UNIVERSITY OF CALIFORNIA

Los Angeles

Towards Efficient Federated Learning: Overcoming Challenges in Communication, Heterogeneity, and Data Scarcity

A dissertation submitted in partial satisfaction
of the requirements for the degree
Doctor of Philosophy in Electrical and Computer Engineering

by

Navjot Singh

2024

ABSTRACT OF THE DISSERTATION

Towards Efficient Federated Learning: Overcoming Challenges in Communication, Heterogeneity, and Data Scarcity

by

Navjot Singh
Doctor of Philosophy in Electrical and Computer Engineering
University of California, Los Angeles, 2024
Professor Suhas Diggavi, Chair

The rapid growth of edge computing, 5G, and IoT technologies has led to a significant increase in distributed data, creating both opportunities and challenges for machine learning. Federated learning has emerged as a promising approach to enable collaborative model training across decentralized devices while not sharing user data. However, effectively implementing federated learning involves addressing several key challenges, including data scarcity, communication efficiency, and data heterogeneity.

This dissertation addresses these challenges through three key contributions. First, to enhance communication efficiency, we develop compressed stochastic gradient descent (SGD) algorithms that incorporate techniques such as event-triggered communication and local iterations. This approach reduces the frequency and size of data exchanges, minimizing communication overhead in decentralized training environments. We provide rigorous theoretical analysis to demonstrate the convergence rates and efficiency gains of these methods.

Second, to further address data heterogeneity, we consider communication efficient multi-task learning for decentralized topologies. These techniques allow for the simultaneous optimization of multiple related tasks, creating personalized models tailored to the unique data distributions and objectives of individual devices while also focusing on communication efficiency of exchanges. We formulate the multi-task learning problem in decentralized settings and provide bounds on the convergence of Gradient Descent and comment on performance improvements compared to traditional methods without compression.

Third, to tackle data scarcity, we leverage transfer learning methods with a focus on linear models. By utilizing pre-trained regression models from diverse source domains, we provide a robust starting point for target models and fine-tune them on limited local data. This method improves model performance and adaptability in data-scarce environments. We offer theoretical guarantees on the excess risk bounds for these transfer learning approaches, ensuring their reliability and effectiveness.

Overall, these contributions seek to enhance the robustness, scalability, and practicality of federated learning, enabling effective and collaborative learning in diverse and distributed environments. This work lays the groundwork for advanced federated learning applications, addressing critical challenges and providing a path forward for future research and development.

The dissertation of Navjot Singh is approved.

Lieven Vandenberghe

Quanqaun Gu

Choh-Jui Hsieh

Suhas Diggavi, Committee Chair

University of California, Los Angeles

2024

To my family . . .

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ACKNOWLEDGMENTS

The dissertation is the culmination of constant support, encouragement and guidance of a number of people I have met through the years in my graduate studies.

I first extend my deep gratitude and give a sincere thanks to my advisor, Prof. Suhas Diggavi for his valuable guidance throughout my graduate student researcher life. I have been greatly inspired by the breadth of his technical knowledge, his way of thinking about research from a deep, fundamental perspective, and on his multi-faceted approach to look at a problem. My time with him allowed me to become a better researcher and a more mature individual. His steady support and his trust in me throughout my time here is immensely appreciated.

This journey would not have been possible without the support of some wonderful collaborators with whom I had the privilege to learn from and work with. I greatly admire them all for intellectually nourishing me to become a more competent researcher. I wholeheartedly thank Deepesh, who has been a great mentor, as well as an elder brother to me through most of my PhD life. I thank Jemin George at the US Army Research Labs for being a collaborator in the early phases of my work. I have also had the singular pleasure of interacting with, and working with, Prof. Tamer Başar and Prof. Xuanyu Cao, and I greatly admire their sense of efficacy and attention to detail. I have also had the pleasure of interacting with Prof. Paulo Tabuada and his students, Jonathan and Matteo. These discussions have been intellectually simulating and have given me a much needed sense of looking at my research from an alternate lens. I would also like to thank all my collaborators at the places I have interned at - Amazon, Microsoft, Cubist. My discussions with them have been extremely helpful for me to tie my academic understanding with real world experience, and also on deciding what I would be most excited to work on in the future.

It takes a village to raise a child, and there is a long list of people who I've had the pleasure to meet at various stages in my life here, and who have made this dissertation possible. I thank Debraj, who I have known since my undergraduate days, for being a dear friend and helping me settle to life in Los Angeles and at UCLA. I am so grateful that our paths crossed here. A few other people started their PhD journeys in ARNI+LICOS labs at the same time as me - Dhaivat, Antonious, Osama- and I have had the great pleasure to develop such strong bonds with them. When I joined, ARNI+LICOS had a large number of graduating students, and I admire them all for their help and advice in many facets of the graduate student life. I thank Yahya, Gaurav and Sundar for being really helpful seniors in my early years at UCLA. I would like to thank dear friends who made LA feel more like home - Parthe, Akshay, Pratik, Sumit. I deeply cherish their valuable advice on things work and personal, and I will miss our countless long conversations about anything and everything. Fan has been a dear friend for most of my time here and my go to walking buddy. I thank you so much for lending an ear to all my rants and worries, and I am feel so happy both of us got to grow together as individuals. I never really knew I liked cats, and I am fortunate I got to meet Xiaonan through you, who has been an excellent work companion and a source of joy. Yashraj and Kush joined much later during my time here, and both have been vital in motivating me through many challenging times. I am immensely grateful and honored to have had their company. On the risk of sounding like a hack, I do lack concise words to put what each of you have done for me. But you have shaped me to become the person I am. For that and so much more, you have my heartfelt, eternal thanks.

The people at the LICOS and ARNI labs have been more like a family to me at UCLA, and have played a crucial role in the completion of this dissertation. I am grateful to the LICOS family, Deepesh, Antonious, Dhaivat, Rida, Kaan, who I got to spend most of my time here with, and also people who came in my later years, Ruida, Yağız, Ufuk, Bora, Bruce. I thank them for so many memories and conversations we have shared together and I feel so lucky to have grown together with them. People from ARNI - Osama, Merve, Chaorui, Mine - I thank you so much for all the conversations we have had. I am truly blessed to have shared my time here with the LICOS+ARNI family. Most of my happy moments at LA were spent getting coffee or walking with you. While mere words cannot fully express my gratitude for all that you have done for me, let this serve as a heartfelt acknowledgment for the records.

I would like to finally thank my family for all they have done for me and for their trust in me. So many small things had to align perfectly for me to achieve this, and I am eternally indebted for their unwavering support and love at each and every one of those moments.

VITA

2018	B.Tech + M.Tech (Electrical Engineering), Indian Institute of Technology, Bombay (IIT Bombay), India
2018	Joined Ph.D. program at ECE Department, UCLA
2020	Advanced to Ph.D. candidacy
2021	Applied Scientist Intern, Amazon, CA
2022	Machine Learning Scientist Intern, Microsoft, WA
2023	Quantitative Researcher Intern, Cubist Systematic Strategies LLC, NY

CHAPTER 1

Introduction

Large-scale distributed optimization has received significant attention recently due to the increasing demand for training models on massively distributed data. This need is particularly pronounced with the advent of edge-computation architectures and the rise of edge applications, where devices generate vast amounts of data, opening up new possibilities for machine learning. The data produced by these devices is often personal and sensitive, necessitating distributed training schemes that do not need explicit sharing of user data, unlike traditional centralized machine learning approaches.

In this distributed landscape, collaboration among devices is essential due to the fragmented nature of data and the inherent limitations of individual devices. Each device typically has access to a small, potentially biased subset of data, which is often insufficient for training robust and accurate models. By collaborating, devices can collectively leverage their distributed data, significantly enhancing the diversity and volume of the available training dataset. This collaborative approach not only improves the generalization performance of the models but also ensures that the learning process respects privacy constraints, as raw data remains local to each device.

Federated learning aims to meet the pressing need for collaborative model training across multiple devices while keeping data localized. This approach is crucial for not sharing user level data and efficiently utilizing the vast amounts of data generated by edge devices. However, implementing federated learning introduces several significant challenges that must be addressed to fully realize its potential.

A primary motivation for federated learning is the issue of data scarcity. Many devices have access to only limited local data, which severely impedes the training of robust and accurate machine learning models. With insufficient data, models are prone to overfitting, performing well on the training data but failing to generalize to new, unseen data. This overfitting leads to poor overall model performance, highlighting the necessity for a collaborative approach where devices share insights derived from their local data without actually sharing the data itself.

Another critical challenge in federated learning is communication efficiency. Synchronizing model updates between numerous devices and a central server can result in substantial communication overhead. This is particularly problematic in bandwidth-constrained environments, where frequent data exchanges can be slow and costly. Efficient communication is essential to minimize the data exchanged during each synchronization round, ensuring that the federated learning process remains scalable and efficient.

Data heterogeneity poses another significant challenge. Data available on different devices can vary widely due to differences in user behavior, local environments, and specific applications. This variability means that a single global model may not perform optimally across all devices. The diversity in data distributions complicates the training process, as it requires the model to generalize well across all diverse datasets. Techniques that can accommodate these differences and personalize the learning process for individual devices are essential for effective federated learning.

Addressing these challenges—data scarcity, communication efficiency, and data heterogeneity—is crucial for unlocking the full potential of federated learning. By developing strategies to handle these issues, we can enhance the capability of federated learning systems to provide robust, accurate, and efficient machine learning solutions that respect user privacy and leverage the collective power of distributed data.

1.1 Contributions of this dissertation and outline

The contributions of this dissertation are designed to address the challenges outlined above, with the goal of realizing efficient and effective machine learning in federated learning environments. In the following, we provide point-by-point description of the specific contributions we have made to overcome them. By systematically targeting these issues, we aim to enhance the performance, scalability, and practicality of federated learning systems.

- **Algorithms for Communication Efficient Decentralized Training**: We investigate schemes for communication-efficient decentralized training of large-scale models on a graph. Recently, two primary techniques for achieving communication efficiency in federated learning have garnered attention: (i) reducing the total number of communication rounds between clients and (ii) reducing the size of transmitted messages or information exchange during each round. We examine the impact of incorporating these techniques into decentralized learning. Specifically, we develop an algorithm based on compressed SGD updates for decentralized optimization, which integrates Nesterov's momentum and minimizes communication through compression and local iterations with event-triggered communication. Momentum-based methods are known for their faster convergence and superior generalization, making them widely adopted for training large-scale machine learning models. However, the theoretical understanding of the role of momentum in the convergence rates of compressed SGD remains an open question. We address this by analyzing the convergence rates for decentralized SGD training with momentum. This forms a core part of Chapter 3, which explains how communication-efficient techniques can be applied to stochastic decentralized optimization of large-scale models, providing both practical algorithms and theoretical guarantees of convergence rates. This work has lead to publications in the conferences IEEE CDC 2020 [SDGD20b], IEEE ISIT 2021 [SDGD21b], the journals IEEE TAC [SDGD22], IEEE JSAIT [SDGD21c], and a Springer chapter [SDD23].

- **Multi-task learning (Personalization) in Decentralized optimization**: While achieving communication efficiency is crucial, addressing the heterogeneity of nodes through personalized models is equally important. Multi-task learning, which involves simultaneously optimizing multiple related tasks, allows for tailored solutions to the distinct problems faced by each device, leveraging their unique data and objectives. This approach enhances model relevance and performance on individual nodes by optimizing for local data rather than relying on a generic global model, thus improving the overall robustness of the system. In Chapter 4, we explore multi-task learning within the context of communication-efficient decentralized optimization, focusing on a multi-agent network where each node has a stochastic local cost function and additional pairwise constraints on the decision variables of neighboring nodes. By minimizing the aggregate objective function, which is the sum of the expected values of the local cost functions, we enhance collaboration, effectively addressing data heterogeneity and improving the overall quality of models in distributed learning systems. This chapter demonstrates how multi-task learning can be integrated into decentralized optimization, yielding personalized models that cater to the specific needs of individual nodes while maintaining the benefits of collaborative training. This work has appeared in the journal Automatica [SCDB24].

- **Representation transfer learning**:
  Participating nodes often face the challenge of limited local data, which impedes the training of accurate models from scratch, leading to overfitting and poor generalization. Transfer learning addresses this issue by leveraging knowledge from pre-trained models on related tasks to enhance learning on target tasks with limited data. By initializing the target model with weights from a pre-trained model and fine-tuning it on the limited data, transfer learning provides a robust starting point, improving model performance and adaptability. In contemporary machine learning, effectively utilizing limited data to build accurate models is crucial, particularly in domains where data acquisition is expensive or time-consuming. Linear regression, a fundamental technique in statistics and machine learning, often faces challenges in such scenarios due to the need for sufficient data to ensure robust model training. Despite its simplicity, linear regression serves as a vital stepping stone toward understanding and implementing more complex models, such as neural networks. In Chapter 5, we explore how transfer learning mitigates data scarcity for linear models. Our approach utilizes pre-trained regression models to transfer from diverse source domains and a single target domain, assuming a structured representation for the data-generating models across both source and target domains. We propose a representation transfer-based learning method tailored to the target model, providing excess risk

bounds for transfer learning and fine-tuning. This method not only addresses immediate challenges but also lays the groundwork for advancements in more sophisticated machine learning models. This work has appeared in the conference IEEE ISIT 2023 [SD23], and is under review for the journal IEEE JSTSP.

We begin by presenting background information and preliminaries in Chapter 2. This foundational material sets the stage for the subsequent chapters, which delve into the technical subject matter of this dissertation in greater detail.

CHAPTER 2

Background and Preliminaries

This chapter introduces key concepts that will be foundational for the subsequent chapters and establishes the notation we will use.

We begin by defining the nature of the optimization problems central to our investigation, emphasizing the constraints inherent to federated learning environments. Key constraints include the limited bandwidth of communication links connecting individual devices and the heterogeneity of data distribution across nodes. A significant motivator for employing collaborative training in a federated setup is the scarcity of data at participating devices. This data insufficiency thus often discourages training a model from scratch that can achieve good generalization performance locally. Consequently, we explore transfer learning methods tailored to such scenarios, where leveraging knowledge from pre-trained models can enhance learning efficacy and model performance.

2.1 Formulating the optimization problem for collaborative learning

In many practical problems of interest, the objective is to minimize a global function that represents the sum of several local objective functions, each associated with different data sources or nodes. Let $\mathbf{x} \in \mathbb{R}^d$ denote the model parameters to be optimized. The overall objective is formulated as:

$$F(\mathbf{x}) = \sum_{i=1}^{n} F_i(\mathbf{x}) \tag{2.1}$$

where n is the total number of nodes and $F_i(\mathbf{x})$ represents the local objective function for the i-th node and is given by:

$$F_i(\mathbf{x}) = \mathbb{E}_{\mathbf{z}\sim P_i}[\ell(\mathbf{z}; \mathbf{x})]$$

Here, P_i denotes the joint data distribution over the feature, target pair (denoted by $\mathbf{z}$) in the context of supervised learning at node i which could vary significantly among the different nodes. We cover the implications of this phenomena below in Section 2.2. We call the objective in (2.1) as the *Population Risk*. Each node k maintains its own local dataset $\mathcal{D}_i$, and the local empirical risk is given by:

$$\hat{F}_i(\mathbf{x}) = \frac{1}{N_i} \sum_{j=1}^{N_i} \ell(\mathbf{z}^{(j)}; \mathbf{x})$$

where N_i is the number of data points at node i, $\mathbf{z}^{(j)}$ denotes the joint features and labels pair of the j-th data point at node i, ℓ is the loss function, and $\mathbf{x}$ denotes the model parameters.

An example of such an objective in federated learning, might that be of multiple hospitals with each hospital $i \in [n]$ having its own dataset $\mathcal{D}_i$ of patient records, with N_i records. The local objective function $\hat{F}_i(\mathbf{x})$ could represent the objective function of a machine learning model, for e.g. medical vision prediction. The global objective $\hat{F}(\mathbf{x})$ is the sum of these local prediction errors across all hospitals, aiming to train a model that minimizes the total error, which we also call the *Empirical Risk*:

$$\hat{F}(\mathbf{x}) = \sum_{i=1}^{n} \frac{1}{N_i} \sum_{j=1}^{N_i} \ell(\mathbf{z}^{(j)}, \mathbf{x}) \tag{2.2}$$

Gradient based schemes, like Stochastic Gradient Decent (SGD), are widely used to solve such non-convex optimization problem. The solution to the problem above finds the model parameters $\mathbf{x}_{emp}$ that minimize this global objective, leveraging the distributed data while preserving data locality and privacy. This aggregate optimization approach facilitates collaboration among the nodes by enabling them to contribute to the global model without sharing their underlying data. We remark that the above objective uses the *empirical* data available at a node, which acts as a proxy to optimizing the overall population risk in (2.1). In particular, we would be interested in obtaining a model from a hypothesis class $\mathcal{X}$, for e.g., the set of possible neural network parameters for a given architecture. It might also be of interest to compare the obtained model $\mathbf{x}_{emp}$ with the best model possible from $\mathcal{X}$ that minimizes (2.1). This forms the notion of *Expected Excess Risk*, which characterizes the performance of the obtained model to the 'best' model that minimizes the population risk:

$$\text{EER}(\mathbf{x}_{emp}) := F(\mathbf{x}_{emp}) - \min_{\mathbf{x} \in \mathcal{X}} F(\mathbf{x}) \tag{2.3}$$

The value of the difference in (2.3) is closely tied to the notion of *generalization* and can be used as a measure for characterizing the *test* performance of a given model parameter. Note that the underlying definition for the Excess risk as provided does not necessarily require us to consider multiple collaborating devices and can be used even for a *centralized* machine learning scenario with a single processing node.

2.2 Heterogeneity across participating nodes

In a federated learning scenario, each node typically possesses a unique data distribution, reflecting diverse user behaviors, local environments, and data collection practices. This non-i.i.d. nature of data across nodes complicates the overall optimization procedure described in (2.2), as the global model must generalize well across varied distributions P_i for $i \in [n]$. Hence, learning a single global model, with the stated differences in local data distributions, can lead to model updates that are not uniformly beneficial for all participants. This underscores the need for *personalization* in federated learning. Personalized models, tailored to the specific data characteristics of each node, can address the limitations of a one-size-fits-all approach. As such, the overall optimization problemn can instead be viewed from a lens of *Multiple Task Learning* where different nodes seek to optimize for different tasks, but solving these problems in a collaborative manner allows for effective information exchange among them. Hence, multi task training can enhance model performance and relevance for individual nodes, ensuring that the learning process accommodates the diverse data environments inherent in federated systems. A generic optimization problem over the given n nodes while taking multiple task training into account can be stated as follows:

$$\min_{\mathbf{x}_1,\ldots,\mathbf{x}_N \in \mathcal{X}} \sum_{i=1}^{n} F_i(\mathbf{x}_i) + \Omega(\mathbf{x}_1, \ldots, \mathbf{x}_n) \tag{2.4}$$

Where the function $\Omega : \mathbb{R}^{d\times\ldots\times d} \to \mathbb{R}$ forms the required constraint on the multiple tasks. As an example, the objective in (2.1) can be recovered by defining Ω such that $\mathbf{x}_1 = \mathbf{x}_2 = \ldots = \mathbf{x}_n$ are the only permissible values, thus imposing a single common model for all nodes. Similarly, setting Ω to a constant value regardless of the input model parameters allows for a decomposition of the overall objection to the individual nodes, leading to no collaboration. Between these two extremes, a non-trivial value of Ω could lead to different models among the participating nodes, while also allowing them to collaborate via update exchanges. We now focus on the nature of these exchanges by first discussing the underlying connectivity structure of the nodes and some know theoretical results of gradient based schemes in them.

2.3 Network topology and convergence

In the context of federated learning, the choice between decentralized and distributed network topologies significantly impacts the efficiency and effectiveness of the training process. We describe each of these briefly below and note the convergence rate of SGD in these scenarios.

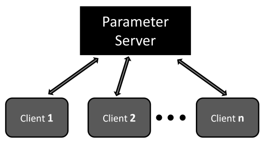


Figure 2.1 Distributed topology with a Parameter server

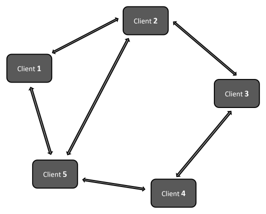


Figure 2.2 Decentralized topology

- **Distributed Topology:** This typically involves a central server (or, *Parameter Server*) that coordinates and aggregates updates from multiple devices. While this approach simplifies the synchronization and coordination processes, it introduces several critical drawbacks. The central server becomes a bottleneck, potentially limiting scalability and increasing vulnerability to failures and attacks. Furthermore, the reliance on central aggregation can lead to higher communication overhead and increased latency, especially as the number of participating devices grows. The convergence rate of SGD in this setup are listed below, assuming we run the procedure for T number of iteration, and for N number of nodes in the network:

  - *Strongly Convex Objectives*[1]: $\mathcal{O}\left(\frac{1}{nT}\right)$
  - *General (Non-convex) Objectives*: $\mathcal{O}\left(\frac{1}{\sqrt{nT}}\right)$

- **Decentralized Topology:** In this framework, each device or node communicates directly with other nodes in a peer-to-peer manner, enhancing robustness and scalability by reducing reliance on a central server. This approach mitigates single points of failure and distributes the computational load more evenly across the network. This can be seen as a generalization of the *distributed* scenario in the case where all the nodes are allowed to communicate with each other. This added generality typically also necessitates sophisticated coordination mechanism to ensure convergence across the network as convergence results would typically depend on the connectivity of the underlying graph structure (c.f. [SDGD20c, LZZL17, LZZ$^+$17]).

[1]A function $f(\mathbf{x})$ is called strongly convex with parameter $\mu > 0$ if for all $\mathbf{x}_1, \mathbf{x}_2 \in \mathbb{R}^d$, the following inequality holds:

$$f(\mathbf{x}_2) \geq f(\mathbf{x}_1) + \nabla f(\mathbf{x}_1)^\intercal(\mathbf{x}_2 - \mathbf{x}_1) + \frac{\mu}{2}\|\mathbf{x}_2 - \mathbf{x}_1\|^2$$

– *Strongly Convex Objectives*: $\mathcal{O}\left(\frac{1}{nT}\right)$

– *General (Non-convex) Objectives*: $\mathcal{O}\left(\frac{1}{\sqrt{nT}}\right)$

2.4 Methods for introducing communication efficiency

The bandlimited nature of communication links between nodes participating in federating learning is a key factor to be kept in mind while designing practical algorithms. Here we discuss ideas which serve as useful methods for alleviating communication bottlenecks in either distributed or decentralized settings.

1. **Compression schemes**

   In the traditional multi-node training setting, using the SGD algorithm for each worker allows exchange of full precision updates between communicating nodes (clients). These updates are usually comprised of floating vectors with each entry represented by 32-bit or 64-bit precision. Thus one can think of compressing the exchanged updates which can lead to savings in communication, and thus facilitate faster training of models over bandlimited communication links. For the purpose of this prospectus, we use the following definition for the compression operator applied on a vector.

   Definition 1 (Compression operator [SCJ18b])**.** *A (possibly randomized) function $\mathcal{C} : \mathbb{R}^d \to \mathbb{R}^d$ is called a compression operator, if there exists an $\omega \in (0, 1]$, such that for every $\mathbf{x} \in \mathbb{R}^d$, we have*

$$\mathbb{E}_{\mathcal{C}} \left\| \mathcal{C}(\mathbf{x}) - \mathbf{x} \right\|^2 \leq (1 - \omega) \left\| \mathbf{x} \right\|^2, \tag{2.5}$$

   where expectation is taken over the randomness of $\mathcal{C}$.

Some important sparsifiers and quantizers following the above definition are:

- Top_k and $Rand_k$ sparsifiers (where only k entries are selected and the rest are set to zero) with $\omega = k/d$ [SCJ18b],
- Stochastic quantizer Q_s from [AGL$^+$17c][2] with $\omega = (1 - \beta_{d,s})$ for $\beta_{d,s} < 1$.
- Deterministic quantizer $\frac{\|\mathbf{x}\|_1}{d} Sign(\mathbf{x})$ from [KRSJ19b] with $\omega = \frac{\|\mathbf{x}\|_1^2}{d\|\mathbf{x}\|_2^2}$.
- For $Comp_k \in \{Top_k, Rand_k\}$, the following are compression operators [BDKD19b]:[3]

  (a) $\frac{1}{(1+\beta_{k,s})} Q_s(Comp_k)$ with $\omega = \left(1 - \frac{k}{d(1+\beta_{k,s})}\right)$ for any $\beta_{k,s} \geq 0$, and

  (b) $\frac{\|Comp_k(\mathbf{x})\|_1 Sign Comp_k(\mathbf{x})}{k}$ with $\omega = \max\left\{\frac{1}{d}, \frac{k}{d}\left(\frac{\|Comp_k(\mathbf{x})\|_1^2}{d\|Comp_k(\mathbf{x})\|_2^2}\right)\right\}$.

2. **Local updates**

An alternate idea to introduce communication efficiency is to limit the number of times information is exchanged between participating nodes. This class of techniques along with stochastic gradient descent are termed as local SGD [Sti19, BDKD19b]. Local SGD techniques allow for skipping communication rounds (aggregation steps between workers) in favor of performing gradient computations, thus trading communication for more computation to explore the parameter space. Depending on the relative frequency of updates for each worker, they are usually classified as 'synchronous' [KRSJ19b, BDKD19b], where each client takes the same number of local gradient update steps before

[2] $Q_s : \mathbb{R}^d \to \mathbb{R}^d$ is a stochastic quantizer, if for every $\mathbf{x} \in \mathbb{R}^d$, we have (i) $\mathbb{E}[Q_s(\mathbf{x})] = \mathbf{x}$ and (ii) $\mathbb{E}[\|\mathbf{x} - Q_s(\mathbf{x})\|_2^2] \leq \beta_{d,s}\|\mathbf{x}\|_2^2$. Q_s from [AGL$^+$17c] satisfies this definition with $\beta_{d,s} = \min\left\{\frac{d}{s^2}, \frac{\sqrt{d}}{s}\right\}$.

[3] [BDKD19b] show that the composition of sparsification and quantization operators is also a valid compression operator, outperforming its individual components in terms of communication savings while still maintaining similar performance.

communication, or 'asynchronous' [RRWN11, BDKD19b], where the nodes are allowed to communicate at different rates by taking different number of update steps. In this work, as we consider each node to take a fixed number of local-SGD steps. In particular, we assume that each node takes H local steps before it tries to communicate to its neighboring nodes.

3. **Threshold based triggering**
   We introduce an additional layer of efficiency by allowing nodes to refuse participating in communication if there is no significant change in the local model parameters since the last time communication occurred. Specifically, we set a 'triggering' threshold condition on the ℓ_2 norm difference between model parameters of the node, and communicate only if the difference exceeds a certain triggering threshold. This scheme can be potentially applied on top of compression and local update steps as discussed before, leading to improved communication efficiency.

For the content of this thesis, we define 'event-triggering' as the combination of local update steps and threshold based triggering rule.

2.5 Representation Transfer

In many practical scenarios, participating nodes might suffer from limited local data, which impedes the training of accurate models. Working with limited data (*data scarcity*) is particularly challenging for devices in resource-constrained environments as insufficient local data often leads to overfitting and poor generalization, as models trained on small datasets fail to capture the broader data distribution effectively. To mitigate this issue, transfer learning can be employed, leveraging knowledge from pre-trained models on related tasks to enhance the learning process on target tasks with limited data.

Transfer learning typically leverages a pre-trained model from a source domain, where ample data is available, and transfers its learned representations to a target domain with limited data. In the context of neural networks, these representations typically involve features learned in the earlier layers of the network. As an example, a Convolutional Neural Network pre-trained on a large dataset like ImageNet learns to identify basic visual elements that can be highly useful for other image recognition tasks, even in vastly different domains like medical imaging.

The process involves initializing the target model with the weights of the pre-trained source model. These weights encapsulate rich and generalizable features, providing a robust starting point for the target task. The target model is then fine-tuned on the limited target data, adjusting the pre-trained representations to better fit the specific characteristics of the target domain. This fine-tuning process helps the model to quickly adapt and improve its performance, leveraging the extensive knowledge encoded in the pre-trained model.

2.6 Tying the various themes

In conclusion, addressing data scarcity in federated learning involves the integration of communication-efficient decentralized training, multi-task learning, and representation transfer. Communication-efficient decentralized training minimizes the overhead of synchronizing model updates, ensuring that collaborative learning remains feasible even with limited bandwidth. Multi-task learning leverages the diversity of tasks across nodes, allowing each device to benefit from the collective knowledge of the network while tailoring the learning process to its specific objectives. Representation transfer enhances learning outcomes by utilizing rich feature sets from pre-trained models on larger, diverse datasets, providing a robust starting point for nodes with limited local data. The common theme across these methodologies is the importance of collaboration in federated learning. By leveraging shared knowledge, devices can transcend the limitations imposed by individual data scarcity, leading to more robust and accurate models. In the subsequent chapters, we will delve deeper into each of these strategies separately, with communication efficient training being central to Chapter 3 and Chapter 4, multi-task learning being central to Chapter 4, and representation transfer learning in the context of linear models for Chapter 5.

CHAPTER 3

Communication efficient decentralized training

The rapid growth of data across diverse and distributed sources has driven the need for advanced machine learning paradigms that can effectively utilize this decentralized data. Federated learning, an innovative approach, enables the training of machine learning models on multiple decentralized devices or servers holding local data samples without requiring data exchange. This approach significantly enhances privacy and security by keeping data localized, thereby reducing the risks associated with centralized data storage and transmission. However, a major challenge in federated learning is ensuring communication efficiency, as the communication overhead can become a bottleneck, particularly in large-scale and resource-constrained environments.

This chapter focuses on developing and analyzing communication-efficient algorithms for federated learning, with a specific emphasis on Stochastic Gradient Descent (SGD) based methods. Efficient communication strategies are essential for minimizing the amount of data exchanged during training, thus improving the overall efficiency and scalability of federated learning systems.

We introduce SQuARM-SGD, an algorithm designed to minimize communication overhead while maintaining robust convergence properties. SQuARM-SGD stands for Sparsified and Quantized Action Regulated Momentum Stochastic Gradient Descent. The algorithm employs several advanced techniques to achieve communication efficiency, including local iterations, where computations are performed locally at each worker node for several iterations before communicating updates, and compression techniques, such as sparsification and quantization, to reduce the size of the communicated data. Additionally, SQuARM-SGD incorporates Nesterov's momentum to accelerate the convergence of gradient-based optimization algorithms.

We provide rigorous theoretical convergence guarantees for SQuARM-SGD in both strongly-convex and non-convex smooth objective settings. Our analysis shows that SQuARM-SGD achieves a convergence rate of $\mathcal{O}\left(\frac{1}{nT}\right)$ for strongly-convex objectives and $\mathcal{O}\left(\frac{1}{\sqrt{nT}}\right)$ for non-convex objectives, where n is the number of worker nodes and T is the number of iterations. These rates match those of vanilla distributed SGD, demonstrating that our communication-efficient approach does not compromise performance.

This work also represents the first theoretical analysis of convergence for compressed gradient updates with momentum in a decentralized setting. SQuARM-SGD's ability to maintain high performance while significantly reducing communication costs makes it particularly suitable for large-scale federated learning applications. Furthermore, it also provides the first convergence analysis for compressed decentralized training with momentum using a weaker set of assumptions than existing literature while incorporating the local SGD and event triggered communication framework of [SDGD19].

In addition to the theoretical analysis, we present experimental comparisons with the current state-of-the-art in decentralized training, highlighting the practical effectiveness of our approach. The results underscore the advantages of SQuARM-SGD in real-world federated learning scenarios, emphasizing its potential to advance the field by addressing one of its most pressing challenges.

By focusing on communication-efficient algorithms like SQuARM-SGD, this chapter contributes to the broader goal of making federated learning more scalable and practical, enabling its deployment in a wider range of applications where data privacy and security are crucial.

3.1 Related work

Communication-efficient decentralized training has received recent attention; see [TGZ$^+$18, SDGD19, RMHP18, ALBR19, TT17, KLSJ20, YJY19] and references therein. The current state-of-the-art in communication efficient decentralized training is CHOCO-SGD [KLSJ20, KSJ19b], which considers sparsification or quantization of the model parameters, without incorporating momentum in their theoretical analyses[1]. Our convergence analyses are very different and significantly more involved than that of CHOCO-SGD, as apart from studying local iterations and event-triggered communication in decentralized SGD, unlike [KLSJ20, KSJ19b], we provide our analyses using virtual sequences, specifically, to handle the use of momentum. The use of local iterations with momentum updates for decentralized setting is studied in [YJY19, WTBR20], but without any compression of exchanged information. [ZHK19] studied momentum SGD with compressed updates (but no local iterations or event-triggering) for the *distributed* setting only, assuming that all workers have access to unbiased gradients. Extending the analysis to *decentralized* setting (where different workers may have local data, potentially generated from

[1]They do report numerics with momentum, and we compare the numerical performance SQuARM-SGD with it in Section 3.5.

different distributions) while incorporating compression, local iterations and event triggered communication in SQuARM-SGD poses several challenges; see Section 3.4 for a detailed discussion. The idea of event-triggering has been explored in the control community [HJT12, DFJ12, SDJ13, Gir15, LNTL17] and in optimization literature [KCM15, CR16, DYG+18]. These papers focus on continuous-time, deterministic optimization algorithms for convex problems; in contrast, we propose event-driven stochastic gradient descent algorithms for both convex and non-convex problems. [CGSY18] propose an adaptive scheme to skip gradient computations in a *distributed* setting for *deterministic* gradients; moreover, their focus is on saving communication rounds, without compressed communication. In summary, to the best of our knowledge, our work is the first to develop and analyze convergence of momentum-based decentralized stochastic optimization, using compressed lazy communication (as described earlier). Moreover, the numerics demonstrate a significant advantage over the state-of-the-art in communication efficiency in terms of number of bits for a given learning performance.

3.2 Problem Setup and Proposed Algorithm

We first formalize the decentralized optimization setting that we work with and set up the notation we follow throughout the chapter. Consider an undirected connected graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ with $\mathcal{V} = [n] := \{1, 2, \ldots, n\}$, where node $i \in [n]$ corresponds to worker i and we denote the neighbors of node i by $\mathcal{N}_i := \{(i,j) : (i,j) \in \mathcal{E}\}$. To each node $i \in [n]$, we associate a dataset $\mathcal{D}_i$ and an objective function $f_i : \mathbb{R}^d \to \mathbb{R}$. We allow the datasets and objective functions to be different for each node and assume that for $i \in [n]$, objective function f_i has the form $f_i(\mathbf{x}) = \mathbb{E}_{\xi_i \sim \mathcal{D}_i}[F_i(\mathbf{x}, \xi_i)]$

where $\xi_i \sim \mathcal{D}_i$ denotes a random sample from $\mathcal{D}_i$, $\mathbf{x}$ denotes the parameter vector, and $F_i(\mathbf{x}, \xi_i)$ denotes the risk associated with sample ξ_i with respect to (w.r.t.) the parameter vector $\mathbf{x}$. Consider the following empirical risk minimization problem, where $f : \mathbb{R}^d \to \mathbb{R}$ is called the global objective function:

$$\arg\min_{\mathbf{x}\in\mathbb{R}^d} \Big(f(\mathbf{x}) := \frac{1}{n} \sum_{i=1}^{n} f_i(\mathbf{x}) \Big), \tag{3.1}$$

The nodes in $\mathcal{G}$ wish to minimize (3.1) collaboratively in a communication-efficient manner.

In order to describe and analyze our algorithm, we need some notations first. Let $W \in \mathbb{R}^{n\times n}$ denote the connectivity matrix of $\mathcal{G}$, where for every $(i, j) \in \mathcal{E}$, the w_{ij}'th entry of W denotes the weight on the edge (i, j) – e.g., w_{ij} may represent the strength of the connection on the edge (i, j) – and for other pairs $(i, j) \notin \mathcal{E}$, the weight w_{ij} is zero. We assume that W is symmetric and doubly stochastic, which means it has non-zero entries with each row and column summing up to 1. Consider the ordered eigenvalues of W, $|\lambda_1(W)| \geq |\lambda_2(W)| \geq \ldots \geq |\lambda_n(W)|$. For such a W associated with a connected graph $\mathcal{G}$, it is known that $\lambda_1(W) = 1$ and $\lambda_i(W) \in (-1, 1)$ for all $i \in \{2, \ldots, n\}$. The spectral gap $\delta \in (0, 1]$ is defined as $\delta := 1 - \lambda_2(W)$. Simple matrices W having $\delta \in (0, 1]$ are known to exist for connected graphs [KSJ19b].

3.2.1 Our proposed algorithm: SQuARM-SGD

We propose SQuARM-SGD to minimize (3.1), which is a decentralized algorithm that combines compression and Nesterov's momentum, together with event-driven communication exchange, where compression is achieved by sparsifying *and* quantizing the exchanges. Each worker is required to complete a fixed number of *local SGD* steps with *momentum*, and communicate *compressed* updates to its neighbors when there is a *significant change* in its local parameters since the last time communication occurred. SQuARM-SGD can be seen as a significant extension of CHOCO-SGD [KLSJ20, KSJ19b], which only performs compression using either sparsification or quantization.

To realize exchange of compressed parameters between workers, for each node $i \in [n]$, all nodes $j \in \mathcal{N}_i$ maintain an estimate $\hat{\mathbf{x}}_i$ of $\mathbf{x}_i$. Each node $i \in [n]$ has access to $\hat{\mathbf{x}}_j$ for all $j \in \mathcal{N}_i$. Our algorithm runs for T iterations and the set of synchronization indices is defined as $\mathcal{I}_T = \{I_{(1)}, \ldots, I_{(k)}, \ldots\} \subseteq [T]$, which are same for all workers and denote the time steps at which workers are allowed to communicate, provided they satisfy a triggering condition. For $\mathcal{I}_T$, we define its gap as $gap(\mathcal{I}_T) := \max_m \{I_{(m)} - I_{(m-1)}\}$. We assume that $gap(\mathcal{I}_T) = H$. SQuARM-SGD is described in Algorithm 1.

For a given connected graph $\mathcal{G}$ with connectivity matrix W, we first initialize a consensus step-size γ (see Theorem 2 for definition), momentum factor β, the sequence of learning rates $\{\eta_t\}_{t=0}^{T}$, triggering threshold sequence $\{c_t\}_{t=0}^{T}$, and momentum vector $\mathbf{v}_i$ for each node i initialized to $\mathbf{0}$. We initialize the copies of all the nodes $\hat{\mathbf{x}}_i = \mathbf{0}$ and allow each node to communicate in the first synchronization round. At each time step t, each worker $i \in [n]$ samples a stochastic gradient $\nabla F(\mathbf{x}_i^{(t)}, \xi_i)$ and takes a local SGD step on parameter $\mathbf{x}_i^{(t)}$ using Nesterov's momentum to form an intermediate parameter $\mathbf{x}_i^{(t+1/2)}$ (line 3-5). If the next iteration corresponds to a synchronization index, i.e, $(t+1) \in \mathcal{I}_T$, then each worker checks the triggering condition (line 8). If satisfied, each worker communicates compressed change in its copy to all its neighbors $\mathcal{N}_i$ (line 9-10); otherwise, it does not communicate in that round (line 12). After receiving the compressed updates of copies from all neighbors, the node i updates the

Algorithm 1 SQuARM-SGD: Sparsified and Quantized Action Regulated Momentum SGD

```
Parameters: $G = ([n], E)$, $W$
1: Initialize: For every $i \in [n]$, set arbitrary $\mathbf{x}_i^{(0)} \in \mathbb{R}^d$, $\hat{\mathbf{x}}_i^{(0)} := \mathbf{0}$, $\mathbf{v}_i^{(-1)} := \mathbf{0}$. Fix the
   momentum coefficient $\beta$, consensus step-size $\gamma$, learning rates $\{\eta_t\}_{t=0}^T$, triggering
   thresholds $\{c_t\}_{t=0}^T$, and synchronization set $\mathcal{I}_T$.
2: for $t = 0$ to $T - 1$ in parallel for all workers $i \in [n]$ do
3:     Sample $\xi_i^{(t)}$, stochastic gradient $\mathbf{g}_i^{(t)} := \nabla F_i(\mathbf{x}_i^{(t)}, \xi_i^{(t)})$
4:     $\mathbf{v}_i^{(t)} = \beta \frac{\eta_{t-1}}{\eta_t} \mathbf{v}_i^{(t-1)} + \mathbf{g}_i^{(t)}$
5:     $\mathbf{x}_i^{(t+\frac{1}{2})} := \mathbf{x}_i^{(t)} - \eta_t(\beta \mathbf{v}_i^{(t)} + \mathbf{g}_i^{(t)})$
6:     if $(t + 1) \in I_T$ then
7:         for neighbors $j \in \mathcal{N}_i \cup i$ do
8:             if $\|\mathbf{x}_i^{(t+\frac{1}{2})} - \hat{\mathbf{x}}_i^{(t)}\|_2^2 > c_t \eta_t^2$ then
9:                 Compute $\mathbf{q}_i^{(t)} := \mathcal{C}(\mathbf{x}_i^{(t+\frac{1}{2})} - \hat{\mathbf{x}}_i^{(t)})$
10:                Send $\mathbf{q}_i^{(t)}$ and receive $\mathbf{q}_j^{(t)}$
11:            else
12:                Send $\mathbf{0}$ and receive $\mathbf{q}_j^{(t)}$
13:            end if
14:            $\hat{\mathbf{x}}_j^{(t+1)} := \mathbf{q}_j^{(t)} + \hat{\mathbf{x}}_j^{(t)}$
15:        end for
16:        $\mathbf{x}_i^{(t+1)} = \mathbf{x}_i^{(t+\frac{1}{2})} + \gamma \sum_{j \in \mathcal{N}_i} w_{ij}(\hat{\mathbf{x}}_j^{(t+1)} - \hat{\mathbf{x}}_i^{(t+1)})$
17:    else
18:        $\hat{\mathbf{x}}_i^{(t+1)} = \hat{\mathbf{x}}_i^{(t)}$ , $\mathbf{x}_i^{(t+1)} = \mathbf{x}_i^{(t+\frac{1}{2})}$ for all $i \in [n]$
19:    end if
20: end for
```

locally available copies and its own copy (line 14). With these updated copies, the worker nodes finally take a consensus (line 16) with appropriate weighting decided by entries of W. In the case when $(t+1) \notin \mathcal{I}_T$, the nodes maintain their copies and move on to next iteration (line 18); thus no communication takes place.

Memory-efficient version of Algorithm 1: At a first glance, it may seem that in Algorithm 1, every node has to store estimates of all its neighbors' parameters in order to perform the consensus step, which may be impractical in large-scale learning. However, note that in the consensus step (line 16), nodes only require the weighted sum of their neighbors' parameters. So, it suffices for each node to store only the weighted sum of all its neighbors' parameters (in addition to its own local parameters and its estimate), and thus completely avoid the need to store all its neighbors parameters. For completeness, we provide a memory-efficient version of SQuARM in Appendix A.

Equivalence to error-feedback mechanisms: In Algorithm 1, though nodes do not explicitly perform local error-compensation (as in [KRSJ19b, BDKD19b]), the error-compensation happens implicitly. To see this, note that nodes maintain copies of their neighbors' parameters and update them as $\hat{\mathbf{x}}_j^{(t+1)} = \hat{\mathbf{x}}_j^{(t)} + \mathcal{C}(\mathbf{x}_j^{(t+\frac{1}{2})} - \hat{\mathbf{x}}_j^{(t)})$ (line 14) and then take the consensus step (line 16). Thus, the error gets accumulated into $\hat{\mathbf{x}}_j^{(t)}$ and is compensated by the term $\mathcal{C}(\mathbf{x}_j^{(t+\frac{1}{2})} - \hat{\mathbf{x}}_j^{(t)})$ in the next round.

3.3 Main Results

In this section we provide the convergence results for SQuARM-SGD (Algorithm 1) under two sets of assumptions: We present our results with the weakest set of assumptions in existing literature in Section 3.3.1 and slightly more general results with stronger assumptions in Section 3.3.2.

3.3.1 Theoretical Results with Relaxed Assumptions

Assumption 1 (Smoothness)**.** *We assume that each local function f_i for $i \in [n]$ is L-smooth, i.e., $\forall \mathbf{x}, \mathbf{y} \in \mathbb{R}^d$, we have $f_i(\mathbf{y}) \leq f_i(\mathbf{x}) + \langle \nabla f_i(\mathbf{x}), \mathbf{y} - \mathbf{x} \rangle + \frac{L}{2}\|\mathbf{y} - \mathbf{x}\|^2$.*

Assumption 2. *We assume that there exists finite constants $\sigma, M \geq 0$, such that for all $\mathbf{x} \in \mathbb{R}^d$ we have:*

$$\frac{1}{n}\sum_{i=1}^{n} \mathbb{E}_{\xi_i}\|\nabla F_i(\mathbf{x}_i, \xi_i) - \nabla f_i(\mathbf{x}_i)\|_2^2 \leq \sigma^2 + \frac{M^2}{n}\sum_{i=1}^{n}\|\nabla f_i(\mathbf{x}_i)\|_2^2, \tag{3.2}$$

where $\nabla F_i(\mathbf{x}, \xi_i)$, $i \in [n]$, denotes an unbiased stochastic gradient, i.e., $\mathbb{E}_{\xi_i}[\nabla F_i(\mathbf{x}, \xi_i)] = \nabla f_i(\mathbf{x})$.

Assumption 3. *We assume that there exists finite constants $G \geq 0$ and $B \geq 1$, such that for all $\mathbf{x} \in \mathbb{R}^d$ we have:*

$$\frac{1}{n}\sum_{i=1}^{n}\|\nabla f_i(\mathbf{x})\|_2^2 \leq G^2 + B^2\|\nabla f(\mathbf{x})\|_2^2. \tag{3.3}$$

These assumptions have appeared in literature before in [Kol] to study decentralized optimization with local iterations; and we extend their results and analyses by incorporating compression and momentum. This extension posed many fundamental technical difficulties, which we describe in detail in Section 3.4.

Remark 1 (Comparison with Existing Assumptions)**.** *Assumptions 2, 3 are weaker than assuming uniform bounds on the variance and the gradient dissimilarity: (i) The uniform bound on the variance [YJY19], i.e., $\mathbb{E}_{\xi_i}\|\nabla F_i(\mathbf{x}_i, \xi_i) - \nabla f_i(\mathbf{x}_i)\|_2^2 \leq \sigma_i^2$ for $i \in [n]$, implies Assumption 2 with $\sigma^2 = \frac{1}{n}\sum_{i=1}^{n}\sigma_i^2$ and $M = 0$; and (ii) The uniform bound on the gradient similarity [YJY19], i.e., $\frac{1}{n}\sum_{i=1}^{n}\|\nabla f_i(\mathbf{x}) - \nabla f(\mathbf{x})\|_2^2 \leq \kappa^2$, implies Assumption 3 with $G = \kappa$ and $B = 1$ – this follows from the identity $\frac{1}{n}\sum_{i=1}^{n}\|\nabla f_i(\mathbf{x}) - \nabla f(\mathbf{x})\|_2^2 = \frac{1}{n}\sum_{i=1}^{n}\|\nabla f_i(\mathbf{x})\|_2^2 - \|\nabla f(\mathbf{x})\|_2^2$. Both Assumptions 2 and 3 are weaker than the uniformly bounded second moment assumption $\mathbb{E}_{\xi_i}\|\nabla F_i(\mathbf{x}_i, \xi_i)\|_2^2 \leq G^2$, which has been standard in the stochastic optimization with compressed gradients*

[SCJ18b, BDKD19b, KLSJ20, ZHK19].

Our main convergence result (stated below) is for general smooth (non-convex) objectives; and our analysis can be readily extended to convex objectives. We derive this result for SQuARM-SGD under Assumptions 1-3 without event-triggered communication; in other words, our analysis is for compressed decentralized momentum SGD with local iterations. We would like to emphasize that incorporating event-triggering component into our analysis can only complicate the calculations and can be done; however, we omitted this because ours is the first analysis of communication-efficient (incorporating both compression and local iterations) decentralized SGD with Nesterov's momentum updates, which by itself hasn't been done before; and further, our analysis is under the weakest set of assumptions known to date. Therefore, in order to bring out the novelty of our convergence analysis without adding unnecessary technicality we decided to present our results without incorporating event-triggered communication.

Theorem 1. *Let $\mathcal{C}$ be a compression operator with parameter $\omega \in (0,1]$ and gap$(\mathcal{I}_T) = H$. Consider running SQuARM-SGD for T iterations with consensus step-size $\gamma = \frac{2\delta\omega^3}{4\delta^2\omega^2+\delta^2+128\lambda^2+24\omega^2\lambda^2}$, (where $\lambda = max_i\{1-\lambda_i(\mathbf{W})\}$), momentum coefficient $\beta \in [0,1)$, and constant learning rate $\eta = (1-\beta)\sqrt{\frac{n}{T}}$. Let the algorithm generate $\{\mathbf{x}_i^{(t)}\}_{t=0}^{T-1}$ for $i \in [n]$. Running the algorithm for $T \geq U_0$ for some constant U_0 defined in Appendix A, the averaged iterates $\overline{\mathbf{x}}^{(t)} := \frac{1}{n}\sum_{i=0}^{n}\mathbf{x}_i^{(t)}$ satisfy:*

$$\frac{\sum_{t=0}^{T-1}\mathbb{E}\|\nabla f(\overline{\mathbf{x}}^{(t)})\|_2^2}{T} = \mathcal{O}\left(\frac{J^2+\sigma^2+(M^2+n)G^2}{\sqrt{nT}} + \frac{(1-\beta)^2 nH^2((M^2+1)G+\sigma^2)}{T\delta^2\omega^3}\right),$$

where $J^2 < \infty$ is such that $\mathbb{E}[f(\overline{\mathbf{x}}^{(0)})] - f^* \leq J^2$.

We prove Theorem 1 in Appendix A. Note that we have used simplified convergence rate expressions in the above result, and derive precise rate expressions in the Appendix.

3.3.2 Theoretical Results with Bounded Second Moment of Stochastic Gradients

In this section, we consider a stronger set of assumptions than the ones before along with the smoothness of objectives, specifically:
(i) *Uniformly bounded variance:* For every $i \in [n]$, we have $\mathbb{E}_{\xi_i}\|\nabla F_i(\mathbf{x},\xi_i)-\nabla f_i(\mathbf{x})\|^2 \leq \sigma_i^2$, for some finite σ_i, where $\nabla F_i(\mathbf{x},\xi_i)$ denotes an unbiased stochastic gradient at worker i with $\mathbb{E}_{\xi_i}[\nabla F_i(\mathbf{x},\xi_i)] = \nabla f_i(\mathbf{x})$. We define $\bar{\sigma}^2 := \frac{1}{n}\sum_{i=1}^n \sigma_i^2$.
(ii) *Uniformly bounded second moment:* For every $i \in [n]$, we have $\mathbb{E}_{\xi_i}\|\nabla F_i(\mathbf{x},\xi_i)\|^2 \leq G^2 < \infty$.

Theorem 2. *Let $\mathcal{C}$ be a compression operator with parameter $\omega \in (0,1]$ and gap$(\mathcal{I}_T) = H$. Consider running SQuARM-SGD for T iterations with consensus step-size $\gamma = \frac{2\delta\omega}{64\delta+\delta^2+16\lambda^2+8\delta\lambda^2-16\delta\omega}$, (where $\lambda = max_i\{1-\lambda_i(\mathbf{W})\}$), a threshold sequence $c_t \leq \frac{c_0}{\eta^{1-\epsilon}}$ for all t where $\epsilon \in (0,1)$ and c_0 is a constant, momentum coefficient $\beta \in [0,1)$, and constant learning rate $\eta = (1-\beta)\sqrt{\frac{n}{T}}$. Let the algorithm generate $\{\mathbf{x}_i^{(t)}\}_{t=0}^{T-1}$ for $i \in [n]$. Then, we have the following guarantees:*

- *[Non-convex:] For $T \geq \max\{16L^2 n, \frac{8L^2\beta^4 n}{(1-\beta)^2}\}$, the averaged iterates $\overline{\mathbf{x}}^{(t)} := \frac{1}{n}\sum_{i=0}^n \mathbf{x}_i^{(t)}$ satisfy:*

$$\frac{\sum_{t=0}^{T-1}\mathbb{E}\|\nabla f(\overline{\mathbf{x}}^{(t)})\|_2^2}{T} = \mathcal{O}\left(\frac{J^2+\bar{\sigma}^2}{\sqrt{nT}} + \frac{c_0 n^{(1+\epsilon)/2}}{\delta^2 T^{(1+\epsilon)/2}} + \frac{nH^2G^2}{T\delta^4\omega^2} + \frac{\beta^4\bar{\sigma}^2}{T(1-\beta)^2}\right),$$

where $J^2 < \infty$ is such that $\mathbb{E}[f(\overline{\mathbf{x}}^{(0)})] - f^* \leq J^2$.

- *[Convex:] If $\{f_i\}_{i\in[n]}$ are convex, then for $T \geq \max\{(8L)^2 n, \frac{(8\beta^2 L)^4 n}{(1-\beta)^2}\}$, we have:*

$$\mathbb{E}[f(\overline{\mathbf{x}}_{avg}^{(T)})] - f^* = \mathcal{O}\left(\frac{\|\overline{\mathbf{x}}^{(0)} - \mathbf{x}^*\|^2 + \bar{\sigma}^2}{\sqrt{nT}} + \frac{c_0 n^{(1+\epsilon)/2}}{\delta^2 T^{(1+\epsilon)/2}} + \frac{n^{3/4}\beta^2 G^2}{(1-\beta)^{3/2} T^{3/4}} + \frac{nH^2G^2}{\delta^4\omega^2 T}\right),$$

where $\overline{\mathbf{x}}_{avg}^{(T)} := \frac{1}{T}\sum_{t=0}^{T-1}\overline{\mathbf{x}}^{(t)}$ for $\overline{\mathbf{x}}^{(t)} = \frac{1}{n}\sum_{i=1}^{n}\mathbf{x}_i^{(t)}$ and $\mathbf{x}^*$ is an optimizer of f with optimal value denoted by f^*.

We have used simplified convergence rate expressions in the above results, and provide precise rate expressions in the proofs provided in Appendix A for non-convex and convex objectives, respectively.

Effects of parameters on convergence: Observe that the factors H, c_0, ω, δ to achieve communication efficiency – H, c_0 for the event-triggered communication, ω for compression, and δ for the connectivity of the underlying graph – do not affect the dominant terms in convergence rate for either non-convex or convex objectives in Theorem 2 and appear only in the higher order terms. This implies that if we run SQuARM-SGD for sufficiently long, precisely, for at least $T_0 := C_0 \times \max\left\{\left(\frac{c_0^2 n^{(2+\epsilon)}}{(J^2+\bar{\sigma}^2)^2\delta^4}\right)^{1/\epsilon}, \frac{n}{(J^2+\bar{\sigma}^2)^2}\left(\frac{nG^2H^2}{\omega^2\delta^4} + \frac{\beta^4\bar{\sigma}^2}{(1-\beta)^2}\right)^2\right\}$ iterations for non-convex objectives and for $T_1 := C_1 \times \max\left\{\left(\frac{c_0^2 n^{2+\epsilon}}{\delta^4(\|\overline{\mathbf{x}}^{(0)}-\mathbf{x}^*\|^2+\bar{\sigma}^2)^2}\right)^{1/\epsilon}, \frac{n^3H^4G^2}{\delta^8\omega^4(\|\overline{\mathbf{x}}^{(0)}-\mathbf{x}^*\|^2+\bar{\sigma}^2)^2}, \frac{n^5G^8\beta^8}{(1-\beta)^6(\|\overline{\mathbf{x}}^{(0)}-\mathbf{x}^*\|^2+\bar{\sigma}^2)^4}\right\}$ for convex objectives with sufficiently large constants C_0 and C_1, respectively, then SQuARM-SGD converges at a rate of $\mathcal{O}\left(1/\sqrt{nT}\right)$. Note that this is the convergence rate of distributed *vanilla* SGD with the same speed-up w.r.t. the number of nodes n in both these settings. Thus, we essentially converge at the same rate as that of vanilla SGD, while saving significantly in terms of total communicated bits; this can also be seen in our numerical results in Section 3.5.

3.4 Preliminaries and Proof Outlines

In this section, we first establish a matrix notation which would be used throughout the proofs of Theorem 2. We then state SQuARM-SGD in matrix notation (which is equivalent to Algorithm 1) and list important facts regarding our updates. We conclude this section with a brief discussion of technical challenges involved in the proofs.

Matrix notation. Consider the set of parameters $\{\mathbf{x}_i^{(t)}\}_{i=1}^n$ at all nodes at timestep t as well as the estimates of the parameters $\{\hat{\mathbf{x}}_i^{(t)}\}_{i=1}^n$. The matrix notation is given by:

$$\begin{aligned}
\mathbf{X}^{(t)} &:= [\mathbf{x}_1^{(t)}, \ldots, \mathbf{x}_n^{(t)}] \in \mathbb{R}^{d\times n}\\
\hat{\mathbf{X}}^{(t)} &:= [\hat{\mathbf{x}}_1^{(t)}, \ldots, \hat{\mathbf{x}}_n^{(t)}] \in \mathbb{R}^{d\times n}\\
\bar{\mathbf{X}}^{(t)} &:= [\bar{\mathbf{x}}^{(t)}, \ldots, \bar{\mathbf{x}}^{(t)}] \in \mathbb{R}^{d\times n}\\
\mathbf{V}^{(t)} &:= [\mathbf{v}_1^{(t)}, \mathbf{v}_2^{(t)}, \ldots, \mathbf{v}_n^{(t)}] \in \mathbb{R}^{d\times n}\\
\boldsymbol{\nabla F}(\mathbf{X}^{(t)}, \boldsymbol{\xi}^{(t)}) &:= [\nabla F_1(\mathbf{x}_1^{(t)}, \xi_1^{(t)}), \ldots, \nabla F_n(\mathbf{x}_n^{(t)}, \xi_n^{(t)})] \in \mathbb{R}^{d\times n}
\end{aligned}$$

Here, $\nabla F_i(\mathbf{x}_i^{(t)}, \xi_i^{(t)})$ denotes the stochastic gradient at node i at timestep t and the vector $\bar{\mathbf{x}}^{(t)}$ denotes the average of node parameters at time t, specifically: $\bar{\mathbf{x}}^{(t)} := \frac{1}{n}\sum_{i=1}^n \mathbf{x}_i^{(t)}$.

Let $\Gamma^{(t)} \subseteq [n]$ be the set of nodes that do not communicate at time t. We define $\mathbf{P}^{(t)} \in \mathbb{R}^{n\times n}$, a diagonal matrix with $\mathbf{P}_{ii}^{(t)} = 0$ for $i \in \Gamma^{(t)}$ and $\mathbf{P}_{ii}^{(t)} = 1$ otherwise.

SQuARM-SGD in matrix notation. Consider Algorithm 1 with synchronization indices given by the set $\mathcal{I}_T = \{0, H, 2H \ldots, mH, \ldots\} \subseteq [T]$ for some constant $H \in \mathbb{N}$. Using the above notation, the sequence of parameters' updates from synchronization index mH to $(m+1)H$, is given by:

$$\mathbf{V}^{(t)} = \beta \mathbf{V}^{(t-1)} + \boldsymbol{\nabla F}(\mathbf{X}^{(t)}, \boldsymbol{\xi}^{(t)}) \tag{3.4}$$

$$\mathbf{X}^{((m+1/2)H)} = \mathbf{X}^{I_{(t)}} - \sum_{t'=mH}^{(m+1)H-1} \eta \left(\beta \mathbf{V}^{(t')} + \boldsymbol{\nabla F}(\mathbf{X}^{(t')}, \boldsymbol{\xi}^{(t')})\right) \tag{3.5}$$

$$\hat{\mathbf{X}}^{((m+1)H)} = \hat{\mathbf{X}}^{(mH)} + \mathcal{C}((\mathbf{X}^{((m+1/2)H)} - \hat{\mathbf{X}}^{(mH)})\mathbf{P}^{((m+1)H-1)}) \tag{3.6}$$

$$\mathbf{X}^{((m+1)H)} = \mathbf{X}^{((m+1/2)H)} + \gamma \hat{\mathbf{X}}^{((m+1)}(\mathbf{W} - \mathbf{I}) \tag{3.7}$$

where $\mathcal{C}(.)$ denotes the compression operator applied column-wise to the argument matrix and $\mathbf{I}$ is the identity matrix. Note that in the update rule for $\hat{\mathbf{X}}^{((m+1)H)}$, we used (i) the fact that $\mathbf{P}$ is a diagonal matrix and that $\mathcal{C}$ is applied column-wise to write $\mathcal{C}(\mathbf{X}^{((m+1/2)H)} - \hat{\mathbf{X}}^{(mH)})\mathbf{P}^{((m+1)H-1)} = \mathcal{C}((\mathbf{X}^{((m+1/2)H)} - \hat{\mathbf{X}}^{(mH)})\mathbf{P}^{((m+1)H-1)})$, and (ii) that $\hat{\mathbf{X}}^{((m+1)H-1)} = \hat{\mathbf{X}}^{(mH)}$, because $\hat{\mathbf{X}}$ does not change in between the synchronization indices.

We now note some useful properties of the iterates in matrix notation which would be used throughout the paper:

1. Since $\mathbf{W} \in [0,1]^{n \times n}$ is a doubly stochastic matrix, we have: $\mathbf{W} = \mathbf{W}^T, \mathbf{W1} = \mathbf{1}$

and $\mathbf{1}^T\mathbf{W} = \mathbf{1}^T$ (where $\mathbf{1}$ is the all ones vector in $\mathbb{R}^n$). This also gives us:

$$\bar{\mathbf{X}}^{(t)} := \mathbf{X}^{(t)}\frac{1}{n}\mathbf{1}\mathbf{1}^T, \qquad \bar{\mathbf{X}}^{(t)}\mathbf{W} = \bar{\mathbf{X}}^{(t)} \tag{3.8}$$

where the first expression follows from the definition of $\bar{\mathbf{X}}^{(t)}$ and the second expression follows because $\mathbf{W}\frac{\mathbf{1}\mathbf{1}^T}{n} = \frac{\mathbf{1}\mathbf{1}^T}{n}\mathbf{W} = \frac{1}{n}\mathbf{1}\mathbf{1}^T$.

2. The average of the iterates in Algorithm 1 follows :

$$\bar{\mathbf{X}}^{(t+1)} = \bar{\mathbf{X}}^{(t+\frac{1}{2})} + \mathbf{1}_{(t+1)\in\mathcal{I}_T}\left[\gamma\hat{\mathbf{X}}^{(t+1)}(\mathbf{W}-\mathbf{I})\frac{1}{n}\mathbf{1}\mathbf{1}^T\right] = \bar{\mathbf{X}}^{(t+\frac{1}{2})} \tag{3.9}$$

where $\mathcal{I}_T$ denotes the set of synchronization indices of Algorithm 1. The above follows because $(\mathbf{W}-\mathbf{I})\frac{1}{n}\mathbf{1}\mathbf{1}^T = \mathbf{W}\frac{\mathbf{1}\mathbf{1}^T}{n} - \frac{\mathbf{1}\mathbf{1}^T}{n} = \mathbf{0}$.

Proposition 1 (Variance Reduction with Independent Samples)**.** *Consider the variance bound* (3.2) *on the stochastic gradient for nodes. If* $\boldsymbol{\xi}^{(t)} = \{\xi_1^{(t)}, \xi_2^{(t)}, \ldots, \xi_n^{(t)}\}$ *denotes the collection of independent stochastic samples for the nodes at any time-step* t*. Then we have:*

$$\mathbb{E}_{\xi_{(t)}}\left\|\frac{1}{n}\sum_{i=1}^{n}\nabla\Big(F_i(\mathbf{x}_i^{(t)}, \xi_i^{(t)}) - \nabla f_i(\mathbf{x}_i^{(t)})\Big)\right\|^2 \leq \frac{\sigma^2}{n} + \frac{M^2}{n^2}\sum_{i=1}^{n}\left\|\nabla f_i(\mathbf{x}_i^{(t)})\right\|_2^2. \tag{3.10}$$

Proposition 2. *For any* t*,* $\mathbb{E}\left\|\mathbf{V}^{(t)}\right\|_F^2$ *is bounded as follows:*

$$(1-\beta)\mathbb{E}\left\|\mathbf{V}^{(t)}\right\|_F^2 \leq \Lambda^{(t)} := \sum_{k=0}^{t}\beta^{t-k}\mathbb{E}\left\|\nabla\mathbf{F}(\mathbf{X}^{(k)}, \boldsymbol{\xi}^{(k)})\right\|_F^2. \tag{3.11}$$

We prove the above propositions in Appendix A in supplementary material.

Technical Challenges: We focus on two major aspects of our work to compare with existing literature: (i) Analysis of compressed decentralized training with triggered communication with mild assumptions. (ii) Performing the resulting analysis by taking into account the momentum updates.

Most works in communication efficient decentralized training assume a bound on the second moment of stochastic gradients [SDGD19, KLSJ20, KSJ19b, TYL$^+$19]. This assumption can be quite strong for settings where the data distribution among the clients is heterogeneous as gradient dissimilarity between the clients can be bounded trivially using the second moment bound (see the note on comparison of assumptions in Remark 1 on page 25). In contrast, we work with a much weaker set of assumptions (see Section 3.3) by not assuming any uniform bound on norm of stochastic gradients, and further allow both the gradient diversity and the variance of stochastic gradients to scale with the norm of gradients compared to existing works [YJY19]. Performing the analyses with these relaxed assumptions is highly challenging as it requires us to carefully consider the error due to quantization, local iterations and triggering per communication round and construct a recursion equation for it and then delicately handle the recursion to bound the error for any time index. We remark that the assumptions considered in our paper have appeared in literature before in [Kol] to study decentralized optimization with only local iterations; our work is a significant extension of their results and analyses as we incorporate compression and momentum as we discuss below.

While momentum updates are almost always used in practice to empirically speedup the training process and to improve generalization performance, it has remained unclear whether convergence with linear speedup with number of nodes n (as in the case of SGD without momentum [LZZL17, BDKD19b, SDGD19, Kol]) is still possible when using momentum. Recently, [YJY19, ZHK19] provided a positive answer to this question, where [YJY19] studies local SGD with momentum in a decentralized setup, but *without* any compressed or event-triggered communication, and [ZHK19] studies compressed *distributed* SGD with momentum for non-convex objectives, but without local iterations or event-triggered communication. Our work is the first to provide convergence rates showing linear speedup with n for compressed *decentralized* optimization using momentum while incorporating local iteration and triggered communication in the analysis, further, with a weaker set of assumptions than considered in previous works. To achieve this, our convergence proofs require the use of virtual sequences which have been promising lately in stochastic optimization; see, for example, [SCJ18b, AHJ$^+$18b, KRSJ19b, BDKD19b, YJY19, ZHK19].

We would like to emphasize that even without momentum and local iterations, analyzing compression in decentralized optimization [KSJ19b, KLSJ20, SDGD19] (whose analysis does not require virtual sequences) is significantly more involved and requires different technical tools than analyzing compression in distributed optimization [AHJ$^+$18b, KRSJ19b]. One of the main reasons for this is as follows: In a decentralized setup, we need to separately show that nodes eventually reach to the same parameters (i.e., consensus happens), which happens trivially in a distributed setup, because in each iteration all worker nodes have the same parameters sent by the master node. On top of that, incorporating momentum updates (which has only been analyzed with compression in distributed setups so far) in decentralized setting is non-trivial and gives similar challenges.

As a consequence, it is not surprising that our proofs are fundamentally different and significantly more challenging from existing works, including [ZHK19, YJY19, KSJ19b, KLSJ20, SDGD19, Kol], as we study momentum updates for decentralized setup with compression, local iterations and event-triggered communication to save on communication bits. As opposed to [ZHK19], we allow *heterogeneous* data setting, where different nodes may have different datasets. Moreover, with all these, we achieve vanilla SGD like convergence rates for both non-convex and convex objectives.

3.5 Experiments

In this section, we compare SQuARM-SGD with CHOCO-SGD [KLSJ20] , which only employs compression (sparsification *or* quantization) and is the state-of-the-art in communication efficient decentralized training. We also provide additional plots for comparison of accuracy vs. wall-clock time and performance over other datasets in sections 3.5.2 and 3.5.4 below.

3.5.1 Experiment on ImageNet dataset

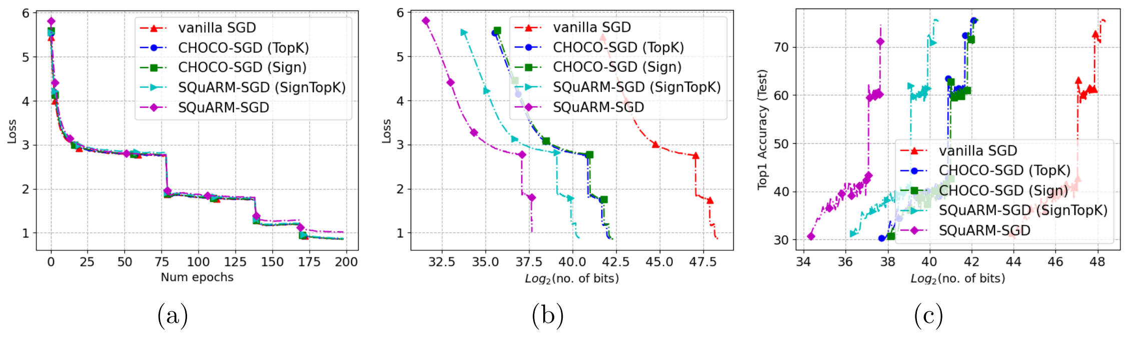
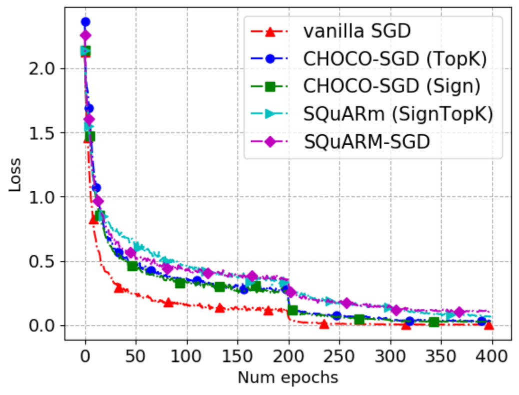


Figure 3.1 Figures 3.2(a) and 3.1(b) correspond to the training loss for the ResNet50 model on the ImageNet dataset compared against no. of epochs and total communicated bits, respectively. Figure 3.2(b) shows comparison for Top-1 test accuracy as function of total number of bits communicated.

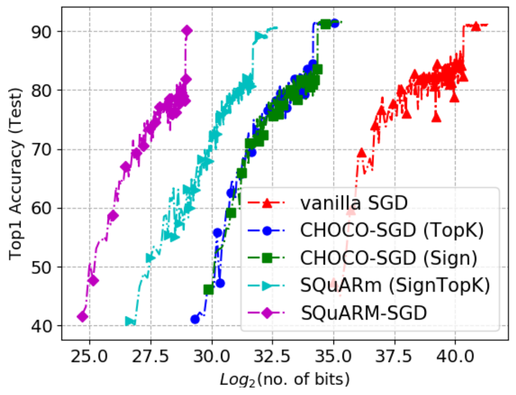
Setup. We compare our algorithm with CHOCO-SGD [KLSJ20] on a large scale learning task of training ResNet50 [HZRS16] model on ImageNet [DDS$^+$09] dataset with $n = 6$ nodes connected in a ring topology. Learning rate is initialized to 0.1, following a schedule consisting of a warmup period of 5 epochs followed by piecewise decay of 5 at epochs 80,140, 170, and we stop training at epoch 200. The SGD algorithm is implemented with momentum with a factor of 0.9 and mini-batch size of 96. SQuARM-SGD consists of $H = 5$ local iterations followed by checking for a triggering condition, and then communicating with the composed *SignTopK* operator, where we take top 1% elements of each tensor and only transmit the sign and norm of the result. The triggering threshold follows a schedule piecewise constant: initialized to 1.0 and increases by 0.25 after every 20 epochs till 100 epochs are complete; while maintaining that $c_t < {}^1/_\eta$ for all t. We compare performance of SQuARM-SGD against CHOCO-SGD with *Sign* and *TopK* compression (taking top 1% of elements of the tensor) and decentralized vanilla SGD. We also provide a plot for using the composed *SignTopK* operator without event-triggering titled 'SQuARM-SGD (Sign-TopK)' for comparison.

Results. We plot global loss function evaluated at the average parameter vector across nodes vs. number of epochs in Figure 3.2(a), where we observe SQuARM-SGD converging at a similar rate as CHOCO-SGD and vanilla decentralized SGD. Figure 3.1(b) shows the evolution of training loss as a function of number of bits, demonstrating that SQuARM-SGD uses significantly less bits than CHOCO-SGD or vanilla-SGD to train the model. Figure 3.2(b) shows the performance for a given bit-budget, where we show the Top-1 test accuracy as a function of the total bits communicated. For Top-1 test accuracy of around 74%, SQuARM requires about $20\times$ less bits than CHOCO with *Sign* or *TopK* compression, and around $1400\times$ less bits than vanilla decentralized SGD to achieve the same Top-1 accuracy.

3.5.2 Experiments on CIFAR-10 dataset



(a)



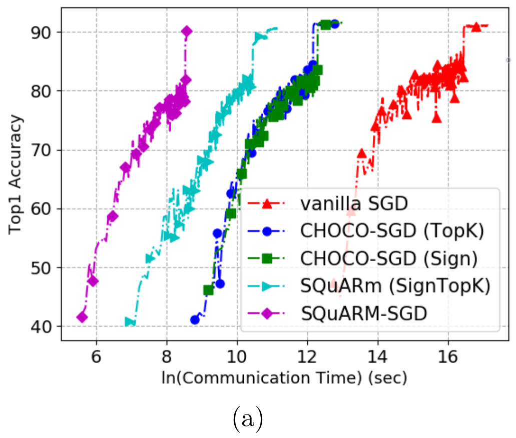
(b)

Figure 3.2 These plots are for training a ResNet20 model (non-convex objective) on the CIFAR-10 dataset. Figure 3.2(a) and 3.2(b) show training loss vs. epochs and Top-1 accuracy vs. total number of bits communicated, respectively.

Setup. We match the setting in CHOCO-SGD [KLSJ20] and perform our experiments on the CIFAR-10 [KNH09] dataset and train a ResNet20 [WWW$^+$16] model with $n = 8$ nodes connected in a ring topology. The training is done on TitanRTX GPUs. Learning rate is initialized to 0.1, following a schedule consisting of a warmup period of 5 epochs followed by piecewise decay of 5 at epoch 200 and 300 and we stop training at epoch 400. The SGD algorithm is implemented with momentum with a factor of 0.9 and mini-batch size of 256. SQuARM-SGD consists of $H = 5$ local iterations and we take top 1% elements of each tensor and only transmit the sign and norm of the result. The triggering threshold follows a schedule piecewise constant: initialized to 2.5 and increases by 1.5 after every 20 epochs till 350 epochs are complete; while maintaining that $c_t < {}^1\!/\eta$ for all t. We compare performance of SQuARM-SGD against CHOCO-SGD with *Sign*, *TopK* compression (taking top 1% of elements of the tensor) and decentralized vanilla SGD [LZZ$^+$17].

Results. From Figure 3.2(a), we observe SQuARM-SGD converging at a similar rate as CHOCO-SGD and vanilla decentralized SGD. Figure 3.2(b) shows the performance for a given bit-budget, where we show the Top-1 test accuracy as a function of the total number of bits communicated. For Top-1 test-accuracy of around 90%, SQuARM requires about 40× less bits than CHOCO with *Sign* or *TopK* compression, and around 3K× less bits than vanilla decentralized SGD to achieve the same Top-1 accuracy.

3.5.3 Wall-clock time comparison for CIFAR-10 dataset with rate-limited communication



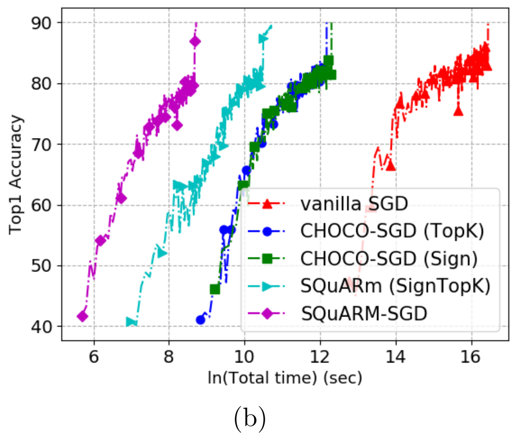


Figure 3.3 Figure 3.3(a) shows Top-1 accuracy as a function of total communication time through a bandlimited rate pipe of 100Kbps. Figure 3.3(b) shows the total time (computation + communication) required to reach a target Top-1 accuracy of 90% for the different schemes.

Setup. We follow the same setup as stated in the Subsection 3.5.2, and further we assume that the communication between workers is rate-limited to 100Kbps. This is typical *average* rate of wireless edge devices sharing a common bandwidth with other devices, therefore devices cannot have sustained high rates.

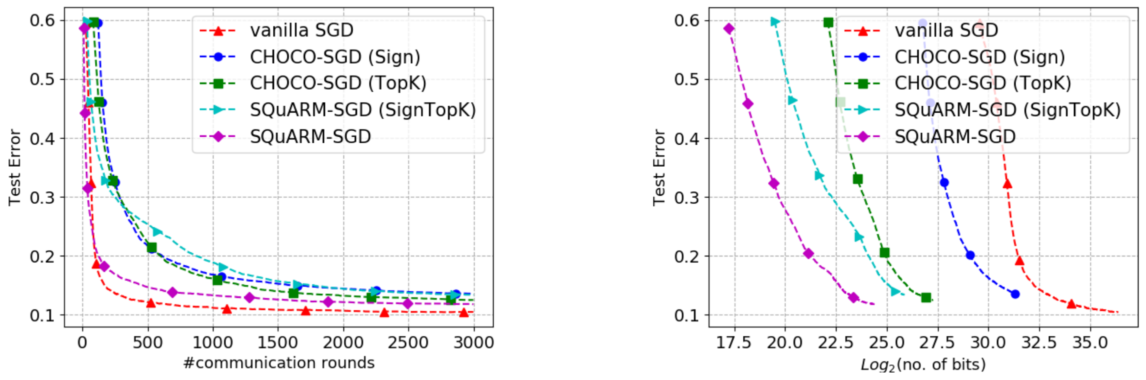
Results. Figure 3.3(a) shows the Top-1 accuracy as a function of time required to communicate bits through a rate pipe constrained to 100Kbps. We observe that, to reach a target accuracy of 90%, SQuARM-SGD saves a factor of around 40× in communication time compared to CHOCO-SGD with *Sign* or *TopK* sparsifier, and around 3K× less communication time than vanilla SGD. Thus, there is a significant saving in communication time for SQuARM-SGD compared to other schemes when communicating over typical rate limited channels. This result is almost a direct translation of the savings in the number of bits needed for training.

In Figure 3.3(b), we plot Top-1 accuracy vs. the wall-clock time, when the communication between workers is over the rate-limited pipelines (limited to 100Kbps). The wall-clock time includes the total time taken in both computation and communication; note that this includes time required to setup the communication (*e.g.,* interrupts in processing) and other overheads, and not just the airtime (time to transmit the bits) for the communication as done in Figure 3.3(a). We observe that, to reach a target test-accuracy of around 90%, SQuARM-SGD saves a factor of around 30× in total time compared to CHOCO-SGD with *Sign* or *TopK* compression, and around 2.5K× compared to vanilla-SGD. The significant advantage to our method comes due to efficient transmission over the rate constrained pipeline.

Note that one can expect uncompressed schemes to do better in terms of computation time, but as shown in the plots above, the high communication time required in uncompressed schemes on rate-constrained links would far outweigh the advantage due to computation time. As a result, the total time required by SQuARM-SGD (that incorporates compression and infrequent communication) to reach a certain accuracy would be significantly smaller than either the uncompressed schemes or the ones using infrequent communication or both.

3.5.4 Experiments for convex objective on MNIST dataset

Setup. We run SQuARM-SGD on MNIST dataset and use multi-class cross-entropy loss to model the local objectives $f_i, i \in [n]$. We consider $n = 60$ nodes connected in a ring topology, each processing a mini-batch size of 5 per iteration and having heterogeneous distribution of data across classes. We work with $\eta_t = 1/(t + 100)$ (based on grid search) and synchronization index $H = 5$. For SQuARM-SGD, we use the composed operator $SignTopK$ ([BDKD19b]) with $k = 10$ (out of 7840 length vector for MNIST dataset). For our experiments, we set the triggering constant $c_0 = 5000$ in and keep it unchanged until a certain number of iterations and then increase it periodically; while still maintaining that $c_t\eta_t^2$ decreases with t (as c_t is $o(t)$) .



(a) (b)

Figure 3.4 Figure 3.4(a) and 3.4(b) are for convex objective simulated on the MNIST dataset, where we plot test error vs number of communication rounds and test error vs total number of bits communicated, respectively, for different algorithms.

Results. In Figure 3.4(a), we observe SQuARM-SGD can reach a target test error in fewer communication rounds while converging at a rate similar to that of vanilla SGD. The advantage to SQuARM-SGD comes from the significant savings in the number of bits communicated to achieve a desired test error, as seen in Figure 3.4(b): to achieve a test error of around 0.12, SQuARM-SGD gets 120× savings as compared to CHOCO-SGD with *Sign* quantizer, around 10-15× savings than CHOCO-SGD with $TopK$ sparsifier, and around 1000× savings than vanilla decentralized SGD.

3.6 Conclusion

In this chapter, we developed a communication efficient framework for decentralized training for clients lying on a graph topology. We used the ideas of compression and local iterations, combined with a threshold based triggering approach to propose an algorithm which significantly saves on communication bits while ensuring collaborative learning between the network clients. We study the convergence rate of our algorithm for strongly-convex and non-convex smooth objectives, proving that communication efficiency using these ideas can be achieved while still ensuring a convergence rate similar to vanilla decentralized SGD training. To the best of our knowledge, our work provides the first convergence analysis for compressed decentralized training with momentum using a weaker set of assumptions than existing literature while incorporating the local SGD and event triggered communication framework. Furthermore, our analysis for convergence rate includes the effect of momentum updates, thus taking a step forward in theoretically analyzing the role of momentum updates in compressed decentralized training, which has not yet been explored. Our experimental results show significant saving of our proposed technique over the current state-of-the-art, thus fulfilling the need for highly-communication efficient collaborative learning algorithms for rate-limited networks.

CHAPTER 4

Decentralized Multi-Task Stochastic Optimization With Compressed Communications

Decentralized optimization has become a cornerstone in the development of modern distributed machine learning systems, especially in scenarios where data is distributed across multiple devices or nodes. While in the previous chapter we primarily focused on achieving communication efficiency in federated learning, it is equally important to address the heterogeneity across nodes by allowing for personalized models. As before, allowing for collaboration in the context of federated learning could potentially allow for better quality of trained models. We will refer to this technicality of personalization as allowing for solving *multiple tasks* – where tasks could refer to the underlying optimization problem being solved at each node. Since each node often possesses unique data and distinct objectives, multi-task learning is essential in applications where a one-size-fits-all approach does not adequately capture the diverse data characteristics and requirements of different nodes.

In this chapter, we delve into the problem of multiple task learning in the context of communication-efficient decentralized optimization. We consider a multi-agent network where each node has a stochastic local cost function that depends on the node's decision variable and a random variable, with additional pairwise constraints on the decision variables of neighboring nodes. The network's aggregate objective function is composed additively of the expected values of the local cost functions at the nodes, and the overall goal is to find the minimizing solution to this aggregate objective function subject to all pairwise constraints.

To achieve this, we develop algorithms that allow decentralized information processing and local computation, with only compressed information exchanges permitted between neighboring nodes. This approach ensures communication efficiency while addressing the need for personalized optimization. Our algorithms provide rigorous performance bounds, demonstrating that they can achieve robust optimization despite the use of compressed communication.

Our analysis shows that the deviation from the global minimum value and the violations of the constraints are upper-bounded by $O(T^{-\frac{1}{2}})$ and $O(T^{-\frac{1}{4}})$, respectively, where T is the number of iterations. These results indicate that the performance of our decentralized algorithms with communication compression is comparable to that of algorithms without compression.

The development of these algorithms is significant as it highlights the feasibility of achieving high-performance personalized optimization in a communication-efficient manner. This capability is critical for applications in various domains, such as Internet of Things (IoT) networks, distributed sensor networks, and collaborative multi-agent systems, where nodes have diverse objectives and constraints.

By focusing on personalized models in decentralized optimization, this chapter contributes to the broader goal of making federated learning more adaptable and practical, addressing the specific needs of individual nodes while ensuring communication efficiency.

4.1 Introduction

The emergence of multi-agent networks and the need to distribute computation across different nodes which have access to only piece of the network-wide data but are allowed to exchange information under some resource constraints, have accelerated research efforts on decentralized and distributed optimization in multiple communities, particularly during the last 10-15 years. Spearheading this activity has been decentralized consensus optimization in static settings, where the goal is to minimize the sum of local cost functions, toward which [NO09a] proposed a decentralized sub-gradient algorithm, whose convergence was further studied in [YLY16]. Following this initial work, several other consensus algorithms were introduced and studied, including alternating direction method of multipliers (ADMM) [SLY$^+$14], exact first-order algorithm [SLWY15], stochastic consensus optimization [SVKB17, LLZ20], and online consensus optimization with time-varying cost functions [MNC14, AGL17a].

Consensus modeling framework requires, in essence, all nodes to converge to the same value. This however may not be appropriate in many network scenarios, where different nodes, even neighboring ones, may ultimately end up with different decision (or action) values. Such a scenario arises in, for example, distributed multitask adaptive signal processing, where the weight vectors at neighboring nodes are not the same [CRS14, NRFS16]. One of the first papers that has analyzed such departure from consensus optimization is [KSR17], where the formulation included proximity constraints between neighboring nodes, which were handled through construction of Lagrangians and using saddle-point algorithms, and extended to the asynchronous setting in [BKR19].

Decentralized algorithms are built on the assumption that there is some exchange of information among the nodes (at least among the neighboring nodes) which then propagates across the network towards achieving the global optimum in the limit. Extensive and frequent exchange of such information is generally practically impossible (due to bandwidth constraints on the edges of the underlying network which constitute the communication links, and computation and storage limitations, among many others), which inherently brings in a restriction on the amount and timing of the exchange of relevant current data. In the literature several studies have addressed these limitations through quantization of information or actions [KBS07,EB16,BEO16, CLB16, ZC16], by using only sign information on some differences [ZYB19, CB21a], by controlling the timing of transmissions through event triggering [CB21b, SDGD21a, SDGD23], or by sparsification [AHJ+18a, KSJ19a, SCJ18a]. Quantization in the context of decentralized optimization (and not consensus problems) has also been studied, with some of the algorithms leading to nonzero errors in convergence (see the early work [RN05, NOOT08]) and others to exact convergence [RMHP19]; see also [AGL+17b] for quantized stochastic optimization. Some recent work has also used error-compensated compression in decentralized optimization, such as [KSJ19a, WHHZ18,SDGD20a,SDGD23]. Recently, error-compensated compressed decentralized training for *online* convex optimization was considered in [CB23].

Most of the existing works on decentralized optimization with quantized/compressed communications are, as discussed above, focused on either *c*onsensus optimization or *u*nconstrained optimization. Research departing from that trend was initiated in [CB20], which addressed the problem of multitask learning (or distributed optimization with pairwise constraints) using quantized communications. More specifically, the model adopted in that paper (with an underlying network topology) associated with each node a stochastic (local, individual) cost and with each pair of neighbors an inequality constraint, e.g., proximity constraint. Note that in such a formulation, different from consensus problems, each node has its own decision variable, but these cannot be picked independently because of the pairwise constraints. Further, the distribution of the random variable in the stochastic local cost function of each node is unknown and each node operates based on sequential feedback information, rendering the formulation distinct from deterministic optimization. The paper developed stochastic saddle-point algorithms with quantized communications between neighbors, and studied the impact of quantization on the optimization performance. One shortcoming of the result of [CB20] is that the scheme developed led to nonzero convergence error; said differently, the algorithm in that work does not lead to convergence to the exact optimal solution as the number of iterations grow. This is precisely the issue we address in this chapter, and achieve exact convergence by employing a saddle-point algorithm along with an approach based on error-compensated communication compression. Before further discussing the contents and contributions of this work, let us point out that saddle-point algorithms (a.k.a. primal-dual algorithms) have been extensively used in literature on constrained optimization, such as deterministic centralized optimization [AHU58, NO09b], decentralized optimization [CNS14], stochastic optimization [KSR17, BKR19, EZC$^+$19], and online optimization [MJY12, CL19].

4.1.1 Contributions

In this chapter, we address the problem of decentralized multi-agent stochastic optimization on a network, where each agent has a local stochastic convex cost function and each pair of neighbors is associated with an inequality constraint. The overall goal is to minimize the total (additive) expected cost of all agents subject to all the constraints on all edges, with all computation carried out at the nodes and with information exchanged among the nodes using compressed communication. We consider two scenarios of interest based on the sample information available locally at the nodes:

- *S*ample Feedback: Each node has access to the local samples of the random variable affecting its local cost function drawn from its distribution at any time instance during the optimization process, and can thus evaluate its cost function and its gradient.

- *B*andit Feedback: Nodes do not have access to the samples, but rather only observe values of the corresponding local cost functions at two points sufficiently close to the original node parameter. For references on bandit feedback in context of optimization (a.k.a. zeroth-order optimization), see [FKM05, ADX10, DJWW15, Sha17, LKC$^+$18, HHG19].

Under both scenarios, the chapter develops a decentralized saddle-point algorithm which leads to zero convergence error, even with a *finite* number of bits for each iteration. Note that previous works in this topic [CB20] required the number of bits to be unbounded for the error to diminish. Specifically, under some standard assumptions, we show that the expected sub-optimality and the expected constraint violations are upper bounded by $O(T^{-\frac{1}{2}})$ and $O(T^{-\frac{1}{4}})$, respectively, where T is the number of iterations, despite the proposed algorithm using a finite number of bits. These bounds match, in order sense, the bounds for algorithms without communication compression. Hence, we get near optimal optimization performance even with finite number of bits under both scenarios. The chapter also provides results of numerical experiments, which corroborate these bounds.

Accordingly, the main contributions of this chapter are:

- Using finite bit compressed sample feedback, with T being the horizon of the optimization problem, achieving $O(1/\sqrt{T})$ closeness to optimum value of the objective function, and achieving $O(T^{-\frac{1}{4}})$ constraint violation—both being the same as in the case without compression.
- Obtaining the same order bounds with bandit feedback, using only two-point feedback values.

4.1.2 Chapter Organization

The rest of the chapter is organized as follows: Section 4.2 provides a precise formulation of the problem under consideration. Section 4.3 develops the saddle-point algorithm (Algorithm 2) under sample feedback, and provides convergence results and performance bounds (Theorem 3) along with essential points of the analyses and proofs. Section 4.4 presents the counterpart of Section 4.3 for bandit feedback, with the corresponding algorithm (Algorithm 3) and corresponding main result on convergence and performance bounds (Theorem 4). Section 5.5 discusses results of some numerical experiments. Section 4.6 provides some concluding remarks. Omitted technical details can be found in Appendix B of this thesis.

4.2 Problem Formulation

We consider an undirected graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ with $|\mathcal{V}| = n$ nodes and $|\mathcal{E}| = m$ connected edges. We assume that each connected node pair $(i, j) \in \mathcal{E}$ allows for bi-directional communication from i to j and j to i. The neighbor set of the node i

is denoted by $\mathcal{N}_i$.

Associated with each node $i \in [n] := \{1, 2, \ldots, n\}$ is an unknown data distribution which we denote by $\mathcal{P}_i$. The samples generated from the distribution are denoted by $\xi_i \sim \mathcal{P}_i$ where $\xi_i \in \Xi_i$. Each node also has a local cost function $f_i : \mathcal{X} \times \Xi_i \to \mathbb{R}^+$ which takes as input a sample $\xi_i \in \Xi_i$ and a local parameter $\mathbf{x}_i \in \mathcal{X} \subset \mathbb{R}^d$ to yield the sample cost $f_i(\mathbf{x}_i, \xi_i)$. Here, the set $\mathcal{X}$ corresponds to the set of feasible parameters the node can choose from, which is the same across all nodes. As an example, for supervised image recognition tasks, the sample ξ_i for a node i may correspond to an image-label pair with the set $\mathcal{X}$ being the set of all neural networks with a width of 2 layers and $\mathbf{x}_i$ a particular 2-layer neural network. The local objective f_i in this case may denote a cross-entropy loss function evaluated using the given image-label pair and the neural network. The expected cost for a node i for parameter $\mathbf{x}_i \in \mathcal{X}$ is denoted by $F_i(\mathbf{x}_i) = \mathbb{E}_{\xi_i \sim \mathcal{P}_i}[f_i(\mathbf{x}_i, \xi_i)]$. In general, we are interested in minimizing the expected cost for all the nodes $i \in [n]$. That is, we are interesting in finding node parameters $\{\mathbf{x}_i\}_{i=1}^n$ that minimize the cost $F(\mathbf{x}) := \sum_{i=1}^n F_i(\mathbf{x}_i)$ where $F_i(\mathbf{x}_i)$ denotes the expected cost of the node i evaluated using parameter $\mathbf{x}_i$ and $\mathbf{x} \in \mathcal{X}^n$ denotes stacking of all the individual node parameters $\{\mathbf{x}_i\}_{i=1}^n$. Further, we assume that the node parameters are related via pairwise constraints on the connected nodes in the graph. Specifically, for any $i \in [n]$ and $j \in \mathcal{N}_i$, there is a function $g_{ij} : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$ such that the inequality $g_{ij}(\mathbf{x}_i, \mathbf{x}_j) \leq 0$ should be satisfied. This may, for example, encode a proximity constraint on the node parameters by having $g_{ij}(\mathbf{x}_i, \mathbf{x}_j) = \|\mathbf{x}_i - \mathbf{x}_j\|_2^2 - c_{ij}$ where $\|.\|_2$ denotes the ℓ_2 norm and $c_{ij} \geq 0$ is a constant. In this chapter, we assume that the constraint functions $g_{ij}(\mathbf{x}_i, \mathbf{x}_j)$ are symmetric in their parameters, i.e., $g_{ij}(\mathbf{x}_i, \mathbf{x}_j) = g_{ji}(\mathbf{x}_j, \mathbf{x}_i)$ for all $\mathbf{x}_i, \mathbf{x}_j \in \mathcal{X}$ and connected node pairs $(i, j) \in \mathcal{E}$, which leads to m number of distinct pairwise constraints for all the parameters. With the notation now in place, we state the learning objective for the multi-task problem can be stated as follows:

$$\min_{\mathbf{x}\in\mathcal{X}^n} F(\mathbf{x}) = \sum_{i=1}^{n} \mathbb{E}_{\boldsymbol{\xi}_i}[f_i(\mathbf{x}_i, \boldsymbol{\xi}_i)] \tag{4.1}$$
$$\text{subject to} \quad g_{ij}(\mathbf{x}_i, \mathbf{x}_j) \le 0, \quad \forall i \in [n], j \in \mathcal{N}_i$$

To solve the problem in (4.1) in a decentralized manner, the nodes need to communicate during the optimization procedure which can be prohibitive for low bandwidth links or when the exchanged information updates among the nodes are large. To this end, in this chapter we consider compression of the information exchanges among the nodes to make the communication efficient. We employ the notion of the compression operator proposed in [SCJ18a]:

Definition 2. *A (possibly randomized) function $\mathcal{C} : \mathbb{R}^d \to \mathbb{R}^d$ is called a compression operator, if there exists a constant $\omega \in (0, 1)$, such that for every $\mathbf{x} \in \mathbb{R}^d$:*

$$\mathbb{E}\|\mathbf{x} - \mathcal{C}(\mathbf{x})\|_2^2 \le (1 - \omega)\|\mathbf{x}\|_2^2 \tag{4.2}$$

where expectation is taken over the randomness of $\mathcal{C}$. We assume $\mathcal{C}(\mathbf{0}) = \mathbf{0}$.

We consider two scenarios of interest based on the sampled information available locally at the nodes:

1. Sample Feedback: In this scenario we assume that each node i has access to the local samples ξ_i drawn from $\mathcal{P}_i$ at any time instance during the optimization procedure and can thus evaluate the cost function and its derivative.

2. Bandit Feedback: In this scenario, nodes do not have a direct access to the

samples, but rather can only observe values of the local cost function at two perturbations from the original node parameter.

We focus on these scenarios separately in Section 4.3 and Section 4.4 respectively, where we develop a compressed decentralized algorithm for optimizing (4.1) for each, and present our theoretical convergence results, the complete proofs for which are provided in Appendix B.

4.3 Decentralized compressed optimization with Sample feedback

In this section we describe our approach for optimizing the objective in (4.1) for the case of *sample feedback*. In this setting, each node $i \in [n]$ has access to the sampled instance ξ_i at any stage of the optimization procedure, and thus can evaluate the local objective $f_i(\mathbf{x}_i, \xi_i)$ based on its local parameter $\mathbf{x}_i$.

4.3.1 Algorithm: with Sample Feedback

We develop a stochastic saddle-point algorithm for solving (4.1) in a decentralized manner with compressed parameter exchanges. Our proposed scheme is presented in Algorithm 2 and is based on finding a saddle point of the modified Lagrangian for the optimization problem in (4.1). For a given sample ξ_i, we define this modified Lagrangian as follows:

$$\mathcal{L}(\mathbf{x}, \boldsymbol{\lambda}) = \sum_{i=1}^{n} \left[f_i(\mathbf{x}_i, \xi_i) + \sum_{j \in \mathcal{N}_i} \left(\lambda_{ij} g_{ij}(\mathbf{x}_i, \mathbf{x}_j) - \frac{\delta\eta}{2} \lambda_{ij}^2 \right) \right] \tag{4.3}$$

On the L.H.S. of (4.3), $\mathbf{x}$ denotes the concatenation of all the model parameters $\{\mathbf{x}_i\}_{i=1}^n$, each of which is in $\mathbb{R}^d$, leading to $\mathbf{x} \in \mathbb{R}^{nd}$. For $i \in [n]$ and $j \in \mathcal{N}_i$, $\lambda_{ij} \geq 0$ denotes the Lagrangian multiplier associated with the constraint $g_{ij}(\mathbf{x}_i, \mathbf{x}_j) \leq 0$. Similarly, $\boldsymbol{\lambda}$ on the L.H.S. denotes the concatenation of all λ_{ij} for $i \in [n]$ and $j \in \mathcal{N}_i$, thus $\boldsymbol{\lambda} \in \mathbb{R}^m$, where m is twice the number of edges in the underlying undirected graph. The last term on the R.H.S. of (4.3) corresponds to a regularizer which mitigates the growth of the Lagrangian multiplier $\boldsymbol{\lambda}$ during the saddle-point algorithm updates. In this term, $\eta > 0$ corresponds to the learning rate of the algorithm and $\delta > 0$ is a control parameter.

To find the saddle point of the Lagrangian in (4.3), we utilize alternating gradient updates of the primal variables concatenated in $\mathbf{x}$, and the dual variables in $\boldsymbol{\lambda}$. For any $i \in [n]$, the gradient of the modified Lagrangian with respect to (w.r.t.) the model parameter $\mathbf{x}_i$ is given by:

$$\nabla_{\mathbf{x}_i} \mathcal{L}(\mathbf{x}, \boldsymbol{\lambda}) = \sum_{j \in \mathcal{N}_i} [\lambda_{ij} \nabla_{\mathbf{x}_i} g_{ij}(\mathbf{x}_i, \mathbf{x}_j) + \lambda_{ji} \nabla_{\mathbf{x}_i} g_{ij}(\mathbf{x}_j, \mathbf{x}_i)] + \nabla_{\mathbf{x}_i} f_i(\mathbf{x}_i, \xi_i) \tag{4.4}$$

The gradient w.r.t. the Lagrangian multiplier λ_{ij} for $i \in [n]$, $j \in \mathcal{N}_i$ is similarly given by:

$$\frac{\partial}{\partial \lambda_{ij}} \mathcal{L}(\mathbf{x}, \boldsymbol{\lambda}) = g_{ij}(\mathbf{x}_i, \mathbf{x}_j) - \delta \eta \lambda_{ij} \tag{4.5}$$

The stochastic algorithm developed for updating the primal and dual variables via equations (4.4) and (4.5) is presented in Algorithm 2, which is described below.

Our proposed scheme in Algorithm 2 is a stochastic saddle-point algorithm to minimize the objective in (4.1) by finding a saddle point of the modified Lagrangian

Algorithm 2 Compressed Decentralized Optimization with Sample Feedback

Initialize: Random raw parameters $\widetilde{\mathbf{x}}_i^{(1)} \in \mathcal{X}$, $\lambda_{ij}^{(1)} = 0$ for each $i \in [n]$, $j \in \mathcal{N}_i$, $\hat{\mathbf{x}}_i^{(0)} = \mathbf{0}$ for each $i \in [n]$, number of iterations T, learning rate η, parameter $\delta > 0$. (Communicate in the first iteration without compression to ensure that $\widetilde{\mathbf{x}}^{(1)} = \hat{\mathbf{x}}^{(1)}$)

```
1: for $t = 1$ to $T$ in parallel for $i \in [n]$ do
2:     Compute $\mathbf{q}_i^{(t)} = \mathcal{C}(\widetilde{\mathbf{x}}_i^{(t)} - \hat{\mathbf{x}}_i^{(t-1)})$
3:     for nodes $k \in \mathcal{N}_i \cup \{i\}$ do
4:         Send $\mathbf{q}_i^{(t)}$ and receive $\mathbf{q}_k^{(t)}$
5:         Update $\hat{\mathbf{x}}_k^{(t)} = \hat{\mathbf{x}}_k^{(t-1)} + \mathbf{q}_k^{(t)}$
6:         Compute $\mathbf{x}_k^{(t)} = \Pi_{\mathcal{X}}(\hat{\mathbf{x}}_k^{(t)})$
7:     end for
8:     Update running average for local parameter:
       $\overline{\mathbf{x}}_{i,avg}^{(t)} = \frac{1}{t}\mathbf{x}_i^{(t)} + \frac{t-1}{t}\overline{\mathbf{x}}_{i,avg}^{(t-1)}$
9:     Sample $\boldsymbol{\xi}_i^{(t)} \sim \mathcal{P}_i$ and compute $\nabla_{\mathbf{x}_i} f_i(\mathbf{x}_i^{(t)}, \boldsymbol{\xi}_i^{(t)})$
10:    For all $j \in \mathcal{N}_i$ compute $\nabla_{\mathbf{x}_i} g_{ij}(\mathbf{x}_i^{(t)}, \mathbf{x}_j^{(t)})$
11:    Update the primal variable by gradient descent:
       $$\widetilde{\mathbf{x}}_i^{(t+1)} = \Pi_{\mathcal{X}}\left(\widetilde{\mathbf{x}}_i^{(t)} - \eta\nabla_{\mathbf{x}_i} f_i(\mathbf{x}_i^{(t)}, \boldsymbol{\xi}_i^{(t)}) - 2\eta \sum_{j\in\mathcal{N}_i} \lambda_{ij}^{(t)} \nabla_{\mathbf{x}_i} g_{ij}(\mathbf{x}_i^{(t)}, \mathbf{x}_j^{(t)})\right)$$
12:    For $j \in \mathcal{N}_i$, update the dual variables through gradient ascent:
       $$\lambda_{ij}^{(t+1)} = \left[\lambda_{ij}^{(t)} + \eta\left(g_{ij}(\mathbf{x}_i^{(t)}, \mathbf{x}_j^{(t)}) - \delta\eta\lambda_{ij}^{(t)}\right)\right]^+$$
13: end for
```

Output: Time averaged parameters $\overline{\mathbf{x}}_{i,avg}^{(T)}$ for all $i \in [n]$.

in (4.3) in a communication efficient manner. Each node is allowed to exchange with its neighboring nodes only compressed parameters, via the compression operator in (4.2). To realize exchange of compressed parameters between workers, for node $i \in [n]$ and its associated *raw* parameter $\widetilde{\mathbf{x}}_i$, all nodes $j \in \mathcal{N}_i$ maintain an estimate $\hat{\mathbf{x}}_i$ of $\widetilde{\mathbf{x}}_i$, so, each node $i \in [n]$ has access to $\hat{\mathbf{x}}_j$ for all $j \in \mathcal{N}_i$. The parameter $\widetilde{\mathbf{x}}_i$ is called raw as it corresponds to the model parameter before any compression in our algorithm. We refer to $\hat{\mathbf{x}}_i$ as the *copy* parameter of the node i.

We first initialize the regularization parameter δ (see Theorem 3 for definition) and learning rate η. We initialize the parameter copies of all the nodes as $\hat{\mathbf{x}}_i = \mathbf{0}$ for all $i \in [n]$ and allow each node to communicate with its neighbors in the first round without any compression. This is to ensure that $\widetilde{\mathbf{x}}_i^{(1)} = \hat{\mathbf{x}}_i^{(1)}$ for all the nodes (this is a requirement to control the error encountered via compression, c.f. Lemma 4). At any time step $t \in [T]$ of the algorithm, node i first computes the compressed update to its copy parameter, given by $\mathbf{q}_i^{(t)}$ (line 2) and then sends and receives these copy parameter updates from its neighbor nodes in $\mathcal{N}_i$ (line 3). Importantly, these copy parameter updates are compressed using the operator in (4.2), and thus the communication is efficient. After receiving the copy updates from its neighbors, each node updates the locally available copy parameters of its neighbors and its own copy parameters (line 5) and ensures that these lie in the set $\mathcal{X}$ by taking a projection[1] to form the local node parameter $\mathbf{x}_i^{(t)}$ (line 6). As the node i has access to the updated copy parameters of its neighbors, it also has access to $\mathbf{x}_j^{(t)}$ for all $j \in \mathcal{N}_i$. With the local node parameter evaluated, the node can update its running average of parameters (line 8).

[1]It can be checked that the computational complexities for projection of all the primal node parameters and the dual parameters are $\mathcal{O}(nd)$ and $\mathcal{O}(m)$, respectively, per iteration.

For the stochastic saddle-point update with sample feedback, at time t , the node $i \in [n]$ can sample a data point $\boldsymbol{\xi}_i^{(t)}$ and evaluate the gradient using the previously computed node parameter $\mathbf{x}_i^{(t)}$ (line 9). Since the node also has access to the parameters $\mathbf{x}_j^{(t)}$ for neighbors $j \in \mathcal{N}_i$, it can compute the gradient w.r.t. the pairwise constraint function g_{ij} evaluated at $\mathbf{x}_i^{(t)}, \mathbf{x}_j^{(t)}$ (line 10). Thus, the node can evaluate the gradient of the modified Lagrangian w.r.t. the primal local node parameters as in (4.4) and take a gradient descent step to update the raw node parameter $\widetilde{\mathbf{x}}_i^{(t)}$. Similarly, the dual variables $\lambda_{ij}^{(t)}$ are also updated via a gradient ascent step (line 12) following (4.5) and then projected on the positive real space.

Symmetry of dual updates: Note that the derived expression for the gradient $\nabla_{\mathbf{x}_i}\mathcal{L}(\mathbf{x}, \boldsymbol{\lambda})$, consists of the dual parameters λ_{ij} and λ_{ji}. Meanwhile, the update in line 11 of Algorithm 2 considers these parameters to be the same for all time $t \in [T]$. We describe the reasoning behind this update in the following induction argument. The dual variables are initialized to 0, that is, $\lambda_{ij}^{(1)} = 0$ for all $i \in [n]$ and $j \in \mathcal{N}_i$. Thus for any connected nodes i, j, for $t = 1$, the condition $\lambda_{ij}^{(t)} = \lambda_{ji}^{(t)}$ holds. Next, we assume that for any arbitrary $\tau \in [T]$, $\tau \neq 1$, it is the case that $\lambda_{ij}^{(\tau)} = \lambda_{ji}^{(\tau)}$. Thus for the time step $t = \tau + 1$, by the update given in line 12 of Algorithm 2, we have:

$$\begin{aligned}
\lambda_{ij}^{(\tau+1)} &= \left[\lambda_{ij}^{(\tau)} + \eta\left(g_{ij}(\mathbf{x}_i^{(\tau)}, \mathbf{x}_j^{(\tau)}) - \delta\eta\lambda_{ij}^{(\tau)}\right)\right]^+ \\
&\overset{(a)}{=} \left[\lambda_{ji}^{(\tau)} + \eta\left(g_{ji}(\mathbf{x}_j^{(\tau)}, \mathbf{x}_i^{(\tau)}) - \delta\eta\lambda_{ji}^{(\tau)}\right)\right]^+ \\
&= \lambda_{ji}^{(\tau+1)}
\end{aligned}$$

where (a) follows from the fact that $\lambda_{ij}^{(\tau)} = \lambda_{ji}^{(\tau)}$ and the symmetry of the pairwise constraints g_{ij} for connected nodes i, j. Thus, as the induction step holds for arbitrary

$\tau \in [T]$ and for the base case $t = 1$, it follows that $\lambda_{ij}^{(t)} = \lambda_{ji}^{(t)}$ for all $t \in [T]$ for all $i \in [n], j \in \mathcal{N}_i$.

Justification for raw parameter updates: Note that in the steps given in lines (9-11) in Algorithm 2, the gradients are evaluated at the node parameters $\{\mathbf{x}_i^{(t)}\}_{i=1}^n$, while the updates are made to the raw parameters $\{\widetilde{\mathbf{x}}_i^{(t)}\}_{i=1}^n$ via gradient descent. The reason for this is that in our scheme, the raw parameters effectively play the role of a *virtual* parameter, which mimic SGD-like updates (c.f. line 11), with the gradients evaluated at a different (perturbed) parameter. The notion of such virtual parameters to analyze convergence has been promising lately in stochastic optimization within the perturbed iterate analysis framework, see [MPP$^+$17, SDGD21a, SCJ18a, BDKD19a]. The key idea to analyze convergence in such settings is to control the difference of the iterates $\left\|\mathbf{x}_i^{(t)} - \widetilde{\mathbf{x}}_i^{(t)}\right\|_2$ for all $i \in [n]$. Controlling this difference is one key contribution of our work, c.f. Lemma 4.

4.3.2 Main Result: Sample Feedback

We now present our theoretical result on the convergence rate of Algorithm 2 for decentralized optimization for the case with sample feedback. We first present and discuss the set of assumptions our result is based on.

Assumption 4. *The set of admissible model parameters $\mathcal{X}$, is closed, convex and bounded, i.e., there exists a constant $R > 0$ such that $\|\widetilde{\mathbf{x}}\|_2 \leq \frac{R}{\sqrt{n}}$, for all $\widetilde{\mathbf{x}} \in \mathcal{X}$.*

Assumption 5. *For any $i \in [n]$, the local objective $f_i(\mathbf{x}_i, \boldsymbol{\xi}_i)$ is convex in $\mathbf{x}_i$ for any $\boldsymbol{\xi}_i \in \Xi_i$. The pairwise constraint function $g_{ij}(\mathbf{x}_i, \mathbf{x}_j)$ is (jointly) convex in $\mathbf{x}_i$ and $\mathbf{x}_j$, for any pair $i \in [n], j \in \mathcal{N}_i$.*

Assumption 6. *For $i \in [n]$ and $\mathbf{x}_i \in \mathcal{X}$, $\exists G_i > 0$ such that:*

$$\mathbb{E}_{\boldsymbol{\xi}_i \sim \mathcal{P}_i} \left[\|\nabla_{\mathbf{x}_i} f_i(\mathbf{x}_i, \boldsymbol{\xi}_i)\|_2^2 \right] \leq G_i^2. \tag{4.6}$$

To simplify the notation, we also define $G := \sqrt{\sum_{i=1}^n G_i^2}$. Additionally, for any $i \in [n], j \in \mathcal{N}_i$, we assume that there exists a constant $G_{ij} > 0$ such that $\forall \mathbf{x}_i, \mathbf{x}_j \in \mathcal{X}$:

$$\left\| \left[\nabla_{\mathbf{x}_i} g_{ij}(\mathbf{x}_i, \mathbf{x}_j)^T, \nabla_{\mathbf{x}_j} g_{ij}(\mathbf{x}_i, \mathbf{x}_j)^T \right]^T \right\|_2 \leq G_{ij}. \tag{4.7}$$

We define $\tilde{G} := \max_{i \in [n], j \in \mathcal{N}_i} G_{ij}$.

Assumption 7. *For any $i \in [n], j \in \mathcal{N}_i$, the pairwise constraint function g_{ij} is bounded. That is, there exists a constant $C_{ij} > 0$ such that $|g_{ij}(\mathbf{x}_i, \mathbf{x}_j)| \leq C_{ij}$, $\forall \mathbf{x}_i, \mathbf{x}_j \in \mathcal{X}$. We define $C^2 := \sqrt{\sum_{i=1}^n \sum_{j \in \mathcal{N}_i} C_{ij}^2}$.*

Assumptions A.4-A.7 are frequently used in convergence rate analysis of convex optimization algorithms, even without compression. The assumption on a bounded parameter space $\mathcal{X}$ and the bounded constraint functions have been made earlier in [CB20, CLG17]. The assumption on boundedness of the gradient of the objectives (Equation (4.6)) has also been made earlier in [CB20, NO09a, CLG17] and boundedness of gradients of the constraint functions (Equation (4.7)) have been assumed in [CB20, YN17, NN13][2].

With these assumptions in place, we now present our main theoretical result

[2]Assumption A.6 for compressed decentralized optimization has been relaxed in one of our previous works [SDGD21a]. The arguments for relaxing this assumption can similarly be extended to the analysis in this chpater, a technicality which we omit in interest of keeping the analysis relatively cleaner, and to focus on the main novelty of analyzing compressed communication in the pairwise multi-task setting.

in Theorem 3 below for the convergence rate of Algorithm 2. The result is stated in terms of the stacked vector $\mathbf{x}$, which corresponds to the concatenation of the parameters $\{\mathbf{x}_i\}_{i=1}^n$, and thus is $n \times d$ dimensional. The vector $\mathbf{x}^*$ represents the stacked optimal parameters which is the solution of the optimization problem (4.1). The proof details for Theorem 3 are presented in Section Appendix B.

Theorem 3. *Consider running Algorithm 2 for T iterations with fixed step size $\eta = \frac{a}{\sqrt{T}}$ for positive constant a and regularization parameter $\delta = \frac{1-\sqrt{1-\frac{64\eta^2(1+m)\tilde{G}^2}{\omega^2}}}{4\eta^2}$ where $\omega \in (0,1)$ is the compression factor. Then, under assumptions A.4 - A.7, for $T \geq \frac{64a^2(1+m)\tilde{G}^2}{\omega^2}$, the expected value of F evaluated at the stacked time-averaged vector $\overline{\mathbf{x}}_{avg}^{(T)} := \frac{1}{T}\sum_{t=1}^{T}\mathbf{x}^{(t)}$ satisfies:*

$$\mathbb{E}[F(\overline{\mathbf{x}}_{avg}^{(T)})] - F(\mathbf{x}^*) \leq \frac{2R^2}{a\sqrt{T}} + \frac{a}{\sqrt{T}}\left(\frac{4}{\omega^2}(1+m)G^2 + 2C^2\right) \tag{4.8}$$

For $i \in [n]$, $j \in \mathcal{N}_i$, the constraint function g_{ij} satisfies:

$$\mathbb{E}\left[g_{ij}(\overline{\mathbf{x}}_{i,avg}^{(T)}, \overline{\mathbf{x}}_{j,avg}^{(T)})\right] \leq \frac{1}{T^{\frac{1}{4}}}\left(\sqrt{\frac{8GR}{a}} + \sqrt{8\delta aGR}\right) + \frac{1}{\sqrt{T}}\sqrt{2\left(2R^2\delta + \frac{4}{\omega^2}(1+m)G^2 + C^2\right)}$$
$$+ \frac{1}{\sqrt{T}}\sqrt{2\delta a^2\left(\frac{4}{\omega^2}(1+m)G^2 + C^2\right)} + \frac{2R}{a\sqrt{T}} \tag{4.9}$$

where the d-dimensional vector $\overline{\mathbf{x}}_{k,avg}^{(T)}$ denotes the time averaged parameter for node $k \in [n]$ in $\overline{\mathbf{x}}_{avg}^{(T)}$.

Theorem 3 establishes that for any given compression requirement $\omega \in (0,1)$, the sub-optimality of the objective, $\mathbb{E}[F(\overline{\mathbf{x}}^{(T)})] - F(\mathbf{x}^*)$, is $\mathcal{O}\left(\frac{1}{T^{1/2}}\right)$, and the expected constraint violation $\mathbb{E}[g_{ij}(\overline{\mathbf{x}}_i^{(T)}, \overline{\mathbf{x}}_j^{(T)})]$ for any connected node pair (i,j) is $\mathcal{O}\left(\frac{1}{T^{1/4}}\right)$. Thus, the difference between the attained objective and the global minimum of (4.1), as well as the constraint violations can be made arbitrarily small by increasing the number of iterations the algorithm is run for.

4.4 Decentralized compressed optimization with Bandit feedback

In this section, we focus on the *bandit feedback* scenario where the nodes do not have direct access to samples drawn from their local data distributions. This could, for example, arise in situations where the samples are high dimensional and thus can be hard to observe or measure. For the model we work with in this chapter, we now assume that the nodes instead can query the value of the local objective function $f_i(\mathbf{x}_i, \xi_i)$ for some particular choices of the parameter $\mathbf{x}_i$. We first formally define the objective query process for the nodes and then describe how this model can be used to develop a stochastic gradient method for optimizing the overall objective (4.1).

Let $\mathbb{S} := \{\mathbf{u} \in \mathbb{R}^d | \left\|\mathbf{u}\right\|_2 = 1\}$ and $\mathbb{B} := \{\mathbf{u} \in \mathbb{R}^d | \left\|\mathbf{u}\right\|_2 \leq 1\}$ be the unit sphere, ball in d-dimensions, respectively. For each node $i \in [n]$, and at any stage in the optimization process, we assume access to two local objective values $f_i(\mathbf{x}_i \pm \zeta \mathbf{u}_i, \xi_i)$ where $\mathbf{u}_i$ is sampled uniformly at random over the unit sphere $\mathbb{S}$ (independent of $\mathbf{x}_i$ or ξ_i), ζ is a small positive constant, and $\mathbf{x}_i$ is the local model parameter. To evaluate the gradient using these objective values, we make use of the following fact from [FKM05]:

Fact 1. *Consider a function $\phi : \mathbb{R}^d \to \mathbb{R}$, and let $\zeta > 0$. Define $\tilde{\phi}(\mathbf{x}) = \mathbb{E}_{\mathbf{u} \sim \mathcal{U}(\mathbb{B})}[\phi(\mathbf{x} + \zeta \mathbf{u})]$ where $\mathcal{U}(\mathbb{B})$ denotes uniform distribution over the unit ball $\mathbb{B} \subset \mathbb{R}^d$. Then:*

1. *If ϕ is convex, then $\tilde{\phi}$ is also convex.*

2. *For any* $\mathbf{x} \in \mathbb{R}^d$*,* $\nabla_{\mathbf{x}} \tilde{\phi}(\mathbf{x}) = \frac{d}{\zeta}\mathbb{E}_{\mathbf{u}\sim\mathcal{U}(\mathbb{S})}[\phi(\mathbf{x}+\zeta\mathbf{u})\mathbf{u}]$ *where* $\mathcal{U}(\mathbb{S})$ *denotes the uniform distribution over the unit sphere* $\mathbb{S} \subset \mathbb{R}^d$*.*

For the node $i \in [n]$, the above fact can be used to estimate the gradient of the local objective function using the values $f_i(\mathbf{x}_i \pm \zeta\mathbf{u}_i, \xi_i)$ where $\mathbf{u}_i \sim \mathcal{U}(\mathbb{S})$. For a given ξ_i, we define $\tilde{f}_i(\mathbf{x}_i, \xi_i) = \mathbb{E}_{\mathbf{v}_i\sim\mathcal{U}(\mathbb{B})}[f_i(\mathbf{x}_i + \zeta\mathbf{v}_i, \xi_i)]$. From the above fact, $\tilde{f}(\mathbf{x}_i, \xi_i)$ is convex in $\mathbf{x}_i$ for a given ξ_i.

Note that as stated, the parameter vector $\mathbf{x}_i \pm \zeta\mathbf{u}_i$ may not lie in the feasible set $\mathcal{X}$ for all range of values of ζ. Thus, we need some restriction on the range of values ζ can take. In the following, we make this argument precise. We first introduce an additional mild assumption on the topology of the set $\mathcal{X}$:

Assumption 8. *The set* $\mathcal{X}$ *has a non-empty interior, that is,* $\exists \mathbf{y}_0 \in \mathcal{X}$*,* $r > 0$*, s.t.* $\mathcal{B}(\mathbf{y}_0, r) \subset \mathcal{X}$*. Here,* $\mathcal{B}(\mathbf{y}_0, r)$ *denotes the open ball of radius* r *centered at* $\mathbf{y}_0$*, i.e.,* $\mathcal{B}(\mathbf{y}_0, r) = \{\mathbf{x} | \, \|\mathbf{x} - \mathbf{y}_0\|_2 \leq r\}$*.*

From the above assumption, by the convexity of $\mathcal{X}$, it can also be concluded that for any $\alpha \in (0, 1)$ and $\mathbf{x} \in \mathcal{X}$, we have $\mathcal{B}((1-\alpha)\mathbf{x} + \alpha\mathbf{y}_0, \alpha r) \subset \mathcal{X}$. We further define the set $\widetilde{\mathcal{X}} = \{(1 - \frac{\zeta}{r})\mathbf{x} + \frac{\zeta}{r}\mathbf{y}_0 | \mathbf{x} \in \mathcal{X}\}$. It can now be readily checked that if $\mathbf{x}_i \in \widetilde{\mathcal{X}}$ for the node i, then $\mathbf{x}_i \pm \zeta\mathbf{u}_i \in \mathcal{X}$, where $\mathbf{u}_i$ is any point on the unit sphere $\mathbb{S}$. Thus in the development of the algorithm below, we project the parameters onto the space $\widetilde{\mathcal{X}}$ to ensure that during the bandit feedback, the evaluated parameter $\mathbf{x}_i \pm \zeta\mathbf{u}_i$ for any node i lies in the space $\mathcal{X}$.

4.4.1 Algorithm: Bandit Feedback

We develop an algorithm for the bandit feedback scenario to find a saddle-point of the modified Lagrangian:

$$\tilde{\mathcal{L}}(\mathbf{x}, \boldsymbol{\lambda}) = \sum_{i=1}^{n} \left[\tilde{f}_i(\mathbf{x}_i, \xi_i) + \sum_{j \in \mathcal{N}_i} \left(\lambda_{ij} g_{ij}(\mathbf{x}_i, \mathbf{x}_j) - \frac{\delta \eta}{2} \lambda_{ij}^2 \right) \right] \tag{4.10}$$

The vector $\mathbf{x} \in \widetilde{\mathcal{X}}^n$ represents the stacked node parameters and $\boldsymbol{\lambda}$ represents the stacked dual variables. Here, the main difference from the modified Lagrangian in sample feedback case presented in (4.3) is that the objectives $\{f_i\}_{i=1}^n$ of the nodes are now replaced by the functions $\{\tilde{f}_i\}_{i=1}^n$. Importantly, the gradient of these functions can be computed via the result of Fact 1 which enables us to develop a primal-dual gradient algorithm to find the saddle point of (4.10). The gradient w.r.t. the primal variable $\mathbf{x}$ is given by:

$$\nabla_{\mathbf{x}_i} \tilde{\mathcal{L}}(\mathbf{x}, \boldsymbol{\lambda}) = \sum_{j \in \mathcal{N}_i} \left[\lambda_{ij} \nabla_{\mathbf{x}_i} g_{ij}(\mathbf{x}_i, \mathbf{x}_j) + \lambda_{ji} \nabla_{\mathbf{x}_i} g_{ij}(\mathbf{x}_j, \mathbf{x}_i) \right] + \nabla_{\mathbf{x}_i} \tilde{f}_i(\mathbf{x}_i, \xi_i) \tag{4.11}$$

Using the result from Fact 1, for any $i \in [n]$ we have:

$$\nabla_{\mathbf{x}} \tilde{f}_i(\mathbf{x}_i, \xi_i) = \frac{d}{2\zeta} \mathbb{E}_{\mathbf{u}_i \sim \mathcal{U}(\mathbb{S})} [f(\mathbf{x}_i + \zeta \mathbf{u}_i, \xi_i) - f(\mathbf{x}_i - \zeta \mathbf{u}_i, \xi_i)] \mathbf{u}_i$$

As the node has access to the values of the local objective function in the bandit feedback scenario, the quantity $\frac{d}{2\zeta}[f(\mathbf{x}_i + \zeta \mathbf{u}_i, \xi_i) - f(\mathbf{x}_i - \zeta \mathbf{u}_i, \xi_i)]$ for a given $\mathbf{u}_i \sim \mathcal{U}(\mathbb{S})$, $\mathbf{x}_i, \xi_i$, serves as an unbiased estimate of $\nabla_{\mathbf{x}} \tilde{f}_i(\mathbf{x}_i, \xi_i)$. We note that such an approximation for the gradient is common in the stochastic optimization literature, e.g. [BPP12, PBFM16]. In contrast to the uniform perturbation we consider in

Fact 1, one can possibly use perturbations arising from distributions such as Gaussian, symmetric Bernoulli distributions as in [Spa92,NND+18]. Using this, we can construct the following estimate for the primal gradient $\nabla_{\mathbf{x}_i}\tilde{\mathcal{L}}(\mathbf{x},\boldsymbol{\lambda})$:

$$\mathbf{p}_i^{(t)} := \frac{d}{2\xi}\left[f_i(\mathbf{x}_i^{(t)}+\zeta\mathbf{u}_i^{(t)},\boldsymbol{\xi}_i^{(t)})-f_i(\mathbf{x}_i^{(t)}-\zeta\mathbf{u}_i^{(t)},\boldsymbol{\xi}_i^{(t)})\right]\mathbf{u}_i^{(t)} + 2\sum_{j\in\mathcal{N}_i}\lambda_{ij}^{(t)}\nabla_{\mathbf{x}_i}g_{ij}(\mathbf{x}_i^{(t)},\mathbf{x}_j^{(t)}) \tag{4.12}$$

The gradient of the Lagrangian in (4.10) w.r.t. the dual parameter λ_{ij} for $i \in [n]$ and $j \in \mathcal{N}_i$ is the same as in the sample feedback scenario and is given in (4.4).

The development of Algorithm 3 is similar to that of Algorithm 2. The main difference is that we now find the saddle point of (4.10) via alternating primal and dual variable gradient updates given in equations (4.12) and (4.5) and project onto the space $\widetilde{\mathcal{X}}$ to ensure that the perturbed parameters lie in $\mathcal{X}$. As before, for a node $i \in [n]$, $\widetilde{\mathbf{x}}_i$ refers to its raw parameter, $\mathbf{x}_i$ as its local parameter, and $\hat{\mathbf{x}}_i$ is the copy parameter.

We initialize the raw parameters $\{\widetilde{\mathbf{x}}_i^{(1)}\}_{i=1}^n$ inside the set $\widetilde{\mathcal{X}}$. During the first round, we assume the communication without compression to ensure that $\widetilde{\mathbf{x}}_i^{(1)} = \hat{\mathbf{x}}_i^{(1)}$ for all $i \in [n]$. At time step $t \in [T]$, the node $i \in [n]$ computes and exchanges its copy parameters and constructs the local node parameter $\mathbf{x}_i^{(t)}$ for which we track the running average (lines 2-8). As samples from the underlying distribution $\mathcal{P}_i$ are not directly revealed to the node in case of bandit feedback; instead it queries the value of the local objective $f_i(.,\xi_i)$ at parameters $\mathbf{x}_i^{(t)} + \zeta\mathbf{u}_i^{(t)}$ and $\mathbf{x}_i^{(t)} - \zeta\mathbf{u}_i^{(t)}$ where $\mathbf{u}_i^{(t)}$ is uniformly sampled over the d-dimensional unit sphere $\mathbb{S}$ (lines 9-10). These values are then used to construct an unbiased estimate of $\nabla_{\mathbf{x}_i}\tilde{\mathcal{L}}(\mathbf{x},\boldsymbol{\lambda})$ using (4.12), and then to update the raw parameter $\widetilde{\mathbf{x}}_i^{(t)}$ along with a projection operation back to the set $\widetilde{\mathcal{X}}$

Algorithm 3 Compressed Decentralized Optimization with Bandit Feedback

Initialize: Random $\widetilde{\mathbf{x}}_i^{(1)} \in \widetilde{\mathcal{X}}$ individually for each $i \in [n]$ and $\lambda_{ij}^{(1)} = 0$ for each $j \in \mathcal{N}_i$. $\hat{\mathbf{x}}_i^{(0)} = \mathbf{0}$ for each $i \in [n]$, number of iterations T, learning rate η, parameters $\zeta, \delta > 0$. (Communicate in the first iteration without compression to ensure that $\widetilde{\mathbf{x}}^{(1)} = \hat{\mathbf{x}}^{(1)}$.)

```
1:  for $t = 1$ to $T$ in parallel for $i \in [n]$ do
2:      Compute $\mathbf{q}_i^{(t)} = \mathcal{C}(\widetilde{\mathbf{x}}_i^{(t)} - \hat{\mathbf{x}}_i^{(t-1)})$
3:      for nodes $k \in \mathcal{N}_i \cup \{i\}$ do
4:          Send $\mathbf{q}_i^{(t)}$ and receive $\mathbf{q}_k^{(t)}$
5:          Update $\hat{\mathbf{x}}_k^{(t)} = \hat{\mathbf{x}}_k^{(t-1)} + \mathbf{q}_k^{(t)}$
6:          Compute $\mathbf{x}_k^{(t)} = \Pi_{\widetilde{\mathcal{X}}}(\hat{\mathbf{x}}_k^{(t)})$
7:      end for
8:      Update running average for local parameter:
        $\overline{\mathbf{x}}_{i,avg}^{(t)} = \frac{1}{t}\mathbf{x}_i^{(t)} + \frac{t-1}{t}\overline{\mathbf{x}}_{i,avg}^{(t-1)}$
9:      Sample $\mathbf{u}_i^{(t)} \sim \mathcal{U}(\mathbb{S})$
10:     Query the two values: $f_i(\mathbf{x}_i^{(t)} \pm \zeta\mathbf{u}_i^{(t)}, \boldsymbol{\xi}_i^{(t)})$
11:     Compute the Lagrangian primal gradient estimate:
        $$\mathbf{p}_i^{(t)} := 2\sum_{j\in\mathcal{N}_i} \lambda_{ij}^{(t)} \nabla_{\mathbf{x}_i} g_{ij}(\mathbf{x}_i^{(t)}, \mathbf{x}_j^{(t)}) + \frac{d}{2\xi}\left[f_i(\mathbf{x}_i^{(t)}+\zeta\mathbf{u}_i^{(t)}, \boldsymbol{\xi}_i^{(t)}) - f_i(\mathbf{x}_i^{(t)}-\zeta\mathbf{u}_i^{(t)}, \boldsymbol{\xi}_i^{(t)})\right]\mathbf{u}_i^{(t)}$$
12:     Update the primal variable via gradient descent:
        $$\widetilde{\mathbf{x}}_i^{(t+1)} = \Pi_{\widetilde{\mathcal{X}}}\left(\widetilde{\mathbf{x}}_i^{(t)} - \eta\mathbf{p}_i^{(t)}\right)$$
13:     For all $j \in \mathcal{N}_i$, update the dual variables via gradient ascent:
        $$\lambda_{ij}^{(t+1)} = \left[\lambda_{ij}^{(t)} + \eta\left(g_{ij}(\mathbf{x}_i^{(t)}, \mathbf{x}_j^{(t)}) - \delta\eta\lambda_{ij}^{(t)}\right)\right]^+$$
14: end for
```

Output: Time averaged parameters $\overline{\mathbf{x}}_{i,avg}^{(T)}$ for all $i \in [n]$.

(lines 11-13). Finally, the dual variables are also updated via gradient descent along with the projection to the positive real space to ensure feasibility (line 13). As in the case of sample feedback, the update of the dual steps in line 13 and the initialization $\lambda_{ij}^{(1)} = 0$ ensures that $\lambda_{ij}^{(t)} = \lambda_{ji}^{(t)}$ for all $t \in [T]$, and for all $i \in [n], j \in \mathcal{N}_i$.

4.4.2 Main Result: Bandit Feedback

We now present the convergence result rate for Algorithm 3 which optimizes (4.1) in the bandit feedback scenario. The proof details are provided in Appendix B.

Theorem 4. *Consider running Algorithm 3 for T iterations with fixed step size $\eta = \frac{a}{\sqrt{T}}$ for positive constant a, with perturbation constant $\zeta = \frac{1}{T}$, and regularization parameter $\delta = \frac{1-\sqrt{1-\frac{256\eta^2(1+m)\tilde{G}^2}{\omega^2}}}{4\eta^2}$, where $\omega \in (0,1)$ is the compression factor. Under Assumptions A.4-A.8, for $T \geq \frac{256a^2(1+m)\tilde{G}^2}{\omega^2}$, the expected value of F evaluated at the time averaged vector $\overline{\mathbf{x}}_{avg}^{(T)} := \frac{1}{T}\sum_{t=1}^{T}\mathbf{x}^{(t)}$ satisfies:*

$$\mathbb{E}[F(\overline{\mathbf{x}}_{avg}^{(T)})]-F(\mathbf{x}^*) \leq \frac{2R^2}{a\sqrt{T}}+\frac{a}{\sqrt{T}}\left[\frac{16}{\omega^2}d^2(1+m)G^2+C^2\right]+\frac{2\sqrt{m}\tilde{G}RC}{\delta ar\sqrt{T}}+\frac{4RG}{rT}+\frac{4\sqrt{n}G}{T} \tag{4.13}$$

where $r \leq \frac{R}{\sqrt{n}}$. For any $i \in [n]$, $j \in \mathcal{N}_i$, we have:

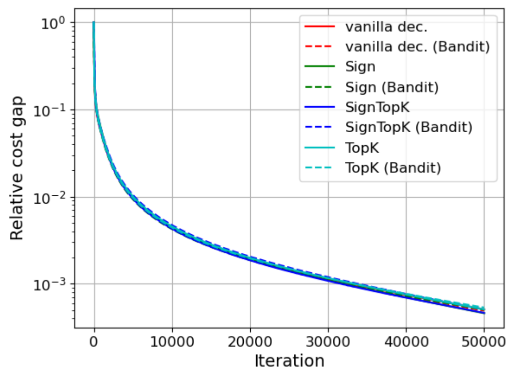
$$\begin{aligned}\mathbb{E}\left[g_{ij}(\overline{\mathbf{x}}_{i,avg}^{(T)}, \overline{\mathbf{x}}_{j,avg}^{(T)})\right] &\leq \frac{1}{T^{1/4}}\left[\sqrt{\frac{8GR}{a}}+\sqrt{\frac{8\delta a(R+r\sqrt{n})G}{r}+8GR\delta a}\right] \\ &+\frac{1}{\sqrt{T}}\sqrt{\left(\frac{32}{\omega^2}d^2(1+m)G^2+2C^2\right)+\frac{4\sqrt{m}\tilde{G}RC}{r\delta a^2}+4R^2\delta} \\ &+\frac{1}{\sqrt{T}}\sqrt{\delta a^2\left(\frac{32}{\omega^2}d^2(1+m)G^2+2C^2\right)+\frac{4\sqrt{m}\tilde{G}RC}{r}}+\frac{1}{T^{3/4}}\sqrt{\frac{8(R+r\sqrt{n})G}{ra}}\end{aligned} \tag{4.14}$$

where $\overline{\mathbf{x}}_{k,avg}^{(T)}$ *is time averaged parameter of node* k*.*

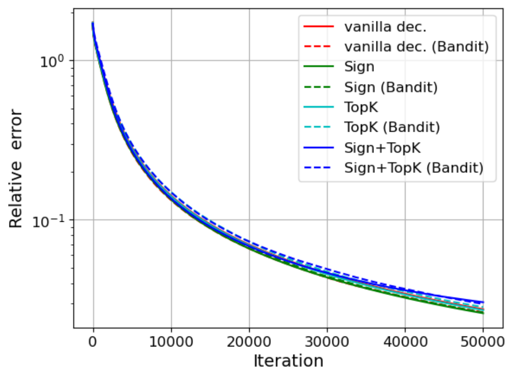
The above result establishes that for a given compression requirement $\omega \in (0,1)$, the sub-optimality of the objective $\mathbb{E}[F(\overline{\mathbf{x}}_{avg}^{(T)})] - F(\mathbf{x}^*)$ is $\mathcal{O}\left(\frac{1}{T^{1/2}}\right)$. Similarly, the expected constraint violation for $i \in [n]$ and $j \in \mathcal{N}_i$ given by $\mathbb{E}\left[g_{ij}(\overline{\mathbf{x}}_{i,avg}^{(T)}, \overline{\mathbf{x}}_{j,avg}^{(T)})\right]$ is $\mathcal{O}\left(\frac{1}{T^{1/4}}\right)$. Thus, in effect by choosing a large enough value of T, the number of iterations Algorithm 3 is run for, the obtained stacked parameter $\overline{\mathbf{x}}_{avg}^{(T)}$ is a good estimate of the optimal solution of the overall objective (4.1). Moreover, the result obtained matches the rate that was obtained for the sample feedback case in Theorem 3, where the nodes had access to the samples at every stage. Theorem 4 thus establishes that even when node access to samples is not assumed, but rather only to a pair of values of the local objectives, the derived convergence rate suffers no degradation.

4.5 Experiments

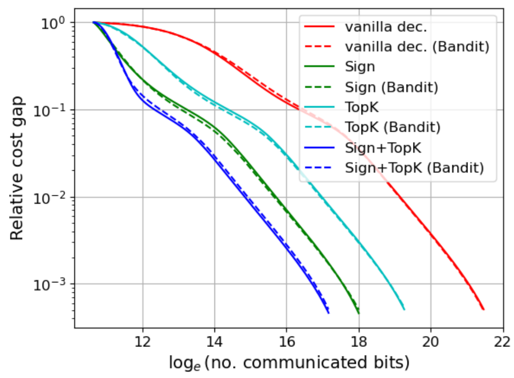
We provide simulations to demonstrate the effectiveness of our proposed scheme for communication efficient optimization.



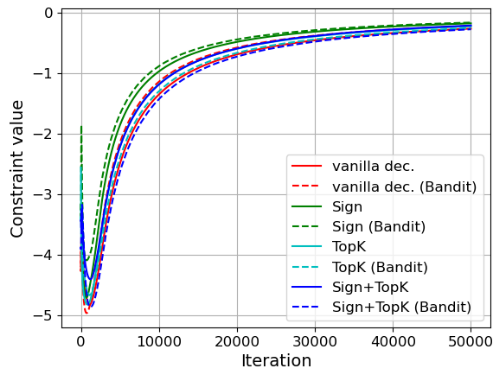
(a) Comparison of relative cost gap $\frac{F(\bar{\mathbf{x}}^{(t)})-F(\mathbf{x}^*)}{F(\bar{\mathbf{x}}^{(1)})-F(\mathbf{x}^*)}$ at iteration t.



(b) Comparison of relative parameter error $\frac{\|\bar{\mathbf{x}}^{(t)}-\mathbf{x}^*\|}{\|\mathbf{x}^*\|}$ at iteration t.



(c) Comparison of relative cost gap comparison for number of bits communicated for different schemes.



(d) Constraint value $g_{ij}(\bar{\mathbf{x}}_i^{(t)}, \bar{\mathbf{x}}_j^{(t)})$ for a randomly chosen edge (i, j).

Figure 4.1 Performance comparison for various schemes on a decentralized QCQP objective in (4.15)

4.5.1 QCQP Objective

4.5.1.1 Setup and Hyperparameters

We consider decentralized optimization on a randomly generated Erdos-Renyi graph of $n = 30$ nodes with an edge probability of 0.15. For each node $i \in [n]$, we consider a quadratic objective given by $f_i(\mathbf{x}_i, \boldsymbol{\xi}) = \mathbf{x}_i^T \mathbf{A}_i \mathbf{x}_i + \mathbf{b}_i^T \mathbf{x}_i$ where $\mathbf{x}_i$ denotes the node model parameter and $\boldsymbol{\xi}_i = (\mathbf{A}_i, \mathbf{b}_i)$ denotes the sample. For each node, $\mathbf{A}_i \in \mathbb{R}^{10\times 10}$ is sampled from a Wishart distribution with 10 degrees of freedom identity scaling matrix, and vector $\mathbf{b}_i$ is sampled from a Gaussian distribution with mean and variance drawn uniformly at random from the interval [0,1] in each iteration.

We consider the feasible parameter space $\mathcal{X}$ to be the Euclidean ball of radius $\frac{40}{\sqrt{30}}$ centered at the origin. For each $i \in [n], j \in \mathcal{N}_i$, we model the constraints on the node parameters as $g_{ij}(\mathbf{x}_i, \mathbf{x}_j) = \|\mathbf{x}_i - \mathbf{x}_j\|^2 + c_{ij}$ where c_{ij} is independently drawn uniformly at random from $[-5, -3]$. The overall objective is thus given by:

$$\begin{aligned} \min_{\{\mathbf{x}_1,\dots,\mathbf{x}_n\}\in\mathcal{X}} F(\mathbf{x}) &= \sum_{i=1}^{n} \mathbb{E}_{\boldsymbol{\xi}_i}[\mathbf{x}_i^T \mathbf{A}_i \mathbf{x}_i + \mathbf{b}_i^T \mathbf{x}_i] \\ \text{s.t. } \|\mathbf{x}_i - \mathbf{x}_j\|_2^2 + c_{ij} &\le 0, \quad \forall i \in [n], j \in \mathcal{N}_i \end{aligned} \tag{4.15}$$

where $\mathbf{x}$ denotes concatenation of $\{\mathbf{x}_i, \dots, \mathbf{x}_n\}$. Note that choosing $c_{ij} \le 0$ for all $i \in [n], j \in \mathcal{N}_i$ implies that the above QCQP has a non-empty feasible set. We set $\eta = 0.001$, and choose $\delta = 100$, and run all considered schemes for a total of 5×10^4 iterations. For gradient estimation in case of bandit feedback, we take $\zeta = 10^{-4}$.

4.5.1.2 Results

The simulation results for optimizing objective (4.15) are presented in Figure 4.1, where we compare vanilla decentralized (no compression) algorithm with our proposed compressed optimization procedure using *Sign* [KRSJ19a], *TopK* [SCJ18a] and composed *Sign* + *TopK* [BDKD19a] compression operators. Schemes with *'Bandit'* in parenthesis indicate those implemented via Algorithm 3 for the case of gradient estimation in bandit feedback, and via Algorithm 2 with sample feedback otherwise. Figure 4.1(a) shows the relative cost gap for the objective given by $\frac{F(\bar{\mathbf{x}}^{(t)})-F(\mathbf{x}^*)}{F(\bar{\mathbf{x}}^{(1)})-F(\mathbf{x}^*)}$, and Figure 4.1(b) shows the difference of the parameter from the optimal value normalized to the latter, given by $\frac{\|\bar{\mathbf{x}}^{(t)}-\mathbf{x}^*\|}{\|\mathbf{x}^*\|}$ for iteration t. We conclude that schemes with compression, including the ones implemented via bandit feedback, effectively perform the same as uncompressed vanilla training to minimize the objective. The benefit of our proposed scheme can be seen in Figure 4.1(c), where we plot the relative cost gap with the number of bits communicated among the nodes, assuming precision of 32bit floats. To achieve a target relative cost gap of around 10^{-3}, compressed schemes use significantly fewer bits than vanilla decentralized training, saving a factor of about $7\times$ with *TopK* compression, factor of $30\times$ when using *Sign* compression operation, and a factor of around $50\times$ for the composed *Sign* + *TopK* compression operator. Figure 4.1(d) shows the constraint $g_{ij}(\bar{\mathbf{x}}_i^{(t)}, \bar{\mathbf{x}}_j^{(t)})$ for a randomly chosen $i \in [n]$ and $j \in [n]$. The constraint value settles to a negative value, which implies that each scheme arrives at an objective value lying in the feasible space of the problem (4.15).

In conclusion, our proposed schemes in Algorithms 2 and 3 for communication efficient decentralized optimization provide performance similar to that in the full precision vanilla decentralized method, while saving substantially in the total number of bits communicated among the nodes during the optimization process.

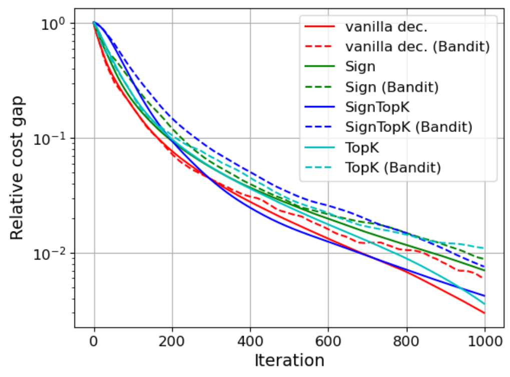
4.5.2 Logistic Regression Objective

4.5.2.1 Setup and Hyperparameters

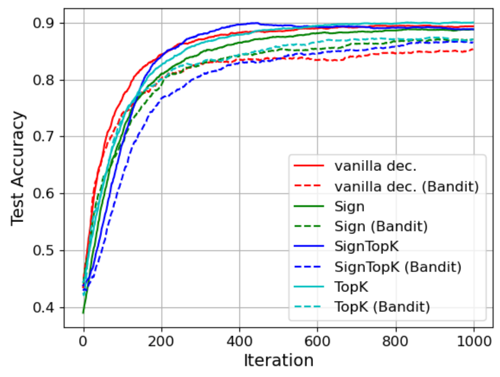
We again work with an Erdos-Renyi graph of $n = 30$ nodes with an edge probability of 0.3. We consider a logistic regression setting where feature vectors $\mathbf{p}_i \in \mathbb{R}^d$ (d=10) for each node are generated from a standard normal distribution. The corresponding output $y_i \in \{1, -1\}$ is sampled under the probability: $p(y_i = 1) = \frac{1}{1+e^{-\mathbf{x}_i^\top \mathbf{p}_i}}$ where $\mathbf{x}_i$ denotes the underlying node model parameter. We generate the underlying model parameters such that they are close (in norm sense) for adjacent nodes. We denote by $\boldsymbol{\xi}_i$ the pair $(\mathbf{p}_i, y_i)$ for each node, which are data samples generated for each iteration of the algorithm. The objective of the nodes is to maximize the log-likelihood of the generated data, which can equivalently be expressed by the following optimization problem:

$$\begin{aligned} \min_{\{\mathbf{x}_1,\ldots,\mathbf{x}_n\}\in\mathcal{X}} F(\mathbf{x}) &= \sum_{i=1}^{n} \mathbb{E}_{\boldsymbol{\xi}_i}[\log(1 + e^{-y_i \mathbf{x}_i^\top \mathbf{p}_i})] \\ \text{s.t. } \|\mathbf{x}_i - \mathbf{x}_j\|_2^2 + c_{ij} &\le 0, \quad \forall i \in [n], j \in \mathcal{N}_i \end{aligned} \tag{4.16}$$

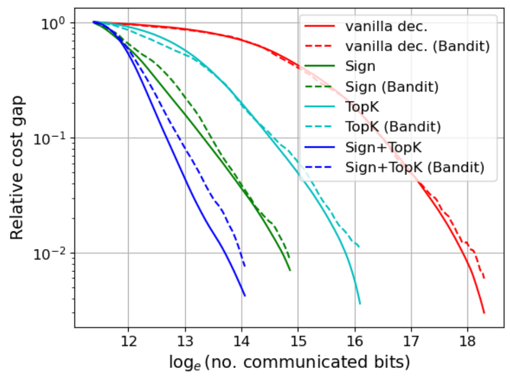
As in the earlier QCQP formulation, we consider $R = 40$ and the constraints on the node parameters to be $g_{ij}(\mathbf{x}_i, \mathbf{x}_j) = \|\mathbf{x}_i - \mathbf{x}_j\|^2 + c_{ij}$ where c_{ij} are independently drawn uniformly at random from $[-10, -7]$. We set $\eta = 0.001$, and choose $\delta = 100$, and run all considered schemes for a total of 10^3 iterations. For gradient estimation in case of bandit feedback, we take $\zeta = 10^{-4}$. To evaluate the models for their generalization capabilities, we also evaluate their classification performance on a test set (comprising of 500 samples per node).



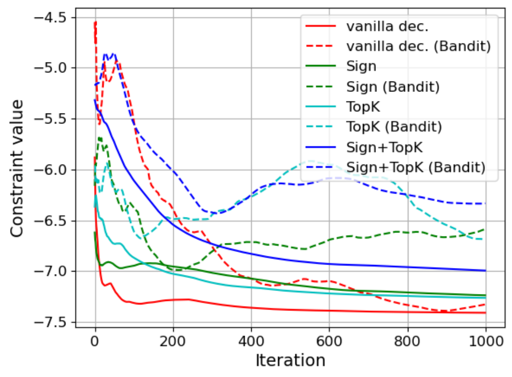
(a) Comparison of relative cost gap $\frac{F(\bar{\mathbf{x}}^{(t)})-F(\mathbf{x}^*)}{F(\bar{\mathbf{x}}^{(1)})-F(\mathbf{x}^*)}$ at iteration t.



(b) Comparison of test accuracy performance at iteration t.



(c) Comparison of relative cost gap comparison for number of bits communicated for different schemes.



(d) Constraint value $g_{ij}(\bar{\mathbf{x}}_i^{(t)}, \bar{\mathbf{x}}_j^{(t)})$ for a randomly chosen edge (i, j).

Figure 4.2 Performance comparison for various schemes on logistic regression training in (4.16).

4.5.2.2 Results

We compare the performance of vanilla decentralized training for optimizing (4.16) against our proposed method using compression in Figure 4.2. Figure 4.2(a) shows the relative sub-optimality of the different model parameters against the true model[3]. Figure 4.2(b) shows the test accuracy performance of the datasets, where we observe all compression schemes achieving similar accuracy as uncompressed vanilla training. To see the gain in using compression, we plot the relative sub-optimality against the total numbers of bits communicated in Figure 4.2(c), where we observe that to achieve a similar level of sub-optimality of around 10^{-2}, compared to vanilla decentralized training, *SignTopK* compression saves a factor of about $50\times$, *Sign* compression saves a factor of $20\times$ and *TopK* compression saves a factor of around $7\times$. This, in conclusion, demonstrates the advantage of using our proposed communication efficient scheme for a logistic regression based classification scenario. The constraint values for all the schemes for a randomly chosen edge are shown in Figure 4.2(d), where we observe that all schemes settle to a negative value, and thus end up in the feasible space of the problem (4.16).

4.6 Conclusion

We proposed and analyzed a communication-efficient saddle-point algorithm for multi-task decentralized learning under sample feedback and bandit feedback data access scenarios. Our theoretical results demonstrated order-wise same performance as un-compressed training for convex objectives while saving significantly on the number of bits transmitted, which is also corroborated by our numerical experiments.

As many learning paradigms consider non-convex objectives, e.g. Deep Learning,

[3]We remark that for a large enough values of c_{ij} such that $g(\mathbf{x}_i, \mathbf{x}_j) \leq 0$ for all $\mathbf{x}_i, \mathbf{x}_j$ (where $\mathbf{x}_i$ denotes the optimal model parameter for node i that generates the data), the optimal solution $\mathbf{x}^*$ is the stacking of all $\mathbf{x}_i, i \in [n]$.

it would be of interest to extend the analysis of our proposed algorithm to such settings as part of future work. It is also of interest to incorporate additional mechanisms for communication reduction along with compression in our proposed algorithm for greater communication efficiency such as local gradient iterations or triggered-communication [BDKD19a,SDGD23], and theoretically analyze the resulting procedure.

CHAPTER 5

Representation Transfer

In contemporary machine learning, effectively utilizing limited data to build accurate models is critically important, particularly in domains where data acquisition is expensive or time-consuming. Linear regression, a fundamental technique in statistics and machine learning, often encounters challenges in such scenarios due to the necessity of sufficient data to ensure robust model training. Despite its relative simplicity, linear regression serves as a crucial stepping stone toward understanding and implementing more complex models, such as neural networks. Enhancing linear regression through innovative methods not only addresses immediate challenges but also lays the groundwork for advancements in more sophisticated machine learning models.

One promising solution to this challenge is leveraging pre-trained models from related domains—an approach known as transfer learning. This chapter explores the construction of a linear regression model for a target data domain characterized by a scarcity of samples. Our innovative approach utilizes a set of pre-trained regression models, each derived from potentially diverse source domains. By assuming a structured representation for the data-generating linear models across both source and target domains, we propose a representation transfer-based learning method tailored to the target model.

The proposed methodology unfolds in two crucial phases:

1. **Representation Adaptation:** This phase involves using the varied source representations to construct a new representation that is finely tuned to the target data. This adaptation leverages the inherent structures and patterns from the source domains to inform the target representation.

2. **Model Fine-Tuning:** In this subsequent phase, the adapted representation serves as an initialization for a comprehensive fine-tuning procedure. This process entails re-training the entire, potentially over-parameterized, regression model on the target data, refining it to better fit the specificities of the target domain.

For each of these phases, we provide rigorous theoretical guarantees in the form of excess risk bounds. These bounds compare the learned model's performance to that of the true data-generating target model, illustrating a notable improvement in sample complexity. Specifically, our method demonstrates a reduction in the number of samples required to achieve a comparable level of excess risk, thereby underscoring the practical benefits of incorporating transfer learning into linear regression tasks.

In this chapter, we delve into the detailed development and implications of this representation transfer-based learning approach. We discuss the theoretical foundations underpinning our method, present empirical results that validate its effectiveness, and highlight the broader impact of transfer learning on enhancing linear regression models in data-constrained environments. This work not only advances the theoretical understanding of transfer learning but also provides practical insights for its application in real-world scenarios where data limitations are a significant concern.

5.1 Introduction

A critical challenge for Deep Learning applications is the scarcity of available labeled data to train large scale models that generalize well to the data distribution. This is captured under the framework of *Few-Shot Learning* where Transfer Learning has emerged as an attractive framework to address this issue [LCS18]. In transfer learning, one typically has access to a model trained on some data domain (hereby called *source domain*) that can be adapted to the data domain of interest (*target domain*). Within this context, a recently proposed strategy is that of *representation transfer learning* [BCV13, Ben12], where one typically assumes a shared structure between the source and target learning tasks. The idea is to then learn a feature mapping for the underlying model (e.g. Neural Network representations) using the sample rich source domain that can be utilized directly on the target domain, for e.g, by training a few layers on top of the obtained network representation. This adaptation utilizes much fewer samples than what is required for training the entire model from scratch, while achieving good generalization performance which has been empirically observed for various large-scale machine learning and signal processing applications including image, speech and language [ZZ22, LCS18, JZW$^+$18, LLS20] tasks.

A defining factor in the need for representation transfer methods is that the source and target domains have different distributions. Learning across different domains has been studied extensively in the context of *Domain Adaptation* (see for e.g. [BCK$^+$07, MMR09]) where it is usually assumed that source and target domain data can be accessed simultaneously. However, in many important practical scenarios of interest, the target data samples (labeled or unlabeled) are not available when training the source models. Transferring the source dataset to the target deployment scenario is infeasible for modern large-scale applications and violates data privacy. Thus there has been an increasing interest in transferring pre-trained source models to the target domain for sample efficient learning.

Despite the immense empirical success of representation transfer learning, development of a theory for understanding the generalization of representation learning methods and the sample complexity requirements is still in its infancy. Recent efforts in this direction have been made in understanding generalization for the simpler case of linear regression models [DHK+21, CLL21, TJJ21, MPRP16]. Within these works, [DHK+21, TJJ21, MPRP16] consider a common low-dimensional representation in the data generation process for the source and target domains, while [CLL21] allows for the general case of data-generating representations being different. However, the analysis presented in that work requires the number of samples at the target to scale with the dimension of the model (see [CLL21, Theorem 3.1]), which is impractical for few-shot learning scenarios.

A related line of work for understanding generalization of large scale models in the small sample regime is through the lens of *benign overfitting.* This is inspired by the surprising (empirical) observation that many large models, even when they overfit, tend to generalize well on the data distribution [BHM18, ADH+19]. In this context, [BLLT20, HMRT22, SBKS20] study this phenomena for linear models and analyze the generalization properties of the *min-norm solution*, where optimization methods like Gradient Descent are known to converge to in this setting [WZBG21, GLSS18]. Specifically, these works seek to understand how the data distribution affects the excess population risk of the min-norm solution relative to the true data-generating linear model.

We make efforts to understanding the generalization of linear models while leveraging pre-trained models inspired by the notions of representation transfer learning and benign-overfitting discussed above. These ideas lend themselves organically to the construction of a sample efficient training method for the target which we describe below briefly, along with our contributions.

This work provides a method for leveraging multiple pre-trained models for linear regression objectives (of dimension d) on a target task of interest in the small sample regime (samples $n_T \ll d$). The proposed two-phase approach leverages representation transfer (Phase 1) and over-parameterized training (Phase 2) to construct the target model, and we provide theoretical bounds for the excess risk for each phase of the training process (Theorem 5 and Theorem 6). In particular, we show that the learned model after the first phase has an excess risk of $\mathcal{O}\left({}^{q}/_{n_T}\right) + \epsilon$, where q is dimension of the subspace spanned by learned source representations and ϵ is a constant that captures the approximation error when utilizing source representations for the target model (c.f. Assumption 9). This provides a gain in sample complexity compared to the baseline $\mathcal{O}\left({}^{d}/_{n_T}\right)$ when learning the target model from scratch when the given source representations span a subspace of dimension much smaller than d (i.e. $q \ll d$). For the case when all representations are the same ($\epsilon = 0$), we recover the result of [DHK$^+$21] for a *single* common representation. Similarly, for the overall model obtained after the second phase, we provide conditions on the target data distribution and the source/target representations that lead to an excess risk much smaller than $\mathcal{O}\left({}^{d}/_{n_T}\right)$. Thus, we theoretically demonstrate the benefit of leveraging pre-trained models for linear regression.

5.1.1 Related Work

The problem of learning with few samples has been studied under the framework of *Few-Shot learning*, where *Meta-learning*– using experiences from previously encountered learning tasks to adapt to new tasks quickly [FAL17], and *Transfer-learning*– transferring model parameters and employing pre-training or fine-tuning methods [Ben12], are two major approaches. Theoretical works on Meta-learning algorithms typically assume some relation between the distribution of source and target tasks, for e.g., being sampled from the same task distribution. A more general framework is that of *Out-of-Domain (OOD)* generalization, where the goal is to learn models in a manner that generalize well to unseen data domains [WLL+22].

Transfer learning, especially, representation transfer learning has shown empirical success for large-scale machine learning [BCV13], however, theoretical works on understanding generalization in this setting are few; see [MB17, GWH16, MPRP16]. A related line of work is representation learning in context of *Domain Adaptation (DA)*, see for e.g. [ZDCZG19, BDBCP07, SLG+21, ZLLJ19]. However, this usually assumes that source and data domains can be accessed simultaneously. There are deviations from this theme in *Multi-Source DA* where the goal is to understand how multiple source models can be combined to generalize well on a target domain of interest, although without changing the learned model based on the target samples [MMR08, LHF20, ARP+21].

In context of leveraging pre-trained models for linear regression, our work is most closely related to [DHK+21, CLL21] that theoretically analyze representation transfer for linear models. In contrast to [DHK+21], we allow for the true target representations to be different among the source models as well as the target, and introduce a notion of closeness between these representations (c.f. Assumption 9). Although a similar setting was considered in [CLL21] where representations are assumed to be close in the ℓ_2 norm, their resulting bound for the fine-tuned model risk shows that the required number of target samples scale with the dimension of the learned model for efficient transfer [CLL21, Theorem 3.1]. In contrast, the proposed method in our work provides analysis relating these bounds to the properties of the target data distribution taking inspiration from works on benign overfitting for linear regression [BLLT20, GLSS18, SBG21]. This enables us to identify conditions on the target data distribution that allow the required target samples to be much smaller than the overall model dimension (see Theorem 6).

5.1.2 Chapter Organization

We set up the problem and define the notation we use throughout the chapter in Section 5.2. Section 5.3 describes our training method for the target task model when given access to multiple pre-trained source models. Section 5.4 establishes excess risk bounds of our proposed scheme, which are proved in Sections C.1 and Section C.2. Section 5.5 provides numerical results and some concluding remarks are presented in Section 5.6. Omitted proof details are provided in Appendix C of this thesis.

5.2 Problem Setup and Notation

Notation: We use boldface for vectors and matrices, with matrices in uppercase. For a matrix $\mathbf{A}$, we denote the projection matrix onto its column space by $\mathbf{P}_{\mathbf{A}} := \mathbf{A}(\mathbf{A}^\top\mathbf{A})^\dagger\mathbf{A}^\top$ where $\mathbf{W}^\dagger$ denotes the Moore-Penrose pseudo-inverse of the matrix $\mathbf{W}$. We define $\mathbf{P}_{\mathbf{A}}^\perp := \mathbf{I} - \mathbf{P}_{\mathbf{A}}$, where $\mathbf{I}$ denotes the identity matrix of appropriate dimensions. We denote by $\mathcal{C}(\mathbf{A})$ the column space of a matrix $\mathbf{A}$ and by $\sigma_i(\mathbf{A})$, $\lambda_i(\mathbf{A})$ its i^{th} largest singular value and eigenvalue, respectively. $\|.\|_F$ denotes the Frobenius norm. For a vector $\mathbf{v}$, $\|\mathbf{v}\|_2$ denotes the ℓ_2 norm, while for a matrix $\mathbf{V}$, $\|\mathbf{V}\|_2$ denotes the spectral norm. $\mathrm{Tr}[\,.\,]$ denotes the trace operation. $\lesssim$ denotes the inequality sign where we ignore the constant factors. The notation $\mathcal{O}$ is the 'big-O' notation and we define $[m] = \{1, 2, \ldots, m\}$.

Setup: We consider m number of source tasks and a single target task. We denote by $\mathcal{X} \subseteq \mathbb{R}^d$ the space of inputs and $\mathcal{Y} \subseteq \mathbb{R}$ the output space. The source and target tasks are associated with data distributions p_i, $i \in [m]$ and p_T, respectively, over the space $\mathcal{X}$. We assume a linear relationship between the input and output pairs for source task $i \in [m]$ given by:

$$y_i = \mathbf{x}_i^\top \mathbf{B}_i^* \mathbf{w}_i^* + z_i, \quad \boldsymbol{\theta}_i^* := \mathbf{B}_i^* \mathbf{w}_i^* \tag{5.1}$$

where $\mathbf{x}_i \in \mathcal{X}$ denotes an input feature vector drawn from distribution p_i, $y_i \in \mathcal{Y}$ is the output, and $z_i \sim \mathcal{N}(0, \sigma^2)$ denotes Gaussian noise. The associated true task parameter $\boldsymbol{\theta}_i^* := \mathbf{B}_i^* \mathbf{w}_i^*$ is comprised of the representation matrix $\mathbf{B}_i^* \in \mathbb{R}^{d \times k}$ which maps the input to a lower $k-$dimensional space (where $k \ll d$) and a head vector $\mathbf{w}_i^* \in \mathbb{R}^k$ mapping the intermediate sample representation to the output. The data generation process for the target task is defined similarly with distribution p_T and associated target task parameter given by $\boldsymbol{\theta}_T^* = \mathbf{B}_T^* \mathbf{w}_T^*$. For sources $i \in [m]$, we define the input covariance matrix $\boldsymbol{\Sigma}_i = \mathbb{E}_{\mathbf{x}_i \sim p_i}[\mathbf{x}_i \mathbf{x}_i^\top]$ and similarly for the target distribution, $\boldsymbol{\Sigma}_T = \mathbb{E}_{\mathbf{x}_T \sim p_T}[\mathbf{x}_T \mathbf{x}_T^\top]$.

In our scenario of interest, the pre-trained models are trained 'offline' on source distributions and are made available to the target task during deployment. That is, for training the target task, we have access to only the models learned by the source tasks and not the source datasets themselves. For learning the pre-trained source models, we assume n_S number of samples for each of the source tasks (thus, mn_S source task samples in total) denoted by the pair $(\mathbf{X}_i, \mathbf{y}_i)$ for source $i \in [m]$ where $\mathbf{X}_i \in \mathbb{R}^{n_S \times d}$ contains row-wise input feature vectors and $\mathbf{y}_i \in \mathbb{R}^{n_S}$ is the vector of corresponding outputs. We similarly have n_T samples $(\mathbf{X}_T, \mathbf{y}_T)$ for the target task where $n_T \ll n_S$. We also assume $n_T \ll d$.

With the data generation process defined above, we now define the expected population risk on the target distribution for $\hat{\boldsymbol{\theta}}$:

$$R(\hat{\boldsymbol{\theta}}) = \mathbb{E}_{\mathbf{x} \sim p_T} \mathbb{E}_{\mathbf{y}|\mathbf{x}^\top \boldsymbol{\theta}_T^*}[(\mathbf{y} - \mathbf{x}^\top \hat{\boldsymbol{\theta}})^2]$$

Our goal is to learn a model $\hat{\boldsymbol{\theta}}$ for the target task that generalizes well to the target data distribution. Thus, we want $\hat{\boldsymbol{\theta}}$ that minimizes the *Expected Excess Risk* defined by:

$$\text{EER}(\hat{\boldsymbol{\theta}}, \boldsymbol{\theta}_T^*) := R(\hat{\boldsymbol{\theta}}) - R(\boldsymbol{\theta}_T^*) \tag{5.2}$$

Since we are given access to only $n_T \ll d$ target samples, it is infeasible to learn a predictor from scratch that performs well for the excess risk defined in (5.2).

To aid learning on the target, we have access to models learned on the source tasks. Specifically, the target has access to the trained source models representations $\{\widehat{\mathbf{B}}_i\}_{i=1}^m$

that are solutions of the following empirical minimization problem:

$$\{\widehat{\mathbf{B}}_i, \widehat{\mathbf{w}}_i\}_{i=1}^m \leftarrow \min_{\{\mathbf{B}_i\},\{\mathbf{w}_i\}} \frac{1}{mn_S} \sum_{i=1}^{m} \|\mathbf{y}_i - \mathbf{X}_i\mathbf{B}_i\mathbf{w}_i\|_2^2 \tag{5.3}$$

Since we have data rich source domains ($n_S \gg d$), we expect the obtained source models $\widehat{\mathbf{B}}_i\widehat{\mathbf{w}}_i$ to be close to $\boldsymbol{\theta}_i^*$ for $i \in [m]$ (c.f. Equation (5.1)). For effective representation transfer, we also want the learned representations $\{\widehat{\mathbf{B}}_i\}$ to be close to the true representations $\{\mathbf{B}_i^*\}$, in the sense that they approximately span the same subspace. We make this notion precise in Lemma 1 stated with our main results in Section 5.4. Given access to the source model representations, our proposed method for training the target model leverages the representation maps $\{\widehat{\mathbf{B}}_i\}_{i=1}^m$ to drastically reduce the sample complexity. We describe our training method in Section 5.3 and provide the excess risk bounds for the resulting target model in Section 5.4.

5.3 Learning with Multiple Pre-trained models

To leverage source representations for training the target model, it is instinctive that there should be a notion of closeness between the true source and target model representation that can be exploited for target task training. We now make this notion precise. We first define as $\mathbf{V}^* \in \mathbb{R}^{d\times l}$ the matrix whose columns are an orthonormal basis of the set of columns of all the source representation matrices $\{\mathbf{B}_i^*\}_{i=1}^m$. The individual source models can thus be represented by $\boldsymbol{\theta}_i^* = \mathbf{V}^*\widetilde{\mathbf{w}}_i^*$ for all $i \in [m]$. The target model $\boldsymbol{\theta}_T^* = \mathbf{B}_T^*\mathbf{w}_T^*$ governing the target data generation is assumed to satisfy the following:

Assumption 9. *Consider the projection of the target model* $\mathbf{B}_T^*\mathbf{w}_T^*$ *to space* $\mathcal{C}(\mathbf{V}^*)$

given by $\mathbf{V}^*\widetilde{\mathbf{w}}_T^*$ *for some* $\widetilde{\mathbf{w}}_T^* \in \mathbb{R}^l$*. Then for some* $\epsilon > 0$*, we have:*

$$\mathbb{E}_{\mathbf{x}\sim p_T}\left[\mathbf{x}^\top \mathbf{V}^*\widetilde{\mathbf{w}}_T^* - \mathbf{x}^\top \mathbf{B}_T^*\mathbf{w}_T^*\right]^2 \leq \epsilon^2$$

The value of ϵ *in Assumption 9 above captures how far away the output of the true target model is to a model learned using the true source representations. Note that if the columns of* $\mathbf{B}_T^*$ *can be constructed by the vectors in* $\mathbf{V}^*$*, the above is satisfied for* $\epsilon = 0$*. Assumption 9 can also be re-written as:*

$$\left\|\Sigma_T^{1/2}\left(\mathbf{V}^*\widetilde{\mathbf{w}}_T^* - \mathbf{B}_T^*\mathbf{w}_T^*\right)\right\|_2^2 \leq \epsilon^2 \tag{5.4}$$

We are given access to n_T samples for the target machine given by $(\mathbf{X}_T, \mathbf{y}_T)$ and pre-trained models representations from the sources $\{\widehat{\mathbf{B}}_i\}_{i=1}^m$ (c.f. Equation (5.3)). Our proposed training scheme consists of two phases, which we will now describe independently in the following. We split the available n_T target samples into n_{T_1}, n_{T_2} for the two respective phases. At a high level, in *Phase 1*, we make use of the available source representations to construct a target representation and adapt it to the target task using n_{T_1} samples. The obtained model is then used as an *initialization* for *Phase 2* where we train the entire (over-parameterized) model, including the representation matrix, using n_{T_2} samples. We provide the resulting excess risk bounds for the model obtained after *Phase 1* and the final target model after *Phase 2* in Section 5.4.

5.3.1 Phase 1: Transferring source representation to target

In the context of utilizing pre-trained models, we will make use of the empirical source representations $\{\widehat{\mathbf{B}}_i\}_{i=1}^m$ to learn the target model. We first construct a matrix

$\widehat{\mathbf{V}} \in \mathbb{R}^{d\times q}$ whose columns are the orthonormal basis of the columns of $\{\widehat{\mathbf{B}}_i\}_{i=1}^m$ which denotes a dictionary of the learned source representation matrices[1]. Note that we have $q \leq mk$. Having constructed the representation, we train a head vector $\widehat{\mathbf{w}}_{T_1} \in \mathbb{R}^q$ minimizing the empirical risk on n_{T_1} samples:

$$\widehat{\mathbf{w}}_{T_1} \leftarrow \min_{\mathbf{w}_T \in \mathbb{R}^q} \frac{1}{n_{T_1}} \left\| \mathbf{y}_{T_1} - \mathbf{X}_{T_1} \widehat{\mathbf{V}} \mathbf{w}_T \right\|_2^2 \tag{5.5}$$

Here, $\mathbf{y}_{T_1} \in \mathbb{R}^{n_{T_1}}$ denotes the first n_{T_1} values of $\mathbf{y}_T$ and $\mathbf{X}_{T_1} \in \mathbb{R}^{n_{T_1}\times d}$ the first n_{T_1} rows of $\mathbf{X}_T$. We denote the resulting model at the end of this phase by $\boldsymbol{\theta}_{\text{Phase}_1} := \widehat{\mathbf{V}}\widehat{\mathbf{w}}_{T_1}$. Since we only have to learn the head vector using the available representation $\widehat{\mathbf{V}}$, the sample complexity requirement is greatly reduced, which is also evident from our bound for $\text{EER}(\boldsymbol{\theta}_{\text{Phase}_1}, \boldsymbol{\theta}_T^*)$ provided in Theorem 5.

5.3.2 Phase 2: Fine-tuning with initialization

The obtained model $\boldsymbol{\theta}_{\text{Phase}_1}$ from the previous phase utilizes empirical source representation for its construction. However, the true target model $\boldsymbol{\theta}_T^*$ may not lie in the space spanned by the source representation and thus $\boldsymbol{\theta}_{\text{Phase}_1}$ lies in a ball centered $\boldsymbol{\theta}_T^*$ whose radius scales with ϵ (c.f. Assumption 13). To move towards the true model $\boldsymbol{\theta}_T^*$, we utilize n_{T_2} number of target samples (independent from the n_{T_1} samples in the previous phase) to train the entire linear model using Gradient Descent (GD) with $\boldsymbol{\theta}_{\text{Phase}_1}$ as the initialization. In particular, the GD procedure minimizes the following

[1]The construction of $\widehat{\mathbf{V}}$ from $\{\widehat{\mathbf{B}}_i\}$ can be done by the Gram-Schmidt process. This can be done in the pre-deployment phase after training the source models, and $\widehat{\mathbf{V}}$ can be made available directly to the target task.

starting from $\boldsymbol{\theta}_{\text{Phase}_1}$:

$$f(\boldsymbol{\theta}) = \frac{1}{n_{T_2}} \left\| \mathbf{y}_{T_2} - \mathbf{X}_{T_2}\boldsymbol{\theta} \right\|_2^2 \tag{5.6}$$

Here, $\mathbf{y}_{T_2} \in \mathbb{R}^{n_{T_2}}$ and $\mathbf{X}_{T_2} \in \mathbb{R}^{n_{T_2} \times d}$ are the remaining sample values from Phase 1. Since $n_{T_2} \ll d$, we are in an over-parameterized regime, for which it is known that GD procedure optimizing the objective in (5.6) converges, under appropriate choice of learning rate, to a solution closest in norm to the initialization [WZBG21, BHX20, GLSS18]; mathematically:

$$\begin{aligned} &\min_{\boldsymbol{\theta}} \left\| \boldsymbol{\theta} - \boldsymbol{\theta}_{\text{Phase}_1} \right\|_2 \\ &\text{s.t. } \left\| \mathbf{y}_{T_2} - \mathbf{X}_{T_2}\boldsymbol{\theta} \right\|_2 = \min_{\mathbf{b}} \left\| \mathbf{y}_{T_2} - \mathbf{X}_{T_2}\mathbf{b} \right\|_2 \end{aligned} \tag{5.7}$$

We denote the solution of the above optimization problem as $\boldsymbol{\theta}_{\text{Phase}_2}$, which forms our final target task model. We provide bounds for $\text{EER}(\boldsymbol{\theta}_{\text{Phase}_2}, \boldsymbol{\theta}_T^*)$ in Theorem 6.

5.4 Main Results

We now provide theoretical bounds on the excess risk for the target (c.f. Equation (5.2)) when leveraging pre-trained source models. In Section 5.4.1, we first state excess risk bounds for the model obtained after Phase 1 (see Section 5.3.1), denoted by $\boldsymbol{\theta}_{\text{Phase}_1} := \widehat{\mathbf{V}}\widehat{\mathbf{w}}_T$, where target representation $\widehat{\mathbf{V}}$ is constructed as a combination of source representations and adapted to the target data using n_{T_1} amount of target samples by training a target-specific head vector $\widehat{\mathbf{w}}_T$. In Section 5.4.2, we provide our overall excess risk for the model $\boldsymbol{\theta}_{\text{Phase}_2}$ (c.f. Equation (5.7)) obtained by re-training the entire (over-parameterized) model via Gradient Descent with n_{T_2} number of target samples (independent form the previously utilized n_{T_1} samples) using $\boldsymbol{\theta}_{\mathrm{Phase}_1}$ as the initialization.

5.4.1 Theoretical results for representation transfer: Phase 1

We work with the following assumptions:

Assumption 10 (Subgaussian features)**.** *We assume that $\mathbb{E}_{\mathbf{x}\sim p_j}[\mathbf{x}] = 0$ for all $j \in [m] \cup \{T\}$. We consider $\bar{p}_j$ to be the* whitening *of p_j (for $j \in [m] \cup \{T\}$) such that $\mathbb{E}_{\bar{\mathbf{x}}\sim \bar{p}_j}[\mathbf{x}] = 0$ and $\mathbb{E}_{\bar{\mathbf{x}}\sim \bar{p}_j}[\bar{\mathbf{x}}\bar{\mathbf{x}}^\top] = \mathbf{I}$. We assume there exists $\rho > 0$ such that the random vector $\bar{\mathbf{x}} \sim \bar{p}_j$ is ρ^2-subgaussian.*

Assumption 11 (Covariance Dominance)**.** *There exists $r > 0$ such that $\Sigma_i \succeq r\Sigma_T$ for all $i \in [m]$.*

Assumption 12 (Diverse source tasks)**.** *Consider the source models $\boldsymbol{\theta}_i^* = \mathbf{V}^*\widetilde{\mathbf{w}}_i^*$ for $i \in [m]$. We assume that the matrix $\widetilde{\mathbf{W}}^* := [\widetilde{\mathbf{w}}_1^*, \ldots, \widetilde{\mathbf{w}}_m^*] \in \mathbb{R}^{l\times m}$ satisfies $\sigma_l^2(\widetilde{\mathbf{W}}^*) \geq \Omega\left(\frac{m}{l}\right)$*

Assumption 13 (Distribution of target task)**.** *We assume that $\widetilde{\mathbf{w}}_T^*$ follows a distribution ν such that $\left\|\mathbb{E}_{\widetilde{\mathbf{w}}\sim\nu}[\widetilde{\mathbf{w}}\widetilde{\mathbf{w}}^\top]\right\|_2$ is $\mathcal{O}\left(\frac{1}{l}\right)$. We denote $\Sigma_{\widetilde{\mathbf{w}}_T^*} = \mathbb{E}_{\widetilde{\mathbf{w}}\sim\nu}[\widetilde{\mathbf{w}}\widetilde{\mathbf{w}}^\top]$.*

Note on Assumptions: Assumption 10 on sub-Gaussian features is commonly used in literature to obtain probabilistic tail bounds [DHK+21, CLL21, SBG21, BLLT20]. Following [DHK+21], Assumption 11 states the target data covariance matrix is covered by the covariance matrices of the source data distributions. We remark that this assumption allows the covariance matrices to be different, in contrast to works [CLL21, TJJ21] that assume a common covariance matrix for all the distributions. Assumption 12 (also made in related works [CHMS21, DHK+21, CLL21]) says that the head vectors corresponding to the matrix $\mathbf{V}^*$ for each source model should span $\mathbb{R}^l$. This effectively allows us to recover the representation $\mathbf{V}^*$ provided enough source machines ($m > l$) that individually capture one or more features of $\mathbf{V}^*$. This assumption is also central to proving our result in Lemma 1 provided below which show that the matrices $\widehat{\mathbf{V}}$ and $\mathbf{V}^*$, whose columns form an orthonormal basis for the span of $\{\widehat{\mathbf{B}}_i\}$ and $\{\mathbf{B}_i^*\}$, respectively, span the same subspace for constructing the target model.

Lemma 1. *Let matrix $\widehat{\mathbf{V}} \in \mathbb{R}^{d\times q}$ be formed by empirical source representations $\{\widehat{\mathbf{B}}_i\}$ obtained from solving (5.3) and the matrix $\mathbf{V}^* \in \mathbb{R}^{d\times l}$ formed from the true representations $\{\mathbf{B}_i^*\}$. Under Assumption 10-12, for any $\mathbf{b} \in \mathbb{R}^l$ such that $\|\mathbf{b}\|_2 = 1$, $n_s \gg \rho^4(d + \log(m/\delta))$ and $n_{T_1} \gg \rho^4(\max\{l, q\} + \log(1/\delta))$, with probability at-least $1 - \delta_1$, we have:*

$$\min_{\mathbf{u}\in\mathbb{R}^q} \left\|\mathbf{X}_{T_1}\widehat{\mathbf{V}}\mathbf{u} - \mathbf{X}_{T_1}\mathbf{V}^*\mathbf{b}\right\|_2 \leq \frac{\sigma^2 n_{T_1}}{r n_S}\left(km + kdm\log(\kappa n_s) + \log\left(\frac{1}{\delta_1}\right)\right)$$

A proof of the lemma above is provided in Section C.1. We now state our main result for the excess risk on after Phase 1.

Theorem 5 (Phase 1 training result)**.** *Fix a failure probability $\delta \in (0, 1)$ and further assume $2k \leq \min\{d, m\}$ and the number of samples in the sources and target satisfy $n_s \gg \rho^4(d + \log(m/\delta))$ and $n_{T_1} \gg \rho^4(\max\{l, q\} + \log(1/\delta))$, respectively. Define $\kappa = \frac{\max_{i\in[m]}\lambda_{\max}(\Sigma_i)}{\min_{i\in[m]}\lambda_{\min}(\Sigma_i)}$ where $\lambda_{\max}(\Sigma_i)$ denotes the maximum eigenvalue of Σ_i. Then with probability at least $1-\delta$ over the samples, under Assumptions 9 - 13, the expected* *excess risk of* $\boldsymbol{\theta}_{Phase_1} := \widehat{\mathbf{V}}\widehat{\mathbf{w}}$ *satisfies:*

$$\mathbb{E}[EER(\boldsymbol{\theta}_{Phase_1}, \boldsymbol{\theta}_T^*)] \lesssim \frac{\sigma^2}{n_{T_1}}(q + \log(1/\delta)) + \epsilon^2 + \sigma^2 \left[\frac{1}{rn_s m}\log\left(\frac{1}{\delta}\right) + \left(\frac{kd\log(\kappa n_s) + k}{rn_s}\right)\right]$$

where expectation is taken over $\widetilde{\mathbf{w}}_T^*$ for the target task (c.f. Assumption 13). We provide proof for Theorem 5 in Section C.1.

Discussion: The bound in Theorem 5 shows the population risk of the learned model $\boldsymbol{\theta}_{\text{Phase}_1}$ lies in a ball centered at the true target model risk $R(\boldsymbol{\theta}_T^*)$ with radius ϵ^2, which represents the *approximation error* for using source representations for the target task (see Assumption 9). Note that the expected excess risk scales as $\mathcal{O}\left(q/n_{T_1}\right)$ with respect to the number of target samples when the representation is learned from the source representations. This demonstrates a sample gain compared to the baseline of $\mathcal{O}\left(d/n_{T_1}\right)$ for learning the entire model (including representation) with the target data when $q \ll d$, that is, when the empirical source representations together span a subspace of dimension much smaller than d. For the case when source and target representations are all the same, $\mathbf{B}_T^* = \mathbf{B}_i^* = \mathbf{B}^* \in \mathbb{R}^{d\times k}$ for all $i \in [m]$, the excess risk scales as $\mathcal{O}\left(k/n_{T_1}\right)$, which recovers the result of [DHK+21].

5.4.2 Theoretical results for overall scheme: Phase1 + Phase2

We require the following additional assumptions:

Assumption 14. *The rows of the target data matrix* $\mathbf{X}_T$ *are linearly independent.*

Assumption 15. *The Gradient Descent procedure to optimize* (5.6) *converges to*

$\boldsymbol{\theta}_{Phase_2}$ *with* $f(\boldsymbol{\theta}_{Phase_2}) = 0$.

Assumption 14 is typically made in literature for analysis in the over-parameterized regime for linear regression, see [BLLT20], and can also be relaxed to hold with high probability instead and incorporated in the analysis [SBG21]. Assumption 15 holds in our setting as the objective in (5.6) is strongly convex and smooth for which GD can converge to the optimum [BV04].

Theorem 6 (Phase 1 + Phase 2 training result)**.** *Consider obtaining the final target model by using* n_{T_1} *samples during Phase 1 for representation transfer and then fine-tuning in Phase 2 with* n_{T_2} *samples (independently drawn from Phase 1). Denote the eigenvalues of the covariance matrix of the underlying data* Σ_T *by* $\{\lambda_i\}_{i=1}^d$. *Then under Assumptions 9-15, the excess risk of the final parameter* $\hat{\boldsymbol{\theta}}_T := \boldsymbol{\theta}_{Phase_2}$ *is bounded as follows with probability at least* $1-\delta$*:*

$$\mathbb{E}[EER(\hat{\boldsymbol{\theta}}_T, \boldsymbol{\theta}_T^*)] \lesssim \frac{\lambda_1}{\lambda_d}\frac{r_0(\Sigma_T)}{n_{T_2}}\left(\frac{\sigma^2}{n_{T_1}}(q + \log(1/\delta)) + \epsilon^2\right) + r\sigma^2 \log\left(\frac{1}{\delta}\right)\left(\frac{k^*}{n_{T_2}} + \frac{n_{T_2}}{R_{k^*}(\Sigma_T)}\right)$$
$$+ \frac{\lambda_1\sigma^2}{\lambda_d}\frac{r_0(\Sigma_T)}{n_{T_2}}\left(\frac{1}{rn_s m}\log\left(\frac{1}{\delta}\right) + \left(\frac{kd\log(\kappa n_s) + k}{rn_s}\right)\right)$$

where $r_k(\Sigma_T) = \frac{\Sigma_{i>k}\lambda_i}{\lambda_{k+1}}$, $R_{k^*}(\Sigma_T) = \frac{(\Sigma_{i>k}\lambda_i)^2}{\Sigma_{i>k}\lambda_i^2}$. *Here, constant* $b > 1$ *and* $k^* = \min\{k \geq 0 : r_k(\Sigma) \geq bn_{T_2}\}$ *with* $k^* \leq \frac{n_{T_2}}{c_1}$ *for some universal constant* $c_1 > 1$.

We provide a proof for Theorem 6 in Section C.2.

Discussion: Theorem 6 shows the excess risk of our overall target model ($\hat{\boldsymbol{\theta}}_T = \boldsymbol{\theta}_{\text{Phase}_2}$) as a function of the number of samples n_S, n_{T_1}, n_{T_2} and parameters depending on the target data covariance matrix, Σ_T. Since we re-train the entire model (including the representation) with n_{T_2} target samples, the population risk of the learned model $R(\boldsymbol{\theta}_{\mathrm{Phase}_2})$ can be made closer to the true risk $R(\boldsymbol{\theta}_T^*)$ by increasing n_{T_2}, which is in contrast to the result of Theorem 5 which shows closeness only in an ϵ^2 radius ball due to using source representation directly to construct the target model.

We now provide a baseline comparison to the standard linear regression scenario where we do no utilize any source models and instead learn the target task model from scratch using the available $n_T = n_{T_1} + n_{T_2}$ samples. The excess risk in this setting is $\mathcal{O}\left(\frac{\sigma^2 d}{n_T}\right)$. If the number of source samples are large enough ($n_S \gg d$) to get a good empirical performance on the source models (c.f. Equation (5.3)), the bound from Theorem 6 demonstrates a sample gain compared to the baseline when:

$$\frac{\lambda_1}{\lambda_d}\frac{r_0(\Sigma_T)}{n_{T_2}}\left(\frac{\sigma^2}{n_{T_1}}(q+\log(1/\delta))+\epsilon^2\right)+c\sigma^2\log\left(\frac{1}{\delta}\right)\left(\frac{k^*}{n_{T_2}}+\frac{n_{T_2}}{R_{k^*}(\Sigma_T)}\right)\ll\frac{\sigma^2 d}{n_{T_1}+n_{T_2}} \tag{5.8}$$

It can be seen that for the above relation to hold, we require:

- The target data covariance matrix Σ_T should be such that the term $R_{k^*}(\Sigma_T)$ is much larger than n_{T_2}, and $k^* \ll n_{T_2}$. This is satisfied, for e.g., in the case when eigenvalues of Σ_T decay slowly from largest to smallest, and are all larger than a small constant [BLLT20].

- Using the definition of $r_0(\Sigma_T) = \sum_{i=1}^{d} \lambda_i/\lambda_1$, the following provides a sufficient condition the first term on the L.H.S. of (5.8):

$$\frac{q\sum_{i=1}^{d}\lambda_i}{n_{T_1}n_{T_2}\lambda_d} \ll \frac{d}{n_{T_1}+n_{T_2}}$$

This, is turn, imposes the following restriction on q, which is the dimension of the subspace formed by the source representations $\{\widehat{\mathbf{B}}_i\}$:

$$q \ll \frac{d\lambda_d n_{T_1} n_{T_2}}{(\sum_{i=1}^{d} \lambda_i)(n_{T_1} + n_{T_2})} \tag{5.9}$$

Since $n_{T_1} + n_{T_2} = n_T$, it is easy to check that the R.H.S. of (5.9) is maximized when $n_{T_1} = n_{T_2} = {}^{n}\!/_{2}$. With this optimal splitting of the target samples for each of the phases, we require $q \ll \frac{d\lambda_d n_T}{2\sum_{i=1} \lambda_i}$ for the inequality in (5.8).

5.5 Numerical Results

We now provide numerical simulations for our proposed scheme for optimizing linear regression objectives in a data scarce regime. To demonstrate the effectiveness of leveraging pre-trained representations and fine-tuning, we consider the case where we have access to the true representation matrix $\mathbf{V}^*$ formed by the source representations. We compare the performance of models obtained after Phase 1 and Phase 2 training for different parameters of interest and discuss their sample complexity requirements.

5.5.1 Setup

We generate the $d \times q$ matrix $\mathbf{V}^*$ matrix with entries sampled from the standard normal distribution, with $d = 1000$ and $q = 50$. We generate $n_{T_1} \in \{100, 200, 300, 1000\}$ number of samples for the target data for Phase 1 to form the matrix $\mathbf{X}_{T_1}$, which is generated with i.i.d Gaussian entries with mean 0 and covariance matrix Σ_T. To simulate slowly decaying eigenvalues of Σ_T, we set them as $\lambda_j = e^{\frac{-j}{\tau}} + \varepsilon$ for $j \in [d]$, where $\tau = 1$ is the decay factor and $\varepsilon = 0.0001$. The true target model $\boldsymbol{\theta}_T^*$ is generated as $\mathbf{u}+\mathbf{v}$ where $\mathbf{u} \in \mathbb{R}^d$ lies in the span of $\mathbf{V}^*$ and $\mathbf{v}$ is Gaussian vector with covariance matrix $\mathrm{I}\sigma_T^2$ and zero mean. We call the expected ratio of $\mathbf{u}$ to $\mathbf{v}$ as the in-out mixture representation ratio.[2] The target output vector $\mathbf{y}_T$ is generated as $\mathbf{y}_{T_1} = \mathbf{X}_T\boldsymbol{\theta}_T^* + \mathbf{z}_{T_1}$, with $\mathbf{z}_{T_1}$ being a Gaussian noise vector. Phase 1 training thus seeks to minimize the objective in (5.5) (with $\widehat{\mathbf{V}}$ replaced by $\mathbf{V}^*$, since we assume access to complete source representations) and we denote the output model $\boldsymbol{\theta}_{\text{Phase}_1}$. For Phase 2 training, we optimize the objective in (5.6) using new $n_{T_2} \in \{100, 200, 300, 1000\}$ number of target samples with $\boldsymbol{\theta}_{\text{Phase}_1}$ as initialization to obtain the final model $\boldsymbol{\theta}_{\text{Phase}_2}$. We compare the performance of $\boldsymbol{\theta}_{\text{Phase}_1}$ and $\boldsymbol{\theta}_{\text{Phase}_2}$ on 500 test samples generated from the target data. As a baseline, we also consider the performance of the scheme which takes $n_{T_1} + n_{T_2}$ number of target samples and trains the model from scratch, i.e., without leveraging the source representations $\mathbf{V}^*$. We denote the model obtained form this scheme as $\boldsymbol{\theta}_0$.

5.5.2 Results

The results from Phase 1 and Phase 2 training for different splits for the number of phase target samples and ϵ values[3] are shown in Table 5.1. The numerical values, which are averaged over 10 independent runs, denote the ratio of the error obtained by the learned model after the respective phase and the error of the underlying true data generation model $\boldsymbol{\theta}_T^*$ for the target data on a test dataset. In a data-scarce

[2]The parameter σ_T^2 indirectly enables us simulate the value of ϵ in Assumption 9, with larger values of σ_T^2 implying that $\boldsymbol{\theta}_T^*$ lies farther away from the subspace spanned by the columns of $\mathbf{V}^*$. A higher σ_T^2 values thus yields a small value for the in-out representation mixture ratio.

[3]The values in the 'In-Out Representation Mixture Ratio' column in Table 5.1 correspond to the ratio of the signal in the subspace spanned by columns of $\mathbf{V}^*$ and the added out of subspace signal (of variance σ_T^2) added to it to generate the true target model $\boldsymbol{\theta}_T^*$. Lower values of this ratio correspond to higher values of σ_T^2.

Table 5.1 Performance comparison for learned models after Phase 1 (Pre-training), Phase 2 (Fine-tuning) and learning from Scratch.

Sample Configuration	**In-Out Representation Mixture Ratio for $\boldsymbol{\theta}_T^*$ (in dB)**	**Phase 1**	**Phase 2**	**Scratch**
$n_{T_1} = 100, n_{T_2} = 100$	50	1.01	1.01	10.17
	20	3.21	3.05	10.41
	10	9.97	9.17	10.98
	5	16.13	15.84	12.74
	1	28.01	26.61	15.96
$n_{T_1} = 200, n_{T_2} = 200$	50	1.01	1.00	9.71
	20	2.03	1.88	9.33
	10	5.71	5.14	10.33
	5	10.09	9.10	11.79
	1	15.30	13.62	13.68
$n_{T_1} = 300, n_{T_2} = 300$	50	1.01	1.00	8.13
	20	1.90	1.69	8.35
	10	5.30	4.46	9.07
	5	9.12	7.60	9.95
	1	14.55	12.12	12.13
$n_{T_1} = 1000, n_{T_2} = 1000$	50	1.01	1.01	1.01
	20	1.74	1.01	1.01
	10	4.72	1.09	1.01
	5	7.92	1.12	1.01
	1	12.47	1.21	1.01

regime, $(n_{T_1}, n_{T_2}) \in \{(100,100),(200,200),(300,300)\}$, the performance of the model learned from scratch (without leveraging source representations; denoted by column *Scratch*) can be unsatisfactory. As expected, even pre-training (Phase 1) and fine-tuning (Phase 2) in low data regimes does not yield good performance if the source models are not useful for the target data, which is the case when the in-representation mixture to out-mixture ratio for $\boldsymbol{\theta}_T^*$ is small, as shown by the performance for values 5dB, 1dB in Table 5.1. However, utilizing source representations gives significantly better performance relative to training from scratch in cases when $\boldsymbol{\theta}_T^*$ doesn't lie far off from $\mathbf{V}^*$ as can be seen by comparing the values of Phase 2 and Scratch training results for in-out signal mixture ratio of 50dB, 20dB, 10dB. Thus leveraging source representations for target training can be beneficial in scare-data regimes when source representation are useful for the target task and thus representation transfer is practical. It can be seen that training from scratch could perform well for a data-rich regime, $(n_{T_1}, n_{T_2}) = (1000, 1000)$. Here, the performance of the learned model after Phase 1 degrades with decreasing in-out representation mixture ratio as the source representations become less useful to learn $\boldsymbol{\theta}_T^*$. Meanwhile, performing fine-tuning in addition to utilizing source representations, as in Phase 2, yields much better performance of the overall learned model with relative errors much less than after Phase 1. Thus, fine-tuning on target data (Phase 2) can be essential in addition to leveraging source models directly by pre-training (Phase 1) when the true model $\boldsymbol{\theta}_T^*$ lies farther away from the subspace spanned by the source representations.

5.6 Conclusion

In this chapter, we proposed a method for training linear regression models via representation transfer learning in the limited sample regime, when given access to multiple pre-trained linear models trained on data domains (sources) different form the target of interest. We established excess risk bounds for the learned target model when (i) source representations are used directly to construct a target representation and adapted to the target task, and (ii) when the entire resulting model is fine-tuned in the over-parameterized regime using target task samples. Our bounds showed a gain in target sample complexity compared to the baseline case of learning without access to the pre-trained models, thus demonstrating the benefit of transfer learning for better generalization in the limited sample regime. Our provided numerical results corroborated this fact and showed superior performance of our proposed scheme compared to learning from scratch in data-scare regimes.

As future extensions to this work, it is of interest to see how non-linear activation functions can be introduced in the model to analyze more complicated architectures like Neural Networks (NNs). Analyzing representation transfer learning with multiple NNs and utilizing recently developed results in benign over-fitting for this setting [CLB22] is an interesting next step. In many scenarios of interest, for training the source task models, *unlabeled* data from the target distribution might be available. While there are empirical works utilizing unlabeled samples in the context of semi-supervised adaptation [Sin21, MSS, ZLLJ19], theoretical results on understanding generalization of representation transfer learning methods (with pre-training/fine-tuning) and their sample complexity requirements are missing and would be an interesting direction to pursue. It is also of interest to understand and construct schemes that can utilize unlabeled target data available during the source training phase, in context of *semi-supervised learning* and understand the generalization properties and sample complexity of the said schemes.

CHAPTER 6

Discussion and Future Direction

In this dissertation, we have systematically addressed the critical challenges in federated learning, specifically focusing on communication efficiency, data heterogeneity, and data scarcity. Through the development of novel methodologies and rigorous theoretical analysis, we have introduced significant advancements that enhance the robustness, scalability, and practical deployment of federated learning systems. These contributions collectively form a comprehensive framework for improving the performance and applicability of federated learning in diverse, real-world environments.

In the following sections, we summarize each of our contributions and then discuss potential future extensions for these works:

6.1 Communication Efficient Learning

Our work on communication-efficient decentralized training, detailed in Chapter 3, focused on developing novel compressed stochastic gradient descent (SGD) algorithms for efficient learning. These algorithms leverage event-triggered communication, local iterations, and compression techniques to significantly reduce the frequency and volume of data exchanges between devices, thereby minimizing communication overhead. Our theoretical analysis established convergence guarantees across different regimes, including strongly convex, convex, and non-convex loss functions. We demonstrated that these methods achieve a convergence rate comparable to that of vanilla SGD, while substantially reducing communication costs. This work highlights the critical importance of optimizing communication protocols to enable scalable federated learning, particularly in bandwidth-constrained environments.

6.2 Multi-task Learning

To address the challenge of data heterogeneity, Chapter 4 explores multi-task learning within decentralized federated learning frameworks. These techniques facilitate the simultaneous optimization of multiple related tasks, enabling the creation of personalized models that cater to the unique data distributions and objectives of individual devices. By incorporating communication-efficient strategies, such as gradient compression and decentralized updates, we ensured that the training process remains both scalable and efficient. Our contributions include formulating the multi-task learning problem in decentralized settings for convex objectives and proposing a communication efficient primal-dual algorithm for optimizing these objectives while satisfying proximity constraints across devices to promote personalization. We derived convergence rates for our proposed communication-efficient algorithm, demonstrating that it achieves rates comparable to those of vanilla SGD in both *sample feedback* and *bandit feedback* scenarios, while showing significant improvements in communication efficiency through our empirical results.

6.3 Representation Transfer Learning

Chapter 5 addresses the challenge of data scarcity, where we consider transfer learning methods with a specific focus on linear models. By utilizing pre-trained regression models from diverse source domains, we provided robust initializations for target models, which were then fine-tuned using limited local data. This approach significantly enhances model performance and adaptability in data-scarce settings. We employed techniques such as representation transfer and fine-tuning to ensure that the transferred knowledge is effectively utilized in the target domain. Theoretical guarantees on the excess risk bounds for these transfer learning methods were provided, ensuring their reliability and effectiveness. This demonstrates the potential of transfer learning to mitigate the limitations posed by insufficient local data in federated learning systems.

6.4 Future Work

Developing a unified framework that seamlessly combines the various facets of our work – communication-efficient decentralized training, multi-task learning, and transfer learning – could provide a more holistic solution to the challenges of federated learning and presents numerous opportunities for future research.

Our work on communication efficient multi-task learning techniques have shown promise in providing task oriented models in an efficient manner; however, there is potential for further refinement. Future research could investigate more sophisticated models for capturing the relationships between tasks, such as using deep neural networks for task embedding, and thus extending the analysis to general objectives.

Additionally, developing adaptive multi-task learning algorithms that can dynamically adjust to changes in data distribution and task requirements would be valuable.

Our transfer learning methods have demonstrated effectiveness in mitigating data scarcity, but there are several areas for further exploration. Extending these techniques to more complex models, such as deep neural networks, could improve their applicability to a wider range of tasks. Additionally, investigating methods for selecting the most relevant pre-trained models for transfer, based on the target domain characteristics, would enhance the effectiveness of transfer learning. Integrating our proposed methods into a federated environment with multiple devices—where each device serves as both a potential target and a source for transfer learning—presents a compelling area for future research. This would enable a more dynamic and collaborative approach to mitigating data scarcity in federated learning systems.

Appendix A

Omitted Details for Chapter 3

A.1 Preliminaries for Convergence with Relaxed Assumptions

Proof of Proposition 1. This simply follows from the independence of the randomness used in sampling stochastic gradients at different workers. $\square$

Proof of Proposition 2. We want to show the following bound on $\mathbb{E}\left\|\mathbf{V}^{(t)}\right\|_F^2$ for any t:

$$\mathbb{E}\left\|\mathbf{V}^{(t)}\right\|_F^2 \leq \frac{1}{(1-\beta)}\sum_{k=0}^{t}\beta^{t-k}\mathbb{E}\left\|\nabla\mathbf{F}(\mathbf{X}^{(k)},\boldsymbol{\xi}^{(k)})\right\|_F^2.$$

For any t, let $\theta_t = \sum_{k=0}^{t}\beta^{t-k}$.

$$\begin{aligned}
\mathbb{E}\left\|\mathbf{V}^{(t)}\right\|_F^2 &= \mathbb{E}\left\|\sum_{k=0}^{t}\beta^{t-k}\nabla\mathbf{F}(\mathbf{X}^{(k)},\boldsymbol{\xi}^{(k)})\right\|_F^2 \\
&= \theta_t^2\mathbb{E}\left\|\sum_{k=0}^{t}\frac{\beta^{t-k}}{\theta_t}\nabla\mathbf{F}(\mathbf{X}^{(k)},\boldsymbol{\xi}^{(k)})\right\|_F^2 \\
&\leq \theta_t\sum_{k=0}^{t}\beta^{t-k}\mathbb{E}\left\|\nabla\mathbf{F}(\mathbf{X}^{(k)},\boldsymbol{\xi}^{(k)})\right\|_F^2 \\
&\leq \frac{1}{1-\beta}\sum_{k=0}^{t}\beta^{t-k}\mathbb{E}\left\|\nabla\mathbf{F}(\mathbf{X}^{(k)},\boldsymbol{\xi}^{(k)})\right\|_F^2. \qquad \text{(A.1)}
\end{aligned}$$

□

A.2 Preliminaries for Convergence with Relaxed Assumptions

Fact 2. *Consider the variance bound on the stochastic gradient for nodes $i \in [n]$:*

$$\mathbb{E}_{\xi_i} \left\| \nabla F_i(\mathbf{x}, \xi_i) - \nabla f_i(\mathbf{x}) \right\|^2 \leq \sigma_i^2,$$

where $\mathbb{E}_{\xi_i}[\nabla F_i(\mathbf{x}, \xi_i)] = \nabla f_i(\mathbf{x})$, then:

$$\mathbb{E}_{\boldsymbol{\xi}^{(t)}} \left\| \frac{1}{n} \sum_{j=1}^{n} \left(\nabla f_j(\mathbf{x}_j^{(t)}) - \nabla F_j(\mathbf{x}_j^{(t)}, \xi_j^{(t)}) \right) \right\|^2 \leq \frac{\bar{\sigma}^2}{n} \tag{A.2}$$

where $\boldsymbol{\xi}^{(t)} = \{\xi_1^{(t)}, \xi_2^{(t)}, \ldots, \xi_n^{(t)}\}$ denotes the stochastic sample for the nodes at any timestep t and $\frac{\sum_{j=1}^{n} \sigma_j^2}{n} = \bar{\sigma}^2$

Proof.

$$\begin{aligned}\mathbb{E}_{\xi^{(t)}} \left\| \frac{1}{n} \sum_{j=1}^{n} \nabla f_j(\mathbf{x}_j^{(t)}) - \frac{1}{n} \sum_{j=1}^{n} \nabla F_j(\mathbf{x}_j^{(t)}, \xi_j^{(t)}) \right\|^2 &= \frac{1}{n^2} \sum_{j=1}^{n} \mathbb{E}_{\xi^{(t)}} \| \nabla f_j(\mathbf{x}_j^{(t)}) - \nabla F_j(\mathbf{x}_j^{(t)}, \xi_j^{(t)}) \|^2 \\ &+ \frac{1}{n^2} \sum_{i \neq j} \mathbb{E}_{\xi^{(t)}} \left\langle \nabla f_i(\mathbf{x}_i^{(t)}) - \nabla F_i(\mathbf{x}_i^{(t)}, \xi_j^{(t)}), \nabla f_j(\mathbf{x}_j^{(t)}) - \nabla F_j(\mathbf{x}_j^{(t)}, \xi_j^{(t)}) \right\rangle\end{aligned}$$

Since ξ_i is independent of ξ_j, the second term is zero in expectation, thus the above reduces to:

$$\mathbb{E}_{\xi^{(t)}} \left\| \frac{1}{n} \sum_{j=1}^{n} \nabla f_j(\mathbf{x}_j^{(t)}) - \frac{1}{n} \sum_{j=1}^{n} \nabla F_j(\mathbf{x}_j^{(t)}, \xi_j^{(t)}) \right\|^2 = \frac{1}{n^2} \sum_{j=1}^{n} \mathbb{E}_{\xi^{(t)}} \| \nabla f_j(\mathbf{x}_j^{(t)}) - \nabla F_j(\mathbf{x}_j^{(t)}, \xi_j^{(t)}) \|^2$$

$$\leq \frac{1}{n^2}\sum_{j=1}^{n}\sigma_j^2 = \frac{\bar{\sigma}^2}{n}$$

$\square$

Fact 3. *Consider the set of synchronization indices* $\{I_{(1)}, I_{(2)}, \ldots, I_{(k)}, \ldots\} \in \mathcal{I}_T$*. We assume that the maximum gap between any two consecitive elements in* $\mathcal{I}_T$ *is bounded by* H*. Let* $\xi^{(t)} = \{\xi_1^{(t)}, \xi_2^{(t)}, \ldots, \xi_n^{(t)}\}$ *denote the stochastic samples for the nodes at any timestep* t*. Consider any two consecutive synchronization indices* $I_{(k)}$ *and* $I_{(k+1)}$*, then for learning rate* η*, we have:*

$$\mathbb{E}\left[\left\|\sum_{t'=I_{(k)}}^{I_{(k+1)}-1}\eta(\beta\mathbf{V}^{(t')} + \boldsymbol{\nabla F}(\mathbf{X}^{(t')}, \boldsymbol{\xi}^{(t')}))\right\|_F^2\right] \leq 2nH^2G^2\eta^2\left(1 + \frac{\beta^2}{(1-\beta)^2}\right). \quad \text{(A.3)}$$

Proof. Using the fact that the sequence gap is bounded by H, we have $I_{(t+1)} - I_{(t)} \leq H$ for all synchronization indices $I_{(t)} \in \mathcal{I}_T$. Thus we have:

$$\begin{aligned}\mathbb{E}\left[\left\|\sum_{t'=I_{(k)}}^{I_{(k+1)}-1}\eta(\beta\mathbf{V}^{(t')} + \boldsymbol{\nabla F}(\mathbf{X}^{(t')}, \boldsymbol{\xi}^{(t')}))\right\|_F^2\right] &\leq H\eta^2\sum_{t'=I_{(k)}}^{I_{(k+1)}-1}\mathbb{E}\left\|\beta\mathbf{V}^{(t')} + \boldsymbol{\nabla F}(\mathbf{X}^{(t')}, \boldsymbol{\xi}^{(t')})\right\|_F^2 \\ &\leq 2H\eta^2\sum_{t'=I_{(k)}}^{I_{(k+1)}-1}\left[\mathbb{E}\left\|\beta\mathbf{V}^{(t')}\right\|_F^2 + \mathbb{E}\left\|\boldsymbol{\nabla F}(\mathbf{X}^{(t')}, \boldsymbol{\xi}^{(t')})\right\|_F^2\right]\end{aligned}$$

Using the bounded gradient assumption and definition of gap H, we can bound the above as:

$$\mathbb{E}\left[\left\|\sum_{t'=I_{(k)}}^{I_{(k+1)}-1}\eta(\beta\mathbf{V}^{(t')} + \boldsymbol{\nabla F}(\mathbf{X}^{(t')}, \boldsymbol{\xi}^{(t')}))\right\|_F^2\right] \leq 2H\eta^2\beta^2\sum_{t'=I_{(k)}}^{I_{(k+1)}-1}\mathbb{E}\left\|\mathbf{V}^{(t')}\right\|_F^2 + 2nH^2G^2\eta^2$$

$$=2H\eta^2\beta^2 \sum_{t'=I_{(k)}}^{I_{(k+1)}-1} \sum_{i=1}^{n} \mathbb{E}\left\|\mathbf{v}_i^{(t')}\right\|^2 + 2nH^2G^2\eta^2 \tag{A.4}$$

Now we show that $\mathbb{E}\left\|\mathbf{v}_i^{(t)}\right\|^2 \leq \frac{G^2}{(1-\beta)^2}$ for all $i \in [n]$ and for every $t \geq 0$. Fix an arbitrary $i \in [n]$ and $t \geq 0$. Define $\theta_t = \sum_{k=0}^{t} \beta^k$, we then have:

$$\begin{aligned}
\mathbb{E}\left\|\mathbf{v}_i^{(t)}\right\|^2 &= \theta_t^2 \mathbb{E}\left\|\sum_{k=0}^{t} \frac{\beta^{t-k}}{\theta_t} \nabla F(\mathbf{x}_i^{(k)}, \xi_i^{(k)})\right\|^2 \\
&\leq \theta_t \sum_{k=0}^{t} \beta^{t-k} \mathbb{E}\left\|\nabla F(\mathbf{x}_i^{(k)}, \xi_i^{(k)})\right\|^2 \\
&\leq \theta_t \sum_{k=0}^{t} \left[\beta^{t-k} G^2\right] \\
&= G^2 \theta_t^2
\end{aligned}$$

Here the first inequality follows from the Jensen's inequality and the second inequality follows from the bounded gradient assumption. We now note the following bound for θ_t:

$$\theta_t = \sum_{k=0}^{t} \beta^k \leq \sum_{k=0}^{\infty} \beta^k \leq \frac{1}{(1-\beta)}$$

Thus, for all t and all $i \in [n]$, we have:

$$\mathbb{E}\left\|\mathbf{v}_i^{(t)}\right\|^2 \leq \frac{G^2}{(1-\beta)^2} \tag{A.5}$$

Substituting the bound $\mathbb{E}\|\mathbf{v}_i^{(t)}\|^2 \le \frac{G^2}{(1-\beta)^2}$ in (A.4) gives

$$\mathbb{E}\left[\left\|\sum_{t'=I_{(k)}}^{I_{(k+1)}-1} \eta(\beta\mathbf{V}^{(t')} + \nabla \boldsymbol{F}(\mathbf{X}^{(t')}, \boldsymbol{\xi}^{(t')}))\right\|_F^2\right] \le 2H^2\eta^2\beta^2 n\frac{G^2}{(1-\beta)^2} + 2nH^2G^2\eta^2.$$

This completes the proof of Fact 3. □

Fact 4 (Triggering rule, [SDGD19])**.** *Consider the set of nodes $\Gamma^{(t)}$ which do not communicate at time t. For a threshold sequence $\{c_t\}_{t=0}^{T-1}$, the triggering rule in Algorithm 1 dictates that*

$$\|\mathbf{x}_i^{(t+\frac{1}{2})} - \hat{\mathbf{x}}_i^{(t)}\|^2 \le c_t\eta^2 \qquad \forall i \in \Gamma^{(t)}.$$

Using the matrix notation, this implies that:

$$\left\|(\mathbf{X}^{(t+\frac{1}{2})} - \hat{\mathbf{X}}^{(t)})(\mathbf{I} - \mathbf{P}^{(t)})\right\|_F^2 \le nc_t\eta^2. \tag{A.6}$$

Fact 5 (Lemma 16, [KSJ19b])**.** *For doubly stochastic matrix $\mathbf{W}$ with second largest eigenvalue $1-\delta = |\lambda_2(\mathbf{W})| < 1$, we have:*

$$\left\|\mathbf{W}^k - \frac{1}{n}\mathbf{1}\mathbf{1}^T\right\| = (1-\delta)^k \tag{A.7}$$

for any non-negative integer k.

Claim 1. *For any $n \in \mathbb{N}$, we have $\left\|\frac{\mathbf{1}\mathbf{1}^T}{n} - \mathbf{I}\right\|_2 = 1$ where $\mathbf{1} = [1\,1\ldots1]_{1\times n}^T$*

Proof. Note that $\frac{\mathbf{1}\mathbf{1}^T}{n}$ is a symmetric doubly stochastic matrix with eigenvalues 1 and 0 (with algebraic multiplicity $n-1$). Thus, it has the eigen-decomposition

$\frac{\mathbf{1}\mathbf{1}^T}{n} = \mathbf{U}\mathbf{D}\mathbf{U}^T$ where columns of $\mathbf{U}$ are orthogonal and $\mathbf{D} = diag([1\,0\ldots0])$, which gives us:

$$\left\|\frac{\mathbf{1}\mathbf{1}^T}{n} - \mathbf{I}\right\|_2 = \left\|\mathbf{U}\mathbf{D}\mathbf{U}^T - \mathbf{U}\mathbf{U}^T\right\|_2 = \|\mathbf{D}-\mathbf{I}\|_2 = \left\|\begin{bmatrix}1 & 0 & \ldots & 0\\ 0 & 0 & \ldots & 0\\ \vdots & \vdots & \ddots & 0\\ 0 & 0 & \ldots & 0\end{bmatrix} - \begin{bmatrix}1 & 0 & \ldots & 0\\ 0 & 1 & \ldots & 0\\ \vdots & \vdots & \ddots & 0\\ 0 & 0 & \ldots & 1\end{bmatrix}\right\|_2 = 1$$

□

A.3 Proof of Theorem 2 (Non-convex objective)

From the recurrence relation of the virtual sequence, we have:

$$\begin{aligned}
\mathbb{E}_{\xi_{(t)}}[f(\widetilde{\mathbf{x}}^{(t+1)})] &= \mathbb{E}_{\xi_{(t)}} f\left(\widetilde{\mathbf{x}}^{(t)} - \frac{\eta}{(1-\beta)}\frac{1}{n}\sum_{i=1}^{n}\nabla F_i(\mathbf{x}_i^{(t)}, \xi_i^{(t)})\right)\\
&\leq f(\widetilde{\mathbf{x}}^{(t)}) - \left\langle \nabla f(\widetilde{\mathbf{x}}^{(t)}), \frac{\eta}{(1-\beta)}\frac{1}{n}\sum_{i=1}^{n}\mathbb{E}_{\xi_{(t)}}[\nabla F_i(\mathbf{x}_i^{(t)}, \xi_i^{(t)})]\right\rangle\\
&\quad + \frac{L}{2}\frac{\eta^2}{(1-\beta)^2}\mathbb{E}_{\xi_{(t)}}\left\|\frac{1}{n}\sum_{i=1}^{n}\nabla F_i(\mathbf{x}_i^{(t)}, \xi_i^{(t)})\right\|^2\\
&\leq f(\widetilde{\mathbf{x}}^{(t)}) - \left\langle \nabla f(\widetilde{\mathbf{x}}^{(t)}), \frac{\eta}{(1-\beta)}\frac{1}{n}\sum_{i=1}^{n}\nabla f_i(\mathbf{x}_i^{(t)})\right\rangle + \frac{L}{2}\frac{\eta^2}{(1-\beta)^2}\left\|\frac{1}{n}\sum_{i=1}^{n}\nabla f_i(\mathbf{x}_i^{(t)})\right\|^2\\
&\quad + \frac{L}{2}\frac{\eta^2}{(1-\beta)^2}\mathbb{E}_{\xi_{(t)}}\left\|\frac{1}{n}\sum_{i=1}^{n}(\nabla f_i(\mathbf{x}_i^{(t)}) - \nabla F_i(\mathbf{x}_i^{(t)}, \xi_i^{(t)})\right\|^2\\
&\leq f(\widetilde{\mathbf{x}}^{(t)}) - \left\langle \nabla f(\widetilde{\mathbf{x}}^{(t)}), \frac{\eta}{(1-\beta)}\frac{1}{n}\sum_{i=1}^{n}\nabla f_i(\mathbf{x}_i^{(t)})\right\rangle + \frac{L}{2}\frac{\eta^2}{(1-\beta)^2}\left\|\frac{1}{n}\sum_{i=1}^{n}\nabla f_i(\mathbf{x}_i^{(t)})\right\|^2\\
&\quad + \frac{L\eta^2\bar{\sigma}^2}{2n(1-\beta)^2} \qquad\qquad (A.8)
\end{aligned}$$

We now focus on bounding the second term in (A.8). First, note the following:

$$\begin{aligned}\left\langle \nabla f(\widetilde{\mathbf{x}}^{(t)}), \frac{1}{n}\sum_{i=1}^{n}\nabla f_i(\mathbf{x}_i^{(t)})\right\rangle &= \left\|\frac{1}{n}\sum_{i=1}^{n}\nabla f_i(\mathbf{x}_i^{(t)})\right\|^2 - \left\langle \frac{1}{n}\sum_{i=1}^{n}\nabla f_i(\mathbf{x}_i^{(t)}) - \nabla f(\widetilde{\mathbf{x}}^{(t)}), \frac{1}{n}\sum_{i=1}^{n}\nabla f_i(\mathbf{x}_i^{(t)})\right\rangle \\ &= \left\|\frac{1}{n}\sum_{i=1}^{n}\nabla f_i(\mathbf{x}_i^{(t)})\right\|^2 - \left\langle \frac{1}{n}\sum_{i=1}^{n}(\nabla f_i(\mathbf{x}_i^{(t)}) - \nabla f_i(\widetilde{\mathbf{x}}^{(t)})), \frac{1}{n}\sum_{i=1}^{n}\nabla f_i(\mathbf{x}_i^{(t)})\right\rangle \\ &\geq \frac{1}{2}\left\|\frac{1}{n}\sum_{i=1}^{n}\nabla f_i(\mathbf{x}_i^{(t)})\right\|^2 - \frac{L^2}{2n}\sum_{i=1}^{n}\left\|\mathbf{x}_i^{(t)} - \widetilde{\mathbf{x}}^{(t)}\right\|^2 \qquad \text{(A.9)}\end{aligned}$$

where in the last inequality, we've used the fact that $2\langle \mathbf{a}, \mathbf{b}\rangle \leq \|\mathbf{a}\|^2 + \|\mathbf{b}\|^2$ for any $\mathbf{a}, \mathbf{b} \in \mathbb{R}^d$ and the $L-$smoothness assumption for objectives $\{f_i\}_{i=1}^n$. We now state how to bound the last term on R.H.S. of (A.9). First, note the bound:

$$\sum_{i=1}^{n}\left\|\mathbf{x}_i^{(t)} - \widetilde{\mathbf{x}}^{(t)}\right\|^2 \leq 2\sum_{i=1}^{n}\left\|\mathbf{x}_i^{(t)} - \overline{\mathbf{x}}^{(t)}\right\|^2 + 2\sum_{i=1}^{n}\left\|\overline{\mathbf{x}}^{(t)} - \widetilde{\mathbf{x}}^{(t)}\right\|^2 \qquad \text{(A.10)}$$

We bound the second term in (A.10) as:

$$\sum_{i=1}^{n}\left\|\mathbf{x}_i^{(t)} - \widetilde{\mathbf{x}}^{(t)}\right\|^2 \leq 2\sum_{i=1}^{n}\left\|\mathbf{x}_i^{(t)} - \overline{\mathbf{x}}^{(t)}\right\|^2 + \frac{2n\beta^4\eta^2}{(1-\beta)^3}\sum_{\tau=0}^{t-1}\left[\beta^{t-\tau-1}\left\|\frac{1}{n}\sum_{i=1}^{n}\nabla F_i(\mathbf{x}_i^{(\tau)}, \xi_i^{(\tau)})\right\|^2\right] \qquad \text{(A.11)}$$

Using the bound (A.11) in (A.9) and substituting it in (A.8), we have the following bound:

$$\begin{aligned}\mathbb{E}_{\xi_{(t)}}[f(\widetilde{\mathbf{x}}^{(t+1)})] \leq f(\widetilde{\mathbf{x}}^{(t)}) &+ \frac{L\eta^2\bar{\sigma}^2}{2n(1-\beta)^2} + \frac{L\eta^2}{2(1-\beta)^2}\left\|\frac{1}{n}\sum_{i=1}^{n}\nabla f_i(\mathbf{x}_i^{(t)})\right\|^2 - \frac{\eta}{2(1-\beta)}\left\|\frac{1}{n}\sum_{i=1}^{n}\nabla f_i(\mathbf{x}_i^{(t)})\right\|^2 \\ &+ \frac{\eta}{(1-\beta)}\frac{L^2}{n}\sum_{i=1}^{n}\left\|\mathbf{x}_i^{(t)} - \overline{\mathbf{x}}^{(t)}\right\|^2 + \frac{L^2\eta^3\beta^4}{(1-\beta)^4}\sum_{\tau=0}^{t-1}\left[\beta^{t-\tau-1}\mathbb{E}_{\xi_{(t)}}\left\|\frac{1}{n}\sum_{i=1}^{n}\nabla F_i(\mathbf{x}_i^{(\tau)}, \xi_i^{(\tau)})\right\|^2\right]\end{aligned}$$

Rearranging the terms, we can write:

$$\left(\frac{\eta}{2(1-\beta)} - \frac{L\eta^2}{2(1-\beta)^2}\right)\left\|\frac{1}{n}\sum_{i=1}^{n}\nabla f_i(\mathbf{x}_i^{(t)})\right\|^2 \leq f(\widetilde{\mathbf{x}}^{(t)}) - \mathbb{E}_{\xi_{(t)}} f(\widetilde{\mathbf{x}}^{(t+1)}) + \frac{L\eta^2\bar{\sigma}^2}{2n(1-\beta)^2}$$
$$+ \frac{L^2\eta}{(1-\beta)n}\sum_{i=1}^{n}\left\|\mathbf{x}_i^{(t)} - \overline{\mathbf{x}}^{(t)}\right\|^2 + \frac{L^2\eta^3\beta^4}{(1-\beta)^4}\sum_{\tau=0}^{t-1}\left[\beta^{t-\tau-1}\mathbb{E}_{\xi_{(t)}}\left\|\frac{1}{n}\sum_{i=1}^{n}\nabla F_i(\mathbf{x}_i^{(\tau)}, \xi_i^{(\tau)})\right\|^2\right]$$

Summing from $t = 0$ to T gives us:

$$\left(\frac{\eta}{2(1-\beta)} - \frac{L\eta^2}{2(1-\beta)^2}\right)\sum_{t=0}^{T-1}\left\|\frac{1}{n}\sum_{i=1}^{n}\nabla f_i(\mathbf{x}_i^{(t)})\right\|^2$$
$$\leq f(\widetilde{\mathbf{x}}^{(0)}) - \mathbb{E}_{\xi_{(t)}} f(\widetilde{\mathbf{x}}^{(T)}) + \frac{L\eta^2\bar{\sigma}^2 T}{2n(1-\beta)^2} + \frac{L^2\eta}{(1-\beta)n}\sum_{t=0}^{T-1}\sum_{i=1}^{n}\mathbb{E}\left\|\mathbf{x}_i^{(t)} - \overline{\mathbf{x}}^{(t)}\right\|^2$$
$$+ \frac{L^2\eta^3\beta^4}{(1-\beta)^4}\sum_{t=0}^{T-1}\sum_{\tau=0}^{t-1}\left[\beta^{t-\tau-1}\mathbb{E}_{\xi_{(t)}}\left\|\frac{1}{n}\sum_{i=1}^{n}\nabla F_i(\mathbf{x}_i^{(\tau)}, \xi_i^{(\tau)})\right\|^2\right]$$

Using the fact that $\mathbb{E}_{\xi_{(t)}}[\nabla F_i(\mathbf{x}_i^{(t)}, \xi_i^{(t)})] = \nabla f_i(\mathbf{x}_i^{(t)})$ for all $i \in [n]$ and for all $t \in [T]$, we have $\mathbb{E}_{\xi_{(t)}}\left\|\frac{1}{n}\sum_{i=1}^{n}\nabla F_i(\mathbf{x}_i^{(t)}, \xi_i^{(t)})\right\|^2 = \mathbb{E}_{\xi_{(t)}}\left\|\frac{1}{n}\sum_{i=1}^{n}\nabla f_i(\mathbf{x}_i^{(t)})\right\|^2 + \mathbb{E}_{\xi_{(t)}}\left\|\frac{1}{n}\sum_{i=1}^{n}(\nabla f_i(\mathbf{x}^{(t)}) - \nabla F_i(\mathbf{x}_i^{(t)}, \xi_i^{(t)}))\right\|^2$. Using this equation along with the variance bound (A.2) from Fact 2, the fact that $\sum_{t=0}^{T-1}\sum_{\tau=0}^{t-1}\beta^{t-\tau-1} \leq {}^{T}\!/_{1-\beta}$ for $\beta \in (0,1)$ and taking expectation w.r.t. the entire process:

$$\leq f(\widetilde{\mathbf{x}}^{(0)}) - \mathbb{E} f(\widetilde{\mathbf{x}}^{(T)}) + \frac{L\eta^2\bar{\sigma}^2 T}{2n(1-\beta)^2} + \frac{L^2\eta}{(1-\beta)n}\sum_{t=0}^{T-1}\sum_{i=1}^{n}\mathbb{E}\left\|\mathbf{x}_i^{(t)} - \overline{\mathbf{x}}^{(t)}\right\|^2$$
$$+ \frac{L^2\eta^3\beta^4\bar{\sigma}^2 T}{n(1-\beta)^5} + \frac{L^2\eta^3\beta^4}{(1-\beta)^4}\sum_{t=0}^{T-1}\sum_{\tau=0}^{t-1}\left[\beta^{t-\tau-1}\mathbb{E}\left\|\frac{1}{n}\sum_{i=1}^{n}\nabla f_i(\mathbf{x}_i^{(\tau)})\right\|^2\right] \tag{A.12}$$

To bound the last term in (A.12), we note that:

$$\begin{aligned}\sum_{t=0}^{T-1}\sum_{\tau=0}^{t-1}\beta^{t-\tau-1}\mathbb{E}\left\|\frac{1}{n}\sum_{i=1}^{n}\nabla f_i(\mathbf{x}_i^{(\tau)})\right\|^2 &= \sum_{\tau=0}^{T-2}\sum_{t=\tau+1}^{T-1}\beta^{t-\tau-1}\mathbb{E}\left\|\frac{1}{n}\sum_{i=1}^{n}\nabla f_i(\mathbf{x}_i^{(\tau)})\right\|^2\\ &\le \frac{1}{(1-\beta)}\sum_{\tau=0}^{T-2}\mathbb{E}\left\|\frac{1}{n}\sum_{i=1}^{n}\nabla f_i(\mathbf{x}_i^{(\tau)})\right\|^2\\ &\le \frac{1}{(1-\beta)}\sum_{t=0}^{T-1}\mathbb{E}\left\|\frac{1}{n}\sum_{i=1}^{n}\nabla f_i(\mathbf{x}_i^{(t)})\right\|^2\end{aligned}$$

Substituting the above bound in (A.12) and rearranging terms, we finally get:

$$\begin{aligned}&\left(\frac{\eta}{2(1-\beta)}-\frac{L\eta^2}{2(1-\beta)^2}-\frac{L^2\eta^3\beta^4}{(1-\beta)^5}\right)\sum_{t=0}^{T-1}\mathbb{E}\left\|\frac{1}{n}\sum_{i=1}^{n}\nabla f_i(\mathbf{x}_i^{(t)})\right\|^2\\ &\quad\le f(\widetilde{\mathbf{x}}^{(0)})-\mathbb{E}f(\widetilde{\mathbf{x}}^{(T)})+\frac{L\eta^2\bar{\sigma}^2T}{2n(1-\beta)^2}+\frac{L^2\eta}{(1-\beta)n}\sum_{t=0}^{T-1}\sum_{i=1}^{n}\mathbb{E}\left\|\mathbf{x}_i^{(t)}-\overline{\mathbf{x}}^{(t)}\right\|^2+\frac{L^2\eta^3\beta^4\bar{\sigma}^2T}{n(1-\beta)^5}\end{aligned} \tag{A.13}$$

If we select $\eta \le \min\left\{\frac{(1-\beta)}{4L}, \frac{(1-\beta)^2}{2\sqrt{2}L\beta^2}\right\}$, it can be shown that $\left(\frac{\eta}{2(1-\beta)}-\frac{L\eta^2}{2(1-\beta)^2}-\frac{L^2\eta^3\beta^4}{(1-\beta)^5}\right) \ge \frac{\eta}{4(1-\beta)}$. This gives:

$$\begin{aligned}\frac{\eta}{4(1-\beta)}\sum_{t=0}^{T-1}\mathbb{E}\left\|\frac{1}{n}\sum_{i=1}^{n}\nabla f_i(\mathbf{x}_i^{(t)})\right\|^2 &\le f(\widetilde{\mathbf{x}}^{(0)})-\mathbb{E}[f(\widetilde{\mathbf{x}}^{(T)})]+\frac{L\eta^2\bar{\sigma}^2T}{2n(1-\beta)^2}++\frac{L^2\eta^3\beta^4\bar{\sigma}^2T}{n(1-\beta)^5}\\ &\quad+\frac{L^2\eta}{(1-\beta)n}\sum_{t=0}^{T-1}\sum_{i=1}^{n}\mathbb{E}\left\|\mathbf{x}_i^{(t)}-\overline{\mathbf{x}}^{(t)}\right\|^2\end{aligned}$$

Multiplying both sides by $\frac{4(1-\beta)}{\eta T}$ and noting that $\mathbb{E}[f(\widetilde{\mathbf{x}}^{(T)})] \ge f^*$, we have:

$$\begin{aligned}\frac{1}{T}\sum_{t=0}^{T-1}\mathbb{E}\left\|\frac{1}{n}\sum_{i=1}^{n}\nabla f_i(\mathbf{x}_i^{(t)})\right\|^2 &\le \frac{4(1-\beta)}{\eta}\frac{(f(\mathbf{x}^{(0)})-f^*)}{T}+\frac{2L\eta\bar{\sigma}^2}{n(1-\beta)}\\ &\quad+\frac{4L^2}{nT}\sum_{t=0}^{T-1}\sum_{i=1}^{n}\mathbb{E}\left\|\mathbf{x}_i^{(t)}-\overline{\mathbf{x}}^{(t)}\right\|^2+\frac{4L^2\eta^2\beta^4\bar{\sigma}^2}{n(1-\beta)^4}\end{aligned} \tag{A.14}$$

Now consider the time average of gradients evaluated at the global average $\overline{\mathbf{x}}^{(t)}$:

$$\begin{aligned}\frac{1}{T}\sum_{t=0}^{T-1}\mathbb{E}\left\|\nabla f(\overline{\mathbf{x}}^{(t)})\right\|^2 &= \frac{1}{T}\sum_{t=0}^{T-1}\mathbb{E}\left\|\frac{1}{n}\sum_{i=1}^{n}\nabla f_i(\overline{\mathbf{x}}^{(t)})\right\|^2\\ &= \frac{1}{T}\sum_{t=0}^{T-1}\mathbb{E}\left\|\frac{1}{n}\sum_{i=1}^{n}(\nabla f_i(\overline{\mathbf{x}}^{(t)})-\nabla f_i(\mathbf{x}_i^{(t)}))+\frac{1}{n}\sum_{i=1}^{n}\nabla f_i(\mathbf{x}_i^{(t)})\right\|^2\\ &\le \frac{2}{T}\sum_{t=0}^{T-1}\mathbb{E}\left\|\frac{1}{n}\sum_{i=1}^{n}(\nabla f_i(\overline{\mathbf{x}}^{(t)})-\nabla f_i(\mathbf{x}_i^{(t)}))\right\|^2+\frac{2}{T}\sum_{t=0}^{T-1}\mathbb{E}\left\|\frac{1}{n}\sum_{i=1}^{n}\nabla f_i(\mathbf{x}_i^{(t)})\right\|^2\\ &\le \frac{2L^2}{nT}\sum_{t=0}^{T-1}\sum_{i=1}^{n}\mathbb{E}\left\|\overline{\mathbf{x}}^{(t)}-\mathbf{x}_i^{(t)}\right\|^2+\frac{2}{T}\sum_{t=0}^{T-1}\mathbb{E}\left\|\frac{1}{n}\sum_{i=1}^{n}\nabla f_i(\mathbf{x}_i^{(t)})\right\|^2\end{aligned} \tag{A.15}$$

where in the first inequality follows from Jensen's inequality and the second inequality follows from the $L-$smoothness assumption. We can bound the last term in (A.15) using (A.14) which gives us:

$$\begin{aligned}\frac{1}{T}\sum_{t=0}^{T-1}\mathbb{E}\left\|\nabla f(\overline{\mathbf{x}}^{(t)})\right\|^2 \le{}& \frac{8(1-\beta)}{\eta}\frac{(f(\mathbf{x}^{(0)})-f^*)}{T}+\frac{4L\eta\bar{\sigma}^2}{n(1-\beta)}\\ &+\left(\frac{8L^2}{nT}+\frac{2L^2}{nT}\right)\sum_{t=0}^{T-1}\sum_{i=1}^{n}\mathbb{E}\left\|\mathbf{x}_i^{(t)}-\overline{\mathbf{x}}^{(t)}\right\|^2+\frac{8L^2\eta^2\beta^4\bar{\sigma}^2}{n(1-\beta)^4}\end{aligned} \tag{A.16}$$

Note that in our matrix form, $\mathbb{E}\left\|\bar{\mathbf{X}}^{(t)}-\mathbf{X}^{(t)}\right\|_F^2=\sum_{i=1}^{n}\mathbb{E}\left\|\mathbf{x}_i^{(t)}-\overline{\mathbf{x}}^{(t)}\right\|^2$. Let $I_{(t+1)_0}\in\mathcal{I}_T$ denote the latest synchronization step before or equal to $(t+1)$. Then we have:

$$\begin{aligned}\mathbf{X}^{(t+1)} &= \mathbf{X}^{I_{(t+1)_0}}-\textstyle\sum_{t'=I_{(t+1)_0}}^{t}\eta(\beta\mathbf{V}^{(t')}+\boldsymbol{\nabla F}(\mathbf{X}^{(t')},\boldsymbol{\xi}^{(t')}))\\ \bar{\mathbf{X}}^{(t+1)} &= \bar{\mathbf{X}}^{I_{(t+1)_0}}-\textstyle\sum_{t'=I_{(t+1)_0}}^{t}\eta(\beta\mathbf{V}^{(t')}+\boldsymbol{\nabla F}(\mathbf{X}^{(t')},\boldsymbol{\xi}^{(t')}))\frac{\mathbf{1}\mathbf{1}^T}{n}\end{aligned}$$

Thus the following holds:

$$\begin{aligned}\mathbb{E}\|\mathbf{X}^{(t+1)}-\bar{\mathbf{X}}^{(t+1)}\|_F^2 &= \mathbb{E}\left\|\mathbf{X}^{I_{(t+1)_0}}-\bar{\mathbf{X}}^{I_{(t+1)_0}}-\textstyle\sum_{t'=I_{(t+1)_0}}^{t}\eta(\beta\mathbf{V}^{(t')}+\boldsymbol{\nabla}\boldsymbol{F}(\mathbf{X}^{(t')},\boldsymbol{\xi}^{(t')}))\left(\mathbf{I}-\tfrac{1}{n}\mathbf{1}\mathbf{1}^T\right)\right\|_F^2\\ &\leq 2\mathbb{E}\|\mathbf{X}^{I_{(t+1)_0}}-\bar{\mathbf{X}}^{I_{(t+1)}}\|_F^2+2\mathbb{E}\left\|\textstyle\sum_{t'=I_{(t+1)_0}}^{t}\eta(\beta\mathbf{V}^{(t')}+\boldsymbol{\nabla}\boldsymbol{F}(\mathbf{X}^{(t')},\boldsymbol{\xi}^{(t')}))\left(\mathbf{I}-\tfrac{1}{n}\mathbf{1}\mathbf{1}^T\right)\right\|_F^2\end{aligned}$$

Using $\|\mathbf{AB}\|_F \leq \|\mathbf{A}\|_F\|\mathbf{B}\|_2$ to split the second term in R.H.S. of above along with (A.7) from Fact 3 (with $k=0$) and further using the bound (A.3), we get:

$$\mathbb{E}\|\mathbf{X}^{(t+1)}-\bar{\mathbf{X}}^{(t+1)}\|_F^2 \leq 2\mathbb{E}\|\mathbf{X}^{I_{(t+1)_0}}-\bar{\mathbf{X}}^{I_{(t+1)_0}}\|_F^2 + 4\eta^2H^2nG^2\left(1+\frac{\beta^2}{(1-\beta)^2}\right) \tag{A.17}$$

We bound the first term in R.H.S. of (A.17) by Lemma 2 stated below:.

Lemma 2. ***(Consensus)*** *Let $\{\mathbf{x}_t^{(i)}\}_{t=0}^{T-1}$ be generated according to Algorithm 1 under assumptions of Theorem 2 with constant stepsize η, a threshold sequence $c_t \leq \frac{c_0}{\eta^{(1-\epsilon)}}$ for all t where $\epsilon \in (0,1)$ and c_0 is constant, and define $\overline{\mathbf{x}}_t := \frac{1}{n}\sum_{i=1}^n \mathbf{x}_t^{(i)}$. Consider the set of synchronization indices $\mathcal{I}_T = \{I_{(1)}, I_{(2)}, \ldots, I_{(t)}, \ldots\}$. Then for any $I_{(t)} \in \mathcal{I}_T$, we have:*

$$\mathbb{E}\sum_{j=1}^n\left\|\overline{\mathbf{x}}^{I_{(t)}}-\mathbf{x}_j^{I_{(t)}}\right\|^2 = \mathbb{E}\|\mathbf{X}^{I_{(t)}}-\bar{\mathbf{X}}^{I_{(t)}}\|_F^2 \leq \frac{4nA\eta^2}{p^2}$$

for constant $A = \frac{p}{2}\left(2H^2G^2\left(1+\frac{\beta^2}{(1-\beta)^2}\right)\left(\frac{16}{\omega}+\frac{4}{p}\right)+\frac{2c_0\omega}{\eta^{(1-\epsilon)}}\right)$ where $p = \frac{\delta\gamma}{8}$, $\delta := 1-|\lambda_2(\mathbf{W})|$, ω is compression parameter for operator $\mathcal{C}$.

Substituting the bound from Lemma 2 in (A.17) and using the fact that $p \leq 1$, we have:

$$\mathbb{E}\|\mathbf{X}^{(t+1)} - \bar{\mathbf{X}}^{(t+1)}\|_F^2 \leq \frac{2\eta^2}{p}\left(2H^2nG^2\left(1+\frac{\beta^2}{(1-\beta)^2}\right)\left(\frac{16}{\omega}+\frac{8}{p}\right)+\frac{2c_0\omega n}{\eta^{(1-\epsilon)}}\right) \quad \text{(A.18)}$$

for the same constant $\epsilon > 0$ as in Lemma 2. Note that the above bound holds for all values of t.

Define $\Lambda := \frac{2}{p}\left(2H^2nG^2\left(1+\frac{\beta^2}{(1-\beta)^2}\right)\left(\frac{16}{\omega}+\frac{8}{p}\right)+\frac{2\omega c_0 n}{\eta^{(1-\epsilon)}}\right)$. Substituting (A.18) in (A.16) gives us:

$$\frac{1}{T}\sum_{t=0}^{T-1}\mathbb{E}\left\|\nabla f(\overline{\mathbf{x}}^{(t)})\right\|^2 \leq \frac{8(1-\beta)}{\eta}\frac{(f(\mathbf{x}^{(0)})-f^*)}{T} + \frac{4L\eta\bar{\sigma}^2}{n(1-\beta)} + \frac{10L^2\Lambda\eta^2}{n} + \frac{8L^2\eta^2\beta^4\bar{\sigma}^2}{n(1-\beta)^4}$$

Expanding on the value of Λ, we have:

$$\begin{aligned}\frac{1}{T}\sum_{t=0}^{T-1}\mathbb{E}\left\|\nabla f(\overline{\mathbf{x}}^{(t)})\right\|^2 \leq{}& \frac{8(1-\beta)}{\eta}\frac{(f(\mathbf{x}^{(0)})-f^*)}{T} + \frac{4L\eta\bar{\sigma}^2}{n(1-\beta)} \\ &+ \frac{20\eta^2L^2}{pn}\left(2H^2nG^2\left(1+\frac{\beta^2}{(1-\beta)^2}\right)\left(\frac{16}{\omega}+\frac{8}{p}\right)\right) \\ &+ \frac{40L^2\omega nc_0\eta^{(1+\epsilon)}}{pn} + \frac{8L^2\eta^2\beta^4\bar{\sigma}^2}{n(1-\beta)^4}\end{aligned}$$

Substituting the value of $\eta = (1-\beta)\sqrt{\frac{n}{T}}$, we get:

$$\begin{aligned}\frac{1}{T}\sum_{t=0}^{T-1}\mathbb{E}\left\|\nabla f(\overline{\mathbf{x}}^{(t)})\right\|^2 \leq{}& \frac{1}{\sqrt{nT}}\left(8(f(\mathbf{x}^{(0)})-f^*)+4L\bar{\sigma}^2\right) + \frac{40L^2(1-\beta)^{(1+\epsilon)}\omega c_0 n^{(1+\epsilon)/2}}{pT^{(1+\epsilon)/2}} \\ &+ \frac{20(1-\beta)^2L^2}{Tp}\left(2H^2nG^2\left(1+\frac{\beta^2}{(1-\beta)^2}\right)\left(\frac{16}{\omega}+\frac{8}{p}\right)\right) + \frac{8L^2\beta^4\bar{\sigma}^2}{T(1-\beta)^2} \\ \leq{}& \frac{1}{\sqrt{nT}}\left(8(f(\mathbf{x}^{(0)})-f^*)+4L\bar{\sigma}^2\right) + \frac{40L^2\omega c_0 n^{(1+\epsilon)/2}(1-\beta)^{(1+\epsilon)}}{pT^{(1+\epsilon)/2}} \\ &+ \frac{80nL^2H^2G^2}{Tp}\left(\frac{16}{\omega}+\frac{8}{p}\right) + \frac{8L^2\beta^4\bar{\sigma}^2}{T(1-\beta)^2}\end{aligned}$$

where in the last inequality, we've used the fact that $(1-\beta)^r \leq 1$, $\beta^r \leq 1$ for $r > 0$. Note that we require $\eta \leq \min\left\{\frac{(1-\beta)}{4L}, \frac{(1-\beta)^2}{2\sqrt{2}L\beta^2}\right\}$, thus for $\eta = (1-\beta)\sqrt{\frac{n}{T}}$, we need to run our algorithm for $T \geq \max\left\{16L^2 n, \frac{8L^2\beta^4 n}{(1-\beta)^2}\right\}$ for the above rate expression to hold. We finally use the fact that $p \leq \omega$ (as $\delta \leq 1$ and $p := \frac{\gamma^*\delta}{8}$ with $\gamma^* \leq \omega$). This completes proof of the non-convex part of Theorem 2. We can further use the fact that $p \geq \frac{\delta^2\omega}{644}$ to get the expression given in the theorem statement.

A.4 Proof of Theorem 2 (Convex objective)

Consider the quantity $\mathbb{E}_{\boldsymbol{\xi}^{(t)}}\|\widetilde{\mathbf{x}}^{(t+1)} - \mathbf{x}^*\|^2$, where expectation is taken over sampling across all the nodes at the t'th iteration:

$$\begin{aligned}
\mathbb{E}_{\boldsymbol{\xi}^{(t)}}\|\widetilde{\mathbf{x}}^{(t+1)} - \mathbf{x}^*\|^2 &= \mathbb{E}_{\boldsymbol{\xi}^{(t)}}\left\|\widetilde{\mathbf{x}}^{(t)} - \frac{\eta}{(1-\beta)n}\sum_{j=1}^{n}\nabla F_j(\mathbf{x}_j^{(t)}, \xi_j^{(t)}) - \mathbf{x}^*\right\|^2 \\
&= \mathbb{E}_{\boldsymbol{\xi}^{(t)}}\left\|\widetilde{\mathbf{x}}^{(t)} - \mathbf{x}^* - \frac{\eta}{(1-\beta)n}\sum_{j=1}^{n}\nabla f_j(\mathbf{x}_j^{(t)}) + \frac{\eta}{(1-\beta)n}\sum_{j=1}^{n}\nabla f_j(\mathbf{x}_j^{(t)}) - \frac{\eta}{n(1-\beta)}\sum_{j=1}^{n}\nabla F_j(\mathbf{x}_j^{(t)}, \xi_j^{(t)})\right\|^2 \\
&= \left\|\widetilde{\mathbf{x}}^{(t)} - \mathbf{x}^* - \frac{\eta}{(1-\beta)n}\sum_{j=1}^{n}\nabla f_j(\mathbf{x}_j^{(t)})\right\|^2 + \frac{\eta^2}{(1-\beta)^2}\mathbb{E}_{\boldsymbol{\xi}^{(t)}}\left\|\frac{1}{n}\sum_{j=1}^{n}\nabla f_j(\mathbf{x}_j^{(t)}) - \frac{1}{n}\sum_{j=1}^{n}\nabla F_j(\mathbf{x}_j^{(t)}, \xi_j^{(t)})\right\|^2 \\
&\quad + \frac{2\eta}{(1-\beta)n}\mathbb{E}_{\boldsymbol{\xi}^{(t)}}\left\langle \widetilde{\mathbf{x}}^{(t)} - \mathbf{x}^* - \frac{\eta}{(1-\beta)n}\sum_{j=1}^{n}\nabla f_j(\mathbf{x}_j^{(t)}), \sum_{j=1}^{n}\nabla f_j(\mathbf{x}_j^{(t)}) - \sum_{j=1}^{n}\nabla F_j(\mathbf{x}_j^{(t)}, \xi_j^{(t)})\right\rangle \\
&\leq \left\|\widetilde{\mathbf{x}}^{(t)} - \mathbf{x}^* - \frac{\eta}{(1-\beta)n}\sum_{j=1}^{n}\nabla f_j(\mathbf{x}_j^{(t)})\right\|^2 + \frac{\eta^2\bar{\sigma}^2}{(1-\beta)^2 n} \qquad \text{(A.19)}
\end{aligned}$$

Where to get the last inequality we used the fact that $\mathbb{E}_{\xi_i^{(t)}}[\nabla F_i(\mathbf{x}_i^{(t)}, \xi_i^{(t)})] = \nabla f_i(\mathbf{x}_i^{(t)})$ for all $i \in [n]$ and the variance bound (A.2) from Fact 2. Now we thus consider the first term in (A.19):

$$\left\|\widetilde{\mathbf{x}}^{(t)} - \mathbf{x}^* - \frac{\eta}{(1-\beta)n}\sum_{j=1}^{n}\nabla f_j(\mathbf{x}_j^{(t)})\right\|^2 = \|\widetilde{\mathbf{x}}^{(t)} - \mathbf{x}^*\|^2 + \frac{\eta^2}{(1-\beta)^2}\underbrace{\left\|\frac{1}{n}\sum_{j=1}^{n}\nabla f_j(\mathbf{x}_j^{(t)})\right\|^2}_{T_1}$$
$$-\frac{2\eta}{(1-\beta)}\underbrace{\left\langle \widetilde{\mathbf{x}}^{(t)} - \mathbf{x}^*, \frac{1}{n}\sum_{j=1}^{n}\nabla f_j(\mathbf{x}_j^{(t)})\right\rangle}_{T_2} \tag{A.20}$$

To bound T_1 in (A.20), note that:

$$\begin{aligned}
T_1 &= \left\|\frac{1}{n}\sum_{j=1}^{n}(\nabla f_j(\mathbf{x}_j^{(t)}) - \nabla f_j(\overline{\mathbf{x}}^{(t)}) + \nabla f_j(\overline{\mathbf{x}}^{(t)}) - \nabla f_j(\mathbf{x}^*))\right\|^2 \\
&\leq \frac{2}{n}\sum_{j=1}^{n}\|\nabla f_j(\mathbf{x}_j^{(t)}) - \nabla f_j(\overline{\mathbf{x}}^{(t)})\|^2 + 2\left\|\frac{1}{n}\sum_{j=1}^{n}\nabla f_j(\overline{\mathbf{x}}^{(t)}) - \frac{1}{n}\sum_{j=1}^{n}\nabla f_j(\mathbf{x}^*)\right\|^2 \\
&\leq \frac{2L^2}{n}\sum_{j=1}^{n}\|\mathbf{x}_j^{(t)} - \overline{\mathbf{x}}^{(t)}\|^2 + 4L(f(\overline{\mathbf{x}}^{(t)}) - f^*)
\end{aligned} \tag{A.21}$$

where in the last inequality, we used $L-$Lipschitz gradient property of objectives $\{f_j\}_{j=1}^n$ to bound the first term and optimality of $\mathbf{x}^*$ for f (i.e., $\nabla f(\mathbf{x}^*) = 0$) and $L-$smoothness property of f to bound the second term as: $\left\|\frac{1}{n}\sum_{j=1}^{n}\nabla f_j(\overline{\mathbf{x}}^{(t)}) - \frac{1}{n}\sum_{j=1}^{n}\nabla f_j(\mathbf{x}^*)\right\|^2 = \left\|\nabla f(\overline{\mathbf{x}}^{(t)}) - \nabla f(\mathbf{x}^*)\right\|^2 \leq 2L\left(f(\overline{\mathbf{x}}^{(t)}) - f^*\right)$.

To bound T_2 in (A.20), note that:

$$-2T_2 = -2\left\langle \widetilde{\mathbf{x}}^{(t)} - \overline{\mathbf{x}}^{(t)}, \frac{1}{n}\sum_{j=1}^{n}\nabla f_j(\mathbf{x}_j^{(t)})\right\rangle - \frac{2}{n}\sum_{j=1}^{n}\left\langle \overline{\mathbf{x}}^{(t)} - \mathbf{x}^*, \nabla f_j(\mathbf{x}_j^{(t)})\right\rangle$$

$$= 2\frac{\beta^2}{(1-\beta)}\left\langle \frac{\eta}{n}\sum_{i=1}^{n}\mathbf{v}_i^{(t-1)}, \frac{1}{n}\sum_{j=1}^{n}\nabla f_j(\mathbf{x}_j^{(t)})\right\rangle - \frac{2}{n}\sum_{j=1}^{n}\left\langle \overline{\mathbf{x}}^{(t)} - \mathbf{x}^*, \nabla f_j(\mathbf{x}_j^{(t)})\right\rangle \tag{A.22}$$

In (A.22), we used the definition of $\widetilde{\mathbf{x}}^{(t)}$ to write $\widetilde{\mathbf{x}}^{(t)} - \overline{\mathbf{x}}^{(t)} = -\frac{\eta\beta^2}{(1-\beta)}\frac{1}{n}\sum_{i=1}^{n}\mathbf{v}_i^{(t-1)}$. Now we note a simple trick for inner-products:

$$\left\langle \frac{\eta}{n}\sum_{i=1}^{n}\mathbf{v}_i^{(t-1)}, \frac{1}{n}\sum_{j=1}^{n}\nabla f_j(\mathbf{x}_j^{(t)})\right\rangle = \left\langle \frac{(\eta)^{3/4}}{n}\sum_{i=1}^{n}\mathbf{v}_i^{(t-1)}, \frac{(\eta)^{1/4}}{n}\sum_{j=1}^{n}\nabla f_j(\mathbf{x}_j^{(t)})\right\rangle. \tag{A.23}$$

This trick is crucial to getting a speedup of n – the number of worker nodes – in our final convergence rate. Using $2\langle \mathbf{a}, \mathbf{b}\rangle \leq \|\mathbf{a}\|^2 + \|\mathbf{b}\|^2$ for bounding (A.23) and then substituting that in (A.22) gives

$$-2T_2 \leq \frac{\beta^2}{(1-\beta)}\left[(\eta)^{3/2}\left\|\frac{1}{n}\sum_{i=1}^{n}\mathbf{v}_i^{(t-1)}\right\|^2 + (\eta)^{1/2}\left\|\frac{1}{n}\sum_{j=1}^{n}\nabla f_j(\mathbf{x}_j^{(t)})\right\|^2\right] - \frac{2}{n}\sum_{j=1}^{n}\left\langle \overline{\mathbf{x}}^{(t)} - \mathbf{x}^*, \nabla f_j(\mathbf{x}_j^{(t)})\right\rangle \tag{A.24}$$

Note that the second term of (A.24) is the same as T_1 from (A.20) and we have already bounded that in (A.21). We now focus on bounding the last term of (A.24). Using expression for convexity and L-smoothness for f_j, $j \in [n]$ respectively, we can bound this as follows:

$$\begin{aligned}
-\frac{2}{n}\sum_{j=1}^{n}\langle \overline{\mathbf{x}}^{(t)} - \mathbf{x}^*, \nabla f_j(\mathbf{x}_j^{(t)})\rangle &= -\frac{2}{n}\sum_{j=1}^{n}\left[\left\langle \overline{\mathbf{x}}^{(t)} - \mathbf{x}_j^{(t)}, \nabla f_j(\mathbf{x}_j^{(t)})\right\rangle + \left\langle \mathbf{x}_j^{(t)} - \mathbf{x}^*, \nabla f_j(\mathbf{x}_j^{(t)})\right\rangle\right] \\
&\leq -\frac{2}{n}\sum_{j=1}^{n}\left[f_j(\overline{\mathbf{x}}^{(t)}) - f_j(\mathbf{x}_j^{(t)}) - \frac{L}{2}\|\overline{\mathbf{x}}^{(t)} - \mathbf{x}_j^{(t)}\|^2 + f_j(\mathbf{x}_j^{(t)}) - f_j(\mathbf{x}^*)\right] \\
&= -2(f(\overline{\mathbf{x}}^{(t)}) - f(\mathbf{x}^*)) + \frac{L}{n}\sum_{j=1}^{n}\|\overline{\mathbf{x}}^{(t)} - \mathbf{x}_j^{(t)}\|^2
\end{aligned} \tag{A.25}$$

Substituting the bounds for the second and the last terms of (A.24) from (A.21) and (A.25), respectively, we get

$$-2T_2 \leq \frac{(\eta)^{3/2}\beta^2}{(1-\beta)} \left\| \frac{1}{n}\sum_{i=1}^{n} \mathbf{v}_i^{(t-1)} \right\|^2 + \frac{(\eta)^{1/2}\beta^2}{(1-\beta)} \left(\frac{2L^2}{n}\sum_{j=1}^{n} \|\mathbf{x}_j^{(t)} - \overline{\mathbf{x}}^{(t)}\|^2 + 4L(f(\overline{\mathbf{x}}^{(t)}) - f^*) \right)$$
$$- 2(f(\overline{\mathbf{x}}^{(t)}) - f(\mathbf{x}^*)) + \frac{L}{n}\sum_{j=1}^{n} \|\overline{\mathbf{x}}^{(t)} - \mathbf{x}_j^{(t)}\|^2$$

Thus we finally have:

$$-\frac{2\eta}{(1-\beta)}T_2 \leq \frac{\eta^{5/2}\beta^2}{(1-\beta)^2} \left\| \frac{1}{n}\sum_{i=1}^{n} \mathbf{v}_i^{(t-1)} \right\|^2 + \left(\frac{2\eta^{3/2}\beta^2L^2}{(1-\beta)^2} + \frac{\eta L}{(1-\beta)} \right) \frac{1}{n}\sum_{j=1}^{n} \|\mathbf{x}_j^{(t)} - \overline{\mathbf{x}}^{(t)}\|^2$$
$$+ \left(\frac{4\eta^{3/2}\beta^2 L}{(1-\beta)^2} - \frac{2\eta}{(1-\beta)} \right) \left(f(\overline{\mathbf{x}}^{(t)}) - f^* \right) \tag{A.26}$$

Substituting (A.21), (A.26) in (A.20) and using the resulting bound back in (A.19), and then taking expectation w.r.t. the entire process, we get:

$$\mathbb{E}\|\widetilde{\mathbf{x}}^{(t+1)} - \mathbf{x}^*\|^2 \leq \mathbb{E}\|\widetilde{\mathbf{x}}^{(t)} - \mathbf{x}^*\|^2 + \frac{\eta^{5/2}\beta^2}{(1-\beta)^2}\mathbb{E}\left\| \frac{1}{n}\sum_{i=1}^{n} \mathbf{v}_i^{(t-1)} \right\|^2 + \frac{\eta^2\bar{\sigma}^2}{(1-\beta)^2 n}$$
$$+ \left(\frac{2\eta^2L^2}{(1-\beta)^2} + \frac{2\eta^{3/2}\beta^2L^2}{(1-\beta)^2} + \frac{\eta L}{(1-\beta)} \right) \frac{1}{n}\sum_{j=1}^{n} \mathbb{E}\|\mathbf{x}_j^{(t)} - \overline{\mathbf{x}}^{(t)}\|^2$$
$$+ \left(\frac{4\eta^2 L}{(1-\beta)^2} + \frac{4\eta^{3/2}\beta^2 L}{(1-\beta)^2} - \frac{2\eta}{(1-\beta)} \right) \left(\mathbb{E}f(\overline{\mathbf{x}}^{(t)}) - f^* \right) \tag{A.27}$$

Using the fact that $\mathbb{E}\left\| \frac{1}{n}\sum_{j=1}^{n} \mathbf{v}_j^{(t)} \right\|^2 \leq \frac{G^2}{(1-\beta)^2}$ for all $t \geq 1$ (see proof of Fact 3), we have:

$$\mathbb{E}\|\widetilde{\mathbf{x}}^{(t+1)} - \mathbf{x}^*\|^2 \leq \mathbb{E}\|\widetilde{\mathbf{x}}^{(t)} - \mathbf{x}^*\|^2 + \frac{\eta^{5/2}\beta^2G^2}{(1-\beta)^4} + \frac{\eta^2\bar{\sigma}^2}{(1-\beta)^2 n}$$

$$+ \left(\frac{2\eta^2 L^2}{(1-\beta)^2} + \frac{2\eta^{3/2}\beta^2 L^2}{(1-\beta)^2} + \frac{\eta L}{(1-\beta)} \right) \frac{1}{n} \sum_{j=1}^{n} \mathbb{E}\|\mathbf{x}_j^{(t)} - \overline{\mathbf{x}}^{(t)}\|^2$$
$$+ \left(\frac{4\eta^2 L}{(1-\beta)^2} + \frac{4\eta^{3/2}\beta^2 L}{(1-\beta)^2} - \frac{2\eta}{(1-\beta)} \right) \left(\mathbb{E} f(\overline{\mathbf{x}}^{(t)}) - f^* \right) \tag{A.28}$$

If we take $\eta \leq \min\left\{ \frac{(1-\beta)}{8L}, \frac{(1-\beta)^2}{(8L\beta^2)^2} \right\}$, then we have:

$$\left(\frac{2\eta^2 L^2}{(1-\beta)^2} + \frac{2\eta^{3/2}\beta^2 L^2}{(1-\beta)^2} + \frac{\eta L}{(1-\beta)} \right) \leq \frac{3\eta L}{2(1-\beta)} \tag{A.29}$$
$$\left(\frac{4\eta^2 L}{(1-\beta)^2} + \frac{4\eta^{3/2}\beta^2 L}{(1-\beta)^2} - \frac{2\eta}{(1-\beta)} \right) \leq -\frac{\eta}{(1-\beta)} \tag{A.30}$$

Substituting the bounds from (A.29) and (A.30) to (A.28) gives

$$\mathbb{E}\|\widetilde{\mathbf{x}}^{(t+1)} - \mathbf{x}^*\|^2 \leq \mathbb{E}\|\widetilde{\mathbf{x}}^{(t)} - \mathbf{x}^*\|^2 + \frac{\eta^{5/2}\beta^2 G^2}{(1-\beta)^4} + \frac{\eta^2 \bar{\sigma}^2}{(1-\beta)^2 n} + \frac{3\eta L}{2(1-\beta)} \frac{1}{n} \sum_{j=1}^{n} \mathbb{E}\|\mathbf{x}_j^{(t)} - \overline{\mathbf{x}}^{(t)}\|^2$$
$$- \frac{\eta}{(1-\beta)} \left(\mathbb{E} f(\overline{\mathbf{x}}^{(t)}) - f^* \right) \tag{A.31}$$

We can now bound the second last term in R.H.S. of (A.31) similar to (A.18) in the proof of non-convex part of Theorem 2 given. This gives us the bound:

$$\mathbb{E}\|\mathbf{X}^{(t+1)} - \bar{\mathbf{X}}^{(t+1)}\|_F^2 \leq \frac{2\eta^2}{p} \left(2H^2 n G^2 \left(1 + \frac{\beta^2}{(1-\beta)^2} \right) \left(\frac{16}{\omega} + \frac{8}{p} \right) + \frac{2c_0 \omega n}{\eta^{(1-\epsilon)}} \right)$$

Using above bound for the term $\sum_{j=1}^{n} \mathbb{E}\|\mathbf{x}_j^{(t)} - \overline{\mathbf{x}}^{(t)}\|^2$ in (A.31) we get:

$$\mathbb{E}\|\widetilde{\mathbf{x}}^{(t+1)} - \mathbf{x}^*\|^2 \leq \mathbb{E}\|\widetilde{\mathbf{x}}^{(t)} - \mathbf{x}^*\|^2 + \frac{\eta^{5/2}\beta^2 G^2}{(1-\beta)^4} + \frac{\eta^2 \bar{\sigma}^2}{(1-\beta)^2 n} - \frac{\eta}{(1-\beta)} \left(\mathbb{E} f(\overline{\mathbf{x}}^{(t)}) - f^* \right)$$

$$+ \frac{3\eta^3 L}{p(1-\beta)} \left(2H^2G^2 \left(1 + \frac{\beta^2}{(1-\beta)^2}\right) \left(\frac{16}{\omega} + \frac{8}{p}\right) + \frac{2c_0\omega}{\eta^{(1-\epsilon)}} \right) \tag{A.32}$$

By rearranging terms in (A.32) and noting that $p \leq \omega$ (as $\delta \leq 1$ and $p := \frac{\gamma^*\delta}{8}$ with $\gamma^* \leq \omega$) and the fact that $\left(1 + \frac{\beta^2}{(1-\beta)^2}\right) \leq \frac{2}{(1-\beta)^2}$ (because $\beta < 1$), we get:

$$\begin{aligned}\mathbb{E}\|\widetilde{\mathbf{x}}^{(t+1)} - \mathbf{x}^*\|^2 \leq \mathbb{E}\|\widetilde{\mathbf{x}}^{(t)} - \mathbf{x}^*\|^2 + \frac{\eta^{5/2}\beta^2G^2}{(1-\beta)^4} + \frac{\eta^2\bar{\sigma}^2}{(1-\beta)^2 n} - \frac{\eta}{(1-\beta)}\left(\mathbb{E}f(\overline{\mathbf{x}}^{(t)}) - f^*\right) \\ + \frac{288\eta^3LH^2G^2}{p^2(1-\beta)^3} + \frac{6c_0\omega L\eta^{(2+\epsilon)}}{p(1-\beta)}\end{aligned} \tag{A.33}$$

Summing (A.33) from $t = 0$ to $T-1$, rearranging terms and diving by T both sides gives us:

$$\begin{aligned}\sum_{t=0}^{T-1} \frac{\left(\mathbb{E}f(\overline{\mathbf{x}}^{(t)}) - f^*\right)}{T} \leq \frac{(1-\beta)}{\eta} \sum_{t=0}^{T-1} \frac{\left(\mathbb{E}\|\widetilde{\mathbf{x}}^{(t)} - \mathbf{x}^*\|^2 - \mathbb{E}\|\widetilde{\mathbf{x}}^{(t+1)} - \mathbf{x}^*\|^2\right)}{T} + \frac{\eta^{3/2}\beta^2G^2}{(1-\beta)^3} + \frac{\eta\bar{\sigma}^2}{(1-\beta)n} \\ + \frac{288\eta^2LH^2G^2}{p^2(1-\beta)^2} + \frac{6c_0\omega L\eta^{(1+\epsilon)}}{p}\end{aligned}$$

Using Jensen's inequality for convex function f on the L.H.S. and setting $\eta = (1-\beta)\sqrt{\frac{n}{T}}$ for $T \geq \max\{(8L)^2 n, \frac{(8\beta^2L)^4 n}{(1-\beta)^2}\}$, for $\overline{\mathbf{x}}_{avg}^{(T)} := \frac{1}{T}\sum_{t=0}^{T-1} \bar{\mathbf{x}}^{(t)}$ we have that:

$$\begin{aligned}\mathbb{E}f(\overline{\mathbf{x}}_{avg}^{(T)}) - f^* \leq \frac{\left(\mathbb{E}\|\widetilde{\mathbf{x}}^{(0)} - \mathbf{x}^*\|^2 - \mathbb{E}\|\widetilde{\mathbf{x}}^{(T)} - \mathbf{x}^*\|^2\right)}{\sqrt{nT}} + \frac{n^{3/4}\beta^2G^2}{(1-\beta)^{3/2}T^{3/4}} + \frac{\bar{\sigma}^2}{\sqrt{nT}} \\ + \frac{288LH^2G^2}{p^2T} + \frac{6c_0\omega L(1-\beta)^{(1+\epsilon)}n^{(1+\epsilon)/2}}{pT^{(1+\epsilon)/2}}\end{aligned}$$

Using the fact that $\widetilde{\mathbf{x}}^{(0)} = \overline{\mathbf{x}}^{(0)}$ and $\epsilon, \beta \in (0, 1)$ we have:

$$\mathbb{E}f(\overline{\mathbf{x}}_{avg}^{(T)}) - f^* \leq \frac{\|\overline{\mathbf{x}}^{(0)} - \mathbf{x}^*\|^2 + \bar{\sigma}^2}{\sqrt{nT}} + \frac{n^{3/4}\beta^2 G^2}{(1-\beta)^{3/2} T^{3/4}} + \frac{384nLH^2G^2}{p^2T} + \frac{6c_0\omega Ln^{(1+\epsilon)/2}}{pT^{(1+\epsilon)/2}}$$

This completes proof of convex part of Theorem 2. We can further use the fact that $p \geq \frac{\delta^2\omega}{644}$ to get the expression given in the theorem statement.

Appendix B

Omitted Details for Chapter 4

B.1 Convergence analysis for Sample Feedback

We first introduce a compact vector notation which we will use throughout the proof. Consider the stacked (concatenated) vector of the node parameter vectors $\{\mathbf{x}_i\}_{i=1}^n$ which we denote by $\mathbf{x}$, and thus is nd-dimensional. Similarly, we define the vector $\boldsymbol{\lambda}$ of size m which stacks together the dual variables λ_{ij} for $i \in [n]$ and $j \in \mathcal{N}_i$. The vector $\mathbf{g}(\mathbf{x})$ represents the the stacked vector of constraint values $g_{ij}(\mathbf{x}_j, \mathbf{x}_j)$, and is also m-dimensional. Finally, $\boldsymbol{\xi}$ denotes the concatenated vector of samples across the nodes. The projection $\Pi_{\mathcal{X}^n}(\mathbf{x})$ refers to projection of $\mathbf{x}$ on the space $\mathcal{X}^n$ where each individual node parameter comprising $\mathbf{x}$ is projected onto $\mathcal{X}$. Under this compact notation, the modified Lagrangian presented in (4.3) can be re-written as:

$$\mathcal{L}(\mathbf{x}, \boldsymbol{\lambda}) = f(\mathbf{x}, \boldsymbol{\xi}) + \boldsymbol{\lambda}^T \mathbf{g}(\mathbf{x}) - \frac{\delta\eta}{2}\|\boldsymbol{\lambda}\|^2 \tag{B.1}$$

We now present a few auxiliary results which we use through the course of the proof. Some of these can be derived from the assumptions made in A.4-A.7.

Fact 6. *Suppose $\mathcal{A} \subset \mathbb{R}^l$ is closed and convex. Then, for any $\mathbf{y} \in \mathbb{R}^l$ and $\mathbf{x} \in \mathcal{A}$, we* *have:*

$$\|\mathbf{x} - \Pi_{\mathcal{A}}(\mathbf{y})\|_2 \leq \|\mathbf{x} - \mathbf{y}\|_2$$

where $\Pi_{\mathcal{A}}(\mathbf{y})$ *denotes the projection of* $\mathbf{y}$ *on the set* $\mathcal{A}$*.*

Fact 7. *(Bound on gradients of the Lagrangian) Consider the Lagrangian function over the primal and dual variables defined in* (B.1)*. We have the following bounds:*

1. $\mathbb{E}\|\nabla_{\boldsymbol{\lambda}}\mathcal{L}(\mathbf{x}^{(t)}, \boldsymbol{\lambda}^{(t)})\|^2 \leq 2C^2 + 2\delta^2\eta^2\mathbb{E}\|\boldsymbol{\lambda}^{(t)}\|^2$
2. $\mathbb{E}\left\|\nabla_{\mathbf{x}}\mathcal{L}(\mathbf{x}^{(t)}, \boldsymbol{\lambda}^{(t)})\right\|^2 \leq (1+m)\left(G^2 + \widetilde{G}^2\mathbb{E}\left\|\boldsymbol{\lambda}\right\|^2\right)$

where $C^2, \tilde{G}$ *and* G *are as defined in Assumptions A.6 and A.7. Proof of this fact can be found in [SCDB].*

Fact 8. *For all* $\mathbf{x} \in \mathcal{X}^n$*, we have:*

$$\mathbb{E}[F(\mathbf{x})] - F(\mathbf{x}^*) > -4GR$$

where $\mathbf{x}^*$ *is an optimal solution of* (4.1)*, and* R, G *are as defined in Assumptions A.4 and A.7, respectively. We provide a proof for Fact 8 in [SCDB].*

B.1.0.1 Proof of Theorem 3

We first consider the following lemma which establishes a relationship between the Lagrangian function and the primal, dual variables in Algorithm 2. The proof for the lemma, provided in the [SCDB], relies on considering the update steps of the primal and dual variables in Algorithm 2 and invoking convexity/concavity arguments for the Lagrangian function.

Lemma 3. *Consider the update steps in Algorithm 2 with learning rate η and parameter $\delta \geq 0$. Under assumptions A.4-A.7, for $\mathbf{x} \in \mathcal{X}^n$ and $\boldsymbol{\lambda} \in \mathbb{R}^m$ with $\boldsymbol{\lambda} \succeq \mathbf{0}$, the summation of the Lagrangian function satisfies:*

$$\sum_{t=1}^{T} \mathbb{E}\left(\mathcal{L}(\mathbf{x}^{(t)}, \boldsymbol{\lambda}) - \mathcal{L}(\mathbf{x}, \boldsymbol{\lambda}^{(t)})\right) \leq \frac{1}{2\eta}\left(\|\boldsymbol{\lambda}\|^2 + 4R^2\right) + \eta T\left((1+m)G^2 + C^2\right)$$
$$+ \frac{1}{2\eta}\sum_{t=1}^{T} \mathbb{E}\left\|\mathbf{x}^{(t)} - \tilde{\mathbf{x}}^{(t)}\right\|^2 + \eta\left((1+m)\tilde{G}^2 + \delta^2\eta^2\right)\sum_{t=1}^{T}\mathbb{E}[\|\boldsymbol{\lambda}^{(t)}\|^2]$$

where $G, C, \tilde{G}$, R are defined in assumptions A.4-A.7.

Using the definition of Lagrangian from (B.1) and $\mathbb{E}[f(\mathbf{x}^{(t)}, \boldsymbol{\xi}^{(t)})] = F(\mathbf{x}^{(t)})$, the L.H.S. of the result in Lemma 3 can also be written as following for any $\boldsymbol{\lambda} \succeq \mathbf{0}$:

$$\mathbb{E}\left[\sum_{t=1}^{T}\left(\mathcal{L}(\mathbf{x}^{(t)}, \boldsymbol{\lambda}) - \mathcal{L}(\mathbf{x}, \boldsymbol{\lambda}^{(t)})\right)\right] = \sum_{t=1}^{T}(\mathbb{E}[F(\mathbf{x}^{(t)})] - F(\mathbf{x}^*)) + \left\langle \boldsymbol{\lambda}, \sum_{t=1}^{T}\mathbb{E}[\mathbf{g}(\mathbf{x}^{(t)})]\right\rangle - \frac{\delta\eta T}{2}\|\boldsymbol{\lambda}\|^2$$
$$- \mathbb{E}\left[\sum_{t=1}^{T}\langle\boldsymbol{\lambda}^{(t)}, \mathbf{g}(\mathbf{x}^*)\rangle\right] + \frac{\delta\eta}{2}\mathbb{E}\left[\sum_{t=1}^{T}\|\boldsymbol{\lambda}^{(t)}\|^2\right]$$

Rearranging the terms and employing the bound from Lemma 3, for any $\boldsymbol{\lambda} \succeq \mathbf{0}$, we thus have:

$$\sum_{t=1}^{T}\left(\mathbb{E}[F(\mathbf{x}^{(t)})] - F(\mathbf{x}^*)\right) + \left\langle \boldsymbol{\lambda}, \sum_{t=1}^{T}\mathbb{E}[\mathbf{g}(\mathbf{x}^{(t)})]\right\rangle - \frac{\delta\eta T}{2}\|\boldsymbol{\lambda}\|^2 - \mathbb{E}\left[\sum_{t=1}^{T}\langle\boldsymbol{\lambda}^{(t)}, \mathbf{g}(\mathbf{x}^*)\rangle\right]$$
$$\leq \frac{1}{2\eta}\left(\|\boldsymbol{\lambda}\|^2 + 4R^2\right) + \frac{1}{2\eta}\sum_{t=1}^{T}\mathbb{E}\left\|\mathbf{e}^{(t)}\right\|^2 + \eta T\left((1+m)G^2 + C^2\right) \tag{B.2}$$
$$+ \eta\left((1+m)\tilde{G}^2 + \delta^2\eta^2 - \frac{\delta}{2}\right)\sum_{t=1}^{T}\mathbb{E}[\|\boldsymbol{\lambda}^{(t)}\|^2] \tag{B.3}$$

where we have defined $\mathbb{E}\left\|\mathbf{e}^{(t)}\right\|^2 := \mathbb{E}\left\|\widetilde{\mathbf{x}}^{(t)} - \mathbf{x}^{(t)}\right\|^2$ on the R.H.S. of (B.2). This term relates to the error between the copies of the parameters at time t (denoted by $\widetilde{\mathbf{x}}^{(t)}$)

and the true parameters of the nodes (given by $\mathbf{x}^{(t)}$). We provide a bound for this term in Lemma 4 stated below, the proof of which is provided in the arXiv version of the paper [SCDB].

Lemma 4. *For the update steps in Algorithm 2, the norm of expected error $\mathbb{E}\|\mathbf{e}^{(t)}\|$ for $t \in [T]$ is bounded as:*

$$\mathbb{E}\|\mathbf{e}^{(t)}\|^2 \le \frac{2\eta^2}{\omega}\sum_{k=0}^{t-2}\left(1-\frac{\omega}{2}\right)^k \mathbb{E}\|\nabla_{\mathbf{x}}\mathcal{L}_{t-1-k}(\mathbf{x}^{(t-1-k)}, \boldsymbol{\lambda}^{(t-1-k)})\|^2$$

Plugging the bound for $\mathbb{E}\|\mathbf{e}^{(t)}\|^2$ from Lemma 4 into (B.2):

$$\begin{aligned}
&\sum_{t=1}^{T}\left(\mathbb{E}[F(\mathbf{x}^{(t)})]-F(\mathbf{x}^*)\right) + \left\langle \boldsymbol{\lambda}, \sum_{t=1}^{T}\mathbb{E}[\mathbf{g}(\mathbf{x}^{(t)})]\right\rangle - \frac{\delta\eta T}{2}\|\boldsymbol{\lambda}\|^2 - \mathbb{E}\left[\sum_{t=1}^{T}\langle \boldsymbol{\lambda}^{(t)}, \mathbf{g}(\mathbf{x}^*)\rangle\right] \\
&\le \frac{1}{2\eta}\left(\|\boldsymbol{\lambda}\|^2 + 4R^2\right) + \eta T\left((1+m)G^2 + C^2\right) \\
&+ \frac{\eta}{\omega}\sum_{t=1}^{T}\sum_{k=0}^{t-2}\left(1-\frac{\omega}{2}\right)^k \mathbb{E}\left\|\nabla_{\mathbf{x}}\mathcal{L}_{t-1-k}(\mathbf{x}^{(t-1-k)}, \boldsymbol{\lambda}^{(t-1-k)})\right\|^2 \\
&\quad + \eta\left((1+m)\tilde{G}^2 + \delta^2\eta^2 - \frac{\delta}{2}\right)\sum_{t=1}^{T}\mathbb{E}[\|\boldsymbol{\lambda}^{(t)}\|^2] \qquad \text{(B.4)} \\
&= \frac{1}{2\eta}\left(\|\boldsymbol{\lambda}\|^2 + 4R^2\right) + \eta T\left((1+m)G^2 + C^2\right) + \eta\left((1+m)\tilde{G}^2 + \delta^2\eta^2 - \frac{\delta}{2}\right)\sum_{t=1}^{T}\mathbb{E}[\|\boldsymbol{\lambda}^{(t)}\|^2] \\
&\qquad + \frac{\eta}{\omega}\sum_{k=1}^{T-1}\sum_{t=k+1}^{T}\left(1-\frac{\omega}{2}\right)^{(t-1-k)}\mathbb{E}\left\|\nabla_{\mathbf{x}}\mathcal{L}_k(\mathbf{x}^{(k)}, \boldsymbol{\lambda}^{(k)})\right\|^2
\end{aligned}$$

where the equality follows from rewriting the double-sum of the second term. Using $\sum_{t=k+1}^{T}\left(1-\frac{\omega}{2}\right)^{(t-1-k)} \le \sum_{t=0}^{\infty}\left(1-\frac{\omega}{2}\right)^{(t)} = \frac{2}{\omega}$, we get:

$$\sum_{t=1}^{T}(\mathbb{E}[F(\mathbf{x}^{(t)})]-F(\mathbf{x}^*)) + \left\langle \boldsymbol{\lambda}, \sum_{t=1}^{T}\mathbb{E}[\mathbf{g}(\mathbf{x}^{(t)})]\right\rangle - \frac{\delta\eta T}{2}\|\boldsymbol{\lambda}\|^2 - \mathbb{E}\left[\sum_{t=1}^{T}\langle \boldsymbol{\lambda}^{(t)}, \mathbf{g}(\mathbf{x}^*)\rangle\right]$$

$$\leq \frac{1}{2\eta}\left(\|\boldsymbol{\lambda}\|^2 + 4R^2\right) + \eta T\left((1+m)G^2 + C^2\right) + \eta\left((1+m)\tilde{G}^2 + \delta^2\eta^2 - \frac{\delta}{2}\right)\sum_{t=1}^{T}\mathbb{E}[\|\boldsymbol{\lambda}^{(t)}\|^2]$$
$$+ \frac{2\eta}{\omega^2}\sum_{t=1}^{T-1}\mathbb{E}\left\|\nabla_{\mathbf{x}}\mathcal{L}_t(\mathbf{x}^{(t)}, \boldsymbol{\lambda}^{(t)})\right\|^2 \tag{B.5}$$

Using the bound from (b) in Fact 7 for the last term in above, and noting that $\frac{2}{\omega^2} > 1$ gives us:

$$\sum_{t=1}^{T}(\mathbb{E}[F(\mathbf{x}^{(t)})] - F(\mathbf{x}^*)) + \left\langle \boldsymbol{\lambda}, \sum_{t=1}^{T}\mathbb{E}[\mathbf{g}(\mathbf{x}^{(t)})]\right\rangle - \frac{\delta\eta T}{2}\|\boldsymbol{\lambda}\|^2 - \mathbb{E}\left[\sum_{t=1}^{T}\langle\boldsymbol{\lambda}^{(t)}, \mathbf{g}(\mathbf{x}^*)\rangle\right]$$
$$\leq \frac{1}{2\eta}\left(\|\boldsymbol{\lambda}\|^2 + 4R^2\right) + \eta T\left(\frac{4}{\omega^2}(1+m)G^2 + C^2\right) + \eta\left(\frac{4}{\omega^2}(1+m)\tilde{G}^2 + \delta^2\eta^2 - \frac{\delta}{2}\right)\sum_{t=1}^{T}\mathbb{E}[\|\boldsymbol{\lambda}^{(t)}\|^2] \tag{B.6}$$

We now focus on the last term in the above equation, which has a coefficient of $\left(\frac{4}{\omega^2}(1+m)\tilde{G}^2 + \delta^2\eta^2 - \frac{\delta}{2}\right)$. To get rid of the last term in the upper bound, we choose the value of δ such that this coefficient is negative. It can be easily checked that the following value of δ satisfies this requirement:

$$\delta = \frac{1 - \sqrt{1 - \frac{64\eta^2(1+m)\tilde{G}^2}{\omega^2}}}{4\eta^2}$$

Note that we require running the algorithm for $T \geq \frac{64a^2(1+m)\tilde{G}^2}{\omega^2}$ for the choice $\eta = \frac{a}{\sqrt{T}}$. For $T \to \infty$ (i.e., $\eta \to 0$), it can be verified that the value of δ converges to a positive constant. Using the above value of δ, the fact $\mathbb{E}\left[\sum_{t=1}^{T}\langle\boldsymbol{\lambda}^{(t)}, \mathbf{g}(\mathbf{x}^*)\rangle\right] \leq 0$ since $\boldsymbol{\lambda}^{(t)} \succeq \mathbf{0}$ for $t \in [T]$ and $\mathbf{g}(\mathbf{x}^*) \preceq \mathbf{0}$ and rearranging the terms, we get:

$$\sum_{t=1}^{T}\left(\mathbb{E}[F(\mathbf{x}^{(t)})] - F(\mathbf{x}^*)\right) + \left\langle \boldsymbol{\lambda}, \sum_{t=1}^{T}\mathbb{E}[\mathbf{g}(\mathbf{x}^{(t)})]\right\rangle - \left(\frac{\delta\eta T}{2} + \frac{1}{2\eta}\right)\|\boldsymbol{\lambda}\|^2$$

$$\leq \frac{2R^2}{\eta} + \eta T\left(\frac{4}{\omega^2}(1+m)G^2 + C^2\right) \tag{B.7}$$

Recall that $\boldsymbol{\lambda}$ can be any non-negative vector. We set it as $\boldsymbol{\lambda} = \frac{\left[\mathbb{E}\left[\sum_{t=1}^T \mathbf{g}(\mathbf{x}^{(t)})\right]\right]^+}{\delta\eta T + \frac{1}{\eta}}$. Plugging this in (B.7) yields:

$$\sum_{t=1}^{T}\left(\mathbb{E}[F(\mathbf{x}^{(t)})] - F(\mathbf{x}^*)\right) + \sum_{i=1}^{n}\sum_{j\in\mathcal{N}_i} \frac{\left(\left[\mathbb{E}\left[\sum_{t=1}^T g_{ij}(\mathbf{x}_i^{(t)}, \mathbf{x}_j^{(t)})\right]\right]^+\right)^2}{2\left(\delta\eta T + \frac{1}{\eta}\right)}$$
$$\leq \frac{2R^2}{\eta} + \eta T\left(\frac{4}{\omega^2}(1+m)G^2 + C^2\right) \tag{B.8}$$

Dividing both sides of (B.8) by T and noting that the second term on the L.H.S. of (B.8) is positive, we can bound the time-average sub-optimality of F as:

$$\sum_{t=1}^{T}\frac{\left(\mathbb{E}[F(\mathbf{x}^{(t)})] - F(\mathbf{x}^*)\right)}{T} \leq \frac{2R^2}{\eta T} + \eta\left(\frac{4}{\omega^2}(1+m)G^2 + C^2\right)$$

Using the convexity of F and setting $\eta = \frac{a}{\sqrt{T}}$ for some positive constant a, concludes the proof of the convergence rate for the objective sub-optimality given in (4.8) in Theorem 3. We now prove our result for the pairwise constraint functions. From Fact 8, $\forall \mathbf{x} \in \mathcal{X}^n$, we have $\mathbb{E}[F(\mathbf{x})] - F(\mathbf{x}^*) > -4GR$. Using this inequality in (B.8):

$$\sum_{i=1}^{n}\sum_{j\in\mathcal{N}_i}\left(\left[\mathbb{E}\left[\sum_{t=1}^{T} g_{ij}(\mathbf{x}_i^{(t)}, \mathbf{x}_j^{(t)})\right]\right]^+\right)^2 \leq \frac{4R^2}{\eta^2} + T\left(4R^2\delta + \frac{8}{\omega^2}(1+m)G^2 + 2C^2 + \frac{8GR}{\eta}\right)$$
$$+ T^2\left(2\delta\eta^2\left(\frac{4}{\omega^2}(1+m)G^2 + C^2\right) + 8\delta\eta GR\right)$$

Note that the above bound also holds for a given $i \in [n]$ and $j \in \mathcal{N}_i$, that is, the R.H.S. of the above equation is also a bound for the term $\left(\left[\mathbb{E}\left[\sum_{t=1}^T g_{ij}(\mathbf{x}_i^{(t)}, \mathbf{x}_j^{(t)})\right]\right]^+\right)^2$.

Taking square root on both sides and using the fact that $\sqrt{\sum_{i=1}^{n} p_i} \leq \sum_{i=1}^{n} \sqrt{p_i}$ for positive $p_1, \ldots, p_n$ yields:

$$\mathbb{E}\left[\sum_{t=1}^{T} g_{ij}(\mathbf{x}_i^{(t)}, \mathbf{x}_j^{(t)})\right] \leq \frac{2R}{\eta} + \sqrt{2T}\sqrt{\left(2R^2\delta + \frac{4}{\omega^2}(1+m)G^2 + C^2 + \frac{4GR}{\eta}\right)} + \sqrt{2}T\sqrt{\left(\delta\eta^2\left(\frac{4}{\omega^2}(1+m)G^2 + C^2\right) + 4\delta\eta GR\right)}$$

Dividing both sides of above by T, using the convexity of constraint function g_{ij} and substituting $\eta = \frac{a}{\sqrt{T}}$ concludes the proof of (4.9) in Theorem 3. □

B.2 Convergence analysis for Bandit Feedback

As done earlier for proof of bandit feedback, we use a compact notation by stacking together the parameters across the nodes. The modified Lagrangian in (4.10) for a time step $t \in [T]$ in this notation is given as:

$$\tilde{\mathcal{L}}(\mathbf{x}^{(t)}, \boldsymbol{\lambda}^{(t)}) = \tilde{f}(\mathbf{x}^{(t)}, \boldsymbol{\xi}^{(t)}) + \langle \boldsymbol{\lambda}^{(t)}, \mathbf{g}(\mathbf{x}^{(t)})\rangle - \frac{\delta\eta}{2}\|\boldsymbol{\lambda}^{(t)}\|^2 \tag{B.9}$$

where $\mathbf{x}^{(t)}$, is of size nd, $\boldsymbol{\lambda}^{(t)}$ is of size m, and $\boldsymbol{\xi}^{(t)}$ is collection of samples across all the nodes at time t. We construct another quantity of interest:

$$\mathcal{H}(\mathbf{x}^{(t)}, \boldsymbol{\lambda}^{(t)}) = \widetilde{\mathcal{L}}(\mathbf{x}^{(t)}, \boldsymbol{\lambda}^{(t)}) + \langle \mathbf{p}^{(t)} - \widetilde{\mathcal{L}}(\mathbf{x}^{(t)}, \boldsymbol{\lambda}^{(t)}), \mathbf{x}^{(t)}\rangle \tag{B.10}$$

It can be seen that $\mathcal{H}(\mathbf{x}^{(t)}), \boldsymbol{\lambda}^{(t)}$ is convex in the parameter $\mathbf{x}^{(t)}$ and concave in $\boldsymbol{\lambda}^{(t)}$ for any t. Further, the gradients of the function $\mathcal{H}(\mathbf{x}^{(t)}), \boldsymbol{\lambda}^{(t)}$ satisfy:

$$\nabla_{\mathbf{x}}\mathcal{H}(\mathbf{x}^{(t)}, \boldsymbol{\lambda}^{(t)}) = \mathbf{p}^{(t)}, \quad \nabla_{\boldsymbol{\lambda}}\mathcal{H}(\mathbf{x}^{(t)}, \boldsymbol{\lambda}^{(t)}) = \nabla_{\boldsymbol{\lambda}}\widetilde{\mathcal{L}}(\mathbf{x}^{(t)}, \boldsymbol{\lambda}^{(t)})$$

To derive our results, we consider another auxiliary result:

Fact 9. *Under Assumptions A.5 and A.6, for all $t \in [T]$, $i \in [n]$ and any $\mathbf{u}, \mathbf{v} \in \mathcal{X}$, we have:*

$$\mathbb{E}_{\boldsymbol{\xi}_i^{(t)}}[f_i(\mathbf{u}, \boldsymbol{\xi}_i^{(t)}) - f_i(\mathbf{v}, \boldsymbol{\xi}_i^{(t)})]^2 \leq 4G_i^2 \left\|\mathbf{u} - \mathbf{v}\right\|^2$$

where $\mathbb{E}_{\boldsymbol{\xi}_i^{(t)}}[.]$ denotes expectation w.r.t. sampling at time-step t for the node i. See [SCDB] for proof.

B.2.0.1 Proof of Theorem 4

We first establish a relationship between the primal, dual variables in Algorithm 3 and the function $\mathcal{H}$ defined in (B.10). This following lemma can be seen as a counterpart of Lemma 3 in the bandit feedback case.

Lemma 5. *Consider the update steps in Algorithm 3 with learning rate η. Under assumptions A.4-A.7, for any $\mathbf{x} \in \widetilde{\mathcal{X}}^n$ and $\boldsymbol{\lambda} \in \mathbb{R}^m$ with $\boldsymbol{\lambda} \succeq \mathbf{0}$, the summation of the function $\mathcal{H}$ (defined in (B.10)) satisfies:*

$$\sum_{t=1}^{T}\mathbb{E}\left[\mathcal{H}(\mathbf{x}^{(t)}, \boldsymbol{\lambda}) - \mathcal{H}(\mathbf{x}, \boldsymbol{\lambda}^{(t)})\right] \leq \frac{\eta}{2}\sum_{t=1}^{T}\mathbb{E}\left(2\|\mathbf{p}^{(t)}\|^2 + \|\nabla_{\boldsymbol{\lambda}}\widetilde{\mathcal{L}}_t(\mathbf{x}^{(t)}, \boldsymbol{\lambda}^{(t)})\|^2\right)$$
$$+\frac{1}{2\eta}\sum_{t=1}^{T}\left\|\mathbf{x}^{(t)} - \widetilde{\mathbf{x}}^{(t)}\right\|^2 + \frac{1}{2\eta}\sum_{t=1}^{T}\left(\left\|\boldsymbol{\lambda}\right\|^2 + 4R^2\right)$$

Now consider $\mathbf{x}^* \in \mathcal{X}^n$, then by definition of $\widetilde{\mathcal{X}}$, we have $(1-\alpha)\mathbf{x}^* + \alpha\tilde{\mathbf{y}}_0 \in \widetilde{\mathcal{X}}^n$ for $\alpha = \frac{\zeta}{r}$ where $\tilde{\mathbf{y}}_0$ and r are defined in Assumption A.8[1]. Substituting $\mathbf{x} = (1-\alpha)\mathbf{x}^* + \alpha\tilde{\mathbf{y}}_0$ in the result from Lemma 5 gives us:

$$\sum_{t=1}^{T} \mathbb{E}\left[\mathcal{H}(\mathbf{x}^{(t)}, \boldsymbol{\lambda}) - \mathcal{H}((1-\alpha)\mathbf{x}^* + \alpha\tilde{\mathbf{y}}_0, \boldsymbol{\lambda}^{(t)})\right] \leq \frac{\eta}{2}\sum_{t=1}^{T}\mathbb{E}\left(2\|\mathbf{p}^{(t)}\|^2 + \|\nabla_{\boldsymbol{\lambda}}\widetilde{\mathcal{L}}_t(\mathbf{x}^{(t)}, \boldsymbol{\lambda}^{(t)})\|^2\right)$$
$$+ \frac{1}{2\eta}\sum_{t=1}^{T}\left\|\mathbf{x}^{(t)} - \widetilde{\mathbf{x}}^{(t)}\right\|^2 + \frac{1}{2\eta}\sum_{t=1}^{T}\left(\|\boldsymbol{\lambda}\|^2 + 4R^2\right) \tag{B.11}$$

The following result bounds the error $\mathbb{E}\left\|\mathbf{e}^{(t)}\right\|^2 := \mathbb{E}\left\|\mathbf{x}^{(t)} - \widetilde{\mathbf{x}}^{(t)}\right\|^2$ for any time t in terms of the summation of $\mathbb{E}\left\|\mathbf{p}^{(t)}\right\|$; see [SCDB] for proof.

Lemma 6. *Consider the error* $\mathbf{e}^{(t)} := \mathbf{x}^{(t)} - \widetilde{\mathbf{x}}^{(t)}$ *for any* $t \in [T]$. *We have:*

$$\mathbb{E}\|\mathbf{e}^{(t)}\|^2 \leq \frac{2\eta^2}{\omega}\sum_{k=0}^{t-2}\left(1 - \frac{\omega}{2}\right)^k \mathbb{E}\left\|\mathbf{p}^{(t-k-1)}\right\|^2$$

Using the result from Lemma 6 in (B.11) and the double sum trick similar to the updates from (B.4) to (B.5) yields:

$$\sum_{t=1}^{T}\mathbb{E}\left[\mathcal{H}(\mathbf{x}^{(t)}, \boldsymbol{\lambda}) - \mathcal{H}((1-\alpha)\mathbf{x}^* + \alpha\tilde{\mathbf{y}}_0, \boldsymbol{\lambda}^{(t)})\right]$$
$$\leq \frac{\eta}{2}\sum_{t=1}^{T}\left(\left(2 + \frac{4}{\omega^2}\right)\|\mathbf{p}^{(t)}\|^2 + \|\nabla_{\boldsymbol{\lambda}}\widetilde{\mathcal{L}}_t(\mathbf{x}^{(t)}, \boldsymbol{\lambda}^{(t)})\|^2\right) + \frac{1}{2\eta}\mathbb{E}\left(4R^2 + \|\boldsymbol{\lambda}\|^2\right) \tag{B.12}$$

We now provide bounds for the first and second terms on the R.H.S. of (B.12) in Proposition 1 below. The proof of this proposition is provided in [SCDB].

Proposition 1. *For the update steps given in Algorithm 3, under Assumptions A.5-A.7, for any* $t \in [T]$, *we have:*

[1]Here, $\tilde{\mathbf{y}}_0 \in \mathbb{R}^{nd}$ denotes the stacking of the d dimensional vector $\mathbf{y}_0$ defined in Assumption A.8

1. $\mathbb{E}\left\|\mathbf{p}^{(t)}\right\|^2 \le 4d^2(1+m)G^2 + 4(1+m)\tilde{G}^2\mathbb{E}\left\|\boldsymbol{\lambda}^{(t)}\right\|^2$

2. $\mathbb{E}\left\|\nabla_{\boldsymbol{\lambda}}\widetilde{\mathcal{L}}_t(\mathbf{x}^{(t)},\boldsymbol{\lambda}^{(t)})\right\|^2 \le 2C^2 + 2\delta^2\eta^2\mathbb{E}\left\|\boldsymbol{\lambda}^{(t)}\right\|^2$

Substituting the bounds from Proposition 1 in (B.12) and using that fact $\frac{2}{\omega^2} > 1$, we have:

$$\sum_{t=1}^{T}\mathbb{E}\left[\mathcal{H}(\mathbf{x}^{(t)},\boldsymbol{\lambda}) - \mathcal{H}((1-\alpha)\mathbf{x}^* + \alpha\tilde{\mathbf{y}}_0,\boldsymbol{\lambda}^{(t)})\right] \le \frac{1}{2\eta}\left(4R^2 + \|\boldsymbol{\lambda}\|^2\right)$$
$$+\eta T\left[\frac{16}{\omega^2}d^2(1+m)G^2 + C^2\right] + \eta\left[\frac{16}{\omega^2}(1+m)\tilde{G}^2 + \delta^2\eta^2\right]\sum_{t=1}^{T}\mathbb{E}\left\|\boldsymbol{\lambda}^{(t)}\right\|^2 \quad \text{(B.13)}$$

We now express the L.H.S. of (B.13) in terms of the Lagrangian $\widetilde{\mathcal{L}}$. This relation is provided in Proposition 2 below, which is proved in [SCDB].

Proposition 2. *For any $\boldsymbol{\lambda} \in \mathbb{R}^m$ with $\boldsymbol{\lambda} \succeq \mathbf{0}$, the updates of Algorithm 3 satisfy:*

$$\sum_{t=1}^{T}\mathbb{E}\left[\mathcal{H}(\mathbf{x}^{(t)},\boldsymbol{\lambda}) - \mathcal{H}((1-\alpha)\mathbf{x}^* + \alpha\tilde{\mathbf{y}}_0,\boldsymbol{\lambda}^{(t)})\right] = \sum_{t=1}^{T}\mathbb{E}\left[\widetilde{\mathcal{L}}_t(\mathbf{x}^{(t)},\boldsymbol{\lambda}) - \widetilde{\mathcal{L}}_t((1-\alpha)\mathbf{x}^* + \alpha\tilde{\mathbf{y}}_0,\boldsymbol{\lambda}^{(t)})\right]$$

where $\mathbf{x}^*$ is the optimal parameter value for the objective (4.1)*, and $\mathcal{H}$, $\widetilde{\mathcal{L}}$ are defined in* (B.10) *and* (B.9)*, respectively.*

Proposition 2 implies the following for (B.13):

$$\sum_{t=1}^{T}\mathbb{E}\left[\widetilde{\mathcal{L}}_t(\mathbf{x}^{(t)},\boldsymbol{\lambda}) - \widetilde{\mathcal{L}}_t((1-\alpha)\mathbf{x}^* + \alpha\tilde{\mathbf{y}}_0,\boldsymbol{\lambda}^{(t)})\right] \le \frac{1}{2\eta}\left(4R^2 + \|\boldsymbol{\lambda}\|^2\right)$$
$$+\eta T\left[\frac{16}{\omega^2}d^2(1+m)G^2 + C^2\right] + \eta\left[\frac{16}{\omega^2}(1+m)\tilde{G}^2 + \delta^2\eta^2\right]\sum_{t=1}^{T}\mathbb{E}\left\|\boldsymbol{\lambda}^{(t)}\right\|^2$$

Using the definition of $\widetilde{\mathcal{L}}$ from (B.9) on the L.H.S. of the above, and rearranging the terms, we have:

$$\begin{aligned}
&\sum_{t=1}^{T}\mathbb{E}\left[\widetilde{f}(\mathbf{x}^{(t)},\boldsymbol{\xi}^{(t)})-\widetilde{f}((1-\alpha)\mathbf{x}^{*}+\alpha\tilde{\mathbf{y}}_0,\boldsymbol{\xi}^{(t)})\right]-\frac{\delta\eta T}{2}\left\|\boldsymbol{\lambda}\right\|^2\\
&+\left\langle\boldsymbol{\lambda},\mathbb{E}\sum_{t=1}^{T}\mathbf{g}(\mathbf{x}^{(t)})\right\rangle-\mathbb{E}\sum_{t=1}^{T}\left\langle\boldsymbol{\lambda}^{(t)},\mathbf{g}((1-\alpha)\mathbf{x}^{*}+\alpha\tilde{\mathbf{y}}_0)\right\rangle\\
&\leq\frac{1}{2\eta}\left(4R^2+\left\|\boldsymbol{\lambda}\right\|^2\right)+\eta T\left[\frac{16}{\omega^2}d^2(1+m)G^2+C^2\right]\\
&\quad+\eta\left[\frac{16}{\omega^2}(1+m)\tilde{G}^2+\delta^2\eta^2-\frac{\delta}{2}\right]\sum_{t=1}^{T}\mathbb{E}\left\|\boldsymbol{\lambda}^{(t)}\right\|^2 \qquad \text{(B.14)}
\end{aligned}$$

Similar to what we did for the sample feedback case in (B.6), we choose the following value of δ to make the coefficient of the last term in (B.14) negative:

$$\delta=\frac{1-\sqrt{1-\frac{256\eta^2(1+m)\tilde{G}^2}{\omega^2}}}{4\eta^2}$$

As before, we require running the algorithm for $T\geq\frac{256a^2(1+m)\tilde{G}^2}{\omega^2}$ for the choice $\eta=\frac{a}{\sqrt{T}}$, and for $T\to\infty$ (i.e., $\eta\to 0$), the above value of δ converges to a positive constant. Plugging the value of δ in (B.14) yields:

$$\begin{aligned}
&\sum_{t=1}^{T}\mathbb{E}\left[\widetilde{f}(\mathbf{x}^{(t)},\boldsymbol{\xi}^{(t)})-\widetilde{f}((1-\alpha)\mathbf{x}^{*}+\alpha\tilde{\mathbf{y}}_0,\boldsymbol{\xi}^{(t)})\right]-\frac{\delta\eta T}{2}\left\|\boldsymbol{\lambda}\right\|^2\\
&+\left\langle\boldsymbol{\lambda},\mathbb{E}\sum_{t=1}^{T}\mathbf{g}(\mathbf{x}^{(t)})\right\rangle-\mathbb{E}\left[\sum_{t=1}^{T}\left\langle\boldsymbol{\lambda}^{(t)},\mathbf{g}((1-\alpha)\mathbf{x}^{*}+\alpha\tilde{\mathbf{y}}_0)\right\rangle\right]\\
&\leq\frac{1}{2\eta}\left(4R^2+\left\|\boldsymbol{\lambda}\right\|^2\right)+\eta T\left[\frac{16}{\omega^2}d^2(1+m)G^2+C^2\right] \qquad \text{(B.15)}
\end{aligned}$$

Our goal is to derive a bound for the sub-optimality of the function $F(\mathbf{x}^{(t)})$. To this end, we will now bound the terms on the L.H.S. of (B.15) in terms of the function F. We first consider the first term on the L.H.S. of (B.15). From the definitions of f

and $\widetilde{f}$ provided in Fact 1:

$$\begin{aligned}
&\mathbb{E}\left[|\widetilde{f}(\mathbf{x}^{(t)}, \boldsymbol{\xi}^{(t)}) - f(\mathbf{x}^{(t)}, \boldsymbol{\xi}^{(t)})|\right] \\
&\overset{(a)}{=} \mathbb{E}\left[\left|\sum_{i=1}^{n} f_i(\mathbf{x}_i^{(t)} + \zeta \mathbf{u}_i^{(t)}, \boldsymbol{\xi}_i^{(t)}) - f_i(\mathbf{x}_i^{(t)}, \boldsymbol{\xi}_i^{(t)})\right|\right] \\
&\overset{(b)}{\leq} \mathbb{E}\left[\sum_{i=1}^{n} \left|f_i(\mathbf{x}_i^{(t)} + \zeta \mathbf{u}_i^{(t)}, \boldsymbol{\xi}_i^{(t)}) - f_i(\mathbf{x}_i^{(t)}, \boldsymbol{\xi}_i^{(t)})\right|\right] \\
&\overset{(c)}{\leq} \mathbb{E}\left[\sum_{i=1}^{n} \sqrt{\mathbb{E}_{\boldsymbol{\xi}_i^{(t)}}\left[f_i(\mathbf{x}_i^{(t)} + \zeta \mathbf{u}_i^{(t)}, \boldsymbol{\xi}_i^{(t)}) - f_i(\mathbf{x}_i^{(t)}, \boldsymbol{\xi}_i^{(t)})\right]^2}\right] \qquad \text{(B.16)}
\end{aligned}$$

where in (a), $\{\mathbf{u}_i^{(t)}\}_{i=1}^{n}$ denote random vectors uniformly distributed over $\mathbb{B}$, (b) uses the triangle inequality, (c) uses the fact $\mathbb{E}[A] \leq \sqrt{\mathbb{E}[A^2]}$ via Jensen's inequality. From Proposition 9 and the fact $\|\mathbf{u}_i^{(t)}\|^2 = 1$ for all $i \in [n]$ (as they lie on the unit sphere $\mathbb{S}$), we have:

$$\mathbb{E}_{\boldsymbol{\xi}_i^{(t)}}[f_i(\mathbf{x}_i^{(t)} + \zeta \mathbf{u}_i^{(t)}, \boldsymbol{\xi}_i^{(t)}) - f_i(\mathbf{x}_i^{(t)}, \boldsymbol{\xi}_i^{(t)})]^2 \leq 4G_i^2\zeta^2 \qquad \text{(B.17)}$$

Plugging the bound from (B.17) in (B.16) and noting that $\sum_{i=1}^{n} G_i \leq \sqrt{n}G$ (using the fact that $G^2 = \sum_{i=1}^{n} G_i^2$):

$$\mathbb{E}\left[|\widetilde{f}(\mathbf{x}^{(t)}, \boldsymbol{\xi}^{(t)}) - f(\mathbf{x}^{(t)}, \boldsymbol{\xi}^{(t)})|\right] \leq 2\zeta\sqrt{n}G$$

Using Jensen's inequality for the L.H.S. of above equation and rearranging the terms finally yields:

$$\mathbb{E}[\widetilde{f}(\mathbf{x}^{(t)}, \boldsymbol{\xi}^{(t)})] \geq \mathbb{E}[F(\mathbf{x}^{(t)})] - 2\zeta\sqrt{n}G \qquad \text{(B.18)}$$

The steps to bound the second term on the L.H.S. of (B.15) are similar. We note that:

$$\begin{aligned}&\mathbb{E}\left[|\widetilde{f}((1-\alpha)\mathbf{x}^* + \alpha\tilde{\mathbf{y}}_0, \boldsymbol{\xi}^{(t)}) - f(\mathbf{x}^*, \boldsymbol{\xi}^{(t)})|\right] \\ &\leq \mathbb{E}\sum_{i=1}^{n}\left(\mathbb{E}_{\boldsymbol{\xi}_i^{(t)}}\left[f_i((1-\alpha)\mathbf{x}_i^* + \alpha\mathbf{y}_0 + \zeta\mathbf{u}_i^{(t)}, \boldsymbol{\xi}^{(t)}) - f_i(\mathbf{x}_i^*, \boldsymbol{\xi}_i^{(t)})\right]^2\right)^{1/2}\end{aligned} \tag{B.19}$$

where the inequality follows the same arguments we used for arriving at (B.16). Further using Proposition 9, we have:

$$\mathbb{E}_{\boldsymbol{\xi}_i^{(t)}}\left[f_i((1-\alpha)\mathbf{x}_i^* + \alpha\mathbf{y}_0 + \zeta\mathbf{u}_i^{(t)}, \boldsymbol{\xi}^{(t)}) - f_i(\mathbf{x}_i^*, \boldsymbol{\xi}_i^{(t)})\right]^2 \leq 4G_i^2\left\|-\alpha\mathbf{x}_i^* + \alpha\mathbf{y}_0 + \zeta\mathbf{u}_i^{(t)}\right\|^2 \tag{B.20}$$

Plugging in the bound from (B.20) into (B.19), using Fact 6 for $\mathbf{x}_i^*, \mathbf{y}_0 \in \mathcal{X}$ along with $\left\|\mathbf{u}_i^{(t)}\right\| = 1$ for all $i \in [n]$, and Jensen's inequality, we have:

$$\mathbb{E}[\widetilde{f}((1-\alpha)\mathbf{x}^* + \alpha\tilde{\mathbf{y}}_0, \boldsymbol{\xi}^{(t)})] \leq F(\mathbf{x}^*) + 4G\alpha R + 2\zeta G\sqrt{n} \tag{B.21}$$

Further, we can also simplify other terms on the L.H.S. of (B.15). We note that:

$$\begin{aligned}\sum_{t=1}^{T}\left\langle\boldsymbol{\lambda}^{(t)}, \mathbf{g}((1-\alpha)\mathbf{x}^*+\alpha\tilde{\mathbf{y}}_0)\right\rangle &= \sum_{t=1}^{T}\left\langle\boldsymbol{\lambda}^{(t)}, \mathbf{g}(\mathbf{x}^*)\right\rangle + \sum_{t=1}^{T}\left\langle\boldsymbol{\lambda}^{(t)}, \mathbf{g}((1-\alpha)\mathbf{x}^*+\alpha\tilde{\mathbf{y}}_0) - \mathbf{g}(\mathbf{x}^*)\right\rangle \\ &\leq \sum_{t=1}^{T}\left\|\boldsymbol{\lambda}^{(t)}\right\|\left\|\mathbf{g}((1-\alpha)\mathbf{x}^*+\alpha\tilde{\mathbf{y}}_0) - \mathbf{g}(\mathbf{x}^*)\right\|\end{aligned} \tag{B.22}$$

where to obtain the last inequality we have used the fact that $\langle\boldsymbol{\lambda}^{(t)}, \mathbf{g}(\mathbf{x}^*)\rangle \leq 0$ for all $t \in [T]$ and the Cauchy-Schwarz inequality. For the second term in the product on the R.H.S. in (B.22), using (4.7) in Assumption 6 $g(\mathbf{x}_i, \mathbf{x}_j)$ are G_{ij}-Lipschitz for all

$i, j \in [n]$, we have:

$$\|\mathbf{g}((1-\alpha)\mathbf{x}^*+\alpha\tilde{\mathbf{y}}_0) - \mathbf{g}(\mathbf{x}^*)\|^2 \leq \sum_{i=1}^{n}\sum_{j\in\mathcal{N}_i} G_{ij}^2 \|-\alpha\mathbf{x}^*+\alpha\tilde{\mathbf{y}}_0\|^2 \leq 4\alpha^2R^2m\tilde{G}^2 \quad \text{(B.23)}$$

where $\tilde{G} := \max_{i\in[n],j\in\mathcal{N}_i} G_{ij}$ and the last inequality follows from noting that $\mathbf{x}^*, \tilde{\mathbf{y}}_0 \in \mathcal{X}^n$ and using Fact 6. We now bound the first term in the product on the R.H.S. in (B.22). From the update equation of $\boldsymbol{\lambda}^{(t)}$ in line 13 of Algorithm 3, we have:

$$\left\|\boldsymbol{\lambda}^{(t+1)}\right\| \leq \left\|\boldsymbol{\lambda}^{(t)} + \eta\nabla_{\boldsymbol{\lambda}}\widetilde{\mathcal{L}}_t(\mathbf{x}^{(t)}, \boldsymbol{\lambda}^{(t)})\right\| \leq (1-\delta\eta^2)\left\|\boldsymbol{\lambda}^{(t)}\right\| + \eta C$$

where the second inequality follows from the gradient update for the dual variable (4.5), the triangle inequality, the fact that $\delta\eta^2 \leq 1$ (since an upper bound for δ is $\frac{1}{4\eta^2}$) and Assumption 7 to bound $\left\|\mathbf{g}(\mathbf{x}^{(t)})\right\|_2$. Continuing the recursion till $t = 1$, it can be shown that $\left\|\boldsymbol{\lambda}^{(t)}\right\| \leq \frac{C}{\delta\eta}$, $\forall t \in [T]$. Using this bound, and (B.23) in (B.22) leads to:

$$\sum_{t=1}^{T}\left\langle\boldsymbol{\lambda}^{(t)}, \mathbf{g}((1-\alpha)\mathbf{x}^*+\alpha\tilde{\mathbf{y}}_0)\right\rangle \leq \frac{2\alpha RC\sqrt{m}\tilde{G}T}{\delta\eta} \quad \text{(B.24)}$$

Finally, using bounds from (B.18), (B.21), (B.24) in (B.15) yields:

$$\begin{aligned}
&\sum_{t=1}^{T}\mathbb{E}\left[F(\mathbf{x}^{(t)}) - F(\mathbf{x}^*)\right] - \frac{\delta\eta T}{2}\|\boldsymbol{\lambda}\|^2 + \left\langle\boldsymbol{\lambda}, \mathbb{E}\sum_{t=1}^{T}\mathbf{g}(\mathbf{x}^{(t)})\right\rangle - \mathbb{E}\left[\sum_{t=1}^{T}\left\langle\boldsymbol{\lambda}^{(t)}, \mathbf{g}((1-\alpha)\mathbf{x}^*+\alpha\tilde{\mathbf{y}}_0)\right\rangle\right] \\
&\leq \frac{1}{2\eta}\left(4R^2 + \|\boldsymbol{\lambda}\|^2\right) + \eta T\left[\frac{16}{\omega^2}d^2(1+m)G^2 + C^2\right] \\
&\quad + 4G\alpha RT + 4\zeta G\sqrt{n}T + \frac{2\alpha RC\sqrt{m}\tilde{G}T}{\delta\eta} \qquad \text{(B.25)}
\end{aligned}$$

Setting $\boldsymbol{\lambda} = \frac{\left[\mathbb{E}\left[\sum_{t=1}^{T} \mathbf{g}(\mathbf{x}^{(t)})\right]\right]^{+}}{\delta\eta T + \frac{1}{\eta}}$ in (B.25) gives:

$$\begin{aligned}
&\sum_{t=1}^{T} \left(\mathbb{E}\left[F(\mathbf{x}^{(t)})\right] - F(\mathbf{x}^{*})\right) + \sum_{i=1}^{n} \sum_{j \in \mathcal{N}_i} \frac{\left(\left[\mathbb{E}\left[\sum_{t=1}^{T} g_{ij}(\mathbf{x}_i^{(t)}, \mathbf{x}_j^{(t)})\right]\right]^{+}\right)^2}{2\left(\delta\eta T + \frac{1}{\eta}\right)} \\
&\leq \frac{2R^2}{\eta} + \eta T \left[\frac{16}{\omega^2} d^2 (1+m) G^2 + C^2\right] + \frac{CT2\sqrt{m}\tilde{G}\alpha R}{\delta\eta} \\
&\qquad + 4\alpha RGT + 4\zeta\sqrt{n}GT
\end{aligned} \tag{B.26}$$

Dividing both sides of (B.26) by T and noting that the second term on the L.H.S. of (B.26) is positive, we can bound the time-average sub-optimality of F as:

$$\begin{aligned}
\sum_{t=1}^{T} \frac{\left(\mathbb{E}\left[F(\mathbf{x}^{(t)})\right] - F(\mathbf{x}^{*})\right)}{T} &\leq \frac{2R^2}{\eta T} + \eta \left[\frac{16}{\omega^2} d^2 (1+m) G^2\right] \\
&\quad + C^2\eta + 2\sqrt{m}\tilde{G}\alpha R \frac{C}{\delta\eta} + 4\alpha RG + 4\zeta\sqrt{n}G
\end{aligned}$$

Using the convexity of F and setting the values $\eta = \frac{a}{\sqrt{T}}$, $\zeta = \frac{1}{T}$ and $\alpha = \frac{1}{rT}$ for some positive constant a, r, concludes the proof for the suboptimality of the function F given in (4.13) of Theorem 4. We now consider the expected constraint violations. From Fact 8, we have that $\forall \mathbf{x} \in \mathcal{X}^n$, $\mathbb{E}[F(\mathbf{x})] - F(\mathbf{x}^{*}) > -4GR$. Using this relation in (B.26) gives:

$$\begin{aligned}
&\sum_{i=1}^{n} \sum_{j \in \mathcal{N}_i} \left(\left[\mathbb{E}\left[\sum_{t=1}^{T} g_{ij}(\mathbf{x}_i^{(t)}, \mathbf{x}_j^{(t)})\right]\right]^{+}\right)^2 \\
&\leq \frac{4R^2}{\eta^2} + T\left[\left(\frac{32}{\omega^2} d^2(1+m)G^2 + 2C^2\right) + \frac{4\sqrt{m}\tilde{G}\alpha RC}{\delta\eta^2} + \frac{8(\alpha R + \zeta\sqrt{n})G}{\eta} + 4R^2\delta + \frac{8GR}{\eta}\right] \\
&\qquad + T^2\left[\delta\eta^2\left(\frac{32}{\omega^2} d^2(1+m)G^2 + 2C^2\right) + 4\sqrt{m}\tilde{G}\alpha RC + 8\delta\eta(\alpha R + \zeta\sqrt{n})G + 8GR\delta\eta\right]
\end{aligned}$$

Note that the above bound also holds for a given $i \in [n]$ and $j \in \mathcal{N}_i$, that is, the R.H.S. of the above equation is also a bound for the term $\left(\left[\mathbb{E}\left[\sum_{t=1}^{T} g_{ij}(\mathbf{x}_i^{(t)}, \mathbf{x}_j^{(t)})\right]\right]^+\right)^2$. Taking the square root of both sides and using the fact $\sqrt{\sum_{i=1}^{n} c_i} \leq \sum_{i=1}^{n} \sqrt{c_i}$ for positive $c_1, \ldots, c_n$, we get:

$$\begin{aligned}
\mathbb{E}\left[\sum_{t=1}^{T} g_{ij}(\mathbf{x}_i^{(t)}, \mathbf{x}_j^{(t)})\right] &\leq \frac{2R}{\eta} \\
&+ \sqrt{T}\left[\left(\frac{32}{\omega^2} d^2(1+m)G^2 + 2C^2\right) + 4\sqrt{m}\tilde{G}\alpha R \frac{C}{\delta\eta^2} + \frac{8(\alpha R + \zeta\sqrt{n})G}{\eta} + 4R^2\delta + \frac{8GR}{\eta}\right]^{1/2} \\
&\quad + T\left[\delta\eta^2\left(\frac{32}{\omega^2} d^2(1+m)G^2 + 2C^2\right) + 4\sqrt{m}\tilde{G}\alpha RC + 8\delta\eta(\alpha R + \zeta\sqrt{n})G + 8GR\delta\eta\right]^{1/2}
\end{aligned}$$

Dividing both sides of the above by T, using the convexity of constraint function g_{ij}, and substituting the values $\eta = \frac{a}{\sqrt{T}}$, $\zeta = \frac{1}{T}$ and $\alpha = \frac{1}{rT}$ concludes proof of (4.14).$\square$

Appendix C

Omitted Details for Chapter 5

C.1 Proof for Phase 1 training

We now provide the proof for Theorem 5 which establishes a generalization bound for the learned model formed by leveraging the pre-trained source models.
We first note the following results from [DHK$^+$21] that will enable us to prove Theorem 5 later in Section C.1.2 (and Theorem 6 in Section C.2).

Claim 2 (Covariance of Source distribution, Claim A.1 of [DHK$^+$21])**.** *Suppose $n_s \gg \rho^4(d + \log(m/\delta))$ for $\delta \in (0,1)$. Then with probability at least $1 - \frac{\delta}{10}$ over the inputs $\mathbf{X}_1, \dots, \mathbf{X}_m$ in the source tasks, for all $i \in [m]$ we have*

$$0.9\boldsymbol{\Sigma}_i \preceq \frac{1}{n_s}\mathbf{X}_i^\top \mathbf{X}_i \preceq 1.1\boldsymbol{\Sigma}_i$$

Claim 3 (Covariance Target distribution, Claim A.2 of [DHK$^+$21])**.** *Suppose $n_T \gg \rho^4(k + \log(1/\delta))$ for $\delta \in (0,1)$. The for any given matrix $\mathbf{B} \in \mathbb{R}^{d \times 2k}$ that is independent of $\mathbf{X}_T$, with probability at least $1 - \frac{\delta}{20}$ over target data X_T, we have*

$$0.9\mathbf{B}^\top \boldsymbol{\Sigma}_T \mathbf{B} \preceq \frac{1}{n_T}\mathbf{B}^\top \mathbf{X}_T^\top \mathbf{X}_T \mathbf{B} \preceq 1.1\mathbf{B}^\top \boldsymbol{\Sigma}_T \mathbf{B}$$

Proposition 3 (Lemma A.7 from [DHK$^+$21])**.** *For matrices* $\mathbf{A}_1, \mathbf{A}_2$ *(with same number of columns) such that* $\mathbf{A}_1^\top \mathbf{A}_1 \succeq \mathbf{A}_2^\top \mathbf{A}_2$ *and for matrices* $\mathbf{B}_1, \mathbf{B}_2$ *of compatible dimensions, we have:*

$$\left\| \mathbf{P}^\perp_{\mathbf{A}_1\mathbf{B}_1} \mathbf{A}_1 \mathbf{B}_2 \right\|_F^2 \geq \left\| \mathbf{P}^\perp_{\mathbf{A}_2\mathbf{B}_1} \mathbf{A}_2 \mathbf{B}_2 \right\|_F^2$$

Proposition 4. *Consider matrices* $\mathbf{A} \in \mathbb{R}^{a\times b}$ *and* $\mathbf{B} \in \mathbb{R}^{b\times c}$*. Then for any* $\mathbf{u} \in \mathbb{R}^a$*, we have:*

$$\|\mathbf{P}^\perp_{\mathbf{A}} \mathbf{u}\|_2^2 \leq \|\mathbf{P}^\perp_{\mathbf{AB}} \mathbf{u}\|_2^2$$

Proof. For given $\mathbf{A} \in \mathbb{R}^{a\times b}$ and $\mathbf{B} \in \mathbb{R}^{b\times c}$ and $\mathbf{u} \in \mathbb{R}^a$, we have:

$$\begin{aligned}
\|\mathbf{P}^\perp_{\mathbf{A}} \mathbf{u}\|_2^2 &= \min_{\mathbf{r}\in\mathbb{R}^b} \|\mathbf{Ar} - \mathbf{u}\|_2^2 \\
&\leq \min_{\mathbf{r}\in\mathcal{C}(\mathbf{B})} \|\mathbf{Ar} - \mathbf{u}\|_2^2 \\
&= \min_{\mathbf{s}\in\mathbb{R}^c} \|\mathbf{ABs} - \mathbf{u}\|_2^2 \\
&= \|\mathbf{P}^\perp_{\mathbf{AB}} \mathbf{u}\|_2^2
\end{aligned}$$

□

C.1.1 Some important results

We now provide proof of results used to establish the resulting bound for Phase 1 training provided in Theorem 5, which is proved later in Section C.1.2. These results provide guarantees on empirical training of the source models (c.f. Lemma 7) as well as the performance of empirically learned source representations on the target data (c.f. Lemma 1).

These results would also be useful for the proof of Theorem 6 presented later in Section C.2.

C.1.1.1 Proof of Lemma 1

We first prove Lemma 1 which establishes a bound on using the learned empirical representation $\widehat{\mathbf{V}}$ on the target data.

[Restating Lemma 1] Consider the matrix $\widehat{\mathbf{V}} \in \mathbb{R}^{d\times q}$ formed by empirical source representations $\{\widehat{\mathbf{B}}_i\}$ obtained from solving (5.3) and the matrix $\mathbf{V}^* \in \mathbb{R}^{d\times l}$ formed from the true representations $\{\mathbf{B}_i^*\}$. For any $\mathbf{b} \in \mathbb{R}^l$ such that $\|\mathbf{b}\|_2 = 1$, with probability at-least $1-\delta_1$, we have:

$$\min_{\mathbf{u}\in\mathbb{R}^q} \left\|\mathbf{X}_T\widehat{\mathbf{V}}\mathbf{u} - \mathbf{X}_T\mathbf{V}^*\mathbf{b}\right\|_2 \le \frac{\sigma^2 n_T}{r n_S}\left(km + kdm\log(\kappa n_s) + \log\left(\frac{1}{\delta_1}\right)\right)$$

Proof. We first note that:

$$\left\|\mathbf{P}^{\perp}_{\mathbf{X}_T\widehat{\mathbf{V}}}\mathbf{X}_T\mathbf{V}^*\mathbf{b}\right\|_2 := \min_{\mathbf{u}\in\mathbb{R}^q}\left\|\mathbf{X}_T\widehat{\mathbf{V}}\mathbf{u} - \mathbf{X}_T\mathbf{V}^*\mathbf{b}\right\|_2$$

Using the fact that $\{\widetilde{\mathbf{w}}_i\}$ span the space $\mathbb{R}^l$, we can write $\mathbf{b} = \widetilde{\mathbf{W}}^*\boldsymbol{\alpha}$ for some $\boldsymbol{\alpha} \in \mathbb{R}^m$ where $\boldsymbol{\alpha}$ is $\mathcal{O}(1)$. We have:

$$\begin{aligned}
\left\|\mathbf{P}^{\perp}_{\mathbf{X}_T\widehat{\mathbf{V}}}\mathbf{X}_T\mathbf{V}^*\mathbf{b}\right\|_2^2 &= \left\|\mathbf{P}^{\perp}_{\mathbf{X}_T\widehat{\mathbf{V}}}\mathbf{X}_T\mathbf{V}^*\widetilde{\mathbf{W}}^*\boldsymbol{\alpha}\right\|_2^2 \\
&\lesssim \left\|\mathbf{P}^{\perp}_{\mathbf{X}_T\widehat{\mathbf{V}}}\mathbf{X}_T\mathbf{V}^*\widetilde{\mathbf{W}}^*\right\|_F^2 \\
&= \sum_{i=1}^m \left\|\mathbf{P}^{\perp}_{\mathbf{X}_T\widehat{\mathbf{V}}}\mathbf{X}_T\mathbf{V}^*\widetilde{\mathbf{w}}_i^*\right\|_2^2 \\
&\overset{(a)}{\lesssim} n_T\sum_{i=1}^m \left\|\mathbf{P}^{\perp}_{\mathbf{\Sigma}_T^{1/2}\widehat{\mathbf{V}}}\mathbf{\Sigma}_T^{1/2}\mathbf{V}^*\widetilde{\mathbf{w}}_i^*\right\|_2^2
\end{aligned} \tag{C.1}$$

$$\overset{(b)}{\lesssim} \frac{n_T}{r} \sum_{i=1}^{m} \left\| \mathbf{P}^{\perp}_{\mathbf{\Sigma}_i^{1/2}\widehat{\mathbf{V}}} \mathbf{\Sigma}_i^{1/2} \mathbf{V}^* \widetilde{\mathbf{w}}_i^* \right\|_2^2$$
$$\overset{(c)}{\lesssim} \frac{n_T}{r n_S} \sum_{i=1}^{m} \left\| \mathbf{P}^{\perp}_{\mathbf{X}_i\widehat{\mathbf{V}}} \mathbf{X}_i \mathbf{V}^* \widetilde{\mathbf{w}}_i^* \right\|_2^2$$

where (a) follows from Claim 3 (with $\mathbf{B} = [\widehat{\mathbf{V}} \quad -\mathbf{V}^*]$), (b) follows from Assumption 11 and (c) from Claim 2. We now note that $\mathbf{V}^*\widetilde{\mathbf{w}}_i^* = \mathbf{B}_i^*\mathbf{w}_i^*$. We now note that $\widehat{\mathbf{V}}$ is the matrix whose columns are an orthonormal basis of the set of columns of the matrices $\{\widehat{\mathbf{B}}_i\}$. Thus for each $i \in [m]$, there exists a matrix $\mathbf{C}_i$ such that $\widehat{\mathbf{B}}_i = \widehat{\mathbf{V}}\mathbf{C}_i$. Now using the result of Proposition 4 we have:

$$\begin{aligned}
\left\| \mathbf{P}^{\perp}_{\mathbf{X}_T\widehat{\mathbf{V}}} \mathbf{X}_T \mathbf{V}^* \mathbf{b} \right\|_2^2 &\overset{Prop.\ 4}{\lesssim} \frac{n_T}{r n_s} \sum_{i=1}^{m} \left\| \mathbf{P}^{\perp}_{\mathbf{X}_i\widehat{\mathbf{B}}_i} \mathbf{X}_i \mathbf{B}_i^* \mathbf{w}_i^* \right\|_2^2 \\
&\leq \frac{n_T}{r n_s} \sum_{i=1}^{m} \left\| \mathbf{P}_{\mathbf{X}_i\widehat{\mathbf{B}}_i} (\mathbf{X}_i \mathbf{B}_i^* \mathbf{w}_i^* + \mathbf{z}_i) - \mathbf{X}_i \mathbf{B}_i^* \mathbf{w}_i^* \right\|_2^2 \\
&= \frac{n_T}{r n_s} \sum_{i=1}^{m} \left\| \mathbf{P}_{\mathbf{X}_i\widehat{\mathbf{B}}_i} \mathbf{y}_i - \mathbf{X}_i \mathbf{B}_i^* \mathbf{w}_i^* \right\|_2^2 \\
&= \frac{n_T}{r n_s} \sum_{i=1}^{m} \left\| \mathbf{X}_i \widehat{\mathbf{B}}_i \widehat{\mathbf{w}}_i - \mathbf{X}_i \mathbf{B}_i^* \mathbf{w}_i^* \right\|_2^2
\end{aligned}$$

The proof can then be concluded by using the result from Lemma 7 stated below that provides a bound for the trained empirical source models in (5.3). □

The following lemma establishes guarantees on the learned empirical source representations.

Lemma 7 (Multi-Source training guarantee)**.** *With probability at least* $1 - \frac{\delta}{5}$, *we have:*

$$\sum_{i=1}^{m} \left\| \mathbf{X}_i (\widehat{\mathbf{B}}_i \widehat{\mathbf{w}}_i - \mathbf{B}_i^* \mathbf{w}_i^*) \right\|_2^2 \leq \sigma^2 \left(km + kdm \log(\kappa n_s) + \log\left(\frac{1}{\delta}\right) \right)$$

First we instantiate Claim 2, which happens with probability atleast $1 - \frac{\delta}{10}$. For the given source datasets and matrix $\mathbf{A} \in \mathbb{R}^{d\times m}$ with columns $\{\mathbf{a}_i\}$, we define the map $\mathcal{X}(\mathbf{A})$ as $\mathcal{X}(\mathbf{A}) = [\mathbf{X}_1\mathbf{a}_1 \quad \mathbf{X}_2\mathbf{a}_2 \dots \quad \mathbf{X}_m\mathbf{a}_m]$. Now consider the matrix $\boldsymbol{\Delta} \in \mathbb{R}^{d\times m}$ whose columns are given by $\{\widehat{\mathbf{B}}_i\widehat{\mathbf{w}}_i - \mathbf{B}_i^*\mathbf{w}_i^*\}_{i=1}^m$. For convenience of notation, we define $\mathcal{X}(\boldsymbol{\Delta}) := [\mathbf{X}_1(\widehat{\mathbf{B}}_1\widehat{\mathbf{w}}_1 - \mathbf{B}_1^*\mathbf{w}_1^*) \quad \dots \quad \mathbf{X}_m(\widehat{\mathbf{B}}_m\widehat{\mathbf{w}}_m - \mathbf{B}_m^*\mathbf{w}_m^*)]$. We are interested in providing a bound for the quantity $|\mathcal{X}(\boldsymbol{\Delta})_F^2$. The i^{th} column of the matrix $\boldsymbol{\Delta}$ can be decomposed as $\mathbf{R}_i\mathbf{r}_i$ where $\mathbf{R}_i \in \mathcal{O}_{d\times 2k}$ (set of tall orthonormal matrices in $d \times 2k$) and $\mathbf{r}_i \in \mathbb{R}^{2k}$.

$$\boldsymbol{\Delta} = [\mathbf{R}_1\mathbf{r}_1 \quad \mathbf{R}_2\mathbf{r}_2 \dots \quad \mathbf{R}_m\mathbf{r}_m]$$

For each $i \in [m]$, we now decompose $\mathbf{X}_i\mathbf{R}_i = \mathbf{U}_i\mathbf{Q}_i$ (where $\mathbf{U}_i \in \mathcal{O}_{n_s\times 2k}$ and $\mathbf{Q} \in \mathbb{R}^{2k\times 2k}$). Since $\{\widehat{\mathbf{B}}_i, \widehat{\mathbf{w}}_i\}_{i=1}^m$ are the optimal solutions for the source regression problems, we have $\sum_{i=1}^m \|\mathbf{y}_i - \mathbf{X}_i\widehat{\mathbf{B}}_i\widehat{\mathbf{w}}_i\|_2^2 \leq \sum_{i=1}^m \|\mathbf{y}_i - \mathbf{X}_i\mathbf{B}_i^*\mathbf{w}_i^*\|_2^2$. Substituting $\mathbf{y}_i = \mathbf{X}_i\mathbf{B}_i^*\mathbf{w}_i^* + \mathbf{z}_i$ for $i \in [m]$, we get $\|\mathcal{X}(\boldsymbol{\Delta})\|_F^2 \leq 2\langle \mathbf{Z}, \mathcal{X}(\boldsymbol{\Delta})\rangle$ (where the inner product of matrices is trace of their product) and we denote the matrix of noise vectors as $\mathbf{Z} := [\mathbf{z}_1 \quad \mathbf{z}_2 \quad \dots \quad \mathbf{z}_m] \in \mathbb{R}^{n_s\times m}$. Now:

$$\begin{aligned}
\langle \mathbf{Z}, \mathcal{X}(\boldsymbol{\Delta})\rangle &= \sum_{i=1}^m \mathbf{z}_i^\top \mathbf{X}_i\mathbf{R}_i\mathbf{r}_i = \sum_{i=1}^m \mathbf{z}_i^\top \mathbf{U}_i\mathbf{Q}_i\mathbf{r}_i && \text{(C.2)}\\
&\leq \sum_{i=1}^m \left\|\mathbf{U}_i^\top \mathbf{z}_i\right\|_2 \|\mathbf{Q}_i\mathbf{r}_i\|_2 \\
&\leq \sqrt{\sum_{i=1}^m \left\|\mathbf{U}_i^\top \mathbf{z}_i\right\|_2} \sqrt{\sum_{i=1}^m \|\mathbf{U}_i\mathbf{Q}_i\mathbf{r}_i\|_2} \\
&= \sqrt{\sum_{i=1}^m \left\|\mathbf{U}_i^\top \mathbf{z}_i\right\|_2} \, \|\mathcal{X}(\boldsymbol{\Delta})\|_F && \text{(C.3)}
\end{aligned}$$

We will now provide a bound for the first term in the product on the R.H.S. of (C.3). Since $\mathbf{U}_i$ depends on $\mathbf{R}_i$, it also depends on the value of $\mathbf{Z}$. To provide a bound, we use an ϵ-net argument to cover all possible values of $\{\mathbf{R}_i\}_{i=1}^m$. We first consider a fixed set of matrices $\{\bar{\mathbf{R}}_i\}_{i=1}^m \subset \mathcal{O}_{d\times 2k}^m$. For these given matrices, we can find decompose $\mathbf{X}_i\bar{\mathbf{R}}_i = \bar{\mathbf{U}}_i\mathbf{Q}_i$ for $i \in [m]$, where $\{\bar{\mathbf{U}}_i\}_{i=1}^m \subset \mathcal{O}_{n_T\times 2k}^m$ do not depend on $\mathbf{Z}$. Thus we have $\frac{1}{\sigma^2}\sum_{i=1}^m \left\|\mathbf{U}_i^\top \mathbf{z}_i\right\|_2 \sim \chi^2(2km)$. Thus w.p at least $1-\delta'$, we have:

$$\sum_{i=1}^m \left\|\mathbf{U}_i^\top \mathbf{z}_i\right\|_2 \lesssim \sigma^2\left(km + \log\left(\frac{1}{\delta'}\right)\right) \tag{C.4}$$

Hence for the given $\{\bar{\mathbf{R}}_i\}_{i=1}^m$, using the result from (C.4) in (C.3), we have:

$$\left\langle \mathbf{Z}, \mathcal{X}(\bar{\mathbf{\Delta}})\right\rangle \lesssim \sigma^2\left(km + \log\left(\frac{1}{\delta'}\right)\right)\|\mathcal{X}(\bar{\mathbf{\Delta}})\|_F$$

where $\bar{\mathbf{\Delta}} = [\bar{\mathbf{R}}_1\mathbf{r}_1 \quad \bar{\mathbf{R}}_2\mathbf{r}_2 \ldots \quad \bar{\mathbf{R}}_m\mathbf{r}_m]$. Now we consider an $\frac{\epsilon}{m}$-net of $\mathcal{O}_{d\times 2k}^m$ denoted by $\mathcal{N}$ of size $|\mathcal{N}| \le \left(\frac{6m\sqrt{2k}}{\epsilon}\right)^{2kdm}$. Using the union bound, with probability at least $1-|\mathcal{N}|\delta'$:

$$< \mathbf{Z}, \mathcal{X}(\bar{\mathbf{\Delta}}) > \lesssim \sigma^2\left(km + \log\left(\frac{1}{\delta'}\right)\right)\|\mathcal{X}(\bar{\mathbf{\Delta}})\|_F,\ \forall\{\bar{\mathbf{R}}_i\}_{i=1}^m \subset \mathcal{N} \tag{C.5}$$

Choose $\delta' = \frac{\delta}{20\left(\frac{6m\sqrt{2k}}{\epsilon}\right)^{2kdm}}$, then the above holds with probability at least $1-\frac{\delta}{20}$. We will now use the results from the following claim, which is proved in Section C.1.1.2 below.

Claim 4. *Under the assumptions of Theorem 5, the following hold:*

1. *W.p at least* $1-\frac{\delta}{20}$, $\|\mathbf{Z}\|_F^2 \lesssim \sigma^2\left(n_s m + \log\left(\frac{1}{\delta}\right)\right)$
2. *If the result in 1) holds and Claim 2 holds , then* $\|\mathbf{\Delta}\|_F^2 \lesssim \frac{\sigma^2(n_s m + \log(\frac{1}{\delta}))}{n_s\lambda_{low}}$ *where*

$\lambda_{low} = \min_{i\in[m]} \lambda_{\min}(\mathbf{\Sigma}_i)$

3. *If the results in 1), 2) above hold and Claim 2 holds, then* $\|\mathcal{X}([\mathbf{R}_1\mathbf{r}_1 \quad \dots \mathbf{R}_m\mathbf{r}_m] - [\bar{\mathbf{R}}_1\mathbf{r}_1 \quad \dots \bar{\mathbf{R}}_m\mathbf{r}_m])\| \lesssim \frac{\kappa\epsilon^2}{m^2}\sigma^2\left(n_s m + \log\left(\frac{1}{\delta}\right)\right)$ *for some* $\{\bar{\mathbf{R}}_i\} \subset \mathcal{N}$ *where* $\kappa = \frac{\max_{i\in[m]}\lambda_{\min}(\mathbf{\Sigma}_i)}{\min_{i\in[m]}\lambda_{\min}(\mathbf{\Sigma}_i)}$

where (a) follows from (C.5) w.p $\geq 1 - \frac{\delta}{20}$, (b) from 1) in Claim 4); w.p. $\geq 1 - \frac{\delta}{20}$ and (c) uses the fact that $\delta' < \delta, k \leq n_s$, and 3) in Claim 4. Since the above result gives an inequality in terms of $\|\mathcal{X}(\mathbf{\Delta})\|_F^2$ and $\|\mathcal{X}(\mathbf{\Delta})\|_F$, we can conclude the following:

$$\|\mathcal{X}(\mathbf{\Delta})\|_F \lesssim \max\left\{\sigma\sqrt{\left(km + \log\left(\frac{1}{\delta'}\right)\right)}, \sigma\sqrt{\frac{\sqrt{\kappa}\epsilon}{m}\left(n_s m + \log\left(\frac{1}{\delta'}\right)\right)}\right\}$$

We choose $\epsilon = \frac{km}{n_s\sqrt{\kappa}}$, and note that $n_S \gg k$, which gives:

$$\|\mathcal{X}(\mathbf{\Delta})\|_F \leq \sigma\sqrt{\left(km + \log\left(\frac{1}{\delta'}\right)\right)}$$

Substituting the value of $\delta' = \frac{\delta}{20\left(\frac{6m\sqrt{2k}}{\epsilon}\right)^{2kdm}}$ and $\epsilon = \frac{km}{n_s\sqrt{\kappa}}$:

$$\|\mathcal{X}(\mathbf{\Delta})\|_F \lesssim \sigma\sqrt{km + kdm\log\left(\frac{mk}{\epsilon}\right) + \log\left(\frac{1}{\delta}\right)}\sigma\sqrt{km + kdm\log\left(n_s\kappa\right) + \log\left(\frac{1}{\delta}\right)}$$

Hence the following holds with probability at least 1 - $\left(\frac{\delta}{10} + \frac{\delta}{20} + \frac{\delta}{20}\right)$:

$$\|\mathcal{X}(\mathbf{\Delta})\|_F^2 \lesssim \sigma^2\left[km + kdm\log\left(n_s\kappa\right) + \log\left(\frac{1}{\delta}\right)\right]$$

C.1.1.2 Proof of Claim 4

1. This follows from the fact that $\frac{1}{\sigma^2}\|\mathbf{Z}\|_F^2 \sim \chi(n_s m)$

2.

$$\begin{aligned}
\|\mathcal{X}(\mathbf{\Delta})\|_F^2 &= \sum_{i=1}^m \|\mathbf{X}_i(\widehat{\mathbf{B}}_i\widehat{\mathbf{w}}_{i_0} - \mathbf{B}_i^*\mathbf{w}_i^*)\|_2^2 \\
&= \sum_{i=1}^m (\widehat{\mathbf{B}}_i\widehat{\mathbf{w}}_{i_0} - \mathbf{B}_i^*\mathbf{w}_i^*)^\top \mathbf{X}_i^\top \mathbf{X}_i(\widehat{\mathbf{B}}_i\widehat{\mathbf{w}}_{i_0} - \mathbf{B}_i^*\mathbf{w}_i^*) \\
&\gtrsim n_s \sum_{i=1}^m (\widehat{\mathbf{B}}_i\widehat{\mathbf{w}}_{i_0} - \mathbf{B}_i^*\mathbf{w}_i^*)^\top \mathbf{\Sigma}_i(\widehat{\mathbf{B}}_i\widehat{\mathbf{w}}_{i_0} - \mathbf{B}_i^*\mathbf{w}_i^*) \\
&\geq n_s \sum_{i=1}^m \lambda_{\min}(\mathbf{\Sigma}_i)\|(\widehat{\mathbf{B}}_i\widehat{\mathbf{w}}_{i_0} - \mathbf{B}_i^*\mathbf{w}_i^*)\|_2^2 \\
&\geq n_s\lambda_{low}\|\mathbf{\Delta}\|_F^2
\end{aligned}$$

where $\lambda_{low} := \min_{i\in[m]}\lambda_{\min}(\mathbf{\Sigma}_i)$. Since $\|\mathcal{X}(\mathbf{\Delta})\|_F^2 \leq 2\langle \mathbf{Z}, \mathcal{X}(\mathbf{\Delta})\rangle \leq 2\|\mathbf{Z}\|_F\|\mathcal{X}(\mathbf{\Delta})\|_F$, we have $\|\mathcal{X}(\mathbf{\Delta})\|_F \leq 2\|\mathbf{Z}\|_F$. Using the result from part 1. of the claim statement combined with the upper bound derived above, we have:

$$\|\mathbf{\Delta}\|_F^2 \lesssim \frac{\sigma^2}{n_s\lambda_{low}}\left(n_s m + \log\left(\frac{1}{\delta}\right)\right)$$

3. For some $\{\bar{\mathbf{R}}_i\} \subset \mathcal{N}$ we have $\sum_{i=1}^m \|\mathbf{R}_i - \bar{\mathbf{R}}_i\|_F \leq \sum_{i=1}^m \frac{\epsilon}{m} = \epsilon$. Therefore:

$$\begin{aligned}
\|\mathcal{X}(\mathbf{\Delta}-\bar{\mathbf{\Delta}})\|_F^2 &= \sum_{i=1}^m \|\mathbf{X}_i(\mathbf{R}_i - \bar{\mathbf{R}}_i)\mathbf{r}_i\|_2^2 \\
&\leq \sum_{i=1}^m \|\mathbf{X}_i\|_2^2\|\mathbf{R}_i - \bar{\mathbf{R}}_i\|_F^2\|\mathbf{r}_i\|_2^2 \\
&\lesssim \frac{n_s\epsilon^2}{m^2}\sum_{i=1}^m \|\mathbf{\Sigma}_i\|_2^2\|\mathbf{r}_i\|_2^2
\end{aligned}$$

$$
\begin{aligned}
&\lesssim \frac{n_s\epsilon^2\lambda_{high}}{m^2}\sum_{i=1}^{m}\|\mathbf{r}_i\|_2^2 \\
&\overset{(a)}{=} \frac{n_s\epsilon^2\lambda_{high}}{m^2}\|\boldsymbol{\Delta}\|_F^2 \\
&\overset{(b)}{\lesssim} \frac{n_s\epsilon^2\lambda_{high}}{m^2}\frac{\sigma^2(n_s m+\log(\frac{1}{\delta}))}{n_s\lambda_{low}} \\
&= \frac{\kappa\epsilon^2\lambda_{high}\sigma^2}{m^2\lambda_{low}}\left(n_s m+\log\left(\frac{1}{\delta}\right)\right)
\end{aligned}
$$

Here, $\lambda_{high} := \max_{i\in[m]}\lambda_{\max}(\boldsymbol{\Sigma}_i)$ where to arrive at (a), we have used the fact that $\{\mathbf{R}_i\}$ have orthonormal columns and used the definition of $\boldsymbol{\Delta}$, and (b) follows from using 2) from the claim statement.

C.1.2 Proof of Theorem 5

Having established the helper results above, we now provide a proof for Theorem 5.

[Restating Theorem 5] Fix a failure probability $\delta \in (0,1)$ and further assume $2k \leq \min\{d, m\}$ and the number of samples in the sources and target satisfy $n_s \gg \rho^4(d+\log(m/\delta))$ and $n_{T_1} \gg \rho^4(\max\{l,q\}+\log(1/\delta))$, respectively. Define $\kappa = \frac{\max_{i\in[m]}\lambda_{\max}(\boldsymbol{\Sigma}_i)}{\min_{i\in[m]}\lambda_{\min}(\boldsymbol{\Sigma}_i)}$ where $\lambda_{\max}(\boldsymbol{\Sigma}_i)$ denotes the maximum eigenvalue of $\boldsymbol{\Sigma}_i$. Then with probability at least $1-\delta$ over the samples, under Assumptions 9 - 13, the expected excess risk of the learned predictor $\widehat{\mathbf{w}}_T$ on the target $(\mathbf{x} \to \mathbf{x}^\top\widehat{\mathbf{V}}\widehat{\mathbf{w}}_T)$ for Phase 1 satisfies:

$$
\mathbb{E}[\text{EER}(\boldsymbol{\theta}_{\text{Phase}_1}, \boldsymbol{\theta}_T^*)] \lesssim \frac{\sigma^2}{n_{T_1}}(q+\log(1/\delta)) + \epsilon^2 + \sigma^2\left[\frac{1}{rn_s m}\log\left(\frac{1}{\delta}\right) + \left(\frac{kd\log(\kappa n_s)+k}{rn_s}\right)\right]
$$

Proof. We will first instantiate Lemma 2. We then instantiate Lemma 3 twice, once with $[\widehat{\mathbf{V}} - \mathbf{V}^*]$ and the other time with $[\mathbf{B}_T^* - \mathbf{V}^*]$. Then we assume that the result from Lemma 7 holds. All these events happen together with probability at least

$1 - \frac{2\delta}{5}$. The expected error for the target distribution is given by:

$$\begin{aligned}
&\mathrm{EER}(\boldsymbol{\theta}_{\mathrm{Phase}_1}, \boldsymbol{\theta}_T^*) = \mathbb{E}_{\mathbf{x}\sim p_T}\left[\mathbf{x}^\top \mathbf{B}_T^* \mathbf{w}_T^* - \mathbf{x}^\top \widehat{\mathbf{V}} \widehat{\mathbf{w}}_T\right]^2 \\
&= \left\| \boldsymbol{\Sigma}_T^{1/2} \left(\widehat{\mathbf{V}} \widehat{\mathbf{w}}_T - \mathbf{B}_T^* \mathbf{w}_T^* \right) \right\|_2^2 \\
&\lesssim \left\| \boldsymbol{\Sigma}_T^{1/2} \left(\widehat{\mathbf{V}} \widehat{\mathbf{w}}_T - \mathbf{V}^* \widetilde{\mathbf{w}}_T^* \right) \right\|_2^2 + \left\| \boldsymbol{\Sigma}_T^{1/2} \left(\mathbf{B}_T^* \mathbf{w}_T^* - \mathbf{V}^* \widetilde{\mathbf{w}}_T^* \right) \right\|_2^2 \\
&\overset{(a)}{\leq} \left\| \boldsymbol{\Sigma}_T^{1/2} \left(\widehat{\mathbf{V}} \widehat{\mathbf{w}}_T - \mathbf{V}^* \widetilde{\mathbf{w}}_T^* \right) \right\|_2^2 + \epsilon^2 \\
&\overset{(b)}{\lesssim} \frac{1}{n_T} \left\| \mathbf{X}_T \left(\widehat{\mathbf{V}} \widehat{\mathbf{w}}_T - \mathbf{V}^* \widetilde{\mathbf{w}}_T^* \right) \right\|_2^2 + \epsilon^2 \\
&= \frac{1}{n_T} \left\| \mathbf{P}_{\mathbf{X}_T \widehat{\mathbf{V}}} \mathbf{y}_T - \mathbf{X}_T \mathbf{V}^* \widetilde{\mathbf{w}}_T^* \right\|_2^2 + \epsilon^2 \\
&\lesssim \frac{1}{n_T} \left\| \mathbf{P}_{\mathbf{X}_T \widehat{\mathbf{V}}} (\mathbf{X}_T \mathbf{V}^* \widetilde{\mathbf{w}}_T^* + \mathbf{z}_T) - \mathbf{X}_T \mathbf{V}^* \widetilde{\mathbf{w}}_T^*) \right\|_2^2 \\
&\qquad + \frac{1}{n_T} \left\| \mathbf{P}_{\mathbf{X}_T \widehat{\mathbf{V}}} (\mathbf{X}_T \mathbf{B}_T^* \mathbf{w}_T^* - \mathbf{X}_T \mathbf{V}^* \widetilde{\mathbf{w}}_T^*) \right\|_2^2 + \epsilon^2
\end{aligned}$$

where (a) follows from Assumption 9 and (b) uses Claim 3. Using the fact that $\|\mathbf{P}_{\mathbf{X}_T \widehat{\mathbf{V}}}\|_2 \leq 1$ (since $\mathbf{P}_{\mathbf{X}_T \widehat{\mathbf{V}}}$ is a projection matrix) and using Claim 3, we have:

$$\begin{aligned}
&\lesssim \frac{1}{n_T} \left\| \mathbf{P}_{\mathbf{X}_T \widehat{\mathbf{V}}} (\mathbf{X}_T \mathbf{V}^* \widetilde{\mathbf{w}}_T^* + \mathbf{z}_T) - \mathbf{X}_T \mathbf{V}^* \widetilde{\mathbf{w}}_T^*) \right\|_2^2 + \epsilon^2 \\
&\lesssim \frac{1}{n_T} \left\| \mathbf{P}^\perp_{\mathbf{X}_T \widehat{\mathbf{V}}} \mathbf{X}_T \mathbf{V}^* \widetilde{\mathbf{w}}_T^* \right\|_2^2 + \epsilon^2 + \frac{1}{n_T} \left\| \mathbf{P}_{\mathbf{X}_T \widehat{\mathbf{V}}} \mathbf{z}_T \right\|_2^2
\end{aligned}$$

where the first inequality follows from Assumption 9 and Claim 3. We can take the expectation over the distribution of $\mathbf{w}_T^*$ and use Assumption 13 to yield:

$$\mathbb{E}_{\widetilde{\mathbf{w}}_T^*}[\mathrm{EER}(\boldsymbol{\theta}_{\mathrm{Phase}_1}, \boldsymbol{\theta}_T^*)] \lesssim \frac{1}{n_T l} \left\| \mathbf{P}^\perp_{\mathbf{X}_T \widehat{\mathbf{V}}} \mathbf{X}_T \mathbf{V}^* \right\|_F^2 + \epsilon^2 + \frac{1}{n_T} \left\| \mathbf{P}_{\mathbf{X}_T \widehat{\mathbf{V}}} \mathbf{z}_T \right\|_2^2 \tag{C.6}$$

We now make use of the following lemma, which is proved below in Section C.1.2.1 that provides a bound for the first term in (C.6).

Lemma 8 (Target Training Guarantee)**.** *Assuming the results in Claim 2, Claim 3 (with* $\mathbf{B} = [\widehat{\mathbf{V}} \quad -\mathbf{V}^*]$*) and Lemma 7 hold, we then have:*

$$\left\|\mathbf{P}^{\perp}_{\mathbf{X}_T\widehat{\mathbf{V}}}\mathbf{X}_T\mathbf{V}^*\right\|_F^2 \lesssim \frac{n_T\sigma^2}{rn_s\sigma_l^2(\widetilde{\mathbf{W}}^*)}\left(km+kdm\log(\kappa n_s)+\log\left(\frac{1}{\delta}\right)\right)$$

Substituting the result from Lemma 8 in (C.6) and using $\sigma_l^2(\widetilde{\mathbf{W}}^*) \geq \frac{m}{l}$, the following bound holds with probability at least $1-\frac{2\delta}{5}$:

$$\mathbb{E}_{\widetilde{\mathbf{w}}_T^*}[\mathrm{EER}(\boldsymbol{\theta}_{\mathrm{Phase}_1},\boldsymbol{\theta}_T^*)] \lesssim \frac{1}{n_T}\left\|\mathbf{P}_{\mathbf{X}_T\widehat{\mathbf{V}}}\mathbf{z}_T\right\|_2^2 + \epsilon^2 + \frac{1}{rn_sm}\sigma^2\left(km+kdm\log(\kappa n_s)+\log\left(\frac{1}{\delta}\right)\right)$$

Finally, the last term in above can be bounded by using a concentration inequality for χ^2-squared distribution. In particular, with probability at least $1-\frac{3\delta}{5}$ we have $\left\|\mathbf{P}_{\mathbf{X}_T\widehat{\mathbf{V}}}\mathbf{z}_T\right\|_2^2 \lesssim \sigma^2(q+\log(1/\delta))$. Thus the following bound holds on $\mathbb{E}_{\mathbf{w}_T^*}[\mathrm{Err}(\widehat{\mathbf{B}}_S,\mathbf{w}_T^*)]$ with probability at least $1-\delta$:

$$\mathbb{E}_{\mathbf{w}_T^*}[\mathrm{EER}(\boldsymbol{\theta}_{\mathrm{Phase}_1},\boldsymbol{\theta}_T^*)] \lesssim \epsilon^2 + \frac{1}{n_T}\sigma^2(q+\log(1/\delta)) + \frac{1}{rn_sm}\sigma^2\left(km+kdm\log(\kappa n_s)+\log\left(\frac{1}{\delta}\right)\right)$$
$$= \sigma^2\left[\frac{1}{rn_sm}\log\left(\frac{1}{\delta}\right)+\left(\frac{kd\log(\kappa n_s)+k}{rn_s}\right)\right]+\frac{\sigma^2}{n_T}(q+\log(1/\delta))+\epsilon^2$$

$\square$

C.1.2.1 Proof of Lemma 8

We start the proof by using Proposition 3 and Claim 3:

$$\sigma_l^2(\widetilde{\mathbf{W}}^*)\|\mathbf{P}^{\perp}_{\mathbf{X}_T\widehat{\mathbf{V}}}\mathbf{X}_T\mathbf{V}^*\|_F^2 \lesssim \sigma_l^2(\widetilde{\mathbf{W}}^*)n_T\left\|\mathbf{P}^{\perp}_{\boldsymbol{\Sigma}_T^{1/2}\widehat{\mathbf{V}}}\boldsymbol{\Sigma}_T^{1/2}\mathbf{V}^*\right\|_F^2 n_T\sum_{i=1}^{m}\left\|\mathbf{P}^{\perp}_{\boldsymbol{\Sigma}_T^{1/2}\widehat{\mathbf{V}}}\boldsymbol{\Sigma}_T^{1/2}\mathbf{V}^*\widetilde{\mathbf{w}}_i^*\right\|_2^2$$

The proof now follows the same procedure as in Proof of Lemma 1 starting from Equation C.1 and following it up with Lemma 7.

C.2 Proof for Phase 2 training

We now provide a proof for our bound in Theorem 6 which establishes excess generalization risk for the model obtained after combined Phase 1 and Phase 2 training.

The data for Phase 2 training is given by $(\mathbf{X}_{T_2}, \mathbf{y}_{T_2})$ where $\mathbf{y}_{T_2} = \mathbf{X}_{T_2}\mathbf{B}_T^*\mathbf{w}_T^* + \mathbf{z}_{T_2}$. Here $\boldsymbol{\theta}_T^* := \mathbf{B}_T^*\mathbf{w}^*$ denotes the true data generating target model. We define $\mathbf{P}_{\parallel} = \mathbf{X}_{T_2}^{\top}(\mathbf{X}_{T_2}\mathbf{X}_{T_2}^{\top})^{-1}\mathbf{X}_{T_2}$ as projection matrix on the row-space of matrix $\mathbf{X}_{T_2}$ and $\mathbf{P}_{\perp} = \mathbf{I} - \mathbf{P}_{\parallel}$.

We first note the following result that establishes where the Gradient Descent solution converges to, the proof of which is given in Section C.2.1.1 below.

Lemma 9. *Under the assumptions of Theorem 6, performing gradient descent on the objective* (5.6) *with the initialization* $\boldsymbol{\theta}^{(0)} := \boldsymbol{\theta}_{Phase_1}$ *and learning rate* η*, yields the solution:*

$$\boldsymbol{\theta}_{GD} := \boldsymbol{\theta}^{(\infty)} = \mathbf{P}_{\parallel}\boldsymbol{\theta}_T^* + \mathbf{P}_{\perp}\boldsymbol{\theta}_{Phase_1} + \mathbf{X}_{T_2}^{\top}(\mathbf{X}_{T_2}\mathbf{X}_{T_2}^{\top})^{-1}\mathbf{z}$$

where $\mathbf{P}_{\parallel} = \mathbf{X}_{T_2}^{\top}(\mathbf{X}_{T_2}\mathbf{X}_{T_2}^{\top})^{-1}\mathbf{X}_{T_2}$ *is the projection matrix on the row-space of matrix*

$\mathbf{X}_{T_2}$ *and* $\mathbf{P}_\perp = \mathbf{I} - \mathbf{P}_\parallel$.

C.2.1 Proof of Theorem 6

We now use the solution of the gradient descent $\hat{\boldsymbol{\theta}}_T := \boldsymbol{\theta}_{GD}$ derived in Lemma 9 and find the excess population risk. The excess risk is given by:

$$\begin{aligned}
\mathrm{EER}(\hat{\boldsymbol{\theta}}_T, \boldsymbol{\theta}_T^*) &= \mathbb{E}_{\mathbf{X}_{T_2}\sim p_T}(\mathbf{X}_{T_2}^\top \boldsymbol{\theta}_T^* - \mathbf{X}_{T_2}^\top \boldsymbol{\theta}_{GD})^2 \\
&= \mathbb{E}_{\mathbf{X}_{T_2}\sim p_T} \mathrm{Tr}[(\boldsymbol{\theta}_T^* - \boldsymbol{\theta}_{GD})^\top \mathbf{X}_{T_2}\mathbf{X}_{T_2}^\top(\boldsymbol{\theta}_T^* - \boldsymbol{\theta}_{GD})] \\
&= \left\| \boldsymbol{\Sigma}_T^{1/2}(\boldsymbol{\theta}_T^* - \boldsymbol{\theta}_{GD}) \right\|_2^2
\end{aligned}$$

Substituting the value of $\boldsymbol{\theta}_{GD}$ from Lemma 9 we get:

$$\mathrm{EER}(\hat{\boldsymbol{\theta}}_T, \boldsymbol{\theta}_T^*) \le 2\left\| \boldsymbol{\Sigma}_T^{1/2}\mathbf{P}_\perp(\boldsymbol{\theta}_T^* - \boldsymbol{\theta}_{T_0}) \right\|_2^2 + 2\left\| \boldsymbol{\Sigma}_T^{1/2}\mathbf{X}_{T_2}^\top(\mathbf{X}_{T_2}\mathbf{X}_{T_2}^\top)^{-1}\mathbf{z} \right\|_2^2 \tag{C.7}$$

We focus on the first term for now. We have:

$$\begin{aligned}
&\left\| \boldsymbol{\Sigma}_T^{1/2}\mathbf{P}_\perp(\boldsymbol{\theta}_T^* - \boldsymbol{\theta}_{T_0}) \right\|_2^2 = (\boldsymbol{\theta}_T^* - \boldsymbol{\theta}_{T_0})^\top \mathbf{P}_\perp^\top \boldsymbol{\Sigma}_T \mathbf{P}_\perp (\boldsymbol{\theta}_T^* - \boldsymbol{\theta}_{T_0}) \\
&= (\boldsymbol{\theta}_T^* - \boldsymbol{\theta}_{T_0})^\top \mathbf{P}_\perp^\top \left(\boldsymbol{\Sigma}_T - \frac{1}{n_T}\mathbf{X}_{T_2}^\top \mathbf{X}_{T_2} \right) \mathbf{P}_\perp (\boldsymbol{\theta}_T^* - \boldsymbol{\theta}_{T_0}) \\
&= \left\| \left(\boldsymbol{\Sigma}_T - \frac{1}{n_T}\mathbf{X}_{T_2}^\top \mathbf{X}_{T_2} \right)^{1/2} \mathbf{P}_\perp (\boldsymbol{\theta}_T^* - \boldsymbol{\theta}_{T_0}) \right\|_2^2 \\
&\le \left\| \boldsymbol{\Sigma}_T - \frac{1}{n_T}\mathbf{X}_{T_2}^\top \mathbf{X}_{T_2} \right\|_2 \left\| \mathbf{P}_\perp (\boldsymbol{\theta}_T^* - \boldsymbol{\theta}_{T_0}) \right\|_2^2
\end{aligned} \tag{C.8}$$

The second term in (C.8) can be bounded by $\|\mathbf{P}_\perp(\boldsymbol{\theta}_T^* - \boldsymbol{\theta}_{T_0})\|_2^2 \leq \|(\boldsymbol{\theta}_T^* - \boldsymbol{\theta}_{T_0})\|_2^2$ by noting that $\|\mathbf{P}_\perp\|_2 \leq 1$. We thus finally get the bound:

$$\mathrm{EER}(\hat{\boldsymbol{\theta}}_T, \boldsymbol{\theta}_T^*) \leq 2\left\|\boldsymbol{\Sigma}_T - \frac{1}{n_T}\mathbf{X}_{T_2}^\top \mathbf{X}_{T_2}\right\|_2 \|(\boldsymbol{\theta}_T^* - \boldsymbol{\theta}_{T_0})\|_2^2 + 2\left\|\boldsymbol{\Sigma}_T^{1/2}\mathbf{X}_{T_2}^\top(\mathbf{X}_{T_2}\mathbf{X}_{T_2}^\top)^{-1}\mathbf{z}\right\|_2^2 \tag{C.9}$$

We now provide a bound for the second term in (C.9). Note:

$$\left\|\boldsymbol{\Sigma}_T^{1/2}\mathbf{X}_{T_2}^\top(\mathbf{X}_{T_2}\mathbf{X}_{T_2}^\top)^{-1}\mathbf{z}\right\|_2^2 = \mathbf{z}^\top(\mathbf{X}_{T_2}\mathbf{X}_{T_2}^\top)^{-1}\mathbf{X}_{T_2}\boldsymbol{\Sigma}_T\mathbf{X}_{T_2}^\top(\mathbf{X}_{T_2}\mathbf{X}_{T_2}^\top)^{-1}\mathbf{z}$$

From [BLLT20, Lemma 9], we can get a high probability bound (probability $> 1-e^{-t}$) on this term for some $t > 0$ as

$$\left\|\boldsymbol{\Sigma}_T^{1/2}\mathbf{X}_{T_2}^\top(\mathbf{X}_{T_2}\mathbf{X}_{T_2}^\top)^{-1}\mathbf{z}\right\|_2^2 \leq (4t+2)\sigma^2\mathrm{Tr}\left((\mathbf{X}_{T_2}\mathbf{X}_{T_2}^\top)^{-1}\mathbf{X}_{T_2}\boldsymbol{\Sigma}_T\mathbf{X}_{T_2}^\top(\mathbf{X}_{T_2}\mathbf{X}_{T_2}^\top)^{-1}\right)$$

To bound the trace term, we use [BLLT20, Lemma 13, 18]: For universal constant $b, c \geq 1$ and $k^* := \min\{k \geq 0 : r_k(\boldsymbol{\Sigma}_T) \geq bn\}$, we have

$$\mathbf{Tr}\left((\mathbf{X}_{T_2}\mathbf{X}_{T_2}^\top)^{-1}\mathbf{X}_{T_2}\boldsymbol{\Sigma}_T\mathbf{X}_{T_2}^\top(\mathbf{X}_{T_2}\mathbf{X}_{T_2}^\top)^{-1}\right) \leq c\left(\frac{k^*}{bn} + \frac{bn}{R_{k^*}(\boldsymbol{\Sigma}_T)}\right)$$

where $r_k(\boldsymbol{\Sigma}_T) = \frac{\boldsymbol{\Sigma}_{i>k}\lambda_i}{\lambda_{k+1}}$, $R_{k^*}(\boldsymbol{\Sigma}_T) = \frac{(\boldsymbol{\Sigma}_{i>k}\lambda_i)^2}{\boldsymbol{\Sigma}_{i>k}\lambda_i^2}$.Substituting the value of $t = \log\left(\frac{2}{\delta}\right)$ Plugging the resulting bound in (C.9), we can finally claim that the following holds with probability at least $1 - \frac{\delta}{2}$:

$$\mathrm{EER}(\hat{\boldsymbol{\theta}}_T, \boldsymbol{\theta}_T^*) \lesssim \left\|\boldsymbol{\Sigma}_T - \frac{1}{n_T}\mathbf{X}_{T_2}^\top \mathbf{X}_{T_2}\right\|_2 \|(\boldsymbol{\theta}_T^* - \boldsymbol{\theta}_{T_0})\|_2^2 + c\sigma^2\log\left(\frac{1}{\delta}\right)\left(\frac{k^*}{bn} + \frac{bn}{R_{k^*}(\boldsymbol{\Sigma}_T)}\right)$$

The covariance approximation error in the first term of above can be bounded by the result in [KL17, Theorem 9]. This yields the following bound on the approximation with probability at least $1-e^{-\delta_1}$ over the choice of data matrix $\mathbf{X}_{T_2}$ for some constant $u > 0$ and $\delta_1 > 1$

$$\left\|\boldsymbol{\Sigma}_T - \frac{1}{n_T}\mathbf{X}_{T_2}^\top \mathbf{X}_{T_2}\right\|_2 \leq u\lambda_1 \max\left\{\sqrt{\frac{\sum_{i=1}^d \lambda_i}{n_T\lambda_1}}, \frac{\sum_{i=1}^d \lambda_i}{n_T\lambda_1}, \sqrt{\frac{\delta_1}{n_T}}, \frac{\delta_1}{n_T}\right\}$$

Substituting $\delta_1 = \log\left(\frac{2}{\delta}\right)$, with probability at least $1 - \frac{\delta}{2}$,

$$\left\|\boldsymbol{\Sigma}_T - \frac{1}{n_T}\mathbf{X}_{T_2}^\top \mathbf{X}_{T_2}\right\|_2 \leq u\lambda_1 \max\left\{\sqrt{\frac{\sum_{i=1}^d \lambda_i}{n_T\lambda_1}}, \frac{\sum_{i=1}^d \lambda_i}{n_T\lambda_1}, \sqrt{\frac{1}{n_T}\log\left(\frac{1}{\delta}\right)}, \frac{1}{n_T}\log\left(\frac{1}{\delta}\right)\right\} \tag{C.10}$$

Denote the eigenvalues of covariance matrix of the target data as $\{\lambda_i\}_{i=1}^d$, with $\lambda_1 \geq \ldots \lambda_d$, we then have:

$$\text{EER}(\hat{\boldsymbol{\theta}}_T, \boldsymbol{\theta}_T^*) \lesssim \left\|\boldsymbol{\Sigma}_T - \frac{1}{n_{T_2}}\mathbf{X}_{T_2}^\top \mathbf{X}_{T_2}\right\|_2 \frac{1}{\lambda_d}\left\|\boldsymbol{\Sigma}_T(\boldsymbol{\theta}_T^* - \boldsymbol{\theta}_{T_0})\right\|_2^2 + c\sigma^2 \log\left(\frac{1}{\delta}\right)\left(\frac{k^*}{bn_{T_2}} + \frac{bn_{T_2}}{R_{k^*}(\boldsymbol{\Sigma}_T)}\right)$$

where $r_k(\boldsymbol{\Sigma}_T) = \frac{\boldsymbol{\Sigma}_{i>k}\lambda_i}{\lambda_{k+1}}$, $R_{k^*}(\boldsymbol{\Sigma}_T) = \frac{(\boldsymbol{\Sigma}_{i>k}\lambda_i)^2}{\boldsymbol{\Sigma}_{i>k}\lambda_i^2}$. Here, constant $b > 1$ and $k^* = \min\{k \geq 0 : r_k(\boldsymbol{\Sigma}) \geq bn\}$ and the covariance estimation term can be bounded by $\left\|\boldsymbol{\Sigma}_T - \frac{1}{n_{T_2}}\mathbf{X}_{T_2}^\top \mathbf{X}_{T_2}\right\|_2 \leq u\lambda_1 \max\left\{\sqrt{\frac{\sum_{i=1}^d \lambda_i}{n_{T_2}\lambda_1}}, \frac{\sum_{i=1}^d \lambda_i}{n_{T_2}\lambda_1}, \sqrt{\frac{1}{n_{T_2}}\log\left(\frac{1}{\delta}\right)}, \frac{1}{n_{T_2}}\log\left(\frac{1}{\delta}\right)\right\}$ with probability at least $1 - \frac{\delta}{2}$. We now substitute the value of $\|\boldsymbol{\Sigma}_T(\boldsymbol{\theta}_T^* - \boldsymbol{\theta}_{T_0})\|_2^2$ from Theorem 5 (as $\boldsymbol{\theta}_{T_0} = \widehat{\mathbf{V}}\widehat{\mathbf{w}}_T$ which was obtained by using n_{T_1} target samples). Thus the final bound after Phase 1 and Phase 2 training, after taking the expectation w.r.t the target model $\boldsymbol{\theta}_T^*$, is given by:

$$\mathbb{E}[\mathrm{EER}(\hat{\boldsymbol{\theta}}_T, \boldsymbol{\theta}_T^*)] \lesssim \left\| \boldsymbol{\Sigma}_T - \frac{1}{n_{T_2}} \mathbf{X}_{T_2}^\top \mathbf{X}_{T_2} \right\|_2 \frac{\sigma^2}{\lambda_d} \frac{1}{r n_s m} \log\left(\frac{1}{\delta}\right) + c\sigma^2 \log\left(\frac{1}{\delta}\right) \left(\frac{k^*}{b n_{T_2}} + \frac{b n_{T_2}}{R_{k^*}(\boldsymbol{\Sigma}_T)}\right)$$
$$+ \left\| \boldsymbol{\Sigma}_T - \frac{1}{n_{T_2}} \mathbf{X}_{T_2}^\top \mathbf{X}_{T_2} \right\|_2 \frac{\sigma^2}{\lambda_d} \left(\frac{kd\log(\kappa n_s) + k}{r n_s}\right)$$
$$+ \left\| \boldsymbol{\Sigma}_T - \frac{1}{n_{T_2}} \mathbf{X}_{T_2}^\top \mathbf{X}_{T_2} \right\|_2 \frac{1}{\lambda_d} \left(\sigma^2 \left[\frac{1}{n_{T_1}}(q + \log(1/\delta))\right] + \epsilon^2\right)$$

where u, c are universal constants. We now substitute the bound for the covariance estimate from (C.10) and simplfy the expression by assuming $\frac{r_0(\boldsymbol{\Sigma}_T)}{n_{T_2}} \geq \frac{1}{n_{T_2}} \log\left(\frac{1}{\delta}\right) \geq 1$ since we have a few target samples:

$$\mathbb{E}[\mathrm{EER}(\hat{\boldsymbol{\theta}}_T, \boldsymbol{\theta}_T^*)] \leq \frac{u\lambda_1}{\lambda_d} \frac{r_0(\boldsymbol{\Sigma}_T)}{n_{T_2}} \left(\frac{\sigma^2}{n_{T_1}}(q + \log(1/\delta)) + \epsilon^2\right) + c\sigma^2 \log\left(\frac{1}{\delta}\right) \left(\frac{k^*}{b n_{T_2}} + \frac{b n_{T_2}}{R_{k^*}(\boldsymbol{\Sigma}_T)}\right)$$
$$+ \frac{u\lambda_1 \sigma^2}{\lambda_d} \frac{r_0(\boldsymbol{\Sigma}_T)}{n_{T_2}} \left(\frac{1}{r n_s m} \log\left(\frac{1}{\delta}\right) + \left(\frac{kd\log(\kappa n_s) + k}{r n_s}\right)\right)$$

C.2.1.1 Proof of Lemma 9

For any time step t of the gradient descent process, the gradient of the objective in (5.6) evaluated at $\boldsymbol{\theta}^{(t)}$ is given by:

$$\nabla f(\boldsymbol{\theta}^{(t)}) = \frac{2}{n_T} \mathbf{X}_{T_2}^\top (\mathbf{X}_{T_2} \boldsymbol{\theta}^{(t)} - \mathbf{y}_{T_2})$$

The gradient update step using step size η is given by:

$$\boldsymbol{\theta}^{(t+1)} = \boldsymbol{\theta}^{(t)} - \frac{2\eta}{n_T} \mathbf{X}_{T_2}^\top (\mathbf{X}_{T_2} \boldsymbol{\theta}^{(t)} - \mathbf{y}_{T_2}) = \boldsymbol{\theta}_{T_0} + \mathbf{X}_{T_2}^\top \mathbf{a}^{(t)}$$

where $\mathbf{a}$ is some vector in $\mathbb{R}^{n_T}$. In the limit, the gradient descent convergence to a solution of the form:

$$\boldsymbol{\theta}^{(\infty)} = \boldsymbol{\theta}_{T_0} + \mathbf{X}_{T_2}^{\top}\mathbf{a} \tag{C.11}$$

Since the problem in (5.6) is over-parameterized, there exists a value $\boldsymbol{\theta}^*$ such that $f(\boldsymbol{\theta}^*) = 0$. This follows from the fact that $\mathbf{X}_{T_2}$ has full row rank. The gradient descent solution, under an appropriate choice of the learning rate, thus converges to this value while yield a zero loss, implying:

$$\begin{aligned}
&\mathbf{X}_{T_2}\boldsymbol{\theta}^{(\infty)} = \mathbf{y}_{T_2} = \mathbf{X}_{T_2}\boldsymbol{\theta}_T^* + \mathbf{z}_{T_2} \\
\Rightarrow &\mathbf{X}_{T_2}(\boldsymbol{\theta}_{T_0} + \mathbf{X}_{T_2}^{\top}\mathbf{a}) = \mathbf{X}_{T_2}\boldsymbol{\theta}_T^* + \mathbf{z}_{T_2} \\
\Rightarrow &\mathbf{a} = (\mathbf{X}_{T_2}\mathbf{X}_{T_2}^{\top})^{-1}(\mathbf{X}_{T_2}(\boldsymbol{\theta}_T^* - \boldsymbol{\theta}_{T_0}) + \mathbf{z}_{T_2})
\end{aligned}$$

Substituting this in (C.11), we get

$$\begin{aligned}
\boldsymbol{\theta}_{GD} &:= \boldsymbol{\theta}^{(\infty)} \\
&= \boldsymbol{\theta}_{T_0} + \mathbf{X}_{T_2}^{\top}(\mathbf{X}_{T_2}\mathbf{X}_{T_2}^{\top})^{-1}(\mathbf{X}_{T_2}(\boldsymbol{\theta}_T^* - \boldsymbol{\theta}_{T_0}) + \mathbf{z}_{T_2}) \\
&= \mathbf{P}_{\parallel}\boldsymbol{\theta}_T^* + \mathbf{P}_{\perp}\boldsymbol{\theta}_{T_0} + \mathbf{X}_{T_2}^{\top}(\mathbf{X}_{T_2}\mathbf{X}_{T_2}^{\top})^{-1}\mathbf{z}_{T_2}
\end{aligned}$$

REFERENCES

[ADH$^+$19] Sanjeev Arora, Simon Du, Wei Hu, Zhiyuan Li, and Ruosong Wang. Fine-grained analysis of optimization and generalization for overparameterized two-layer neural networks. In *International Conference on Machine Learning*, pages 322–332. PMLR, 2019.

[ADX10] A. Agarwal, O. Dekel, and L. Xiao. Optimal algorithms for online convex optimization with multi-point bandit feedback. In *Proceedings of the 23rd Annual Conference on Learning Theory*, pages 28–40, 2010.

[AGL17a] M. Akbari, B. Gharesifard, and T. Linder. Distributed online convex optimization on time-varying directed graphs. *IEEE Transactions on Control of Network Systems*, 4(3):417–428, Sept. 2017.

[AGL$^+$17b] 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*, pages 1709–1720, 2017.

[AGL$^+$17c] Dan Alistarh, Demjan Grubic, Jerry Li, Ryota Tomioka, and Milan Vojnovic. QSGD: communication-efficient SGD via gradient quantization and encoding. In *Advances in Neural Information Processing Systems, NIPS*, pages 1709–1720, 2017.

[AHJ$^+$18a] D. Alistarh, T. Hoefler, M. Johansson, S. Khirirat, N. Konstantinov, and C. Renggli. The convergence of sparsified gradient methods. In *Advances in Neural Information Processing Systems*, volume 31, 2018.

[AHJ$^+$18b] Dan Alistarh, Torsten Hoefler, Mikael Johansson, Nikola Konstantinov, Sarit Khirirat, and Cédric Renggli. The convergence of sparsified gradient methods. In *Advances in Neural Information Processing Systems, NeurIPS*, pages 5973–5983, 2018.

[AHU58] K. J. Arrow, L. Hurwicz, and H. Uzawa. *Studies in Linear and Nonlinear Programming.* Cambridge University Press, New York, NY, USA, 1958.

[ALBR19] Mahmoud Assran, Nicolas Loizou, Nicolas Ballas, and Michael Rabbat. Stochastic gradient push for distributed deep learning. In *International Conference on Machine Learning, ICML*, pages 344–353, 2019.

[ARP$^+$21] Sk Miraj Ahmed, Dripta S Raychaudhuri, Sujoy Paul, Samet Oymak, and Amit K Roy-Chowdhury. Unsupervised multi-source domain adaptation without access to source data. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 10103–10112, 2021.

[BCK$^+$07] John Blitzer, Koby Crammer, Alex Kulesza, Fernando Pereira, and Jennifer Wortman. Learning bounds for domain adaptation. *Advances in neural information processing systems*, 20, 2007.

[BCV13] Yoshua Bengio, Aaron Courville, and Pascal Vincent. Representation learning: A review and new perspectives. *IEEE transactions on pattern analysis and machine intelligence*, 35(8):1798–1828, 2013.

[BDBCP07] Shai Ben-David, John Blitzer, Koby Crammer, and Fernando Pereira. Analysis of representations for domain adaptation. *Advances in neural information processing systems*, 20, 2007.

[BDKD19a] D. Basu, D. Data, C. Karakus, and S. N. Diggavi. Qsparse-local-sgd: Distributed sgd with quantization, sparsification, and local computations. *Advances in Neural Information Processing Systems*, 32, 2019.

[BDKD19b] Debraj Basu, Deepesh Data, Can Karakus, and Suhas N. Diggavi. Qsparse-local-sgd: Distributed SGD with quantization, sparsification and local computations. In *Advances in Neural Information Processing Systems, NeurIPS*, pages 14668–14679, 2019.

[Ben12] Yoshua Bengio. Deep learning of representations for unsupervised and transfer learning. In *Proceedings of ICML workshop on unsupervised and transfer learning*, pages 17–36. JMLR Workshop and Conference Proceedings, 2012.

[BEO16] T. Başar, S. R. Etesami, and A. Olshevsky. Convergence time of quantized metropolis consensus over time-varying networks. *IEEE Transactions on Automatic Control*, 61(12):4048–4054, Dec. 2016.

[BHM18] Mikhail Belkin, Daniel J Hsu, and Partha Mitra. Overfitting or perfect fitting? risk bounds for classification and regression rules that interpolate. *Advances in neural information processing systems*, 31, 2018.

[BHX20] Mikhail Belkin, Daniel Hsu, and Ji Xu. Two models of double descent for weak features. *SIAM Journal on Mathematics of Data Science*, 2(4):1167–1180, 2020.

[BKR19] A. S. Bedi, A. Koppel, and K. Rajawat. Asynchronous saddle point algorithm for stochastic optimization in heterogeneous networks. *IEEE Transactions on Signal Processing*, 67(7):1742–1757, April 2019.

[BLLT20] Peter L Bartlett, Philip M Long, Gábor Lugosi, and Alexander Tsigler. Benign overfitting in linear regression. *Proceedings of the National Academy of Sciences*, 117(48):30063–30070, 2020.

[BPP12] S. Bhatnagar, H. L. Prasad, and L. A. Prashanth. *Stochastic recursive algorithms for optimization: Simultaneous perturbation methods*, volume 434 of *Springer Series in Operations Research and Financial Engineering*. Springer, 2012.

[BV04] Stephen Boyd and Lieven Vandenberghe. *Convex optimization*. Cambridge university press, 2004.

[CB20] X. Cao and T. Başar. Decentralized multi-agent stochastic optimization with pairwise constraints and quantized communications. *IEEE Transactions on Signal Processing*, 68:3296–3311, 2020.

[CB21a] X. Cao and T. Başar. Decentralized online convex optimization based on signs of relative states. *Automatica*, 129:109676, July 2021.

[CB21b] X. Cao and T. Başar. Decentralized online convex optimization with event-triggered communications. *IEEE Transactions on Signal Processing*, 69:284–299, 2021.

[CB23] X. Cao and T. Başar. Decentralized online convex optimization with compressed communications. *Automatica*, 156:111186, 2023.

[CGSY18] Tianyi Chen, Georgios Giannakis, Tao Sun, and Wotao Yin. Lag: Lazily aggregated gradient for communication-efficient distributed learning. In *Advances in Neural Information Processing Systems, NeurIPS*, pages 5050–5060, 2018.

[CHMS21] Liam Collins, Hamed Hassani, Aryan Mokhtari, and Sanjay Shakkottai. Exploiting shared representations for personalized federated learning.

In *International Conference on Machine Learning*, pages 2089–2099. PMLR, 2021.

[CL19] X. Cao and K. J. R. Liu. Online convex optimization with time-varying constraints and bandit feedback. *IEEE Transactions on Automatic Control*, 64(7):2665–2680, July 2019.

[CLB16] M. El Chamie, J. Liu, and T. Başar. Design and analysis of distributed averaging with quantized communication. *IEEE Transactions on Automatic Control*, 61(12):3870–3884, Dec. 2016.

[CLB22] Niladri S Chatterji, Philip M Long, and Peter L Bartlett. The interplay between implicit bias and benign overfitting in two-layer linear networks. *Journal of machine learning research*, 23(263):1–48, 2022.

[CLG17] T. Chen, Q. Ling, and G. Giannakis. An online convex optimization approach to proactive network resource allocation. *IEEE Transactions on Signal Processing*, 65(24):6350–6364, 2017.

[CLL21] Kurtland Chua, Qi Lei, and Jason D Lee. How fine-tuning allows for effective meta-learning. *Advances in Neural Information Processing Systems*, 34:8871–8884, 2021.

[CNS14] T.-H. Chang, A. Nedić, and A. Scaglione. Distributed constrained optimization by consensus-based primal-dual perturbation method. *IEEE Transactions on Automatic Control*, 59(6):1524–1538, June 2014.

[CR16] Weisheng Chen and Wei Ren. Event-triggered zero-gradient-sum distributed consensus optimization over directed networks. *Automatica*, 65:90–97, 2016.

[CRS14] J. Chen, C. Richard, and A. H. Sayed. Multitask diffusion adaptation over networks. *IEEE Transactions on Signal Processing*, 62(16):4129–4144, Aug. 2014.

[DDS$^+$09] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. Imagenet: A large-scale hierarchical image database. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR*, pages 248–255, 2009.

[DFJ12] Dimos V. Dimarogonas, Emilio Frazzoli, and Karl Henrik Johansson. Distributed event-triggered control for multi-agent systems. *IEEE Transactions on Automatic Control*, 57(5):1291–1297, 2012.

[DHK$^+$21] Simon Shaolei Du, Wei Hu, Sham M. Kakade, Jason D. Lee, and Qi Lei. Few-shot learning via learning the representation, provably. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021.* OpenReview.net, 2021.

[DJWW15] 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*, 61(5):2788–2806, May 2015.

[DYG$^+$18] Wen Du, Xinlei Yi, Jemin George, Karl Henrik Johansson, and Tao Yang. Distributed optimization with dynamic event-triggered mechanisms. In *IEEE Conference on Decision and Control, CDC*, pages 969–974, 2018.

[EB16] S. R. Etesami and T. Başar. Convergence time for unbiased quantized consensus over static and dynamic networks. *IEEE Transactions on Automatic Control*, 61(2):443–455, Feb. 2016.

[EZC$^+$19] M. Eisen, C. Zhang, L. F. Chamon, D. D. Lee, and A. Ribeiro. Learning optimal resource allocations in wireless systems. *IEEE Transactions on Signal Processing*, 67(10):2775–2790, May 2019.

[FAL17] Chelsea Finn, Pieter Abbeel, and Sergey Levine. Model-agnostic meta-learning for fast adaptation of deep networks. In *International conference on machine learning*, pages 1126–1135. PMLR, 2017.

[FKM05] 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 16th Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 385–394, 2005.

[Gir15] Antoine Girard. Dynamic triggering mechanisms for event-triggered control. *IEEE Transactions on Automatic Control*, 60(7):1992–1997, 2015.

[GLSS18] Suriya Gunasekar, Jason Lee, Daniel Soudry, and Nathan Srebro. Characterizing implicit bias in terms of optimization geometry. In *International Conference on Machine Learning*, pages 1832–1841. PMLR, 2018.

[GWH16] Tomer Galanti, Lior Wolf, and Tamir Hazan. A theoretical framework for deep transfer learning. *Information and Inference: A Journal of the IMA*, 5(2):159–209, 2016.

[HHG19] D. Hajinezhad, M. Hong, and A. Garcia. Zone: Zeroth order nonconvex multi-agent optimization over networks. *IEEE Transactions on Automatic Control*, 64(10):3995–4010, Oct. 2019.

[HJT12] W. P. M. H. Heemels, Karl Henrik Johansson, and Paulo Tabuada. An introduction to event-triggered and self-triggered control. In *IEEE Conference on Decision and Control, CDC*, pages 3270–3285, 2012.

[HMRT22] Trevor Hastie, Andrea Montanari, Saharon Rosset, and Ryan J Tibshirani. Surprises in high-dimensional ridgeless least squares interpolation. *The Annals of Statistics*, 50(2):949–986, 2022.

[HZRS16] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *IEEE Conference on Computer Vision and Pattern Recognition, CVPR*, pages 770–778, 2016.

[JZW$^+$18] Ye Jia, Yu Zhang, Ron Weiss, Quan Wang, Jonathan Shen, Fei Ren, Patrick Nguyen, Ruoming Pang, Ignacio Lopez Moreno, Yonghui Wu, et al. Transfer learning from speaker verification to multispeaker text-to-speech synthesis. *Advances in neural information processing systems*, 31, 2018.

[KBS07] A. Kashyap, T. Başar, and R. Srikant. Quantized consensus. *Automatica*, 43(7):1192–1203, July 2007.

[KCM15] Solmaz S. Kia, Jorge Cortés, and Sonia Martínez. Distributed convex optimization via continuous-time coordination algorithms with discrete-time communication. *Automatica*, 55:254–264, 2015.

[KL17] Vladimir Koltchinskii and Karim Lounici. Concentration inequalities and moment bounds for sample covariance operators. *Bernoulli*, 23(1):110–133, 2017.

[KLSJ20] Anastasia Koloskova, Tao Lin, Sebastian U. Stich, and Martin Jaggi. Decentralized Deep Learning with Arbitrary Communication Compression. In *International Conference on Learning Representations, ICLR*, 2020.

[KNH09] Alex Krizhevsky, Vinod Nair, and Geoffrey Hinton. Cifar-10. *Canadian Institute for Advanced Research*, 2009.

[Kol]

[KRSJ19a] S. P. Karimireddy, Q. Rebjock, S. Stich, and M. Jaggi. Error feedback fixes signsgd and other gradient compression schemes. In *International Conference on Machine Learning*, pages 3252–3261, 2019.

[KRSJ19b] Sai Praneeth Karimireddy, Quentin Rebjock, Sebastian U. Stich, and Martin Jaggi. Error feedback fixes signsgd and other gradient compression schemes. In *International Conference on Machine Learning, ICML*, pages 3252–3261, 2019.

[KSJ19a] A. Koloskova, S. U. Stich, and M. Jaggi. Decentralized stochastic optimization and gossip algorithms with compressed communication. In *International Conference on Machine Learning*, pages 3478–3487, 2019.

[KSJ19b] Anastasia Koloskova, Sebastian U. Stich, and Martin Jaggi. Decentralized Stochastic Optimization and Gossip Algorithms with Compressed Communication. In *International Conference on Machine Learning, ICML*, pages 3478–3487, 2019.

[KSR17] A. Koppel, B. M. Sadler, and A. Ribeiro. Proximity without consensus in online multiagent optimization. *IEEE Transactions on Signal Processing*, 65(12):3062–3077, June 2017.

[LCS18] Q Sun Y Liu, TS Chua, and B Schiele. Meta-transfer learning for few-shot learning. In *2018 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018.

[LHF20] Jian Liang, Dapeng Hu, and Jiashi Feng. Do we really need to access the source data? source hypothesis transfer for unsupervised domain adaptation. In *International Conference on Machine Learning*, pages 6028–6039. PMLR, 2020.

[LKC$^+$18] S. Liu, B. Kailkhura, P.-Y. Chen, P. Ting, S. Chang, and L. Amini. Zeroth-order stochastic variance reduction for nonconvex optimization. In *Proceedings of the Conference on Neural Information Processing Systems*, pages 3727–3737, 2018.

[LLS20] Zhiyuan Liu, Yankai Lin, and Maosong Sun. *Representation learning for natural language processing*. Springer Nature, 2020.

[LLZ20] G. Lan, S. Lee, and Y. Zhou. Communication-efficient algorithms for decentralized and stochastic optimization. *Mathematical Programming*, 180:237–284, 2020.

[LNTL17] Yaohua Liu, Cameron Nowzari, Zhi Tian, and Qing Ling. Asynchronous periodic event-triggered coordination of multi-agent systems. In *IEEE Conference on Decision and Control, CDC*, pages 6696–6701, 2017.

[LZZ$^+$17] Xiangru Lian, Ce Zhang, Huan Zhang, Cho-Jui Hsieh, Wei Zhang, and Ji Liu. Can decentralized algorithms outperform centralized algorithms? a case study for decentralized parallel stochastic gradient descent. In *Advances in Neural Information Processing Systems, NIPS*, pages 5330–5340, 2017.

[LZZL17] Xiangru Lian, Wei Zhang, Ce Zhang, and Ji Liu. Asynchronous decentralized parallel stochastic gradient descent. In *International Conference on Machine Learning, ICML*, pages 3043–3052, 2017.

[MB17] Daniel McNamara and Maria-Florina Balcan. Risk bounds for transferring representations with and without fine-tuning. In *International conference on machine learning*, pages 2373–2381. PMLR, 2017.

[MJY12] M. Mahdavi, R. Jin, and T. Yang. Trading regret for efficiency: Online convex optimization with long term constraints. *Journal of Machine Learning Research*, 13:2503–2528, 2012.

[MMR08] Yishay Mansour, Mehryar Mohri, and Afshin Rostamizadeh. Domain adaptation with multiple sources. *Advances in neural information processing systems*, 21, 2008.

[MMR09] Yishay Mansour, Mehryar Mohri, and Afshin Rostamizadeh. Domain adaptation with multiple sources. In *Advances in neural information processing systems*, pages 1041–1048, 2009.

[MNC14] D. Mateos-Nunez and J. Cortés. Distributed online convex optimization over jointly connected digraphs. *IEEE Transactions on Network Science and Engineering*, 1(1):23–37, Jan.-June 2014.

[MPP$^+$17] H. Mania, X. Pan, D. Papailiopoulos, B. Recht, K. Ramchandran, and M. Jordan. Perturbed iterate analysis for asynchronous stochastic optimization. *SIAM Journal on Optimization*, 27(4):2202–2229, 2017.

[MPRP16] Andreas Maurer, Massimiliano Pontil, and Bernardino Romera-Paredes. The benefit of multitask representation learning. *Journal of Machine Learning Research*, 17(81):1–32, 2016.

[MSS] Samarth Mishra, Kate Saenko, and Venkatesh Saligrama. Surprisingly simple semi-supervised domain adaptation with pretraining and consistency. In *NeurIPS 2021 Workshop on Distribution Shifts: Connecting Methods and Applications*.

[NN13] I. Necoara and V. Nedelcu. Rate analysis of inexact dual first-order methods application to dual decomposition. *IEEE Transactions on Automatic Control*, 59(5):1232–1243, 2013.

[NND$^+$18] L. Nguyen, P. H. Nguyen, M. Dijk, P. Richtárik, K. Scheinberg, and M. Takáč. Sgd and hogwild! convergence without the bounded gradients assumption. In *International Conference on Machine Learning*, pages 3750–3758, 2018.

[NO09a] A. Nedic and A. Ozdaglar. Distributed subgradient methods for multiagent optimization. *IEEE Transactions on Automatic Control*, 54(1):48–61, Jan. 2009.

[NO09b] A. Nedić and A. Ozdaglar. Subgradient methods for saddle-point problems. *Journal of Optimization Theory and Applications*, 142(1):205–228, 2009.

[NOOT08] A. Nedic, A. Olshevsky, A. Ozdaglar, and J. N. Tsitsiklis. Distributed subgradient methods and quantization effects. In *Proceedings of the 47th IEEE Conference on Decision and Control*, pages 4177–4184, 2008.

[NRFS16] R. Nassif, C. Richard, A. Ferrari, and A. H. Sayed. Multitask diffusion adaptation over asynchronous networks. *IEEE Transactions on Signal Processing*, 64(11):2835–2850, June 2016.

[PBFM16] L. A. Prashanth, S. Bhatnagar, M. Fu, and S. Marcus. Adaptive system optimization using random directions stochastic approximation. *IEEE Transactions on Automatic Control*, 62(5):2223–2238, 2016.

[RMHP18] Amirhossein Reisizadeh, Aryan Mokhtari, Hamed Hassani, and Ramtin Pedarsani. Quantized decentralized consensus optimization. In *IEEE Conference on Decision and Control, CDC*, pages 5838–5843, 2018.

[RMHP19] A. Reisizadeh, A. Mokhtari, H. Hassani, and R. Pedarsani. An exact quantized decentralized gradient descent algorithm. *IEEE Transactions on Signal Processing*, 67(19):4934–4947, Oct. 2019.

[RN05] M. G. Rabbat and R. D. Nowak. Quantized incremental algorithms for distributed optimization. *IEEE Journal on Selected Areas in Communications*, 23(4):798–808, April 2005.

[RRWN11] Benjamin Recht, Christopher Re, Stephen Wright, and Feng Niu. Hogwild: A lock-free approach to parallelizing stochastic gradient descent. In *Advances in neural information processing systems*, pages 693–701, 2011.

[SBG21] Gal Shachaf, Alon Brutzkus, and Amir Globerson. A theoretical analysis of fine-tuning with linear teachers. *Advances in Neural Information Processing Systems*, 34:15382–15394, 2021.

[SBKS20] Vatsal Shah, Soumya Basu, Anastasios Kyrillidis, and Sujay Sanghavi. On generalization of adaptive methods for over-parameterized linear regression. *arXiv preprint arXiv:2011.14066*, 2020.

[SCDB] N. Singh, X. Cao, S. Diggavi, and T. Başar. Decentralized multi-task stochastic optimization with compressed communications. `https://arxiv.org/abs/2112.12373`.

[SCDB24] Navjot Singh, Xuanyu Cao, Suhas Diggavi, and Tamer Başar. Decentralized multi-task stochastic optimization with compressed communications. *Automatica*, 159:111363, 2024.

[SCJ18a] S. U. Stich, J.-B. Cordonnier, and M. Jaggi. Sparsified sgd with memory. In *Advances in Neural Information Processing Systems*, volume 31, pages 4452–4463, 2018.

[SCJ18b] Sebastian U. Stich, Jean-Baptiste Cordonnier, and Martin Jaggi. Sparsified SGD with Memory. In *Advances in Neural Information Processing Systems, NeurIPS*, pages 4447–4458, 2018.

[SD23] Navjot Singh and Suhas Diggavi. Representation transfer learning via multiple pre-trained models for linear regression. In *2023 IEEE International Symposium on Information Theory (ISIT)*, pages 561–566. IEEE, 2023.

[SDD23] Navjot Singh, Deepesh Data, and Suhas Diggavi. Communication efficient distributed learning. In *Artificial Intelligence for Edge Computing*, pages 199–222. Springer, 2023.

[SDGD19] Navjot Singh, Deepesh Data, Jemin George, and Suhas Diggavi. SPARQ-SGD: Event-triggered and compressed communication in decentralized stochastic optimization. *arXiv preprint arXiv:1910.14280*, 2019.

[SDGD20a] N. Singh, D. Data, J. George, and S. Diggavi. Sparq-sgd: Event-triggered and compressed communication in decentralized optimization. In *IEEE Conference on Decision and Control*, pages 3449–3456, 2020.

[SDGD20b] Navjot Singh, Deepesh Data, Jemin George, and Suhas Diggavi. Sparq-sgd: Event-triggered and compressed communication in decentralized optimization. In *2020 59th IEEE Conference on Decision and Control (CDC)*, pages 3449–3456. IEEE, 2020.

[SDGD20c] Navjot Singh, Deepesh Data, Jemin George, and Suhas Diggavi. Squarm-sgd: Communication-efficient momentum sgd for decentralized optimization. *arXiv preprint arXiv:2005.07041*, 2020.

[SDGD21a] N. Singh, D. Data, J. George, and S. Diggavi. Squarm-sgd: Communication-efficient momentum sgd for decentralized optimization. *IEEE Journal on Selected Areas in Information Theory*, 2(3):954–969, 2021.

[SDGD21b] Navjot Singh, Deepesh Data, Jemin George, and Suhas Diggavi. Squarm-sgd: Communication-efficient momentum sgd for decentralized optimization. In *2021 IEEE International Symposium on Information Theory (ISIT)*, pages 1212–1217. IEEE, 2021.

[SDGD21c] Navjot Singh, Deepesh Data, Jemin George, and Suhas Diggavi. Squarm-sgd: Communication-efficient momentum sgd for decentralized optimization. *IEEE Journal on Selected Areas in Information Theory*, 2(3):954–969, 2021.

[SDGD22] Navjot Singh, Deepesh Data, Jemin George, and Suhas Diggavi. Sparq-sgd: Event-triggered and compressed communication in decentralized optimization. *IEEE Transactions on Automatic Control*, 68(2):721–736, 2022.

[SDGD23] N. Singh, D. Data, J. George, and S. Diggavi. Sparq-sgd: Event-triggered and compressed communication in decentralized optimization. *IEEE Transactions on Automatic Control*, 68(2):721–736, 2023.

[SDJ13] Georg S. Seyboth, Dimos V. Dimarogonas, and Karl Henrik Johansson. Event-based broadcasting for multi-agent average consensus. *Automatica*, 49(1):245–252, 2013.

[Sha17] O. Shamir. An optimal algorithm for bandit and zero-order convex optimization with two-point feedback. *Journal of Machine Learning Research*, 18(52):1–11, 2017.

[Sin21] Ankit Singh. Clda: Contrastive learning for semi-supervised domain adaptation. *Advances in Neural Information Processing Systems*, 34:5089–5101, 2021.

[SLG$^+$21] Petar Stojanov, Zijian Li, Mingming Gong, Ruichu Cai, Jaime Carbonell, and Kun Zhang. Domain adaptation with invariant representation learning: What transformations to learn? *Advances in Neural Information Processing Systems*, 34:24791–24803, 2021.

[SLWY15] W. Shi, Q. Ling, G. Wu, and W. Yin. Extra: An exact first-order algorithm for decentralized consensus optimization. *SIAM Journal on Optimization*, 25(2):944–966, 2015.

[SLY$^+$14] W. Shi, Q. Ling, K. Yuan, G. Wu, and W. Yin. On the linear convergence of the admm in decentralized consensus optimization. *IEEE Transactions on Signal Processing*, 62(7):1750–1761, April 2014.

[Spa92] J. C. Spall. Multivariate stochastic approximation using a simultaneous perturbation gradient approximation. *IEEE Transactions on Automatic Control*, 37(3):332–341, 1992.

[Sti19] Sebastian U. Stich. Local SGD Converges Fast and Communicates Little. In *International Conference on Learning Representations, ICLR*, 2019.

[SVKB17] M. O. Sayin, N. D. Vanli, S. S. Kozat, and T. Başar. Stochastic subgradient algorithms for strongly convex optimization over distributed networks. *IEEE Transactions on Network Science and Engineering*, 4(4):248–260, Oct.-Dec. 2017.

[TGZ$^+$18] Hanlin Tang, Shaoduo Gan, Ce Zhang, Tong Zhang, and Ji Liu. Communication compression for decentralized training. In *Advances in Neural Information Processing Systems, NeurIPS*, pages 7663–7673, 2018.

[TJJ21] Nilesh Tripuraneni, Chi Jin, and Michael Jordan. Provable meta-learning of linear representations. In *International Conference on Machine Learning*, pages 10434–10443. PMLR, 2021.

[TT17] Tatiana Tatarenko and Behrouz Touri. Non-convex distributed optimization. *IEEE Transactions on Automatic Control*, 62(8):3744–3757, 2017.

[TYL$^+$19] Hanlin Tang, Chen Yu, Xiangru Lian, Tong Zhang, and Ji Liu. Doublesqueeze: Parallel stochastic gradient descent with double-pass error-compensated compression. In *International Conference on Machine Learning, ICML*, pages 6155–6165, 2019.

[WHHZ18] J. Wu, W. Huang, J. Huang, and T. Zhang. Error compensated quantized sgd and its applications to large-scale distributed optimization. In *International Conference on Machine Learning*, pages 5325–5333, 2018.

[WLL$^+$22] Jindong Wang, Cuiling Lan, Chang Liu, Yidong Ouyang, Tao Qin, Wang Lu, Yiqiang Chen, Wenjun Zeng, and Philip Yu. Generalizing to unseen domains: A survey on domain generalization. *IEEE Transactions on Knowledge and Data Engineering*, 2022.

[WTBR20] Jianyu Wang, Vinayak Tantia, Nicolas Ballas, and Michael Rabbat. Slowmo: Improving communication-efficient distributed sgd with slow momentum. In *International Conference on Learning Representations, ICLR*, 2020.

[WWW$^+$16] Wei Wen, Chunpeng Wu, Yandan Wang, Yiran Chen, and Hai Li. Learning structured sparsity in deep neural networks. In *Advances in Neural Information Processing Systems, NIPS*, pages 2074–2082, 2016.

[WZBG21] Jingfeng Wu, Difan Zou, Vladimir Braverman, and Quanquan Gu. Direction matters: On the implicit bias of stochastic gradient descent with moderate learning rate. In *9th International Conference on Learning Representations, ICLR 2021, Virtual Event, Austria, May 3-7, 2021*. OpenReview.net, 2021.

[YJY19] Hao Yu, Rong Jin, and Sen Yang. On the linear speedup analysis of communication efficient momentum SGD for distributed non-convex optimization. In *International Conference on Machine Learning, ICML*, pages 7184–7193, 2019.

[YLY16] K. Yuan, Q. Ling, and W. Yin. On the convergence of decentralized gradient descent. *SIAM Journal on Optimization*, 26(3):1835–1854, 2016.

[YN17] H. Yu and M. J. Neely. A simple parallel algorithm with an $o(\frac{1}{t})$ convergence rate for general convex programs. *SIAM Journal on Optimization*, 27(2):759–783, 2017.

[ZC16] S. Zhu and B. Chen. Quantized consensus by the admm: Probabilistic versus deterministic quantizers. *IEEE Transactions on Signal Processing*, 64(7):1700–1713, April 2016.

[ZDCZG19] Han Zhao, Remi Tachet Des Combes, Kun Zhang, and Geoffrey Gordon. On learning invariant representations for domain adaptation. In *International conference on machine learning*, pages 7523–7532. PMLR, 2019.

[ZHK19] Shuai Zheng, Ziyue Huang, and James Kwok. Communication-efficient distributed blockwise momentum sgd with error-feedback. In *Advances in Neural Information Processing Systems, NeurIPS*, pages 11446–11456, 2019.

[ZLLJ19] Yuchen Zhang, Tianle Liu, Mingsheng Long, and Michael I Jordan. Bridging theory and algorithm for domain adaptation. *arXiv preprint arXiv:1904.05801*, 2019.

[ZYB19] J. Zhang, K. You, and T. Başar. Distributed discrete-time optimization in multi-agent networks using only sign of relative state. *IEEE Transactions on Automatic Control*, 64(6):2352–2367, June 2019.

[ZZ22] Jing Zhao and Wei-Qiang Zhang. Improving automatic speech recognition performance for low-resource languages with self-supervised models. *IEEE Journal of Selected Topics in Signal Processing*, 16(6):1227–1241, 2022.