. Since stochastic gradient descent (SGD) is the workhorse of the modern Big-Data machinery, we begin with a quick review of the basic results using SGD.

Stochastic Gradient Descent (SGD)

For nonconvex problems, to achieve an ϵ-stationary solution, $O(1/\epsilon^2)$ IFO calls are required [62]. In the absence of additional assumptions, this bound cannot be improved [9]. However, with additional assumptions[1], this bound can be improved using variance reduction methods.

Variance Reduction

For the sake of simplicity of the discussion in this section, we assume that all the samples in the finite-sum problem (4.2) or the online problem (4.3) are available at a single node. To solve this problem, gradient descent (GD) and SGD are the two classical approaches. Since GD entails computing the full gradient ($O(N \times n)$ operations) at each iteration, for large values of $N \times n$ or in situations where n is infinite (as in the online setting (4.3)), SGD (with $O(1)$ computations per iteration) is the only viable option.

Empirically, SGD has been observed to have good performance initially, but its progress slows down near the solution. One of the reasons behind this is the variance inherent in the stochastic gradient estimator used. A number of variance reduction estimators, for example, SAGA [42] and SVRG [90], have been proposed in the literature to ameliorate this problem. See [68] for a survey of variance-reduction methods.

The IFO complexity required for finite-sum problems was first improved to $O((Nn)^{2/3}\epsilon^{-1})$ in [146, 8], and then further improved to $O((Nn)^{1/2}\epsilon^{-1})$ in [52, SPIDER], [187], [208, SNVRG], [141, SARAH]. Moreover, it is proved in [52] that $O((Nn)^{1/2}\epsilon^{-1})$ is the optimal complexity for problems where $N \times n \leq O(\epsilon^{-2})$. Similarly, for online problems, variance reduction methods first improved upon SGD to achieve the IFO complexity of $O(\epsilon^{-5/3})$ [106], which was again improved to $O(\epsilon^{-3/2})$ [52, SPIDER], [187]. It was shown in [9] that the IFO complexity of $O(\epsilon^{-3/2})$ is optimal for online problems.

All these methods achieve variance reduction by periodically computing high-precision gra-

---

[1] For example, smoothness of individual stochastic functions $\{f_i(\cdot;\xi_j\})$ in (4.2), (4.3), rather than just smoothness of the overall objective function. We shall use a slightly weaker assumption in our work (Assumption 1).

dient estimators using large batches of samples. In contrast, some recent works [38, 138] use adaptive gradient methods to achieve variance reduction. These methods, which derive motivation from popular adaptive methods like Adam [97], Adagrad [49], etc. do not require computation of large-batch gradient estimates. All the complexity results we just discussed, are summarized in Table 4.1.

Algorithm	IFO complexity
GD [136]	$O\left(Nn/\epsilon\right)$
SGD [62]	$O\left(1/\epsilon^2\right)$
SVRG [146]	$O\left(\frac{(Nn)^{2/3}}{\epsilon}\right)$
SCSG [106]	$O\left(\min\left\{\frac{(Nn)^{2/3}}{\epsilon}, \frac{1}{\epsilon^{5/3}}\right\}\right)$
SPIDER/SNVRG [52, 208]	$O\left(\min\left\{\frac{(Nn)^{1/2}}{\epsilon}, \frac{1}{\epsilon^{3/2}}\right\}\right)$
STORM [38]	$\tilde{O}\left(\frac{1}{\epsilon^{3/2}}\right)$

Table 4.1: IFO complexity of different algorithms to reach an ϵ-stationary point, for the stochastic smooth non-convex optimization problem. $\tilde{O}(\cdot)$ hides logarithmic factors.

Distributed Stochastic Gradient Descent (SGD)

There is a vast and ever-growing body of literature on distributed SGD. However, here we limit our discussion almost exclusively to work in nonconvex optimization. A classical method to solve (4.1) is Parallel Mini-batch SGD [44], [107]. In each iteration, all the workers, in parallel, carry a stochastic gradient update, and send their updated iterate to the server. The server aggregates these, and broadcasts the average back to the workers. The workers and the server repeat the same process.[2] The approach achieves an ϵ-FoS point, with a linear speedup with the number of workers. This is because the total IFO complexity $O(\epsilon^{-2})$ is independent of the number of workers N. Hence each node only computes $O(\frac{1}{N}\epsilon^{-2})$ gradients.

The exchange of gradients and iterates between the workers and the server at each iteration results in a significant communication requirement, leading to communication complexity of $O(\frac{1}{N}\epsilon^{-2})$. To alleviate the communication cost, several approaches have recently been proposed.

---

[2] Alternatively, the worker nodes might send their local gradient estimators to the server.

These include communicating compressed gradients to the server, as in quantized SGD [189, 7] or sparsified SGD [48, 6]. The motivation behind these is to reduce the communication cost, while not significantly affecting the convergence rate. The number of communication rounds, however, still remains the same.

Model Averaging: To cut back on the communication costs, the nodes might decide to make the communication and the subsequent averaging infrequent. The resulting class of algorithms is referred to as **Parallel Restarted SGD** or **local SGD**. The IFO complexity is still $O(\epsilon^{-2})$. However, savings on communication costs are demonstrated.

The entire algorithm is divided into *rounds*, each spanning I iterations. Within each round, all the N nodes run I steps of SGD *locally* and *in parallel*. At the end of each round, the worker send their updated iterates to the server. The server returns the average of these local iterates to the workers. The workers *restart* their local iterations in the next round from this *common* point. Hence the name Parallel Restarted SGD.

In [203], the proposed approach achieves communication savings using model averaging. Further reduction in the communication requirement is achieved in [201], by adding momentum to the vanilla SGD run locally at the nodes. The communication cost is again improved in [200], where the authors used dynamic batch sizes. For non-convex functions satisfying the Polyak-Lojasiewicz (PL) condition, linear speedup is achieved using only $O(\log(\epsilon^{-1}))$ communication rounds. And for general nonconvex problems, linear speedup is achieved, while using $O(\epsilon^{-1}\log(N^{-1}\epsilon^{-1})) = \tilde{O}(\epsilon^{-1})$ communication rounds ($\tilde{O}(\cdot)$ subsumes logarithmic factors).

Other approaches based on infrequent averaging include LAG [29], in which at every iteration, *fresh* gradients are requested only from a subset of the workers. However, the approach has only been explored in the deterministic setting. In [14], communication compression (quantization and sparsification) is incorporated into local SGD to further enhance communication savings. In [70], redundancy is introduced in the training data to achieve communication savings. The authors of [114] provide a comprehensive empirical study of distributed SGD, with focus on communication efficiency and generalization performance.

Work	Communication		IFO Complexity
	$\mathcal{D}_i \equiv \mathcal{D}$	$\mathcal{D}_i \neq \mathcal{D}_j$	
[44]	$O\left(\frac{1}{N\epsilon^2}\right)$	NA	$O\left(\frac{1}{\epsilon^2}\right)$
[203][3]	$O\left(\frac{1}{\epsilon^{3/2}}\right)$	$O\left(\frac{1}{\epsilon^{3/2}}\right)$	
[87]	$O\left(\frac{N^2}{\epsilon}\right)$	$O\left(\frac{\sqrt{N}}{\epsilon^{3/2}}\right)$	
[201]	$O\left(\frac{N}{\epsilon}\right)$	$O\left(\frac{1}{\epsilon^{3/2}}\right)$	
[200]	$\tilde{O}\left(\frac{1}{\epsilon}\right)$	NA	
[149]	$O\left(\frac{1}{\epsilon^{3/2}}\right)$	NA	
[71]	$O\left(\frac{1}{\epsilon}\right)$	$O\left(\frac{1}{\epsilon}\right)$	
[93]	$O\left(\frac{1}{\epsilon^{3/2}}\right)$	$O\left(\frac{1}{\epsilon^{3/2}}\right)$	$O\left(\frac{1}{\epsilon^{3/2}}\right)$
[40]	$O\left(\frac{1}{\epsilon^{3/2}}\right)$	$O\left(\frac{1}{\epsilon^{3/2}}\right)$	$O\left(\frac{1}{\epsilon^{3/2}}\right)$
Our work	$O\left(\frac{1}{\epsilon}\right)$	$O\left(\frac{1}{\epsilon}\right)$	$\min\left\{\frac{\sqrt{Nn}}{\epsilon}, \frac{1}{\epsilon^{3/2}}\right\}$

Table 4.2: Communication and computation complexities of different algorithms to reach an ϵ-FoS point, for the stochastic smooth non-convex optimization problem.

Federated Learning

Federated Learning (FL) [98] is a recently proposed edge-computing paradigm, that works under the same worker-server architecture we consider in this chapter. However, the number of workers/clients is typically very large. Therefore, rather than communicating with every worker node in each communication round, the server selects a small subset of the workers in each round, and only communicates with them [109]. This sampling of workers is what distinguishes FL from our work. We next outline some of the work in this direction which resembles ours.

After the seminal work in [123], a number of works have appeared which prove convergence of FL under different settings. In FedPAQ [149], the authors incorporated quantization into the worker-server communication, achieving IFO complexity of $O(1/\epsilon^2)$, with communication cost of $O(1/(N\sqrt{\epsilon}))$. Data heterogeneity is incorporated in [71]. Recently, adaptive gradient methods have been incorporated into the FL paradigm. In [40], the authors proposed an extension of the STORM [38] algorithm to FL. The workers compress the messages sent to the server, and the server samples only a subset of clients in each round. The IFO complexity achieved is optimal $O(1/\epsilon^{1.5})$, with $O(1/\epsilon^{1.5})$ rounds of communication required. In [93], the same guarantees are achieved,

---

[3] The approach in [203] requires the additional assumption that the gradients have bounded second moments.

but under the requirement of all the workers being sampled in each round. Table 4.2 contains a comparison of the IFO and comunication complexities of some of the methods in the preceding discussion, with our proposed approach. $\mathcal{D}_i$ denotes the distribution of data at node i ((4.3)). Note that not all approaches are applicable to the more general setting where the distributions at different nodes are non-identical.

Distributed Stochastic Variance Reduction Methods

The existing literature on distributed variance-reduction methods is almost exclusively focused on convex and strongly convex problems. Empirically, these methods have been shown to be promising [148, AIDE]. Single node SVRG requires gradient estimators at each iteration to be unbiased. This is a major challenge for distributed variants of SVRG. The existing approaches try to bypass this by simulating sampling extra data [105], [160, DANE], [183]. To the best of our knowledge, [27] is the only work that avoids this additional sampling.

4.1.3 Our Contributions

In this chapter, we propose a distributed variant of the SPIDER algorithm, Parallel Restarted SPIDER (PR-SPIDER), to solve the non-convex optimization problem (4.1). Note that PR-SPIDER is a non-trivial extension of both SPIDER and parallel-restarted SGD. This is because we need to optimize both communication and computation complexities. Averaging at every step, or too often, negatively impacts the communication savings. Too infrequent averaging leads to error terms building up as we see in the analysis, which has adverse impact on convergence.

- For the online setting (4.3), our proposed approach achieves the optimal overall (aggregated across nodes) IFO complexity of $O\left(\frac{\sigma}{\epsilon^{3/2}} + \frac{\sigma^2}{\epsilon}\right)$. This result improves the long-standing best known result of $O(\sigma^2\epsilon^{-2})$, while also achieving the linear speedup achieved in the existing literature. The communication complexity achieved $O(\epsilon^{-1})$ is also the best known in the literature. To the best of our knowledge, the only other work to achieve the same communication complexity is [200].

- For the finite-sum problem (4.2), our proposed approach achieves the overall IFO complexity of $O(\sqrt{Nn}\epsilon^{-1})$. For problems where $N \times n \leq O(\epsilon^{-2})$, this is the optimal result one can achieve, even if all the $N \times n$ functions are available at a single location [52]. At the same time, we also achieve the best known communication complexity $O(\epsilon^{-1})$.

- Compared to several existing approaches which require the samples across nodes to follow the same distribution, our approach is more general in the sense that the data distribution across nodes may be different (the federated learning problem [98]).

4.1.4 Notations and Assumptions

Given a positive integer N, the set $\{1, \ldots, N\}$ is denoted by the shorthand $[1 : N]$. $\|\cdot\|$ denotes the vector ℓ_2 norm. Boldface letters are used to denote vectors. We use $x \wedge y$ to denote the minimum of two numbers $x, y \in \mathbb{R}$.

Following assumptions hold for the rest of the chapter.

Assumption 1. *Lipschitz-ness:* All the functions are mean-squared L-smooth.[4]

$$\mathbb{E}_\xi \|\nabla f(\mathbf{x}; \xi) - \nabla f(\mathbf{y}; \xi)\|^2 \leq L^2 \|\mathbf{x} - \mathbf{y}\|^2, \forall\, \mathbf{x}, \mathbf{y} \in \mathbb{R}^d.$$

Assumption 2. All the nodes begin the algorithm from the same starting point $\mathbf{x}^0$.

Assumption 3. Unbiasedness: All the stochastic gradients are unbiased, i.e., $\mathbb{E}_\xi \nabla f_i(\mathbf{x}; \xi) = \nabla f_i(\mathbf{x})$, $\forall\, i \in [1 : N], \forall\, \mathbf{x}$.

In the next section, we discuss our approach to solve the finite-sum problem (4.2). We propose PR-SPIDER, a distributed, parallel variant of the SPIDER algorithm [52, 187], followed by its convergence analysis.

---

[4] This assumption might seem stringent. However, we can always choose L as the maximum Lipschitz constant corresponding to all the functions across all the nodes.

4.2 Parallel Restarted SPIDER - Finite Sum Case

We consider a network of N worker nodes connected to a server node. The objective is to solve (4.2). Note that the number of sample functions at different nodes $i \neq j$, for $i, j \in [1:N]$, can be non-uniform, i.e., $n_i \neq n_j$. However, to ease the notational burden slightly, we assume $n_j = n$, $\forall\, j \in [1:N]$.

4.2.1 Proposed Algorithm

The proposed algorithm is inspired by the recently proposed SPIDER algorithm [52, 187] for single-node stochastic nonconvex optimization. Like numerous variance-reduced approaches proposed in the literature, our algorithm also proceeds in epochs.

Algorithm 5 PR-SPIDER - Finite Sum Case

```
Input: Initial iterate $\mathbf{x}^0_{i,m} = \mathbf{x}^0, \mathbf{v}^0_{i,m} = \nabla f\left(\mathbf{x}^0\right) \forall\, i \in [1:N]$
for $s = 0$ to $S-1$ do
    $\mathbf{x}^{s+1}_{i,0} = \mathbf{x}^{s}_{i,m}, \forall\, i \in [1:N]$
    $\mathbf{v}^{s+1}_{i,0} = \mathbf{v}^{s}_{i,m}, \forall\, i \in [1:N]$
    $\mathbf{x}^{s+1}_{i,1} = \mathbf{x}^{s+1}_{i,0} - \gamma \mathbf{v}^{s+1}_{i,0}$
    for $t = 1$ to $m-1$ do
        Compute $\mathbf{v}^{s+1}_{i,t}$, using (4.4) $\forall\, i \in [1:N]$
        if $t \bmod I = 0$ then
            $\mathbf{x}^{s+1}_{i,t} = \bar{\mathbf{x}}^{s+1}_{t} \triangleq \frac{1}{N}\sum_{j=1}^{N} \mathbf{x}^{s+1}_{j,t}, \forall\, i \in [1:N]$
            $\mathbf{v}^{s+1}_{i,t} = \bar{\mathbf{v}}^{s+1}_{t} \triangleq \frac{1}{N}\sum_{j=1}^{N} \mathbf{v}^{s+1}_{j,t}, \forall\, i \in [1:N]$
        end if
        $\mathbf{x}^{s+1}_{i,t+1} = \mathbf{x}^{s+1}_{i,t} - \gamma \mathbf{v}^{s+1}_{i,t}, \forall\, i \in [1:N]$
    end for
    if $s < S-1$ then
        $\bar{\mathbf{x}}^{s+1}_{m} = \frac{1}{N}\sum_{j=1}^{N} \mathbf{x}^{s+1}_{j,m}$
        $\mathbf{x}^{s+1}_{i,m} = \bar{\mathbf{x}}^{s+1}_{m}, \forall\, i \in [1:N]$
        $\bar{\mathbf{v}}^{s+1}_{m} = \frac{1}{N}\sum_{j=1}^{N} \nabla f_j\left(\mathbf{x}^{s+1}_{j,m}\right) = \nabla f\left(\bar{\mathbf{x}}^{s+1}_{m}\right)$
        $\mathbf{v}^{s+1}_{i,m} = \bar{\mathbf{v}}^{s+1}_{m}, \forall\, i \in [1:N]$
    end if
end for
Return
```

At the beginning of each epoch, each worker node has access to the same iterate, and the full gradient $\nabla f(\cdot)$ computed at this value. These are used to compute the first iterate of the epoch $\mathbf{x}_{i,1}^{s+1}, \forall\, i \in [1:N]$. This is followed by the inner loop (step 6-13). At iteration t in s-th epoch, the worker nodes first compute an estimator of the gradient, $\mathbf{v}_{i,t}^{s+1}$ as follows (step 7).

$$\mathbf{v}_{i,t}^{s+1} = \mathbf{v}_{i,t-1}^{s+1} + \nabla f_i(\mathbf{x}_{i,t}^{s+1}; \mathcal{B}_{i,t}^{s+1}) - \nabla f_i(\mathbf{x}_{i,t-1}^{s+1}; \mathcal{B}_{i,t}^{s+1}), \tag{4.4}$$

where $\nabla f_i(\mathbf{x}_{i,t}^{s+1}; \mathcal{B}_{i,t}^{s+1}) = \frac{1}{B} \sum_{\xi \in \mathcal{B}_{i,t}^{s+1}} \nabla f_i(\mathbf{x}_{i,t}^{s+1}; \xi)$, and $|\mathcal{B}_{i,t}^{s+1}| = B, \forall\, i, t, s$. This estimator is computed iteratively, using the previous estimate $\mathbf{v}_{i,t-1}^{s+1}$, the current iterate $\mathbf{x}_{i,t}^{s+1}$, and the previous iterate $\mathbf{x}_{i,t-1}^{s+1}$. The sample set $\mathcal{B}_{i,t}^{s+1}$ of size B is picked uniformly randomly at each node, and independent of the other nodes. Such an estimator has been proposed in the literature for single-node stochastic nonconvex optimization [52, 187, 138, 140], and has even been proved to be optimal in certain regimes.

Using the gradient estimator, the worker node computes the next iterate $\mathbf{x}_{i,t+1}^{s+1}$. This process is repeated $m-1$ times. Once every I iterations of the inner loop (whenever t mod $I = 0$), the nodes send their local iterates and gradient estimators $\{\mathbf{x}_{i,t}^{s+1}, \mathbf{v}_{i,t}^{s+1}\}_{i=1}^{N}$ to the server node. The server, in turn, computes their averages and returns the averages $\{\bar{\mathbf{x}}_{t}^{s+1}, \bar{\mathbf{v}}_{t}^{s+1}\}$ to all the nodes (steps 8-11). The next iteration at each node proceeds using this iterate and direction.

At the end of the inner loop $(t = m)$, all the worker nodes send their local iterates $\{\mathbf{x}_{i,m}^{s+1}\}_{i=1}^{N}$ to the server. The server computes the model average $\bar{\mathbf{x}}_{m}^{s+1}$, and returns it to all the workers (steps 15-16). The workers compute the full gradient of their respective functions $\{\nabla f_i(\bar{\mathbf{x}}_{m}^{s+1})\}_{i=1}^{N}$ at this point, and send it to the server. The server averages these, and returns this average (which is essentially $\nabla f(\bar{\mathbf{x}}_{m}^{s+1})$) to the worker nodes (steps 17-18). Consequently, all the worker nodes start the next epoch at the same point, and along the same descent direction. This "restart" of the local computation is along the lines of Parallel-Restarted SGD [166, 203].

4.2.2 Convergence Result

Theorem 4.2.1. *For the finite-sum problem under Assumptions 1, 3, for small enough step size* $0 < \gamma < \frac{1}{8IL}$*,*

$$\min_{s,t} \left[\mathbb{E} \left\| \nabla f \left(\bar{\mathbf{x}}_t^{s+1} \right) \right\|^2 + \frac{1}{N} \sum_{i=1}^{N} \mathbb{E} \left\| \mathbf{x}_{i,t}^{s+1} - \bar{\mathbf{x}}_t^{s+1} \right\|^2 \right] \leq \frac{2 \left(f(\mathbf{x}^0) - f_* \right)}{T\gamma}. \tag{4.5}$$

The above result now can be directly used to compute the bounds on the sample complexity (see Definition 4.1.2) and the communication complexity (see Definition 4.1.3) of the proposed algorithm PR-SPIDER for the finite sum problem (4.2).

Computation and Communication Complexity

Based on Theorem 4.2.1, we can bound the computation (IFO) and communication complexity (Definition 4.1.2, 4.1.3) of Algorithm 5.

Corollary 1. *To reach an* ϵ*-FoS point (Definition 4.1.1), the IFO complexity is bounded as*

$$O\left(\frac{\sqrt{Nn}}{\epsilon}\right) \qquad \text{for } N \times n \leq O(\epsilon^{-2}).$$

The communication complexity (the number of communication rounds) is bounded as

$$O\left(\frac{1}{\epsilon}\right).$$

PROOF: Given (4.5), at an ϵ-FoS point, the total number of iterations T must satisfy

$$\frac{2 \left(f(\mathbf{x}^0) - f_* \right)}{T\gamma} = \epsilon \quad \Rightarrow \quad T = \frac{2 \left(f(\mathbf{x}^0) - f_* \right)}{\gamma\epsilon} = \frac{CI}{\epsilon}, \tag{4.6}$$

for constants C, I. Choose the number of iterations per epoch $m = I\sqrt{Nn}$, and the per-iteration mini-batch size (see (4.4)) $B = \frac{1}{I}\sqrt{\frac{n}{N}}$. Recall that once every epoch, all the N nodes compute their full gradients (line 17, Algorithm 5). Also, in every inner iteration of each epoch, each node computes two stochastic gradients, each with mini-batch B (line 7). Consequently, the IFO complexity is bounded as

$$\begin{aligned}
&N \times \left(\left\lceil \frac{T}{m} \right\rceil \cdot n + T \cdot 2B \right) \leq N \times \left(\left(\frac{CI}{m\epsilon} + 1 \right) n + \frac{CI}{\epsilon} 2B \right) \\
&\leq N \times \left(\frac{CI}{\epsilon} \left(\frac{n}{m} + 2B \right) + n \right) \\
&= O\left(\frac{\sqrt{Nn}}{\epsilon} \right) \qquad\qquad \text{for } N \times n \leq O(\epsilon^{-2}).
\end{aligned}$$

Since communication happens once every I iterations, the number of communication rounds is bounded using (4.6) as $\lceil \frac{T}{I} \rceil \leq 1 + \frac{C}{\epsilon} = O\left(\frac{1}{\epsilon}\right)$. ■

Remark. (Optimality and Linear Speedup)

- This is the optimal sample (IFO complexity achievable for $N \times n \leq O(\epsilon^{-2})$ (see [52] for the single node result).
- Given the total IFO cost in Corollary 1, each of the N nodes needs to compute $O\left(\frac{1}{\epsilon}\sqrt{\frac{n}{N}}\right)$ stochastic gradients, $\frac{1}{N}$ the total number of stochastic gradients needed. Hence, increasing the number of nodes speeds up the algorithm linearly in N.

Next, we present the proof of Theorem 4.2.1.

4.2.3 Convergence Analysis

We first give a brief outline of the reasoning behind the proof. If we were to run our algorithm in a single node setting ($N = 1$), with the update equation $\mathbf{x}_{t+1}^{s+1} = \mathbf{x}_t^{s+1} - \gamma \mathbf{v}_t^{s+1}$, the only source of error would be the mismatch between the gradient estimate $\mathbf{v}_t^{s+1}$ and the actual gradient $\nabla f(\mathbf{x}_t^{s+1})$. With $N > 1$ nodes, where each node only has access to part of the objective function, two additional sources of error need to be accounted for.

- Mismatch between local iterates $\sum_{i=1}^{N} \mathbb{E}\|\mathbf{x}_{i,t}^{s+1} - \bar{\mathbf{x}}_t^{s+1}\|^2$;

- Mismatch between the local gradient estimators $\sum_{i=1}^{N} \mathbb{E}\|\mathbf{v}_{i,t}^{s+1} - \bar{\mathbf{v}}_t^{s+1}\|^2$.

As we shall see, effectively bounding the errors arising from these three sources requires periodic averaging of both the iterates and the local gradient estimators (lines 9, 10), and will form a crucial part of our analysis.

We begin with the L-smoothness of f (Assumption 1). Averaging the local iterate updates (line 12), we get $\bar{\mathbf{x}}_{t+1}^{s+1} = \bar{\mathbf{x}}_t^{s+1} - \gamma\bar{\mathbf{v}}_t^{s+1}$. Consequently,

$$\begin{aligned}
\mathbb{E}f\left(\bar{\mathbf{x}}_{t+1}^{s+1}\right) &\leq \mathbb{E}f\left(\bar{\mathbf{x}}_t^{s+1}\right) - \gamma\mathbb{E}\left\langle\nabla f\left(\bar{\mathbf{x}}_t^{s+1}\right), \bar{\mathbf{v}}_t^{s+1}\right\rangle + \frac{\gamma^2 L}{2}\mathbb{E}\|\bar{\mathbf{v}}_t^{s+1}\|^2 \\
&= \mathbb{E}f\left(\bar{\mathbf{x}}_t^{s+1}\right) - \frac{\gamma}{2}\mathbb{E}\|\nabla f\left(\bar{\mathbf{x}}_t^{s+1}\right)\|^2 - \frac{\gamma}{2}(1 - L\gamma)\mathbb{E}\|\bar{\mathbf{v}}_t^{s+1}\|^2 \\
&\quad + \frac{\gamma}{2}\mathbb{E}\|\nabla f\left(\bar{\mathbf{x}}_t^{s+1}\right) - \bar{\mathbf{v}}_t^{s+1}\|^2, \qquad (4.7)
\end{aligned}$$

where we use $\langle a, b\rangle = \frac{||a||^2 + ||b||^2 - ||a-b||^2}{2}$. The last term in (4.7) quantifies the difference between the gradient of f at the average iterate and the average of local descent directions. Next, we bound this term. We assume $t > 0$ (for $t = 0, \mathbb{E}\|\nabla f(\bar{\mathbf{x}}_0^{s+1}) - \bar{\mathbf{v}}_0^{s+1}\|^2 = 0$).

$$\mathbb{E}\|\nabla f\left(\bar{\mathbf{x}}_t^{s+1}\right) - \bar{\mathbf{v}}_t^{s+1}\|^2 \leq \frac{2L^2}{N}\sum_{i=1}^{N}\mathbb{E}\left\|\mathbf{x}_{i,t}^{s+1} - \bar{\mathbf{x}}_t^{s+1}\right\|^2 + 2\mathbb{E}\left\|\bar{\mathbf{v}}_t^{s+1} - \frac{1}{N}\sum_{i=1}^{N}\nabla f_i\left(\mathbf{x}_{i,t}^{s+1}\right)\right\|^2 \qquad (4.8)$$

where (4.8) follow from $\mathbb{E}\|\sum_{i=1}^{n} X_i\|^2 \leq n\sum_{i=1}^{n}\mathbb{E}\|X_i\|^2$ and the mean-squared L-smoothness (Assumption 1). The first term in (4.8) is the consensus error of local iterates, while the second term is the gradient error. In the following two lemmas, we bound both these errors.

Gradient Estimate Error

First, we bound the error in the average (across nodes) gradient estimator.

Lemma 4.2.2. *For* $0 < t < m, 0 \leq s \leq S-1$, *the sequence of iterates* $\{\mathbf{x}_{i,t}^{s+1}\}_{i,t}$ *and* $\{\mathbf{v}_{i,t}^{s+1}\}_{i,t}$

generated by Algorithm 5 satisfies

$$\mathbb{E}\left\|\bar{\mathbf{v}}_t^{s+1}-\frac{1}{N}\sum_{i=1}^{N}\nabla f_i\left(\mathbf{x}_{i,t}^{s+1}\right)\right\|^2 \le \underbrace{\mathbb{E}\left\|\bar{\mathbf{v}}_0^{s+1}-\frac{1}{N}\sum_{i=1}^{N}\nabla f_i\left(\mathbf{x}_{i,0}^{s+1}\right)\right\|^2}_{E_0^{s+1}} + \frac{L^2}{N^2B}\sum_{i=1}^{N}\sum_{\ell=0}^{t-1}\mathbb{E}\left\|\mathbf{x}_{i,\ell+1}^{s+1}-\mathbf{x}_{i,\ell}^{s+1}\right\|^2 \tag{4.9}$$

PROOF: We have relegated the proof to Appendix A.3.1. However, a few comments are in order. Due to the recursive nature of the gradient estimator in (4.4), we can express the average gradient estimator at time t, $\bar{\mathbf{v}}_t^{s+1}$ in terms of the average estimator at time $t-1$, $\bar{\mathbf{v}}_{t-1}^{s+1}$,

$$\bar{\mathbf{v}}_t^{s+1} = \bar{\mathbf{v}}_{t-1}^{s+1} + \frac{1}{N}\sum\nolimits_{i=1}^{N}\left[\nabla f_i(\mathbf{x}_{i,t}^{s+1};\mathcal{B}_{i,t}^{s+1}) - \nabla f_i(\mathbf{x}_{i,t-1}^{s+1};\mathcal{B}_{i,t}^{s+1})\right]. \tag{4.10}$$

This process is repeated recursively. Owing to the computation of full-gradients and the subsequent averaging at the start of each epoch (lines 17-18), we can bound the error accumulation in the average gradient estimator in terms of the corresponding error at the beginning of the epoch, and the average cumulative difference of the consecutive local iterates (second term in (4.9)). ■

Remark. In the bound in Lemma 4.2.2, we define the first term as E_0^{s+1}. Note that by Algorithm 5 (for finite-sum problems),

$$\begin{aligned}
\bar{\mathbf{v}}_0^{s+1} &= \frac{1}{N}\sum\nolimits_{i=1}^{N}\mathbf{v}_{i,0}^{s+1} = \frac{1}{N}\sum\nolimits_{i=1}^{N}\mathbf{v}_{i,m}^{s} = \bar{\mathbf{v}}_m^{s} && \text{(steps 4, 18 in Algorithm 5)}\\
&= \frac{1}{N}\sum\nolimits_{i=1}^{N}\nabla f_j\left(\bar{\mathbf{x}}_m^{s}\right) && \text{(steps 15-17 in Algorithm 5)}\\
&= \frac{1}{N}\sum\nolimits_{i=1}^{N}\nabla f_j\left(\bar{\mathbf{x}}_{j,0}^{s+1}\right) && \text{(step 3 in Algorithm 5)}
\end{aligned}$$

Therefore, $E_0^{s+1} = 0, \forall s$. However, we retain it (as in [170]), as it shall be needed in the analysis of online problems.

Network Disagreements Errors

Next, we bound the network disagreements of the local estimates relative to global averages. There are two such error terms, corresponding to: 1) the local gradient estimators, and 2) the local iterates. For this purpose, we first define

$$\tau(\ell) = \begin{cases} \arg\max_j \{j \mid j < \ell, j \text{ mod } I = 0\} & \text{if } \ell \text{ mod } I \neq 0 \\ \ell & \text{otherwise.} \end{cases} \tag{4.11}$$

Note that, $\tau(\ell)$ is the largest iteration index smaller than (or equal to) ℓ, which is a multiple of I. Basically, looking back from ℓ, $\tau(\ell)$ is the latest time index when averaging happened in the current epoch (steps 9-10).

Lemma 4.2.3. *Given* $0 \leq \ell \leq m$, $\alpha > 0, \delta > 0, \theta > 0$ *such that* $\theta = \delta + 8\gamma^2 L^2(1 + \frac{1}{\delta})$*. For* ℓ *mod* $I \neq 0$,

$$\sum_{i=1}^{N} \mathbb{E}\left\|\mathbf{v}_{i,\ell}^{s+1} - \bar{\mathbf{v}}_{\ell}^{s+1}\right\|^2 \leq 8\gamma^2 N L^2 \left(1 + \frac{1}{\delta}\right) \sum_{j=\tau(\ell)}^{\ell-1} (1+\theta)^{\ell-1-j} \mathbb{E}\left\|\bar{\mathbf{v}}_{j}^{s+1}\right\|^2 \tag{4.12}$$

$$\sum_{i=1}^{N} \mathbb{E}\left\|\mathbf{x}_{i,\ell}^{s+1} - \bar{\mathbf{x}}_{\ell}^{s+1}\right\|^2 \leq \left(1 + \frac{1}{\alpha}\right) \gamma^2 \sum_{i=1}^{N} \sum_{j=\tau(\ell)+1}^{\ell-1} (1+\alpha)^{\ell-1-j} \mathbb{E}\left\|\mathbf{v}_{i,j}^{s+1} - \bar{\mathbf{v}}_{j}^{s+1}\right\|^2 \tag{4.13}$$

If ℓ *mod* $I = 0$, $\sum_{i=1}^{N} \mathbb{E}\|\mathbf{v}_{i,\ell}^{s+1} - \bar{\mathbf{v}}_{\ell}^{s+1}\|^2 = \sum_{i=1}^{N} \mathbb{E}\|\mathbf{x}_{i,\ell}^{s+1} - \bar{\mathbf{x}}_{\ell}^{s+1}\|^2 = 0.$

PROOF: See Appendix A.3.2. ■

Remark. Following a similar reasoning as in Lemma 4.2.2, we use the recursive nature of the gradient estimator (4.10) to bound the network error at time t

$$\sum_{i=1}^{N} \mathbb{E}\|\mathbf{v}_{i,t}^{s+1} - \bar{\mathbf{v}}_{t}^{s+1}\|^2$$

in terms of the corresponding error at time $t-1$. Owing to the periodic averaging of local $\{\mathbf{v}_{i,t}^{s+1}\}_i$ estimates every I iterations (line 10), this error build-up over time is limited. More precisely, in the absence of within-epoch averaging, the sum over time in (4.12) would start at $j = 0$, rather than $j = \tau(\ell)$, leading to a greater error.

Remark. Using an analogous reasoning, we bound the network error corresponding to local iterates at time t

$$\textstyle\sum_{i=1}^{N} \mathbb{E}\|\mathbf{x}_{i,t}^{s+1} - \bar{\mathbf{x}}_t^{s+1}\|^2$$

in terms of the corresponding error at time $t-1$. Here, we see the need for periodic averaging of local iterates $\{\mathbf{x}_{i,t}^{s+1}\}_i$ estimates every I iterations (line 9).

Now, substituting the bound derived in Lemma 4.2.2 in (4.8), we get

$$\begin{aligned} &\mathbb{E}\left\|\nabla f\left(\bar{\mathbf{x}}_t^{s+1}\right) - \bar{\mathbf{v}}_t^{s+1}\right\|^2 \\ &\leq \frac{2L^2}{N}\sum_{i=1}^{N}\mathbb{E}\|\mathbf{x}_{i,t}^{s+1} - \bar{\mathbf{x}}_t^{s+1}\|^2 + 2E_0^{s+1} + \frac{2L^2}{N^2B}\sum_{i=1}^{N}\sum_{\ell=0}^{t-1}\mathbb{E}\left\|\mathbf{x}_{i,\ell+1}^{s+1} - \mathbf{x}_{i,\ell}^{s+1}\right\|^2 \\ &\leq \frac{4\gamma^2L^2}{N^2B}\sum_{i=1}^{N}\sum_{\ell=0}^{t-1}\left[\mathbb{E}\|\mathbf{v}_{i,\ell}^{s+1} - \bar{\mathbf{v}}_\ell^{s+1}\|^2 + \mathbb{E}\|\bar{\mathbf{v}}_\ell^{s+1}\|^2\right] + \frac{2L^2}{N}\sum_{i=1}^{N}\mathbb{E}\left\|\mathbf{x}_{i,t}^{s+1} - \bar{\mathbf{x}}_t^{s+1}\right\|^2 + 2E_0^{s+1}. \end{aligned} \tag{4.14}$$

Using the upper bounds on the two network error terms from Lemma 4.2.3 in (4.14), we get

$$\begin{aligned} \mathbb{E}\left\|\nabla f\left(\bar{\mathbf{x}}_t^{s+1}\right) - \bar{\mathbf{v}}_t^{s+1}\right\|^2 &\leq \frac{4\gamma^2L^2}{N^2B}N\sum_{\ell=0}^{t-1}\mathbb{E}\left\|\bar{\mathbf{v}}_\ell^{s+1}\right\|^2 + 2E_0^{s+1} \\ &+ \frac{2L^2}{N}\left(1+\frac{1}{\alpha}\right)\gamma^2\sum_{\ell=\tau(t)+1}^{t-1}(1+\alpha)^{t-1-\ell}8\gamma^2NL^2\big(1+\tfrac{1}{\delta}\big)\sum_{j=\tau(\ell)}^{\ell-1}(1+\theta)^{\ell-1-j}\mathbb{E}\left\|\bar{\mathbf{v}}_j^{s+1}\right\|^2 \\ &+ \frac{4\gamma^2L^2}{N^2B}\sum_{\ell=0}^{t-1}8\gamma^2NL^2\big(1+\tfrac{1}{\delta}\big)\sum_{j=\tau(\ell)}^{\ell-1}(1+\theta)^{\ell-1-j}\mathbb{E}\left\|\bar{\mathbf{v}}_j^{s+1}\right\|^2. \end{aligned} \tag{4.15}$$

Remark. Again, note that in the last two terms in (4.15), in the absence of any averaging within an epoch, the summation $\sum_{j=\tau(\ell)}^{\ell-1}(1+\theta)^{\ell-1-j}\mathbb{E}\|\bar{\mathbf{v}}_j^{s+1}\|^2$ would instead start from $j = 0$, leading to greater error. To check this rapid error build-up, we introduced within-epoch averaging every I iterations (steps 9-10) in Algorithm 5.

We substitute this upper bound in (4.7) to derive the following descent lemma on function values.

Descent over one epoch

Lemma 4.2.4. *In each epoch* $s \in [1, S]$,

$$\mathbb{E}f\left(\bar{\mathbf{x}}_m^{s+1}\right) \le \mathbb{E}f\left(\bar{\mathbf{x}}_0^{s+1}\right) - \frac{\gamma}{2}\sum_{t=0}^{m-1}\mathbb{E}\left\|\nabla f\left(\bar{\mathbf{x}}_t^{s+1}\right)\right\|^2 - \frac{\gamma}{2}(1-L\gamma)\sum_{t=0}^{m-1}\mathbb{E}\left\|\bar{\mathbf{v}}_t^{s+1}\right\|^2 + \sum_{t=0}^{m-1}\mathbb{E}\left\|\bar{\mathbf{v}}_t^{s+1}\right\|^2\left[\frac{64\gamma^5L^4m}{NB\delta^2} + \frac{2\gamma^3L^2m}{NB} + \frac{256\gamma^5L^4}{\delta^4}\right]. \tag{4.16}$$

PROOF: Substituting the upper bound (4.15) in (4.7) we get

$$\begin{aligned}
&\mathbb{E}f\left(\bar{\mathbf{x}}_{t+1}^{s+1}\right)\\
&\le \mathbb{E}f\left(\bar{\mathbf{x}}_t^{s+1}\right) - \frac{\gamma}{2}\mathbb{E}\left\|\nabla f\left(\bar{\mathbf{x}}_t^{s+1}\right)\right\|^2 - \frac{\gamma}{2}(1-L\gamma)\mathbb{E}\left\|\bar{\mathbf{v}}_t^{s+1}\right\|^2 + \frac{\gamma}{2}2E_0^{s+1}\\
&\quad + \frac{\gamma}{2}\frac{4\gamma^2L^2}{N^2B}\sum_{\ell=0}^{t-1}8\gamma^2NL^2\big(1+\frac{1}{\delta}\big)\sum_{j=\tau(\ell)}^{\ell-1}(1+\theta)^{\ell-1-j}\mathbb{E}\left\|\bar{\mathbf{v}}_j^{s+1}\right\|^2 + \frac{\gamma}{2}\frac{4\gamma^2L^2}{N^2B}N\sum_{\ell=0}^{t-1}\mathbb{E}\left\|\bar{\mathbf{v}}_\ell^{s+1}\right\|^2\\
&\quad + \frac{\gamma}{2}\frac{2L^2}{N}\left(1+\frac{1}{\alpha}\right)\gamma^2\sum_{\ell=\tau(t)+1}^{t-1}(1+\alpha)^{t-1-\ell}8\gamma^2NL^2\big(1+\frac{1}{\delta}\big)\sum_{j=\tau(\ell)}^{\ell-1}(1+\theta)^{\ell-1-j}\mathbb{E}\left\|\bar{\mathbf{v}}_j^{s+1}\right\|^2\\
&= \mathbb{E}f\left(\bar{\mathbf{x}}_t^{s+1}\right) - \frac{\gamma}{2}\mathbb{E}\left\|\nabla f\left(\bar{\mathbf{x}}_t^{s+1}\right)\right\|^2 - \frac{\gamma}{2}(1-L\gamma)\mathbb{E}\left\|\bar{\mathbf{v}}_t^{s+1}\right\|^2 + \gamma E_0^{s+1}\\
&\quad + \frac{16\gamma^5L^4}{NB}\big(1+\frac{1}{\delta}\big)\sum_{\ell=0}^{t-1}\sum_{j=\tau(\ell)}^{\ell-1}(1+\theta)^{\ell-1-j}\mathbb{E}\left\|\bar{\mathbf{v}}_j^{s+1}\right\|^2 + \frac{2\gamma^3L^2}{NB}\sum_{\ell=0}^{t-1}\mathbb{E}\left\|\bar{\mathbf{v}}_\ell^{s+1}\right\|^2\\
&\quad + 8\gamma^5L^4\left(1+\frac{1}{\alpha}\right)\left(1+\frac{1}{\delta}\right)\sum_{\ell=\tau(t)+1}^{t-1}(1+\alpha)^{t-1-\ell}\sum_{j=\tau(\ell)}^{\ell-1}(1+\theta)^{\ell-1-j}\mathbb{E}\left\|\bar{\mathbf{v}}_j^{s+1}\right\|^2
\end{aligned} \tag{4.17}$$

Note that, as discussed earlier, for finite sum problems, $E_0^{s+1} \triangleq \mathbb{E}\left\|\bar{\mathbf{v}}_0^{s+1} - \frac{1}{N}\sum_{i=1}^N \nabla f_i\left(\mathbf{x}_{i,0}^{s+1}\right)\right\|^2 = 0$. Further, summing (4.17) over $t = 0, \ldots, m-1$, we get

$$\mathbb{E}f\left(\bar{\mathbf{x}}_m^{s+1}\right) \le \mathbb{E}f\left(\bar{\mathbf{x}}_0^{s+1}\right) - \frac{\gamma}{2}\sum_{t=0}^{m-1}\mathbb{E}\left\|\nabla f\left(\bar{\mathbf{x}}_t^{s+1}\right)\right\|^2 - \frac{\gamma}{2}(1-L\gamma)\sum_{t=0}^{m-1}\mathbb{E}\left\|\bar{\mathbf{v}}_t^{s+1}\right\|^2$$

$$+\frac{16\gamma^5L^4}{NB}(1+\frac{1}{\delta})\sum_{t=0}^{m-1}\sum_{\ell=0}^{t-1}\sum_{j=\tau(\ell)}^{\ell-1}(1+\theta)^{\ell-1-j}\mathbb{E}\left\|\bar{\mathbf{v}}_j^{s+1}\right\|^2+\frac{2\gamma^3L^2}{NB}\sum_{t=0}^{m-1}\sum_{\ell=0}^{t-1}\mathbb{E}\left\|\bar{\mathbf{v}}_\ell^{s+1}\right\|^2$$

$$+8\gamma^5L^4\left(1+\frac{1}{\alpha}\right)(1+\frac{1}{\delta})\sum_{t=0}^{m-1}\sum_{\ell=\tau(t)+1}^{t-1}\sum_{j=\tau(\ell)}^{\ell-1}(1+\alpha)^{t-1-\ell}(1+\theta)^{\ell-1-j}\mathbb{E}\left\|\bar{\mathbf{v}}_j^{s+1}\right\|^2 \tag{4.18}$$

Substituting the algebraic upper bounds derived in Lemmas A.3.1, A.3.2 and A.3.3, we get the result of Lemma 4.2.4. ■

Lemma 4.2.4 quantifies the decay in function value across one epoch. Clearly, the extent of decay depends on the precise values of algorithm parameters γ, δ, B, m.

Proof of Theorem 4.2.1

Rearranging the terms in (4.16),

$$\begin{aligned}&\sum_{t=0}^{m-1}\mathbb{E}\left\|\nabla f\left(\bar{\mathbf{x}}_t^{s+1}\right)\right\|^2+\left(1-L\gamma-\left[\frac{128\gamma^4L^4m}{NB\delta^2}+\frac{4\gamma^2L^2m}{NB}+\frac{512\gamma^4L^4}{\delta^4}\right]\right)\sum_{t=0}^{m-1}\mathbb{E}\left\|\bar{\mathbf{v}}_t^{s+1}\right\|^2\\&\quad\leq\frac{2}{\gamma}\left(\mathbb{E}f\left(\bar{\mathbf{x}}_0^{s+1}\right)-\mathbb{E}f\left(\bar{\mathbf{x}}_m^{s+1}\right)\right).\end{aligned} \tag{4.19}$$

Further, summing across all epochs $s = 0, \ldots, S-1$, and dividing by T, we get

$$\begin{aligned}&\frac{1}{T}\sum_{s=0}^{S-1}\sum_{t=0}^{m-1}\left[\mathbb{E}\left\|\nabla f\left(\bar{\mathbf{x}}_t^{s+1}\right)\right\|^2+\left(1-L\gamma-\left[\frac{128\gamma^4L^4m}{NB\delta^2}+\frac{4\gamma^2L^2m}{NB}+\frac{512\gamma^4L^4}{\delta^4}\right]\right)\mathbb{E}\left\|\bar{\mathbf{v}}_t^{s+1}\right\|^2\right]\\&\quad\leq\frac{2}{T\gamma}\sum_{s=0}^{S-1}\left(\mathbb{E}f\left(\bar{\mathbf{x}}_0^{s+1}\right)-\mathbb{E}f\left(\bar{\mathbf{x}}_m^{s+1}\right)\right)=\frac{2}{T\gamma}\left(\mathbb{E}f\left(\bar{\mathbf{x}}^{0}\right)-\mathbb{E}f\left(\bar{\mathbf{x}}_m^{S+1}\right)\right)\\&\quad\leq\frac{2\left(f(\mathbf{x}^0)-f_*\right)}{T\gamma},\end{aligned} \tag{4.20}$$

where $f_* = \min_x f(x)$. Note that in (4.20), for small enough, but constant step size γ, we can ensure that

$$1-L\gamma-\left[\frac{128\gamma^4L^4m}{NB\delta^2}+\frac{4\gamma^2L^2m}{NB}+\frac{512\gamma^4L^4}{\delta^4}\right]\geq\frac{1}{2}. \tag{4.21}$$

In Appendix A.3.4, we see that $\gamma = \frac{1}{8LI}$ which satisfies (4.21). Further, note that

$$\begin{aligned}
&\min_{s,t}\left[\mathbb{E}\left\|\nabla f\left(\bar{\mathbf{x}}_t^{s+1}\right)\right\|^2 + \frac{1}{N}\sum_{i=1}^{N}\mathbb{E}\left\|\mathbf{x}_{i,t}^{s+1} - \bar{\mathbf{x}}_t^{s+1}\right\|^2\right] \\
&\leq \frac{1}{T}\sum_{s=0}^{S-1}\sum_{t=0}^{m-1}\left[\mathbb{E}\left\|\nabla f\left(\bar{\mathbf{x}}_t^{s+1}\right)\right\|^2 + \frac{1}{N}\sum_{i=1}^{N}\mathbb{E}\left\|\mathbf{x}_{i,t}^{s+1} - \bar{\mathbf{x}}_t^{s+1}\right\|^2\right]
\end{aligned} \tag{4.22}$$

We can upper bound the second term in the right hand side of (4.22) as follows.

$$\begin{aligned}
&\frac{1}{T}\sum_{s=0}^{S-1}\sum_{t=0}^{m-1}\frac{1}{N}\sum_{i=1}^{N}\mathbb{E}\left\|\mathbf{x}_{i,t}^{s+1} - \bar{\mathbf{x}}_t^{s+1}\right\|^2 \\
&\leq \frac{1}{T}\sum_{s=0}^{S-1}\sum_{t=0}^{m-1}\frac{1}{N}8\gamma^4 NL^2\left(1+\frac{1}{\delta}\right)\left(1+\frac{1}{\alpha}\right)\sum_{\ell=\tau(t)+1}^{t-1}(1+\alpha)^{t-1-\ell}\sum_{j=\tau(\ell)}^{\ell-1}(1+\theta)^{\ell-1-j}\mathbb{E}\left\|\bar{\mathbf{v}}_j^{s+1}\right\|^2 && (4.23) \\
&\leq \frac{256\gamma^4 L^4}{\delta^4}\frac{1}{T}\sum_{s=0}^{S-1}\sum_{t=0}^{m-1}\mathbb{E}\left\|\bar{\mathbf{v}}_t^{s+1}\right\|^2 && (4.24) \\
&\leq \frac{1}{2}\frac{1}{T}\sum_{s=0}^{S-1}\sum_{t=0}^{m-1}\mathbb{E}\left\|\bar{\mathbf{v}}_t^{s+1}\right\|^2 && (4.25)
\end{aligned}$$

where, (4.23) follows from the two inequalities in Lemma 4.2.3; (4.24) follows from Lemma A.3.3; (4.25) follows from (4.21). Replacing (4.25) in (4.22), we get

$$\begin{aligned}
&\min_{s,t}\left[\mathbb{E}\left\|\nabla f\left(\bar{\mathbf{x}}_t^{s+1}\right)\right\|^2 + \frac{1}{N}\sum_{i=1}^{N}\mathbb{E}\left\|\mathbf{x}_{i,t}^{s+1} - \bar{\mathbf{x}}_t^{s+1}\right\|^2\right] \\
&\leq \frac{1}{T}\sum_{s=0}^{S-1}\sum_{t=0}^{m-1}\left[\mathbb{E}\left\|\nabla f\left(\bar{\mathbf{x}}_t^{s+1}\right)\right\|^2 + \frac{1}{2}\mathbb{E}\left\|\bar{\mathbf{v}}_t^{s+1}\right\|^2\right] \\
&\leq \frac{1}{T}\sum_{s=0}^{S-1}\sum_{t=0}^{m-1}\Bigg[\mathbb{E}\left\|\nabla f\left(\bar{\mathbf{x}}_t^{s+1}\right)\right\|^2 \\
&\qquad + \left(1 - L\gamma - \left[\frac{128\gamma^4 L^4 m}{NB\delta^2} + \frac{4\gamma^2 L^2 m}{NB} + \frac{512\gamma^4 L^4}{\delta^4}\right]\right)\mathbb{E}\left\|\bar{\mathbf{v}}_t^{s+1}\right\|^2\Bigg] && (4.26) \\
&\leq \frac{2\left(f(\mathbf{x}^0) - f_*\right)}{T\gamma}. && (4.27)
\end{aligned}$$

Here, (4.26) follows from the choice of γ that satisfies (4.21), and (4.27) follows from (4.20).

Remark. It follows from (4.6) that for large enough T, Algorithm 5 returns an ϵ-FoS point $\bar{\mathbf{x}}_t^{s+1}$. Further, it follows from Theorem 4.2.1 that all the local iterates $\{\mathbf{x}_{i,t}^{s+1}\}_i$ are within $O(\sqrt{\epsilon})$-radius neighborhood of this point. So, we also achieve consensus across nodes.

In the next section, we solve the online variant of the problem (4.3). The algorithm proposed, and the accompanying convergence analysis builds upon the finite-sum approach.

4.3 Parallel Restarted SPIDER - Online Case

We consider a network of N worker nodes connected to a server node. The objective is to solve (4.3). Note that the distribution of samples at different nodes can potentially be different, i.e., $\mathcal{D}_i \neq \mathcal{D}_j$, for $i \neq j$.

4.3.1 Proposed Algorithm

The pseudo-code of the approach is given in Algorithm 6. In the following, we only highlight the steps which are different from Algorithm 5. The proposed algorithm is pretty much the same as Algorithm 5, except a few changes to account for the fact that for problem (4.3), exact gradients can never be computed. Hence full gradient computations are replaced by batch stochastic gradient computation, where the batches are of size n_b. Again, batch sizes across nodes can be non-uniform. However, we avoid doing so for the sake of simpler notations.

At the end of the inner loop ($t = m$), first the local iterates are averaged and returned to the workers (steps 15-16). Next, the workers compute batch stochastic gradients of their respective functions $\{\frac{1}{n_b}\sum_{\xi_i} \nabla f_i\left(\cdot; \xi_i\right)\}_{i=1}^N$ at the common point $\bar{\mathbf{x}}_m^{s+1}$, and send these to the server. The server averages these, and returns this average $\bar{\mathbf{v}}_m^{s+1}$ to the worker nodes (steps 17-18). As in Algorithm 5, all the worker nodes start the next epoch at the same point, and along the same descent direction. However, unlike Algorithm 5, this direction is not $\nabla f(\bar{\mathbf{x}}_m^{s+1})$.

Algorithm 6 PR-SPIDER - Online Case

```
Input: Initial iterate $\mathbf{x}^0_{i,m} = \mathbf{x}^0, \mathbf{v}^0_{i,m} = \frac{1}{N}\sum_{j=1}^{N} \frac{1}{n_b}\sum_{\xi_j} \nabla f_j\left(\mathbf{x}^0;\xi_j\right), \forall\, i \in [1:N]$
for $s = 0$ to $S-1$ do
    $\mathbf{x}^{s+1}_{i,0} = \mathbf{x}^{s}_{i,m}, \forall\, i \in [1:N]$
    $\mathbf{v}^{s+1}_{i,0} = \mathbf{v}^{s}_{i,m}, \forall\, i \in [1:N]$
    $\mathbf{x}^{s+1}_{i,1} = \mathbf{x}^{s+1}_{i,0} - \gamma \mathbf{v}^{s+1}_{i,0}$
    for $t = 1$ to $m-1$ do
        Compute $\mathbf{v}^{s+1}_{i,t}$, using (4.4) $\forall\, i \in [1:N]$
        if $t$ mod $I = 0$ then
            $\mathbf{x}^{s+1}_{i,t} = \bar{\mathbf{x}}^{s+1}_{t} \triangleq \frac{1}{N}\sum_{j=1}^{N} \mathbf{x}^{s+1}_{j,t}, \forall\, i \in [1:N]$
            $\mathbf{v}^{s+1}_{i,t} = \bar{\mathbf{v}}^{s+1}_{t} \triangleq \frac{1}{N}\sum_{j=1}^{N} \mathbf{v}^{s+1}_{j,t}, \forall\, i \in [1:N]$
        end if
        $\mathbf{x}^{s+1}_{i,t+1} = \mathbf{x}^{s+1}_{i,t} - \gamma \mathbf{v}^{s+1}_{i,t}, \forall\, i \in [1:N]$
    end for
    if $s < S-1$ then
        $\bar{\mathbf{x}}^{s+1}_{m} = \frac{1}{N}\sum_{j=1}^{N} \mathbf{x}^{s+1}_{j,m}$
        $\mathbf{x}^{s+1}_{i,m} = \bar{\mathbf{x}}^{s+1}_{m}, \forall\, i \in [1:N]$
        $\bar{\mathbf{v}}^{s+1}_{m} = \frac{1}{N}\sum_{i=1}^{N} \frac{1}{n_b}\sum_{\xi_i} \nabla f_i\left(\mathbf{x}^{s+1}_{i,m};\xi_i\right)$
        $\mathbf{v}^{s+1}_{i,m} = \bar{\mathbf{v}}^{s+1}_{m}, \forall\, i \in [1:N]$
    end if
end for
Return
```

Assumption 4. *Bounded Variance:* There exists constant σ such that

$$\mathbb{E}_{\xi}\left\|\nabla f_i\left(\mathbf{x};\xi\right) - \nabla f\left(\mathbf{x}\right)\right\|^2 \leq \sigma^2, \quad \forall\, i \in [1:N]. \tag{4.28}$$

See [41] for a discussion on the necessity of such bounded gradient dissimilarity assumption in settings where local data at different nodes follows heterogeneous distributions.

4.3.2 Convergence Result

Theorem 4.3.1. *For the online problem* (4.3) *with* N *nodes, under Assumptions 1-4, for small enough step size* $0 < \gamma < \frac{1}{8IL}$,

$$\min_{s,t}\left[\mathbb{E}\|\nabla f\left(\bar{\mathbf{x}}^{s+1}_{t}\right)\|^2 + \frac{L^2}{N}\sum_{i=1}^{N}\mathbb{E}\|\mathbf{x}^{s+1}_{i,t} - \bar{\mathbf{x}}^{s+1}_{t}\|^2\right] \leq \frac{2\left(f(\mathbf{x}^0) - f_*\right)}{T\gamma} + \frac{2\sigma^2}{Nn_b}, \tag{4.29}$$

where n_b is the large batch-size used for gradient estimators at the start of each epoch (line 17 in Algorithm 6).

Remark. Note that, in comparison to Theorem 4.2.1, in Theorem 4.3.1 we have an additional term $\frac{2\sigma^2}{Nn_b}$, which accounts for the variance of the gradients computed at the start of each epoch (line 17 in Algorithm 6).

Based on Theorem 4.3.1, we can bound the computation (IFO) and communication complexity of Algorithm 6.

Sample and Communication Complexity

Corollary 2. *To reach an ϵ-FoS point (Definition 4.1.1), the IFO complexity is bounded as*

$$O\left(\frac{\sigma}{\epsilon^{3/2}} + \frac{\sigma^2}{\epsilon}\right).$$

The communication complexity (the number of communication rounds) is bounded as

$$O\left(\frac{1}{\epsilon}\right).$$

PROOF: Given (4.29), to reach an ϵ-FoS point (Definition 4.1.1), a sufficient condition for the total number of iterations T and batch-size n_b (for gradient estimators at the start of each epoch - line 17 in Algorithm 6) is

$$\frac{2\left(f(\mathbf{x}^0) - f_*\right)}{T\gamma} = \frac{\epsilon}{2} \qquad \wedge \qquad \frac{2\sigma^2}{Nn_b} = \frac{\epsilon}{2}. \tag{4.30}$$

Consequently, T, n_b satisfy

$$T = \frac{2\left(f(\mathbf{x}^0) - f_*\right)}{\gamma\epsilon} = \frac{CI}{\epsilon} \qquad \text{and} \qquad n_b = \frac{4\sigma^2}{N\epsilon}, \tag{4.31}$$

for constants C, I. As in Section 4.2.2, choosing $m = I\sqrt{Nn_b}, B = \frac{1}{I}\sqrt{\frac{n_b}{N}}$, the IFO complexity is bounded as

$$
\begin{aligned}
&N \times \left(\left\lceil \frac{T}{m} \right\rceil \cdot n_b + T \cdot 2B \right) \leq N \times \left(\left(\frac{CI}{m\epsilon} + 1 \right) n_b + \frac{CI}{\epsilon} 2B \right) \\
&\leq N \times \left(\frac{CI}{\epsilon} \cdot \left(\frac{n_b}{m} + B \right) + n_b \right) \\
&= O\left(\frac{\sigma}{\epsilon^{3/2}} + \frac{\sigma^2}{\epsilon} \right).
\end{aligned}
$$

Since communication happens once every I iterations, the communication complexity is $\left\lceil \frac{T}{I} \right\rceil \leq 1 + \frac{C}{\epsilon} = O\left(\frac{1}{\epsilon}\right)$. ■

Remark. (Optimality and Linear Speedup). As shown in [9], $O(\epsilon^{-3/2})$ is the optimal sample complexity for online problems. As stated in Section 4.2, we achieve linear speedup in the number of nodes N.

Remark. It follows from (4.30) that for Algorithm 6 to reach an ϵ-FoS point, in addition to T being large (as in the finite sample case), we also need $n_b = O(\frac{1}{N\epsilon})$. In other words, since computing the full gradients is infeasible in online problems (4.3), the nodes instead need to compute large-batch gradients once every epoch.

4.3.3 Convergence Analysis

The convergence analysis for online problems (4.3) follows closely the analysis in Section 4.2. The difference arises because here, at the start of each epoch, we cannot compute the full gradient of the local functions. Instead we compute stochastic gradients using large batch-sizes n_b (see line 17 in Algorithm 6). Therefore, we need to bound the departure of the gradient estimator from the actual gradients at the start of an epoch. Incorporating this additional factor, the remaining analysis follows similarly as in Section 4.2.

Gradient Estimate Error at $t = 0$

Lemma 4.3.2. *For* $0 \leq s \leq S-1$*, the sequences of iterates* $\{\mathbf{x}_{i,t}^{s+1}\}_{i,t}$ *and* $\{\mathbf{v}_{i,t}^{s+1}\}_{i,t}$ *generated by Algorithm 5 satisfy*

$$E_0^{s+1} = \mathbb{E}\left\|\bar{\mathbf{v}}_0^{s+1} - \frac{1}{N}\sum_{i=1}^{N} \nabla f_i\left(\mathbf{x}_{i,0}^{s+1}\right)\right\|^2 \leq \frac{\sigma^2}{Nn_b}. \tag{4.32}$$

PROOF: We have from the definition of E_0^{s+1}

$$\begin{aligned}
E_0^{s+1} &= \mathbb{E}\left\|\bar{\mathbf{v}}_0^{s+1} - \frac{1}{N}\sum_{i=1}^{N} \nabla f_i\left(\mathbf{x}_{i,0}^{s+1}\right)\right\|^2 \\
&= \mathbb{E}\left\|\frac{1}{N}\sum_{i=1}^{N}\frac{1}{n_b}\sum_{\xi_i} \nabla f_i\left(\mathbf{x}_{i,0}^{s+1};\xi_i\right) - \frac{1}{N}\sum_{i=1}^{N} \nabla f_i\left(\mathbf{x}_{i,0}^{s+1}\right)\right\|^2 \quad \text{(steps 4, 17-18 in Algorithm 6)} \\
&= \frac{1}{N^2}\sum_{i=1}^{N}\mathbb{E}\left\|\frac{1}{n_b}\sum_{\xi_i}\left\{\nabla f_i\left(\mathbf{x}_{i,0}^{s+1};\xi_i\right) - \nabla f_i\left(\mathbf{x}_{i,0}^{s+1}\right)\right\}\right\|^2 \qquad (4.33) \\
&= \frac{1}{N^2}\sum_{i=1}^{N}\frac{1}{n_b^2}\mathbb{E}\sum_{\xi_i}\left\|\nabla f_i\left(\mathbf{x}_{i,0}^{s+1};\xi_i\right) - \nabla f_i\left(\mathbf{x}_{i,0}^{s+1}\right)\right\|^2 \qquad (4.34) \\
&\leq \frac{\sigma^2}{Nn_b}. \qquad (4.35)
\end{aligned}$$

(4.33) follows since for $i \neq j$

$$\begin{aligned}
&\mathbb{E}\left\langle \sum_{\xi_i}\left\{\nabla f_i\left(\mathbf{x}_{i,0}^{s+1};\xi_i\right) - \nabla f_i\left(\mathbf{x}_{i,0}^{s+1}\right)\right\}, \sum_{\xi_j}\left\{\nabla f_j\left(\mathbf{x}_{j,0}^{s+1};\xi_j\right) - \nabla f_j\left(\mathbf{x}_{j,0}^{s+1}\right)\right\}\right\rangle \\
&\quad = \mathbb{E}\left\langle \mathbb{E}\sum_{\xi_i}\left\{\nabla f_i\left(\mathbf{x}_{i,0}^{s+1};\xi_i\right) - \nabla f_i\left(\mathbf{x}_{i,0}^{s+1}\right)\right\}, \mathbb{E}\sum_{\xi_j}\left\{\nabla f_j\left(\mathbf{x}_{j,0}^{s+1};\xi_j\right) - \nabla f_j\left(\mathbf{x}_{j,0}^{s+1}\right)\right\}\right\rangle = 0.
\end{aligned}$$

This is because, given $\mathbf{x}_{i,0}^{s+1} = \bar{\mathbf{x}}_m^s$, the samples at each node are picked uniformly randomly, and independent of other nodes. (4.34) follows since samples at any node are also picked independent of each other. Therefore, for any two distinct samples $\xi_i \neq \zeta_i$,

$$\mathbb{E}\left\langle \nabla f_i\left(\mathbf{x}_{i,0}^{s+1};\xi_i\right) - \nabla f_i\left(\mathbf{x}_{i,0}^{s+1}\right), \nabla f_i\left(\mathbf{x}_{i,0}^{s+1};\zeta_i\right) - \nabla f_i\left(\mathbf{x}_{i,0}^{s+1}\right)\right\rangle$$

$$= \mathbb{E}\left\langle \mathbb{E}_{\xi_i}\nabla f_i\left(\mathbf{x}_{i,0}^{s+1};\xi_i\right) - \nabla f_i\left(\mathbf{x}_{i,0}^{s+1}\right), \mathbb{E}_{\zeta_i}\nabla f_i\left(\mathbf{x}_{i,0}^{s+1};\zeta_i\right) - \nabla f_i\left(\mathbf{x}_{i,0}^{s+1}\right)\right\rangle = 0.$$

Finally, (4.35) follows from Assumption 4. ■

Remark. Recall that we mentioned E_0^{s+1} in Lemma 4.2.2, even though it was zero for finite-sum problems. We can utilize the bound in Lemma 4.2.2, incorporating E_0^{s+1} from Lemma 4.3.2. Lemma 4.2.3 holds true as it is, since it depends only on periodic averaging of iterates $\{\mathbf{x}_{i,t}^{s+1}\}_{i,t}$ and gradient estimators $\{\mathbf{v}_{i,t}^{s+1}\}_{i,t}$ every I iterations.

Next we state the descent lemma for the online case.

Descent over one epoch

Using Lemma 4.2.2, Lemma 4.2.3 and Lemma 4.3.2 we can bound the expected decay in function value over one epoch.

Lemma 4.3.3. *In each epoch $s \in [1, S]$,*

$$\begin{aligned}\mathbb{E}f\left(\bar{\mathbf{x}}_m^{s+1}\right) \leq \mathbb{E}f\left(\bar{\mathbf{x}}_0^{s+1}\right) - \frac{\gamma}{2}\sum_{t=0}^{m-1}\mathbb{E}\left\|\nabla f\left(\bar{\mathbf{x}}_t^{s+1}\right)\right\|^2 - \frac{\gamma}{2}(1-L\gamma)\sum_{t=0}^{m-1}\mathbb{E}\left\|\bar{\mathbf{v}}_t^{s+1}\right\|^2 \\ + \sum_{t=0}^{m-1}\mathbb{E}\left\|\bar{\mathbf{v}}_t^{s+1}\right\|^2\left[\frac{64\gamma^5L^4m}{NB\delta^2} + \frac{2\gamma^3L^2m}{NB} + \frac{256\gamma^5L^4}{\delta^4}\right] + \frac{\gamma\sigma^2m}{Nn_b}.\end{aligned} \tag{4.36}$$

PROOF: Substituting (4.35) in (4.17) and summing (4.17) over $t = 0, \ldots, m-1$, we get

$$\begin{aligned}\mathbb{E}f\left(\bar{\mathbf{x}}_m^{s+1}\right) \leq{}& \mathbb{E}f\left(\bar{\mathbf{x}}_0^{s+1}\right) - \frac{\gamma}{2}\sum_{t=0}^{m-1}\mathbb{E}\left\|\nabla f\left(\bar{\mathbf{x}}_t^{s+1}\right)\right\|^2 - \frac{\gamma}{2}(1-L\gamma)\sum_{t=0}^{m-1}\mathbb{E}\left\|\bar{\mathbf{v}}_t^{s+1}\right\|^2 \\ &+ \frac{16\gamma^5L^4}{NB}\left(1+\frac{1}{\delta}\right)\sum_{t=0}^{m-1}\sum_{\ell=0}^{t-1}\sum_{j=\tau(\ell)}^{\ell-1}(1+\theta)^{\ell-1-j}\mathbb{E}\left\|\bar{\mathbf{v}}_j^{s+1}\right\|^2 + \frac{2\gamma^3L^2}{NB}\sum_{t=0}^{m-1}\sum_{\ell=0}^{t-1}\mathbb{E}\left\|\bar{\mathbf{v}}_\ell^{s+1}\right\|^2 \\ &+ 8\gamma^5L^4\left(1+\frac{1}{\alpha}\right)\left(1+\frac{1}{\delta}\right)\sum_{t=0}^{m-1}\sum_{\ell=\tau(t)+1}^{t-1}\sum_{j=\tau(\ell)}^{\ell-1}(1+\alpha)^{t-1-\ell}(1+\theta)^{\ell-1-j}\mathbb{E}\left\|\bar{\mathbf{v}}_j^{s+1}\right\|^2 \\ &+ \frac{\gamma\sigma^2m}{Nn_b}\end{aligned} \tag{4.37}$$

(4.37) is the same as (4.18) except the additional last term in (4.37). Therefore, again using the upper bounds derived in Lemmas A.3.1, A.3.2 and A.3.3 in (4.37), we get (4.36). ■

Note that $\frac{\gamma\sigma^2 m}{N n_b}$ is the only extra term compared to Lemma 4.2.4. Therefore, given Lemma 4.3.3, the proof of Theorem 4.3.1 is a straightforward extension of the proof in the finite-sum case.

Proof of Theorem 4.3.1

Rearranging the terms in (4.36), and summing over epoch index s, analogous to (4.20) we get

$$\begin{aligned}
&\frac{1}{T}\sum_{s=0}^{S-1}\sum_{t=0}^{m-1}\left[\mathbb{E}\big\|\nabla f\left(\bar{\mathbf{x}}_t^{s+1}\right)\big\|^2 + \left(1 - L\gamma - \left[\frac{128\gamma^4 L^4 m}{NB\delta^2} + \frac{4\gamma^2 L^2 m}{NB} + \frac{512\gamma^4 L^4}{\delta^4}\right]\right)\mathbb{E}\big\|\bar{\mathbf{v}}_t^{s+1}\big\|^2\right] \\
&\le \frac{2}{T\gamma}\sum_{s=0}^{S-1}\left(\mathbb{E}f\left(\bar{\mathbf{x}}_0^{s+1}\right) - \mathbb{E}f\left(\bar{\mathbf{x}}_m^{s+1}\right)\right) + \frac{2}{T\gamma}\sum_{s=0}^{S-1}\frac{\gamma\sigma^2 m}{N n_b} \\
&= \frac{2}{T\gamma}\left(\mathbb{E}f\left(\bar{\mathbf{x}}^0\right) - \mathbb{E}f\left(\bar{\mathbf{x}}_m^{S+1}\right)\right) + \frac{2\sigma^2}{N n_b} \\
&\le \frac{2\left(f(\mathbf{x}^0) - f_*\right)}{T\gamma} + \frac{2\sigma^2}{N n_b}.
\end{aligned} \tag{4.38}$$

As in the finite-sum case, for small enough step size γ, (4.21) holds. Again, in Appendix A.3.4, we see a possible choice of γ. Consequently, we have

$$\begin{aligned}
&\min_{s,t}\left[\mathbb{E}\left\|\nabla f\left(\bar{\mathbf{x}}_t^{s+1}\right)\right\|^2 + \frac{L^2}{N}\sum\nolimits_{i=1}^{N}\mathbb{E}\left\|\mathbf{x}_{i,t}^{s+1} - \bar{\mathbf{x}}_t^{s+1}\right\|^2\right] \\
&\le \frac{1}{T}\sum_{s=0}^{S-1}\sum_{t=0}^{m-1}\left[\mathbb{E}\left\|\nabla f\left(\bar{\mathbf{x}}_t^{s+1}\right)\right\|^2 + \frac{1}{2}\mathbb{E}\|\bar{\mathbf{v}}_t^{s+1}\|^2\right] && (4.39) \\
&\le \frac{2\left(f(\mathbf{x}^0) - f_*\right)}{T\gamma} + \frac{2\sigma^2}{N n_b}, && (4.40)
\end{aligned}$$

where, (4.39) follows from (4.25). (4.40) follows from the choice of γ in (4.21) and from (4.38).

Remark. It follows from (4.30) that for large enough T and n_b, Algorithm 6 returns an ϵ-FoS point $\bar{\mathbf{x}}_t^{s+1}$. Further, it follows from (4.40) that all the local iterates $\{\mathbf{x}_{i,t}^{s+1}\}_i$ are within $O(\sqrt{\epsilon})$-radius neighborhood of this point. So, we also achieve consensus across nodes.

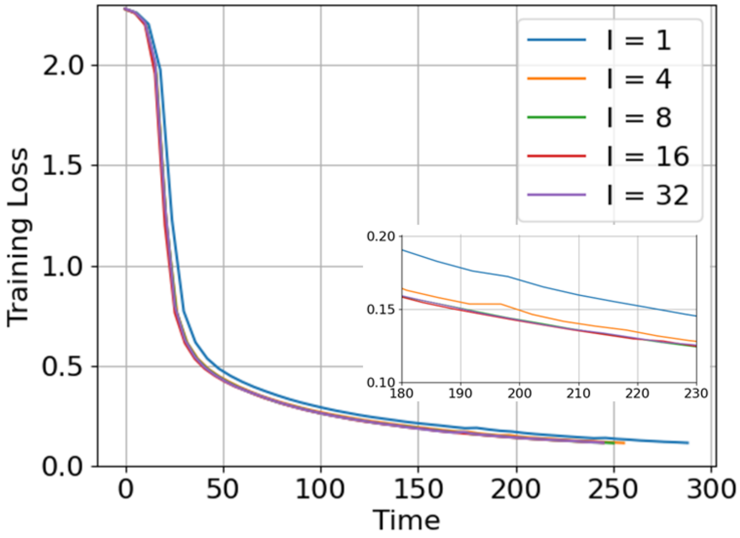
4.4 Simulation Results

In this section, we describe the experiments we conducted to supplement our theoretical results. We conducted image classification tasks on the MNIST [104] and CIFAR10 [100] datasets. For MNIST, we trained a shallow convolutional neural network (CNN), while for CIFAR10, we trained the Resnet20 architecture [78]. Our experiments were conducted on a cluster of 8 NVIDIA Quadro RTX 5000 GPUs. The algorithms are implemented in PyTorch 1.0. We use Open MPI library to communicate between GPU nodes.

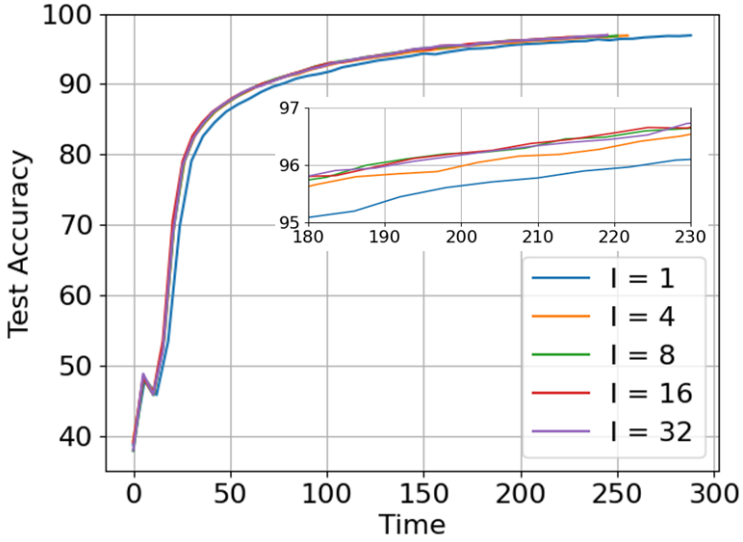
Data partitioning: The complete dataset is partitioned into 8 *disjoint* sets, locally available to the 8 GPU nodes. This is in contrast to the standard implementations [203, 201] where each node is assumed to have access to the entire dataset. In our implementation, the each node never gets to access the samples present at the other nodes. This makes the training process more challenging, even more so when the communication is infrequent.

For the MNIST dataset, we trained a CNN using local batch-size of 50, with learning rate of 0.005. To compare the impact of infrequent communication ($I > 1$) between the workers and server, we compare the performance for $I \in \{1, 4, 8, 16, 32\}$. Note that for $I = 1$, the nodes communicate with the server at each iteration. This is equivalent to running the SPIDER algorithm [52], with its optimal convergence guarantees. We compare the training loss (Figure 4.1a) and the test accuracy (Figure 4.1b) for different values of I. To make the benefits of infrequent communication more explicit, we have plotted wall clock time (in seconds) of the algorithm on the x-axis. Each epoch of the algorithm entails going through the data twice, once in the inner loop, and second while computing the full gradient. Higher values of I result in roughly 15% faster run time, without much loss in performance. The inset plots in both figures show the speedup in the algorithm, as I increases, for achieving the same test accuracy of loss function value.

For the CIFAR-10 dataset, we used local batch-size of 250, with learning rate of 0.04. Since each node has 6250 samples, each epoch has $m = 25$ inner loop steps. To compare the impact of infrequent communication ($I > 1$) between the workers and server, we compare the performance for $I \in \{1, 4, 8, 16, > m\}$. $I > m$ means communication happens between the workers and server



(a) Training loss vs wall clock time

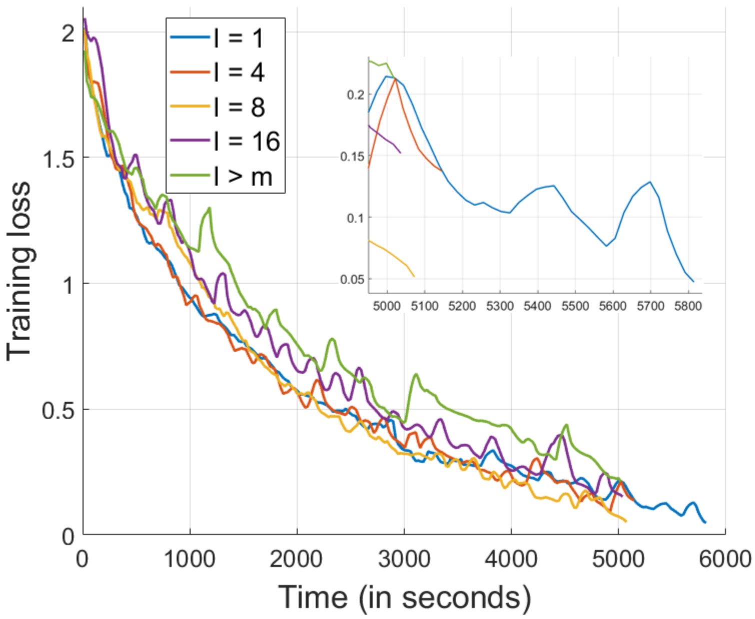


(b) Test accuracy vs wall clock time

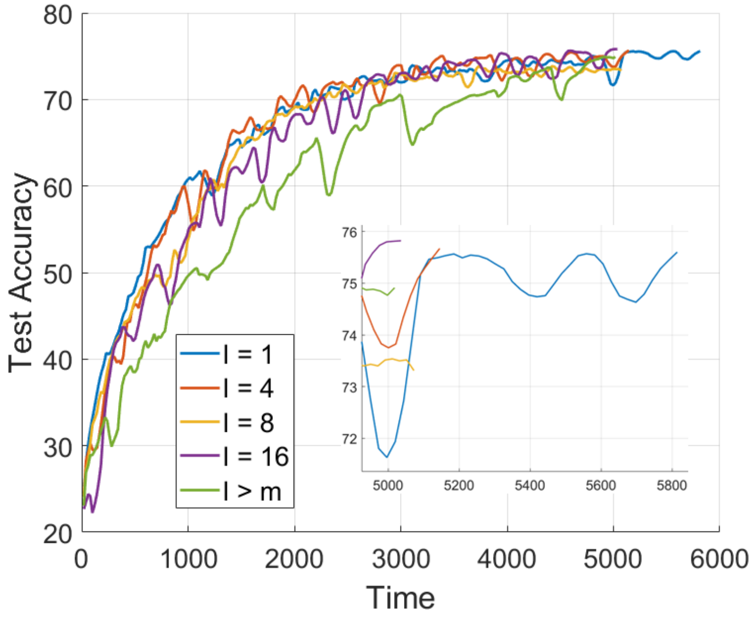
Fig. 4.1: Convergence of PR-SPIDER for over MNIST dataset.

only at the end of an epoch, when full gradients are computed and aggregated. We compare the training loss (Figure 4.2a) and the test accuracy (Figure 4.2b) for different values of I. Again, we have plotted wall clock time of the algorithm on the x-axis. Higher values of I result in roughly $12 - 15\%$ faster run time, without much loss in performance. The inset plots in both figures show the speedup in the algorithm, as I increases, for achieving similar test accuracy or loss function values.

Note that in both the figures, the gain (in terms of wall clock time) from $I = 1$ to $I = 4$



(a) Training loss vs wall clock time



(b) Test accuracy vs wall clock time

Fig. 4.2: Convergence of PR-SPIDER for ResNet20 over CIFAR-10 dataset.

is much more significant than the gain achieved by further increasing I ($I = 8, 16$, etc.). This is because of the following reason. Consider Figure 4.2. With 270k parameters, ResNet20 is a relatively small CNN. This enables carrying out experiments with it on moderate-sized GPUs, like the one we used. However, for this reason, between communication between nodes and local computation at nodes, the latter is relatively more time consuming. Hence, even though we save resources by communicating less ($I = 8$ or higher), the gain is less pronounced. On the other hand, larger networks, for example, ResNet18 [78] (with over 11 million parameters) require more communication resources (but are more difficult to tune with moderate GPUs). In our experiments with ResNet18, using higher values of I, we observed as much as 50% gain in wall clock time (relative to $I = 1$).

4.5 Drawbacks of PR-SPIDER and A Novel Algorithm to Fix Them

As shown in Sections 4.2 and 4.3, the PR-SPIDER algorithm achieves the optimal computation complexity, along with state-of-the-art communication cost. However, since PR-SPIDER is based on the variance-reduced SPIDER algorithm [52, 187], along with the optimal convergence rate, it also inherits the drawbacks of the former. In the following discussion, we discuss these shortcomings of PR-SPIDER. We then propose a novel algorithm to address them.

4.5.1 Drawbacks of PR-SPIDER

As illustrated in Section 4.2.2 for the finite-sum case, all the nodes need to compute the complete gradient of their local objective functions $\{\nabla f_i(\cdot)\}_i$ at the start of each outer loop. This incurs a computation cost of $O(n)$. In the online case (Section 4.3.2), the exact gradient computation is replaced by the computation of stochastic gradient over a large sample of size $n_b = O(\epsilon^{-1})$. These large-batch gradient computations pose significant challenge for distributed computation, since in presence of heterogeneous nodes, with unequal sample-sizes and/or disparate computational capabilities, the faster nodes have to wait for the slower nodes to finish their computations, before the algorithm can proceed. Also, the double-loop structure of the variance-reduced algorithms like SVRG [146] and SPIDER poses practical challenges. See [138, 38] for more discussion on the drawbacks of double-loop variance-reduced methods.

4.5.2 STEM Algorithm

To counter the drawbacks of the double-loop variance reduction methods discussed above, variants of the momentum-based SGD methods have been proposed over the years. This line of work was pioneered by SARAH [138]. The proposed estimator required the computation of large-batch gradients at the nodes only at the beginning of the algorithm. For all the subsequent iterations, only computation of stochastic gradients is required. This requirement for massive gradient computation at the outset was alleviated by STORM [38]. In our subsequent work, we propose a distributed variant of the STORM algorithm, which we call Stochastic Two-sided Momentum (STEM).

Notation: Since STEM is a single-loop algorithm contrary to PR-SPIDER, we modify the notation a bit: the iterate at node i at time t is denoted as $\mathbf{x}_t^i$. Similarly, the descent direction at node i at time t is denoted as $\mathbf{v}_t^i$.

Let us discuss the key steps of STEM, listed in Algorithm 7. In Step 10, each node locally updates its model parameters using the local direction $\mathbf{v}_t^i$, computed by using b stochastic gradients at two consecutive iterates $\mathbf{x}_t^i$ and $\mathbf{x}_{t+1}^i$. After every I local steps, the WNs share their current local models $\{\mathbf{x}_{t+1}^i\}_{i=1}^N$ and directions $\{\mathbf{v}_{t+1}^i\}_{i=1}^N$ with the SN. The SN aggregates these quantities, and performs a server-side momentum step, before returning $\bar{\mathbf{x}}_{t+1}$ and $\bar{\mathbf{v}}_{t+1}$ to all the WNs. Because both the WNs and the SN perform momentum based updates, we call the algorithm a stochastic *two-sided* momentum algorithm. The key parameters are: b the minibatch size, I the local update steps between two communication rounds, $\{\gamma_t\}$ the stepsizes, and $\{a_t\}$ the momentum parameters.

We show that in the FL setting, the local directions together with the local models have to be aggregated by the SN so to avoid being influenced too much by the local data. More importantly, besides the WNs, the SN also needs to perform updates using the (aggregated) momentum directions. Finally, such *two-sided* momentum updates have to be done carefully with the correct choice of minibatch size b, and the local update frequency I. Overall, it is the judicious choice of all these design elements that results in the optimal sample and communication complexities. Next, we present the convergence guarantees of the STEM algorithm.

Algorithm 7 The Stochastic Two-Sided Momemtum (STEM) Algorithm

```
Input: Parameters: $c > 0$, the number of local updates $I$, batch size $b$, stepsizes $\{\gamma_t\}$.
Initialize: Iterate $\mathbf{x}_1^i = \bar{\mathbf{x}}_1 = \frac{1}{K}\sum_{i=1}^N \mathbf{x}_1^i$, descent direction $\mathbf{v}_1^i = \bar{\mathbf{v}}_1 = \frac{1}{K}\sum_{i=1}^N \mathbf{v}_1^i$ with
    $\mathbf{v}_1^i = \frac{1}{B}\sum_{\xi_1^i \in \mathcal{B}_1^i} \nabla f^i(\mathbf{x}_1^i; \xi_1^i)$ and $|\mathcal{B}_1^i| = B$ for $k \in [N]$.
Perform: $\mathbf{x}_2^i = \mathbf{x}_1^i + \gamma_1 \mathbf{v}_1^i, \ \forall\, k \in [N]$
for $t = 1$ to $T$ do
    for $i = 1$ to $N$ do                    # at the WN
        $\mathbf{v}_{t+1}^i = \frac{1}{b}\sum_{\xi_{t+1}^i \in \mathcal{B}_{t+1}^i} \nabla f^i(\mathbf{x}_{t+1}^i; \xi_{t+1}^i) + (1 - a_{t+1})\Big(d_t^i - \frac{1}{b}\sum_{\xi_{t+1}^i \in \mathcal{B}_{t+1}^i} \nabla f^i(\mathbf{x}_{t+1}^i; \xi_{t+1}^i)\Big)$
    where we choose $|\mathcal{B}_{t+1}^i| = b$, and $a_{t+1} = c \cdot \gamma_t^2$;
        if $t$ mod $I = 0$ then                         # at the server
            $\mathbf{v}_{t+1}^i = \bar{\mathbf{v}}_{t+1} := \frac{1}{N}\sum_{i=1}^N \mathbf{v}_{t+1}^i$
            $\mathbf{x}_{t+1}^i = \bar{\mathbf{x}}_{t+1} := \frac{1}{N}\sum_{i=1}^N \mathbf{x}_{t+1}^i - \gamma_{t+1}\bar{\mathbf{v}}_{t+1}$      # server-side momentum step
        else $\mathbf{x}_{t+2}^i = \mathbf{x}_{t+1}^i - \gamma_{t+1}\mathbf{v}_{t+1}^i$                  # worker-side momentum step
        end if
    end for
end for
Return: $\bar{\mathbf{x}}_a$ chosen uniformly randomly from $\{\bar{\mathbf{x}}_t\}_{t=1}^T$
```

4.5.3 Main results: convergence guarantees for STEM

In this section, we analyze the performance of STEM. We first present our main result, and then provide discussions about a few parameter choices. In the next subsection, we discuss a special case of STEM related to the classical FedAvg and minibatch SGD algorithms.

Theorem 4.5.1. *Under the Assumptions 1-4, suppose the stepsize sequence is chosen as:*

$$\gamma_t = \frac{\bar{\kappa}}{(w_t + \sigma^2 t)^{1/3}}, \text{ where } \bar{\kappa} = \frac{(bK\sigma)^{2/3}}{L}, \quad w_t = \max\left\{2\sigma^2, 4096L^3I^3\bar{\kappa}^3 - \sigma^2 t, \frac{c^3\bar{\kappa}^3}{4096L^3I^3}\right\}. \tag{4.41}$$

Further, let us set $c = \frac{64L^2}{bK} + \frac{\sigma^2}{24\bar{\kappa}^3 LI} = L^2\left(\frac{64}{bK} + \frac{1}{24(bK)^2 I}\right)$*, and set the initial batch size as* $B = bI$*; set the local updates* I *and minibatch size* b *as follows:*

$$I = \mathcal{O}\big((T/K^2)^{\nu/3}\big), \quad b = \mathcal{O}\big((T/K^2)^{1/2-\nu/2}\big) \tag{4.42}$$

where ν satisfies $\nu \in [0,1]$. Then for $\bar{\mathbf{x}}_a$ chosen according to Algorithm 7, we have:

$$\mathbb{E}\|\nabla f(\bar{\mathbf{x}}_a)\|^2 = \mathcal{O}\bigg(\frac{f(\bar{\mathbf{x}}_1) - f^*}{N^{2\nu/3}T^{1-\nu/3}}\bigg) + \tilde{\mathcal{O}}\bigg(\frac{\sigma^2}{N^{2\nu/3}T^{1-\nu/3}}\bigg). \tag{4.43}$$

Corollary 3. *For any $\nu \in [0,1]$, we have*

Sample Complexity: *The sample complexity of STEM is $\tilde{\mathcal{O}}(\epsilon^{-3/2})$. This implies that each WN requires at most $\tilde{\mathcal{O}}(K^{-1}\epsilon^{-3/2})$ gradient computations, thereby achieving linear speedup with the number of WNs present in the network.*

Communication Complexity: *The communication complexity of STEM is $\tilde{\mathcal{O}}(\epsilon^{-1})$.*

We refer the reader to [95] for the proof of the theoretical results. However, a few remarks are in order.

Remark (Near-Optimal sample and communication complexities). Theorem 4.5.1 suggests that when I and b are selected appropriately, then STEM achieves $\tilde{\mathcal{O}}(\epsilon^{-3/2})$ and $\tilde{\mathcal{O}}(\epsilon^{-1})$ sample and communication complexities. Taking them separately, these complexity bounds are the best achievable by the existing FL algorithms (upto logarithmic factors and regardless of the sample or batch Lipschitz smooth assumption); see Table 4.2. We note that the $\mathcal{O}(\epsilon^{-3/2})$ complexity is the best possible that can be achieved by centralized SGD with the sample Lipschitz gradient assumption [52]. On the other hand, the $\mathcal{O}(\epsilon^{-1})$ complexity bound is also *likely* to be the optimal, since in [206] the authors showed that even when the local steps use a class of (deterministic) first-order algorithms, $\mathcal{O}(\epsilon^{-1})$ is the best achievable communication complexity.

Remark (The Optimal Batch Sizes and Local Updates Trade-off). The parameter $\nu \in [0,1]$ is used to balance the local minibatch sizes b, and the number of local updates I. Eqs. in (4.42) suggest that when ν increases from 0 to 1, b decreases and I increases. Specifically, if $\nu = 1$, then b is a constant but $I = \mathcal{O}(T^{1/3}/K^{2/3})$. In this case, each WN chooses a small minibatch while executing multiple local updates, and STEM resembles a FedAvg [123] (a.k.a. Local SGD) algorithm but with double-sided momentum update directions. In contrast, if $\nu = 0$, then $b = \mathcal{O}(T^{1/2}/K)$ but I is a constant. In this case, each WN chooses a large batch size while executing only a few, or even one, local updates, and STEM resembles the Minibatch SGD, but again with different update directions, and is referred to as Minibatch STEM.

4.5.4 Simulation Results for STEM

In this section, we validate the proposed STEM algorithm and compare its performance with the de facto standard FL algorithm FedAvg [123], and the recently proposed SCAFFOLD [94]. The goal of our experiments are three-fold: (1) To show that STEM performs on par, if not better, compared to other algorithms in both moderate and high heterogeneity settings, (2) there are multiple ways to reach the desired solution accuracy, one can either choose a large batch size and perform only a few local updates or select a smaller batch size and perform multiple local updates, and finally, (3) if the local updates and the batch sizes are not chosen appropriately, the WNs might need to perform excessive computations to achieve the desired solution accuracy, thereby slowing down convergence.

Data and Parameter Settings: We compare the algorithms for image classification tasks on CIFAR-10 dataset with 100 WNs in the network. Each WN implements a two-hidden-layer convolutional neural network (CNN) architecture followed by three linear layers. All the experiments are implemented on a single NVIDIA Quadro RTX 5000 GPU. We consider two settings, one with moderate heterogeneity and the other with high heterogeneity. For both settings, the data is partitioned into disjoint sets among the WNs. In the moderate heterogeneity setting, the WNs have access to partitioned data from all the classes but for the high heterogeneity setting the data is partitioned such that each WN can access data from only a subset (5 out of 10 classes) of classes. Each WN has access to 490 samples for training and 90 samples for testing purposes.

For STEM, we set $w_t = 1$, $c = \bar{c}/\bar{\kappa}^2$ and tune for $\bar{\kappa}$ and $\bar{c}$ in the range $\bar{\kappa} \in [0.01, 0.5]$ and $\bar{c} \in [1, 10]$, respectively. We note that for small batch sizes $\bar{\kappa} \in [0.01, 0.1]$, whereas for larger batch sizes $\bar{\kappa} \in [0.3, 0.5]$ perform well. We diminish γ_t according to in each epoch[5]. For SCAFFOLD and FedAvg, the stepsize choices of 0.1 and 0.01 perform well for large and smaller batch sizes,

---

[5] We define epoch as a single pass over the whole data.

respectively. We use cross entropy as the loss function and evaluate the algorithm performance under a few settings discussed next.

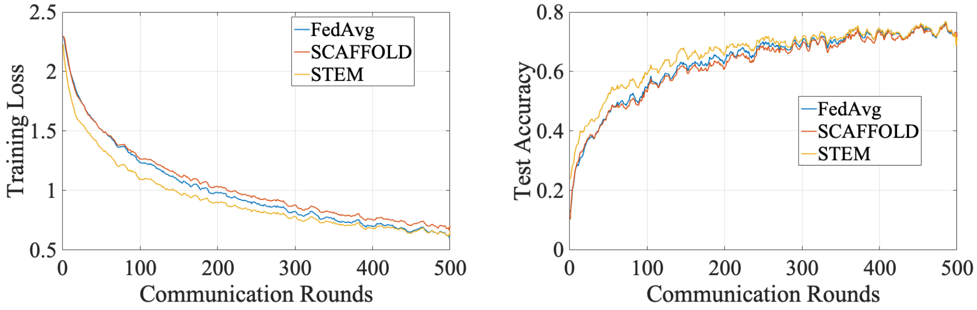


Fig. 4.3: Training loss and testing accuracy in the moderate heterogeneity setting. $b = 8$ and $I = 61$.

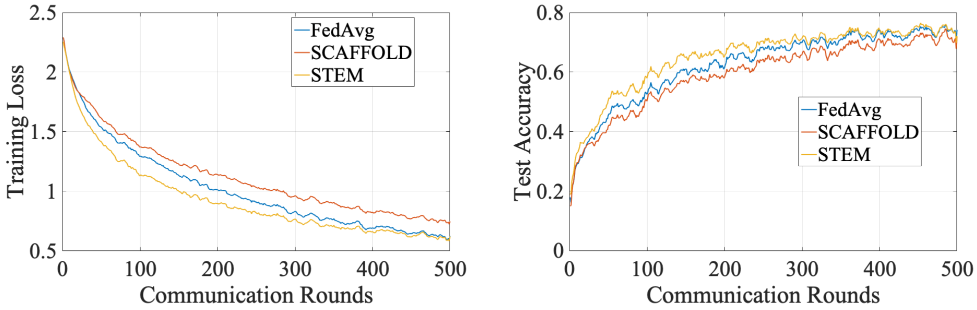


Fig. 4.4: Training loss and testing accuracy in the moderate heterogeneity setting. $b = 64$ and $I = 7$.

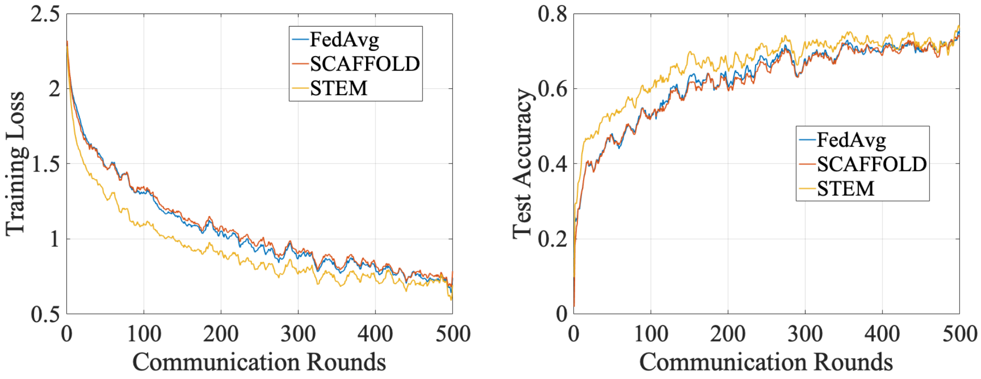


Fig. 4.5: Training loss and testing accuracy in the high heterogeneity setting. $b = 8$ and $I = 61$.

Discussion: In Figures 4.3 and 4.4, we compare the training and testing performance of STEM with FedAvg and SCAFFOLD for CIFAR-10 dataset under moderate heterogeneity setting. For Figure 4.3, we choose $b = 8$ and $I = 61$, whereas for Figure 4.4, we choose $b = 64$ and $I = 7$.

We first note that for both cases STEM performs better than FedAvg and SCAFFOLD. Moreover, observe that for both settings, small batches with multiple local updates (Figure 4.3) and large batches with few local updates (Figure 4.4), the algorithms converge with approximately similar performance, corroborating the theoretical analysis. Next, in Figure 4.5 we evaluate the performance of the proposed algorithms on CIFAR-10 with high heterogeneity setting for $b = 8$ and $I = 61$. We note that STEM outperforms FedAvg and SCAFFOLD in this setting as well.

4.6 Summary

In this chapter, we proposed a distributed variance-reduced algorithm, PR-SPIDER, for stochastic non-convex optimization. Our algorithm is a non-trivial extension of SPIDER [52], the single-node stochastic optimization algorithm, and parallel-restarted SGD [203, 200]. We proved convergence of our algorithm to a first-order stationary solution. The proposed approach achieves the best known communication complexity $O(\epsilon^{-1})$. In terms of IFO complexity, we have achieved the optimal rate, significantly improving the state-of-the-art, while maintaining the linear speedup achieved by existing methods. For finite-sum problems, we achieved the optimal $O(\frac{\sqrt{Nn}}{\epsilon})$ overall IFO complexity. On the other hand, for online problems, we achieved the optimal $O\left(\frac{\sigma}{\epsilon^{3/2}} + \frac{\sigma^2}{\epsilon}\right)$, a massive improvement over the existing $O(\frac{\sigma^2}{\epsilon^2})$. In addition, unlike many existing approaches, our algorithm is general enough to allow non-identical distributions of data across workers. Finally, we discussed some drawbacks of PR-SPIDER algorithm, and proposed another algorithm, STEM, which addresses these, while achieving similar convergence guarantees.

CHAPTER 5

STOCHASTIC ZEROTH-ORDER OPTIMIZATION: HYBRID GRADIENT DESCENT

In the previous chapter, we assumed access to the IFO oracle (Definition 4.1.2), which returns the loss function value and the gradient vector at the specified point. In this chapter, we solve the stochastic nonconvex optimization problem, under more general conditions, where the analytical expressions of the objective functions are either expensive or infeasible to obtain. Hence, we only have access to function values.

5.1 Introduction

Zeroth-order (ZO) methods can be seen as gradient-less versions of first-order (gradient-based) optimization methods [152, 136, 61]. ZO optimization involves approximating the full/stochastic gradient of the function using only the function values, and using this gradient estimator in the first-order (FO) optimization framework. Owing to their theoretical *closeness* to FO methods, ZO methods often have convergence rates comparable to their FO counterparts, with an additional small-degree polynomial in the problem dimension d [50, 137].

5.1.1 Motivating Application: Adversarial Example Generation

ZO optimization has recently been shown to be powerful in evaluating the adversarial robustness of deep neural networks (DNNs), by generating black-box adversarial examples, e.g., crafted images with imperceptible perturbations, to deceive a well-trained DNN using only input-output model queries [28, 80, 81, 34, 120, 177]. The internal configurations of the victim DNN systems are not revealed to the attackers and the only mode of interaction with the systems is by submitting inputs and receiving the predicted outputs.

5.1.2 Related Work

The general ZO approach involves computing a gradient estimate $\hat{\nabla} f(\mathbf{x})$ of the loss function f at the point $\mathbf{x}$, and then plugging this estimate into a FO method. One way to estimate the gradient is by querying the function at a single randomly chosen point in the vicinity of $\mathbf{x}$ [55]. More effectively, multi-point (e.g., two-point) approaches are used [3, 137], leading to better variance and improved complexity results.

The initial work on multi-point estimators was largely limited to convex problems. For smooth, deterministic problems, the authors in [137] proposed the ZO gradient descent (ZO-GD) algorithm and proved $O(d/T)$ convergence rate, where d denotes the problem size and T is the number of iterations. A ZO-mirror descent algorithm [50] extended this to the stochastic case, achieving the rate of $O(\sqrt{d}/\sqrt{T})$. The authors also proved the result to be *order-optimal*. For nonsmooth problems, the authors in [158] proved the same rate to be optimal, while also extending the analysis to non-Euclidean problems.

In the nonconvex domain, the first stochastic algorithm, ZO-SGD [61] utilized vectors sampled from normal distribution, and achieved $O(\sqrt{d}/\sqrt{T})$ convergence rate. The same rate was achieved by [60], while using random vectors sampled from the surface of the unit sphere. In [113], an asynchronous ZO-SCD approach was proposed for parallel architecture settings, which achieved $O(\sqrt{d}/\sqrt{T})$ rate. Following the recent progress in variance reduction methods for first-order optimization: SAGA [43], SVRG [91, 147], SARAH [139], SPIDER [53], to name a few, the ZO extensions of variance reduced methods have also appeared in recent years. ZO-SVRG [119] improved the iteration complexity to $O(d/T)$, but at the expense of an increased function query complexity. The iteration complexity is further improved in [53, 86]. ZO-counterparts of FO methods have also been proposed in other contexts, such as constrained optimization [10], adaptive momentum methods [33], mitigation of extreme components of gradient noise [117], and distributed optimization over networks [72, 173].

ZO algorithms often suffer from the high variance of gradient estimates. The existing estimators involve choosing between saving on the function query cost [61], and achieving higher accuracy of the gradient estimates [113].

5.1.3 Our Contributions

We summarize our contributions below:

- We propose a novel function value based gradient estimator (we call HGE, hybrid gradient estimator), which takes advantage of both the query-efficient random gradient estimate and the variance-reduced coordinate-wise gradient estimate. We also develop a coordinate importance sampling method to further improve the variance of HGE under a fixed number of function queries.

- We propose a ZO hybrid gradient descent (ZO-HGD) optimization method with the aid of HGE. We show that ZO-HGD is general since it covers ZO stochastic gradient descent (ZO-SGD) [61] and ZO stochastic coordinate descent (ZO-SCD) [113] as special cases. We provide a comprehensive theoretical analysis for the convergence of ZO-HGD across different optimization domains, showing that ZO-HGD is efficient in both iteration and function query complexities.

- We demonstrate the effectiveness of ZO-HGD through a real-world application to generating adversarial examples from a black-box deep neural network [28, 80]. We show that ZO-HGD outperforms ZO-SGD, ZO-SCD and ZO sign-based SGD methods in striking a graceful balance between query efficiency and attack success rate.

In the following section, we state the problem, and discuss two of the existing zeroth-order gradient estimators.

5.2 Preliminaries

In this section, we begin by presenting the formulation of the black-box optimization problems of our interest. We then review two commonly-used ZO gradient estimators, random gradient estimator (RGE) and coordinate-wise gradient estimator (CGE). We shall define the notation wherever we introduce a new mathematical entity. We refer the reader to Appendix A.4.1 for a tabular summary of all the notations used.

We consider the following black-box stochastic optimization problem

$$\min_{\mathbf{x}\in\mathbb{R}^d} f(\mathbf{x}) \triangleq \mathbb{E}_{\xi\sim\Xi} F(\mathbf{x};\xi), \tag{5.1}$$

where $\mathbf{x}$ denotes the d-dimensional optimization variable, $\xi \in \Xi$ denotes a stochastic variable with distribution Ξ (e.g., distribution of training samples), and $F(\cdot;\xi)$ is a smooth, possibly nonconvex loss function. By black-box, we mean that the objective function in (5.1) is only accessible via functional evaluations. To enable theoretical analysis, we impose two commonly-used assumptions on problem (5.1).

Assumption 1. *Gradient Lipschitz continuity:* The loss function f in (5.1) has Lipschitz continuous gradient with parameter L, i.e., $f(\mathbf{y}) \leq f(\mathbf{x}) + \langle \nabla f(\mathbf{x}), \mathbf{y}-\mathbf{x}\rangle + \frac{L}{2}\left\|\mathbf{y}-\mathbf{x}\right\|_2^2$, for all $\mathbf{x}, \mathbf{y}$, where ∇f denotes the gradient of f.

Assumption 2. *Bounded variance of stochastic gradients:* Suppose $(\mathbf{x})_i$ denotes the ith coordinate of a vector $\mathbf{x}$, and ζ is a given constant, then $\mathbb{E}[(\nabla F(\mathbf{x};\xi) - \nabla f(\mathbf{x}))_i^2] \leq \zeta^2, \forall\, i$.

Assumption 1 is fairly standard in the theoretical analysis of nonconvex optimization [61]. Assumption 2 enables a finer control on the variance of CGE [113, 19]. It also implies that $\mathbb{E}\left\|\nabla F(\mathbf{x};\xi)-\nabla f(\mathbf{x})\right\|^2 \leq \sigma^2 \triangleq d\zeta^2$.

RGE. Considering a set of *random* directional vectors $\{\mathbf{u}_i\}_{i=1}^{n_\mathrm{r}}$, RGE of the individual loss function $F(\mathbf{x};\xi)$ is given by the average of the finite difference approximations of the directional derivatives of $F(\mathbf{x};\xi)$ along these random directions [60, 119]:

$$\hat{\nabla}_{\mathrm{RGE}}F(\mathbf{x};\xi)=\frac{1}{n_\mathrm{r}}\sum_{i=1}^{n_\mathrm{r}}\frac{d\left[F(\mathbf{x}+\mu_\mathrm{r}\mathbf{u}_i;\xi)-F(\mathbf{x};\xi)\right]}{\mu_\mathrm{r}}\mathbf{u}_i, \tag{5.2}$$

where $\mu_\mathrm{r}>0$ is a perturbation radius (also called smoothing parameter), and each $\mathbf{u}_i$ is drawn from the uniform distribution on a unit sphere U_0 centered at $\mathbf{0}$ [60]. We define the smooth approximation of a function $g(\mathbf{x})$ with smoothing parameter μ_r as $g_{\mu_\mathrm{r}}(\mathbf{x})=\mathbb{E}_{\mathbf{u}\in U_0}[g(\mathbf{x}+\mu_\mathrm{r}\mathbf{u})]$. The rationale behind RGE (5.2) is that $\hat{\nabla}_{\mathrm{RGE}}F(\mathbf{x};\xi)$ is an *unbiased* estimate of $\nabla F_{\mu_\mathrm{r}}(\mathbf{x};\xi)$, leading to $\mathbb{E}_{\xi,\{\mathbf{u}_i\}}[\hat{\nabla}_{\mathrm{RGE}}F(\mathbf{x};\xi)]=\nabla f_{\mu_\mathrm{r}}(\mathbf{x})$. Note that $\nabla f_{\mu_\mathrm{r}}(\mathbf{x})$ itself is a biased approximation of the true gradient $\nabla f(\mathbf{x})$, with the bias controlled by μ_r.

CGE. Different from RGE, CGE is constructed by finite difference approximations of directional derivatives along the canonical basis vectors $\{\mathbf{e}_i\}_{i=1}^{d}$ in $\mathbb{R}^d$. The i-th component of CGE is defined as [96]

$$\hat{\nabla}_{\mathrm{CGE}}F_i(\mathbf{x};\xi)=\frac{\left[F(\mathbf{x}+\mu_{\mathrm{c},i}\mathbf{e}_i;\xi)-F(\mathbf{x}-\mu_{\mathrm{c},i}\mathbf{e}_i;\xi)\right]}{2\mu_{\mathrm{c},i}}\mathbf{e}_i, \tag{5.3}$$

where $\mu_{\mathrm{c},i}>0$ denotes the coordinate-wise smoothing parameter. $\hat{\nabla}_{\mathrm{CGE}}F(\mathbf{x};\xi)=\sum_{i=1}^{d}\hat{\nabla}_{\mathrm{CGE}}F_i(\mathbf{x};\xi)$ is the *full* CGE. If a random subset of the coordinates $\mathcal{I}\subseteq[d]$ is used (here $[d]$ denotes the set $\{1,2,\ldots,d\}$), rather than the full coordinate set $[d]$, we get the stochastic coordinate-wise gradient estimator [113, 69]

$$\hat{\nabla}_{\mathrm{CGE}}F_{\mathcal{I}}(\mathbf{x};\xi)=\frac{d}{n_\mathrm{c}}\sum_{i=1}^{d}I(i\in\mathcal{I})\hat{\nabla}_{\mathrm{CGE}}F_i(\mathbf{x};\xi), \tag{5.4}$$

where the cardinality of $\mathcal{I}$ is denoted by $|\mathcal{I}| = n_{\mathrm{c}}$, and $I(i \in \mathcal{I})$ is the indicator function which takes the value 1 if $i \in \mathcal{I}$ and 0 otherwise. Note that [113] sampled the elements of $\mathcal{I}$ *uniformly*, i.e., $\Pr(i \in \mathcal{I}) = n_{\mathrm{c}}/d$, for all $i \in [d]$. Hence, the multiplicative factor d/n_{c} in (5.4) ensures unbiasedness $\mathbb{E}_{\mathcal{I}}[\hat{\nabla}_{\mathrm{CGE}} F_{\mathcal{I}}(\mathbf{x}; \xi)] = \hat{\nabla}_{\mathrm{CGE}} F(\mathbf{x}; \xi)$. In this work, we generalize (5.4) to the case where the coordinates are instead, sampled *non-uniformly*.

Using the gradient estimate $\hat{\nabla} F(\mathbf{x}; \xi)$ based on either RGE or CGE, we can further define the approximation of the stochastic gradient of the objective in (5.1), over a set of stochastic samples $\{\xi \in \mathcal{B}\}$. This leads to

$$\hat{\nabla} F(\mathbf{x}; \mathcal{B}) = \frac{1}{|\mathcal{B}|} \sum_{\xi \in \mathcal{B}} \hat{\nabla} F(\mathbf{x}; \xi). \tag{5.5}$$

We also remark that compared to RGE, full CGE takes $O(d/n_{\mathrm{r}})$ times more function queries if $d > n_{\mathrm{r}}$. However, it has $O(d/n_{\mathrm{r}})$ times smaller gradient estimation error [119]. Inspired by this observation, we ask:

> "Can we design a convex combination of RGE and CGE to to strike an optimized balance between the gradient estimation accuracy and the query efficiency?"

Gradient estimators based on convex combinations of existing estimators have been proposed for solving stochastic nonconvex problems using first-order methods [175, 176, 115]. However, to the best of our knowledge, such estimators have not been considered in the zeroth-order context. Next, we propose our hybrid estimator, and discuss its theoretical properties.

5.3 Hybrid Gradient Estimator

Spurred by the capabilities and limitations of RGE and CGE, in what follows we propose a new hybrid gradient estimator (HGE) and its variance-controlled version using a coordinate importance sampling method.

Proposed HGE. In order to achieve the desired tradeoff between estimation accuracy and query efficiency, we combine RGE with CGE to obtain HGE

$$\hat{\nabla}_{\mathrm{HGE}} F(\mathbf{x}; \mathcal{B}^{\mathrm{r}}, \mathcal{B}^{\mathrm{c}}, \mathcal{I}) = \alpha \hat{\nabla}_{\mathrm{RGE}} F(\mathbf{x}; \mathcal{B}^{\mathrm{r}}) + (1-\alpha) \hat{\nabla}_{\mathrm{CGE}} F_{\mathcal{I}}(\mathbf{x}; \mathcal{B}^{\mathrm{c}}, \mathbf{p}), \tag{5.6}$$

where $\alpha \in [0,1]$ is the combination coefficient, $\mathcal{B}^{\mathrm{r}}$ and $\mathcal{B}^{\mathrm{c}}$ are mini-batches of stochastic samples as introduced in (5.5), $\hat{\nabla}_{\mathrm{RGE}} F(\mathbf{x}; \mathcal{B}^{\mathrm{r}})$ is given by (5.2) and (5.5), and $\hat{\nabla}_{\mathrm{CGE}} F_{\mathcal{I}}(\mathbf{x}; \mathcal{B}^{\mathrm{c}}, \mathbf{p})$ denotes the stochastic CGE (5.4) with the coordinate selection probability $\Pr(i \in \mathcal{I}) = p_i$, namely,

$$\hat{\nabla}_{\mathrm{CGE}} F_{\mathcal{I}}(\mathbf{x}; \xi, \mathbf{p}) = \sum_{i=1}^{d} \frac{I(i \in \mathcal{I})}{p_i} \hat{\nabla}_{\mathrm{CGE}} F_i(\mathbf{x}; \xi). \tag{5.7}$$

Different from (5.4), the coordinates of $\mathcal{I}$ in (5.7) can be sampled with unequal probabilities. It can be easily shown that $\mathbb{E}[\hat{\nabla}_{\mathrm{CGE}} F_{\mathcal{I}}(\mathbf{x}; \mathcal{B}^{\mathrm{c}}, \mathbf{p})] = \hat{\nabla}_{\mathrm{CGE}} f(\mathbf{x})$. In (5.6), α and $\mathbf{p}$ are design parameters. Their optimal values help us improve the variance of HGE relative to RGE, and the function query complexity relative to CGE. We next discuss how to optimize these parameters.

Design of coordinate selection probabilities $\mathbf{p}$. Recall that if no prior knowledge on gradient estimation is available, then a simple choice of $\mathbf{p}$ is that of equal probability across all coordinates [113], namely, the uniform distribution used in (5.4). However in HGE, a RGE $\mathbf{g} \triangleq \hat{\nabla}_{\mathrm{RGE}} F(\mathbf{x}; \mathcal{B}^{\mathrm{r}})$, known prior to computing CGE, can be employed as a probe estimate. Thus, we ask if $\mathbf{p}$ could be designed based on $\mathbf{g}$ to reflect the importance of coordinates to be selected. We next show that this question can be addressed by formulating the coordinate selection problem as the problem of gradient sparsification [188].

Given $\mathbf{p}$, the i-th coordinate of $\mathbf{g}$ is dropped with probability $1 - p_i$. This can be modeled using a Bernoulli random variable Z_i: $\Pr(Z_i = 1) = p_i$ and $\Pr(Z_i = 0) = 1 - p_i$. Defining $(Q(\mathbf{g}))_i = Z_i \cdot (g_i / p_i)$, note that $Q(\mathbf{g})$ is an unbiased estimator of $\mathbf{g}$. The variance can be bounded using $\|Q(\mathbf{g})\|^2 = \sum_{i=1}^{d} g_i^2 / p_i$, while the expected sparsity of $Q(\mathbf{g})$ is $\sum_{i=1}^{d} p_i$. To determine $\mathbf{p}$, we minimize the variance of the sparsified RGE under a constraint on the expected sparsity of the vector. That is, we solve the problem:

$$\min_{\mathbf{p}} \sum_{i=1}^{d} \frac{g_i^2}{p_i} \quad \text{subject to} \quad \sum_{i=1}^{d} p_i \leq n_{\mathrm{c}}, 0 < p_i \leq 1, \forall i. \tag{5.8}$$

where $n_{\mathrm{c}} \in \mathbb{N}_+$ denotes the coordinate selection budget, and $\mathbb{N}_+$ is the set of positive integers. The solution to (5.8) is given by the following proposition.

Proposition 5.3.1. *Suppose we denote by $g_{(1)}, g_{(2)}, \ldots, g_{(d)}$ the components of vector* g*, arranged in descending order of magnitudes. First, we find the smallest k such that*

$$|g_{(k+1)}| \, (n_{\mathrm{c}} - k) \leq \sum_{i=k+1}^{d} |g_{(i)}|,$$

is true, and denote by S_k the set of coordinates with the top k largest magnitudes of $|g_i|$. Then the i-th component of the probability vector $\mathbf{p}$ *is computed as*

$$p_i = \begin{cases} 1, & \textit{if } i \in S_k \\ \frac{|g_i|(n_{\mathrm{c}}-k)}{\sum_{j=k+1}^{d} |g_j|}, & \textit{if } i \notin S_k. \end{cases}$$

PROOF: See Appendix A.4.2. ■

The probability value p_i depends on the relative magnitude of g_i, with respect to the other elements of g. If n_{c} is large enough, then the coordinates corresponding to the largest elements are always sampled $(k > 0)$. In the extreme case of all entries having the same magnitude (for example, g is the 1-bit compressed version of a real-valued vector), $k = 0$ and $p_i = n_{\mathrm{c}}/d$ for all i. The probabilities $\{p_i\}$ obtained in Proposition 5.3.1 are used to compute (5.7).

Design of combination coefficient α. Once we select $\mathbf{p}$, we intend to select a combination coefficient α which can minimize the variance of HGE. In fact, the closed form expression of this variance is not tractable. Hence, we first upper bound the variance in the following proposition. Then, α is selected to minimize this upper bound.

Proposition 5.3.2. *The variance of HGE* (5.6) *is bounded as*

$$\begin{aligned}\mathbb{E}\left\|\hat{\nabla}_{\mathrm{HGE}}F(\mathbf{x};\mathcal{B}^{\mathrm{r}},\mathcal{B}^{\mathrm{c}},\mathcal{I})-\nabla f(\mathbf{x})\right\|^2 \leq 2\alpha^2\mathbb{E}\left\|\hat{\nabla}_{\mathrm{RGE}}F(\mathbf{x};\mathcal{B}^{\mathrm{r}})-\nabla f(\mathbf{x})\right\|^2 \\ + 2(1-\alpha)^2\mathbb{E}\left\|[\hat{\nabla}_{\mathrm{CGE}}F_{\mathcal{I}}(\mathbf{x};\mathcal{B}^{\mathrm{c}},\mathbf{p})]-\nabla f(\mathbf{x})\right\|^2.\end{aligned} \tag{5.9}$$

Moreover, suppose Assumption 1 and 2 hold. Given the probability vector $\mathbf{p}$*, the individual gradient estimators* $\hat{\nabla}_{\mathrm{RGE}}F(\mathbf{x};\mathcal{B}^{\mathrm{r}})$ *and* $\hat{\nabla}_{\mathrm{CGE}}F_{\mathcal{I}}(\mathbf{x};\mathcal{B}^{\mathrm{c}},\mathbf{p})$ *satisfy*

$$\begin{aligned}\mathbb{E}\left\|\hat{\nabla}_{\mathrm{RGE}}F(\mathbf{x};\mathcal{B}^{\mathrm{r}})-\nabla f(\mathbf{x})\right\|^2 \leq \frac{2}{|\mathcal{B}^{\mathrm{r}}|}\left(1+\frac{d}{n_{\mathrm{r}}}\right)\|\nabla f(\mathbf{x})\|^2+\frac{2\sigma^2}{|\mathcal{B}^{\mathrm{r}}|}\left(1+\frac{d}{n_{\mathrm{r}}}\right) \\ + \left(1+\frac{2}{|\mathcal{B}^{\mathrm{r}}|}+\frac{2}{n_{\mathrm{r}}|\mathcal{B}^{\mathrm{r}}|}\right)\frac{\mu_{\mathrm{r}}^2L^2d^2}{4},\end{aligned} \tag{5.10}$$

$$\begin{aligned}\mathbb{E}\left\|[\hat{\nabla}_{\mathrm{CGE}}F_{\mathcal{I}}(\mathbf{x};\mathcal{B}^{\mathrm{c}},\mathbf{p})]-\nabla f(\mathbf{x})\right\|^2 \leq \sum_{i=1}^{d}\frac{1}{p_i}\Big[2\left(\nabla f(\mathbf{x})\right)_i^2+\frac{3}{|\mathcal{B}^{\mathrm{c}}|}\left(\zeta^2+\frac{L^2\mu_{\mathrm{c},i}^2}{2}\right)+\frac{L^2\mu_{\mathrm{c},i}^2}{2}\Big] \\ -2\left\|\nabla f(\mathbf{x})\right\|^2,\end{aligned} \tag{5.11}$$

where recall from (5.6) *that* $\mathcal{B}^{\mathrm{c}}$ *and* $\mathcal{B}^{\mathrm{r}}$ *denote the sets of stochastic samples,* n_{r} *denotes the number of random directions (per stochastic sample) used in computing* $\hat{\nabla}_{\mathrm{RGE}}$ (5.2)*,* μ_{r} *is the RGE smoothing parameter,* $\{\mu_{\mathrm{c},i}\}_i$ *are the coordinate-wise CGE smoothing parameters, and* ζ *is the coordinate-wise variance (Assumption 2), with* $\sigma^2 = d\zeta^2$*.*

PROOF: First we prove (5.10). The proof of (5.11) is discussed in Appendix A.4.3. The variance of RGE is given by

$$\begin{aligned}\mathbb{E}\left\|\hat{\nabla}_{\mathrm{RGE}}F\left(\mathbf{x};\mathcal{B}^{\mathrm{r}}\right)-\nabla f\left(\mathbf{x}\right)\right\|^2 &= \mathbb{E}\left\|\hat{\nabla}_{\mathrm{RGE}}F\left(\mathbf{x};\mathcal{B}^{\mathrm{r}}\right)-\nabla f_{\mu_{\mathrm{r}}}(\mathbf{x})+\nabla f_{\mu_{\mathrm{r}}}(\mathbf{x})-\nabla f(\mathbf{x})\right\|^2 \\ &\overset{(a)}{=} \mathbb{E}\left\|\hat{\nabla}_{\mathrm{RGE}}F\left(\mathbf{x};\mathcal{B}^{\mathrm{r}}\right)-\nabla f_{\mu_{\mathrm{r}}}(\mathbf{x})\right\|^2+\left\|\nabla f_{\mu_{\mathrm{r}}}(\mathbf{x})-\nabla f(\mathbf{x})\right\|^2\end{aligned} \tag{5.12}$$

where (a) follows since $\mathbb{E}\hat{\nabla}_{\mathrm{RGE}}F(\mathbf{x};\mathcal{B}^{\mathrm{r}}) = \nabla f_{\mu_{\mathrm{r}}}(\mathbf{x})$ (see the discussion following (5.2)). Next, we upper bound the first term in (5.12).

$$
\begin{aligned}
&\mathbb{E}\left\|\hat{\nabla}_{\mathrm{RGE}}F\left(\mathbf{x};\mathcal{B}^{\mathrm{r}}\right)-\nabla f_{\mu_{\mathrm{r}}}(\mathbf{x})\right\|^2=\mathbb{E}_{\{\mathbf{u}\},\mathcal{B}^{\mathrm{r}}}\left\|\hat{\nabla}_{\mathrm{RGE}}F\left(\mathbf{x};\mathcal{B}^{\mathrm{r}}\right)-\nabla f_{\mu_{\mathrm{r}}}(\mathbf{x})\right\|^2\\
&\overset{(b)}{=}\mathbb{E}_{\{\mathbf{u}\},\mathcal{B}^{\mathrm{r}}}\left\|\frac{1}{|\mathcal{B}^{\mathrm{r}}|}\sum_{\xi\in\mathcal{B}^{\mathrm{r}}}\frac{1}{n_{\mathrm{r}}}\sum_{i=1}^{n_{\mathrm{r}}}\frac{d\left[F(\mathbf{x}+\mu_{\mathrm{r}}\mathbf{u}_{i,\xi};\xi)-F(\mathbf{x};\xi)\right]}{\mu_{\mathrm{r}}}\mathbf{u}_{i,\xi}-\nabla f_{\mu_{\mathrm{r}}}(\mathbf{x})\right\|^2\\
&=\frac{1}{|\mathcal{B}^{\mathrm{r}}|^2n_{\mathrm{r}}^2}\sum_{\xi\in\mathcal{B}^{\mathrm{r}}}\mathbb{E}_{\{\mathbf{u}\},\xi}\left\|\sum_{i=1}^{n_{\mathrm{r}}}\left(\frac{d\left[F(\mathbf{x}+\mu_{\mathrm{r}}\mathbf{u}_{i,\xi};\xi)-F(\mathbf{x};\xi)\right]}{\mu_{\mathrm{r}}}\mathbf{u}_{i,\xi}-\nabla f_{\mu_{\mathrm{r}}}(\mathbf{x})\right)\right\|^2\\
&\quad+\frac{1}{|\mathcal{B}^{\mathrm{r}}|^2n_{\mathrm{r}}^2}\sum_{\xi\neq\chi}\left\langle\mathbb{E}_{\xi}\sum_{i=1}^{n_{\mathrm{r}}}\mathbb{E}_{\mathbf{u}_{i,\xi}}\left(\frac{d\left[F(\mathbf{x}+\mu_{\mathrm{r}}\mathbf{u}_{i,\xi};\xi)-F(\mathbf{x};\xi)\right]}{\mu_{\mathrm{r}}}\mathbf{u}_{i,\xi}-\nabla f_{\mu_{\mathrm{r}}}(\mathbf{x})\right),\right.\\
&\qquad\qquad\left.\mathbb{E}_{\chi}\sum_{i=1}^{n_{\mathrm{r}}}\mathbb{E}_{\mathbf{u}_{i,\chi}}\left(\frac{d\left[F(\mathbf{x}+\mu_{\mathrm{r}}\mathbf{u}_{i,\chi};\chi)-F(\mathbf{x};\chi)\right]}{\mu_{\mathrm{r}}}\mathbf{u}_{i,\chi}-\nabla f_{\mu_{\mathrm{r}}}(\mathbf{x})\right)\right\rangle\\
&\overset{(c)}{=}\frac{1}{|\mathcal{B}^{\mathrm{r}}|^2n_{\mathrm{r}}^2}\sum_{\xi\in\mathcal{B}^{\mathrm{r}}}\sum_{i=1}^{n_{\mathrm{r}}}\mathbb{E}_{\mathbf{u}_{i,\xi},\xi}\left\|\frac{d\left[F(\mathbf{x}+\mu_{\mathrm{r}}\mathbf{u}_{i,\xi};\xi)-F(\mathbf{x};\xi)\right]}{\mu_{\mathrm{r}}}\mathbf{u}_{i,\xi}-\nabla f_{\mu_{\mathrm{r}}}(\mathbf{x})\right\|^2\\
&\quad+\frac{1}{|\mathcal{B}^{\mathrm{r}}|^2n_{\mathrm{r}}^2}\sum_{\xi\in\mathcal{B}^{\mathrm{r}}}\sum_{i\neq j}\mathbb{E}_{\xi}\left\langle\mathbb{E}_{\mathbf{u}_{i,\xi}}\left(\frac{d\left[F(\mathbf{x}+\mu_{\mathrm{r}}\mathbf{u}_{i,\xi};\xi)-F(\mathbf{x};\xi)\right]}{\mu_{\mathrm{r}}}\mathbf{u}_{i,\xi}-\nabla f_{\mu_{\mathrm{r}}}(\mathbf{x})\right),\right.\\
&\qquad\qquad\left.\mathbb{E}_{\mathbf{u}_{j,\xi}}\left(\frac{d\left[F(\mathbf{x}+\mu_{\mathrm{r}}\mathbf{u}_{j,\xi};\xi)-F(\mathbf{x};\xi)\right]}{\mu_{\mathrm{r}}}\mathbf{u}_{j,\xi}-\nabla f_{\mu_{\mathrm{r}}}(\mathbf{x})\right)\right\rangle\\
&\overset{(d)}{=}\frac{1}{|\mathcal{B}^{\mathrm{r}}|n_{\mathrm{r}}}\mathbb{E}_{\xi_1,\mathbf{u}_{1,\xi_1}}\left\|\frac{d\left[F(\mathbf{x}+\mu_{\mathrm{r}}\mathbf{u}_{1,\xi_1};\xi_1)-F(\mathbf{x};\xi_1)\right]}{\mu_{\mathrm{r}}}\mathbf{u}_{1,\xi_1}-\nabla f_{\mu_{\mathrm{r}}}(\mathbf{x})\right\|^2\\
&\quad+\frac{(n_{\mathrm{r}}-1)}{|\mathcal{B}^{\mathrm{r}}|n_{\mathrm{r}}}\mathbb{E}_{\xi_1}\left\|\nabla F_{\mu_{\mathrm{r}}}(\mathbf{x},\xi_1)-\nabla f_{\mu_{\mathrm{r}}}(\mathbf{x})\right\|^2\qquad\text{(for some }\xi_1\in\mathcal{B}^{\mathrm{r}}\text{)},
\end{aligned}
\tag{5.13}
$$

where (b) follows from the definition of RGE (5.2), (5.5). $\{\mathbf{u}_{i,\xi}\}$ denotes the set of random directions associated with sample ξ. (c) follows since the samples of $\mathcal{B}^{\mathrm{r}}$ are picked independently of each other, and the random directions associated with sample ξ, $\{\mathbf{u}_{i,\xi}\}$ are also independent of the random directions corresponding to another sample χ. Also,

$$
\mathbb{E}_{\xi}\mathbb{E}_{\mathbf{u}_{i,\xi}}\left[\frac{d\left[F(\mathbf{x}+\mu_{\mathrm{r}}\mathbf{u}_{i,\xi};\xi)-F(\mathbf{x};\xi)\right]}{\mu_{\mathrm{r}}}\mathbf{u}_{i,\xi}\right]=\nabla f_{\mu_{\mathrm{r}}}(\mathbf{x}).
$$

Hence,

$$\Bigg\langle \mathbb{E}_{\xi} \sum_{i=1}^{n_{\mathrm{r}}} \mathbb{E}_{\mathbf{u}_{i,\xi}} \left(\frac{d\left[F(\mathbf{x}+\mu_{\mathrm{r}}\mathbf{u}_{i,\xi};\xi) - F(\mathbf{x};\xi)\right]}{\mu_{\mathrm{r}}} \mathbf{u}_{i,\xi} - \nabla f_{\mu_{\mathrm{r}}}(\mathbf{x}) \right), \\ \mathbb{E}_{\chi} \sum_{i=1}^{n_{\mathrm{r}}} \mathbb{E}_{\mathbf{u}_{i,\chi}} \left(\frac{d\left[F(\mathbf{x}+\mu_{\mathrm{r}}\mathbf{u}_{i,\chi};\chi) - F(\mathbf{x};\chi)\right]}{\mu_{\mathrm{r}}} \mathbf{u}_{i,\chi} - \nabla f_{\mu_{\mathrm{r}}}(\mathbf{x}) \right) \Bigg\rangle = 0.$$

Further, in (5.13), (d) follows since

$$\mathbb{E}_{\mathbf{u}} \frac{d\left[F(\mathbf{x}+\mu_{\mathrm{r}}\mathbf{u};\xi) - F(\mathbf{x};\xi)\right]}{\mu_{\mathrm{r}}} \mathbf{u} = \nabla f_{\mu_{\mathrm{r}}}(\mathbf{x};\xi).$$

Also, both the samples $\xi \in \mathcal{B}^{\mathrm{r}}$ and the random directions $\{\mathbf{u}_{i,\xi}\}$ are picked independently and uniformly. We denote by ξ_1 and $\mathbf{u}_{i,\xi}$, a representative sample of both sets respectively. Next, we upper bound the two terms in (5.13). From Assumption 2 and [60, Lemma 4.2], we obtain

$$\mathbb{E}_{\xi,\mathbf{u}} \left\| \frac{d\left[F(\mathbf{x}+\mu_{\mathrm{r}}\mathbf{u};\xi) - F(\mathbf{x};\xi)\right]}{\mu_{\mathrm{r}}} \mathbf{u} - \nabla f_{\mu_{\mathrm{r}}}(\mathbf{x}) \right\|^2 \leq 2d\left[\|\nabla f(\mathbf{x})\|^2 + \sigma^2\right] + \frac{\mu_{\mathrm{r}}^2 L^2 d^2}{2}. \quad (5.14)$$

Also, the second term in (5.13) can be upper bounded as follows:

$$\begin{aligned} \mathbb{E}_{\xi}\left\|\nabla F_{\mu_{\mathrm{r}}}(\mathbf{x},\xi) - \nabla f_{\mu_{\mathrm{r}}}(\mathbf{x})\right\|^2 &\leq \mathbb{E}_{\xi}\left\|\nabla F_{\mu_{\mathrm{r}}}(\mathbf{x},\xi)\right\|^2 \\ &\leq 2\mathbb{E}_{\xi}\left\|\nabla F(\mathbf{x},\xi)\right\|^2 + \frac{\mu_{\mathrm{r}}^2 L^2 d^2}{2} \qquad \text{from [119, Lemma 1]} \\ &\leq 2\left[\mathbb{E}_{\xi}\left\|\nabla F(\mathbf{x},\xi) - \nabla f(\mathbf{x})\right\|^2 + \|\nabla f(\mathbf{x})\|^2\right] + \frac{\mu_{\mathrm{r}}^2 L^2 d^2}{2} \\ &\leq 2\left[\|\nabla f(\mathbf{x})\|^2 + \sigma^2\right] + \frac{\mu_{\mathrm{r}}^2 L^2 d^2}{2}. \qquad (5.15) \end{aligned}$$

Substituting (5.14), (5.15) in (5.13), we get

$$\mathbb{E}\left\|\hat{\nabla}_{\mathrm{RGE}} F\left(\mathbf{x};\mathcal{B}^{\mathrm{r}}\right) - \nabla f_{\mu_{\mathrm{r}}}(\mathbf{x})\right\|^2 \leq \frac{2}{|\mathcal{B}^{\mathrm{r}}|}\left(1 + \frac{d}{n_{\mathrm{r}}}\right)\left[\|\nabla f(\mathbf{x})\|^2 + \sigma^2\right] + \left(1 + \frac{1}{n_{\mathrm{r}}}\right)\frac{\mu_{\mathrm{r}}^2 L^2 d^2}{2|\mathcal{B}^{\mathrm{r}}|}. \quad (5.16)$$

Substituting (5.16) in (5.12), and using $\|\nabla f_{\mu_{\mathrm{r}}}(\mathbf{x}) - \nabla f(\mathbf{x})\| \leq \frac{\mu_{\mathrm{r}} d L}{2}$ from [119, Lemma 1], we get

$$\mathbb{E}\left\|\hat{\nabla}_{\mathrm{RGE}} F\left(\mathbf{x}; \mathcal{B}^{\mathrm{r}}\right) - \nabla f\left(\mathbf{x}\right)\right\|^2 \leq \frac{2}{|\mathcal{B}^{\mathrm{r}}|}\left(1 + \frac{d}{n_{\mathrm{r}}}\right)\left[\|\nabla f(\mathbf{x})\|^2 + \sigma^2\right] + \left(1 + \frac{2}{|\mathcal{B}^{\mathrm{r}}|} + \frac{2}{n_{\mathrm{r}}|\mathcal{B}^{\mathrm{r}}|}\right)\frac{\mu_{\mathrm{r}}^2 L^2 d^2}{4}. \tag{5.17}$$

■

The accuracy of $\hat{\nabla}_{\mathrm{RGE}}$ depends critically on the number of random directions n_{r} (5.2). Also, for $|\mathcal{B}^{\mathrm{r}}| \to \infty$, $\hat{\nabla}_{\mathrm{RGE}} \to \nabla f_{\mu_{\mathrm{r}}}$, and the bound (5.10) reduces to $O(\mu_{\mathrm{r}}^2 L^2 d^2)$. This is precisely the bound for the deterministic case [60, Lemma 4.1]. Similarly, the accuracy of $\hat{\nabla}_{\mathrm{CGE}}$ depends on the sampling probabilities $\{p_i\}$. If $p_i = 1$ for all i, and $|\mathcal{B}^{\mathrm{c}}| \to \infty$, $\hat{\nabla}_{\mathrm{CGE}} F_{\mathcal{I}}(\mathbf{x}; \mathcal{B}^{\mathrm{c}}, \mathbf{p}) \to \hat{\nabla}_{\mathrm{CGE}} f(\mathbf{x})$. The bound in (5.11) then reduces to the deterministic case upper bound $O(L^2 \sum_{i=1}^{d} \mu_{\mathrm{c},i}^2)$ [86, Lemma 3].

Substituting (5.10), (5.11) in (5.9), and minimizing over α, we obtain the optimal value α^*. We define $\bar{P} = \frac{1}{d}\sum_{i=1}^{d} \frac{1}{p_i}$. On simplification (see Appendix A.4.4), α^* reduces to

$$\alpha^* = \left[1 + \frac{(1 + d/n_{\mathrm{r}})}{\bar{P}}\right]^{-1}, \tag{5.18}$$

Recall that n_{r} and $\sum_i p_i (= n_{\mathrm{c}})$ respectively, are the function query budgets of RGE and CGE. Considering some extreme cases, if $n_{\mathrm{r}} \to \infty$, since $\bar{P} \geq 1$, we get $\alpha^* \geq \frac{1}{2}$. If $\bar{P} \to \infty$, then $\alpha^* \to 1$ and RGE dominates the estimator. On the other hand, if $n_{\mathrm{r}} = 0$, then $\alpha^* = 0$ since there is no RGE to assign any weight. The relative values of n_{r}, $\bar{P}$ determine the exact weights assigned to RGE, CGE in (5.6).

A relatively simple case is the one with uniform sampling of coordinates, i.e., $p_i = n_{\mathrm{c}}/d$, for all i. In this case, n_{c} is the query budget of CGE, and $\alpha^* = (1 + n_{\mathrm{c}}/d + n_{\mathrm{c}}/n_{\mathrm{r}})^{-1}$. If $n_{\mathrm{c}} \geq n_{\mathrm{r}}$, then $\alpha^* < \frac{1}{2}$ and HGE (5.6) reasonably assigns higher weight to CGE.

Next, we use the proposed HGE estimator to propose a zeroth-order hybrid gradient descent (ZO-HGD) algorithm. We shall also discuss the convergence properties of the algorithm for smoooth, nonconvex functions, and for smooth, convex and strongly convex functions.

5.4 ZO Hybrid Gradient Descent

In this section, we introduce the ZO hybrid gradient descent (ZO-HGD) algorithm to solve problem (5.1) with the aid of HGE (5.6). We then derive its convergence rate and discuss the performance of ZO-HGD in several special cases.

5.4.1 The Algorithm

Algorithm 8 ZO-HGD to solve problem (5.1)

```
1: Input: Initial point $\mathbf{x}_0$, number of iterations $T$, step sizes $\{\eta_t\}$, smoothing parameter $\mu_{\mathrm{r}}$ and number of random directions $n_{\mathrm{r}}$ for RGE, smoothing parameters $\{\mu_{\mathrm{c},i}\}_{i=1}^{d}$ for CGE, random sample set $\Xi$, coordinate selection budget $\{n_{\mathrm{c,t}}\}$ for CGE (5.8)
2: for $t = 0$ to $T - 1$ do
3:     Sample mini-batches $\mathcal{B}_t^{\mathrm{r}}, \mathcal{B}_t^{\mathrm{c}}$ from $\Xi$
4:     Compute RGE $\nabla_{\mathrm{r},t} = \hat{\nabla}_{\mathrm{RGE}} F(\mathbf{x}_t; \mathcal{B}_t^{\mathrm{r}})$ as (5.2) & (5.5)
5:     Compute importance sampling probabilities $\mathbf{p}_t$ with $n_{\mathrm{c}} = n_{\mathrm{c,t}}, \mathbf{g} = \nabla_{\mathrm{r},t}$ in Proposition 5.3.1
6:     Sample coordinate set $\mathcal{I}_t$ of cardinality $n_{\mathrm{c,t}}$ with $\Pr(i \in \mathcal{I}_t) = p_{t,i}$
7:     Compute CGE $\nabla_{\mathrm{c},t} = \hat{\nabla}_{\mathrm{CGE}} F_{\mathcal{I}_t}(\mathbf{x}_t; \mathcal{B}_t^{\mathrm{c}}, \mathbf{p}_t)$ as (5.7)
8:     Update: $\mathbf{x}_{t+1} = \mathbf{x}_t - \eta_t \left(\alpha_t \nabla_{\mathrm{r},t} + (1 - \alpha_t) \nabla_{\mathrm{c},t}\right)$
9: end for
10: Output: A solution $\bar{\mathbf{x}}_T$ is picked uniformly randomly from $\{\mathbf{x}_t\}_{t=1}^{T}$.
```

We present ZO-HGD in Algorithm 8, which consists of three main steps. First, a coarse gradient estimate $\nabla_{\mathrm{r},t}$ is acquired using RGE (line 4), and is employed as a probe signal to determine the probabilities $\mathbf{p}_t$ of coordinate-wise importance sampling (line 5). Second, a stochastic CGE is generated using the subset of coordinates $\mathcal{I}_t$ sampled according to $\mathbf{p}_t$ (line 6-7). Third, a HGE based descent step is used to update the optimization variable $\mathbf{x}_t$ per iteration (line 8).

The expected total number of function evaluations in Algorithm 8, known as the *Function Query Cost* (FQC), is given by $\sum_{t=0}^{T-1}(2n_{\mathrm{r}}|\mathcal{B}_t^{\mathrm{r}}| + 2n_{\mathrm{c,t}}|\mathcal{B}_t^{\mathrm{c}}|)$, where recall that $n_{\mathrm{c,t}}$ is the size of the coordinate set $\mathcal{I}_t$ sampled in line 6. If we assume the coordinate set size to be constant over time, i.e., $n_{\mathrm{c,t}} = n_{\mathrm{c}}$ at all times t, FQC reduces to $O(T(n_{\mathrm{r}} + n_{\mathrm{c}}))$. Note that if $n_{\mathrm{c,t}} = 0$, then HGE reduces to RGE, with $\alpha_t = 1$. Accordingly, Algorithm 8 becomes mini-batch ZO-SGD [60]. On the other hand, if $n_{\mathrm{r}} = 0$, HGE reduced to CGE and $\alpha_t = 0$. In this case, no prior knowledge is available to compute the sampling probability vector $\mathbf{p}_t$. If we sample the coordinates uniformly, i.e., $p_{t,i} = n_{\mathrm{c}}/d$, for all i, t, ZO-HGD reduces to ZO-SCD [113].

5.4.2 Technical challenges of ZO-HGD.

ZO-HGD is a non-trivial extension of both ZO-SGD and ZO-SCD because of the following differences. First, the *non-uniform* sampling of the coordinate sets $\{\mathcal{I}_t\}$ in CGE complicates the derivation of the upper bound on the variance of CGE (5.11), compared to the uniform sampling case [113]. Second, as discussed in Section 5.3, the choice of the *combination coefficients* $\{\alpha_t\}$ controls the variance of the proposed HGE relative to both RGE and CGE. However, α has a non-linear dependence on n_{r} (query budget for RGE), and $\{p_{t,i}\}$ (sampling probabilities for CGE). This makes choosing α_t nontrivial to achieve the graceful tradeoff between convergence speed and FQC. Next, we elaborate on the convergence of ZO-HGD.

5.4.3 Convergence analysis.

For ease of notation, as in Algorithm 8, we denote

$$\nabla_{\mathrm{r},t} = \hat{\nabla}_{\mathrm{RGE}} F(\mathbf{x}_t; \mathcal{B}_t^{\mathrm{r}}), \nabla_{\mathrm{c},t} = \hat{\nabla}_{\mathrm{CGE}} F_{\mathcal{I}_t}(\mathbf{x}_t; \mathcal{B}_t^{\mathrm{c}}, \mathbf{p}_t).$$

Also, we assume the coordinate-wise smoothing parameters in CGE to be fixed, i.e., $\mu_{\mathrm{c},i} = \mu_{\mathrm{c}}$ for all i. Recall that the function query budget of HGE depends only on n_{r} (number of random vectors in RGE), and the sampling probability values $\{p_{t,i}\}$ for CGE. The expected number of coordinates used in CGE at time t is $\sum_{i=1}^{d} p_{t,i} = n_{\mathrm{c}}$.

Our analysis begins with using the L-smoothness of f (Assumption 1) in the update step (line

8) in Algorithm 8,

$$f(\mathbf{x}_{t+1}) \leq f(\mathbf{x}_t) - \eta_t \left\langle \nabla f(\mathbf{x}_t), \alpha_t \nabla_{\mathrm{r},t} + (1-\alpha_t)\nabla_{\mathrm{c},t} \right\rangle + \frac{\eta_t^2 L}{2} \left\| \alpha_t \nabla_{\mathrm{r},t} + (1-\alpha_t)\nabla_{\mathrm{c},t} \right\|^2. \quad (5.19)$$

We denote by $Y_t \triangleq \{\mathcal{I}_t, \mathcal{B}_t^{\mathrm{c}}, \mathcal{B}_t^{\mathrm{r}}\}$ the randomness at step t. We denote by $\mathcal{Y}_t$ the σ-algebra generated by $\{Y_0, Y_1, \ldots, Y_{t-1}\}$. Taking conditional expectation in (5.19),

$$\begin{aligned}\mathbb{E}_{Y_t}\left[f(\mathbf{x}_{t+1}) \mid \mathcal{Y}_t\right] \leq f(\mathbf{x}_t) + \eta_t \alpha_t \underbrace{\left\langle -\nabla f(\mathbf{x}_t), \nabla f_{\mu_{\mathrm{r}}}(\mathbf{x}_t) \right\rangle}_{\mathrm{I}} + \eta_t (1-\alpha_t) \underbrace{\left\langle -\nabla f(\mathbf{x}_t), \hat{\nabla}_{\mathrm{CGE}} f(\mathbf{x}_t) \right\rangle}_{\mathrm{II}} \\ + \eta_t^2 L \alpha_t^2 \underbrace{\mathbb{E}_{Y_t}[\|\nabla_{\mathrm{r},t}\|^2 \mid \mathcal{Y}_t]}_{\mathrm{III}} + \eta_t^2 L (1-\alpha_t)^2 \underbrace{\mathbb{E}_{Y_t}[\|\nabla_{\mathrm{c},t}\|^2 \mid \mathcal{Y}_t]}_{\mathrm{IV}}, \quad (5.20)\end{aligned}$$

which holds since

(i) $\mathbb{E}_{Y_t}\left[\nabla_{\mathrm{r},t} \mid \mathcal{Y}_t\right] = \nabla f_{\mu_{\mathrm{r}}}(\mathbf{x}_t)$, where recall (Section 5.2) that $f_{\mu_{\mathrm{r}}}$ is a smooth approximation of f

(ii) $\mathbb{E}_{Y_t}\left[\nabla_{\mathrm{c},t} \mid \mathcal{Y}_t\right] = \hat{\nabla}_{\mathrm{CGE}} f(\mathbf{x}_t)$, which follows from (5.4), where $\hat{\nabla}_{\mathrm{CGE}} f$ is the full-coordinate CGE of f

(iii) $\|\mathbf{a}+\mathbf{b}\|^2 \leq 2\|\mathbf{a}\|^2 + 2\|\mathbf{b}\|^2$.

Next, we bound the terms I-IV of (5.20) in the following two results. We begin with I, II.

Proposition 5.4.1. *Suppose f satisfies Assumption 1, then the quantities I and II in* (5.20) *are upper bounded as*

$$\begin{aligned}\mathrm{I} &\leq -\frac{3}{4}\|\nabla f(\mathbf{x}_t)\|^2 + \left(\frac{\mu_{\mathrm{r}} d L}{2}\right)^2, \\ \mathrm{II} &\leq -\frac{3}{4}\|\nabla f(\mathbf{x}_t)\|^2 + L^2 d \mu_{\mathrm{c}}^2.\end{aligned}$$

where recall that μ_{r} and μ_{c} are the smoothing parameters, respectively for RGE and CGE.

PROOF: The inner product term I in (5.20) can be bounded as follows.

$$\begin{aligned}
\mathrm{I} &= -\left\langle \nabla f(\mathbf{x}_t), \nabla f_{\mu_\mathrm{r}}(\mathbf{x}_t) \right\rangle = -\left[\frac{\|\nabla f(\mathbf{x}_t)\|^2 + \|\nabla f_{\mu_\mathrm{r}}(\mathbf{x}_t)\|^2 - \|\nabla f(\mathbf{x}_t) - \nabla f_{\mu_\mathrm{r}}(\mathbf{x}_t)\|^2}{2}\right] \\
&\overset{(a)}{\leq} -\frac{\|\nabla f(\mathbf{x}_t)\|^2}{2} + \frac{1}{2}\left(\|\nabla f(\mathbf{x}_t) - \nabla f_{\mu_\mathrm{r}}(\mathbf{x}_t)\|^2 - \frac{\|\nabla f(\mathbf{x}_t)\|^2}{2}\right) + \frac{1}{2}\left(\frac{\mu_\mathrm{r} d L}{2}\right)^2 \\
&\leq -\frac{3}{4}\|\nabla f(\mathbf{x}_t)\|^2 + \frac{\mu_\mathrm{r}^2 d^2 L^2}{4} \qquad (5.21)
\end{aligned}$$

where (a) follows from the inequalities (i) $\|\mathbf{a}\|^2 \geq \frac{\|\mathbf{b}\|^2}{2} - \|\mathbf{a}-\mathbf{b}\|^2$, and (ii) $\|\nabla f(\mathbf{x}) - \nabla f_{\mu_\mathrm{r}}(\mathbf{x})\| \leq \frac{\mu_\mathrm{r} d L}{2}$ [60, Lemma 4.1]. Similarly, we can upper bound II in (5.20).

$$\begin{aligned}
\mathrm{II} = -\left\langle \nabla f(\mathbf{x}_t), \hat{\nabla}_{\mathrm{CGE}} f(\mathbf{x}_t) \right\rangle &\leq -\frac{3}{4}\|\nabla f(\mathbf{x}_t)\|^2 + \left\|\nabla f(\mathbf{x}_t) - \hat{\nabla}_{\mathrm{CGE}} f(\mathbf{x}_t)\right\|^2 \\
&\overset{(b)}{\leq} -\frac{3}{4}\|\nabla f(\mathbf{x}_t)\|^2 + L^2 d \mu_\mathrm{c}^2, \qquad (5.22)
\end{aligned}$$

where (b) follows from [86, Lemma 3]. ■

In first-order problems [17], given an unbiased estimator ($\tilde{\nabla} f$) of the gradient ∇f, we simply get $\mathbb{E}\langle \nabla f(\mathbf{x}), \tilde{\nabla} f(\mathbf{x})\rangle = \|\nabla f(\mathbf{x})\|^2$. However, $\nabla_{\mathrm{r},t}$ (RGE) and $\nabla_{\mathrm{c},t}$ (CGE) in Algorithm 8 could be biased estimates of ∇f. Proposition 5.4.1 bounds the deviation of the inner products I, II from $-\|\nabla f(\mathbf{x})\|^2$, in terms of the respective smoothness parameters $\mu_\mathrm{c}, \mu_\mathrm{r}$. We next bound III, IV in (5.20).

Proposition 5.4.2. *Suppose f satisfies Assumption 1, 2, then the quantities III and IV in* (5.20) *are upper bounded as*

$$\begin{aligned}
\mathrm{III} = \mathbb{E}_{Y_t}[\|\nabla_{\mathrm{r},t}\|^2 \mid \mathcal{Y}_t] &\leq \left[2 + \frac{4}{|\mathcal{B}^\mathrm{r}|}\left(1 + \frac{d}{n_\mathrm{r}}\right)\right]\|\nabla f(\mathbf{x})\|^2 + \frac{4\sigma^2}{|\mathcal{B}^\mathrm{r}|}\left(1 + \frac{d}{n_\mathrm{r}}\right) \\
&\quad + \left(1 + \frac{2}{|\mathcal{B}^\mathrm{r}|} + \frac{2}{n_\mathrm{r}|\mathcal{B}^\mathrm{r}|}\right)\frac{\mu_\mathrm{r}^2 L^2 d^2}{2}, \\
\mathrm{IV} = \mathbb{E}_{Y_t}[\|\nabla_{\mathrm{c},t}\|^2 \mid \mathcal{Y}_t] &\leq \sum_{i=1}^{d} \frac{1}{p_{t,i}}\left[2\left(\nabla f(\mathbf{x}_t)\right)_i^2 + \frac{3\zeta^2}{|\mathcal{B}_t^\mathrm{c}|} + \frac{L^2\mu_\mathrm{c}^2}{2}\left(1 + \frac{3}{|\mathcal{B}_t^\mathrm{c}|}\right)\right].
\end{aligned}$$

PROOF: First, we prove the bound on III.

$$
\begin{aligned}
\mathrm{III} &= \mathbb{E}_{Y_t}\left[\|\nabla_{\mathrm{r},t}\|^2 \mid \mathcal{Y}_t\right] = \mathbb{E}\left[\|\nabla_{\mathrm{r},t} - \nabla f(\mathbf{x}_t) + \nabla f(\mathbf{x}_t)\|^2 \middle| \mathcal{Y}_t\right] \\
&\overset{(a)}{\leq} 2\mathbb{E}\left[\|\nabla_{\mathrm{r},t} - \nabla f(\mathbf{x}_t)\|^2 \middle| \mathcal{Y}_t\right] + 2\left\|\nabla f(\mathbf{x}_t)\right\|^2 \\
&\overset{(b)}{\leq} \frac{4}{|\mathcal{B}_t^{\mathrm{r}}|}\left(1 + \frac{d}{n_{\mathrm{r}}}\right)\left[\|\nabla f(\mathbf{x})\|^2 + \sigma^2\right] + \left(1 + \frac{2}{|\mathcal{B}_t^{\mathrm{r}}|} + \frac{2}{n_{\mathrm{r}}|\mathcal{B}_t^{\mathrm{r}}|}\right)\frac{\mu_{\mathrm{r}}^2 L^2 d^2}{2} + 2\left\|\nabla f(\mathbf{x}_t)\right\|^2,
\end{aligned}
$$

where (a) follows from $\|\mathbf{a} + \mathbf{b}\|^2 \leq 2\|\mathbf{a}\|^2 + 2\|b\|^2$; (b) follows from (5.10) in Proposition 5.3.2. Next, we bound IV. Here, rather than using the bound in Proposition 5.3.2, we follow a slightly different route to derive a tighter bound.

$$
\begin{aligned}
\mathrm{IV} &= \mathbb{E}\left[\|\nabla_{\mathrm{c},t}\|^2 \mid \mathcal{Y}_t\right] = \mathbb{E}\left[\left\|\sum_{i=1}^{d} \frac{I(i \in \mathcal{I}_t)}{p_{t,i}} \hat{\nabla}_{\mathrm{CGE}} F_i(\mathbf{x}_t; \mathcal{B}_t^{\mathrm{c}})\right\|^2 \middle| \mathcal{Y}_t\right] \qquad \text{(see (5.7))} \\
&\overset{(c)}{=} \sum_{i=1}^{d} \mathbb{E}\left[\left\|\frac{I(i \in \mathcal{I}_t)}{p_{t,i}} \hat{\nabla}_{\mathrm{CGE}} F_i(\mathbf{x}_t; \mathcal{B}_t^{\mathrm{c}})\right\|^2 \middle| \mathcal{Y}_t\right] \\
&= \sum_{i=1}^{d} \frac{1}{p_{t,i}^2}\left[\mathbb{E}_{\mathcal{I}_t}\left(I(i \in \mathcal{I}_t)\right)^2 \mathbb{E}_{\mathcal{B}_t^{\mathrm{c}}}\left\|\frac{1}{|\mathcal{B}_t^{\mathrm{c}}|}\sum_{\xi \in \mathcal{B}_t^{\mathrm{c}}} \hat{\nabla}_{\mathrm{CGE}} F_i(\mathbf{x}_t; \xi)\right\|^2 \middle| \mathcal{Y}_t\right] \\
&\overset{(d)}{=} \sum_{i=1}^{d} p_{t,i}\frac{1}{p_{t,i}^2}\mathbb{E}\left[\frac{1}{|\mathcal{B}_t^{\mathrm{c}}|}\left\|\hat{\nabla}_{\mathrm{CGE}} F_i(\mathbf{x}_t; \xi_1) - \hat{\nabla}_{\mathrm{CGE}} f_i(\mathbf{x}_t)\right\|^2 + \left\|\hat{\nabla}_{\mathrm{CGE}} f_i(\mathbf{x}_t)\right\|^2 \middle| \mathcal{Y}_t\right]. \qquad (5.23)
\end{aligned}
$$

where (c) follows from (5.3) since $\hat{\nabla}_{\mathrm{CGE}} F_i(\mathbf{x}_t; \mathcal{B}_t^{\mathrm{c}})$ is aligned with the canonical basis vector $\mathbf{e}_i$, for all i. (d) follows since $\mathbb{E}[(I(i \in \mathcal{I}_t))^2] = p_{t,i}$ as seen in Appendix A.4.3. Also, $\mathcal{I}_t$ is independent of $\mathcal{B}_t^{\mathrm{c}}$, and the elements of $\mathcal{B}_t^{\mathrm{c}}$ are again sampled independent of each other, and since $\mathbb{E}_{\mathcal{B}_t^{\mathrm{c}}} \hat{\nabla}_{\mathrm{CGE}} F_i(\mathbf{x}_t; \mathcal{B}_t^{\mathrm{c}}) = \hat{\nabla}_{\mathrm{CGE}} f_i(\mathbf{x}_t)$. Next, we upper bound the two terms in (5.23).

$$
\mathbb{E}\left\|\hat{\nabla}_{\mathrm{CGE}} F_i(\mathbf{x}_t; \xi_1) - \hat{\nabla}_{\mathrm{CGE}} f_i(\mathbf{x}_t)\right\|^2 \leq 3\left[\zeta^2 + \frac{L^2 \mu_{\mathrm{c},i}^2}{2}\right], \tag{5.24}
$$

follows from (S60). The second term in (5.23) can be bounded using (S61),

$$
\left\|\hat{\nabla}_{\mathrm{CGE}} f_i(\mathbf{x}_t)\right\|^2 \leq \frac{L^2 \mu_{\mathrm{c},i}^2}{2} + 2\left(\nabla f(\mathbf{x}_t)\right)_i^2, \tag{5.25}
$$

where $(\mathbf{x})_i$ denotes the i-th coordinate of the vector $\mathbf{x}$. Substituting (5.24), (5.25) in (5.23), we get

$$\mathbb{E}\left[\|\nabla_{\mathrm{c},t}\|^2 \mid \mathcal{Y}_t\right] \leq \sum_{i=1}^{d} \frac{1}{p_{t,i}} \left[\frac{3\zeta^2}{|\mathcal{B}_t^{\mathrm{c}}|} + \frac{L^2 \mu_{\mathrm{c},i}^2}{2}\left(1 + \frac{3}{|\mathcal{B}_t^{\mathrm{c}}|}\right) + 2\left(\nabla f(\mathbf{x}_t)\right)_i^2\right].$$

Taking $\mu_{\mathrm{c},i} = \mu_{\mathrm{c}}$, for all $i \in [d]$, we get the bound on IV. ■

The bounds in Proposition 5.4.2 are expressed in terms of the gradient norm, variance σ^2, and the parameters which characterize RGE and CGE: number of random directions n_{r}, importance sampling probabilities $\{p_{t,i}\}$, batch-sizes $|\mathcal{B}^{\mathrm{c}}|$ and $|\mathcal{B}^{\mathrm{r}}|$, and the smoothness parameters μ_{r} and μ_{c}. If $|\mathcal{B}^{\mathrm{r}}| \to \infty$, then $\hat{\nabla}_{\mathrm{RGE}} \to \nabla f_{\mu_{\mathrm{r}}}$ as seen in Sec. 5.3, and the upper bound in Proposition 5.4.2 reduces to $2\mathbb{E}\|\nabla f(\mathbf{x})\|^2 + \mu_{\mathrm{r}}^2 L^2 d^2/2$. This is precisely the bound acquired using the exact ZO gradient estimator $\nabla f_{\mu_{\mathrm{r}}}$ [119, Lemma 1]. Here $\mu_{\mathrm{r}}^2 L^2 d^2/2$ quantifies the inevitable bias due to the use of the ZO gradient estimator $\nabla f_{\mu_{\mathrm{r}}}$.

Now that we have bounded all the terms in (5.20), we are ready to present the main result of this section. We define $\bar{P}_T = \frac{1}{dT}\sum_{t=0}^{T-1}\sum_{i=1}^{d}\frac{1}{p_{t,i}}$, and $d_{\mathrm{nr}} = 1 + d/n_{\mathrm{r}}$.

Theorem 5.4.3. *Suppose Assumption 1 and 2 hold, and the set of coordinate-wise probabilities* $\{p_{t,i}\}$ *satisfy* $p_{t,i} \geq c_t > 0$, *for all* i, t. *We choose the smoothing parameters* $\mu_{\mathrm{c}}, \mu_{\mathrm{r}}$ *such that* $\mu_{\mathrm{c}} = (\mu_{\mathrm{r}}\sqrt{d})/2$ *and* $\mu_{\mathrm{c}} L\sqrt{d} \leq \sigma$, *where* σ *is the variance. We take constant step sizes* $\eta_t = \eta \leq \frac{1}{24L}\min\{3c_t, 1/d_{\mathrm{nr}}\}$ *for all* t, *and combination coefficients* $\alpha_t = \alpha = \left[1 + (d_{\mathrm{nr}}/\bar{P}_T)\right]^{-1}$ *for all* t. *With* $\mu_{\mathrm{c}} = \mathcal{O}\left((d_{\mathrm{nr}}/(d^2T))^{1/4}(1 + d_{\mathrm{nr}}/\bar{P}_T)^{-1/4}\right)$, *we obtain*

$$\mathbb{E}\|\nabla f(\bar{\mathbf{x}}_T)\|^2 \leq \mathcal{O}\left(\sqrt{\frac{d_{\mathrm{nr}}}{T}\frac{1}{1 + d_{\mathrm{nr}}/\bar{P}_T}}\right). \tag{5.26}$$

Before discussing the proof of the theorem, a few comments are in order.

Remark. For the sake of simplifying the theorem statement, the combination coefficients $\{\alpha_t\}$ are assumed constant over time. Although this choice of $\alpha_t \equiv \alpha, \forall\, t$ is not informative from an implementation perspective (we do not know the sampling probabilities $\{p_{t,i}\}$ ahead of time), the expression of α still captures the trade-off between RGE and CGE through the ratio $\frac{d_{\mathrm{nr}}}{\bar{P}_T}$. Also, the two smoothing parameters $\mu_{\mathrm{c}}, \mu_{\mathrm{r}}$ are related as $\mu_{\mathrm{c}} = (\mu_{\mathrm{r}}\sqrt{d})/2$. Therefore, in Theorem 5.4.3 we have only given explicit expression for μ_{c}.

PROOF: Substituting the bounds on I, II from Proposition 5.4.1, and the bounds on $\mathrm{III}, \mathrm{IV}$ from Proposition 5.4.2 in (5.20), and taking expectation over the entire randomness till time t, we get

$$\begin{aligned}
\mathbb{E}f(\mathbf{x}_{t+1}) \le \mathbb{E}f(\mathbf{x}_t) &- \frac{3\eta_t\alpha_t}{4}\mathbb{E}\left\|\nabla f(\mathbf{x}_t)\right\|^2 + \eta_t\alpha_t\frac{\mu_{\mathrm{r}}^2 d^2 L^2}{4} - \frac{3\eta_t(1-\alpha_t)}{4}\mathbb{E}\left\|\nabla f(\mathbf{x}_t)\right\|^2 \\
&+ \eta_t(1-\alpha_t)L^2 d\mu_{\mathrm{c}}^2 + \eta_t^2 L\alpha_t^2\left[2\left(1+\frac{2}{|\mathcal{B}_t^{\mathrm{r}}|}+\frac{2d}{n_{\mathrm{r}}|\mathcal{B}_t^{\mathrm{r}}|}\right)\left\|\nabla f(\mathbf{x})\right\|^2\right. \\
&\qquad\left. + \frac{4\sigma^2}{|\mathcal{B}_t^{\mathrm{r}}|}\left(1+\frac{d}{n_{\mathrm{r}}}\right) + \left(1+\frac{2}{|\mathcal{B}_t^{\mathrm{r}}|}+\frac{2}{n_{\mathrm{r}}|\mathcal{B}_t^{\mathrm{r}}|}\right)\frac{\mu_{\mathrm{r}}^2 L^2 d^2}{2}\right] \\
&+ \eta_t^2 L(1-\alpha_t)^2\sum_{i=1}^{d}\frac{1}{p_{t,i}}\left[\frac{3}{|\mathcal{B}_t^{\mathrm{c}}|}\left(\zeta^2+\frac{L^2\mu_{\mathrm{c},i}^2}{2}\right)+\frac{L^2\mu_{\mathrm{c},i}^2}{2}+2\mathbb{E}\left(\nabla f(\mathbf{x}_t)\right)_i^2\right].
\end{aligned} \tag{5.27}$$

Note that $\sigma^2 = d\zeta^2$. Using $p_{t,i} \ge c_t$ for all $i \in [d]$, and $|\mathcal{B}_t^{\mathrm{r}}| \ge 1, |\mathcal{B}_t^{\mathrm{c}}| \ge 1, n_{\mathrm{r}} \ge 1$, (5.27) leads to

$$\begin{aligned}
\mathbb{E}f(\mathbf{x}_{t+1}) \le \mathbb{E}f(\mathbf{x}_t) &- \eta_t\left\{\frac{3}{4} - 6\eta_t L\alpha_t^2\left(1+\frac{d}{n_{\mathrm{r}}}\right) - 2L\frac{\eta_t(1-\alpha_t)^2}{c_t}\right\}\mathbb{E}\left\|\nabla f(\mathbf{x}_t)\right\|^2 \\
&+ \eta_t\alpha_t\frac{L^2\mu_{\mathrm{r}}^2 d^2}{4} + \eta_t(1-\alpha_t)L^2 d\mu_{\mathrm{c}}^2 + 4\eta_t^2 L\alpha_t^2\left(1+\frac{d}{n_{\mathrm{r}}}\right)\left[\mu_{\mathrm{r}}^2 d^2 L^2 + \sigma^2\right] \\
&+ 4L\eta_t^2(1-\alpha_t)^2\sum_{i=1}^{d}\frac{1}{p_{t,i}}\left[\zeta^2 + L^2\mu_{\mathrm{c}}^2\right].
\end{aligned} \tag{5.28}$$

Henceforth, we assume constant step-sizes $\eta_t = \eta$ for all t, and constant combination coefficients $\alpha_t = \alpha$, for all t. We denote $P_t = \frac{1}{d}\sum_{i=1}^{d}\frac{1}{p_{t,i}}$. Rearranging the terms in (5.28), summing over $t = 0$ to $T-1$, and dividing by ηT we get

$$\begin{aligned}
&\frac{1}{T}\sum_{t=0}^{T-1}\left\{\frac{3}{4} - \eta L 6\alpha^2\left(1+\frac{d}{n_{\mathrm{r}}}\right) - 2L\frac{\eta(1-\alpha)^2}{c_t}\right\}\mathbb{E}\left\|\nabla f(\mathbf{x}_t)\right\|^2 \\
&\le \frac{f(\mathbf{x}_0) - \mathbb{E}f(\mathbf{x}_T)}{\eta T} + L^2 d\mu_{\mathrm{c}}^2 + \alpha\left(\frac{L^2\mu_{\mathrm{r}}^2 d^2}{4} - L^2 d\mu_{\mathrm{c}}^2\right) \\
&\qquad + 4\eta L\alpha^2\left(1+\frac{d}{n_{\mathrm{r}}}\right)\left[\mu_{\mathrm{r}}^2 d^2 L^2 + \sigma^2\right] + 4L\eta(1-\alpha)^2\left[\sigma^2 + L^2\mu_{\mathrm{c}}^2 d\right]\frac{1}{T}\sum_{t=0}^{T-1}P_t.
\end{aligned} \tag{5.29}$$

To ease the notation, we define the following constants.

$$A = 4\sigma^2 + 4L^2\mu_{\mathrm{c}}^2 d,\ C = 4\left(1 + \frac{d}{n_{\mathrm{r}}}\right)\left[\mu_{\mathrm{r}}^2 d^2 L^2 + \sigma^2\right],\ (\Delta f) = f(\mathbf{x}_0) - f^*,\ \bar{P}_T = \frac{1}{T}\sum_{t=0}^{T-1} P_t.$$

Further, we select the smoothing parameters $\mu_{\mathrm{c}}, \mu_{\mathrm{r}}$ such that $\frac{L^2\mu_{\mathrm{r}}^2 d^2}{4} = L^2 d\mu_{\mathrm{c}}^2$. This simplifies (5.29) to

$$\begin{aligned}\frac{1}{T}\sum_{t=0}^{T-1}&\left\{\frac{3}{4} - \eta L4\alpha^2\left(1 + \frac{d}{n_{\mathrm{r}}}\right) - 2L\frac{\eta(1-\alpha)^2}{c_t}\right\}\mathbb{E}\left\|\nabla f(\mathbf{x}_t)\right\|^2 \\ &\leq \frac{(\Delta f)}{\eta T} + L^2 d\mu_{\mathrm{c}}^2 + \alpha^2\eta LC + L\eta(1-\alpha)^2 A\bar{P}_T.\end{aligned}$$

We choose η such that $\frac{3}{4} - \eta L6\alpha^2\left(1 + \frac{d}{n_{\mathrm{r}}}\right) - 2L\frac{\eta(1-\alpha)^2}{c_t} \geq \frac{1}{2}, \forall\, t$. This leads to

$$\begin{aligned}\eta &\leq \frac{1}{8L}\left[3\alpha^2\left(1 + \frac{d}{n_{\mathrm{r}}}\right) + \frac{(1-\alpha)^2}{c_t}\right]^{-1} \\ \Rightarrow \eta &\leq \frac{1}{24L}\min\left\{3c_t, \frac{n_{\mathrm{r}}}{d + n_{\mathrm{r}}}\right\}, \forall\, t,\end{aligned} \tag{5.30}$$

where (5.30) follows since $\left[3\alpha^2\left(1 + \frac{d}{n_{\mathrm{r}}}\right) + \frac{(1-\alpha)^2}{c_t}\right]^{-1}$ must attain its minimum value over $[0, 1]$ at one of the end points. Optimizing for η in (5.30) while satisfying (5.30), we get

$$\frac{1}{2T}\sum_{t=0}^{T-1}\mathbb{E}\left\|\nabla f(\mathbf{x}_t)\right\|^2 \leq 2\sqrt{\frac{(\Delta f)L}{T}\left(\alpha^2 C + (1-\alpha)^2 A\bar{P}_T\right)} + L^2 d\mu_{\mathrm{c}}^2. \tag{5.31}$$

Note that in (5.31), $L^2 d\mu_{\mathrm{c}}^2$ needs to be small to ensure convergence, and the smoothness parameter μ_{c} can be designed for that purpose. However, σ^2 being the variance, is not in our control. We choose the smoothing parameters $\mu_{\mathrm{c}}, \mu_{\mathrm{r}}$ to be small enough such that $\sigma^2 \geq L^2\mu_{\mathrm{c}}^2 d = L^2 d^2\mu_{\mathrm{r}}^2/4$. Consequently, $A \leq 8\sigma^2$, $C \leq 8\sigma^2\left(1 + \frac{d}{n_{\mathrm{r}}}\right)$. Substituting in (5.31), we get

$$\frac{1}{2T}\sum_{t=0}^{T-1}\mathbb{E}\left\|\nabla f(\mathbf{x}_t)\right\|^2 \leq 2\sqrt{\frac{(\Delta f)L}{T}8\sigma^2\left(\alpha^2\left(1 + \frac{d}{n_{\mathrm{r}}}\right) + (1-\alpha)^2\bar{P}_T\right)} + L^2 d\mu_{\mathrm{c}}^2. \tag{5.32}$$

Then, the optimal combination coefficient α^* is given by

$$\alpha^* = \left[1 + \frac{1 + \frac{d}{n_\mathrm{r}}}{\bar{P}_T}\right]^{-1}. \tag{5.33}$$

Substituting (5.33) in (5.32), we get

$$\frac{1}{2T}\sum_{t=0}^{T-1} \mathbb{E}\left\|\nabla f(\mathbf{x}_t)\right\|^2 \leq 2\sqrt{\frac{(\Delta f)L}{T} 8\sigma^2 \left(\frac{1 + \frac{d}{n_\mathrm{r}}}{1 + \frac{1+\frac{d}{n_\mathrm{r}}}{\bar{P}_T}}\right)} + L^2 d\mu_\mathrm{c}^2. \tag{5.34}$$

Since $\sigma^2 \geq L^2\mu_\mathrm{r}^2 d^2$, we choose μ_c such that

$$\begin{aligned} L^2 d\mu_\mathrm{c}^2 &= \mathcal{O}\left(\sqrt{\frac{(1+d/n_\mathrm{r})}{T}\frac{1}{1+\frac{(1+d/n_\mathrm{r})}{\bar{P}_T}}}\right) \\ \Rightarrow \mu_\mathrm{c} &= \mathcal{O}\left(\left(\frac{(1+d/n_\mathrm{r})}{d^2 T}\right)^{1/4} \frac{1}{\left(1+\frac{(1+d/n_\mathrm{r})}{\bar{P}_T}\right)^{1/4}}\right). \end{aligned}$$

Substituting $\mu_\mathrm{c}, \mu_\mathrm{r} = \frac{2\mu_\mathrm{c}}{\sqrt{d}}$ in (5.34) gives us

$$\mathbb{E}\left\|\nabla f(\bar{\mathbf{x}}_T)\right\|^2 \leq \frac{1}{T}\sum_{t=0}^{T-1}\mathbb{E}\left\|\nabla f(\mathbf{x}_t)\right\|^2 \leq \mathcal{O}\left(\sqrt{\frac{(1+d/n_\mathrm{r})}{T}\frac{1}{1+\frac{(1+d/n_\mathrm{r})}{\bar{P}_T}}}\right), \tag{5.35}$$

where the first inequality follows from Jensen's inequality. This completes the proof. ■

5.4.4 Comparison with Other Methods

We compare ZO-HGD with several existing methods in Table 5.1, in terms of the allowed smoothing parameter values, convergence rate and FQC. We compare these quantities in terms of their dependence on the problem dimension d, and the total number of iterations T. As illustrated, the smoothing parameters values for ZO-HGD are less restrictive than other ZO algorithms. A few clarifying remarks about the methods we compare with, are in order.

Remark.
- ZO-GD [137] is a deterministic algorithm ($|\mathcal{D}|$ is the size of the entire dataset), while all the other algorithms, including ours, are focused on solving stochastic problems.
- We have stated generalized results for ZO-SGD [61] (with b-point gradient estimator (GE)) and ZO-SCD [113] (with b coordinate GE).
- For ZO-SVRG [119] and ZO-signSGD [117], b is the number of random direction vectors used to compute a multi-point GE. ZO-SVRG gives better convergence rate, but under the assumption that all the individual stochastic functions $F(\cdot, \xi)$ in (5.1) are smooth. This is stronger than Assumption 1, which we use.
- For ZO-signSGD [117], b is the number of random direction vectors used to compute a multi-point GE. ZO-signSGD [117] only converges to a neighborhood of the stationary point. It ensures convergence only in a neighborhood of the stationary point. $\sqrt{d/b}$ give the size of the neighborhood.

Recall in our work, n_{r} is the number of random direction vectors used to compute RGE, n_{c} is the number of coordinates used to compute CGE, $d_{\mathrm{nr}} = 1 + d/n_{\mathrm{r}}$. We present more insights into Theorem 5.4.3, and the results in Table 5.1 in the next subsection.

5.4.5 Tradeoff between RGE and CGE in ZO-HGD.

Next, we consider some special cases of Theorem 5.4.3, based on the values of $\{p_{t,i}\}$ and n_{r}. Large n_{r} leads to a more accurate RGE (5.10), while large values of $\{p_{t,i}\}$ ensure a more accurate CGE (5.11). The tradeoff between RGE and CGE is captured by the ratio $d_{\mathrm{nr}}/\bar{P}_T$ in (5.26). Table 5.1 highlights the performance of ZO-HGD in three regimes, when $d_{\mathrm{nr}}/\bar{P}_T$ is $\ll 1, \gg 1$, and ≈ 1. We discuss one of the cases in detail, the remaining follow similar reasoning.

***Regime 1** ($d_{\mathrm{nr}} = 1 + \frac{d}{n_{\mathrm{r}}} \gg \bar{P}_T$).*

Since $\bar{P}_T$ is the mean of inverse probability values $\{1/p_{t,i}\}$, $\frac{1}{\bar{P}_T} \gg \frac{n_{\mathrm{r}}}{d}$ implies that on average, sampling probabilities are much greater than n_{r}/d. In other words, the per-iteration query budget

Method	Smoothing parameter	Convergence rate	FQC
ZO-GD	$O\left(\frac{1}{\sqrt{dT}}\right)$	$O\left(\frac{d}{T}\right)$	$O\left(\lvert\mathcal{D}\rvert T\right)$
ZO-SGD	$O\left(\frac{1}{d\sqrt{bT}}\right)$	$O\left(\sqrt{\frac{d}{bT}}\right)$	$O\left(Tb\right)$
ZO-SCD	$O\left(\frac{1}{\sqrt{bT}}+\left(\frac{1}{dbT}\right)^{1/4}\right)$	$O\left(\sqrt{\frac{d}{bT}}\right)$	$O\left(Tb\right)$
ZO-SVRG	$O\left(\frac{1}{\sqrt{dT}}\right)$	$O\left(\frac{d}{T}+\frac{1}{b}\right)$	$O\left(\lvert\mathcal{D}\rvert s+bsm\right)$ $T=sm$
ZO-signSGD	$O\left(\frac{1}{\sqrt{dT}}\right)$	$O\left(\sqrt{\frac{d}{T}}+\sqrt{\frac{d}{b}}\right)$	$O\left(bT\right)$
Our work: $(d_{\mathrm{nr}} \gg \bar{P}_T)$	$\mu_{\mathrm{c}} = O\left(\left(\frac{\bar{P}_T}{d^2T}\right)^{1/4}\right)$ $\mu_{\mathrm{r}} = \frac{2\mu_{\mathrm{c}}}{\sqrt{d}}$	$O\left(\sqrt{\frac{\bar{P}_T}{T}}\right)$	$O(Tn_{\mathrm{c}})$
Our work: $(d_{\mathrm{nr}} \ll \bar{P}_T)$	$\mu_{\mathrm{r}} = O\left(\frac{1}{d}\left(\frac{d_{\mathrm{nr}}}{T}\right)^{1/4}\right)$ $\mu_{\mathrm{c}} = \frac{\mu_{\mathrm{r}}\sqrt{d}}{2}$	$O\left(\sqrt{\frac{d_{\mathrm{nr}}}{T}}\right)$	$O(Tn_{\mathrm{r}})$
Our work: $d_{\mathrm{nr}} \approx \bar{P}_T$, $p_{t,i} = \frac{n_{\mathrm{c}}}{d}, \forall\, t, i$	$\mu_{\mathrm{r}} = O\left(\frac{1}{d}\left(\frac{1}{T}+\frac{d}{T(n_{\mathrm{r}}+n_{\mathrm{c}})}\right)^{1/4}\right)$ $\mu_{\mathrm{c}} = \frac{\mu_{\mathrm{r}}\sqrt{d}}{2}$	$O\left(\sqrt{\frac{1}{T}+\frac{d}{T(n_{\mathrm{r}}+n_{\mathrm{c}})}}\right)$	$O(T(n_{\mathrm{r}}+n_{\mathrm{c}}))$

Table 5.1: Comparison of different ZO algorithms, for solving smooth, nonconvex optimization problems.

of CGE (n_{c}) is much higher compared to that of RGE (n_{r}). From Theorem 5.4.3, we can obtain the CGE smoothing parameter

$$\mu_{\mathrm{c}} = \mathcal{O}\left(\left(\frac{\bar{P}_T}{d^2T}\right)^{1/4}\right)$$

and the convergence rate

$$\mathbb{E}\|\nabla f(\bar{\mathbf{x}}_T)\|^2 \leq \mathcal{O}\left(\sqrt{\frac{\bar{P}_T}{T}}\right). \tag{5.36}$$

Since $n_{\mathrm{c}} \gg n_{\mathrm{r}}$, the FQC is also dominated by CGE

$$O(T \cdot (n_{\mathrm{r}} + n_{\mathrm{c}})) = O(T \cdot n_{\mathrm{c}}).$$

All these results are also stated in Table 5.1.

Uniform sampling: In the special case of uniformly sampling all the coordinates for CGE, i.e., $p_{t,i} = n_\mathrm{c}/d$, $\bar{P}_T = d/n_\mathrm{c}$. Consequently, the convergence rate (5.36) reduces to

$$\mathbb{E}\|\nabla f(\bar{\mathbf{x}}_T)\|^2 \leq \mathcal{O}\left(\sqrt{\frac{d}{n_\mathrm{c}T}}\right),$$

and the smoothing parameter μ_c is

$$\mu_\mathrm{c} = \mathcal{O}\left(\frac{1}{(dn_\mathrm{c}T)^{1/4}}\right).$$

These are precisely the results for ZO-SCD in Table 5.1 (with b replaced by n_c). Also, FQC to achieve $\mathbb{E}\|\nabla f(\bar{\mathbf{x}}_T)\|^2 \leq \epsilon$ is

$$O(Tn_\mathrm{c}) = \mathcal{O}\left(\frac{d}{\epsilon^2}\right).$$

Remark. The reader would note that our method uses two smoothing parameters $\mu_\mathrm{c}, \mu_\mathrm{r}$, corresponding respectively to CGE and RGE components of HGE, which are related as $\mu_\mathrm{c} = (\mu_\mathrm{r}\sqrt{d})/2$ (see Theorem 5.4.3). In Table 5.1, we have only specified μ_c explicitly for the regime $d_\mathrm{nr} \gg \bar{P}_T$. This is because this regime is dominated by CGE.

The results for the other **two regimes**: 2) when $1 + \frac{d}{n_\mathrm{r}} \ll \bar{P}_T$, and 3) when $1 + \frac{d}{n_\mathrm{r}}$ and $\bar{P}_T$ are comparable, are stated in Table 5.1. For details, please see Appendix A.4.5.

Remark.

- As explained in Appendix A.4.5, in regime 2 ($d_\mathrm{nr} = 1 + \frac{d}{n_\mathrm{r}} \ll \bar{P}_T$), the query budget, and consequently the convergence rate and FQC are dominated by RGE. Therefore, we have only specified the corresponding smoothing parameter μ_r explicitly in Table 5.1 for this regime. The convergence rate and FQC, both depend only on the quantities related to RGE, namely $d_\mathrm{nr}, n_\mathrm{r}$. Also, this regime is comparable to ZO-SGD. Note that for comparable query budgets (b for ZO-SGD and n_r for our method) the bound on smoothing parameter μ_r values is larger than the corresponding bound in ZO-SGD.

- In Regime 3 ($d_\mathrm{nr} = 1 + \frac{d}{n_\mathrm{r}} \approx \bar{P}_T$), since the query budgets for RGE n_r and CGE n_c are comparable, we see dependence on both, in the smoothing paramters, convergence rate and FQC in Table 5.1. Since μ_r is the smaller of the two, and it is desirable for smoothing parameters to be large, we have only specified μ_{r} explicitly.

5.5 ZO-HGD for Convex Optimization

We now analyze the convergence properties of Algorithm 8 in cases where the objective function $f : \mathbb{R}^d \to \mathbb{R}$ is convex and strongly convex.

Assumption 3. The function f is defined on a compact set and is convex, such that $f(\mathbf{y}) \geq f(\mathbf{x}) + \langle \nabla f(\mathbf{x}), \mathbf{y} - \mathbf{x} \rangle$, $\forall\, \mathbf{x}, \mathbf{y} \in \mathrm{dom} f$. Further, the diameter of the set $\mathrm{dom} f$ is bounded by R, i.e., $\|\mathbf{y} - \mathbf{x}\| \leq R, \forall\, \mathbf{x}, \mathbf{y} \in \mathrm{dom} f$.

Assumption 4. $\|\nabla f(\mathbf{x})\| \leq G, \forall\, \mathbf{x} \in \mathrm{dom} f$ for constant G.

Assumption 5. ($\bar{\sigma}$*-Strong Convexity*). $\forall\, \mathbf{x}, \mathbf{y} \in \mathrm{dom} f$, f satisfies $f(\mathbf{y}) \geq f(\mathbf{x}) + \langle \nabla f(\mathbf{x}), \mathbf{y} - \mathbf{x} \rangle + \frac{\bar{\sigma}}{2} \|\mathbf{y} - \mathbf{x}\|^2$.

Assumption 4 is fairly standard in the convex optimization literature [50, 167]. We quantify the error in terms of the average difference between the expected function values at successive iterates and the optimal function value, $(1/T) \sum_{t=1}^{T} (\mathbb{E} f(\mathbf{x}_t) - f^*)$. For brevity, we only state the theorem for our convergence result.

Theorem 5.5.1. *Suppose Assumption 1, 2, 3, 4 hold and the coordinate-wise probabilities* $\{p_{t,i}\}$ *satisfy* $p_{t,i} \geq \bar{c} > 0$ *for all* i, t. *We define* $d_{\mathrm{nr}} = 1 + d/n_{\mathrm{r}}$.

1. *If the smoothing parameters* $\mu_{\mathrm{c}}, \mu_{\mathrm{r}}$ *are chosen such that* $\mu_{\mathrm{c}} = (\mu_{\mathrm{r}}\sqrt{d})/2$, $\sigma \geq L\mu_{\mathrm{c}}\sqrt{d}$. *A suitably chosen* α *gives*

$$\frac{1}{T} \sum_{t=1}^{T} \mathbb{E} f(\mathbf{x}_t) - f^* \leq O\left(\sqrt{\frac{1}{T} \frac{d_{\mathrm{nr}}}{1 + \bar{c} d_{\mathrm{nr}}}}\right).$$

2. *If Assumption 5 also holds, the smoothing parameters* $\mu_{\mathrm{c}}, \mu_{\mathrm{r}}$ *are such that* $\mu_{\mathrm{r}} = \mu_{\mathrm{c}}\sqrt{(dL)/\bar{\sigma}}$, *and the step-size* $\eta_t = \frac{8}{\bar{\sigma}(a+t)}$ *with* $a > 1$. *We define* $\hat{\mathbf{x}}_T = \frac{1}{S_T} \sum_{t=0}^{T-1} w_t \mathbf{x}_t$, *where* $w_t =$

$(a+t)^2$, *and* $S_T = \sum_{t=0}^{T-1} w_t$. *Then*

$$\mathbb{E} f(\hat{\mathbf{x}}_T) - f^* \le O\left(\frac{1}{T}\frac{d_{\mathrm{nr}}}{1+\bar{c}d_{\mathrm{nr}}}\right).$$

PROOF: See Appendix A.4.6 for details. ■

As in Section 5.4, we consider the special case of uniform sampling, $p_{t,i} = n_{\mathrm{c}}/d$ for all i, t. Let $q = n_{\mathrm{r}} + n_{\mathrm{c}}$. For convex functions, the convergence rate reduces to $O(\sqrt{(1+d/q)/T})$. For $O(q)$ function evaluations per iteration, this rate is order optimal [50]. For strongly convex functions, the convergence rate reduces to $\mathbb{E} f(\hat{\mathbf{x}}_T) - f^* \le O((1+d/q)/T)$. Note that this latter guarantee holds for a weighted average of iterates.

5.6 Simulation Results

In this section, we demonstrate the effectiveness of ZO-HGD via a real-world application of generating *adversarial examples* from a *black-box* deep neural network (DNN) [28]. Here the adversarial examples are defined by inputs with imperceptible perturbations crafted to mislead the DNN's prediction, and they provide a means of measuring the robustness of DNNs against adversarial perturbations [66, 26, 196].

We begin by formally presenting the problem of generating black-box adversarial examples. Let $(\mathbf{x}, t)$ denote a legitimate image $\mathbf{x}$ with the true label t. And $\mathbf{x}' = \mathbf{x} + \boldsymbol{\delta}$ denotes an adversarial example with the adversarial perturbation $\boldsymbol{\delta}$. Our goal is to design $\boldsymbol{\delta}$ for misclassifying M images $\{\mathbf{x}_i\}_{i=1}^M$ when a DNN is used as a decision maker. This leads to the optimization problem [26]

$$\underset{\boldsymbol{\delta}}{\text{minimize}} \quad \tfrac{\lambda}{M}\textstyle\sum_{i=1}^M f_{\boldsymbol{\theta}}(\mathbf{x}_i + \boldsymbol{\delta}) + \|\boldsymbol{\delta}\|_2^2 \tag{5.37}$$

where $f_{\boldsymbol{\theta}}(\mathbf{x}_i + \boldsymbol{\delta})$ is specified as the C&W untargeted attack loss [26] evaluated on a DNN model with parameters $\boldsymbol{\theta}$, and $\lambda > 0$ is a regularization parameter that strikes a balance between minimizing the attack loss and the ℓ_2 distortion. To solve problem (5.37), we consider the more realistic case in which the adversary does not have access to model parameters $\boldsymbol{\theta}$, and thus, it optimizes $\boldsymbol{\delta}$ only through function evaluations. Moreover, if $M = 1$ in problem (5.37), then the resulting solution is known as per-image adversarial perturbation [66]. And if $M > 1$, then the solution provides the *universal adversarial perturbation* applied to multiple benign images simultaneously [131].

In experiments, we consider a DNN model with 5 convolutional layers and 2 fully connected layers, and train it over the CIFAR-10 dataset for image classification [100]. We focus on the scenario to generate universal adversarial perturbations. Specifically, we conduct 50 random trials to solve problem (5.37), each of which randomly selects $M = 10$ images from CIFAR-10 testing data and sets $\lambda = 10$. We compare our proposed ZO-HGD algorithm with three baselines, ZO-SGD [61], ZO-SCD [113] and ZO-signSGD [117]. The rationale behind comparing with these baselines is that ZO-SGD has demonstrated a superior performance in query efficiency over ZO optimization methods that use extra variance reduction techniques [119], and ZO-SCD and ZO-signSGD are used as the backbones of many black-box attack generation algorithms via ZO optimization [28, 80]. In ZO-HGD (Algorithm 8), we set $n_{\mathrm{c}} = 50$ as the coordinate selection budget in CGE and $n_{\mathrm{r}} = 50$ when constructing RGE, and choose the combination coefficient to weight RGE and CGE in HGE as $\alpha_t = t/T$. In all of the methods, we set the maximum number of iterations $T = 1000$, the same query budget $(n_{\mathrm{c}} + n_{\mathrm{r}}) = 100$ per iteration, and a constant smoothing parameter 0.001. We pick the best learning rate for each method by greedily searching over the interval $[10^{-4}, 10^{-1}]$.

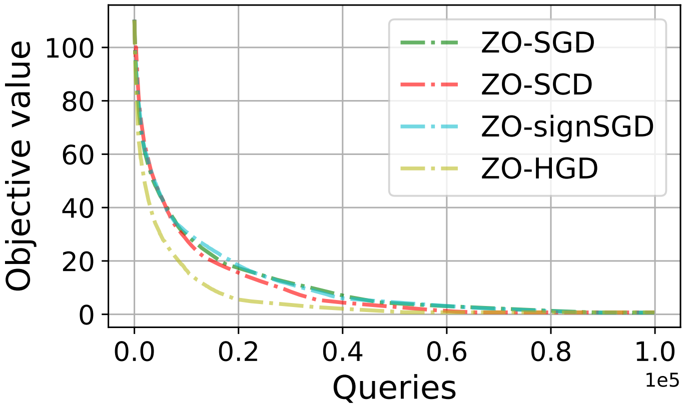


Fig. 5.1: Averaged objective value when solving problem (5.37) over 50 random trials versus the number of function queries.

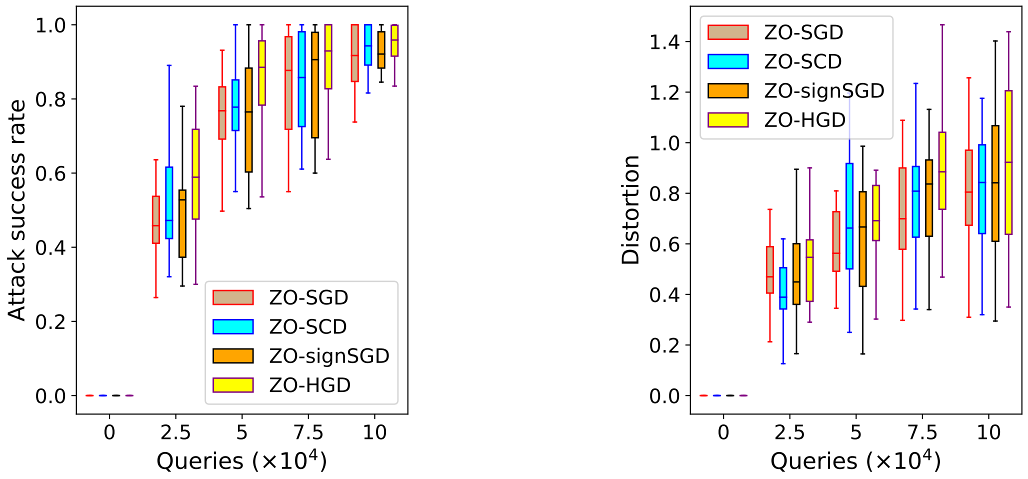


Fig. 5.2: Comparison of ZO-HGD with other methods for generating universal adversarial perturbations.

In Figure 5.1, we present the averaged objective value of problem (5.37) over 50 random trials versus the number of function queries. And in Figure 5.2, we show the attack success rate (ASR) of 50×10 perturbed images, and the associated ℓ_2-norm distortion for attack generation. We compare our proposed ZO-HGD with ZO-SGD, ZO-SCD and ZO-signSGD. Here the box plot summarizes the results of 50 random trials to solve problem (5.37). As we can see, the convergence speed of ZO-HGD is faster than ZO-SGD and ZO-SCD. This is supported by Figure 5.1: A smaller objective value is achieved using less number of function queries than the other algorithms. By dissecting the objective value of (5.37), Figure 5.2-(a) shows that ZO-HGD yields a significant ASR improvement over other algorithms even at the early iterations (e.g., at the use of $2.5 \sim 5 \times 10^4$ queries). Moreover, we observe from Figure 5.2-(b) that all the considered ZO algorithms result in comparable ℓ_2 distortion strength in general. This implies that the ASR improvement introduced by ZO-HGD is not at a significant cost of increasing distortion norm.

To further support this point, Table 5.2 presents the concrete adversarial examples and their metrics obtained from different ZO optimization methods. The results are summarized in terms of the number of queries required to achieve the first successful attack, the resulting adversarial examples, predicted label, and final ℓ_2 distortion. As we can see, ZO-HGD requires the least number of function queries to achieve the first successful attack of each image, and keeps the strength of converged perturbations comparable across methods.

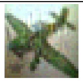
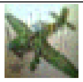
Original label	horse	car	bird	flight	frog
ZO-SGD ℓ_2: 0.67					
Query #	37	255	25	176	635
Predicted label	dog	ship	dog	car	car
ZO-SCD ℓ_2: 0.71					
Query #	35	239	21	166	610
Predicted label	dog	ship	dog	car	car
ZO-signSGD ℓ_2: 0.69					
Query #	39	285	29	188	675
Predicted label	dog	ship	dog	car	car
ZO-HGD ℓ_2: 0.77					
Query #	30	210	19	152	573
Predicted label	dog	ship	dog	car	car

Table 5.2: Randomly selected 5 over 10 images in one trial from our universal black-box attack generation using ZO-SGD, ZO-SCD, ZO-signSGD and ZO-HGD.

5.7 Summary

In this chapter, we proposed a novel hybrid gradient estimator (HGE) for black-box stochastic optimization, combining the advantages of the query-efficient random gradient estimator, and the more accurate coordinate-wise gradient estimator. Using importance sampling based coordinate selection, we further improved the variance of HGE. Building on top of this estimator, we proposed a ZO-hybrid gradient descent (ZO-HGD) algorithm, which generalized ZO-SGD and ZO-SCD. We conducted theoretical analysis of the method for smooth functions, which are non-convex, convex and strongly convex, and rigorously demonstrated the efficiency in terms of both iteration complexity and function query cost. The theoretical findings were corroborated by a real-world application to generate adversarial examples from a black-box deep neural network.

CHAPTER 6

CONCLUSION

6.1 Summary

In this dissertation, we studied two types of distributed system topologies: 1) mesh topology, and 2) star-shaped topology. We studied these systems under different contexts.

In Chapters 2 and 3, we considered systems with mesh topology. In Chapter 2, we proposed an efficient, decentralized BP message passing based method for simultaneous agent localization and multi-target tracking, for an unknown number of targets, under measurement-origin uncertainty. To save on the computational and communication costs of particle filters, we modeled the agent and target state densities using single Gaussians and Gaussian mixtures (GMs). We also proposed a novel decentralized Gibbs method for efficiently computing products of Gaussian mixtures. Through extensive simulations, we showed that the performance of the proposed decentralized algorithm is close to its centralized counterpart, and much better than the conventional separate localization and MTT approach. This work was focused on passive sensing, where the agents do not actively pursue the moving targets.

In Chapter 3, we solved the more abstract distributed online convex optimization (OCO) problem, with time-varying (potentially adversarial) constraints. We proposed a distributed primal-dual mirror descent based approach. We utilized the challenging, but more realistic metric of dynamic regret and fit. Regret quantifies the cumulative loss incurred by the system relative to the case where all the loss functions and constraints are known at a single location, ahead of time. Similarly, fit quantifies the cumulative constraint violation. We proved both the dynamic regret, as well as fit to be sublinear under mild, commonly used assumptions.

In Chapter 4, we considered distributed systems with star-shaped topology. We proposed a distributed variance-reduced algorithm, PR-SPIDER, for stochastic non-convex optimization. We proved convergence of our algorithm to a first-order stationary solution, while achieving the best known communication complexity. In terms of IFO complexity, we achieved the optimal rate. In addition, unlike many existing approaches, our algorithm is general enough to allow non-identical distributions of data across workers.

The work in Chapter 4 assumed access to gradients of the local objective functions. This might not always be feasible. In Chapter 5, we considered the more general problem, where we only have access to function values, and not the gradients. To simplify the analysis, we only considered the single-node case for this problem. We proposed a novel hybrid gradient estimator (HGE) for black-box stochastic optimization, combining the advantages of the query-efficient random gradient estimator, and the more accurate coordinate-wise gradient estimator. We conducted theoretical analysis of the method for smooth functions, which are non-convex, convex and strongly convex, and rigorously demonstrated the efficiency in terms of both iteration complexity and function query cost.

Next, we discuss some promising future directions of the work presented in this dissertation.

6.2 Future Directions

6.2.1 Federated Learning

We briefly discussed the recently proposed paradigm of Federated Learning (FL) [98] in Chapter 4. FL is a more general framework than conventional distributed optimization, in the following respects [92, 109].

- The number of workers/clients is typically very large (maybe even millions). Owing to the large number of nodes, not all are expected to participate in the learning process at any time.
- The participating clients can vary widely, both in terms of their individual capabilities (systems heterogeneity), as well in terms of the local loss function they posses (statistical heterogeneity).
- Privacy concerns about the local data present at the individual nodes need to be addressed.
- Owing to the heterogeneity of clients, questions of fairness and robustness also arise.

Past few years have seen a rapidly growing body of literature addressing these and other related questions on FL. Some of these directions and the corresponding open questions are quite close to the work done in Chapters 4 and 5, hence we briefly discuss these.

Communication Efficiency and Heterogeneity

Our work in Chapter 4 focused on achieving the optimal computational cost, while also minimizing the communication cost. For this purpose, the nodes performed local steps for some iterations, before communicating with the server. Other works have focused on additionally compressing the updates shared between the nodes and the server [161, 93, 71, 39, 67]. So far, the existing guarantees do not match the lower bounds on communication complexity which exist in the literature, making this an active area of research. Also, a principled understanding of how to model different kinds of heterogeneity is missing in the literature. Recently, some papers have illustrated that heterogeneity can lead the FedAvg algorithm to converge to the solution of a problem different from the original one [184, 130]. More fundamental understanding of statistical heterogeneity, beyond simple bounded gradient dissimilarity assumptions (as Assumption 4 in Chapter 4) is also missing in the literature.

Client Selection

In presence of a large number of clients, each round involves sampling a subset of clients to update the model. Conventional approaches, like FedAvg [123] involve sampling a subset of clients uniformly randomly. However, some recent work has shown the advantages of non-uniform sampling of clients [35]. Different sampling approaches have been proposed in the recent past [32, 151, 56]. However, a more principled understanding is still lacking.

Privacy and Fairness Guarantees

One of the motivations for FL is the reluctance of individual nodes to share their local data with a central server due to privacy concerns. Therefore, evaluating the privacy guarantees of different algorithms is one of the central themes of FL research. More recently, an interesting line of research has focused on the privacy-accuracy-communication trade-off. For example, the work in [64, 65, 31]. Also, due to the heterogeneity inherent in FL, fair resource allocation is another major concern [110, 108]. Both privacy and fairness, and their underlying trade-off is an interesting direction worth pursuing in federated learning research.

6.2.2 Online Nonconvex Optimization

Since the seminal work by Zinkevich [209], the work in online convex optimization has exclusively focused on convex objective functions. However, in the recent past, following the spectacular success of Deep Learning, research in nonconvex optimization has really taken off. Refer to [84] for a slightly outdated, yet comprehensive treatment. This has led to some initial work in the online domain with nonconvex objective functions. Perhaps the first work in this direction was [77], in which the authors defined the notion of *local regret* - the metric analogous to regret in the convex setting. This has been followed by a host of other papers [197, 4, 59, 169]. This line of work is especially of interest to us, since the motivating problem behind our work is that of distributed localization and tracking. The current paradigm of OCO, though captures the time-varying aspect of the problem, is too restrictive in terms of the underlying assumptions on the objective functions. Extending our work to nonconvex functions will bring us one step closer to modeling the full-generality of our motivating problem.

6.2.3 Stochastic Compositional Optimization

Compositional optimization is concerned with solving problems of the form

$$\min_{x} f(g(x))$$

where the functions f, g might themselves be expected values. This more general framework has applications in reinforcement learning, the recent paradigm of model-agnostic meta-learning (MAML) [54], and GAN-based compressed sensing [21], to name a few applications. Despite the large body of work in stochastic optimization, compositional optimization is relatively under-explored. Following the excellent work in [185], more recent work, including [186, 112], proposed improved algorithms for stochastic compositional optimization. Recently in [63], optimal convergence rates was achieved for two-level problems. This was generalized for the case of multi-level problems in [205, 11, 207]. However, this framework has so far not been considered in the FL context. With the recent application of MAML in FL [88, 51], the ground is ripe for more work in this direction.

APPENDIX A

APPENDIX

A.1 Cooperative Self-localization and Multi-Target Tracking

We compare the performance of a particle-filter based implementation of our proposed method (PM) with that of a reference method (RM) that performs agent self-localization SPAWN [192] followed by [126] for MTT. Figure A1 shows the ground truth trajectory of the mobile agents and

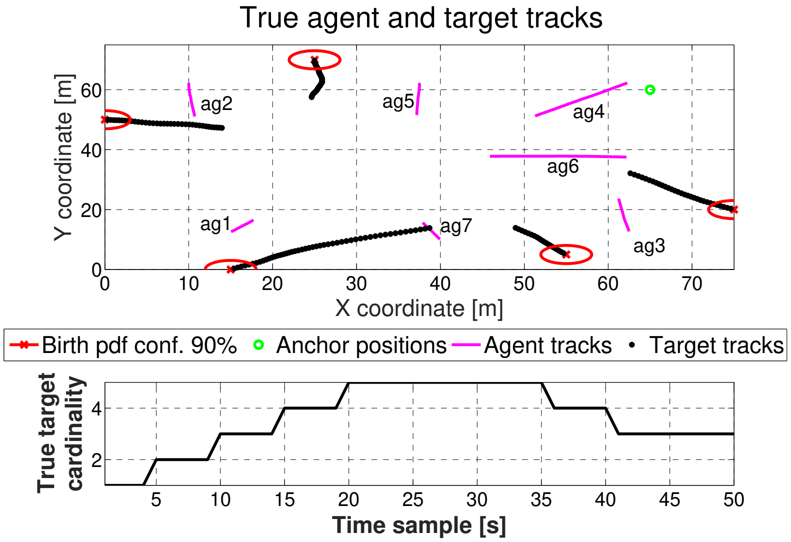


Fig. A1: Ground truth trajectories of agents and targets, plotted over time.

the targets, over a span of 50 time steps and the true number of targets as a function of time. We have considered a network composed of one fixed agent (i.e., anchor) and $S = 7$ mobile agents and a maximum of 5 targets, over a region of interest (ROI) given by $[0, 75 \text{ m}] \times [0, 75 \text{ m}]$. Each mobile agent has a measurement and communication range of 30 m. The anchor has a measurement and communication range of 100 m.

The kinematic model of the agents and targets is the same as in Section 2.6. The measurement noise in the agent state vector $\mathbf{R}_n$ at $\mathbf{R}_n = \text{diag}(2m^2, 1deg^2)$. Birth PTs are appended to the persistent PTs in the prediction step of the filters. The birth locations are shown in Figure A1. Birth probabilities are set to 10^{-3}, the probability of target survival is 0.99 and target probability of detection is $P_d^{(s)} = 0.9 \; \forall \; s$. At each frame, the clutter process follows a Poisson distribution with rate 5, and the clutter points are distributed uniformly over the ROI. Particle representations of agent and target tracks are employed in the two filters with 500 particles per track. The threshold on the probability of existence for an estimated target is set to 0.5 in both filters.

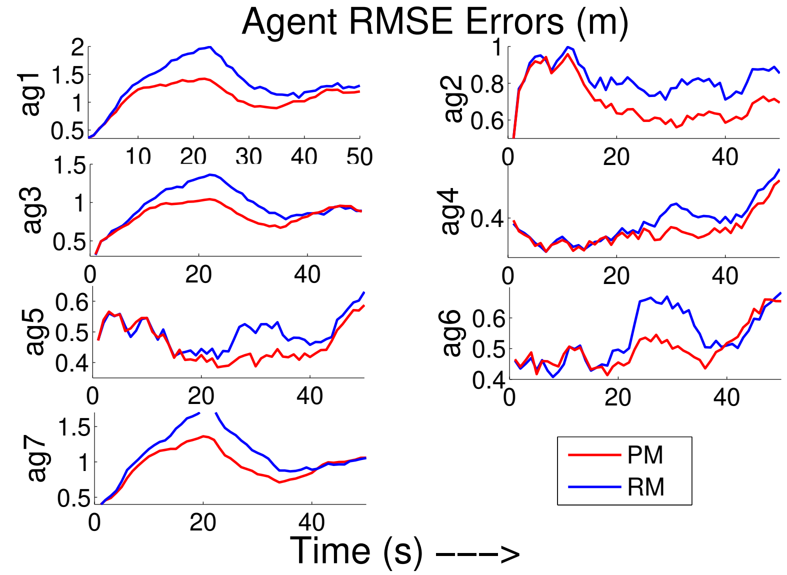


Fig. A2: Average RMSE in agent location estimates over time.

Keeping the agent and target tracks fixed, 100 Monte Carlo simulation runs are obtained by regenerating the measurement sets. In Figure A2, we have plotted the average root mean squared errors (RMSE) in the agent location estimates over time, for all the agents. For agents close to the anchor or with multiple neighboring agents (e.g., agents $4, 5, 6$), the self-localization performance of the two methods is similar. However, for agents which do not have many neighbouring agents to facilitate self-localization (e.g., agents $1, 2, 7$), our approach takes advantage of the probabilistic information transfer from target tracking. Therefore, the PM has a consistent performance improvement over RM.

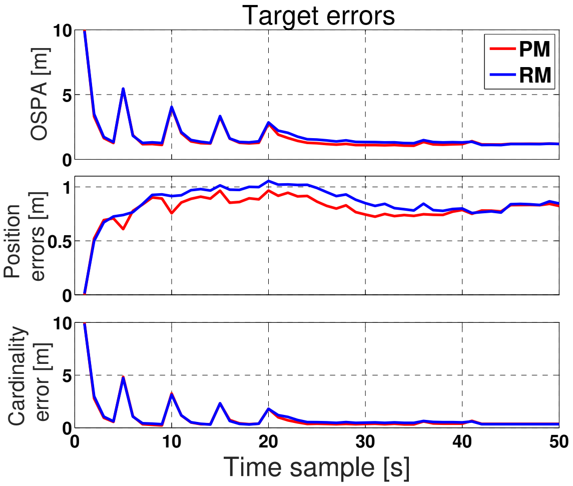


Fig. A3: Mean OSPA error, position error, and cardinality error for targets plotted over time.

In Figure A3, we showcase target tracking performance via the Optimum Sub-Pattern Assignment (OSPA) error averaged over the same 100 Monte Carlo simulation runs. The OSPA employs two parameters: the cut-off, set to 10m and the order, set to 1. The spikes in the cardinality OSPA correspond with the times of target births, due to the delay in estimating a new track for the two filters. Since the proposed method provides better estimates of agent locations, it also results in better estimates of target locations specifically in the precision of target tracks. The performance gap between the two methods is smaller for MTT than for CS, since there are always a few sufficiently well localized agents providing good target tracks.

A.2 Distributed Online Convex Optimization

A.2.1 Proof of Lemma 3.5.1

PROOF: We prove (3.20) by induction. $\mathbf{q}_{i,0} = \mathbf{0}$ by initialization. Also, since $g_{i,0}(\cdot) \equiv \mathbf{0}$, $\mathbf{b}_{i,1} = \mathbf{0}$. Therefore, $\mathbf{q}_{i,1} = \mathbf{0}_m \Rightarrow \|\mathbf{q}_{i,1}\| \leq \frac{F}{\beta_1}$. Let us assume (3.20) be true at time t, $\forall\, i \in [n]$. For $t+1$, first note that

$$(1 - \gamma_{t+1}\beta_{t+1})\mathbf{q}_{i,t} + \gamma_{t+1}\mathbf{b}_{i,t+1} \preceq (1 - \gamma_{t+1}\beta_{t+1})\mathbf{q}_{i,t} + \gamma_{t+1} g_{i,t}(\mathbf{y}_{i,t+1}), \tag{S1}$$

where (S1) follows from step 8 in Algorithm 4 using convexity of $g_{i,t}(\cdot)$. Also, for vectors $\mathbf{x}, \mathbf{y}$ such that $\mathbf{x} \preceq \mathbf{y}$, $\|[\mathbf{x}]_+\| \leq \|\mathbf{y}\|$. Therefore,

$$\begin{aligned}
\|\mathbf{q}_{i,t+1}\| &\leq (1 - \gamma_{t+1}\beta_{t+1})\,\|\mathbf{q}_{i,t}\| + \gamma_{t+1}\,\|g_{i,t}(\mathbf{y}_{i,t+1})\| \\
&\leq (1 - \gamma_{t+1}\beta_{t+1})\frac{F}{\beta_t} + \gamma_{t+1} F \overset{(a)}{\leq} \frac{F}{\beta_{t+1}}, \forall\, i \in [n],
\end{aligned} \tag{S2}$$

where (a) follows since $\{\beta_t\}$ is a non-increasing sequence.

Next we prove (3.21). First note that

$$\begin{aligned}
\|\mathbf{q}_{i,t+1} - \mathbf{q}_i\|^2 &\overset{(b)}{\leq} \|(1 - \gamma_{t+1}\beta_{t+1})\mathbf{q}_{i,t} + \gamma_{t+1}\mathbf{b}_{i,t+1} - \mathbf{q}_i\|^2 \\
&= \|\mathbf{q}_{i,t} - \mathbf{q}_i\|^2 + \gamma_{t+1}^2\|\mathbf{b}_{i,t+1} - \beta_{t+1}\mathbf{q}_{i,t}\|^2 + 2\gamma_{t+1}\mathbf{q}_{i,t}^T[\nabla g_{i,t}(\mathbf{x}_{i,t})](\mathbf{y}_{i,t+1} - \mathbf{x}_{i,t}) \\
&- 2\gamma_{t+1}\mathbf{q}_i^T[\nabla g_{i,t}(\mathbf{x}_{i,t})](\mathbf{y}_{i,t+1} - \mathbf{x}_{i,t}) + 2\gamma_{t+1}(\mathbf{q}_{i,t} - \mathbf{q}_i)^T g_{i,t}(\mathbf{x}_{i,t}) \\
&- 2\gamma_{t+1}\beta_{t+1}(\mathbf{q}_{i,t} - \mathbf{q}_i)^T\mathbf{q}_{i,t}
\end{aligned} \tag{S3}$$

where (b) follows from (3.19). We first bound the second term in (S3).

$$\begin{aligned}
\|\mathbf{b}_{i,t+1} - \beta_{t+1}\mathbf{q}_{i,t}\| &\leq \|\mathbf{b}_{i,t+1}\| + \beta_{t+1}\|\mathbf{q}_{i,t}\| \\
&\overset{(c)}{\leq} \|\nabla g_{i,t}(\mathbf{x}_{i,t})\|\|\mathbf{y}_{i,t+1} - \mathbf{x}_{i,t}\| + \|g_{i,t}(\mathbf{x}_{i,t})\| + \beta_{t+1}\frac{F}{\beta_{t+1}}
\end{aligned}$$

$$\overset{(d)}{\leq} Gd((X)) + 2F = B_1, \tag{S4}$$

where (c) follows from Cauchy-Schwarz inequality and (3.20). (d) follows from assumptions (B3), (B4) and (3.7). For the fourth term in (S3),

$$\begin{aligned} -2\gamma_{t+1}\mathbf{q}_i^T[\nabla g_{i,t}(\mathbf{x}_{i,t})](\mathbf{y}_{i,t+1} - \mathbf{x}_{i,t}) &\leq 2\gamma_{t+1}\|\mathbf{q}_i\|\|\nabla g_{i,t}(\mathbf{x}_{i,t})\|\|\mathbf{y}_{i,t+1} - \mathbf{x}_{i,t}\| \\ &\overset{(e)}{\leq} 2\gamma_{t+1}\left(\frac{G^2\alpha_{t+1}}{\mu}\|\mathbf{q}_i\|^2 + \frac{\mu}{4\alpha_{t+1}}\|\mathbf{y}_{i,t+1} - \mathbf{x}_{i,t}\|^2\right), \end{aligned} \tag{S5}$$

where (e) follows from (B4) and using Young's inequality $xy \leq \frac{x^2}{2a} + \frac{ay^2}{2}$, for $a > 0$. For the last term in (S3)

$$\begin{aligned} -2\gamma_{t+1}\beta_{t+1}(\mathbf{q}_{i,t} - \mathbf{q}_i)^T\mathbf{q}_{i,t} &\overset{(f)}{=} -2\gamma_{t+1}\beta_{t+1}\left[\frac{\|\mathbf{q}_{i,t} - \mathbf{q}_i\|^2 + \|\mathbf{q}_{i,t}\|^2 - \|\mathbf{q}_i\|^2}{2}\right] \\ &\leq \gamma_{t+1}\beta_{t+1}\left(\|\mathbf{q}_i\|^2 - \|\mathbf{q}_{i,t} - \mathbf{q}_i\|^2\right), \end{aligned} \tag{S6}$$

where (f) follows from $\mathbf{x}^T\mathbf{y} = \frac{\|\mathbf{x}\|^2+\|\mathbf{y}\|^2-\|\mathbf{x}-\mathbf{y}\|^2}{2}$. Define $\Delta_{t+1} \triangleq \sum_{i=1}^{n}[\|\mathbf{q}_{i,t+1} - \mathbf{q}_i\|^2 - (1 - \gamma_{t+1}\beta_{t+1})\|\mathbf{q}_{i,t} - \mathbf{q}_i\|^2]$. Using the upper bounds (S4)-(S6) in (S3), and summing over all $i \in [n]$ we get (3.21). ■

A.2.2 Proof of Lemma 3.5.2

PROOF: Using convexity of $f_{i,t}$

$$\begin{aligned} f_{i,t}(\mathbf{x}_{i,t}) - f_{i,t}(\mathbf{x}_t^*) &\leq \langle \nabla f_{i,t}(\mathbf{x}_{i,t}), \mathbf{x}_{i,t} - \mathbf{x}_t^* \rangle \\ &= \langle \nabla f_{i,t}(\mathbf{x}_{i,t}), \mathbf{x}_{i,t} - \mathbf{y}_{i,t+1} \rangle + \langle \nabla f_{i,t}(\mathbf{x}_{i,t}), \mathbf{y}_{i,t+1} - \mathbf{x}_t^* \rangle \end{aligned} \tag{S7}$$

For the first term in (S7)

$$\langle \nabla f_{i,t}(\mathbf{x}_{i,t}), \mathbf{x}_{i,t} - \mathbf{y}_{i,t+1} \rangle \leq G\|\mathbf{x}_{i,t} - \mathbf{y}_{i,t+1}\|$$

$$\leq \frac{G^2\alpha_{t+1}}{\mu} + \frac{\mu}{4\alpha_{t+1}}\|\mathbf{y}_{i,t+1} - \mathbf{x}_{i,t}\|^2, \tag{S8}$$

where (S8) follows from assumption (B4). For the second term in (S7)

$$\langle \nabla f_{i,t}(\mathbf{x}_{i,t}), \mathbf{y}_{i,t+1} - \mathbf{x}_t^*\rangle \leq \langle a_{i,t+1}, \mathbf{y}_{i,t+1} - \mathbf{x}_t^*\rangle + \left\langle [\nabla g_{i,t}(\mathbf{x}_{i,t})]^T \mathbf{q}_{i,t}, \mathbf{x}_t^* - \mathbf{y}_{i,t+1}\right\rangle, \tag{S9}$$

where (S9) follows from step 6 in Algorithm 4. Next, we bound the two inner products in (S9). First,

$$\begin{aligned}&\left\langle [\nabla g_{i,t}(\mathbf{x}_{i,t})]^T \mathbf{q}_{i,t}, \mathbf{x}_t^* - \mathbf{x}_{i,t} + \mathbf{x}_{i,t} - \mathbf{y}_{i,t+1}\right\rangle \\ &\leq \mathbf{q}_{i,t}^T \left[(g_{i,t}(\mathbf{x}_t^*) - g_{i,t}(\mathbf{x}_{i,t})) + \nabla g_{i,t}(\mathbf{x}_{i,t})\left(\mathbf{x}_{i,t} - \mathbf{y}_{i,t+1}\right)\right], \qquad \text{(S10)}\end{aligned}$$

where (S10) follows from the convexity of $g_{i,t}(\cdot)$. Secondly,

$$\begin{aligned}\langle a_{i,t+1}, \mathbf{y}_{i,t+1} - \mathbf{x}_t^*\rangle &\leq \frac{1}{\alpha_{t+1}}\Big[\mathcal{D}_{\mathcal{R}}(\mathbf{x}_t^*, \mathbf{x}_{i,t}) - \mathcal{D}_{\mathcal{R}}(\mathbf{x}_t^*, \mathbf{y}_{i,t+1}) - \mathcal{D}_{\mathcal{R}}(\mathbf{y}_{i,t+1}, \mathbf{x}_{i,t})\Big] \qquad \text{(S11)} \\ \leq \frac{1}{\alpha_{t+1}}\mathcal{D}_{\mathcal{R}}(\mathbf{x}_t^*, \mathbf{x}_{i,t}) &- \frac{1}{\alpha_{t+2}}\mathcal{D}_{\mathcal{R}}(\mathbf{x}_{t+1}^*, \mathbf{x}_{i,t+1}) + \frac{1}{\alpha_{t+2}}\mathcal{D}_{\mathcal{R}}(\mathbf{x}_{t+1}^*, \mathbf{x}_{i,t+1}) - \frac{1}{\alpha_{t+2}}\mathcal{D}_{\mathcal{R}}(\mathbf{x}_t^*, \mathbf{x}_{i,t+1}) \\ + \frac{1}{\alpha_{t+2}}\mathcal{D}_{\mathcal{R}}(\mathbf{x}_t^*, \mathbf{x}_{i,t+1}) &- \frac{1}{\alpha_{t+2}}\mathcal{D}_{\mathcal{R}}(\mathbf{x}_t^*, \mathbf{y}_{i,t+1}) + \frac{1}{\alpha_{t+2}}\mathcal{D}_{\mathcal{R}}(\mathbf{x}_t^*, \mathbf{y}_{i,t+1}) - \frac{1}{\alpha_{t+1}}\mathcal{D}_{\mathcal{R}}(\mathbf{x}_t^*, \mathbf{y}_{i,t+1}) \\ - \frac{\mu}{2\alpha_{t+1}}\|\mathbf{y}_{i,t+1} - \mathbf{x}_{i,t}\|^2, & \qquad \text{(S12)}\end{aligned}$$

where, (S11) follows from Lemma 3.3.1. (S12) follows from (3.12). Next, we analyze some of the component terms in (S12).

$$\frac{1}{\alpha_{t+2}}\left[\mathcal{D}_{\mathcal{R}}(\mathbf{x}_{t+1}^*, \mathbf{x}_{i,t+1}) - \mathcal{D}_{\mathcal{R}}(\mathbf{x}_t^*, \mathbf{x}_{i,t+1})\right] \leq \frac{K}{\alpha_{t+2}}\|\mathbf{x}_{t+1}^* - \mathbf{x}_t^*\|. \tag{S13}$$

which follows from assumption (C2). We use the upper bounds in (S10), (S12)-(S13) to further upper bound (S9). Next, we use (S8) and this new bound on (S9) to upper bound (S7). Then, summing (S7) over $i \in [n]$ and $t \in [T]$, we get

$$\begin{aligned}
&\sum_{t=1}^{T}\sum_{i=1}^{n}\left[f_{i,t}(\mathbf{x}_{i,t}) - f_{i,t}(\mathbf{x}_t^*)\right] \leq \frac{nG^2}{\mu}\sum_{t=1}^{T}\alpha_{t+1} + \sum_{t=1}^{T}\sum_{i=1}^{n}\frac{\mu}{4\alpha_{t+1}}\|\mathbf{y}_{i,t+1} - \mathbf{x}_{i,t}\|^2 \\
&+ \sum_{t=1}^{T}\sum_{i=1}^{n}\mathbf{q}_{i,t}^T\left[g_{i,t}(\mathbf{x}_t^*) - g_{i,t}(\mathbf{x}_{i,t})\right] + \sum_{t=1}^{T}\sum_{i=1}^{n}\mathbf{q}_{i,t}^T\nabla g_{i,t}(\mathbf{x}_{i,t})\left(\mathbf{x}_{i,t} - \mathbf{y}_{i,t+1}\right) \\
&+ \sum_{t=1}^{T}\sum_{i=1}^{n}\left[\frac{1}{\alpha_{t+1}}\mathcal{D}_{\mathcal{R}}(\mathbf{x}_t^*, \mathbf{x}_{i,t}) - \frac{1}{\alpha_{t+2}}\mathcal{D}_{\mathcal{R}}(\mathbf{x}_{t+1}^*, \mathbf{x}_{i,t+1})\right] \\
&+ \sum_{t=1}^{T}\frac{1}{\alpha_{t+2}}\left[\sum_{i=1}^{n}\mathcal{D}_{\mathcal{R}}(\mathbf{x}_t^*, \mathbf{x}_{i,t+1}) - \sum_{i=1}^{n}\mathcal{D}_{\mathcal{R}}(\mathbf{x}_t^*, \mathbf{y}_{i,t+1})\right] \\
&+ \sum_{t=1}^{T}\sum_{i=1}^{n}\frac{K}{\alpha_{t+2}}\|\mathbf{x}_{t+1}^* - \mathbf{x}_t^*\| + \sum_{t=1}^{T}\sum_{i=1}^{n}\left(\frac{1}{\alpha_{t+2}} - \frac{1}{\alpha_{t+1}}\right)\mathcal{D}_{\mathcal{R}}(\mathbf{x}_t^*, \mathbf{y}_{i,t+1}) \\
&- \sum_{t=1}^{T}\sum_{i=1}^{n}\frac{\mu}{2\alpha_{t+1}}\|\mathbf{y}_{i,t+1} - \mathbf{x}_{i,t}\|^2. \qquad \text{(S14)}
\end{aligned}$$

Here, by assumption, at the optima $\mathbf{x}_t^*$

$$\mathbf{q}_{i,t}^T g_{i,t}(\mathbf{x}_t^*) \leq 0, \quad \forall\, i, t. \qquad \text{(S15)}$$

Also,

$$\begin{aligned}
&\sum_{i=1}^{n}\mathcal{D}_{\mathcal{R}}(\mathbf{x}_t^*, \mathbf{x}_{i,t+1}) - \sum_{i=1}^{n}\mathcal{D}_{\mathcal{R}}(\mathbf{x}_t^*, \mathbf{y}_{i,t+1}) = \sum_{i=1}^{n}\Big[\mathcal{D}_{\mathcal{R}}\big(\mathbf{x}_t^*, \sum_{j=1}^{n}[\mathbf{W}_t]_{ij}\mathbf{y}_{j,t+1}\big) - \mathcal{D}_{\mathcal{R}}(\mathbf{x}_t^*, \mathbf{y}_{i,t+1})\Big] \\
&\leq \sum_{i=1}^{n}\Big[\sum_{j=1}^{n}[\mathbf{W}_t]_{ij}\mathcal{D}_{\mathcal{R}}\big(\mathbf{x}_t^*, \mathbf{y}_{j,t+1}\big) - \mathcal{D}_{\mathcal{R}}(\mathbf{x}_t^*, \mathbf{y}_{i,t+1})\Big] \qquad \text{(S16)} \\
&= \Big[\sum_{j=1}^{n}\mathcal{D}_{\mathcal{R}}\big(\mathbf{x}_t^*, \mathbf{y}_{j,t+1}\big) - \sum_{i=1}^{n}\mathcal{D}_{\mathcal{R}}(\mathbf{x}_t^*, \mathbf{y}_{i,t+1})\Big] = 0. \qquad \text{(S17)}
\end{aligned}$$

(S16) follows from step 12 in Algorithm 4, and assumption (C1). (S17) follows from assumption (A1) about the network. Finally,

$$\sum_{t=1}^{T}\sum_{i=1}^{n}\left(\frac{1}{\alpha_{t+2}} - \frac{1}{\alpha_{t+1}}\right)\mathcal{D}_{\mathcal{R}}(\mathbf{x}_t^*, \mathbf{y}_{i,t+1})$$

$$\leq \sum_{t=1}^{T} \left(\frac{1}{\alpha_{t+2}} - \frac{1}{\alpha_{t+1}} \right) nKd((X)) \leq \frac{nKd((X))}{\alpha_{T+2}}, \tag{S18}$$

where (S18) follows from (3.15), and using the fact that $\{\alpha_t\}$ is a non-increasing sequence.

Using (S15), (S17)-(S18) in (S14), we get (3.27). ■

A.2.3 Proof of Lemma 3.5.3

PROOF: We start with applying Lemma 3.3.1 to step 7 in Algorithm 4.

$$\begin{aligned} &\alpha_{t+1} \langle \mathbf{a}_{i,t+1}, \mathbf{y}_{i,t+1} - \mathbf{x}_{i,t} \rangle \leq -\mathcal{D}_{\mathcal{R}}(\mathbf{x}_{i,t}, \mathbf{y}_{i,t+1}) - \mathcal{D}_{\mathcal{R}}(\mathbf{y}_{i,t+1}, \mathbf{x}_{i,t}) \\ &\Rightarrow \mu \left\| \mathbf{y}_{i,t+1} - \mathbf{x}_{i,t} \right\|^2 \leq \alpha_{t+1} \left\| \mathbf{a}_{i,t+1} \right\| \left\| \mathbf{y}_{i,t+1} - \mathbf{x}_{i,t} \right\| \end{aligned} \tag{S19}$$

where (S19) follows from (3.12). Consequently,

$$\begin{aligned} &\left\| \mathbf{y}_{i,t+1} - \mathbf{x}_{i,t} \right\| \leq \frac{\alpha_{t+1}}{\mu} \left\| \mathbf{a}_{i,t+1} \right\| \leq \frac{\alpha_{t+1}}{\mu} \left(\left\| \nabla f_{i,t}(\mathbf{x}_{i,t}) \right\| + \left\| \nabla g_{i,t}(\mathbf{x}_{i,t}) \mathbf{q}_{i,t} \right\| \right) \\ &\leq \frac{\alpha_{t+1}}{\mu} \left(G + G \frac{F}{\beta_t} \right) \leq \frac{G\alpha_{t+1}}{\mu} \left(1 + \frac{F}{\beta_{t+1}} \right). \end{aligned} \tag{S20}$$

Note that the bound in (S20) is independent of i. Next, define $\mathbf{e}_{i,t} = \mathbf{y}_{i,t+1} - \mathbf{x}_{i,t}$. Note that,

$$\mathbf{y}_{i,t+1} = \mathbf{x}_{i,t} + \mathbf{e}_{i,t} = \sum_{j=1}^{n} [\mathbf{W}_t]_{ij} \mathbf{y}_{j,t} + \mathbf{e}_{i,t}, \tag{S21}$$

and

$$\bar{\mathbf{y}}_{t+1} = \bar{\mathbf{x}}_t + \bar{\mathbf{e}}_t = \frac{1}{n} \sum_{i=1}^{n} \sum_{j=1}^{n} [\mathbf{W}_t]_{ij} \mathbf{y}_{j,t} + \bar{\mathbf{e}}_t = \bar{\mathbf{y}}_t + \bar{\mathbf{e}}_t = \sum_{\tau=0}^{t} \bar{\mathbf{e}}_\tau. \tag{S22}$$

Since we assume $f_{i,0}(\cdot) \equiv 0, \forall\, i \in [n]$, $g_{i,0}(\cdot) = \mathbf{0}, \forall\, i \in [n]$ and $\mathbf{x}_{i,0} = \mathbf{0}$, from the algorithm, $\mathbf{y}_{i,1} = \mathbf{0}, \forall\, i \in [n]$. Further, defining

$$\mathbf{y}_t \triangleq \left[\mathbf{y}_{1,t}^T, \ldots, \mathbf{y}_{n,t}^T\right]^T, \ \mathbf{e}_t \triangleq \left[\mathbf{e}_{1,t}^T, \ldots, \mathbf{e}_{n,t}^T\right]^T \tag{S23}$$

we see from (S21) that $\mathbf{y}_{t+1} = (\mathbf{W} \otimes I)\mathbf{y}_t + (I \otimes I)\mathbf{e}_t$. Hence

$$\mathbf{y}_{i,t+1} = \sum_{\tau=0}^{t} \sum_{j=1}^{n} \left[\mathbf{W}^{t-\tau}\right]_{ij} \mathbf{e}_{j,\tau}. \tag{S24}$$

Substituting (S24) in (S21), and using it with (S22), we

$$\begin{aligned} \mathbf{x}_{i,t} - \bar{\mathbf{x}}_t &= \sum_{j=1}^{n} [\mathbf{W}]_{ij} \mathbf{y}_{j,t} - \bar{\mathbf{y}}_t \\ &= \sum_{j=1}^{n} \left([\mathbf{W}]_{ij} - \frac{1}{n} \right) \sum_{\tau=0}^{t-1} \sum_{k=1}^{n} \left[\mathbf{W}^{t-1-\tau}\right]_{jk} \mathbf{e}_{k,\tau} = \sum_{\tau=0}^{t-1} \sum_{k=1}^{n} \left(\left[\mathbf{W}^{t-\tau}\right]_{ik} - \frac{1}{n} \right) \mathbf{e}_{k,\tau} \end{aligned} \tag{S25}$$

Consequently,

$$\|\mathbf{x}_{i,t} - \bar{\mathbf{x}}_t\| \le \sum_{\tau=0}^{t-1} \sum_{k=1}^{n} \left| \left[\mathbf{W}^{t-\tau}\right]_{ik} - \frac{1}{n} \right| \|\mathbf{e}_{k,\tau}\|, \tag{S26}$$

A standard property of doubly-stochastic matrices is that

$$\left| \left[\mathbf{W}^{t}\right]_{ik} - \frac{1}{n} \right| \le \sqrt{n} \sigma_2^t(\mathbf{W}). \tag{S27}$$

Substituting the bound on $\|\mathbf{e}_{k,\tau}\|$ from (S20), and (S27) we get (3.28). ■

A.3 PR-SPIDER: Convergence of the Finite-sum case

A.3.1 Proof of Lemma 4.2.2

PROOF: We have

$$\begin{aligned}
&\mathbb{E}\left\|\bar{\mathbf{v}}_t^{s+1}-\frac{1}{N}\sum_{i=1}^{N}\nabla f_i\left(\mathbf{x}_{i,t}^{s+1}\right)\right\|^2\\
&=\mathbb{E}\Bigg\|\bar{\mathbf{v}}_{t-1}^{s+1}-\frac{1}{N}\sum_{i=1}^{N}\nabla f_i\left(\mathbf{x}_{i,t-1}^{s+1}\right)+\frac{1}{N}\sum_{j=1}^{N}\frac{1}{B}\sum_{\xi_{j,t}^{s+1}}\left[\nabla f_j\big(\mathbf{x}_{j,t}^{s+1};\xi_{j,t}^{s+1}\big)-\nabla f_j\big(\mathbf{x}_{j,t-1}^{s+1};\xi_{j,t}^{s+1}\big)\right]\\
&\qquad+\frac{1}{N}\sum_{i=1}^{N}\left[\nabla f_i\left(\mathbf{x}_{i,t-1}^{s+1}\right)-\nabla f_i\left(\mathbf{x}_{i,t}^{s+1}\right)\right]\Bigg\|^2 \qquad \text{(S28)}\\
&=\mathbb{E}\left\|\bar{\mathbf{v}}_{t-1}^{s+1}-\frac{1}{N}\sum_{i=1}^{N}\nabla f_i\left(\mathbf{x}_{i,t-1}^{s+1}\right)\right\|^2\\
&\quad+\mathbb{E}\Bigg\|\frac{1}{N}\sum_{i=1}^{N}\frac{1}{B}\sum_{\xi_{i,t}^{s+1}}\left[\nabla f_i\big(\mathbf{x}_{i,t}^{s+1};\xi_{i,t}^{s+1}\big)-\nabla f_i\big(\mathbf{x}_{i,t-1}^{s+1};\xi_{i,t}^{s+1}\big)+\nabla f_i\left(\mathbf{x}_{i,t-1}^{s+1}\right)-\nabla f_i\left(\mathbf{x}_{i,t}^{s+1}\right)\right]\Bigg\|^2\\
&\quad+\mathbb{E}\Bigg\langle\underbrace{\frac{1}{N}\sum_{i=1}^{N}\frac{1}{B}\sum_{\xi_{i,t}^{s+1}}\left[\nabla f_i\big(\mathbf{x}_{i,t}^{s+1};\xi_{i,t}^{s+1}\big)-\nabla f_i\big(\mathbf{x}_{i,t-1}^{s+1};\xi_{i,t}^{s+1}\big)+\nabla f_i\left(\mathbf{x}_{i,t-1}^{s+1}\right)-\nabla f_i\left(\mathbf{x}_{i,t}^{s+1}\right)\right]}_{\mathbb{E}(\cdot)=\mathbf{0}}\\
&\qquad\qquad,\bar{\mathbf{v}}_{t-1}^{s+1}-\frac{1}{N}\sum_{i=1}^{N}\nabla f_i\left(\mathbf{x}_{i,t-1}^{s+1}\right)\Bigg\rangle \qquad \text{(S29)}\\
&=\mathbb{E}\left\|\bar{\mathbf{v}}_{t-1}^{s+1}-\frac{1}{N}\sum_{i=1}^{N}\nabla f_i\left(\mathbf{x}_{i,t-1}^{s+1}\right)\right\|^2\\
&\quad+\frac{1}{N^2}\sum_{i=1}^{N}\mathbb{E}\Bigg\|\frac{1}{B}\sum_{\xi_{i,t}^{s+1}}\left[\nabla f_i\big(\mathbf{x}_{i,t}^{s+1};\xi_{i,t}^{s+1}\big)-\nabla f_i\big(\mathbf{x}_{i,t-1}^{s+1};\xi_{i,t}^{s+1}\big)+\nabla f_i\left(\mathbf{x}_{i,t-1}^{s+1}\right)-\nabla f_i\left(\mathbf{x}_{i,t}^{s+1}\right)\right]\Bigg\|^2\\
&\quad+\frac{1}{N^2}\sum_{i\neq j}\mathbb{E}\Bigg\langle\frac{1}{B}\sum_{\xi_{i,t}^{s+1}}\left[\nabla f_i\big(\mathbf{x}_{i,t}^{s+1};\xi_{i,t}^{s+1}\big)-\nabla f_i\big(\mathbf{x}_{i,t-1}^{s+1};\xi_{i,t}^{s+1}\big)+\nabla f_i\left(\mathbf{x}_{i,t-1}^{s+1}\right)-\nabla f_i\left(\mathbf{x}_{i,t}^{s+1}\right)\right],\\
&\qquad\frac{1}{B}\sum_{\xi_{j,t}^{s+1}}\left[\nabla f_j\big(\mathbf{x}_{j,t}^{s+1};\xi_{j,t}^{s+1}\big)-\nabla f_j\big(\mathbf{x}_{j,t-1}^{s+1};\xi_{j,t}^{s+1}\big)+\nabla f_j\left(\mathbf{x}_{j,t-1}^{s+1}\right)-\nabla f_j\left(\mathbf{x}_{j,t}^{s+1}\right)\right]\Bigg\rangle \qquad \text{(S30)}\\
&=\mathbb{E}\left\|\bar{\mathbf{v}}_{t-1}^{s+1}-\frac{1}{N}\sum_{i=1}^{N}\nabla f_i\left(\mathbf{x}_{i,t-1}^{s+1}\right)\right\|^2
\end{aligned}$$

$$+\frac{1}{N^2}\sum_{i=1}^{N}\frac{1}{B^2}\mathbb{E}\sum_{\xi_{i,t}^{s+1}}\left\|\nabla f_i\big(\mathbf{x}_{i,t}^{s+1};\xi_{i,t}^{s+1}\big)-\nabla f_i\big(\mathbf{x}_{i,t-1}^{s+1};\xi_{i,t}^{s+1}\big)+\nabla f_i\left(\mathbf{x}_{i,t-1}^{s+1}\right)-\nabla f_i\left(\mathbf{x}_{i,t}^{s+1}\right)\right\|^2$$

$$+\frac{1}{N^2}\sum_{i=1}^{N}\frac{1}{B^2}\mathbb{E}\sum_{\xi_{i,t}^{s+1}\neq\zeta_{i,t}^{s+1}}\Big\langle\underbrace{\mathbb{E}_{\xi_{i,t}^{s+1}}\left[\nabla f_i\big(\mathbf{x}_{i,t}^{s+1};\xi_{i,t}^{s+1}\big)-\nabla f_i\big(\mathbf{x}_{i,t-1}^{s+1};\xi_{i,t}^{s+1}\big)\right]+\nabla f_i\left(\mathbf{x}_{i,t-1}^{s+1}\right)-\nabla f_i\left(\mathbf{x}_{i,t}^{s+1}\right)}_{=\mathbf{0}},$$

$$\underbrace{\mathbb{E}_{\zeta_{i,t}^{s+1}}\left[\nabla f_i\big(\mathbf{x}_{i,t}^{s+1};\zeta_{i,t}^{s+1}\big)-\nabla f_i\big(\mathbf{x}_{i,t-1}^{s+1};\zeta_{i,t}^{s+1}\big)\right]+\nabla f_i\left(\mathbf{x}_{i,t-1}^{s+1}\right)-\nabla f_i\left(\mathbf{x}_{i,t}^{s+1}\right)}_{=\mathbf{0}}\Big\rangle \tag{S31}$$

$$\leq\mathbb{E}\left\|\bar{\mathbf{v}}_{t-1}^{s+1}-\frac{1}{N}\sum_{i=1}^{N}\nabla f_i\left(\mathbf{x}_{i,t-1}^{s+1}\right)\right\|^2+\frac{1}{(NB)^2}\sum_{i=1}^{N}\mathbb{E}\sum_{\xi_{i,t}^{s+1}}\left\|\nabla f_i\big(\mathbf{x}_{i,t}^{s+1};\xi_{i,t}^{s+1}\big)-\nabla f_i\big(\mathbf{x}_{i,t-1}^{s+1};\xi_{i,t}^{s+1}\big)\right\|^2 \tag{S32}$$

$$\leq\mathbb{E}\left\|\bar{\mathbf{v}}_{t-1}^{s+1}-\frac{1}{N}\sum_{i=1}^{N}\nabla f_i\left(\mathbf{x}_{i,t-1}^{s+1}\right)\right\|^2+\frac{1}{N^2}\sum_{i=1}^{N}\frac{L^2}{B}\mathbb{E}\left\|\mathbf{x}_{i,t}^{s+1}-\mathbf{x}_{i,t-1}^{s+1}\right\|^2 \tag{S33}$$

$$\leq\underbrace{\mathbb{E}\left\|\bar{\mathbf{v}}_{0}^{s+1}-\frac{1}{N}\sum_{i=1}^{N}\nabla f_i\left(\mathbf{x}_{i,0}^{s+1}\right)\right\|^2}_{E_0^{s+1}}+\frac{L^2}{N^2B}\sum_{i=1}^{N}\sum_{\ell=0}^{t-1}\mathbb{E}\left\|\mathbf{x}_{i,\ell+1}^{s+1}-\mathbf{x}_{i,\ell}^{s+1}\right\|^2 \tag{S34}$$

(S28) follows from step 7 in Algorithm 5. The cross term in (S29) is zero since

$$\mathbb{E}_{\xi_{i,t}^{s+1}}\left[\nabla f_i\big(\mathbf{x}_{i,t}^{s+1};\xi_{i,t}^{s+1}\big)-\nabla f_i\big(\mathbf{x}_{i,t-1}^{s+1};\xi_{i,t}^{s+1}\big)\right]=\nabla f_i\left(\mathbf{x}_{i,t}^{s+1}\right)-\nabla f_i\left(\mathbf{x}_{i,t-1}^{s+1}\right), \tag{S35}$$

for all $\xi_{i,t}^{s+1}\in\mathcal{B}_{i,t}^{s+1}$, $\forall\ t,\forall\ s,\forall\ i\in[1:N]$. The cross term in (S31) is zero since at a single node i, the samples in the mini-batch $\mathcal{B}_{i,t}^{s+1}$ are sampled independent of each other. (S32) follows from (S35) and using $\mathbb{E}\left\|\mathbf{x}-\mathbb{E}(\mathbf{x})\right\|^2\leq\mathbb{E}\left\|\mathbf{x}\right\|^2$. (S33) follows from mean-squared L-smooth property of each stochastic function $f_i(\cdot,\xi)$. (S34) follows by recursive application of (S33), to the beginning of the epoch. ■

A.3.2 Proof of Lemma 4.2.3

PROOF: First we prove (4.12). For ℓ such that $\ell \bmod I \neq 0$

$$
\begin{aligned}
&\sum_{i=1}^{N}\mathbb{E}\left\|\mathbf{v}_{i,\ell}^{s+1}-\bar{\mathbf{v}}_{\ell}^{s+1}\right\|^2\\
&\le \sum_{i=1}^{N}\Bigg[(1+\delta)\,\mathbb{E}\left\|\mathbf{v}_{i,\ell-1}^{s+1}-\bar{\mathbf{v}}_{\ell-1}^{s+1}\right\|^2\\
&\qquad+\left(1+\frac{1}{\delta}\right)\mathbb{E}\Bigg\|\frac{1}{B}\sum_{\xi_{i,\ell}^{s+1}}\left\{\nabla f_i\big(\mathbf{x}_{i,\ell}^{s+1};\xi_{i,\ell}^{s+1}\big)-\nabla f_i\big(\mathbf{x}_{i,\ell-1}^{s+1};\xi_{i,\ell}^{s+1}\big)\right\}\\
&\qquad\qquad-\frac{1}{N}\sum_{j=1}^{N}\frac{1}{B}\sum_{\xi_{j,\ell}^{s+1}}\left\{\nabla f_j\big(\mathbf{x}_{j,\ell}^{s+1};\xi_{j,\ell}^{s+1}\big)-\nabla f_j\big(\mathbf{x}_{j,\ell-1}^{s+1};\xi_{j,\ell}^{s+1}\big)\right\}\Bigg\|^2\Bigg]
\end{aligned}
\tag{S36}
$$

$$
\begin{aligned}
&\le \sum_{i=1}^{N}\Bigg[(1+\delta)\,\mathbb{E}\left\|\mathbf{v}_{i,\ell-1}^{s+1}-\bar{\mathbf{v}}_{\ell-1}^{s+1}\right\|^2\\
&\qquad+\left(1+\frac{1}{\delta}\right)\frac{2}{B}\mathbb{E}\sum_{\xi_{i,\ell}^{s+1}}\left\|\nabla f_i\big(\mathbf{x}_{i,\ell}^{s+1};\xi_{i,\ell}^{s+1}\big)-\nabla f_i\big(\mathbf{x}_{i,\ell-1}^{s+1};\xi_{i,\ell}^{s+1}\big)\right\|^2\\
&\qquad+\left(1+\frac{1}{\delta}\right)\frac{2}{N}\sum_{j=1}^{N}\frac{1}{B}\mathbb{E}\sum_{\xi_{j,\ell}^{s+1}}\left\|\nabla f_j\big(\mathbf{x}_{j,\ell}^{s+1};\xi_{j,\ell}^{s+1}\big)-\nabla f_j\big(\mathbf{x}_{j,\ell-1}^{s+1};\xi_{j,\ell}^{s+1}\big)\right\|^2\Bigg]
\end{aligned}
\tag{S37}
$$

$$
\begin{aligned}
&\le \sum_{i=1}^{N}\Bigg[(1+\delta)\,\mathbb{E}\left\|\mathbf{v}_{i,\ell-1}^{s+1}-\bar{\mathbf{v}}_{\ell-1}^{s+1}\right\|^2+\left(1+\frac{1}{\delta}\right)2L^2\,\mathbb{E}\left\|\mathbf{x}_{i,\ell}^{s+1}-\mathbf{x}_{i,\ell-1}^{s+1}\right\|^2\\
&\qquad+\left(1+\frac{1}{\delta}\right)\frac{2L^2}{N}\sum_{j=1}^{N}\mathbb{E}\left\|\mathbf{x}_{j,\ell}^{s+1}-\mathbf{x}_{j,\ell-1}^{s+1}\right\|^2\Bigg]
\end{aligned}
\tag{S38}
$$

$$
\le \sum_{i=1}^{N}\Bigg[(1+\delta)\,\mathbb{E}\left\|\mathbf{v}_{i,\ell-1}^{s+1}-\bar{\mathbf{v}}_{\ell-1}^{s+1}\right\|^2+8\gamma^2L^2\left(1+\frac{1}{\delta}\right)\left\{\mathbb{E}\left\|\mathbf{v}_{i,\ell-1}^{s+1}-\bar{\mathbf{v}}_{\ell-1}^{s+1}\right\|^2+\mathbb{E}\left\|\bar{\mathbf{v}}_{\ell-1}^{s+1}\right\|^2\right\}\Bigg]
$$

$$
\le \sum_{i=1}^{N}\Big(1+\underbrace{\delta+8\gamma^2L^2\big(1+\frac{1}{\delta}\big)}_{\theta}\Big)\mathbb{E}\left\|\mathbf{v}_{i,\ell-1}^{s+1}-\bar{\mathbf{v}}_{\ell-1}^{s+1}\right\|^2+8\gamma^2NL^2\big(1+\frac{1}{\delta}\big)\mathbb{E}\left\|\bar{\mathbf{v}}_{\ell-1}^{s+1}\right\|^2
\tag{S39}
$$

where, (S36) follows from step 7 in Algorithm 5 and Young's inequality, for $\delta > 0$; (S37) follows from Jensen's inequality; (S38) follows from the mean-squared L-smooth assumption (A1). Applying (S39) recursively, we obtain

$$
\begin{aligned}
&\sum_{i=1}^{N} \mathbb{E}\left\|\mathbf{v}_{i,\ell}^{s+1} - \bar{\mathbf{v}}_{\ell}^{s+1}\right\|^2 \\
&\leq \sum_{i=1}^{N} (1+\theta)^2 \mathbb{E}\left\|\mathbf{v}_{i,\ell-2}^{s+1} - \bar{\mathbf{v}}_{\ell-2}^{s+1}\right\|^2 + 8\gamma^2 N L^2 \big(1 + \frac{1}{\delta}\big) \Big[\mathbb{E}\left\|\bar{\mathbf{v}}_{\ell-1}^{s+1}\right\|^2 + (1+\theta)\mathbb{E}\left\|\bar{\mathbf{v}}_{\ell-2}^{s+1}\right\|^2\Big] \\
&\leq \sum_{i=1}^{N} (1+\theta)^{\ell-\tau(\ell)} \mathbb{E}\left\|\mathbf{v}_{i,\tau(\ell)}^{s+1} - \bar{\mathbf{v}}_{\tau(\ell)}^{s+1}\right\|^2 + 8\gamma^2 N L^2 \big(1 + \frac{1}{\delta}\big) \sum_{j=\tau(\ell)}^{\ell-1} (1+\theta)^{\ell-1-j} \mathbb{E}\left\|\bar{\mathbf{v}}_{j}^{s+1}\right\|^2 \\
&= 8\gamma^2 N L^2 \big(1 + \frac{1}{\delta}\big) \sum_{j=\tau(\ell)}^{\ell-1} (1+\theta)^{\ell-1-j} \mathbb{E}\left\|\bar{\mathbf{v}}_{j}^{s+1}\right\|^2 \qquad \text{(S40)}
\end{aligned}
$$

where (S40) follows since averaging happens at time index $\tau(\ell)$ (step 10, Algorithm 5), i.e., $\mathbf{v}_{i,\tau(\ell)}^{s+1} = \bar{\mathbf{v}}_{\tau(\ell)}^{s+1}, \forall\, i \in [1:N]$.

Next, we prove (4.13). For ℓ such that $\ell \bmod I \neq 0$

$$
\begin{aligned}
&\sum_{i=1}^{N} \mathbb{E}\left\|\mathbf{x}_{i,\ell}^{s+1} - \bar{\mathbf{x}}_{\ell}^{s+1}\right\|^2 = \sum_{i=1}^{N} \mathbb{E}\left\|\big[\mathbf{x}_{i,\ell-1}^{s+1} - \gamma \mathbf{v}_{i,\ell-1}^{s+1}\big] - \big[\bar{\mathbf{x}}_{\ell-1}^{s+1} - \gamma \bar{\mathbf{v}}_{t-1}^{s+1}\big]\right\|^2 \\
&\leq \sum_{i=1}^{N} \left[(1+\alpha)\mathbb{E}\left\|\mathbf{x}_{i,\ell-1}^{s+1} - \bar{\mathbf{x}}_{\ell-1}^{s+1}\right\|^2 + \left(1 + \frac{1}{\alpha}\right)\gamma^2 \mathbb{E}\left\|\mathbf{v}_{i,\ell-1}^{s+1} - \bar{\mathbf{v}}_{\ell-1}^{s+1}\right\|^2\right] \qquad \text{(S41)} \\
&\leq \sum_{i=1}^{N} \left[(1+\alpha)^{\ell-\tau(\ell)}\mathbb{E}\left\|\mathbf{x}_{i,\tau(\ell)}^{s+1} - \bar{\mathbf{x}}_{\tau(\ell)}^{s+1}\right\|^2 + \left(1 + \frac{1}{\alpha}\right)\gamma^2 \sum_{j=\tau(\ell)}^{\ell-1} (1+\alpha)^{\ell-1-j} \mathbb{E}\left\|\mathbf{v}_{i,j}^{s+1} - \bar{\mathbf{v}}_{j}^{s+1}\right\|^2\right] \\
&= \left(1 + \frac{1}{\alpha}\right)\gamma^2 \sum_{i=1}^{N} \sum_{j=\tau(\ell)+1}^{\ell-1} (1+\alpha)^{\ell-1-j} \mathbb{E}\left\|\mathbf{v}_{i,j}^{s+1} - \bar{\mathbf{v}}_{j}^{s+1}\right\|^2 \qquad \text{(S42)}
\end{aligned}
$$

where (S41) follows from Young's inequality, with $\alpha > 0$; (S42) follows since averaging happens at time index $\tau(\ell)$ (step 9-10, Algorithm 5). Consequently $\mathbf{x}_{i,\tau(\ell)}^{s+1} = \bar{\mathbf{x}}_{\tau(\ell)}^{s+1}, \mathbf{v}_{i,\tau(\ell)}^{s+1} = \bar{\mathbf{v}}_{\tau(\ell)}^{s+1}, \forall\, i \in [1:N]$. We can further upper bound (S42) using the bound on $\sum_{i=1}^{N} \mathbb{E}\|\mathbf{v}_{i,j}^{s+1} - \bar{\mathbf{v}}_{j}^{s+1}\|^2$ in (S40). ■

A.3.3 Intermediate results for Lemma 4.2.4

In the following analysis, the following facts shall be utilized repeatedly.

(F1) We choose δ, γ such that $\delta < \theta = \delta + 8\gamma^2 L^2 \big(1 + \frac{1}{\delta}\big) < 2\delta < 1$.

(F2) We choose $\alpha = \frac{\delta}{2}$. Therefore, $\theta - \alpha > \frac{\delta}{2}$.

(F3) $\left(1 + \frac{1}{\delta}\right) < \frac{2}{\delta}$ and $\left(1 + \frac{1}{\alpha}\right) < \frac{2}{\alpha}$.

(F4) $I = \left\lfloor \frac{1}{\theta} \right\rfloor \leq \frac{1}{\theta}$. Also, for $\theta < 1$, $\left\lfloor \frac{1}{\theta} \right\rfloor \geq \frac{1}{2\theta} > \frac{1}{4\delta}$.

Now, we state first of the three lemmas we will require to prove Lemma 4.2.4.

Lemma A.3.1.

$$\frac{16\gamma^5 L^4}{NB}\big(1 + \frac{1}{\delta}\big) \sum_{t=0}^{m-1} \sum_{\ell=0}^{t-1} \sum_{j=\tau(\ell)}^{\ell-1} (1+\theta)^{\ell-1-j} \mathbb{E}\left\|\bar{\mathbf{v}}_j^{s+1}\right\|^2 \leq \frac{64\gamma^5 L^4 m}{NB\delta^2} \sum_{t=0}^{m-1} \mathbb{E}\left\|\bar{\mathbf{v}}_t^{s+1}\right\|^2.$$

PROOF: From the left hand side of the inequality, we have

$$\begin{aligned}
&\frac{16\gamma^5 L^4}{NB}\big(1 + \frac{1}{\delta}\big) \sum_{t=0}^{m-1} \sum_{\ell=0}^{t-1} \sum_{j=\tau(\ell)}^{\ell-1} (1+\theta)^{\ell-1-j} \mathbb{E}\left\|\bar{\mathbf{v}}_j^{s+1}\right\|^2 \\
&= \frac{16\gamma^5 L^4}{NB}\big(1 + \frac{1}{\delta}\big) \sum_{t=0}^{m-1} \Bigg[\sum_{\ell=0}^{I-1} \sum_{j=0}^{\ell-1} (1+\theta)^{\ell-1-j} \mathbb{E}\left\|\bar{\mathbf{v}}_j^{s+1}\right\|^2 \\
&\quad + \sum_{\ell=I}^{2I-1} \sum_{j=I}^{\ell-1} (1+\theta)^{\ell-1-j} \mathbb{E}\left\|\bar{\mathbf{v}}_j^{s+1}\right\|^2 \ldots + \sum_{\ell=\tau(t)}^{t-1} \sum_{j=\tau(t)}^{\ell-1} (1+\theta)^{\ell-1-j} \mathbb{E}\left\|\bar{\mathbf{v}}_j^{s+1}\right\|^2 \Bigg] && \text{(S43)} \\
&= \frac{16\gamma^5 L^4}{NB}\big(1 + \frac{1}{\delta}\big) \sum_{t=0}^{m-1} \Bigg[\sum_{\ell=0}^{I-1} \sum_{j=0}^{\ell-1} (1+\theta)^{\ell-1-j} \mathbb{E}\left\|\bar{\mathbf{v}}_j^{s+1}\right\|^2 \\
&\quad + \sum_{\ell=0}^{I-1} \sum_{j=0}^{\ell-1} (1+\theta)^{\ell-1-j} \mathbb{E}\left\|\bar{\mathbf{v}}_{j+I}^{s+1}\right\|^2 \ldots + \sum_{\ell=0}^{t-1-\tau(t)} \sum_{j=0}^{\ell-1} (1+\theta)^{\ell-1-j} \mathbb{E}\left\|\bar{\mathbf{v}}_{j+\tau(t)}^{s+1}\right\|^2 \Bigg] && \text{(S44)} \\
&\leq \frac{16\gamma^5 L^4}{NB}\big(1 + \frac{1}{\delta}\big) \left[\sum_{t=0}^{m-1} \sum_{\ell=0}^{I-1} (1+\theta)^{\ell-1} \right] \sum_{j=0}^{m-1} \mathbb{E}\left\|\bar{\mathbf{v}}_j^{s+1}\right\|^2 && \text{(S45)} \\
&\leq \frac{16\gamma^5 L^4}{NB}\big(1 + \frac{1}{\delta}\big) \sum_{t=0}^{m-1} \mathbb{E}\left\|\bar{\mathbf{v}}_t^{s+1}\right\|^2 m \left[\frac{(1+\theta)^I - 1}{\theta} \right] \\
&\leq \frac{16\gamma^5 L^4 m}{NB} \frac{2}{\delta} \frac{e-1}{\theta} \sum_{t=0}^{m-1} \mathbb{E}\left\|\bar{\mathbf{v}}_t^{s+1}\right\|^2 \\
&\overset{(c)}{\leq} \frac{64\gamma^5 L^4 m}{NB\delta^2} \sum_{t=0}^{m-1} \mathbb{E}\left\|\bar{\mathbf{v}}_t^{s+1}\right\|^2 && \text{(S46)}
\end{aligned}$$

where, (S44) follows from (S43) by simple re-indexing. (S45) follows from (S44) by upper bounding the coefficients of all terms with the one corresponding to $j = 0$. Note that $(1+\frac{1}{x})^x$ in increasing function for $x > 0$ and $\lim_{x\to\infty}(1 + \frac{1}{x})^x = e$ (Euler's constant). (S46) follows from (F1)-(F3). ■

Next, we present the second lemma we will use to prove Lemma 4.2.4.

Lemma A.3.2.

$$\frac{2\gamma^3 L^2}{NB}\sum_{t=0}^{m-1}\sum_{\ell=0}^{t-1}\mathbb{E}\left\|\bar{\mathbf{v}}_\ell^{s+1}\right\|^2 \le \frac{2\gamma^3 L^2 m}{NB}\sum_{t=0}^{m-1}\mathbb{E}\left\|\bar{\mathbf{v}}_t^{s+1}\right\|^2.$$

PROOF: We have from the left hand side of the inequality

$$\begin{aligned}
&\frac{2\gamma^3 L^2}{NB}\sum_{t=0}^{m-1}\sum_{\ell=0}^{t-1}\mathbb{E}\left\|\bar{\mathbf{v}}_\ell^{s+1}\right\|^2\\
&= \frac{2\gamma^3 L^2}{NB}\left[\mathbb{E}\left\|\bar{\mathbf{v}}_0^{s+1}\right\|^2(m-1) + \mathbb{E}\left\|\bar{\mathbf{v}}_1^{s+1}\right\|^2(m-2) + \ldots + \mathbb{E}\left\|\bar{\mathbf{v}}_{m-2}^{s+1}\right\|^2(1)\right]\\
&\le \frac{2\gamma^3 L^2 m}{NB}\sum_{t=0}^{m-1}\mathbb{E}\left\|\bar{\mathbf{v}}_t^{s+1}\right\|^2.
\end{aligned}$$

■

Finally, we present the third intermediate lemma before presenting the proof of Lemma 4.2.4.

Lemma A.3.3.

$$\begin{aligned}
8\gamma^5 L^4\left(1+\frac{1}{\alpha}\right)(1+\frac{1}{\delta})\sum_{t=0}^{m-1}\sum_{\ell=\tau(t)+1}^{t-1}\sum_{j=\tau(\ell)}^{\ell-1}(1+\alpha)^{t-1-\ell}(1+\theta)^{\ell-1-j}\mathbb{E}\left\|\bar{\mathbf{v}}_j^{s+1}\right\|^2\\
\le \frac{256\gamma^5 L^4}{\delta^4}\sum_{t=0}^{m-1}\mathbb{E}\left\|\bar{\mathbf{v}}_t^{s+1}\right\|^2.
\end{aligned}$$

PROOF: Suppose $\tau(m-1) = (p-1)I$ (see (4.11)), we have

$$\begin{aligned}
&8\gamma^5 L^4\left(1+\frac{1}{\alpha}\right)(1+\frac{1}{\delta})\sum_{t=0}^{m-1}\sum_{\ell=\tau(t)+1}^{t-1}\sum_{j=\tau(\ell)}^{\ell-1}(1+\alpha)^{t-1-\ell}(1+\theta)^{\ell-1-j}\mathbb{E}\left\|\bar{\mathbf{v}}_j^{s+1}\right\|^2\\
&= 8\gamma^5 L^4\left(1+\frac{1}{\alpha}\right)\left(1+\frac{1}{\delta}\right)\left[\sum_{t=0}^{I-1}\sum_{\ell=1}^{t-1}\sum_{j=0}^{\ell-1}(1+\alpha)^{t-1-\ell}(1+\theta)^{\ell-1-j}\mathbb{E}\left\|\bar{\mathbf{v}}_j^{s+1}\right\|^2\right.
\end{aligned}$$

$$+\sum_{t=I}^{2I-1}\sum_{\ell=I+1}^{t-1}\sum_{j=I}^{\ell-1}(1+\alpha)^{t-1-\ell}(1+\theta)^{\ell-1-j}\mathbb{E}\left\|\bar{\mathbf{v}}_j^{s+1}\right\|^2\dots$$

$$\left.+\sum_{t=(p-1)I}^{m-1}\;\sum_{\ell=(p-1)I+1}^{t-1}\;\sum_{j=(p-1)I}^{\ell-1}(1+\alpha)^{t-1-\ell}(1+\theta)^{\ell-1-j}\mathbb{E}\left\|\bar{\mathbf{v}}_j^{s+1}\right\|^2\right] \tag{S47}$$

$$=8\gamma^5L^4\left(1+\frac{1}{\alpha}\right)\left(1+\frac{1}{\delta}\right)\left[\sum_{t=0}^{I-1}\sum_{\ell=1}^{t-1}\sum_{j=0}^{\ell-1}(1+\alpha)^{t-1-\ell}(1+\theta)^{\ell-1-j}\mathbb{E}\left\|\bar{\mathbf{v}}_j^{s+1}\right\|^2\right.$$

$$+\sum_{t=0}^{I-1}\sum_{\ell=1}^{t-1}\sum_{j=0}^{\ell-1}(1+\alpha)^{t-1-\ell}(1+\theta)^{\ell-1-j}\mathbb{E}\left\|\bar{\mathbf{v}}_{j+I}^{s+1}\right\|^2\dots$$

$$\left.+\sum_{t=0}^{m-1-(p-1)I}\;\sum_{\ell=1}^{t-1}\sum_{j=0}^{\ell-1}(1+\alpha)^{t-1-\ell}(1+\theta)^{\ell-1-j}\mathbb{E}\left\|\bar{\mathbf{v}}_{j+(p-1)I}^{s+1}\right\|^2\right]$$

$$\leq 8\gamma^5L^4\frac{2}{\alpha}\frac{2}{\delta}\sum_{j=0}^{m-1}\mathbb{E}\left\|\bar{\mathbf{v}}_j^{s+1}\right\|^2\left[\sum_{t=0}^{I-1}\sum_{\ell=0}^{t-1}(1+\alpha)^{t-1-\ell}(1+\theta)^{\ell-1}\right] \tag{S48}$$

$$=\frac{32\gamma^5L^4}{\alpha\delta}\sum_{j=0}^{m-1}\mathbb{E}\left\|\bar{\mathbf{v}}_j^{s+1}\right\|^2\left[\sum_{t=0}^{I-1}\frac{(1+\alpha)^{t-1}}{(1+\theta)}\frac{\left(\frac{1+\theta}{1+\alpha}\right)^t-1}{\left(\frac{1+\theta}{1+\alpha}\right)-1}\right]$$

$$\leq\frac{32\gamma^5L^4}{\alpha\delta}\sum_{j=0}^{m-1}\mathbb{E}\left\|\bar{\mathbf{v}}_j^{s+1}\right\|^2\left[\sum_{t=0}^{I-1}\frac{(1+\theta)^t-(1+\alpha)^t}{\theta-\alpha}\right]$$

$$\leq\frac{32\gamma^5L^4}{\alpha\delta}\sum_{t=0}^{m-1}\mathbb{E}\left\|\bar{\mathbf{v}}_t^{s+1}\right\|^2\left[\frac{(1+\theta)^I-1}{\theta(\theta-\alpha)}\right]$$

$$\leq\frac{32\gamma^5L^4}{(\delta/2)\delta}\sum_{t=0}^{m-1}\mathbb{E}\left\|\bar{\mathbf{v}}_t^{s+1}\right\|^2\left[\frac{e-1}{\delta(\delta/2)}\right] \tag{S49}$$

$$=\frac{256\gamma^5L^4}{\delta^4}\sum_{t=0}^{m-1}\mathbb{E}\left\|\bar{\mathbf{v}}_t^{s+1}\right\|^2. \tag{S50}$$

In (S47), we split the summation over t into blocks of length I. For any block $\sum_{t=(c)I}^{(c+1)I-1}$, $\tau(t)=cI$. Therefore, ℓ varies from $\ell=[cI+1:t-1]$. Further, over this range of ℓ, $\tau(\ell)=cI$. Note that in this block, the largest coefficient corresponds to the smallest j, i.e., $\mathbb{E}\|\bar{\mathbf{v}}_{cI}^{s+1}\|^2$. We use these upper bounds on coefficients in (S48). We also use (F3). (S49) follows from (F1), (F2), (F4). ■

A.3.4 Choice of γ

Suppose γ be selected such that

$$L\gamma \leq \frac{1}{8} \ \wedge \ \frac{128\gamma^4 L^4 m}{NB\delta^2} \leq \frac{1}{8} \ \wedge \ \frac{4\gamma^2 L^2 m}{NB} \leq \frac{1}{8} \ \wedge \ \frac{512\gamma^4 L^4}{\delta^4} \leq \frac{1}{8} \tag{S51}$$

As we shall see in the proof of Corollary 1, the optimal choices $m = I\sqrt{Nn}, B = \frac{1}{I}\sqrt{\frac{n}{N}}$. Substituting these values in one of the inequalities above, we get

$$\begin{aligned} \frac{128\gamma^4 L^4 I\sqrt{Nn}}{N\frac{1}{I}\sqrt{\frac{n}{N}}} \leq \frac{\delta^2}{8} \quad &\Rightarrow \quad 128\gamma^4 L^4 I^2 \leq \frac{1}{8I^2} \qquad \because \quad \delta < \frac{1}{I} \text{ (F4)} \\ &\Rightarrow \quad \gamma \leq \frac{1}{4\sqrt{2}LI}. \end{aligned} \tag{S52}$$

Repeating a similar reasoning for all the inequalities in (S51), we get

$$\gamma \leq \min\left\{\frac{1}{8L}, \frac{1}{4\sqrt{2}LI}, \frac{1}{8LI}\right\} \quad \Rightarrow \quad \gamma \leq \frac{1}{8LI}. \tag{S53}$$

Therefore, we can choose a constant step size γ, small enough (and independent of N, n), such that (4.21) holds.

A.4 Zeroth-order Hybrid Gradient Descent

A.4.1 Mathematical Notations

RGE $\hat{\nabla}_{\mathrm{RGE}} F(\mathbf{x}; \mathcal{B}^{\mathrm{r}})$; see (5.2), (5.5)	
$\{\mathbf{u}_i\}$	Uniformly random directions on a unit sphere U_0
n_{r}	Number of random directions, per stochastic sample ξ
μ_{r}	Smoothing parameter
$\mathcal{B}^{\mathrm{r}}$	Mini-batch of stochastic samples with cardinality $\lvert\mathcal{B}^{\mathrm{r}}\rvert$
CGE $\hat{\nabla}_{\mathrm{CGE}} F_{\mathcal{I}}(\mathbf{x}; \mathcal{B}^{\mathrm{c}}, \mathbf{p})$; see (5.4), (5.5)	
$\mu_{\mathrm{c},i}$	Smoothing parameter for the i-th component of CGE; see (5.3)
μ_{c}	Common smoothing parameter, i.e., $\mu_{\mathrm{c},i} = \mu_{\mathrm{c}}$ for all $i \in [d]$
$\{\mathbf{e}_i\}_{i=1}^{d}$	Canonical basis vectors in $\mathbb{R}^d$ ($\mathbf{e}_i$ is a vector of all 0's, except the i-th entry)
$\mathcal{I}$	Coordinate set used in computing CGE (5.4)
$\mathcal{B}^{\mathrm{c}}$	Mini-batch of stochastic samples with cardinality $\lvert\mathcal{B}^{\mathrm{c}}\rvert$
$\mathbf{p}$	Vector of coordinate selection probabilities (5.7)
p_i	$\mathrm{Pr}(i \in \mathcal{I})$; i-th element of $\mathbf{p}$
n_{c}	Coordinate selection budget of CGE
Algorithm 8: HGD - $\mathbf{x}_{t+1} = \mathbf{x}_t - \eta_t \left(\alpha_t \nabla_{\mathrm{r},t} + (1-\alpha_t)\nabla_{\mathrm{c},t}\right)$	
$\mathbf{x}_t$	Iterate of the algorithm at the t-th time step
T	Total number of iterations
η_t	Step-size at time t
$n_{\mathrm{c,t}}$	Coordinate selection budget of CGE at time t
$\alpha_t, \mathcal{B}^{\mathrm{r}}_t, \mathcal{I}_t, \mathcal{B}^{\mathrm{c}}_t, \mathbf{p}_t$	The time-dependent versions of the corresponding quantities
$p_{t,i}$	i-th element of $\mathbf{p}_t$
$\nabla_{\mathrm{r},t}$	$\hat{\nabla}_{\mathrm{RGE}} F(\mathbf{x}_t; \mathcal{B}^{\mathrm{r}}_t)$
$\nabla_{\mathrm{c},t}$	$\hat{\nabla}_{\mathrm{CGE}} F_{\mathcal{I}_t}(\mathbf{x}_t; \mathcal{B}^{\mathrm{c}}_t, \mathbf{p}_t)$
Y_t	$= \{\{\mathbf{u}_{t,i}\}_i, \mathcal{I}_t, \mathcal{B}^{\mathrm{c}}_t, \mathcal{B}^{\mathrm{r}}_t\}$, the randomness at step t
$\mathcal{Y}_t$	σ-algebra generated by $= \{Y_0, Y_1, \ldots, Y_{t-1}\}$
Quantities used in the analysis of Algorithm 8	
d_{nr}	$1 + d/n_{\mathrm{r}}$
$\bar{P}_T$	$\frac{1}{T}\sum_{t=0}^{T-1} \frac{1}{d}\sum_{i=1}^{d} \frac{1}{p_{t,i}}$
c_t	Lower bound such that $p_{t,i} \geq c_t > 0$ for all $i \in [d]$, at all times t
$\frac{1}{\bar{c}_T}$	$\frac{1}{T}\sum_{t=0}^{T-1} \frac{1}{c_t}$
q	$= n_{\mathrm{r}} + n_{\mathrm{c}}$, such that the total per-iteration function query cost is $O(q)$

A.4.2 Proof of Proposition 5.3.1

The proof is motivated by the analysis of gradient sparsification proposed in [188]. We denote by $\mathbf{g} = [g_1, \ldots, g_d]^T \triangleq \hat{\nabla}_{\mathrm{RGE}} F(\mathbf{x}; \mathcal{B}^{\mathrm{r}})$, the RGE to be employed as a probe estimate. We also use the vector $[g_{(1)}, \ldots, g_{(d)}]^T$ to denote a vector with the same entries as $\mathbf{g}$, but sorted in the order of decreasing magnitude. We denote the vector of probabilities $\mathbf{p} = [p_1, \ldots, p_d]^T$.

Given an upper bound on the expected sparsity $\sum_{i=1}^{d} p_i$, we compute the set of probabilities which minimize the variance of the sparse estimator $Q(\mathbf{g})$, by solving the following problem.

$$\min \sum_{i=1}^{d} \frac{g_i^2}{p_i} \qquad \text{s.t.} \qquad \sum_{i=1}^{d} p_i \leq n_{\mathrm{c}}, 0 < p_i \leq 1, \forall i. \tag{S54}$$

PROOF: Introducing Lagrange multipliers λ, $\{\mu_i\}_{i=1}^{d}$ in (S54), the solution is given by solving the following problem:

$$\min_{p} \max_{\lambda \geq 0} \max_{\mu \succeq \mathbf{0}} L(p, \lambda, \mu) = \sum_{i=1}^{d} \frac{g_i^2}{p_i} + \lambda^2 \left(\sum_{i=1}^{d} p_i - n_{\mathrm{c}} \right) + \sum_{i=1}^{d} \mu_i (p_i - 1). \tag{S55}$$

The KKT conditions for (S55) are given by,

$$\begin{aligned} -\frac{g_i^2}{p_i^2} + \lambda^2 + \mu_i = 0, \forall\, i \qquad & \text{(Stationarity)} \\ \lambda^2 \left(\sum_{i=1}^{d} p_i - n_{\mathrm{c}} \right) = 0, \qquad & \text{(Complementary slackness 1)} \\ \mu_i (p_i - 1) = 0, \forall\, i, \qquad & \text{(Complementary slackness 2).} \end{aligned}$$

Combining the stationarity condition with the fact that for $p_i < 1$, $\mu_i = 0$, we get

$$p_i = \begin{cases} 1, & \mu_i \neq 0 \\ \frac{|g_i|}{\lambda}, & \mu_i = 0. \end{cases} . \tag{S56}$$

It follows from (S56) that if $|g_i| \geq |g_j|$, then $p_i \geq p_j$. Therefore, there is a dominating set of coordinates, say S, such that $p_j = 1, \forall j \in S$, and $|g_j|, j \in S$ are the largest absolute magnitudes in the vector $\mathbf{g}$. Suppose the set S has size $k (0 \leq k \leq d)$. Consequently, we get,

$$p_{(i)} = \begin{cases} 1, & i \leq k \\ \frac{|g_{(i)}|}{\lambda}, & i > k. \end{cases} . \tag{S57}$$

Clearly, $\lambda > 0$, hence $\sum_{i=1}^{d} p_i = n_{\mathrm{c}}$. Then, from (S57),

$$k + \frac{1}{\lambda} \sum_{i=k+1}^{d} |g_{(i)}| = n_{\mathrm{c}} \qquad \Leftrightarrow \qquad \lambda = \frac{\sum_{i=k+1}^{d} |g_{(i)}|}{n_{\mathrm{c}} - k}.$$

Consequently, for $i > k$,

$$p_i = \frac{|g_{(i)}|(n_{\mathrm{c}} - k)}{\sum_{i=k+1}^{d} |g_{(i)}|}$$

Hence, the probabilities are calculated using the following steps:

1. Find the smallest k such that

   $$|g_{(k+1)}| \, (n_{\mathrm{c}} - k) \leq \sum_{i=k+1}^{d} |g_{(i)}|,$$

   is true. Denote by S_k, the set of coordinates with the top k largest magnitudes of $|g_i|$.

2. Set the probability vector $\mathbf{p}$ by

   $$p_i = \begin{cases} 1, & \text{if } i \in S_k \\ \frac{|g_i|(n_{\mathrm{c}} - k)}{\sum_{i=k+1}^{d} |g_i|}, & \text{if } i \notin S_k. \end{cases}$$

■

A.4.3 Proof of Proposition 5.3.2

Proof of (5.11)

PROOF: The variance of CGE is bounded as

$$\begin{aligned}
&\mathbb{E}\left\|\hat{\nabla}_{\text{CGE}}F_{\mathcal{I}}(\mathbf{x};\mathcal{B}^{\text{c}},\mathbf{p})-\nabla f(\mathbf{x})\right\|^2\\
&=\mathbb{E}\left\|\hat{\nabla}_{\text{CGE}}F_{\mathcal{I}}(\mathbf{x};\mathcal{B}^{\text{c}},\mathbf{p})-\hat{\nabla}_{\text{CGE}}f(\mathbf{x})+\hat{\nabla}_{\text{CGE}}f(\mathbf{x})-\nabla f(\mathbf{x})\right\|^2\\
&\overset{(e)}{=}\mathbb{E}\left\|\hat{\nabla}_{\text{CGE}}F_{\mathcal{I}}(\mathbf{x};\mathcal{B}^{\text{c}},\mathbf{p})-\hat{\nabla}_{\text{CGE}}f(\mathbf{x})\right\|^2+\left\|\hat{\nabla}_{\text{CGE}}f(\mathbf{x})-\nabla f(\mathbf{x})\right\|^2, \qquad \text{(S58)}
\end{aligned}$$

where $\hat{\nabla}_{\text{CGE}}f(\mathbf{x})$ is the full coordinate CGE, and (e) follows since $\mathbb{E}[\hat{\nabla}_{\text{CGE}}F_{\mathcal{I}}(\mathbf{x};\mathcal{B}^{\text{c}},\mathbf{p})]=\hat{\nabla}_{\text{CGE}}f(\mathbf{x})$. Now, we bound the first term in (S58).

$$\begin{aligned}
&\mathbb{E}\left\|\hat{\nabla}_{\text{CGE}}F_{\mathcal{I}}(\mathbf{x};\mathcal{B}^{\text{c}},\mathbf{p})-\hat{\nabla}_{\text{CGE}}f(\mathbf{x})\right\|^2\overset{(f)}{=}\mathbb{E}\left\|\sum_{i=1}^{d}\frac{I(i\in\mathcal{I})}{p_i}\hat{\nabla}_{\text{CGE}}F_i(\mathbf{x};\mathcal{B}^{\text{c}})-\sum_{i=1}^{d}\hat{\nabla}_{\text{CGE}}f_i(\mathbf{x})\right\|^2\\
&\overset{(g)}{=}\sum_{i=1}^{d}\mathbb{E}\left\|\frac{I(i\in\mathcal{I})}{p_i}\hat{\nabla}_{\text{CGE}}F_i(\mathbf{x};\mathcal{B}^{\text{c}})-\hat{\nabla}_{\text{CGE}}f_i(\mathbf{x})\right\|^2\\
&\overset{(j)}{=}\sum_{i=1}^{d}\frac{1}{p_i^2}\left[\mathbb{E}\left\|I(i\in\mathcal{I})\hat{\nabla}_{\text{CGE}}F_i(\mathbf{x};\mathcal{B}^{\text{c}})-I(i\in\mathcal{I})\hat{\nabla}_{\text{CGE}}f_i(\mathbf{x})\right\|^2\right.\\
&\qquad\qquad\left.+\mathbb{E}\left\|I(i\in\mathcal{I})\hat{\nabla}_{\text{CGE}}f_i(\mathbf{x})-p_i\hat{\nabla}_{\text{CGE}}f_i(\mathbf{x})\right\|^2\right]\\
&=\sum_{i=1}^{d}\frac{1}{p_i^2}\left[\mathbb{E}_{\mathcal{I}}I^2(i\in\mathcal{I})\mathbb{E}_{\mathcal{B}^{\text{c}}}\left\|\hat{\nabla}_{\text{CGE}}F_i(\mathbf{x};\mathcal{B}^{\text{c}})-\hat{\nabla}_{\text{CGE}}f_i(\mathbf{x})\right\|^2\right.\\
&\qquad\qquad\left.+\mathbb{E}_{\mathcal{I}}(I(i\in\mathcal{I})-p_i)^2\left\|\hat{\nabla}_{\text{CGE}}f_i(\mathbf{x})\right\|^2\right]\\
&\overset{(k)}{=}\sum_{i=1}^{d}\frac{1}{p_i}\left[\mathbb{E}_{\mathcal{B}^{\text{c}}}\left\|\hat{\nabla}_{\text{CGE}}F_i(\mathbf{x};\mathcal{B}^{\text{c}})-\hat{\nabla}_{\text{CGE}}f_i(\mathbf{x})\right\|^2+(1-p_i)\left\|\hat{\nabla}_{\text{CGE}}f_i(\mathbf{x})\right\|^2\right], \qquad \text{(S59)}
\end{aligned}$$

where the steps in the derivation of (S59) follow by the reasoning given below:

- (f) follows from the definitions of $\hat{\nabla}_{\text{CGE}}F_{\mathcal{I}}(\mathbf{x};\mathcal{B}^{\text{c}},\mathbf{p})$ in (5.7);
- (g) follows since $\hat{\nabla}_{\text{CGE}}F_i(\mathbf{x}_t;\mathcal{B}_t^{\text{c}}),\hat{\nabla}_{\text{CGE}}f_i(\mathbf{x})$ is aligned with the canonical basis vector $\mathbf{e}_i$, for all i;

- (j) follows since the stochastic sample set $\mathcal{B}^{\mathrm{c}}$ is sampled independent of the coordinate set $\mathcal{I}$. Therefore,

$$\begin{aligned}&\mathbb{E}_{\mathcal{I},\mathcal{B}^{\mathrm{c}}}\left\langle I(i\in\mathcal{I})\hat{\nabla}_{\mathrm{CGE}}F_i(\mathbf{x};\mathcal{B}^{\mathrm{c}})-I(i\in\mathcal{I})\hat{\nabla}_{\mathrm{CGE}}f_i(\mathbf{x}),I(i\in\mathcal{I})\hat{\nabla}_{\mathrm{CGE}}f_i(\mathbf{x})-p_i\hat{\nabla}_{\mathrm{CGE}}f_i(\mathbf{x})\right\rangle\\&\quad=\mathbb{E}_{\mathcal{I}}\left\langle(I(i\in\mathcal{I}))\left(\mathbb{E}_{\mathcal{B}^{\mathrm{c}}}\hat{\nabla}_{\mathrm{CGE}}F_i(\mathbf{x};\mathcal{B}^{\mathrm{c}})-\hat{\nabla}_{\mathrm{CGE}}f_i(\mathbf{x})\right),(I(i\in\mathcal{I})-p_i)\hat{\nabla}_{\mathrm{CGE}}f_i(\mathbf{x})\right\rangle=0,\end{aligned}$$

- (k) follows since the indicator function $I(i\in\mathcal{I})$ is a Bernoulli random variable with $\Pr(i\in\mathcal{I})=p_i$. Therefore, $\mathbb{E}_{\mathcal{I}}[I^2(i\in\mathcal{I})]=1\cdot\Pr(i\in\mathcal{I})+0\cdot\Pr(i\notin\mathcal{I})=p_i$. Also, variance of this Bernoulli random variable is given by $\mathrm{var}(I(i\in\mathcal{I}))=\mathbb{E}_{\mathcal{I}}(I(i\in\mathcal{I})-p_i)^2=p_i(1-p_i)$.

Next, we bound the first term in (S59) as follows.

$$\begin{aligned}\mathbb{E}_{\mathcal{B}^{\mathrm{c}}}&\left\|\hat{\nabla}_{\mathrm{CGE}}F_i(\mathbf{x};\mathcal{B}^{\mathrm{c}})-\hat{\nabla}_{\mathrm{CGE}}f_i(\mathbf{x})\right\|^2=\frac{1}{|\mathcal{B}^{\mathrm{c}}|^2}\sum_{\xi\in\mathcal{B}^{\mathrm{c}}}\mathbb{E}_{\xi}\left\|\hat{\nabla}_{\mathrm{CGE}}F_i(\mathbf{x};\xi)-\hat{\nabla}_{\mathrm{CGE}}f_i(\mathbf{x})\right\|^2\\&=\frac{1}{|\mathcal{B}^{\mathrm{c}}|}\mathbb{E}_{\xi_1}\left\|\hat{\nabla}_{\mathrm{CGE}}F_i(\mathbf{x};\xi_1)-\hat{\nabla}_{\mathrm{CGE}}f_i(\mathbf{x})\right\|^2\qquad\text{for some }\xi_1\in\mathcal{B}^{\mathrm{c}}\\&\overset{(\ell)}{\leq}\frac{3}{|\mathcal{B}^{\mathrm{c}}|}\mathbb{E}_{\xi_1}\left[\left\|\hat{\nabla}_{\mathrm{CGE}}F_i(\mathbf{x};\xi_1)-\mathbf{e}_i\mathbf{e}_i^T\nabla F(\mathbf{x};\xi_1)\right\|^2\right.\\&\qquad\left.+\left\|\mathbf{e}_i\mathbf{e}_i^T\left(\nabla F(\mathbf{x};\xi_1)-\nabla f(\mathbf{x})\right)\right\|^2+\left\|\mathbf{e}_i\mathbf{e}_i^T\nabla f(\mathbf{x})-\hat{\nabla}_{\mathrm{CGE}}f_i(\mathbf{x})\right\|^2\right]\\&\overset{(m)}{\leq}\frac{3}{|\mathcal{B}^{\mathrm{c}}|}\left[\frac{L^2\mu_{\mathrm{c},i}^2}{4}+\zeta^2+\frac{L^2\mu_{\mathrm{c},i}^2}{4}\right]=\frac{3}{|\mathcal{B}^{\mathrm{c}}|}\left(\zeta^2+\frac{L^2\mu_{\mathrm{c},i}^2}{2}\right),\end{aligned}\tag{S60}$$

where (ℓ) follows from the inequality $\|\sum_{i=1}^{s}\mathbf{x}_i\|^2\leq s\sum_{i=1}^{s}\|\mathbf{x}_i\|^2$. (m) follows from the coordinate-wise variance bound in Assumption 2 and [119, Lemma 3.2]. The second term in (S59) can be bounded as

$$\begin{aligned}\left\|\hat{\nabla}_{\mathrm{CGE}}f_i(\mathbf{x})\right\|^2&\leq 2\left\|\hat{\nabla}_{\mathrm{CGE}}f_i(\mathbf{x})-\mathbf{e}_i\mathbf{e}_i^T\nabla f(\mathbf{x})\right\|^2+2\left\|\mathbf{e}_i\mathbf{e}_i^T\nabla f(\mathbf{x})\right\|^2\\&\leq\frac{L^2\mu_{\mathrm{c},i}^2}{2}+2\left(\nabla f(\mathbf{x})\right)_i^2,\end{aligned}\tag{S61}$$

where $(\mathbf{x})_i$ denotes the i-th coordinate of the vector $\mathbf{x}$. Substituting (S60), (S61) in (S59), we get

$$\begin{aligned}
&\mathbb{E}\left\|[\hat{\nabla}_{\mathrm{CGE}}F_{\mathcal{I}}(\mathbf{x};\mathcal{B}^{\mathrm{c}},\mathbf{p})] - \hat{\nabla}_{\mathrm{CGE}}f(\mathbf{x})\right\|^2 \\
&\leq \sum_{i=1}^{d}\frac{1}{p_i}\left[\frac{3}{|\mathcal{B}^{\mathrm{c}}|}\left(\zeta^2 + \frac{L^2\mu_{\mathrm{c},i}^2}{2}\right) + (1-p_i)\left\{\frac{L^2\mu_{\mathrm{c},i}^2}{2} + 2\left(\nabla f(\mathbf{x})\right)_i^2\right\}\right] \\
&= \sum_{i=1}^{d}\frac{1}{p_i}\left[\frac{3}{|\mathcal{B}^{\mathrm{c}}|}\left(\zeta^2 + \frac{L^2\mu_{\mathrm{c},i}^2}{2}\right) + \frac{L^2\mu_{\mathrm{c},i}^2}{2} + 2\left(\nabla f(\mathbf{x})\right)_i^2\right] - \frac{L^2}{2}\sum_{i=1}^{d}\mu_{\mathrm{c},i}^2 - 2\left\|\nabla f(\mathbf{x})\right\|^2. \quad \text{(S62)}
\end{aligned}$$

Next, we bound the second term $\left\|\nabla f(\mathbf{x}) - \hat{\nabla}_{\mathrm{CGE}}f(\mathbf{x})\right\|^2$ in (S58). This is done by improving slightly the corresponding result in [119, Lemma 3].

$$\begin{aligned}
\left\|\nabla f(\mathbf{x}) - \hat{\nabla}_{\mathrm{CGE}}f(\mathbf{x})\right\|^2 &= \left\|\sum_{i=1}^{d}\left(\mathbf{e}_i\mathbf{e}_i^T\hat{\nabla}_{\mathrm{CGE}}f(\mathbf{x}) - \left(\nabla f(\mathbf{x})\right)_i\right)\right\|^2 \\
&= \sum_{i=1}^{d}\left\|\left(\mathbf{e}_i\mathbf{e}_i^T\hat{\nabla}_{\mathrm{CGE}}f(\mathbf{x}) - \left(\nabla f(\mathbf{x})\right)_i\right)\right\|^2 \leq \sum_{i=1}^{d}\frac{L^2\mu_{\mathrm{c},i}^2}{4}, \quad \text{(S63)}
\end{aligned}$$

where $(\mathbf{x})_i$ denotes the i-th component of the vector $\mathbf{x}$. The last inequality follows from [119, Lemma 3]. Finally, substituting (S62), (S63) in (S58), and rearranging the terms, we get

$$\begin{aligned}
&\mathbb{E}\left\|[\hat{\nabla}_{\mathrm{CGE}}F_{\mathcal{I}}(\mathbf{x};\mathcal{B}^{\mathrm{c}},\mathbf{p})] - \nabla f(\mathbf{x})\right\|^2 \\
&\quad\leq \sum_{i=1}^{d}\frac{1}{p_i}\left[\frac{3}{|\mathcal{B}^{\mathrm{c}}|}\left(\zeta^2 + \frac{L^2\mu_{\mathrm{c},i}^2}{2}\right) + \frac{L^2\mu_{\mathrm{c},i}^2}{2} + 2\left(\nabla f(\mathbf{x})\right)_i^2\right] - 2\left\|\nabla f(\mathbf{x})\right\|^2. \quad \text{(S64)}
\end{aligned}$$

■

A.4.4 Choice of α

Substituting the upper bounds from (5.10), (5.11) in (5.9), we get

$$\begin{aligned}
&\mathbb{E}\left\|\hat{\nabla}_{\mathrm{HGE}}F(\mathbf{x};\mathcal{B}^{\mathrm{r}},\mathcal{B}^{\mathrm{c}},\mathcal{I}) - \nabla f(\mathbf{x})\right\|^2 \\
&\leq 2\alpha^2\mathbb{E}\left\|\hat{\nabla}_{\mathrm{RGE}}F(\mathbf{x};\mathcal{B}^{\mathrm{r}}) - \nabla f(\mathbf{x})\right\|^2 + 2(1-\alpha)^2\mathbb{E}\left\|[\hat{\nabla}_{\mathrm{CGE}}F_{\mathcal{I}}(\mathbf{x};\mathcal{B}^{\mathrm{c}},\mathbf{p})] - \nabla f(\mathbf{x})\right\|^2
\end{aligned}$$

$$\leq 2\alpha^2 \left[\frac{2}{|\mathcal{B}^{\mathrm{r}}|} \left(1 + \frac{d}{n_{\mathrm{r}}}\right) \mathbb{E} \left\| \nabla f(\mathbf{x}) \right\|^2 + \frac{2\sigma^2}{|\mathcal{B}^{\mathrm{r}}|} \left(1 + \frac{d}{n_{\mathrm{r}}}\right) + \left(1 + \frac{2}{|\mathcal{B}^{\mathrm{r}}|} + \frac{2}{n_{\mathrm{r}}|\mathcal{B}^{\mathrm{r}}|}\right) \frac{\mu_{\mathrm{r}}^2 L^2 d^2}{4} \right]$$
$$+ 2(1-\alpha)^2 \left[\sum_{i=1}^{d} \frac{1}{p_i} \Big[2 \left(\nabla f(\mathbf{x})\right)_i^2 + \frac{3}{|\mathcal{B}^{\mathrm{c}}|} \left(\zeta^2 + \frac{L^2 \mu_{\mathrm{c},i}^2}{2} \right) + \frac{L^2 \mu_{\mathrm{c},i}^2}{2} \Big] - 2 \left\| \nabla f(\mathbf{x}) \right\|^2 \right]. \quad \text{(S65)}$$

The following steps to choose α are quite similar to those involved in the proof of Theorem 5.4.3. We describe them here as well, for ease of understanding. In (S65), for the time being, we ignore the terms involving $\nabla f(\mathbf{x})$. We assume the CGE smoothing parameters to be constant across coordinates, i.e., $\mu_{\mathrm{c},i} = \mu_{\mathrm{c}}$, for all $i \in [d]$. Using $|\mathcal{B}^{\mathrm{r}}| \geq 1, |\mathcal{B}^{\mathrm{c}}| \geq 1$, from (S65) we can bound the remaining terms to get

$$2\alpha^2 \left[2\sigma^2 \left(1 + \frac{d}{n_{\mathrm{r}}}\right) + \left(3 + \frac{2}{n_{\mathrm{r}}}\right) \frac{\mu_{\mathrm{r}}^2 L^2 d^2}{4} \right] + 2(1-\alpha)^2 \left[\sum_{i=1}^{d} \frac{1}{p_i} \Big[3\zeta^2 + 2L^2 \mu_{\mathrm{c}}^2 \Big] \right]. \quad \text{(S66)}$$

We denote $\bar{P} = \frac{1}{d} \sum_{i=1}^{d} \frac{1}{p_i}$. Note that $\sigma^2 = d\zeta^2$. Also, we choose the smoothing parameters $\mu_{\mathrm{c}}, \mu_{\mathrm{r}}$ to be small enough such that $\sigma^2 \geq L^2 \mu_{\mathrm{c}}^2 d, \sigma^2 \geq L^2 d^2 \mu_{\mathrm{r}}^2 / 4$. Consequently,

$$2\sigma^2 \left(1 + \frac{d}{n_{\mathrm{r}}}\right) + \left(3 + \frac{2}{n_{\mathrm{r}}}\right) \frac{\mu_{\mathrm{r}}^2 L^2 d^2}{4} \leq 5\sigma^2 \left(1 + \frac{d}{n_{\mathrm{r}}}\right),$$
$$\bar{P} \left[3\sigma^2 + 2L^2 \mu_{\mathrm{c}}^2 d \right] \leq 5\sigma^2 \bar{P}.$$

Substituting in (S66), we get

$$10\sigma^2 \left[\alpha^2 \left(1 + \frac{d}{n_{\mathrm{r}}}\right) + (1-\alpha)^2 \bar{P} \right]. \quad \text{(S67)}$$

Then, the optimal combination coefficient α^*, which minimizes (S67) is given by

$$\alpha^* = \left[1 + \frac{1 + \frac{d}{n_{\mathrm{r}}}}{\bar{P}} \right]^{-1}. \quad \text{(S68)}$$

A.4.5 Special Cases of Theorem 5.4.3

Regime 2: $d_{\mathrm{nr}} = 1 + \frac{d}{n_{\mathrm{r}}} \ll \bar{P}_T$**.**

Since $\bar{P}_T \geq 1$, this implies that $\frac{d}{n_{\mathrm{r}}} \ll \bar{P}_T$ and $\frac{1}{\bar{P}_T} \ll \frac{n_{\mathrm{r}}}{d}$ implies that on average, sampling probabilities are much smaller than n_{r}/d. In other words, the per-iteration query budget of CGE is much smaller than RGE, and $\alpha \to 1$ (see (5.33)). With smoothing parameters $\mu_{\mathrm{c}} = O\left((d_{\mathrm{nr}}/d^2T)^{1/4}\right)$, $\mu_{\mathrm{r}} = O\left(d^{-1}(d_{\mathrm{nr}}/T)^{1/4}\right)$, the resulting convergence rate is

$$\mathbb{E}\left\|\nabla f(\bar{\mathbf{x}}_T)\right\|^2 \leq O\left(\sqrt{\frac{1+\frac{d}{n_{\mathrm{r}}}}{T}}\right).$$

FQC to achieve $\mathbb{E}\|\nabla f(\bar{\mathbf{x}}_T)\|^2 \leq \epsilon$ is given by $O(T \cdot n_{\mathrm{r}} + \sum_{t=0}^{T-1}\sum_{i=1}^{d} p_{t,i}) = O(T \cdot (n_{\mathrm{r}} + n_{\mathrm{c}}))$. In the special case of uniform distribution for CGE, i.e., $p_{t,i} = n_{\mathrm{c}}/d$, $\bar{P}_T = d/n_{\mathrm{c}}$. $\frac{d}{n_{\mathrm{r}}} \ll \bar{P}_T$ implies $n_{\mathrm{c}} \ll n_{\mathrm{r}}$. Hence, FQC to achieve $\mathbb{E}\|\nabla f(\bar{\mathbf{x}}_T)\|^2 \leq \epsilon$ is $O(T \cdot n_{\mathrm{r}}) = O(d/\epsilon^2)$ (assuming $n_{\mathrm{r}} = O(d)$). Naturally, both the convergence rate and FQC are dominated by RGE. Also, note that compared to ZO-SGD, we achieve the same convergence rate, while the bound on μ_{r} is more relaxed (see Table 5.1).

Regime 3: $d_{\mathrm{nr}} = 1 + \frac{d}{n_{\mathrm{r}}}$ ***and*** $\bar{P}_T$ ***are comparable in value.***

To gain some insight into this case where the function query budgets of RGE and CGE are comparable, we again look at the uniform distribution $p_{t,i} = \frac{n_{\mathrm{c}}}{d}, \forall\ t, i$. So, $\bar{P}_T = \frac{d}{n_{\mathrm{c}}}$. The total per-iteration function query cost of HGE is $O(n_{\mathrm{r}} + n_{\mathrm{c}})$. First, note that

$$\begin{aligned}\frac{1+\frac{d}{n_{\mathrm{r}}}}{1+\frac{1+\frac{d}{n_{\mathrm{r}}}}{\bar{P}_T}} = \frac{1+\frac{d}{n_{\mathrm{r}}}}{1+\left(1+\frac{d}{n_{\mathrm{r}}}\right)\frac{n_{\mathrm{c}}}{d}} &= \frac{dn_{\mathrm{r}}+d^2}{n_{\mathrm{r}}n_{\mathrm{c}}+d(n_{\mathrm{r}}+n_{\mathrm{c}})} \\ &\leq \frac{n_{\mathrm{r}}}{n_{\mathrm{r}}+n_{\mathrm{c}}} + \frac{d}{n_{\mathrm{r}}+n_{\mathrm{c}}} \leq 1 + \frac{d}{n_{\mathrm{r}}+n_{\mathrm{c}}}.\end{aligned}$$

With $\mu_{\mathrm{c}} = O\left(\left(\frac{1}{d^2 T}\left(1 + \frac{d}{n_{\mathrm{r}} + n_{\mathrm{c}}}\right)\right)^{1/4}\right)$, $\mu_{\mathrm{r}} = O\left(\frac{1}{d}\left(\frac{1}{T}\left(1 + \frac{d}{n_{\mathrm{r}} + n_{\mathrm{c}}}\right)\right)^{1/4}\right)$, the resulting convergence rate is

$$\mathbb{E}\left\|\nabla f(\bar{\mathbf{x}}_T)\right\|^2 \leq O\left(\sqrt{\frac{1}{T}\left(1 + \frac{d}{n_{\mathrm{r}} + n_{\mathrm{c}}}\right)}\right).$$

FQC to achieve $\mathbb{E}\|\nabla f(\bar{\mathbf{x}}_T)\|^2 \leq \epsilon$ is given by $O(T \cdot n_{\mathrm{r}} + T \cdot n_{\mathrm{c}}) = O(d/\epsilon^2)$ (assuming $n_{\mathrm{r}} + n_{\mathrm{c}} = O(d)$).

A.4.6 Convex Case

Before, proceeding with the proof of Theorem 5.5.1, we prove some intermediate results, which shall be used along the way. First, using convexity of f (Assumption 3)

$$\begin{aligned} \sum_{t=1}^{T}\left(f(\mathbf{x}_t) - f(\mathbf{x}^*)\right) &\leq \sum_{t=1}^{T}\left\langle \nabla f(\mathbf{x}_t), \mathbf{x}_t - \mathbf{x}^*\right\rangle \\ &= \sum_{t=1}^{T}\left\langle \nabla_t, \mathbf{x}_t - \mathbf{x}^*\right\rangle + \sum_{t=1}^{T}\left\langle \nabla f(\mathbf{x}_t) - \nabla_t, \mathbf{x}_t - \mathbf{x}^*\right\rangle \end{aligned} \tag{S69}$$

where $x^* = \operatorname{argmin}_{\mathbf{x}\in\mathrm{dom}f} f(\mathbf{x})$, and we denote the descent direction used in Algorithm 8 as $\nabla_t \triangleq \alpha_t \nabla_{\mathrm{r},t} + (1-\alpha_t)\nabla_{\mathrm{c},t}$. Next, we bound both the terms in (S69) separately in the following two results.

Proposition A.4.1. *Under Assumption 3, and using non-increasing step-sizes $\{\eta_t\}$ in Algorithm 8*

$$\sum_{t=1}^{T}\left\langle \nabla_t, \mathbf{x}_t - \mathbf{x}^*\right\rangle \leq \frac{R^2}{2\eta_T} + \sum_{t=1}^{T}\frac{\eta_t}{2}\left\|\nabla_t\right\|^2 \tag{S70}$$

PROOF: The results follows by a straightforward application of the Young's inequality, and using Assumption 3 to bound $\|\mathbf{x}_t - \mathbf{x}^*\|$ with R. ■

Proposition A.4.2.

$$\sum_{t=1}^{T}\mathbb{E}\left\langle \nabla f(\mathbf{x}_t) - \nabla_t, \mathbf{x}_t - \mathbf{x}^*\right\rangle \leq RL\sum_{t=1}^{T}\left(\frac{\alpha_t \mu_{\mathrm{r}} d}{2} + (1-\alpha_t)\sqrt{d}\mu_{\mathrm{c}}\right). \tag{S71}$$

PROOF:

$$\begin{aligned}
&\sum_{t=1}^{T} \mathbb{E}\left\langle \nabla f(\mathbf{x}_t) - \nabla_t, \mathbf{x}_t - \mathbf{x}^* \right\rangle \\
&\quad = \sum_{t=1}^{T} \left[\alpha_t \mathbb{E}\left[\langle \nabla f(\mathbf{x}_t) - \nabla_{\mathrm{r},t}, \mathbf{x}_t - \mathbf{x}^* \rangle \mid \mathcal{Y}_t \right] + (1-\alpha_t)\mathbb{E}\left[\langle \nabla f(\mathbf{x}_t) - \nabla_{\mathrm{c},t}, \mathbf{x}_t - \mathbf{x}^* \rangle \mid \mathcal{Y}_t \right] \right] \\
&\quad \overset{(a)}{=} \sum_{t=1}^{T} \left[\alpha_t \mathbb{E}\left\langle \nabla f(\mathbf{x}_t) - \nabla f_{\mu_\mathrm{r}}(\mathbf{x}_t), \mathbf{x}_t - \mathbf{x}^* \right\rangle + (1-\alpha_t)\mathbb{E}\left\langle \nabla f(\mathbf{x}_t) - \hat{\nabla}_{\mathrm{CGE}} f(\mathbf{x}_t), \mathbf{x}_t - \mathbf{x}^* \right\rangle \right] \\
&\quad \overset{(b)}{\leq} \sum_{t=1}^{T} \left[\alpha_t \mathbb{E}\left\| \mathbf{x}_t - \mathbf{x}^* \right\|_2 \left(\frac{\mu_\mathrm{r} L d}{2} \right) + (1-\alpha_t)\mathbb{E}\left\| \mathbf{x}_t - \mathbf{x}^* \right\|_2 L\sqrt{d}\mu_\mathrm{c} \right],
\end{aligned}$$

where, (a) follows from $\mathbb{E}\nabla_{\mathrm{r},t} = \nabla f_{\mu_\mathrm{r}}(\mathbf{x}_t)$, and $\mathbb{E}\nabla_{\mathrm{c},t} = \hat{\nabla}_{\mathrm{CGE}} f(\mathbf{x}_t)$. (b) follows from Cauch-Schwarz inequality, and by using the bounds $\|\nabla f(\mathbf{x}_t) - \nabla f_{\mu_\mathrm{r}}(\mathbf{x}_t)\| \leq \frac{\mu_\mathrm{r} L d}{2}$ [60, Lemma 4.1], and $\|\nabla f(\mathbf{x}_t) - \hat{\nabla}_{\mathrm{CGE}} f(\mathbf{x}_t)\| \leq L\sqrt{d}\mu_\mathrm{c}$ [86, Lemma 3]. The result follows since $\|\mathbf{x}_t - \mathbf{x}^*\|_2 \leq R$ from Assumption 3. ■

Next we bound $\|\nabla_t\|^2$ which appears in Proposition A.4.1. We make use of Assumption 4 for this.

Proposition A.4.3.

$$\begin{aligned}
\mathbb{E}\left\|\nabla_t\right\|^2 \leq 2\alpha_t^2 &\left[\left(2 + \frac{4}{|\mathcal{B}^\mathrm{r}|}\left(1 + \frac{d}{n_\mathrm{r}}\right)\right) \|\nabla f(\mathbf{x})\|^2 + \frac{4\sigma^2}{|\mathcal{B}^\mathrm{r}|}\left(1 + \frac{d}{n_\mathrm{r}}\right) \right. \\
&\left. + \left(1 + \frac{2}{|\mathcal{B}^\mathrm{r}|} + \frac{2}{n_\mathrm{r}|\mathcal{B}^\mathrm{r}|}\right) \frac{\mu_\mathrm{r}^2 L^2 d^2}{2} \right] \\
&+ 2(1-\alpha_t)^2 \sum_{i=1}^{d} \frac{1}{p_{t,i}} \left[\frac{3\zeta^2}{|\mathcal{B}_t^\mathrm{c}|} + \frac{L^2\mu_\mathrm{c}^2}{2}\left(1 + \frac{3}{|\mathcal{B}_t^\mathrm{c}|}\right) + 2\left(\nabla f(\mathbf{x}_t)\right)_i^2 \right].
\end{aligned} \tag{S72}$$

PROOF: The proof follows by substituting the bounds on $\mathbb{E}\left\|\nabla_{\mathrm{r},t}\right\|^2$ and $\mathbb{E}\left\|\nabla_{\mathrm{c},t}\right\|^2$ from Proposition 5.4.2. ■

A.4.7 Proof of Theorem 5.5.1 (Convex Case)

The set of coordinate-wise probabilities $\{p_{t,i}\}$ are chosen such that $p_{t,i} \geq \bar{c},\ \forall\, i, t$.

PROOF: Using Proposition A.4.1, A.4.2, A.4.3 in (S69), we get

$$\begin{aligned}\sum_{t=1}^{T}\left(\mathbb{E}f(\mathbf{x}_t)-f(\mathbf{x}^*)\right) &\leq \frac{R^2}{2\eta_T}+RL\sum_{t=1}^{T}\left(\frac{\alpha_t\mu_{\mathrm{r}}d}{2}+(1-\alpha_t)\sqrt{d}\mu_{\mathrm{c}}\right)\\ &\quad+\sum_{t=1}^{T}\eta_t(1-\alpha_t)^2\sum_{i=1}^{d}\frac{1}{p_{t,i}}\left[\frac{3\zeta^2}{|\mathcal{B}_t^{\mathrm{c}}|}+\frac{L^2\mu_{\mathrm{c}}^2}{2}\left(1+\frac{3}{|\mathcal{B}_t^{\mathrm{c}}|}\right)+2\left(\nabla f(\mathbf{x}_t)\right)_i^2\right]\\ &\quad+\sum_{t=1}^{T}\eta_t\alpha_t^2\left[\left(2+\frac{4}{|\mathcal{B}^{\mathrm{r}}|}\left(1+\frac{d}{n_{\mathrm{r}}}\right)\right)\|\nabla f(\mathbf{x})\|^2+\frac{4\sigma^2}{|\mathcal{B}^{\mathrm{r}}|}\left(1+\frac{d}{n_{\mathrm{r}}}\right)\right.\\ &\qquad\left.+\left(1+\frac{2}{|\mathcal{B}^{\mathrm{r}}|}+\frac{2}{n_{\mathrm{r}}|\mathcal{B}^{\mathrm{r}}|}\right)\frac{\mu_{\mathrm{r}}^2L^2d^2}{2}\right].\end{aligned} \tag{S73}$$

Here, note that

$$\sum_{i=1}^{d}\frac{1}{p_{t,i}}\left(\nabla f(\mathbf{x}_t)\right)_i^2\leq\frac{1}{\bar{c}}\|\nabla f(\mathbf{x}_t)\|^2 \tag{S74}$$

We take constant combination coefficients, i.e., $\alpha_t=\alpha$ for all t, and constant step-sizes $\eta_t=\eta$ for all t. We also use Assumption 4 to bound $\|\nabla f(\mathbf{x}_t)\|$. We denote $\bar{P}_T=\frac{1}{Td}\sum_{t=0}^{T-1}\sum_{i=1}^{d}\frac{1}{p_{t,i}}$. To focus only on the effect of the number of random directions in RGE n_{r}, and the probabilities of CGE $\{p_{t,i}\}$ on convergence, we get rid of the sample set sizes using $|\mathcal{B}_t^{\mathrm{r}}|\geq 1, |\mathcal{B}_t^{\mathrm{c}}|\geq 1$ for all t. Dividing both sides of (S73) by T, we get

$$\begin{aligned}\frac{1}{T}\sum_{t=1}^{T}\left(\mathbb{E}f(\mathbf{x}_t)-f^*\right) &\leq \frac{R^2}{2\eta T}+RL\left(\frac{\alpha\mu_{\mathrm{r}}d}{2}+(1-\alpha)\sqrt{d}\mu_{\mathrm{c}}\right)\\ &\quad+\alpha^2\eta\left[6\left(1+\frac{d}{n_{\mathrm{r}}}\right)G^2+\left(\frac{3}{2}+\frac{1}{n_{\mathrm{r}}}\right)\mu_{\mathrm{r}}^2d^2L^2+4\left(1+\frac{d}{n_{\mathrm{r}}}\right)\sigma^2\right]\\ &\quad+(1-\alpha)^2\eta\left[2G^2\frac{1}{\bar{c}}+\bar{P}_T\left(3\sigma^2+2L^2\mu_{\mathrm{c}}^2d\right)\right].\end{aligned} \tag{S75}$$

As in Theorem 5.4.3, we choose the smoothing parameters such that $\mu_{\mathrm{c}}=\frac{\mu_{\mathrm{r}}\sqrt{d}}{2}$. Again, using the same reasoning as in Theorem 5.4.3, $\sigma^2\geq L^2\mu_{\mathrm{c}}^2d, \sigma^2\geq\mu_{\mathrm{r}}^2L^2d^2$. Also, note that $\bar{P}_T\leq\frac{1}{\bar{c}}$.

Consequently (S75) simplifies to

$$\begin{aligned}
&\frac{1}{T}\sum_{t=1}^{T}\left(\mathbb{E}f(\mathbf{x}_t)-f^*\right)\\
&\quad\leq \frac{R^2}{2\eta T}+RL\sqrt{d}\mu_{\mathrm{c}}+6\alpha^2\eta\left[G^2\left(1+\frac{d}{n_{\mathrm{r}}}\right)+\sigma^2\left(1+\frac{d}{n_{\mathrm{r}}}\right)\right]+6(1-\alpha)^2\eta\left[\frac{G^2+\sigma^2}{\bar{c}}\right]\\
&\quad\overset{(a)}{\leq} 2\sqrt{\frac{3R^2}{T}\left[\alpha^2\left(G^2+\sigma^2\right)\left(1+\frac{d}{n_{\mathrm{r}}}\right)+(1-\alpha)^2\left(\frac{G^2+\sigma^2}{\bar{c}}\right)\right]}+RL\sqrt{d}\mu_{\mathrm{c}}, \qquad \text{(S76)}
\end{aligned}$$

where (a) follows by optimizing over η. Choosing the value of α

$$\alpha^*=\left[1+\bar{c}\left(1+\frac{d}{n_{\mathrm{r}}}\right)\right]^{-1} \qquad \text{(S77)}$$

and substituting in (S76), we get

$$\frac{1}{T}\sum_{t=1}^{T}\left(\mathbb{E}f(\mathbf{x}_t)-f^*\right)\leq O\left(R\sqrt{\frac{(G^2+\sigma^2)}{T}\frac{\left(1+\frac{d}{n_{\mathrm{r}}}\right)}{\left[1+\bar{c}\left(1+\frac{d}{n_{\mathrm{r}}}\right)\right]}}\right)+RL\sqrt{d}\mu_{\mathrm{c}}. \qquad \text{(S78)}$$

The choice of $\mu_{\mathrm{c}}=O\left(\sqrt{\frac{1}{dT}\frac{\left(1+\frac{d}{n_{\mathrm{r}}}\right)}{\left[1+\bar{c}\left(1+\frac{d}{n_{\mathrm{r}}}\right)\right]}}\right)$ in (S78) yields

$$\frac{1}{T}\sum_{t=1}^{T}\left(\mathbb{E}f(\mathbf{x}_t)-f^*\right)\leq O\left(R\sqrt{\frac{1}{T}\frac{\left(1+\frac{d}{n_{\mathrm{r}}}\right)}{\left[1+\bar{c}\left(1+\frac{d}{n_{\mathrm{r}}}\right)\right]}}\right). \qquad \text{(S79)}$$

■

Similar to as we did in the nonconvex case (Appendix A.4.5), we next explore the convergence conditions of Theorem 5.5.1 under some special cases. The reasoning is quite similar in this case as well.

Regime	Smoothing parameter μ_c	Convergence rate
$1+\frac{d}{n_\mathrm{r}} \gg \frac{1}{\bar{c}}$	$\mathcal{O}\left(\sqrt{\frac{1}{d\bar{c}T}}\right)$	$\mathcal{O}\left(\sqrt{\frac{1}{T\bar{c}}}\right)$
$1+\frac{d}{n_\mathrm{r}} \ll \frac{1}{\bar{c}}$	$\mathcal{O}\left(\sqrt{\frac{1}{dT}\left(1+\frac{d}{n_\mathrm{r}}\right)}\right)$	$\mathcal{O}\left(\sqrt{\frac{1}{T}\left(1+\frac{d}{n_\mathrm{r}}\right)}\right)$
$1+\frac{d}{n_\mathrm{r}} \approx \frac{1}{\bar{c}}$ $p_{t,i} \equiv \frac{n_\mathrm{c}}{d}$	$\mathcal{O}\left(\sqrt{\frac{1}{dT}\left(1+\frac{d}{n_\mathrm{r}+n_\mathrm{c}}\right)}\right)$	$\mathcal{O}\left(\sqrt{\frac{1}{T}\left(1+\frac{d}{n_\mathrm{r}+n_\mathrm{c}}\right)}\right)$

Table A1: Comparison of smoothing parameters and convergence rates, for difference regimes of RGE and CGE query budgets

A.4.8 Special Cases of Theorem 5.5.1 (Convex Case)

Similar to as we did in the nonconvex case (Appendix A.4.5), we next explore the convergence conditions of Theorem 5.5.1 under some special cases. We summarize the results for three regimes in Table A1, depending on the relative values of $1+\frac{d}{n_\mathrm{r}}$ and $\frac{1}{\bar{c}}$. The reasoning is quite similar.

Regime 1: $1+\frac{d}{n_\mathrm{r}} \gg \frac{1}{\bar{c}}$**.**

Since $\frac{1}{\bar{c}} \geq 1$, this implies that $\frac{d}{n_\mathrm{r}} \gg \frac{1}{\bar{c}}$, or $\bar{c} \gg \frac{n_\mathrm{r}}{d}$. This implies that the sampling probabilities are much greater than n_r/d. In other words, the per-iteration query budget of CGE is much higher than RGE, and $\alpha \to 0$ (see (S77)). With smoothing parameter $\mu_\mathrm{c} = O\left(\sqrt{\frac{1}{d\bar{c}T}}\right)$, the resulting convergence rate is

$$\mathbb{E}\left\|\nabla f(\bar{\mathbf{x}}_T)\right\|^2 \leq O\left(\sqrt{\frac{1}{T\bar{c}}}\right).$$

To gain further insight, we consider the special case of uniform distribution, such that $p_{t,i} = \frac{n_\mathrm{c}}{d}, \forall\, t, i$. Hence, $\frac{1}{\bar{c}} = \frac{d}{n_\mathrm{c}}$, where $\frac{d}{n_\mathrm{r}} \gg \frac{1}{\bar{c}}$ implies $n_\mathrm{c} \gg n_\mathrm{r}$. Consequently, the convergence rate becomes

$$\mathbb{E}\left\|\nabla f(\bar{\mathbf{x}}_T)\right\|^2 \leq O\left(\sqrt{\frac{d}{n_\mathrm{c}T}}\right).$$

And the FQC to achieve $\mathbb{E}\|\nabla f(\bar{\mathbf{x}}_T)\|^2 \leq \epsilon$ is $O(T \cdot n_\mathrm{c} + T \cdot n_\mathrm{r}) = O(T \cdot n_\mathrm{c}) = O(d/\epsilon^2)$.

Regime 2: $1+\frac{d}{n_{\mathrm{r}}} \ll \frac{1}{\bar{c}}$**.**

Since $\frac{1}{\bar{c}} \geq 1$, this implies that $\frac{d}{n_{\mathrm{r}}} \ll \frac{1}{\bar{c}}$, or $\bar{c} \ll \frac{n_{\mathrm{r}}}{d}$. This implies that A sufficient condition under which this holds is if the sampling probabilities are all much smaller than n_{r}/d. In other words, the per-iteration query budget of CGE is much smaller than RGE, and $\alpha \rightarrow 1$ (see (S77)). With smoothing parameter $\mu_{\mathrm{c}} = O\left(\sqrt{\frac{1+\frac{d}{n_{\mathrm{r}}}}{dT}}\right)$, the resulting convergence rate is

$$\mathbb{E}\left\|\nabla f(\bar{\mathbf{x}}_T)\right\|^2 \leq O\left(\sqrt{\frac{1+\frac{d}{n_{\mathrm{r}}}}{T}}\right).$$

To gain further insight, we consider the special case of uniform distribution, such that $p_{t,i} = \frac{n_{\mathrm{c}}}{d}, \forall\, t, i$. Hence, $\frac{1}{\bar{c}} = \frac{d}{n_{\mathrm{c}}}$, where $\frac{d}{n_{\mathrm{r}}} \ll \frac{1}{\bar{c}}$ implies $n_{\mathrm{c}} \ll n_{\mathrm{r}}$. Consequently, the FQC to achieve $\mathbb{E}\|\nabla f(\bar{\mathbf{x}}_T)\|^2 \leq \epsilon$ is $O(T \cdot n_{\mathrm{c}} + T \cdot n_{\mathrm{r}}) = O(T \cdot n_{\mathrm{r}}) = O(d/\epsilon^2)$ (assuming $n_{\mathrm{r}} = O(d)$).

Regime 3: $d_{\mathrm{nr}} = 1+\frac{d}{n_{\mathrm{r}}}$ ***and*** $\frac{1}{\bar{c}}$ ***are comparable in value.***

To gain some insight into this case where the function query budgets of RGE and CGE are comparable, we again look at the uniform distribution $p_{t,i} = \frac{n_{\mathrm{c}}}{d}, \forall\, t, i$. So, $\frac{1}{\bar{c}} = \frac{d}{n_{\mathrm{c}}}$. The total per-iteration function query cost of HGE is $O(n_{\mathrm{r}} + n_{\mathrm{c}})$. First, note that

$$\begin{aligned}\frac{1+\frac{d}{n_{\mathrm{r}}}}{1+\bar{c}\left(1+\frac{d}{n_{\mathrm{r}}}\right)} &= \frac{1+\frac{d}{n_{\mathrm{r}}}}{1+\left(1+\frac{d}{n_{\mathrm{r}}}\right)\frac{n_{\mathrm{c}}}{d}} = \frac{dn_{\mathrm{r}}+d^2}{n_{\mathrm{r}}n_{\mathrm{c}}+d(n_{\mathrm{r}}+n_{\mathrm{c}})} \\ &\leq \frac{n_{\mathrm{r}}}{n_{\mathrm{r}}+n_{\mathrm{c}}} + \frac{d}{n_{\mathrm{r}}+n_{\mathrm{c}}} \leq 1+\frac{d}{n_{\mathrm{r}}+n_{\mathrm{c}}}.\end{aligned}$$

With $\mu_{\mathrm{c}} = O\left(\sqrt{\frac{1}{dT}\left(1+\frac{d}{n_{\mathrm{r}}+n_{\mathrm{c}}}\right)}\right)$ and $\mu_{\mathrm{r}} = O\left(\frac{1}{d}\sqrt{\frac{1}{T}\left(1+\frac{d}{n_{\mathrm{r}}+n_{\mathrm{c}}}\right)}\right)$, the resulting convergence rate is

$$\mathbb{E}\left\|\nabla f(\bar{\mathbf{x}}_T)\right\|^2 \leq O\left(\sqrt{\frac{1}{T}\left(1+\frac{d}{n_{\mathrm{r}}+n_{\mathrm{c}}}\right)}\right).$$

FQC to achieve $\mathbb{E}\|\nabla f(\bar{\mathbf{x}}_T)\|^2 \leq \epsilon$ is given by $O(T \cdot n_\mathrm{r} + T \cdot n_\mathrm{c}) = O(d/\epsilon^2)$ (assuming $n_\mathrm{r} + n_\mathrm{c} = O(d)$). For $O(n_\mathrm{r} + n_\mathrm{c})$ function evaluations per iteration ($O(n_\mathrm{r})$ for RGE, $O(n_\mathrm{c})$ for CGE), this rate is order optimal [50].

A.4.9 Proof of Theorem 5.5.1 (Strongly Convex Case)

For strongly convex functions, the error can be expressed either in terms of the *regret* $\sum_{t=0}^{T-1} (\mathbb{E}f(\mathbf{x}_t) - f^*)$ (as in the convex case), or in terms of distance of the iterate from the optima $\|\mathbf{x}_t - \mathbf{x}^*\|^2$, because of the following inequality

$$\frac{\bar{\sigma}}{2} \|\mathbf{x} - \mathbf{x}^*\|^2 \leq f(\mathbf{x}) - f^*, \qquad \forall\, \mathbf{x} \in \mathrm{dom} f.$$

We begin with the following general result.

Lemma A.4.4. *Suppose f satisfies Assumption 1, 5. Then, the smooth approximation f_μ of the function f, defined as $f_\mu = \mathbb{E}_{\mathbf{u}\in U_0}[f(\mathbf{x} + \mu\mathbf{u})]$ is also $\bar{\sigma}$-strongly convex.*

PROOF: We use the following property of strongly convex functions.

$$f(\alpha\mathbf{x} + (1-\alpha)\mathbf{y}) \leq \alpha f(\mathbf{x}) + (1-\alpha) f(\mathbf{y}) - \frac{\bar{\sigma}\alpha(1-\alpha)}{2} \|\mathbf{x} - \mathbf{y}\|^2, \qquad \alpha \in [0,1].$$

Now,

$$\begin{aligned}
f_\mu(\alpha\mathbf{x} + (1-\alpha)\mathbf{y}) &= \mathbb{E}_{\mathbf{u}\in U_0}\left[f\left(\alpha\mathbf{x} + (1-\alpha)\mathbf{y} + \mu\mathbf{u}\right)\right] \\
&= \mathbb{E}_{\mathbf{u}\in U_0}\left[f\left(\alpha(\mathbf{x} + \mu\mathbf{u}) + (1-\alpha)(\mathbf{y} + \mu\mathbf{u})\right)\right] \\
&\leq \mathbb{E}_{\mathbf{u}\in U_0}\left[\alpha f(\mathbf{x} + \mu\mathbf{u}) + (1-\alpha) f(\mathbf{y} + \mu\mathbf{u}) - \frac{\bar{\sigma}\alpha(1-\alpha)}{2} \|\mathbf{x} + \mu\mathbf{u} - \mathbf{y} - \mu\mathbf{u}\|^2\right] \\
&= \alpha\mathbb{E}_{\mathbf{u}\in U_0}\left[f(\mathbf{x} + \mu\mathbf{u})\right] + (1-\alpha)\mathbb{E}_{\mathbf{u}\in U_0}\left[f(\mathbf{y} + \mu\mathbf{u})\right] - \frac{\bar{\sigma}\alpha(1-\alpha)}{2} \|\mathbf{x} - \mathbf{y}\|^2 \\
&= \alpha f_\mu(\mathbf{x}) + (1-\alpha) f_\mu(\mathbf{y}) - \frac{\bar{\sigma}\alpha(1-\alpha)}{2} \|\mathbf{x} - \mathbf{y}\|^2
\end{aligned}$$

■

We start with the following intermediate result.

Proposition A.4.5. *Given f satisfies Assumption 1, 5, the iterates of Algorithm 8 $\{\mathbf{x}_t\}_t$ satisfy*

$$\mathbb{E}\left\|\mathbf{x}_{t+1}-\mathbf{x}^*\right\|^2 \leq \left(1-\frac{\eta_t\bar{\sigma}}{2}\right)\mathbb{E}\left\|\mathbf{x}_t-\mathbf{x}^*\right\|^2 - 2\eta_t e_t + 2\eta_t\alpha_t L\mu_{\mathrm{r}}^2 + 2\eta_t(1-\alpha_t)\frac{L^2 d\mu_{\mathrm{c}}^2}{\bar{\sigma}} \\ + 2\eta_t^2\left[\alpha_t^2\mathbb{E}\left\|\nabla_{\mathrm{r},t}\right\|^2 + (1-\alpha_t)^2\mathbb{E}\left\|\nabla_{\mathrm{c},t}\right\|^2\right], \tag{S80}$$

where, $e_t = \mathbb{E}f(\mathbf{x}_t) - f^*$. Recall that η_t is the step-size at time t, α_t is the combination coefficient at time t, $\mu_{\mathrm{r}}, \mu_{\mathrm{c}}$ are the smoothing parameters for RGE, CGE respectively.

PROOF: Using the iterate update equation (line 8) in Algorithm 8

$$\left\|\mathbf{x}_{t+1}-\mathbf{x}^*\right\|^2 = \left\|\mathbf{x}_t - \eta_t\left(\alpha_t\nabla_{\mathrm{r},t} + (1-\alpha_t)\nabla_{\mathrm{c},t}\right) - \mathbf{x}^*\right\|^2 \\ = \left\|\mathbf{x}_t-\mathbf{x}^*\right\|^2 + \eta_t^2\left\|\alpha_t\nabla_{\mathrm{r},t} + (1-\alpha_t)\nabla_{\mathrm{c},t}\right\|^2 - 2\eta_t\left\langle \mathbf{x}_t-\mathbf{x}^*, \alpha_t\nabla_{\mathrm{r},t} + (1-\alpha_t)\nabla_{\mathrm{c},t}\right\rangle. \tag{S81}$$

We take expectation and consider the individual terms in (S81) one at a time.

$$\mathbb{E}\left\|\alpha_t\nabla_{\mathrm{r},t} + (1-\alpha_t)\nabla_{\mathrm{c},t}\right\|^2 \leq 2\alpha_t^2\mathbb{E}\left\|\nabla_{\mathrm{r},t}\right\|^2 + 2(1-\alpha_t)^2\mathbb{E}\left\|\nabla_{\mathrm{c},t}\right\|^2, \tag{S82}$$

where (S82) follows from $\|a+b\|^2 \leq 2\|a\|^2 + 2\|b\|^2$. Next,

$$-\eta_t\mathbb{E}\left[\left\langle \mathbf{x}_t-\mathbf{x}^*, \alpha_t\nabla_{\mathrm{r},t} + (1-\alpha_t)\nabla_{\mathrm{c},t}\right\rangle \mid \mathcal{Y}_t\right] \\ = -\eta_t\alpha_t\mathbb{E}\left\langle \mathbf{x}_t-\mathbf{x}^*, \nabla f_{\mu_{\mathrm{r}}}(\mathbf{x}_t)\right\rangle - \eta_t(1-\alpha_t)\mathbb{E}\left\langle \mathbf{x}_t-\mathbf{x}^*, \hat{\nabla}_{\mathrm{CGE}}f(\mathbf{x}_t)\right\rangle. \tag{S83}$$

Using Lemma A.4.4, we can upper bound

$$-\left\langle \mathbf{x}_t-\mathbf{x}^*, \nabla f_{\mu_{\mathrm{r}}}(\mathbf{x}_t)\right\rangle \leq -\left(f_{\mu_{\mathrm{r}}}(\mathbf{x}_t) - f_{\mu_{\mathrm{r}}}(\mathbf{x}^*)\right) - \frac{\bar{\sigma}}{2}\left\|\mathbf{x}_t-\mathbf{x}^*\right\|^2 \\ \leq -\left(f(\mathbf{x}_t) - f^*\right) + \mu_{\mathrm{r}}^2 L - \frac{\bar{\sigma}}{2}\left\|\mathbf{x}_t-\mathbf{x}^*\right\|^2, \tag{S84}$$

where, (S84) follows from [86, Lemma 5.1]. Also,

$$
\begin{aligned}
-\left\langle \mathbf{x}_t-\mathbf{x}^*, \hat{\nabla}_{\mathrm{CGE}} f(\mathbf{x}_t)\right\rangle &= -\left\langle \mathbf{x}_t-\mathbf{x}^*, \hat{\nabla}_{\mathrm{CGE}} f(\mathbf{x}_t)-\nabla f(\mathbf{x}_t)\right\rangle - \langle \mathbf{x}_t-\mathbf{x}^*, \nabla f(\mathbf{x}_t)\rangle \\
&\overset{(a)}{\leq} \|\mathbf{x}_t-\mathbf{x}^*\| \left\|\hat{\nabla}_{\mathrm{CGE}} f(\mathbf{x}_t)-\nabla f(\mathbf{x}_t)\right\| - (f(\mathbf{x}_t)-f(\mathbf{x}^*)) - \frac{\bar{\sigma}}{2}\|\mathbf{x}_t-\mathbf{x}^*\|^2 \\
&\overset{(b)}{\leq} \frac{\bar{\sigma}}{4}\|\mathbf{x}_t-\mathbf{x}^*\|^2 + \frac{1}{\bar{\sigma}}\left\|\hat{\nabla}_{\mathrm{CGE}} f(\mathbf{x}_t)-\nabla f(\mathbf{x}_t)\right\|^2 - (f(\mathbf{x}_t)-f(\mathbf{x}^*)) - \frac{\bar{\sigma}}{2}\|\mathbf{x}_t-\mathbf{x}^*\|^2 \\
&\overset{(c)}{\leq} \frac{1}{\bar{\sigma}} L^2 d \mu_{\mathrm{c}}^2 - (f(\mathbf{x}_t)-f^*) - \frac{\bar{\sigma}}{4}\|\mathbf{x}_t-\mathbf{x}^*\|^2, \qquad \text{(S85)}
\end{aligned}
$$

where (a) follows from Cauchy-Schwarz inequality, and strong convexity of f; (b) follows from Young's inequality $\mathbf{x}^T\mathbf{y} \leq \frac{a}{2}\|\mathbf{x}\|^2 + \frac{1}{2a}\|\mathbf{y}\|^2$; (c) follows from [86, Lemma 3] Substituting (S82), (S84), (S85) in (S81), we get the statement of the Lemma. ■

Next, we can upper bound $\mathbb{E}\|\nabla_{\mathrm{r},t}\|^2$, $\mathbb{E}\|\nabla_{\mathrm{c},t}\|^2$ using Proposition 5.4.2, and $p_{t,i} \geq \bar{c} > 0$ for all t, i.

Proposition A.4.6. *Suppose f satisfies Assumption 1, 2, 4. Then*

$$
\begin{aligned}
\mathbb{E}\|\nabla_{\mathrm{r},t}\|^2 &\leq 6(G^2+\sigma^2)\left(1+\frac{d}{n_{\mathrm{r}}}\right) \\
\mathbb{E}\|\nabla_{\mathrm{c},t}\|^2 &\leq 6\frac{(G^2+\sigma^2)}{\bar{c}}.
\end{aligned}
$$

PROOF:

$$
\begin{aligned}
\mathbb{E}\|\nabla_{\mathrm{r},t}\|^2 &\leq \left[2+\frac{4}{|\mathcal{B}^{\mathrm{r}}|}\left(1+\frac{d}{n_{\mathrm{r}}}\right)\right]\mathbb{E}\|\nabla f(\mathbf{x})\|^2 + \frac{4\sigma^2}{|\mathcal{B}^{\mathrm{r}}|}\left(1+\frac{d}{n_{\mathrm{r}}}\right) + \left(1+\frac{2}{|\mathcal{B}^{\mathrm{r}}|}+\frac{2}{n_{\mathrm{r}}|\mathcal{B}^{\mathrm{r}}|}\right)\frac{\mu_{\mathrm{r}}^2 L^2 d^2}{2} \\
&\overset{(a)}{\leq} 6\left(1+\frac{d}{n_{\mathrm{r}}}\right)\mathbb{E}\|\nabla f(\mathbf{x})\|^2 + 4\sigma^2\left(1+\frac{d}{n_{\mathrm{r}}}\right) + \left(\frac{3}{2}+\frac{1}{n_{\mathrm{r}}}\right)\mu_{\mathrm{r}}^2 L^2 d^2 \\
&\overset{(b)}{\leq} 6\left(1+\frac{d}{n_{\mathrm{r}}}\right)G^2 + 6\sigma^2\left(1+\frac{d}{n_{\mathrm{r}}}\right) \\
&= 6(G^2+\sigma^2)\left(1+\frac{d}{n_{\mathrm{r}}}\right), \qquad \text{(S86)}
\end{aligned}
$$

where (a) follows since $|\mathcal{B}_t^{\mathrm{r}}| \geq 1$; (b) follows from Assumption 4, and by using the smoothing parameter μ_{r} small enough such that $\sigma \geq \mu_{\mathrm{r}} L d$. Next,

$$\begin{aligned}
\mathbb{E}\left\|\nabla_{\mathrm{c},t}\right\|^2 &\leq \sum_{i=1}^{d} \frac{1}{p_{t,i}}\Big[2\mathbb{E}\left(\nabla f(\mathbf{x}_t)\right)_i^2 + \frac{3\zeta^2}{|\mathcal{B}_t^{\mathrm{c}}|} + \frac{L^2\mu_{\mathrm{c}}^2}{2}\left(1+\frac{3}{|\mathcal{B}_t^{\mathrm{c}}|}\right)\Big] \\
&\overset{(c)}{\leq} \frac{2\mathbb{E}\left\|\nabla f(\mathbf{x})\right\|^2}{\bar{c}} + \frac{3\sigma^2}{\bar{c}} + \frac{2L^2\mu_{\mathrm{c}}^2 d}{\bar{c}} \\
&\overset{(d)}{\leq} 6\frac{(G^2+\sigma^2)}{\bar{c}}, \qquad \text{(S87)}
\end{aligned}$$

where (c) follows since $p_{t,i} \geq \bar{c}$, $|\mathcal{B}_t^{\mathrm{c}}| \geq 1$, and $\sigma^2 = d\zeta^2$; (d) follows from Assumption 4, and by choosing μ_{c} small enough such that $\sigma \geq L\mu_{\mathrm{c}}\sqrt{d}$. ■

Substituting the bounds from Proposition A.4.6 in Proposition A.4.5, and assuming constant combination coefficient $\alpha_t = \alpha$, for all t, we get

$$\begin{aligned}
\mathbb{E}\left\|\mathbf{x}_{t+1}-\mathbf{x}^*\right\|^2 &\leq \left(1-\frac{\eta_t\bar{\sigma}}{2}\right)\mathbb{E}\left\|\mathbf{x}_t-\mathbf{x}^*\right\|^2 - 2\eta_t e_t + 2\eta_t\alpha L\mu_{\mathrm{r}}^2 + 2\eta_t(1-\alpha)\frac{L^2 d\mu_{\mathrm{c}}^2}{\bar{\sigma}} \\
&\quad + 12\eta_t^2(G^2+\sigma^2)\left[\alpha^2\left(1+\frac{d}{n_{\mathrm{r}}}\right) + (1-\alpha)^2\frac{1}{\bar{c}}\right], \\
&\overset{(e)}{\leq} \left(1-\frac{\eta_t\bar{\sigma}}{2}\right)\mathbb{E}\left\|\mathbf{x}_t-\mathbf{x}^*\right\|^2 - 2\eta_t e_t + 2\eta_t\frac{L^2 d\mu_{\mathrm{c}}^2}{\bar{\sigma}} + 12\eta_t^2(G^2+\sigma^2)\frac{\left(1+\frac{d}{n_{\mathrm{r}}}\right)}{1+\bar{c}\left(1+\frac{d}{n_{\mathrm{r}}}\right)}, \qquad \text{(S88)}
\end{aligned}$$

where (e) follows by choosing $\mu_{\mathrm{r}} = \mu_{\mathrm{c}}\sqrt{\frac{dL}{\bar{\sigma}}}$, and by choosing α to minimize the right hand side. Next, we state the following general result to help us bound (S88).

Lemma A.4.7. *Let* $\{a_t\}_{t\geq 0}, a_t \geq 0$, $\{e_t\}_{t\geq 0}, e_t \geq 0$ *be sequences satisfying*

$$a_{t+1} \leq \left(1-\frac{\eta_t\bar{\sigma}}{2}\right)a_t - \eta_t e_t + \eta_t A + \eta_t^2 B,$$

for $\eta_t = \frac{8}{\bar{\sigma}(a+t)}$, *and* $A, B \geq 0$, $\bar{\sigma} > 0, a > 1$. *Then,*

$$\frac{1}{S_T}\sum_{t=0}^{T-1} w_t e_t \leq A + \frac{4T(T+2a)}{\bar{\sigma}S_T}B + \frac{a^3\bar{\sigma}}{8S_T}a_0, \qquad \text{(S89)}$$

for $w_t = (a+t)^2$ *and* $S_T = \sum_{t=0}^{T-1} w_t \geq \frac{1}{3}T^3$.

PROOF: The proof borrows from the proof in [167, Lemma 3.3] with some minor modifications. Multiplying by $\frac{w_t}{\eta_t}$, and simplifying, we get

$$\sum_{t=0}^{T-1} w_t e_t \leq \frac{w_0}{\eta_0} a_0 + A \sum_{t=0}^{T-1} w_t + \sum_{t=0}^{T-1} w_t \eta_t B. \tag{S90}$$

Here, $\frac{w_0}{\eta_0} \leq \frac{\bar{\sigma} a^3}{8}$. Using $S_T = \sum_{t=0}^{T-1} w_t \geq T^3/3$, $\sum_{t=0}^{T-1} w_t \eta_t \leq \frac{4T(T+2a)}{\bar{\sigma}}$, we get the result. ■

Comparing (S88) and Lemma A.4.7, note that

$$e_t = \frac{1}{2}\mathbb{E}\left\|\mathbf{x}_t - \mathbf{x}^*\right\|^2, \qquad A = \frac{L^2 d\mu_{\mathrm{c}}^2}{\bar{\sigma}}, \qquad B = 6(G^2+\sigma^2)\frac{\left(1+\frac{d}{n_{\mathrm{r}}}\right)}{1+\bar{c}\left(1+\frac{d}{n_{\mathrm{r}}}\right)}.$$

Then, for $\hat{\mathbf{x}}_T = \frac{1}{S_T}\sum_{t=0}^{T-1} w_t \mathbf{x}_t$,

$$\begin{aligned}\mathbb{E}f(\hat{\mathbf{x}}_T) - f^* &\leq \frac{1}{S_T}\sum_{t=0}^{T-1} w_t e_t \leq A + \frac{4T(T+2a)}{\bar{\sigma} S_T} B + \frac{a^3\bar{\sigma}}{8S_T} a_0 \\ &\leq \mathcal{O}\left(\frac{L^2 d\mu_{\mathrm{c}}^2}{\bar{\sigma}} + \frac{(G^2+\sigma^2)}{\bar{\sigma}T}\frac{\left(1+\frac{d}{n_{\mathrm{r}}}\right)}{1+\bar{c}\left(1+\frac{d}{n_{\mathrm{r}}}\right)}\right).\end{aligned} \tag{S91}$$

The choice of $\mu_{\mathrm{c}} = \mathcal{O}\left(\sqrt{\frac{1}{dT}\frac{\left(1+\frac{d}{n_{\mathrm{r}}}\right)}{\left[1+\bar{c}\left(1+\frac{d}{n_{\mathrm{r}}}\right)\right]}}\right)$ in (S91) yields

$$\mathbb{E}f(\hat{\mathbf{x}}_T) - f^* \leq \mathcal{O}\left(\frac{(G^2+\sigma^2)}{\bar{\sigma}T}\frac{\left(1+\frac{d}{n_{\mathrm{r}}}\right)}{1+\bar{c}\left(1+\frac{d}{n_{\mathrm{r}}}\right)}\right). \tag{S92}$$

The special cases of Theorem 5.5.1 (for strongly convex functions) can be derived in a similar way, as we did for convex functions in Appendix A.4.8.

REFERENCES

[1] M. V. Afonso, J. M. Bioucas-Dias, and M. A. Figueiredo, “An augmented lagrangian approach to the constrained optimization formulation of imaging inverse problems,” *IEEE Transactions on Image Processing*, vol. 20, no. 3, pp. 681–695, 2010.

[2] A. Agarwal and L. Bottou, “A lower bound for the optimization of finite sums,” *arXiv preprint arXiv:1410.0723*, 2014.

[3] A. Agarwal, O. Dekel, and L. Xiao, “Optimal algorithms for online convex optimization with multi-point bandit feedback.” in *COLT*. Citeseer, 2010, pp. 28–40.

[4] N. Agarwal, A. Gonen, and E. Hazan, “Learning in non-convex games with an optimization oracle,” *arXiv preprint arXiv:1810.07362*, 2018.

[5] H. Aghajan and A. Cavallaro, *Multi-Camera Networks: Principles and Applications*. Academic press, 2009.

[6] A. F. Aji and K. Heafield, “Sparse communication for distributed gradient descent,” *arXiv preprint arXiv:1704.05021*, 2017.

[7] D. Alistarh, D. Grubic, J. Li, R. Tomioka, and M. Vojnovic, “QSGD: Communication-efficient SGD via gradient quantization and encoding,” in *Advances in Neural Information Processing Systems*, 2017, pp. 1709–1720.

[8] Z. Allen-Zhu and E. Hazan, “Variance reduction for faster non-convex optimization,” in *International conference on machine learning*, 2016, pp. 699–707.

[9] Y. Arjevani, Y. Carmon, J. C. Duchi, D. J. Foster, N. Srebro, and B. Woodworth, "Lower bounds for non-convex stochastic optimization," *arXiv preprint arXiv:1912.02365*, 2019.

[10] K. Balasubramanian and S. Ghadimi, "Zeroth-order (non)-convex stochastic optimization via conditional gradient and gradient updates," in *Advances in Neural Information Processing Systems*, 2018, pp. 3455–3464.

[11] K. Balasubramanian, S. Ghadimi, and A. Nguyen, "Stochastic multi-level composition optimization algorithms with level-independent convergence rates," *arXiv preprint arXiv: 2008.10526*, 2020.

[12] Y. Bar-Shalom, F. Daum, and J. Huang, "The probabilistic data association filter," *IEEE Control Syst. Mag.*, vol. 29, no. 6, pp. 82–100, 2009.

[13] Y. Bar-Shalom, P. K. Willett, and X. Tian, *Tracking and data fusion: A Handbook of Algorithms*. YBS publishing, 2011.

[14] D. Basu, D. Data, C. Karakus, and S. N. Diggavi, "Qsparse-local-sgd: Distributed sgd with quantization, sparsification, and local computations," *IEEE Journal on Selected Areas in Information Theory*, vol. 1, no. 1, pp. 217–226, 2020.

[15] G. Battistelli, L. Chisci, C. Fantacci, A. Farina, and A. Graziano, "Consensus CPHD Filter for Distributed Multitarget Tracking," *IEEE J. Sel. Topics Signal Process.*, vol. 7, no. 3, pp. 508–520, 2013.

[16] H. H. Bauschke and J. M. Borwein, "Joint and separate convexity of the bregman distance," in *Studies in Computational Mathematics*. Elsevier, 2001, vol. 8, pp. 23–36.

[17] A. Beck, *First-order methods in optimization*. SIAM, 2017.

[18] E. H. Bergou, E. Gorbunov, and P. Richtarik, "Stochastic three points method for unconstrained smooth minimization," *SIAM Journal on Optimization*, vol. 30, no. 4, pp. 2726–2749, 2020.

[19] J. Bernstein, Y.-X. Wang, K. Azizzadenesheli, and A. Anandkumar, “signsgd: Compressed optimisation for non-convex problems,” in *International Conference on Machine Learning*, 2018, pp. 560–569.

[20] O. Besbes, Y. Gur, and A. Zeevi, “Non-stationary stochastic optimization,” *Operations research*, vol. 63, no. 5, pp. 1227–1244, 2015.

[21] A. Bora, A. Jalal, E. Price, and A. G. Dimakis, “Compressed sensing using generative models,” in *International Conference on Machine Learning*. PMLR, 2017, pp. 537–546.

[22] D. M. Bortz and C. T. Kelley, “The simplex gradient and noisy optimization problems,” in *Computational methods for optimal design and control*. Springer, 1998, pp. 77–90.

[23] M. Brambilla, G. Soatti, and M. Nicoli, “Precise Vehicle Positioning by Cooperative Feature Association and Tracking in Vehicular Networks,” in *Proc. IEEE SSP*, 2018, pp. 648–652.

[24] F. Bullo, J. Cortes, and S. Martinez, *Distributed Control of Robotic Networks: A Mathematical Approach to Motion Coordination Algorithms*. Princeton University Press, 2009.

[25] X. Cao, J. Zhang, and H. V. Poor, “A virtual-queue-based algorithm for constrained online convex optimization with applications to data center resource allocation,” *IEEE Journal of Selected Topics in Signal Processing*, vol. 12, no. 4, pp. 703–716, 2018.

[26] N. Carlini and D. Wagner, “Towards evaluating the robustness of neural networks,” in *2017 ieee symposium on security and privacy (sp)*. IEEE, 2017, pp. 39–57.

[27] S. Cen, H. Zhang, Y. Chi, W. Chen, and T.-Y. Liu, “Convergence of distributed stochastic variance reduced methods without sampling extra data,” *arXiv preprint arXiv:1905.12648*, 2019.

[28] P.-Y. Chen, H. Zhang, Y. Sharma, J. Yi, and C.-J. Hsieh, “Zoo: Zeroth order optimization based black-box attacks to deep neural networks without training substitute models,” in

Proceedings of the 10th ACM Workshop on Artificial Intelligence and Security. ACM, 2017, pp. 15–26.

[29] T. Chen, G. Giannakis, T. Sun, and W. Yin, “LAG: Lazily aggregated gradient for communication-efficient distributed learning,” in *Advances in Neural Information Processing Systems*, 2018, pp. 5050–5060.

[30] T. Chen, Q. Ling, and G. B. Giannakis, “An online convex optimization approach to proactive network resource allocation,” *IEEE Transactions on Signal Processing*, vol. 65, no. 24, pp. 6350–6364, 2017.

[31] W.-N. Chen, P. Kairouz, and A. Özgür, “Breaking the communication-privacy-accuracy trilemma,” *arXiv preprint arXiv:2007.11707*, 2020.

[32] W. Chen, S. Horvath, and P. Richtarik, “Optimal client sampling for federated learning,” *arXiv preprint arXiv:2010.13723*, 2020.

[33] X. Chen, S. Liu, K. Xu, X. Li, X. Lin, M. Hong, and D. Cox, “Zo-adamm: Zeroth-order adaptive momentum method for black-box optimization,” in *Advances in Neural Information Processing Systems*, 2019, pp. 7204–7215.

[34] M. Cheng, T. Le, P.-Y. Chen, J. Yi, H. Zhang, and C.-J. Hsieh, “Query-efficient hard-label black-box attack: An optimization-based approach,” *arXiv preprint arXiv:1807.04457*, 2018.

[35] Y. J. Cho, J. Wang, and G. Joshi, “Client selection in federated learning: Convergence analysis and power-of-choice selection strategies,” *arXiv preprint arXiv:2010.01243*, 2020.

[36] A. R. Conn, N. I. Gould, and P. L. Toint, *Trust region methods*. SIAM, 2000.

[37] A. R. Conn, K. Scheinberg, and L. N. Vicente, *Introduction to derivative-free optimization*. SIAM, 2009.

[38] A. Cutkosky and F. Orabona, "Momentum-based variance reduction in non-convex SGD," in *Advances in Neural Information Processing Systems 32*. Curran Associates, Inc., 2019, pp. 15 236–15 245.

[39] R. Das, A. Acharya, A. Hashemi, S. Sanghavi, I. S. Dhillon, and U. Topcu, "Faster non-convex federated learning via global and local momentum," *arXiv preprint arXiv:2012.04061*, 2020.

[40] R. Das, A. Hashemi, S. Sanghavi, and I. S. Dhillon, "Improved convergence rates for non-convex federated learning with compression," *arXiv preprint arXiv:2012.04061*, 2020.

[41] D. Data and S. Diggavi, "Byzantine-resilient high-dimensional sgd with local iterations on heterogeneous data," *arXiv preprint arXiv:2006.13041*, 2020.

[42] A. Defazio, F. Bach, and S. Lacoste-Julien, "SAGA: A fast incremental gradient method with support for non-strongly convex composite objectives," in *Advances in neural information processing systems*, 2014, pp. 1646–1654.

[43] ——, "Saga: A fast incremental gradient method with support for non-strongly convex composite objectives," in *Advances in neural information processing systems*, 2014, pp. 1646–1654.

[44] O. Dekel, R. Gilad-Bachrach, O. Shamir, and L. Xiao, "Optimal distributed online prediction using mini-batches," *Journal of Machine Learning Research*, vol. 13, no. Jan, pp. 165–202, 2012.

[45] E. Delande, E. Duflos, P. Vanheeghe, and D. Heurguier, "Multi-sensor PHD: Construction and Implementation by Space Partitioning," in *Proc. IEEE ICASSP*, 2011, pp. 3632–3635.

[46] A. G. Dimakis, S. Kar, J. M. F. Moura, M. G. Rabbat, and A. Scaglione, "Gossip algorithms for distributed signal processing," *Proc. IEEE*, vol. 98, no. 11, pp. 1847–1864, 2010.

[47] A. Doucet, N. de Freitas, and N. Gordon, *Sequantial Monte Carlo Methods in Practice*. New York City, NY: Springer-Verlag, 2001.

[48] N. Dryden, T. Moon, S. A. Jacobs, and B. Van Essen, "Communication quantization for data-parallel training of deep neural networks," in *2016 2nd Workshop on Machine Learning in HPC Environments (MLHPC)*. IEEE, 2016, pp. 1–8.

[49] J. Duchi, E. Hazan, and Y. Singer, "Adaptive subgradient methods for online learning and stochastic optimization." *Journal of machine learning research*, vol. 12, no. 7, 2011.

[50] J. C. Duchi, M. I. Jordan, M. J. Wainwright, and A. Wibisono, "Optimal rates for zero-order convex optimization: The power of two function evaluations," *IEEE Transactions on Information Theory*, vol. 61, no. 5, pp. 2788–2806, 2015.

[51] A. Fallah, A. Mokhtari, and A. Ozdaglar, "Personalized federated learning with theoretical guarantees: A model-agnostic meta-learning approach," *Advances in Neural Information Processing Systems*, vol. 33, 2020.

[52] C. Fang, C. J. Li, Z. Lin, and T. Zhang, "SPIDER: Near-optimal non-convex optimization via stochastic path-integrated differential estimator," in *Advances in Neural Information Processing Systems*, 2018, pp. 689–699.

[53] ——, "Spider: Near-optimal non-convex optimization via stochastic path-integrated differential estimator," in *Advances in Neural Information Processing Systems*, 2018, pp. 689–699.

[54] C. Finn, P. Abbeel, and S. Levine, "Model-agnostic meta-learning for fast adaptation of deep networks," in *International Conference on Machine Learning*. PMLR, 2017, pp. 1126–1135.

[55] A. D. Flaxman, A. T. Kalai, and H. B. McMahan, “Online convex optimization in the bandit setting: gradient descent without a gradient,” in *Proceedings of the sixteenth annual ACM-SIAM symposium on Discrete algorithms*, 2005, pp. 385–394.

[56] Y. Fraboni, R. Vidal, L. Kameni, and M. Lorenzi, “Clustered sampling: Low-variance and improved representativity for clients selection in federated learning,” *arXiv preprint arXiv:2105.05883*, 2021.

[57] M. Frohle, C. Lindberg, and H. Wymeersch, “Cooperative Localization of Vehicles without Inter-Vehicle Measurements,” in *Proc. IEEE WCNC*, 2018, pp. 1–6.

[58] R. G. Gallager, *Stochastic Processes: Theory for Applications.* Cambridge, UK: Cambridge University Press, 2013.

[59] X. Gao, X. Li, and S. Zhang, “Online learning with non-convex losses and non-stationary regret,” in *International Conference on Artificial Intelligence and Statistics*, 2018, pp. 235–243.

[60] X. Gao, B. Jiang, and S. Zhang, “On the information-adaptive variants of the admm: an iteration complexity perspective,” *Journal of Scientific Computing*, vol. 76, no. 1, pp. 327–363, 2018.

[61] S. Ghadimi and G. Lan, “Stochastic first-and zeroth-order methods for nonconvex stochastic programming,” *SIAM Journal on Optimization*, vol. 23, no. 4, pp. 2341–2368, 2013.

[62] ——, “Stochastic first-and zeroth-order methods for nonconvex stochastic programming,” *SIAM Journal on Optimization*, vol. 23, no. 4, pp. 2341–2368, 2013.

[63] S. Ghadimi, A. Ruszczynski, and M. Wang, “A single timescale stochastic approximation method for nested stochastic optimization,” *SIAM Journal on Optimization*, vol. 30, no. 1, pp. 960–979, 2020.

[64] A. Girgis, D. Data, S. Diggavi, P. Kairouz, and A. T. Suresh, “Shuffled model of differential privacy in federated learning,” in *International Conference on Artificial Intelligence and Statistics*. PMLR, 2021, pp. 2521–2529.

[65] A. M. Girgis, D. Data, S. Diggavi, P. Kairouz, and A. T. Suresh, “Shuffled model of federated learning: Privacy, accuracy and communication trade-offs,” *IEEE Journal on Selected Areas in Information Theory*, vol. 2, no. 1, pp. 464–478, 2021.

[66] I. Goodfellow, J. Shlens, and C. Szegedy, “Explaining and harnessing adversarial examples,” *2015 ICLR*, vol. arXiv preprint arXiv:1412.6572, 2015.

[67] E. Gorbunov, K. Burlachenko, Z. Li, and P. Richtárik, “Marina: Faster non-convex distributed learning with compression,” *arXiv preprint arXiv:2102.07845*, 2021.

[68] R. M. Gower, M. Schmidt, F. Bach, and P. Richtarik, “Variance-reduced methods for machine learning,” *Proceedings of the IEEE*, vol. 108, no. 11, pp. 1968–1983, 2020.

[69] B. Gu, Z. Huo, and H. Huang, “Asynchronous stochastic block coordinate descent with variance reduction,” *arXiv preprint arXiv:1610.09447*, 2016.

[70] F. Haddadpour, M. M. Kamani, M. Mahdavi, and V. Cadambe, “Trading redundancy for communication: Speeding up distributed SGD for non-convex optimization,” in *International Conference on Machine Learning*, 2019, pp. 2545–2554.

[71] F. Haddadpour, M. M. Kamani, A. Mokhtari, and M. Mahdavi, “Federated learning with compression: Unified analysis and sharp guarantees,” *arXiv preprint arXiv:2007.01154*, 2020.

[72] D. Hajinezhad, M. Hong, and A. Garcia, “Zone: Zeroth-order nonconvex multiagent optimization over networks,” *IEEE Transactions on Automatic Control*, vol. 64, no. 10, pp. 3995–4010, 2019.

[73] E. C. Hall and R. M. Willett, “Online convex optimization in dynamic environments,” *IEEE Journal of Selected Topics in Signal Processing*, vol. 9, no. 4, pp. 647–662, 2015.

[74] O. A. Hanna, Y. H. Ezzeldin, C. Fragouli, and S. Diggavi, “Quantizing data for distributed learning,” *arXiv preprint arXiv:2012.07913*, 2020.

[75] E. Hazan, A. Agarwal, and S. Kale, “Logarithmic regret algorithms for online convex optimization,” *Machine Learning*, vol. 69, no. 2-3, pp. 169–192, 2007.

[76] E. Hazan *et al.*, “Introduction to online convex optimization,” *Foundations and Trends® in Optimization*, vol. 2, no. 3-4, pp. 157–325, 2016.

[77] E. Hazan, K. Singh, and C. Zhang, “Efficient regret minimization in non-convex games,” *arXiv preprint arXiv:1708.00075*, 2017.

[78] K. He, X. Zhang, S. Ren, and J. Sun, “Deep residual learning for image recognition,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 770–778.

[79] P. Horridge and S. Maskell, “Real-Time Tracking Of Hundreds Of Targets With Efficient Exact JPDAF Implementation,” in *Proc. Int. Conf. Inf. Fusion*, 2006, pp. 1–8.

[80] A. Ilyas, L. Engstrom, A. Athalye, and J. Lin, “Black-box adversarial attacks with limited queries and information,” *arXiv preprint arXiv:1804.08598*, 2018.

[81] A. Ilyas, L. Engstrom, and A. Madry, “Prior convictions: Black-box adversarial attacks with bandits and priors,” in *International Conference on Learning Representations*, 2019. [Online]. Available: https://openreview.net/forum?id=BkMiWhR5K7

[82] V. N. Ioannidis, A. S. Zamzam, G. B. Giannakis, and N. D. Sidiropoulos, “Coupled graph and tensor factorization for recommender systems and community detection,” *IEEE Transactions on Knowledge and Data Engineering*, 2019.

[83] A. Jadbabaie, A. Rakhlin, S. Shahrampour, and K. Sridharan, "Online optimization: Competing with dynamic comparators," in *Artificial Intelligence and Statistics*, 2015, pp. 398–406.

[84] P. Jain, P. Kar *et al.*, "Non-convex optimization for machine learning," *Foundations and Trends® in Machine Learning*, vol. 10, no. 3-4, pp. 142–336, 2017.

[85] R. Jenatton, J. Huang, and C. Archambeau, "Adaptive algorithms for online convex optimization with long-term constraints," in *International Conference on Machine Learning*, 2016, pp. 402–411.

[86] K. Ji, Z. Wang, Y. Zhou, and Y. Liang, "Improved zeroth-order variance reduced algorithms and analysis for nonconvex optimization," *arXiv preprint arXiv:1910.12166*, 2019.

[87] P. Jiang and G. Agrawal, "A linear speedup analysis of distributed deep learning with sparse and quantized communication," in *Advances in Neural Information Processing Systems*, 2018, pp. 2525–2536.

[88] Y. Jiang, J. Konečnỳ, K. Rush, and S. Kannan, "Improving federated learning personalization via model agnostic meta learning," *arXiv preprint arXiv:1909.12488*, 2019.

[89] M. J. Johnson, J. Saunderson, and A. S. Willsky, "Analyzing Hogwild parallel Gaussian Gibbs sampling," in *Proc. Int. Conf. Neural Inf. Process. Syst.*, 2013, pp. 2715–2723.

[90] R. Johnson and T. Zhang, "Accelerating stochastic gradient descent using predictive variance reduction," in *Advances in neural information processing systems*, 2013, pp. 315–323.

[91] ——, "Accelerating stochastic gradient descent using predictive variance reduction," in *Advances in Neural Information Processing Systems 26*. Curran Associates, Inc., 2013, pp. 315–323.

[92] P. Kairouz, H. B. McMahan, B. Avent, A. Bellet, M. Bennis, A. N. Bhagoji, K. Bonawitz, Z. Charles, G. Cormode, R. Cummings *et al.*, "Advances and open problems in federated learning," *arXiv preprint arXiv:1912.04977*, 2019.

[93] S. P. Karimireddy, M. Jaggi, S. Kale, M. Mohri, S. J. Reddi, S. U. Stich, and A. T. Suresh, "Mime: Mimicking centralized stochastic algorithms in federated learning," *arXiv preprint arXiv:2008.03606*, 2020.

[94] S. P. Karimireddy, S. Kale, M. Mohri, S. Reddi, S. Stich, and A. T. Suresh, "Scaffold: Stochastic controlled averaging for federated learning," in *International Conference on Machine Learning*. PMLR, 2020, pp. 5132–5143.

[95] P. Khanduri, P. Sharma, H. Yang, M. Hong, J. Liu, K. Rajawat, and P. K. Varshney, "Stem: A stochastic two-sided momentum algorithm achieving near-optimal sample and communication complexities for federated learning," *arXiv preprint arXiv:2106.10435*, 2021.

[96] J. Kiefer, J. Wolfowitz *et al.*, "Stochastic estimation of the maximum of a regression function," *The Annals of Mathematical Statistics*, vol. 23, no. 3, pp. 462–466, 1952.

[97] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *International Conference on Learning Representations (ICLR)*, 2015.

[98] J. Konečnỳ, H. B. McMahan, D. Ramage, and P. Richtárik, "Federated optimization: Distributed machine learning for on-device intelligence," *arXiv preprint arXiv:1610.02527*, 2016.

[99] A. Koppel, F. Y. Jakubiec, and A. Ribeiro, "A saddle point algorithm for networked online convex optimization," *IEEE Transactions on Signal Processing*, vol. 63, no. 19, pp. 5149–5164, 2015.

[100] A. Krizhevsky, G. Hinton *et al.*, "Learning multiple layers of features from tiny images," 2009.

[101] F. R. Kschischang, B. J. Frey, and H.-A. Loeliger, “Factor Graphs and the Sum-Product Algorithm,” *IEEE Trans. Inf. Theory*, vol. 47, no. 2, pp. 498–519, 2001.

[102] J. Larson, M. Menickelly, and S. M. Wild, “Derivative-free optimization methods,” *arXiv preprint arXiv:1904.11585*, 2019.

[103] T. Léauté and B. Faltings, “Protecting privacy through distributed computation in multi-agent decision making,” *Journal of Artificial Intelligence Research*, vol. 47, pp. 649–695, 2013.

[104] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, “Gradient-based learning applied to document recognition,” *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.

[105] J. D. Lee, Q. Lin, T. Ma, and T. Yang, “Distributed stochastic variance reduced gradient methods by sampling extra data with replacement,” *The Journal of Machine Learning Research*, vol. 18, no. 1, pp. 4404–4446, 2017.

[106] L. Lei and M. I. Jordan, “Less than a single pass: Stochastically controlled stochastic gradient method,” *arXiv preprint arXiv:1609.03261*, 2016.

[107] M. Li, D. G. Andersen, A. J. Smola, and K. Yu, “Communication efficient distributed machine learning with the parameter server,” in *Advances in Neural Information Processing Systems*, 2014, pp. 19–27.

[108] T. Li, S. Hu, A. Beirami, and V. Smith, “Ditto: Fair and robust federated learning through personalization,” *arXiv: 2012.04221*, 2020.

[109] T. Li, A. K. Sahu, A. Talwalkar, and V. Smith, “Federated learning: Challenges, methods, and future directions,” *IEEE Signal Processing Magazine*, vol. 37, no. 3, pp. 50–60, 2020.

[110] T. Li, M. Sanjabi, A. Beirami, and V. Smith, “Fair resource allocation in federated learning,” *arXiv preprint arXiv:1905.10497*, 2019.

[111] X. Li, X. Yi, and L. Xie, "Distributed online optimization for multi-agent networks with coupled inequality constraints," *arXiv preprint arXiv:1805.05573*, 2018.

[112] X. Lian, M. Wang, and J. Liu, "Finite-sum composition optimization via variance reduced gradient descent," in *Artificial Intelligence and Statistics*. PMLR, 2017, pp. 1159–1167.

[113] X. Lian, H. Zhang, C.-J. Hsieh, Y. Huang, and J. Liu, "A comprehensive linear speedup analysis for asynchronous stochastic parallel optimization from zeroth-order to first-order," in *Advances in Neural Information Processing Systems*, 2016, pp. 3054–3062.

[114] T. Lin, S. U. Stich, K. K. Patel, and M. Jaggi, "Don't use large mini-batches, use local SGD," *arXiv preprint arXiv:1808.07217*, 2018.

[115] D. Liu, L. M. Nguyen, and Q. Tran-Dinh, "An optimal hybrid variance-reduced algorithm for stochastic composite nonconvex optimization," *arXiv preprint arXiv:2008.09055*, 2020.

[116] J. Liu and C. Zhang, "Distributed learning systems with first-order methods," *arXiv preprint arXiv:2104.05245*, 2021.

[117] S. Liu, P. Y. Chen, X. Chen, and M. Hong, "Signsgd via zeroth-order oracle," in *7th International Conference on Learning Representations, ICLR 2019*, 2019.

[118] S. Liu, P.-Y. Chen, B. Kailkhura, G. Zhang, A. Hero, and P. K. Varshney, "A primer on zeroth-order optimization in signal processing and machine learning," *arXiv preprint arXiv:2006.06224*, 2020.

[119] S. Liu, B. Kailkhura, P.-Y. Chen, P. Ting, S. Chang, and L. Amini, "Zeroth-order stochastic variance reduction for nonconvex optimization," in *Advances in Neural Information Processing Systems*, 2018, pp. 3727–3737.

[120] S. Liu, S. Lu, X. Chen, Y. Feng, K. Xu, A. Al-Dujaili, M. Hong, and U.-M. O'Reilly, "Min-max optimization without gradients: Convergence and applications to black-box evasion and poisoning attacks," in *Proc. Int. Conf. Machine Learning*, 2020.

[121] M. Mahdavi, R. Jin, and T. Yang, “Trading regret for efficiency: online convex optimization with long term constraints,” *Journal of Machine Learning Research*, vol. 13, no. Sep, pp. 2503–2528, 2012.

[122] R. P. Mahler, *Statistical multisource-multitarget information fusion*. Norwood, MA, USA: Artech House, Inc., 2007.

[123] B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y Arcas, “Communication-efficient learning of deep networks from decentralized data,” in *Artificial Intelligence and Statistics*. PMLR, 2017, pp. 1273–1282.

[124] F. Meyer, T. Kropfreiter, J. L. Williams, R. Lau, F. Hlawatsch, P. Braca, and M. Z. Win, “Message Passing Algorithms for Scalable Multitarget Tracking,” *Proc. IEEE*, vol. 106, no. 2, pp. 221–259, 2018.

[125] F. Meyer, P. Braca, F. Hlawatsch, M. Micheli, and K. D. LePage, “Scalable Adaptive Multitarget Tracking using Multiple Sensors,” in *Proc. IEEE Globecom Workshops*, 2016, pp. 1–6.

[126] F. Meyer, P. Braca, P. Willett, and F. Hlawatsch, “A scalable algorithm for tracking an unknown number of targets using multiple sensors,” *IEEE Trans. Signal Process.*, vol. 65, no. 13, pp. 3478–3493, 2017.

[127] F. Meyer, O. Hlinka, H. Wymeersch, E. Riegler, and F. Hlawatsch, “Distributed localization and tracking of mobile networks including noncooperative objects,” *IEEE Trans. Signal Inf. Process. Netw.*, vol. 2, no. 1, pp. 57–71, 2016.

[128] F. Meyer, Z. Liu, and M. Z. Win, “Network Localization and Navigation Using Measurements with Uncertain Origin,” in *Proc. Int. Conf. Inf. Fusion*, 2018, pp. 1–7.

[129] F. Meyer and M. Z. Win, “Joint Navigation and Multitarget Tracking in Networks,” in *Proc. IEEE ICC Workshops*, 2018, pp. 1–6.

[130] A. Mitra, R. Jaafar, G. J. Pappas, and H. Hassani, “Achieving linear convergence in federated learning under objective and systems heterogeneity,” *arXiv preprint arXiv:2102.07053*, 2021.

[131] S.-M. Moosavi-Dezfooli, A. Fawzi, O. Fawzi, and P. Frossard, “Universal adversarial perturbations,” in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 1765–1773.

[132] J. F. Mota, J. M. Xavier, P. M. Aguiar, and M. Püschel, “D-admm: A communication-efficient distributed algorithm for separable optimization,” *IEEE Transactions on Signal Processing*, vol. 61, no. 10, pp. 2718–2723, 2013.

[133] K. G. Murty and S. N. Kabadi, “Some np-complete problems in quadratic and nonlinear programming,” *Mathematical Programming*, vol. 39, no. 1-2, pp. 117–129, 1987.

[134] S. Nannuru, S. Blouin, M. Coates, and M. Rabbat, “Multisensor CPHD Filter,” *IEEE Trans. Aerosp. Electron. Syst.*, vol. 52, no. 4, pp. 1834–1854, 2016.

[135] S. Nannuru, M. Coates, M. Rabbat, and S. Blouin, “General Solution and Approximate Implementation of the Multisensor Multitarget CPHD Filter,” in *Proc. IEEE ICASSP*, 2015, pp. 4055–4059.

[136] Y. Nesterov, “Introductory lectures on convex programming volume i: Basic course,” *Lecture notes*, vol. 3, no. 4, p. 5, 1998.

[137] Y. Nesterov and V. Spokoiny, “Random gradient-free minimization of convex functions,” *Foundations of Computational Mathematics*, vol. 17, no. 2, pp. 527–566, 2017.

[138] L. M. Nguyen, J. Liu, K. Scheinberg, and M. Takáč, “SARAH: A novel method for machine learning problems using stochastic recursive gradient,” in *Proceedings of the 34th International Conference on Machine Learning-Volume 70*. JMLR. org, 2017, pp. 2613–2621.

[139] ——, “Sarah: A novel method for machine learning problems using stochastic recursive gradient,” in *Proceedings of the 34th International Conference on Machine Learning-Volume 70*. JMLR. org, 2017, pp. 2613–2621.

[140] ——, “Stochastic recursive gradient algorithm for nonconvex optimization,” *arXiv preprint arXiv:1705.07261*, 2017.

[141] L. M. Nguyen, M. van Dijk, D. T. Phan, P. H. Nguyen, T.-W. Weng, and J. R. Kalagnanam, “Optimal finite-sum smooth non-convex optimization with SARAH,” *arXiv preprint arXiv:1901.07648*, 2019.

[142] R. Olfati-Saber, J. A. Fax, and R. M. Murray, “Consensus and Cooperation in Networked Multi-agent Systems,” *Proc. IEEE*, vol. 95, no. 1, pp. 215–233, 2007.

[143] U. Orguner and M. Demırekler, “Analysis of single Gaussian approximation of Gaussian mixtures in Bayesian filtering applied to mixed multiple-model estimation,” *Int. J. Control*, vol. 80, no. 6, pp. 952–967, 2007.

[144] S. Paternain, S. Lee, M. M. Zavlanos, and A. Ribeiro, “Distributed constrained online learning,” *arXiv preprint arXiv:1903.06310*, 2019.

[145] S. Paternain and A. Ribeiro, “Online learning of feasible strategies in unknown environments,” *IEEE Transactions on Automatic Control*, vol. 62, no. 6, pp. 2807–2822, 2016.

[146] S. J. Reddi, A. Hefny, S. Sra, B. Poczos, and A. Smola, “Stochastic Variance Reduction for Nonconvex Optimization,” in *Int. Conf. Machine Learn.*, 2016, pp. 314–323.

[147] S. J. Reddi, A. Hefny, S. Sra, B. Póczos, and A. Smola, “Stochastic variance reduction for nonconvex optimization,” in *International conference on machine learning*, 2016, pp. 314–323.

[148] S. J. Reddi, J. Konečnỳ, P. Richtárik, B. Póczós, and A. Smola, “AIDE: Fast and communication efficient distributed optimization,” *arXiv preprint arXiv:1608.06879*, 2016.

[149] A. Reisizadeh, A. Mokhtari, H. Hassani, A. Jadbabaie, and R. Pedarsani, "Fedpaq: A communication-efficient federated learning method with periodic averaging and quantization," in *International Conference on Artificial Intelligence and Statistics*. PMLR, 2020, pp. 2021–2031.

[150] B. Ristić, S. Arulampalam, and N. Gordon, *Beyond the Kalman Filter: Particle Filters for Tracking Applications*. Artech House, 2004.

[151] E. Rizk, S. Vlaski, and A. H. Sayed, "Federated learning under importance sampling," *arXiv preprint arXiv:2012.07383*, 2020.

[152] H. Robbins and S. Monro, "A stochastic approximation method," *The annals of mathematical statistics*, pp. 400–407, 1951.

[153] A. A. Saucan, M. J. Coates, and M. Rabbat, "A multisensor multi-Bernoulli filter," *IEEE Trans. Signal Process.*, vol. 65, no. 20, pp. 5495–5509, 2017.

[154] A. A. Saucan and P. K. Varshney, "Distributed cross-entropy δ-GLMB filter for multi-sensor multi-target tracking," in *Int. Conf. Inform. Fusion (FUSION)*, 2018.

[155] D. Schuhmacher, B.-T. Vo, and B.-N. Vo, "A Consistent Metric for Performance Evaluation of Multi-Object Filters," *IEEE Trans. Signal Process.*, vol. 56, no. 8, pp. 3447–3457, 2008.

[156] S. Shahrampour and A. Jadbabaie, "Distributed online optimization in dynamic environments using mirror descent," *IEEE Transactions on Automatic Control*, vol. 63, no. 3, pp. 714–725, 2017.

[157] S. Shalev-Shwartz *et al.*, "Online learning and online convex optimization," *Foundations and Trends® in Machine Learning*, vol. 4, no. 2, pp. 107–194, 2012.

[158] O. Shamir, "An optimal algorithm for bandit and zero-order convex optimization with two-point feedback," *The Journal of Machine Learning Research*, vol. 18, no. 1, pp. 1703–1713, 2017.

[159] O. Shamir and N. Srebro, “Distributed stochastic optimization and learning,” in *2014 52nd Annual Allerton Conference on Communication, Control, and Computing (Allerton)*. IEEE, 2014, pp. 850–857.

[160] O. Shamir, N. Srebro, and T. Zhang, “Communication-efficient distributed optimization using an approximate newton-type method,” in *International conference on machine learning*, 2014, pp. 1000–1008.

[161] N. Singh, D. Data, J. George, and S. Diggavi, “Sparq-sgd: Event-triggered and compressed communication in decentralized stochastic optimization,” *arXiv preprint arXiv:1910.14280*, 2019.

[162] A. Smola and S. Narayanamurthy, “An architecture for parallel topic models,” *Proc. VLDB Endow.*, vol. 3, no. 1-2, pp. 703–710, 2010.

[163] G. Soatti, M. Nicoli, N. Garcia, B. Denis, R. Raulefs, and H. Wymeersch, “Enhanced Vehicle Positioning in Cooperative ITS by Joint Sensing of Passive Features,” in *Proc. IEEE ITSC*, 2017, pp. 1–6.

[164] ——, “Implicit Cooperative Positioning in Vehicular Networks,” *IEEE Trans. Intell. Transp. Syst.*, no. 99, pp. 1–17, 2018.

[165] G. Soldi and P. Braca, “Online Estimation of Unknown Parameters in Multisensor-Multitarget Tracking: a Belief Propagation Approach,” in *Proc. Int. Conf. Inf. Fusion*, 2018, pp. 2151–2157.

[166] S. U. Stich, “Local SGD converges fast and communicates little,” *arXiv preprint arXiv:1805.09767*, 2018.

[167] S. U. Stich, J.-B. Cordonnier, and M. Jaggi, “Sparsified sgd with memory,” in *Advances in Neural Information Processing Systems*, 2018, pp. 4447–4458.

[168] E. B. Sudderth, A. T. Ihler, M. Isard, W. T. Freeman, and A. S. Willsky, "Nonparametric belief propagation," *Commun. ACM*, vol. 53, no. 10, pp. 95–103, 2010.

[169] A. S. Suggala and P. Netrapalli, "Online non-convex learning: Following the perturbed leader is optimal," *arXiv preprint arXiv:1903.08110*, 2019.

[170] H. Sun, S. Lu, and M. Hong, "Improving the sample and communication complexity for decentralized non-convex optimization: A joint gradient estimation and tracking approach," *arXiv preprint arXiv:1910.05857*, 2019.

[171] W. Sun, D. Dey, and A. Kapoor, "Safety-aware algorithms for adversarial contextual bandit," in *Proceedings of the 34th International Conference on Machine Learning-Volume 70*. JMLR. org, 2017, pp. 3280–3288.

[172] B. Swenson, R. Murray, H. V. Poor, and S. Kar, "Distributed stochastic gradient descent: Nonconvexity, nonsmoothness, and convergence to local minima," *arXiv preprint arXiv:2003.02818*, 2020.

[173] Y. Tang and N. Li, "Distributed zero-order algorithms for nonconvex multi-agent optimization," in *2019 57th Annual Allerton Conference on Communication, Control, and Computing (Allerton)*. IEEE, 2019, pp. 781–786.

[174] A. Terenin, D. Simpson, and D. Draper, "Asynchronous Gibbs sampling," *arXiv:1509.08999v5*, 2018.

[175] Q. Tran-Dinh, N. H. Pham, D. T. Phan, and L. M. Nguyen, "Hybrid stochastic gradient descent algorithms for stochastic nonconvex optimization," *arXiv preprint arXiv:1905.05920*, 2019.

[176] ——, "A hybrid stochastic optimization framework for composite nonconvex optimization," *arXiv preprint arXiv:1907.03793*, 2019.

[177] C.-C. Tu, P. Ting, P.-Y. Chen, S. Liu, H. Zhang, J. Yi, C.-J. Hsieh, and S.-M. Cheng, "Autozoom: Autoencoder-based zeroth order optimization method for attacking black-box neural networks," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 33, 2019, pp. 742–749.

[178] J. Vermaak, S. J. Godsill, and P. Perez, "Monte Carlo filtering for multi target tracking and data association," *IEEE Trans. Aerosp. Electron. Syst.*, vol. 41, no. 1, pp. 309–332, 2005.

[179] S. Vlaski and A. H. Sayed, "Distributed learning in non-convex environments—part i: Agreement at a linear rate," *IEEE Transactions on Signal Processing*, vol. 69, pp. 1242–1256, 2021.

[180] B.-N. Vo, B.-T. Vo, and D. Phung, "Labeled random finite sets and the Bayes multi-target tracking filter," *IEEE Trans. Signal Process.*, vol. 62, no. 24, pp. 6554–6567, 2014.

[181] B.-N. Vo and W.-K. Ma, "The Gaussian Mixture Probability Hypothesis Density Filter," *IEEE Trans. Signal Process.*, vol. 54, no. 11, pp. 4091–4104, 2006.

[182] J. Wang, W. Wang, and N. Srebro, "Memory and communication efficient distributed stochastic optimization with minibatch prox," in *Conference on Learning Theory*. PMLR, 2017, pp. 1882–1919.

[183] ——, "Memory and communication efficient distributed stochastic optimization with minibatch-prox," *arXiv preprint arXiv:1702.06269*, 2017.

[184] J. Wang, Q. Liu, H. Liang, G. Joshi, and H. V. Poor, "Tackling the objective inconsistency problem in heterogeneous federated optimization," *arXiv preprint arXiv:2007.07481*, 2020.

[185] M. Wang, E. X. Fang, and H. Liu, "Stochastic compositional gradient descent: algorithms for minimizing compositions of expected-value functions," *Mathematical Programming*, vol. 161, no. 1-2, pp. 419–449, 2017.

[186] M. Wang, J. Liu, and E. X. Fang, “Accelerating stochastic composition optimization,” *Journal of Machine Learning Research*, 2017.

[187] Z. Wang, K. Ji, Y. Zhou, Y. Liang, and V. Tarokh, “SpiderBoost: A class of faster variance-reduced algorithms for nonconvex optimization,” *arXiv preprint arXiv:1810.10690*, 2018.

[188] J. Wangni, J. Wang, J. Liu, and T. Zhang, “Gradient sparsification for communication-efficient distributed optimization,” in *Advances in Neural Information Processing Systems*, 2018, pp. 1299–1309.

[189] W. Wen, C. Xu, F. Yan, C. Wu, Y. Wang, Y. Chen, and H. Li, “TernGrad: Ternary gradients to reduce communication in distributed deep learning,” in *Advances in neural information processing systems*, 2017, pp. 1509–1519.

[190] J. Williams and R. Lau, “Approximate Evaluation of Marginal Association Probabilities with Belief Propagation,” *IEEE Trans. Aerosp. Electron. Syst.*, vol. 50, no. 4, pp. 2942–2959, 2014.

[191] J. L. Williams and R. A. Lau, “Convergence of Loopy Belief Propagation for Data Association,” in *Proc. IEEE ISSNIP*, 2010, pp. 175–180.

[192] H. Wymeersch, J. Lien, and M. Z. Win, “Cooperative localization in wireless networks,” *Proc. IEEE*, vol. 97, no. 2, pp. 427–450, 2009.

[193] L. Xiao, S. Boyd, and S. Lall, “A scheme for robust distributed sensor fusion based on average consensus,” in *Proc. Int. Symp. Inf. Process. Sensor Netw.*, 2005, pp. 63–70.

[194] E. P. Xing, Q. Ho, W. Dai, J. K. Kim, J. Wei, S. Lee, X. Zheng, P. Xie, A. Kumar, and Y. Yu, “Petuum: A new platform for distributed machine learning on big data,” *IEEE Transactions on Big Data*, vol. 1, no. 2, pp. 49–67, June 2015.

[195] E. P. Xing, Q. Ho, P. Xie, and D. Wei, “Strategies and principles of distributed machine learning on big data,” *Engineering*, vol. 2, no. 2, pp. 179–195, 2016.

[196] K. Xu, S. Liu, P. Zhao, P.-Y. Chen, H. Zhang, Q. Fan, D. Erdogmus, Y. Wang, and X. Lin, "Structured adversarial attack: Towards general implementation and better interpretability," in *International Conference on Learning Representations*, 2019.

[197] L. Yang, L. Deng, M. H. Hajiesmaili, C. Tan, and W. S. Wong, "An optimal algorithm for online non-convex learning," *Proceedings of the ACM on Measurement and Analysis of Computing Systems*, vol. 2, no. 2, pp. 1–25, 2018.

[198] J. Yedidia, W. Freeman, and Y. Weiss, "Constructing free energy approximations and generalized belief propagation algorithms," *IEEE Tran. Inf. Theory*, vol. 51, no. 7, pp. 2282–2312, 2005.

[199] X. Yi, X. Li, L. Xie, and K. H. Johansson, "Distributed online convex optimization with time-varying coupled inequality constraints," *arXiv preprint arXiv:1903.04277*, 2019.

[200] H. Yu and R. Jin, "On the computation and communication complexity of parallel SGD with dynamic batch sizes for stochastic non-convex optimization," in *International Conference on Machine Learning*, 2019, pp. 7174–7183.

[201] H. Yu, R. Jin, and S. Yang, "On the linear speedup analysis of communication efficient momentum SGD for distributed non-convex optimization," in *International Conference on Machine Learning*, 2019, pp. 7184–7193.

[202] H. Yu, M. Neely, and X. Wei, "Online convex optimization with stochastic constraints," in *Advances in Neural Information Processing Systems*, 2017, pp. 1428–1438.

[203] H. Yu, S. Yang, and S. Zhu, "Parallel restarted SGD with faster convergence and less communication: Demystifying why model averaging works for deep learning," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 33, 2019, pp. 5693–5700.

[204] J. Yuan and A. Lamperski, "Online convex optimization for cumulative constraints," in *Advances in Neural Information Processing Systems*, 2018, pp. 6137–6146.

[205] J. Zhang and L. Xiao, “Multilevel composite stochastic optimization via nested variance reduction,” *SIAM Journal on Optimization*, vol. 31, no. 2, pp. 1131–1157, 2021.

[206] X. Zhang, M. Hong, S. Dhople, W. Yin, and Y. Liu, “Fedpd: A federated learning framework with optimal rates and adaptivity to non-iid data,” *arXiv preprint arXiv:2005.11418*, 2020.

[207] Z. Zhang and G. Lan, “Optimal algorithms for convex nested stochastic composite optimization,” *arXiv preprint arXiv:2011.10076*, 2020.

[208] D. Zhou, P. Xu, and Q. Gu, “Stochastic nested variance reduced gradient descent for nonconvex optimization,” in *Adv. Neural Inf. Process. Systems*, 2018, pp. 3921–3932.

[209] M. Zinkevich, “Online convex programming and generalized infinitesimal gradient ascent,” in *Proceedings of the 20th International Conference on Machine Learning (ICML-03)*, 2003, pp. 928–936.

VITA

NAME OF AUTHOR: Pranay Sharma

PLACE OF BIRTH: Kanpur, Uttar Pradesh, India

DATE OF BIRTH: December 20, 1990

GRADUATE AND UNDERGRADUATE SCHOOLS ATTENDED:

Indian Institute of Technology, Kanpur, India

DEGREES AWARDED:

B.Tech-M.Tech (Dual degree), 2013,
Indian Institute of Technology, Kanpur, India