KTH ROYAL INSTITUTE
OF TECHNOLOGY

Doctoral Thesis in Electrical Engineering

First-Order Algorithms for Communication Efficient Distributed Learning

SARIT KHIRIRAT

Stockholm, Sweden 2022

First-Order Algorithms for Communication Efficient Distributed Learning

SARIT KHIRIRAT

Academic Dissertation which, with due permission of the KTH Royal Institute of Technology, is submitted for public defence for the Degree of Doctor of Philosophy on Wednesday 9th of March 2022 at 2:00 PM in U1, Kungliga Tekniska Högskolan, Brinellvägen 26, Stockholm.

Doctoral Thesis in Electrical Engineering
KTH Royal Institute of Technology
Stockholm, Sweden 2022

© Sarit Khirirat

ISBN 978-91-8040-134-0
TRITA-EECS-AVL-2022:9

Printed by: Universitetsservice US-AB, Sweden 2022

Abstract

Innovations in numerical optimization, statistics and high performance computing have enabled tremendous advances in machine learning algorithms, fuelling applications from natural language processing to autonomous driving. To deal with increasing data volumes, and to keep the training times of increasingly complex machine learning models reasonable, modern optimization algorithms distribute both data and computations over a large number of machines. However, these algorithms face challenges, from hardware and data heterogeneity (as different machines have different processors and data) to privacy and security issues (where data can be extracted from the transmitted parameters). Among these challenges, communication overhead constitutes a major performance bottleneck of the algorithms. Communicating millions of problem parameters between machines has been reported to consume up to 80% of the training time. To alleviate the communication bottleneck, we develop theory and strategies in this thesis to design communication-efficient optimization algorithms.

In the first part, we provide unified analysis frameworks for first-order algorithms using direct or error-compensated compression. We first identify key characteristics of the compression errors induced by many of the most popular compression strategies in the literature. We then perform a unified analysis of the convergence properties of first-order algorithms for general families of deterministic and stochastic gradient compression algorithms. Our results give explicit expressions for how compression accuracy and the amount of asynchrony affect the step-sizes and guaranteed convergence times. We next turn our attention to error-compensated compression algorithms. We develop theoretical explanations for why error-compensated compression algorithms attain solutions with arbitrarily higher accuracy than direct compression algorithms. Our results provide strong convergence guarantees of error-compensated compression algorithms for distributed and federated learning problems.

In the second part, we provide flexible tuning frameworks to optimize convergence performance of compression algorithms for a variety of system architectures. We start by analyzing data-dependent complexity that explains why direct compression algorithms are more communication-efficient than full-precision algorithms in practice. This complexity leads to automatic tuning strategies that enable popular compression algorithms on different communication networks to maximize both the convergence progress towards the solution and the communication efficiency. We then turn our attention to diminishing step-size schedules to maximize the convergence speed of the algorithms using noisy gradients. Our analysis framework is based on two classes of systems that characterize the impact of the step-sizes on the speed of noisy gradient algorithms. Our results show that such step-size schedules enable these algorithms to enjoy the optimal rate.

Applications of the algorithms in the thesis to central machine learning problems on benchmark data validate our theoretical results.

Abstract

Enorma framsteg inom maskininlärningsalgoritmer förbättrar centrala tillämpningar av artificiell intelligens, från naturlig språkbehandling till autonom körning, tack vare innovation inom numerisk optimering och högpresterande datorsystem. Dessa optimeringsbaserade algoritmer använder miljarder maskiner för att samordnat lösa storskaliga problem inom önskvärd träningstid. Emellertid utgör de utmaningar, från hårdvaru- och dataheterogenitet (eftersom olika enheter har olika datorkraft och data) till integritets- och säkerhetsproblematik (där data kan extraheras från de överförda parametrarna). Bland dessa utmaningar utgör kommunikationsoverhead en stor del av prestandaflaskhalsen för algoritmerna. Att kommunicera miljoner problemparametrar mellan maskiner har rapporterats förbruka upp till 80 % av träningstiden. För att lätta kommunikationsflaskhalsen utvecklar vi teori och strategier i denna avhandling för att designa kommunikationseffektiva optimeringsalgoritmer. I den första delen tillhandahåller vi enhetliga analysramverk för att analysera prestanda för första ordningens optimeringsalgoritmer med direkt eller felkompenserad komprimering, på en enda maskin och över flera maskiner. Vi skisserar först definitioner av kompressionstekniker som täcker in många kompressorer av praktiskt intresse. Sedan analyserar vi konvergens av första ordningens algoritmer som använder antingen deterministisk eller stokastisk kompression. Våra resultat påvisar den explicita effekten av kompressionsnoggrannhet och asynkrona fördröjningar på steglängd, konvergenshastighet och lösningsnoggrannhet för direktkomprimeringsalgoritmer. Vi vänder sedan vår uppmärksamhet till felkompenserade komprimeringsalgoritmer. Vi utvecklar teoretiska förklaringar till varför felkompenserade komprimeringsalgoritmer uppnår lösningar med godtyckligt högre noggrannhet än direkta komprimeringsalgoritmer. Våra resultat visar starka konvergensgarantier för felkompenserade komprimeringsalgoritmer för distribuerade och federerade inlärningsproblem. I den andra delen tillhandahåller vi flexibla inställningsramverk för att optimera konvergensprestanda för komprimeringsalgoritmer för en mängd olika systemarkitekturer. Vi börjar med att analysera databeroende komplexitet som förklarar varför direktkomprimeringsalgoritmer är mer kommunikationseffektiva än fullprecisionsalgoritmer i praktiken. Denna komplexitet leder till automatiska inställningsstrategier som möjliggör populära komprimeringsalgoritmer på olika kommunikationsnätverk för att maximera både framskridandet av konvergensen mot lösningen och kommunikationseffektiviteten. Vi riktar sedan vår uppmärksamhet mot steglängdsminskningsscheman för att maximera konvergenshastigheten för de algoritmer som använder stokastiska gradienter. Vår analysram baseras på två klasser av system som kännetecknar steglängdernas inverkan på hastigheten av stokastiska gradientalgoritmer. Våra resultat visar att sådana steglängdsscheman gör det möjligt för dessa algoritmer att åtnjuta den optimala

hastigheten. Simuleringar av algoritmer i denna avhandling på verkliga problem med referensdatamängder validerar våra teoretiska resultat.

Acknowledgements

I first would like to express my gratitude to my main supervior, Mikael Johansson, for mentoring me during my Master's thesis project which introduces me to active research areas in optimization methods for machine learning and control applications; and for facilitating me to go through this academic journey as a doctoral student supported by the Wallenberg Artificial Intelligence, Autonomous Systems and Software Program (WASP) funded by Knut and Alice Wallenberg Foundation. My most memorable advice from him is that research outputs with strong contributions require time and patience to learn and understand problems before solving them.

Thanks to Mikael, I am indebted to strong collaborators whom I have worked with. I first would like to thank Hamid, my previous supervisor of my Master's thesis, for the journey of introducing me into research areas in big-data optimization. I am grateful for collaborating with Dan Alistarh, Sindri Magnússon and Arda Aytekin, who introduce communication-efficient machine learning, one of the most active machine and federated learning research areas. I thank Xuyang Wu, Xiaoyu Wang, and Erik Berglund for current and ongoing research collaborations in zeroth-order and distributed optimization.

The division of Decision and Control Systems at KTH provides me with a dynamic research community. I enjoy myself spending time discussing, connecting and interacting with PhD students and postdocs from other groups. I would like to thank colleagues in our research group for intellectual friendship: Sindri, Arda, Vien, Martin, Malin, Hamed, Erik, Xiaoyu, Xuyang, Jacob, Dominik and Max; and also colleagues outside the group: Alexander, Mina, Rodrigo, Robert, Ines, and others.

Last but not least, I thank for my family's support from Thailand.

Sarit Khirirat,
Stockholm, February 2022.

Contents

Chapter 1

Introduction

In science and engineering we are often interested in finding the *best* decision among many possible alternatives. For instance, statisticians and control engineers aim to determine the best estimators which describe data sets, and the most accurate mathematical models based on signals of processes, respectively. The search for optimal solutions to engineering problems can often be formulated in the framework of mathematical programming. Here, optimization problems are typically written in the standard form

$$\begin{aligned} &\text{minimize} \quad F(x) \\ &\text{subject to} \quad x \in \mathcal{X}, \end{aligned} \tag{1.1}$$

where x is the decision variable, $\mathcal{X}$ is the constraint set restricting the admissible decisions, and $F(x)$ is the objective function measuring the cost of the decision. Once we have formulated the problem in mathematical terms, we can apply a wide variety of optimization algorithms, which start from an initial solution and iteratively improve the solution until an optimal solution is found

Over the last two decades, the popularity of optimization has increased dramatically. There are several reasons for this increase in popularity: on the one hand, the emergence of efficient general-purpose convex optimization algorithms and the abundance of computing power means that we can solve larger problems quicker and more reliably than ever; on the other hand, an increased awareness of optimization in general, and convex optimization in particular, has fueled the application of optimization techniques to many challenging real-world problems in wireless networks, signal processing, machine learning and control. We illustrate three successful real-world examples solved by optimization tools: resource allocation, binary classification, and adversarial attacks.

Example 1.1 (Resource Allocation)**.** The problem of allocating scarce resources to users arise in several application areas such as smart grids, wireless networks and finance.

Consider a network with n users and a single supplier in Figure 1.1. The objective is allocating resource budgets $q^i \in \mathbb{R}_+$ from the supplier to user $i = 1, 2, \ldots, n$ to maximize the sum of all local utility functions from the user $\sum_{i=1}^{n} U^i(q^i)$ subject to capacity constraints, i.e. the total resource capacity from the supplier $Q \in \mathbb{R}_+$. The resource allocation problem can be cast as the following optimization program:

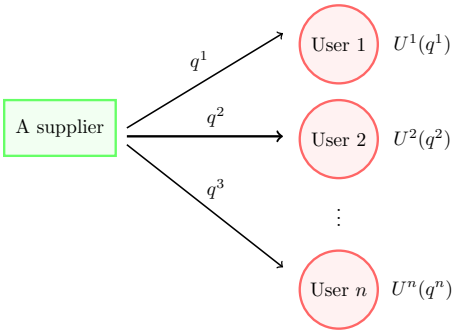


Figure 1.1: A network of n users and a single supplier.

$$\begin{aligned} \underset{q^1,q^2,\ldots,q^n}{\text{maximize}} \quad & \sum_{i=1}^{n} U^i(q^i) \\ \text{subject to} \quad & \sum_{i=1}^{n} q^i \leq Q. \end{aligned}$$

As the networks are growing to meet the increasing demand of multiple users (e.g. several IoT devices in the wireless networks), we need to solve larger optimization problems to coordinate the networks efficiently. Therefore, we need novel optimization algorithms which can handle the unprecedented scale of these problems.

Example 1.2 (Machine Learning)**.** Machine learning (ML) has become extremely successful in extracting useful information directly from data. ML tasks are usually formulated as optimization problems and therefore optimization plays a central role in ML. Among important ML problems is classification, which attempts to learn an optimal prediction for labelling new data samples. One prominent example is image recognition [1], where we try to classify objects from images (e.g. the hand-written digits from 0 to 9 in Figure 1.2).

We can formulate classification tasks into the form of optimization problem (1.1) as follows: Given the collection of data $\{(a^1, b^1), (a^2, b^2), \ldots, (a^n, b^n)\}$ with the feature a^i and its associated class label b^i

$$\underset{x\in\mathbb{R}^d}{\text{minimize}} \quad \frac{1}{n}\sum_{i=1}^{n} \ell(x; a^i, b^i),$$

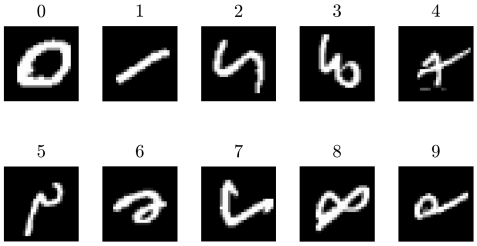


Figure 1.2: Sample digit images of the MNIST dataset from 0 to 9.

where x is the parameter of a classification rule, and $\ell(x; a^i, b^i)$ is called a loss function measuring the error of predicting the class label based on the input a^i and the rule x when the true label b^i is known.

Optimizing these ML models on the increasingly large volumes of data (e.g. collected from millions of images [2], of audio recordings [3], or of songs [4]) motivates the need of developing novel efficient optimization algorithms.

Example 1.3 (Adversarial Attacks)**.** Adversarial attacks are methods for crafting adversarial examples, which are slightly modified data samples, to test if trained neural network models misclassify these samples or not.

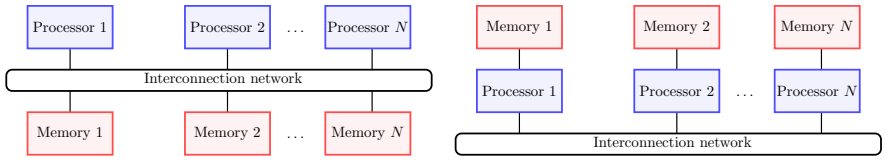
Given a data sample (e.g. an image) x_0, the adversarial attack tries to find an adversarial sample x, which the trained neural network model classifies into an incorrect target class label l. One possible adversarial attack task can be formulated into the following optimization problem:

$$\begin{aligned} \underset{x\in\mathbb{R}^d}{\text{minimize}} \quad & f(x,l) = \max\left\{\max_{i\neq l} \log[Z(x)]_i - \log[Z(x)]_l, -\omega\right\} \\ \text{subject to} \quad & \|x - x_0\| \leq \varepsilon. \end{aligned}$$

Here, $[Z(x)]_i$ is the predicted probability that x belongs to class i, $Z(x)$ is the output of all layers except the softmax layer from the trained network, $\omega > 0$ is a parameter which measures how likely the trained network will classify x with the incorrect label l, and $\varepsilon > 0$ is a parameter which measures how much x differs from x_0

As the trained neural network models and images become huge in dimension, novel optimization algorithms are required to produce adversarial examples efficiently.

These aforementioned examples highlight the importance of addressing large-scale optimization problems that traditional algorithms cannot handle. To solve such huge problems within reasonable training times, we must develop scalable parallel and distributed optimization algorithms.



(a) Shared memory (b) Distributed memory

Figure 1.3: The simplified view of shared and distributed memory systems.

1.1 High Performance Computing Architectures

Traditional computer programs are written in a *serial* fashion to perform one execution at a time. However, a single computer using the serial program cannot efficiently solve the huge-dimensional optimization problem on massive data. This necessitates the use of *parallel programming*. The key idea is first to split the whole optimization problem into smaller sub-problems. These sub-problems can be then solved concurrently by multiple processors in a single computer, or by multiple connected computers. Finally, parallel programs are written to allow multiple processors (or computers) to perform concurrent computations to find optimal solutions faster and/or to deal with larger optimization problems. For training deep neural network models such as ResNet, AlexNet and LSTM, parallel programs can be written to exploit multiple GPUs to significantly reduce processing time even on several data samples [5, 6]. Such parallel programs can be implemented by using many software frameworks such as `MPI` [7], `MapReduce` [8], `Aparche Spark` [9] and `POLO` [10, 11]. In addition, parallel computing architectures can be categorized based on system memory or coordination.

Shared-Memory and Distributed-Memory Systems

Parallel computing architectures have different memory structures to utilize multiple processors efficiently. In essence, they can be categorized mainly into shared memory and distributed memory.

The shared-memory system contains multiple processors, which all share access to every memory space via an interconnected network called a memory bus (see Figure 1.3a). Thanks to this feature, some languages provide parallel programming standards (e.g. `OpenMP` [12]) that conveniently split computational tasks among the processors in the shared-memory system. However, the memory bus in this system requires huge bandwidth to guarantee that the increasing number of processors all can directly access the shared memory space. This limitation of the system connections restricts the number of connected processors and peak performance [13].

In the distributed-memory system, each processor has immediate access to its local memory, whereas some global information can be coordinated among the processors using the interconnection network (see Figure 1.3b). Coordinations among the processors involve a message passing model that requires explicit use of send or receive primitives. These coordination primitives need not only the use of third-party libraries, such as `OpenMPI` [14] and `ZMQ` [15], but also substantial changes in the serial programs to generate the efficient parallel programs.

Centralized and Decentralized Coordination

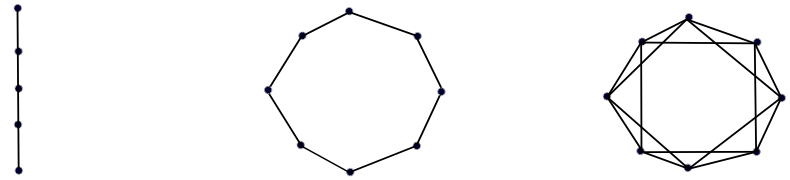
Both shared-memory and distributed-memory systems need coordination between the processors to collaboratively solve large-scale problems. The coordination can be either centralized or decentralized.

In the centralized coordination, every computing node (e.g. processor or computer) participates in collective communication, e.g. broadcasting the same information to or aggregating information from all the nodes. One common centralized coordination architecture is *master-server*. In the master-server architecture, each worker node executes its tasks independently of others, and transmits its output (e.g. its local solution or gradient) only to the master node (see Figure 1.6). Among implementations that use this architecture are `Project Adam` [16], `Tensorflow` [17] and `POLO` [10, 11].

Unlike the centralized coordination, the decentralized coordination does not need any central coordinator. In particular, each node communicates only with its neighboring nodes under the given network topology (see topology examples in Figure 1.4). This topology can be either *undirected* where each node exchanges data to the neighboring nodes in both directions, or *directed* where each node transmits data to the neighboring nodes in only one direction; see Figure 1.5. The topology captures many real-life communication models, and it thus plays a central role in the costs of information exchanged between multiple nodes. Implementations for the decentralized coordination rely on `AllReduce` operations [18] using popular frameworks and packages [19, 20, 7]. In `AllReduce`, all worker nodes perform reduction operations, which aggregate all the computations and generate the output. `AllReduce` can be implemented by third-party libraries such as `Horovod` [21] and `Tensorflow` [17].

1.2 New Challenges in Scalable Optimization

Modern optimization algorithms which utilize the parallel architectures are important to solve scalable problems. Parallelization reduces computation times at the cost of large communication overhead, which becomes a main performance bottleneck of algorithms. This bottleneck results from data communication between machines. In this process, data is represented by series



(a) (b) (c)

Figure 1.4: Examples of decentralized coordination networks: (a) line topology, (b) circular topology, and (c) 4-connected ring topology.

(a) Undirected network (b) Directed network

Figure 1.5: Examples of undirected and directed decentralized coordination networks of three nodes.

of bits or binary digits (0 or 1) for efficient transmission. For instance, the communication of each machine needs $32d$ bits to transmit the vector consisting of d single-precision floating points. To get a sense of communication costs, transmitting a single stochastic gradient using 32 bits per element requires 40 MB for a neural network model with 10 million parameters [22]. This implies that if we run the gradient-based algorithm on the 4G network, then we expect to transmit roughly one gradient per second, and typically need to transmit thousands of gradients per machine for the algorithm to obtain the sufficiently high-accuracy solution. These huge communication costs can easily overburden the algorithms running on architectures especially with resource-constrained devices such as the Arm Cortex-M microcontrollers with only 500KB available memory [23, 24].

Since communication can dominate overall training time [22], this communication bottleneck motivates a fundamental need to design communication-efficient optimization algorithms using the parallel architectures.

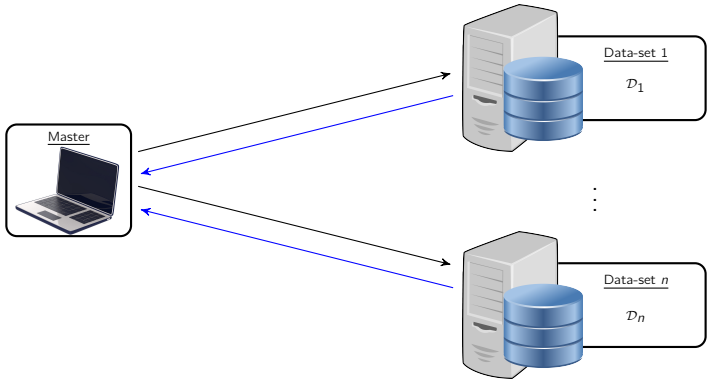


Figure 1.6: Master-server architecture.

1.3 Communication-Efficient Optimization

Communication constitutes a major performance bottleneck for optimization algorithms using parallel architectures to solve state-of-the-art machine learning applications. For instance, running stochastic gradient-based algorithms require 80% and 70% of training time for communication to train neural networks called AlexNet with 62 million parameters and LSTM with 13 million parameters, respectively [22]. These communication bottlenecks are mainly attributed to the size of transmitted information per machine and to communication frequency between the machines. This necessitates the development of communication-efficient optimization algorithms to alleviate this communication bottleneck, which can directly decrease overall time or energy for training the machine learning models. To reduce communication bandwidth, two commonly used mechanisms in these algorithms include infrequent communication protocols and low-precision information.

Infrequent Communication

Among popular mechanisms to decrease communication are infrequent communication protocols that reduce frequency of communicating information between the machines. One simplest protocol is *asynchronous communication*, where the algorithms update the solution using the currently available information sent from a single machine. Although this protocol decreases communication frequency, it introduces asynchrony into the algorithms. When training on the architectures with many stragglers (the slowest machines) which produce long communication delays [25, 26, 27], these asynchronous algorithms often suffer from slow convergence performance. To diminish the communication delays, one commonly used alternative is to allow the machines to perform

some local updates before transmitting their local information. The algorithms using this protocol are called *federated learning* algorithms. While federated learning algorithms reduce communication by increasing local computations, they face novel challenges, e.g., in hardware and data heterogeneity (as different machines access to different local processors and data) and in privacy and security (where attackers can extract valuable data from transmitted information). Thus, designing federated learning algorithms that guarantee high communication efficiency and practical performance while alleviating these technical issues is a challenging task.

Low-Precision Information

Another mechanism to reduce communication is to apply *compression* on information before transmission. Two successful examples of such compression strategies include *sparsification* which transmits only a few elements of information, and *quantization* which reduces information to a low-precision representation. Although optimization algorithms using state-of-the-art compression operators save communication bandwidth, they often suffer from slow convergence performance [28, 29]. This leads to significant developments in novel compression mechanisms that enable these algorithms to attain both communication savings and convergence performance comparable to the algorithms using full-precision information [22, 30, 31].

1.4 Research Challenges

Motivated by the need to solve this communication bottleneck problem, the challenging research goal is to develop optimization algorithms which reduce the communication while maintaining competitive performance. In particular, the ultimate research question is

Can we design scalable optimization algorithms with significantly reduced inter-node communication, which are able to alleviate the communication bottleneck, and converge to high-accuracy solutions faster and more efficiently than full-precision algorithms?

To answer this question, we focus on compression operators, novel mechanisms and hyperparameter tuning frameworks for communication-efficient optimization algorithms throughout the thesis. First, we provide a unified convergence of popular optimization algorithms with many compressors of interest. In particular, we characterize the explicit impact of compression level on theoretical convergence guarantees. Second, we develop novel compensation mechanisms that enable general compression algorithms to gain fast convergence speed and high solution accuray. We also provide theoretical justifications on why

and when these mechanisms significantly improve the performance of compression algorithms, and also theoretical convergence guarantees for compression algorithms using these mechanisms. Third, we propose tuning frameworks to improve communication efficiency and convergence performance of general compression algorithms. In particular, we show tuning strategies that allow popular compression operators to adjust their compression level adaptively to maximize communication efficiency and convergence speed of algorithms, and strategies to fine-tune step-sizes (or learning rates) that enable the algorithms to attain fast convergence rate and high solution accuracy.

1.5 Thesis Organization and Contributions

We now summarize thesis outline and the contributions in each chapter.

Chapter 2: Preliminaries

In this chapter, we introduce notations and definitions used in this thesis. We review background of convex optimization; first-order methods, performance analysis and applications on scalable parallel architectures; and important families of compressors.

Chapter 3: Compressed Gradient Methods

Compression and asynchronous computation have emerged as two key techniques for achieving scalability in optimization algorithms. However, only a few theoretical results quantify how these techniques impact algorithmic convergence performance. This chapter presents a unified convergence analysis for several first-order algorithms with compression and asynchronous computation. Our results characterize the explicit impact of asychrony delays and compression accuracy on convergence guarantee in terms of iteration and communication complexity. Numerical results confirm that fast convergence of these algorithms using limited information exchange is indeed possible.

This chapter is a summary of the following works:

- Sarit Khirirat, Mikael Johansson, and Dan Alistarh. "Gradient compression for communication-limited convex optimization." *2018 IEEE Conference on Decision and Control (CDC).* IEEE, 2018.

- Sarit Khirirat, Hamid Reza Feyzmahdavian, and Mikael Johansson. "Distributed learning with compressed gradients." *arXiv preprint arXiv:1806.06573* (2018).

Chapter 4: Error-compensated Gradient Methods

Compression algorithms have been proposed to reduce the communication load at the price of a loss in solution accuracy. Recently, it has been shown how compression errors can be compensated for in the optimization algorithm to improve the solution accuracy. Even though error-compensated compression algorithms have displayed superior performance in practice, there is very limited theoretical support for quantifying the observed improvements in solution accuracy. In this chapter, we show that error compensation, unlike naive compression, avoids accumulation of compression errors on quadratic problems. We also present strong convergence guarantees of error compensation schemes for stochastic gradient descent. Our numerical experiments highlight the benefits of error compensation algorithms.

This chapter is a summary of the following works:

- 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*, pp. 5973-5983. 2018.

- Sarit Khirirat, Sindri Magnússon, and Mikael Johansson. "Convergence Bounds for Compressed Gradient Methods with Memory Based Error Compensation." *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2019.
  Best Student Paper Award sponsored by Hitachi.

- Sarit Khirirat, Sindri Magnússon and Mikael Johansson, "Compressed Gradient Methods With Hessian-Aided Error Compensation," in *IEEE Transactions on Signal Processing*, vol. 69, pp. 998-1011, 2021.

Chapter 5: Federated Learning with Error-compensated Compression

State-of-the-art methods in federated learning applications reduce communication frequency, but are not guaranteed to converge toward the optimal solutions. These methods also experience a communication bottleneck, especially when the devices are power-constrained and communicate over a shared medium. This chapter presents `ECO-FedSplit`, an algorithm that increases the communication efficiency of federated learning without sacrificing solution accuracy. The key is to compress inter-device communication and to compensate for information losses in a theoretically justified manner. We prove strong convergence properties of `ECO-FedSplit` on strongly convex optimization problems and show that the algorithm yields a highly accurate solution

with dramatically reduced communication. Extensive numerical experiments validate our theoretical result on real data sets.

This chapter is based on the following work:

- Sarit Khirirat, Sindri Magnússon, and Mikael Johansson. "`Eco-Fedsplit`: Federated Learning with Error-Compensated Compression." Manuscript submitted for publication.

Chapter 6: Adaptive Compression Framework

Early works on compression algorithms for solving learning tasks focused on the bottleneck between CPUs and GPUs, but communication-efficiency is now needed in different system architectures, from high-performance clusters to energy-constrained IoT devices. In the current practice, compression levels are typically chosen before training and settings that work well for one task may be vastly sub-optimal for another dataset on another architecture. This chapter proposes a flexible framework which adapts the compression level to the true information at each iteration, maximizing the improvement in the objective function that is achieved per communicated bit. Our framework is easy to adapt from one technology to the next by modelling how the communication cost depends on the compression level for the specific technology. Theoretical results and practical experiments indicate our automatic tuning significantly increases communication efficiency on several state-of-the-art compressors.

This chapter is based on the following work:

- Sarit Khirirat, Sindri Magnússon, Arda Aytekin, and Mikael Johansson. "A Flexible Framework for Communication-Efficient Machine Learning." In *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 35, no. 9, pp. 8101-8109. 2021.

Chapter 7: Improved Step-Size Schedules for Noisy Gradient Methods

Noisy gradient algorithms have emerged as one of the most popular algorithms for distributed optimization with massive data. Choosing proper step-size schedules is an important task to tune in the algorithms for good performance. For the algorithms to attain fast convergence and high accuracy, it is intuitive to use large step-sizes in the initial iterations when the gradient noise is typically small compared to the algorithm-steps, and reduce the step-sizes as the algorithm progresses. This intuition has been confirmed in theory and practice for stochastic gradient descent. However, similar results are lacking for other methods using approximate gradients. This chapter shows that the diminishing step-size strategies can indeed be applied for a broad class of noisy gradient algorithms. Our analysis framework is based on two classes of systems that

characterize the impact of the step-sizes on the convergence performance of many algorithms. Our results show that such step-size schedules enable these algorithms to enjoy the optimal rate. We exemplify our results on stochastic compression algorithms. Our experiments validate fast convergence of these algorithms with the step decay schedules.

The chapter is summarized from the following works:

- Sarit Khirirat, Xiaoyu Wang, Sindri Magnússon, and Mikael Johansson. "Improved Step-Size Schedules for Noisy Gradient Methods." *ICASSP 2021-2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2021.

- Sarit Khirirat, Xiaoyu Wang, Sindri Magnússon and Mikael Johansson, "Improved Step-Size Schedules for Proximal Noisy Gradient Methods." Manuscript submitted for publication.

Chapter 8: Conclusion and Future Work

We conclude thesis contributions and discuss future research directions.

Chapter 2
Preliminaries

In this chapter, we summarize relevant theoretical background. We first introduce notations that are used throughout the thesis. We next review definitions of convex optimization and proximal operators, and introduce well-known numerical optimization methods. For designing scalable optimization methods, main classes of computing architectures are then introduced. Finally, major definitions of compressors and examples are provided.

2.1 Notations

Throughout the thesis, we reserve $\mathbb{N}$, $\mathbb{N}_0$, $\mathbb{Z}$, $\mathbb{R}$, and $\mathbb{R}_+$ to be the set of natural numbers, the set of natural numbers including zero, the set of integers, the set of real numbers, and the set of positive real numbers, respectively. For $a, b \in \mathbb{Z}$, we let $[a, b]$ be the set $\{a, a+1, a+2, \ldots, b\}$ where $a \leq b$. We use $\mathbb{R}^d$ to denote the d-dimensional vector space of real-valued vectors $x = [x^1, x^2, \ldots, x^d]^T$ which have their coordinates from a set of real numbers. For any vector $x \in \mathbb{R}^d$, $\|x\|$, $\|x\|_1$ and $\|x\|_0$ are the ℓ_2 norm, the ℓ_1 norm and the ℓ_0 norm, respectively; $\lceil x \rceil_+ = \max\{0, x\}$; $\text{sign}(x)$ is its sign vector; and $\text{supp}(x) = \{i : x^i \neq 0\}$ is the support set. For real-valued functions $g(k)$ and $h(k)$, $g(k) = \mathcal{O}(h(k))$ implies that $g(k)$ is of the same order as $h(k)$, i.e. $g(k) \leq Mh(k)$ for some $M \in \mathbb{R}$ and for all $k \in \mathbb{N}$, while $g(k) = o(h(k))$ means that $g(k)$ is asymptotically smaller than $h(k)$, i.e.

$$\lim_{k \to \infty} \frac{g(k)}{h(k)} = 0.$$

We use $\mathbb{R}^{d_1 \times d_2}$ to denote the set of real-valued matrices with d_1 rows and d_2 columns. For a symmetric matrix $A \in \mathbb{R}^{d \times d}$, $\lambda_1(A), \ldots, \lambda_d(A)$ denote the eigenvalues of A in an increasing order (including multiplicities), and its spectral norm is defined by $\|A\| = \max_i |\lambda_i(A)|$. For the fixed-point operator $\mathcal{T} : \mathbb{R}^d \to \mathbb{R}^d$ and a positive integer p, we denote $\mathcal{T}^p(x) = \mathcal{T} \circ \mathcal{T} \circ \ldots \circ \mathcal{T} x$ (p times).

2.2 Convex Optimization

In this section, we provide basic definitions of convex optimization. We begin by stating definitions of a *convex set* and a *convex function*.

Definition 2.1 (Convex set)**.** A set $\mathcal{X} \subset \mathbb{R}^d$ is called *convex* if for any $x, y \in \mathcal{X}$, we have

$$\theta x + (1-\theta)y \in \mathcal{X}, \quad \text{for} \quad \theta \in [0,1].$$

This definition implies that if the set is convex, then there always exists a straight line between any two points in the set which also lies within the set.

Definition 2.2 (Convex function)**.** A function $F : \mathbb{R}^d \to \mathbb{R}$ is called *convex* if for any $x, y \in \mathbb{R}^d$, we have

$$F(\theta x + (1-\theta)y) \leq \theta F(x) + (1-\theta)F(y), \quad \text{for} \quad \theta \in [0,1].$$

In this definition, if the function $F(x)$ where $x \in \mathbb{R}$ is convex, then the line segment between points $(x, F(x))$ and $(y, F(y))$ always lies above (or on) the actual curve of $F(x)$. For a differentiable and convex function $F(x)$, the following inequality also holds

$$F(y) \geq F(x) + \langle \nabla F(x), y - x \rangle, \quad \text{for } x, y \in \mathbb{R}^d.$$

In the thesis, we are interested in solving convex optimization problems on the form:

$$\begin{aligned} \text{minimize} \quad & F(x) \\ \text{subject to} \quad & x \in \mathcal{X}, \end{aligned} \tag{2.1}$$

where $F(x)$ is a convex function, and a constraint set $\mathcal{X} \subseteq \mathbb{R}^d$ is convex.

The solution x from the optimization problem can be either *globally optimal*, i.e. $F(x) \leq F(z)$ for all $x, z \in \mathbb{R}^d$ or *locally optimal* where there exists a positive constant R such that $F(x) \leq F(z)$ for $\|z - x\| \leq R$. One nice property of convex optimization is that any locally optimal solution is also globally optimal [32]. In addition, these problem classes can be solved by classical optimization algorithms to obtan a solution with high accuracy. Apart from convexity of problems, we require additional properties of the objective function $F(x)$ listed below:

Definition 2.3. A function $F : \mathbb{R}^d \to \mathbb{R}$ is smooth with $L > 0$ if for all $x, y \in \mathbb{R}^d$

$$\|\nabla F(x) - \nabla F(y)\| \leq L\|x - y\|. \tag{2.2}$$

This L-smoothness condition is equivalent to [33, Theorem 2.1.5]

$$F(y) \leq F(x) + \langle \nabla F(x), y - x \rangle + \frac{L}{2}\|y - x\|^2, \quad \forall x, y \in \mathbb{R}^d. \tag{2.3}$$

Definition 2.4. A function $F : \mathbb{R}^d \to \mathbb{R}$ is strongly convex with $\mu > 0$ if

$$F(y) \geq F(x) + \langle \nabla F(x), y - x \rangle + \frac{\mu}{2}\|y - x\|^2, \forall x, y \in \mathbb{R}^d. \tag{2.4}$$

Even though these conditions seem to restrict problem classes, many real-world engineering applications surprisingly fulfill these properties. Based on the definitions of smoothness and strong convexity assumptions, we can analyze convergence behaviors for optimization algorithms throughout this thesis.

2.3 Proximal Operator

In this section, we introduce a proximal operator which can be exploited to modify classical optimization algorithms for solving constrained, non-smooth, or distributed problems.

Definition 2.5 (Proximal operator)**.** A proximal operator of a closed proper convex function $f : \mathbb{R}^d \to \mathbb{R} \cup \{+\infty\}$ with a positive scalar α is defined by

$$\mathbf{prox}_{\alpha F}(x) = \underset{y \in \mathbb{R}^d}{\operatorname{argmin}} \left(F(y) + \frac{1}{2\alpha} \|y - x\|^2 \right). \tag{2.5}$$

Definition 2.5 implies $\mathbf{prox}_{\alpha F}(x)$ is a point that compromises between minimizing F and being near to x. The proximal operator reduces to the Euclidean projection onto a closed nonempty convex set $\mathcal{C}$ denoted by $\mathbf{prox}_{\alpha F}(x) = \pi_{\mathcal{C}}(x) = \operatorname{argmin}_{y \in \mathcal{C}} \|y - x\|$ if $F(x)$ is an indicator function on $\mathcal{C}$ (i.e. $F(x) = I_{\mathcal{C}}(x)$ where $I_{\mathcal{C}}(x) = 0$ for $x \in \mathcal{C}$ and $+\infty$ otherwise), and to the soft-thresholding operator defined by $[\mathbf{prox}_{\alpha F}(x)]^i = \left[x^i - \alpha\right]_+ \operatorname{sign}(x^i)$ if $F(x) = \|x\|_1$. Next, we introduce two key definitions related to the definition of the proximal operator: a Moreau envelope and a reflected proximal operator.

Definition 2.6 (Moreau envelope)**.** The Moreau envelope (or Moreau-Yoshida regularization) of a closed proper convex function $F : \mathbb{R}^d \to \mathbb{R} \cup \{+\infty\}$ with a positive scalar α is defined on

$$F_\alpha(x) = \min_{y \in \mathbb{R}^d} \left(F(y) + \frac{1}{2\alpha} \|y - x\|^2 \right). \tag{2.6}$$

Definition 2.7. A reflected proximal operator of a closed proper convex function $F : \mathbb{R}^d \to \mathbb{R} \cup \{+\infty\}$ with a positive scalar α is defined by

$$\mathbf{refl}_{\alpha F}(x) = 2\mathbf{prox}_{\alpha F}(x) - x, \tag{2.7}$$

where $\mathbf{prox}_{\alpha F}(x)$ is the proximal operator.

These definitions above allow for constructing a class of popular algorithms based on gradient information or operator splitting schemes for solving constrained, non-smooth or distributed problems throughout the thesis.

2.4 Numerical Optimization Methods

In this section, we introduce popular numerical optimization methods in the optimization and machine learning community: first-order methods, operator splitting methods, and zeroth-order methods.

2.4.1 First-order Methods

First-order methods are most commonly used optimization methods that utilize the gradient of objective functions to update solutions. The simplest first-order method is *gradient descent* that updates the solution x_k as follows: Given the initial solution x_0 and the positive step-size γ

$$x_{k+1} = x_k - \gamma \nabla F(x_k). \tag{2.8}$$

Theoretical convergence for gradient descent has been well-established, e.g. in [33, 34]. If we consider Problem (2.1) with the convex, smooth objective function and $\mathcal{X} = \mathbb{R}^d$, then the sequence $\{x_k\}_{k\in\mathbb{N}}$ generated by gradient descent (3.2) with $\gamma = 1/L$ converges at a sub-linear rate:

$$F(x_k) - F(x^\star) \leq \frac{L}{2k}\|x_0 - x^\star\|^2,$$

where $k \in \mathbb{N}$ is the number of iteration counts, $x^\star$ is the optimal solution, and $F(x^\star)$ is the optimal value. Furthermore, for Problem (2.1) with the strongly-convex, smooth objective function, gradient descent (3.2) with $\gamma = 1/L$ enjoys a linear rate:

$$F(x_k) - F(x^\star) \leq \left(1 - \frac{\mu}{L}\right)^k (F(x_0) - F(x^\star)).$$

Although several algorithms such as (quasi-)Newton methods have stronger convergence performance than gradient descent, they increase the memory requirements and per-iteration costs, and therefore cannot scale well for larger problem instances. Unlike these algorithms, gradient descent exhibits low per-iteration costs and has competitive convergence guarantees in practice. However, as training data become massive in terms of dimension and/or the number of data points, computing a single gradient can cause computionally expensive load. Running classical gradient descent on the data-intensive applications is thus prohibitive. This leads to modifications of first-order methods which are suitable for these large-scale problems as we discuss next.

Incremental Gradient Methods

Among popular modification of first-order methods is an *incremental gradient* method (IGM). To understand its advantages over classical gradient descent,

we need to exploit nice structure of optimization problems. In essence, we turn our attention to the finite-sum optimization on the form:

$$\underset{x\in\mathbb{R}^d}{\text{minimize}} \quad F(x) = \frac{1}{n}\sum_{i=1}^{n} F^i(x), \tag{2.9}$$

where each $F^i(x)$ is a component function. For learning applications, each $F^i(x)$ can quantify the statistical error between the solution x and the optimal solution $x^\star$ with respect to the i^{th} paritition of data. When the number of components n is large, computing a single gradient of the whole objective function $F(x)$ is often prohibitive (since we need to compute gradients of all component functions). Fortunately, first-order methods are robust to using gradient approximations.

IGM relies on the gradient of a single component function $F^i(x)$ as the approximate gradient of the entire objective function $F(x)$. IGM updates the solution x_k according to:

$$x_{k+1} = x_k - \gamma\nabla F^{i_k}(x_k), \tag{2.10}$$

where i_k is the index of the component function selected at each iteration k. The selection protocol of the index i_k can be stochastic or deterministic. While *stochastic gradient descent* (SGD) is IGM with i_k uniformly selected with replacement from the set $\{1, 2, \ldots, n\}$, *random reshuffling* is IGM with i_k is sampled without replacement (i.e. by choosing a random permutation on $\{1, 2, \ldots, n\}$).

IGM is more advantageous than classical gradient descent (3.2) in many aspects. The most obvious advantage of using IGM rather than gradient descent is its lower per-iteration cost by a factor of $1/n$, [35, 1]. This leads to fast convergence speed toward the solution and small memory requirements for deploying IGM to solve huge-scale problems. Despite its benefits, IGM trades off fast convergence for high solution accuracy. Estimating the true gradient based on only a single component function enables small computation time but crude gradient estimates with high variance, [1, 36, 37, 38, 39]. Therefore, it is preferable to design algorithms which reduce the variance but still achieve competitive convergence performance.

One common variance reduction technique is *mini-batching*, where several component functions are processed per iteration. The mini-batching technique is useful for converting the serial IGM into parallel and distributed ones in order to provide better training accuracy and scalability [40]. Another technique is to form a gradient approximation which guarantees the convergence toward the solution with high solution accuracy. Details of this technique are provided in the next section.

Incremental Aggregate Gradient Methods

Another major modification of first-order methods is the *incremental aggregate gradient* (IAG) method. IAG computes the gradient of a single component function like IGM, while keeping a memory of the most recent gradients of all component functions. In essence, IAG estimates the gradient according to:

$$g_k^{\mathrm{IAG}} = \frac{1}{n}\sum_{i=1}^{n} \nabla F^i(x_{k-\tau_i^k}) \approx \frac{1}{n}\sum_{i=1}^{n} \nabla F^i(x_k).$$

Here, each $\tau_i^k \in \mathbb{N}_0$ quantifies the latest time since the gradient of component function i was computed and stored. To solve finite-sum problems (2.9), IAG updates the solution x_k via

$$x_{k+1} = x_k - \gamma g_k^{\mathrm{IAG}}. \tag{2.11}$$

This method has the same per-iteration cost to compute the gradient of a single component function $F^i(x)$ as IGM, but like other variance-reduction methods such as SVRG [41] and SAGA [42] it needs extra memory requirements to store the gradients of every component function. Unlike IGM, IAG attains global convergence toward the exact optimum for deterministic problems. However, it suffers from slow convergence rate when running on architecture with slow machines or *stragglers*, which result in a large value of τ_i^k.

To reduce the adverse impact of stragglers, it is recommended to add data redundancy among the machines so that the method needs to wait for only gradients from fast machines. The methods using redundancy techniques are called coded gradient methods which are described next.

Coded Gradient Methods

Although distributed methods have gained increasing popularity, their performance is strongly dictated by stragglers, machines that are slow to process and submit their information. To reduce the impact of stragglers, one alternative approach is adding data redundancy across the machines and updating the solution without waiting for the stragglers. For finite-sum problems (2.9), the goal is to design the matrix $D \in \mathbb{R}^{n\times m}$ for assigning m parts of redundant data across n machines which have access to their private functions $F^i(x)$. Here, $D_{ij} \neq 0$ if the i^{th} part of data is available on machine j, and $D_{ij} = 0$ otherwise. After assigning data redundancy across the machines according to D, all machines compute the coded gradient

$$G_j = \sum_{i=1}^{n} \frac{D_{ij}}{\bar{\alpha}_i} g_k^i,$$

where g_k^i is the stochastic gradient with respect to $F^i(x_k)$, and $\bar{\alpha}_i = \sum_{j=1}^m D_{ij}$. Then, the central server waits and aggregates only a few gradients from the machines according to:

$$g_k^{\text{coded}} = \frac{1}{n} \sum_{j=1}^{m} w_j G_j, \tag{2.12}$$

where $w \in \mathbb{R}^m$ represents a decoding coefficient vector where $w_j = 1$ if the server receives the gradient from machine j, and 0 otherwise. The server updates the solution x_k using the descent update based on the coded gradient, i.e.

$$x_{k+1} = x_k - \gamma g_k^{\text{coded}}. \tag{2.13}$$

These coded gradient methods have been shown empirically to outperform traditional distributed gradient methods (or uncoded gradient methods), and many coding schemes have been developed to improve training performance of coded gradient methods [43, 44, 45, 46, 47, 48].

2.4.2 Operator Splitting Methods

Operator splitting methods have been successfully used to decompose large-scale distributed optimization problems into sub-problems that can be solved fast. Without loss of generality, the abstract form of distributed problems is the minimization of the sum of two objective functions on the form:

$$\underset{x \in \mathbb{R}^d}{\text{minimize}} \quad F(x) + g(x), \tag{2.14}$$

where $F(x)$ and $g(x)$ are closed, proper convex functions. Popular examples of operator-splitting methods solving these problems include the forward-backward splitting (FBS) method [49], and the relaxed Peaceman-Rachford splitting (relaxed PRS) method [50].

The FBS method, also called the proximal gradient method, has been extensively studied in [51, 52, 53]. This method can be used to solve Problem (2.14) where $F(x)$ is differentiable and L-smooth, and $g(x)$ is non-smooth but convex. The FBS method updates the solution x_k according to

$$x_{k+1} = \mathbf{prox}_{\gamma g}(x_k - \gamma \nabla F(x_k)), \tag{2.15}$$

where γ is a fixed positive step-size. Note that when we let $g(x) = 0$, the FBS method is classical gradient descent in Section 2.4.1. Inspired by the FBS method, many proximal gradient-based methods have been further developed to handle large-scale non-smooth problems. Key examples of such methods are proximal stochastic gradient methods [54, 55, 56] and proximal IAG (PIAG) [26, 57, 58].

Another key operator-splitting method for solving Problem (2.14) is the relaxed PRS method, which updates the solution x_k via

$$x_{k+1} = (1 - \lambda_k)x_k + \lambda_k \mathbf{refl}_{\gamma F}(\mathbf{refl}_{\gamma g}(x_k)), \tag{2.16}$$

for positive tuning parameters λ, γ. The relaxed PRS method with $\lambda_k = 1/2$ and $\lambda_k = 1$ are, respectively, the Douglas-Rachford splitting (DRS) method [59] and Peaceman-Rachford splitting (PRS) method [60].

2.4.3 Zeroth-order Methods

Several learning and decision making applications can be cast as optimization problems, where we can get access only to objective function values. Popular application examples include sensor selection problems in smart grids or wireless network systems [61, 62, 63], optimal hyper-parameter tuning for learning models [64, 65], and black-box adversarial attacks on neural network models [66, 67, 68, 69]. This motivates the development of zeroth-order (or derivative-free) optimization, which studies numerical methods for minimizing a function without having access to its gradients.

One of the earliest approaches to zeroth-order optimization was direct search. In each direct search step, the objective function is queried at various points, usually close to ones previously visited. If a point provides an improvement over the function value at the current solution, it is stored in memory as the algorithms progress. Prominent direct search methods include the Nelder-Mead simplex method [70], and the mesh adaptive direct search (MADS) method [71]. Another popular type of zeroth-order methods uses function value differences to estimate the gradients [72, Section 3.4]. Two commonly used approaches for approximating gradients for problems of minimizing the objective function $F(x)$ are

$$g_\tau(x) = \frac{F(x + \tau u) - F(x)}{\tau} u, \quad \text{and} \quad \hat{g}_\tau(x) = \frac{F(x + \tau u) - F(x - \tau u)}{2\tau} u \tag{2.17}$$

where $u \in \mathbb{R}^d$ is generated from the Gaussian distribution with zero mean and unit variance, with a positive scalar τ. However, for d-dimensional problems, these methods are d times slower than classical gradient descent in Section 2.4.1, and also suffers from poor performance particularly for ill-conditioned high-dimensional problems.

2.5 Performance Analysis of Optimization Methods

Several methods have been introduced to tackle different problem families in Section 2.4. To study and compare convergence performance of these op-

timization methods for given problem classes, we often start by measuring computational efforts of each method (e.g. its running time) to obtain the optimal solution with target accuracy on solving problems in numerical simulations. Inspired by these empirical performance metrics, the literature in numerical optimization focuses on two popular complexities that characterize the convergence rate of given optimization methods in mathematical terms: *iteration complexity* and *communication complexity*.

The iteration complexity measures the number of iterations needed to run a single method to reach the optimal solution with ϵ-accuracy. The iteration complexities of first-order methods have been well-established in the literature. For instance, gradient descent in Section 2.4.1 needs

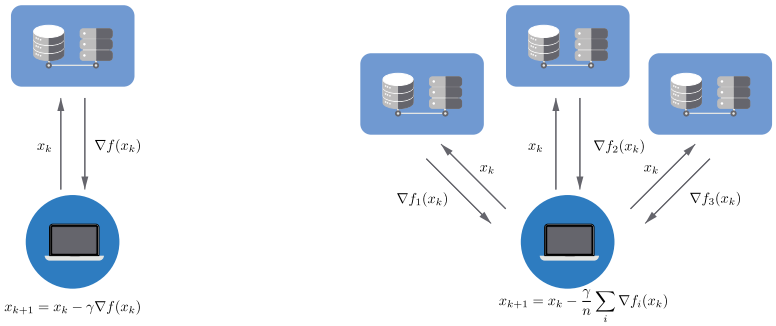
$$\frac{L\|x_0 - x^\star\|^2}{2}\frac{1}{\epsilon} \quad \text{and} \quad \frac{L}{\mu}\log\left(\frac{F(x_0) - F(x^\star)}{\epsilon}\right) \quad \text{iterations}$$

to reach $F(x_k) - F(x^\star) \leq \epsilon$ for convex and strongly convex optimization, respectively. These classical theoretical lower bounds of gradient descent have been derived in [73, 74].

The communication complexity measures the number of communicated bits required for the method to reach the ϵ-accuracy. Despite extensive studies of the iteration complexity, there are very few works which study the communication complexity and its lower bound. For instance, Tsitsiklis and Luo [75] derived lower bounds for the method using two machines to solve problems of minimizing the sum of two objective functions $F^1(x) + F^2(x)$. However, theoretical results for the methods utilizing multiple machines to solve general finite-sum problems are lacking. Characterizing communication complexities of optimization methods can imply their practical performance, especially when the methods are run on the network of several computing machines to solve sizable problems. This is because in these problem settings the methods have high communication overhead, even to the point where communication dominates the overall running time of algorithms [22].

2.6 Architectures for Scalable Optimization

The scale and complexity in learning and decision making applications have been dramatically increasing. This leads to the popularity of using parallel architectures to reduce training time. In the thesis, we study first-order methods due to their easy implementations under these parallel architectures. In essence, we categorize these methods based on the use of a) full gradient communication or b) partial gradient communication; see Figure 2.1. On the one hand, we solve the optimization problem under the full gradient communication by communicating the full gradient information in every iteration. Such communication usually appears in dual decomposition methods; see Subsection 2.6.1. On the other hand, the first-order methods under the partial



(a) (b)

Figure 2.1: Two common communication architectures for distributed gradient methods: 1) full gradient communication (left) and 2) partial gradient communication (right).

gradient communication rely on gradient evaluations based on different nodes, each with its local data. We now review first-order methods under these communication architectures separately in more details.

2.6.1 Full Gradient Communication: Dual Decomposition

Resource allocation is a class of distributed optimization problems where a group of n nodes aim to minimize the sum of their local utility function over a set of shared resource constraints. In particular, the nodes collaboratively solve

$$\begin{aligned} \underset{q_1,\ldots,q_n}{\text{minimize}} \quad & \sum_{i=1}^{n} U^i(q^i) \\ \text{subject to} \quad & q^i \in \mathcal{Q}^i, \quad i = 1, \ldots, n \\ & h(q^1, q^2, \ldots, q^n) = 0. \end{aligned} \tag{2.18}$$

Each node has a utility function $U^i(q)$ depending on its own private resource allocation q^i, constrained by the set $\mathcal{Q}^i$. The decision variables are coupled through the total resource constraint $h(q^1, q^2, \ldots, q^n) = 0$, which captures system-wide physical or economical limitations.

Resource allocation problems arise naturally in wireless networks, data communications, and smart grids, [76, 77, 78]. In data communications we optimize the data flows between n source-destination pairs through an $L-$link communications network by solving the utility minimization problem (2.18) with $h(q^1, q^2, \ldots, q^n) = \sum_{s \in \mathcal{S}_l} q^s - c_l$ for $l \in [1, L]$ [78]. Here, $\mathcal{S}_l$ is the set of source-destination pairs that use link l, and $U^i(\cdot)$ represents the utility of data flow i to communicate at rate q^i. In electric power systems, where problems on

the form of (2.18) are used to optimize the electricity generation and consumptions of a group of electric devices (*e.g.*, smart meters, household appliances and renewable generators), $h(q^1, q^2, \dots, q^n) = 0$ represents the physics of the grid.

The solution to problems on this form is typically decomposed by considering the dual problem [78, 79, 80, 81, 82, 83]. To illustrate this procedure, we consider the following dual problem which is equivalent to solving (2.18) (under mild technical conditions [32, chapter 5])

$$\underset{x}{\text{maximize}} \quad F(x) := \min_{q} L(q, x), \tag{2.19}$$

In this formulation, x is the dual variable, $F(x)$ is the dual objective function, and

$$L(q, x) = \sum_{i=1}^{n} U^i(q^i) + x^T h(q^1, \dots, q^n),$$

is the Lagrangian of Problem (2.18). The dual function is concave and the dual gradient (or a dual subgradient) is given by

$$\nabla F(x) = h(q^1(x), \dots, q^n(x)), \qquad q(x) = \operatorname*{argmin}_{q} L(q, x).$$

In many networks the dual gradient is obtained from measurements of the effect of the current decisions. Often, we only get a stochastic version of the gradient denoted by $g(x, \xi)$ where ξ is a random variable. If the primal problem has structure, then dual gradient methods can often be used to decompose its solution. For example, in many network applications $h(q^1(x), \dots, q^n(x)) = \sum_{i=1}^{n} h^i(q^1(x), \dots, q^n(x))$. Then, the equivalent dual problem (2.19) can be solved by gradient descent, leading to the following iteration

$$\begin{aligned} q^i_{k+1} =& \operatorname*{argmin}_{q} U^i(q^i_k) + \langle x_k, h^i(q^i_k) \rangle, \qquad i = 1, \dots, n \\ x_{k+1} =& x_k + \gamma g(x_k; \xi_k) \end{aligned}$$

where γ is a step-size parameter. Notice that the essential step in the algorithm is the communication of the stochastic dual gradient $g(x_k, \xi_k) \approx h(q^1_{k+1}, \dots, q^n_{k+1})$ which allows each node i to update q^i_{k+1} in parallel based on the dual variable x_k. To effectively communicate the gradient, it must first be compressed into a finite number of bits. Throughout the thesis, we study gradient descent with compression strategies, and compensation schemes which help obtain fast convergence guarantees despite limited information exchange.

2.6.2 Partial Gradient Communication

Problems on the form of (2.9) appear, *e.g.*, in machine learning and signal processing where we wish to find optimal estimators based on data from multiple

nodes. One important example is *empirical risk minimization* (ERM) where labelled data is split among n nodes which collaborate to find the optimal estimate. In particular, if each node $i \in \{1, 2, \ldots, n\}$ has access to its local data with feature vectors $\mathbf{z}^i = (z_1^i, \ldots, z_m^i)$ and labels $\mathbf{y}^i = (y_1^i, \ldots, y_m^i)$ with $z_j^i \in \mathbb{R}^d$ and $y_j^i \in \mathbb{R}$, then the local objective functions are

$$F^i(x) = \frac{1}{m} \sum_{j=1}^{m} \ell(x; z_j^i, y_j^i) + \frac{\lambda}{2} ||x||^2, \quad \text{for } i = 1, 2, \ldots, n \tag{2.20}$$

where $\ell(\cdot)$ is some loss function and $\lambda > 0$ is a regularization parameter. The ERM formulation covers many important machine learning problems. For example, we obtain the least-squares regression problem by letting $\ell(x; z, y) = (1/2)(y - z^T x)^2$, the logistic regression problem when $\ell(x; z, y) = \log(1 + \exp(-y \cdot z^T x))$, the support vector machine (SVM) problem if $\ell(x; z, y) = \left[1 - y \cdot z^T x\right]_+$, and the robust phase retrieval problem in image and speech processing if $\ell(x; z, y) = |(z^T x)^2 - y^2|$.

When the data set on each node is large, the above optimization problem can be typically solved by first-order methods under the master-slave architectures. One common algorithm is distributed stochastic gradient descent. In each iteration, the master node broadcasts a decision variable x_k, while each worker node i computes a stale stochastic gradient $g^i(x_{k-\tau_k^i})$ by evaluating its objective function gradient on a random subset of its local data $\mathcal{D}^i$. After the master receives the information from some worker nodes, it can perform the update

$$x_{k+1} = x_k - \gamma \frac{1}{n} \sum_{i=1}^{n} g^i \left(x_{k-\tau_k^i} \right). \tag{2.21}$$

Here, the stochastic gradient preserves the unbiasedness assumption, i.e.

$$\mathbf{E} g^i(x) = \nabla F^i(x), \quad \forall x \in \mathbb{R}^d. \tag{2.22}$$

Note that $\tau_i^k \in \mathbb{N}_0$ quantifies the communication delay when the gradient of worker node i is received by the master node at each iteration k. This communication delay model allows to study distributed stochastic gradient descent for both asynchronous and synchronous architecture (when $\tau_i^k = 0$ amd $\tau_i^k > 0$ for all i, k, respectively) [84, 85, 86]. In particular, this method has been studied and implemented under the asynchronous communication [26, 87, 84, 88, 89] and under the synchronous protocol [22, 84, 89, 45].

As the dimension of decision vectors continue to increase, a significant time is spent communicating gradient and decision vectors between computing nodes. For example, communication has been reported to account for up to 80% of the total training time of state-of-the-art deep neural network models with millions of parameters such as AlexNet, ResNet and LSTM [22]. To reduce communication time, it is suggested to increase the number of local updates and/or to compress the information on the optimization algorithms.

2.7 Federated Learning

Another emerging framework for partial gradient communication architectures in Section 2.6.2 is federated learning (FL), initially proposed by McMahan et al. [90]. Unlike traditional distributed first-order methods, FL methods utilize machines to cooperate to solve problems locally and transmit local model parameters, rather than private data, to a central server. To solve finite-sum optimization problems (2.9), FL methods perform the following update:

$$x_{k+1} = (1-\lambda)x_k + \lambda\frac{1}{n}\sum_{i=1}^{n}\mathcal{T}^p_{\gamma F^i}(x_k), \tag{2.23}$$

where γ, λ are positive tuning parameters. Popular FL methods `FedAvg` (also known as local SGD) [91, 92, 93] and `FedProx` [94] are covered by Equation (2.23) with $\mathcal{T}_{\gamma F^i}(x) = x - \gamma\nabla F^i(x)$ and with $\mathcal{T}_{\gamma F^i}(x) = \mathbf{prox}_{\gamma F^i}(x)$, respectively, and $\lambda = 1$.

Although FL cover data-sensitive applications in hospitals, insurance corporations and government agencies, it poses challenges from hardware and data heterogeneity (as different devices have different capabilities and data) to privacy and security issues (where data can be extracted from the transmitted parameters). Among these challenges, the communication overhead constitutes a major performance bottleneck of FL algorithms. This is because model parameters have typically huge size. Transmitting a single ML model (with 32 bits per element) requires 40 MB for a neural network with 10 million parameters [22]. It takes several hours for running FL on 4G connected devices to get a reasonable performance. This huge communication can overburden FL training over a network with limited resources. It is thus suggested to reduce communicated information on the FL methods to train large-scale problems over the communication-constrained networks.

2.8 Compression Operators

To reduce the time of transmitting the full-precision vector, it is recommended to compress the vector information. Two successful compression schemes include *sparsification* which transmits a few elements of the information, and *quantization* which converts the information to a low-precision representation.

Sparsification is one common compression scheme that transmits only the T most important elements of the full-precision vector $v \in \mathbb{R}^d$, i.e.

$$[Q(v)]^j = \begin{cases} v^j & \text{if } j \in I_T(v) \\ 0 & \text{otherwise.} \end{cases} \tag{2.24}$$

where $I_T(v)$ is the index set for the T components of v with the highest absolute magnitude. Rather than sending the T largest elements, the stochastic version

of sparsification is to send a few elements according to

$$[Q(v)]^j = (v^j/p^j)\xi^j, \tag{2.25}$$

where $\xi^j \sim \texttt{Bernouli}(p^j)$, $T = \sum_{j=1}^d p^j$ is the number of non-zero elements, and $p^j \in (0,1]$ is the probability of whether v^j is selected (i.e. $\xi^j = 1$). Ideally, p^j represents the magnitude of v^j, so that p^j should be large if $|v^j|$ is large relative to the other entries. There are many heuristics to choose p^j. For example, if we set $p^j = |v^j|/\|v\|_q$ with $q = 2$, $q = \infty$, and $q \in (0, \infty]$, then we get, respectively, `QSGD` in [22] with $s = 1$, `TernGrad` in [95], and the ℓ_q-quantization in [96]. We can also tune p automotically, see [96]. Further note that for the sparsified vector from (2.24) or (2.25), we need to encode the T largest elements and the vector sparsity pattern.

Quantization is another popular compression scheme reducing the precision of vector elements. One quantization example is sign compression which transmits only a positive scalar c and sign of every vector element, i.e.

$$[Q(v)]^j = c \cdot \text{sign}(v^j). \tag{2.26}$$

When we let $c = 1$, $c = \|v\|$ and $c = \|v\|_1/d$, sign compression in (2.26) recovers, respectively, the binary quantization [97], the ternary quantization [22] and the scaled binary quantization [29]. Also notice that the sign compression in (2.26) requires encoding the positive scalar c and the sign of all vector elements.

To improve communication efficiency, we may both sparsify and quantize the vector. For instance, we can combine the sparsification (2.24) with the quantization (2.26), i.e. for a positive scalar c

$$[Q(v)]^j = \begin{cases} c \cdot \text{sign}(v^j) & \text{if } j \in I_T(v) \\ 0 & \text{otherwise.} \end{cases} \tag{2.27}$$

Note that in the sparsification with quantization (2.27) we encode the positive scalar c, the vector sparsity pattern, and the sign of all non-negative vector elements.

To this end, we introduce three main definitions which capture the compression schemes of broad interest: ϵ-compressor, bounded relative error quantizer and unbiased random quantizer.

2.8.1 ϵ-Compressor

The operator $Q(\cdot)$ is an ϵ-compressor if the Euclidean distance between the full vector and its compressed version is bounded by a positive scalar ϵ, i.e.

$$\|Q(v) - v\| \leq \epsilon, \quad \forall v \in \mathbb{R}^d.$$

The ϵ-compressor requires only the bounded magnitude of the compression errors. A small value of ϵ implies the high precision of given compression.At the extreme when $\epsilon = 0$, we have $Q(v) = v$.

2.8.2 Bounded Relative Error Quantizer

The operator $Q(\cdot)$ is called a *bounded relative error quantizer (BREQ)* if the compressed version of the vector v satisfies the following inequalities

$$\langle Q(v), v\rangle \geq \sigma\|v\|^2, \quad \text{and} \quad \|Q(v)\|^2 \leq \beta\|v\|^2$$

for positive constants σ, β. Note that as σ and β both are close to 1, the BREQ implies high precision of given compression (e.g. $Q(v) = v$ when $\sigma = \beta = 1$ for extreme cases). In addition, these conditions imply that

$$\|Q(v) - v\|^2 \leq (1 - 2\sigma + \beta)\|v\|^2.$$

Unlike the ϵ-compressor, the BREQ has a compression error which scales linearly in the Euclidean norm of the input vector v. If $\|v\| \leq C$ for some positive constants C, then the BREQ is the ϵ-compressor with $\epsilon = C\sqrt{1 - 2\sigma + \beta}$.

2.8.3 Unbiased Random Quantizer

The operator $Q(\cdot)$ is called a *unbiased random quantizer (URQ)* if the compressed version of the vector v satisfies

$$\mathbf{E}[Q(v)] = v, \quad \text{and} \quad \mathbf{E}\|Q(v)\|^2 \leq \alpha\|v\|^2$$

for a positive scalar α. Note that $Q(\cdot)$ is unbiased and variance-bounded. Unlike the ϵ-compressor and the BREQ, the URQ requires the unbiased property of $Q(v)$ and yields $Q(v)$ which is close to v as α is close to 1. In addition, by these conditions the URQ satisfies

$$\mathbf{E}\|Q(v) - v\|^2 \leq (\alpha - 1)\|v\|^2.$$

Like the BREQ, the URQ has the compression error scaling linearly in the Euclidean norm of v. If $\|v\| \leq C$ for some positive constants C, the URQ is the ϵ-compressor with $\epsilon = C\sqrt{\alpha - 1}$.

Throughout the thesis, we provide detailed definitions and examples of all stated compression operators. In particular, we present a unified analysis for first-order algorithms with each family of compressors in the next chapter.

Part I

Analysis

Chapter 3

Compressed Gradient Methods

Distributed machine learning problems involve empirical risk minimization, and can be cast as the following finite-sum optimization problem

$$\underset{x\in\mathbb{R}^d}{\text{minimize}} \quad F(x) = \sum_{i=1}^{n} F^i(x). \tag{3.1}$$

Here, $x \in \mathbb{R}^d$ is a vector representing the d parameters of a machine learning model, and each $F^i(\cdot)$ is the loss based on a single data point or subset of data points locally stored on node i. To solve these problems on massive training data within desirable training times, one can resort to the partial gradient communication architecture (in Section 2.6.2) that splits computational tasks among multiple nodes. The architecture is either a *synchronous* or an *asynchronous* version.

In the synchronous architecture, a master node waits for all the information evaluated by worker nodes before it makes an update [98, 84]. One natural implementation of first-order methods on this architecture is *gradient descent* (GD). After receiving all the gradients by the worker nodes $\nabla F^i(\cdot)$, the master node updates the iterate x_k via:

$$x_{k+1} = x_k - \gamma \sum_{i=1}^{n} \nabla F^i(x_k) \tag{3.2}$$

for some positive step-size γ. However, insisting on the synchronous operation leads to long communication times (due to waiting for the slowest worker node to complete) and the benefits of parallelization diminish as the number of worker nodes increases.

To alleviate this bottleneck, the asynchronous architecture (such as the parameter server framework [99]) enables the master node to update its parameters every time it receives new information from a worker node. Among popular first-order methods naturally implemented in this architecture is *incremental aggregate gradient* (IAG) [100], where the master node executes the update according to:

$$x_{k+1} = x_k - \gamma \sum_{i=1}^{n} \nabla F^i(x_{k-\tau_k^i}). \tag{3.3}$$

Here τ_k^i describes the staleness of the gradient information from worker node i available to the master node at iteration k. Under the assumption of bounded

staleness, $\tau_k^i \leq \tau$ for all k, i, convergence guarantees for IAG have been established for several classes of loss functions, see *e.g.* [100, 27, 101, 26].

To analyze performance of these distributed first-order algorithms, *iteration complexity* is the traditional performance indicator which measures total running time of the algorithm. The iteration complexity is well-studied; fundamental lower bounds have been derived [102, 33] and algorithms with order-optimal convergence rates have been developed [34, 33]. However, when problems become large in dimension d and the number of data points, the communication cost for exchanging gradients between the nodes escalates. Therefore, designing the algorithm that reduces the number of iterations k to obtain an optimal solution with target accuracy is no longer sufficient. Rather, it becomes essential to design the algorithm that minimizes the total amount of communication required to reach the target solution accuracy.

A natural way to limit the amount of exchanged communication is to compress the gradients before transmitting them between the nodes in the network. The gradient compression can be stochastic [95, 103, 22] or deterministic [82, 22, 104], and empirical studies have demonstrated that they can yield significant savings for many gradient-based algorithms in network traffic [103, 22, 95, 105, 95, 82]. To understand theoretical effect of gradient compression, it is essential to study *communication complexity* which measures amounts of communication required to reach an ϵ-optimal solution. Although the communication complexity has received some attention in the past [106, 107, 108, 109], this concept is much less well understood than iteration complexity. For instance, lower bounds on communication complexity exist only for special classes of functions, and optimal algorithms are yet unknown even for unconstrained problems.

In this chapter, we establish unified analysis for first-order algorithms operating on two families of deterministic and stochastic gradient compression: the bounded relative error quantizer (BREQ) and unbiased randomized quantizers (URQ). We characterize and utilize useful inequalities for each family of compression to establish per-iteration convergence rates of GD and IAG operating on compressed gradients. Our convergence rate results give explicit expressions for how compression accuracy and staleness bounds affect the expected running time of the algorithms to reach the ϵ-accuracy. Based on these results, we characterize the trade-off between iteration and communication complexity under gradient compression. Finally, we validate the theoretical results on parameter estimation problems over benchmark data sets.

3.1 Gradient Compression

To study the iteration and communication complexity of first-order algorithms using gradient compression, we first consider BREQ and URQ. Both of these

definitions cover deterministic and stochastic gradient compression of practical interest in the literature.

3.1.1 Bounded Relative Error Quantizer (BREQ)

We define BREQ which represent general deterministic gradient compression.

Definition 3.1. The operator $Q : \mathbb{R}^d \to \mathbb{R}^d$ is called a *bounded relative error quantizer (BREQ)* if for all $v \in \mathbb{R}^d$ and for some positive constants α, β

(a) $\langle Q(v), v\rangle \geq \alpha\|v\|^2$, and

(b) $\|Q(v)\|^2 \leq \beta\|v\|^2$.

Note the first inequality is satisfied by any quantizer for which $\operatorname{sgn}(Q(v)) = \operatorname{sgn}(v)$. Moreover, conditions (a) and (b) imply that

$$\|Q(v) - v\|^2 \leq (1 - 2\alpha + \beta)\|v\|^2,$$

so the relative quantization error induced by Q is indeed bounded. Thus, the BREQ parameters α, β quantifies compression accuracy. For the extreme case when $\alpha = \beta = 1$, $Q(v) = v$. Next, we give three main deterministic gradient compression examples which are BREQs.

Examples of BREQs

Gradient compression schemes can be sparsification which sets small vector elements to be zero, quantization which each vector element is represented by a low number of bits, or both. The first scheme is deterministic sparsification which can be done by keeping top K elements with the highest absolute magnitude. We define this formally below.

Definition 3.2. The deterministic sparsification $Q_G^K : \mathbb{R}^d \mapsto \mathbb{R}^d$ is

$$[Q_G^K(v)]^i = \begin{cases} [v]^{\pi(i)} & \text{if } i \leq K \\ 0 & \text{otherwise} \end{cases}$$

where π is a permutation of $\{1, \ldots, d\}$ such that $|v^{\pi(k)}| \geq |v^{\pi(k+1)}|$ for all $k \in \{1, \ldots, d-1\}$.

The case $K = 1$ has been treated by Nutini *et al.* [110]. A naive encoding of a vector processed by the deterministic sparsification requires $K(\log_2(d) + b)$ bits: $\log_2(d)$ bits to represent each index and b bits to represent the corresponding entry of the K non-zero values.

Another compression scheme to reduce the size of the gradient vector is to quantize the individual elements. At the extreme, one can consider three-level

(ternary) quantization, where each vector element is quantized to the levels $\{-1, 0, 1\}$. The convergence of gradient descent with the three-level quantizer has been studied in [82]. One drawback with this quantizer is that the absence of magnitude information about the original gradient leads to a residual error in the gradient descent. To avoid this problem, one can rather code each element of v to $\{-\|v\|, 0, \|v\|\}$. We thus consider the following quantizer:

Definition 3.3. The deterministic ternary quantization $Q_T : \mathbb{R}^d \mapsto \mathbb{R}^d$ is

$$[Q_T(v)]^i = \|v\|\text{sgn}(v^i).$$

The required number of bits to encode the gradient by the deterministic ternary quantization is $2d + b$: b bits to encode the norm of the vector, and 2 bits for each element to encode its sign.

Finally, one can also combine sparsification and quantization; the compressed gradient is then represented by its (uncompressed) magnitude and the sign of a few entries. Such compression has been recently proposed in, *e.g.*, [22, 95]. We now provide the definition.

Definition 3.4. The deterministic sparsification and quantization $Q_D : \mathbb{R}^d \mapsto \mathbb{R}^d$ is defined as

$$[Q_D(v)]^i = \begin{cases} \|v\|\text{sgn}(v^i) & \text{if } i \in I(v) \\ 0 & \text{otherwise} \end{cases}$$

where $I(v)$ is the smallest subset of $\{1, \ldots, d\}$ such that

$$\sum_{i \in I(v)} |v^i| \geq \|v\|.$$

The deterministic sparsification and quantization, analyzed in Alistarh *et al.* [22], requires $|I(v)|(\log_2(d) + 1) + b$ bits to encode the gradient.

Furthermore, the above-mentioned compression schemes are easily proved to be all the BREQ with the following parameters:

Lemma 3.1. *The deterministic sparsification $Q_G^K(\cdot)$ is a BREQ with $\alpha = K/d$ and $\beta = 1$. In addition, for any $g \in \mathbb{R}^d$,*

$$(K/d)\|v\|^2 \leq \|Q_G^K(v)\|^2 \tag{3.4}$$

Lemma 3.2. *The deterministic ternary quantization $Q_T(\cdot)$ is a BREQ with $\alpha = 1, \beta \leq d$.*

Lemma 3.3. *The deterministic sparsification and quantization $Q_D(\cdot)$ is a BREQ with $\alpha = 1, \beta \leq \sqrt{d}$.*

Lemma 3.1, 3.2 and 3.3 all quantify how accurate the compressors are. For example, the deterministic sparsification with $K = d$ leads to the condition that $Q(v) = v$, while the deterministic ternary quantization is less accurate than the deterministic sparsification and quantization by $\sqrt{d}$.

3.1.2 Unbiased Random Quantizer (URQ)

We next define URQ which covers general stochastic gradient compression.

Definition 3.5. A mapping $Q : \mathbb{R}^d \to \mathbb{R}^d$ is called an *unbiased random quantizer* if, for every $v \in \mathbb{R}^d$,

1. $\text{supp}(Q(v)) \subseteq \text{supp}(v)$
2. $\mathbf{E}\{Q(v)\} = v$
3. $\mathbf{E}\{\|Q(v)\|^2\} \leq \alpha\|v\|^2$

for some finite positive α. In addition, Q is said to be *sign-preserving* if

$$[Q(v)]^i v^i \geq 0$$

for every $v \in \mathbb{R}^d$ and $i \in [1, d]$.

URQs satisfy some additional useful inequalities. First,

$$\mathbf{E}\{\|Q(v)\|_0\} \leq c,$$

for any $v \in \mathbb{R}^d$ and a finite positive constant $c \leq d$. The sign-preserving property guarantees the same direction between the compressed vector and the full one. This property of Q also implies that

$$\mathbf{E}\|Q(v) - v\|^2 \leq \beta\|v\|^2,$$

for any $v \in \mathbb{R}^d$ and a finite positive constant $\beta \leq \alpha - 1$. As we will show next, it is typically possible to derive better bounds for c and β when we consider specific classes of stochastic gradient compression.

Examples of URQs

Several stochastic gradient compression algorithms have been proposed for solving distributed problems under limited communications. Like BREQs in Section 3.1.1, stochastic gradient compression can use either sparsification or quantization techniques. Important examples include the *stochastic sparsification* [103], the *stochastic ternary quantization* [95], and the *stochastic sparsification and quantization* [22] as defined below.

Definition 3.6. The stochastic sparsification $S : \mathbb{R}^d \to \mathbb{R}^d$ is defined as

$$S^i(v) = \begin{cases} v^i/p^i & \text{with probability } p^i \\ 0 & \text{otherwise} \end{cases},$$

where p^i is probability that coordinate i is selected.

Note that when the stochastic sparsification uses the same probability for each coordinate, it will effectively result in a randomized coordinate descent. Choosing $p^i = |v^i|/\|v\|$, on the other hand, will result in the stochastic ternary quantization [95]:

Definition 3.7. The stochastic ternary quantization $T : \mathbb{R}^d \to \mathbb{R}^d$ is defined as

$$T^i(v) = \begin{cases} \|v\| \operatorname{sign}(v^i) & \text{with probability } |v^i|/\|v\| \\ 0 & \text{otherwise} \end{cases}.$$

The stochastic sparsification and quantization [22], defined next, combines sparsification of the gradient vector with quantization of its element to further reduce the amount of information exchanged.

Definition 3.8. The stochastic sparsification and quantization $Q_b : \mathbb{R}^d \to \mathbb{R}^d$ is defined as

$$Q_b^i(v) = \|v\| \operatorname{sign}(v^i)\xi(v, i, s),$$

where

$$\xi(v, i, s) = \begin{cases} l/s & \text{with probability } 1 - p\left(|v^i|/\|v\|, s\right) \\ (l+1)/s & \text{otherwise} \end{cases},$$

and $p(a, s) = as - l$ for any $a \in [0, 1]$. Here, s is the number of quantization levels distributed between 0 and 1, and $l \in [0, s)$ such that $|v^i|/\|v\| \in [l/s, (l+1)/s]$.

Notice that the stochastic ternary quantization is the stochastic sparsification and quantization with $s = 1$ (and hence $l = 0$). In addition, these aforementioned compression examples can be easily shown to be sign-preserving URQs. Specifically, we have the following results:

Proposition 3.1 ([103])**.** *The stochastic sparsification $S : \mathbb{R}^d \to \mathbb{R}^d$ is a sign-preserving URQ, which satisfies*

1. $\mathbf{E}\{\|S(v)\|^2\} \leq (1/p_{\min})\|v\|^2$ *where* $p_{\min} = \min_{i \in [1,d]} p^i$ *, and*
2. $\mathbf{E}\{\|S(v))\|_0\} = \sum_{i=1}^d p^i$.

Proposition 3.2 (Lemma 3.4 in [22])**.** *The stochastic sparsification and quantization $Q_b : \mathbb{R}^d \to \mathbb{R}^d$ is a sign-preserving URQ, which satisfies*

1. $\mathbf{E}\{\|Q_b(v)\|^2\} \leq \left(1 + \min\left(d/s^2, \sqrt{d}/s\right)\right)\|v\|^2$, *and*
2. $\mathbf{E}\{\|Q_b(v))\|_0\} \leq s(s + \sqrt{d})$.

Proposition 3.1 and 3.2 both imply that $\mathbf{E}\|Q(v)\|^2$ is close to $\|v\|^2$ if the URQs are sufficiently accurate; *e.g.*, when we set $p^i = 1$ for all i in the stochastic sparsification (we send the full vector) and when we let $s \to \infty$ in the stochastic sparsification and quantization (we send the exact solution). Although the probability p^i in the stochastic sparsification can be time-varying (*e.g.*, when we set $p^i \propto v^i$), we assume a time-invariant α-value in the analysis to simplify notation.

3.2 Assumptions

To facilitate our analysis throughout this chapter, we impose the following assumptions which are commonly used on the optimization problem (3.1).

Assumption 3.1. Each $F^i : \mathbb{R}^d \to \mathbb{R}$ is L-smooth, i.e. there exists $L > 0$ such that

$$F^i(y) \leq F^i(x) + \langle \nabla F^i(x), y - x \rangle + \frac{L}{2}\|y - x\|^2 \qquad \forall x, y \in \mathbb{R}^d.$$

Note that Assumption 3.1 implies that F is also smooth with $\bar{L} \leq mL$.

Assumption 3.2. The whole objective function $F : \mathbb{R}^d \to \mathbb{R}$ is $\mu-$strongly convex, *i.e.* there exits $\mu > 0$ such that

$$F(y) \geq F(x) + \langle \nabla F(x), y - x \rangle + \frac{\mu}{2}\|y - x\|^2 \qquad \forall x, y \in \mathbb{R}^d.$$

Assumption 3.3. Each $F^i : \mathbb{R}^d \to \mathbb{R}$ has a bounded gradient, i.e. there exists a positive scalar C such that

$$\|\nabla F^i(x)\| \leq C.$$

Notice that the optimization problem satisfying Assumption 3.3 is the low-rank least-squares matrix completion problem which arises in Euclidean distance estimation, clustering and other applications [105, 111].

Furthermore, by exploiting a data sparsity pattern we can derive a smaller Lipschitz constant which enable larger step-sizes and faster convergence rates. To quantify the sparsity pattern, we further make the additional assumption as follows:

Assumption 3.4. Each $F^i : \mathbb{R}^d \to \mathbb{R}$ can be written as $F^i(x) = \ell(\langle a^i, x \rangle, b^i)$, such that $\text{supp}(\nabla F^i(x)) = \text{supp}(a^i)$ for given data $\left\{(a^i, b^i)\right\}_{i=1}^n$ with $a^i \in \mathbb{R}^d$ and $b^i \in \mathbb{R}$.

Note that Assumption 3.4 does not require boundedness of the gradients. Both least-squares problems, where $F^i(x) = (1/2)(\langle a^i, x \rangle - b^i)^2$ and $\nabla F^i(x) = (\langle a^i, x \rangle - b^i)a^i$, and logistic regression problems, where $F^i(x) =$

$\log(1+\exp(-b^i\langle a^i,x\rangle))$ and $\nabla F^i(x) = -\left(b^i/\left(1+\exp(-b^i\langle a^i,x\rangle)\right)\right)a^i$ satisfy the assumption. When Assumption 3.4 is satisfied, the sparsity pattern of component function gradients can be computed off-line directly from the data. We will consider two important sparsity measures: the average and maximum conflict graph degree of the data, defined as

$$\Delta_{\text{ave}} = \frac{1}{n}\sum_{i=1}^{n}\left\{\sum_{j=1,j\neq i}^{n}\mathbf{1}\{\text{supp}(a^i)\cap\text{supp}(a^j)\neq\emptyset\}\right\}, \quad \text{and}$$

$$\Delta_{\max} = \max_{i\in[1,n]}\left\{\sum_{j=1,j\neq i}^{n}\mathbf{1}\{\text{supp}(a^i)\cap\text{supp}(a^j)\neq\emptyset\}\right\}.$$

These sparsity measures allow us to derive a tighter bound $\bar{L}$ for the Lipschitz constant of the total loss $F(x)=\sum_{i=1}^n F^i(x)$, as shown next:

Lemma 3.4. *Consider Problem* (3.1) *under Assumption 3.4. If $\ell(\cdot)$ is smooth with $L>0$, then the gradient of the total loss $F(x)=\sum_{i=1}^n F^i(x)$ is smooth with*

$$\bar{L} = L\sqrt{n(1+\Delta)},$$

where $\Delta=\min(\Delta_{\text{ave}},\Delta_{\max})$.

Proof. See Appendix 3.C.1. □

These sparsity measures are used to tighten our convergence results.

3.3 Convergence Analysis under BREQ

In this section, we establish a unified convergence analysis of the gradient descent algorithm under BREQ in terms of both iteration counts and number of communicated bits. We thus consider

$$x_{k+1} = x_k - \gamma Q\left(\nabla F(x_k)\right), \tag{3.5}$$

where γ is a positive step size and $Q(\cdot)$ is the BREQ introduced in Section 3.1.1. The following theorem characterizes its convergence rate results.

Theorem 3.1. *Consider Problem* (3.1)*, and the iterates $\{x_k\}_{k\in\mathbb{N}}$ generated by gradient descent with BREQ in* (3.5) *with $\gamma=\alpha/(\beta L)$.*

a) ***Strongly-convex:*** *If $F(\cdot)$ is L-smooth and μ-strongly convex, then*

$$F(x_k)-F(x^\star)\leq\left(1-\mu\frac{\alpha^2}{2\beta L}\right)^k\left(F(x_0)-F(x^\star)\right).$$

b) ***Convex:*** *If $F(\cdot)$ is convex and L-smooth, then*

$$F(x_k) - F(x^\star) \leq \frac{1}{k}\frac{2\beta L}{\alpha^2}R^2,$$

where $R \geq \|x_k - x^\star\|$ for all k.

c) ***Non-convex:*** *If the whole objective function $F(\cdot)$ is L-smooth, then*

$$\min_{l\in[0,k-1]} \|\nabla F(x_l)\|^2 \leq \frac{1}{k}\frac{2\beta L}{\alpha^2}\left(F(x_0) - F(x_k)\right),$$

Proof. See Appendix 3.B.2. □

From Theorem 3.1, gradient descent under BREQ attains linear convergence for strongly convex problems and sub-linear convergence for convex and non-convex problems. In particular, the tuning parameter and the convergence rate for gradient descent under BREQ are penalized by α/β and by $\alpha^2/(2\beta)$, respectively, compared to a similar analysis of full-precision gradient descent. This penalized factor quantifies the accuracy of BREQ. For instance, the factor $\alpha^2/(2\beta)$ ranges from $1/(2\sqrt{d})$ for the deterministic ternary quantization to $K^2/(2d^2)$ for the deterministic sparsification. This theorem also allows us to estimate the iteration and communication complexity, which are penalized by the precision of BREQ $\alpha^2/(2\beta)$, as shown next.

Corollary 3.1. *Consider Problem* (3.1) *where the whole objective function $f(\cdot)$ is L-smooth, and the iterates $\{x_k\}_{k\in\mathbb{N}}$ generated by gradient descent with BREQ in* (3.5) *with $\gamma = \alpha/(\beta L)$. Let c be the required number of bits to encode one compressed vector.*

a) ***Strongly-convex:*** *If $F(\cdot)$ is L-smooth and μ-strongly convex, then the algorithm reaches $F(x_T) - F(x^\star) \leq \varepsilon$ within*

$$T^\star = \frac{2\beta}{\alpha^2}\frac{L}{\mu}\log\left(\frac{F(x_0) - F(x^\star)}{\varepsilon}\right),$$

iterations under which $B^\star = \lceil cT^\star \rceil$ bits are sent.

b) ***Convex:*** *If $F(\cdot)$ is convex and L-smooth, then the algorithm reaches $F(x_T) - F(x^\star) \leq \varepsilon$ within*

$$T^\star = \frac{2\beta}{\alpha^2}\frac{LR^2}{\varepsilon},$$

iterations under which $B^\star = \lceil cT^\star \rceil$ bits are sent. Here, $R \geq \|x_k - x^\star\|$ for all k.

c) ***Non-convex:*** *If the whole objective function $F(\cdot)$ is L-smooth, then the algorithm reaches $\min_{k\in[0,T-1]} \|\nabla F(x_k)\|^2 \leq \varepsilon$ within*

$$T^\star = \frac{2\beta}{\alpha^2}\frac{L[F(x_0) - F(x_T)]}{\varepsilon},$$

iterations under which $B^\star = \lceil cT^\star \rceil$ bits are sent.

Proof. See Appendix 3.B.3. □

3.3.1 Tighter Convergence of Deterministic Sparsification

The deterministic sparsification has a small value of α, which translates into a high convergence penalty in the unified analysis. However, by exploiting the lower bound on $\|Q(v)\|$ stated in Lemma 3.1, we can give convergence guarantees that are of the same order of magnitude as the other BREQs.

Theorem 3.2. *Consider Problem* (3.1) *and the iterates $\{x_k\}_{k\in\mathbb{N}}$ generated by (3.5) using the deterministic sparsification with $\gamma = 1/L$.*

a) ***Strongly-convex:*** *If $F(\cdot)$ is L-smooth and μ-strongly convex, then*

$$F(x_k) - F(x^\star) \leq \left(1 - \frac{K}{2d}\frac{\mu}{L}\right)^k (F(x_0) - F(x^\star)).$$

b) ***Convex:*** *If $F(\cdot)$ is convex and L-smooth, then*

$$F(x_k) - F(x^\star) \leq \frac{1}{k}\frac{2dL}{K}R^2,$$

where $R \geq \|x_k - x^\star\|$ for all k.

c) ***Non-convex:*** *If the whole objective function $F(\cdot)$ is L-smooth, then*

$$\min_{l\in[0,k-1]} \|\nabla F(x_l)\|^2 \leq \frac{1}{k}\frac{2dL}{K}\left(F(x_0) - F(x_k)\right).$$

Proof. See Appendix 3.B.4. □

Theorem 3.2 implies the d/K larger step-size and convergence speed than Theorem 3.1. In the similar fashion for the generic BREQ analysis, we can derive the following bound on the communication complexity of the deterministic sparsification methods.

Corollary 3.2. *Consider Problem* (3.1) *and the iterates $\{x_k\}_{k\in\mathbb{N}}$ generated by (3.5) using the deterministic sparsification with $\gamma = 1/L$. Let c be the required number of bits to encode one compressed vector.*

a) ***Strongly-convex:*** *If $F(\cdot)$ is L-smooth and μ-strongly convex, then the algorithm reaches $F(x_T) - F(x^\star) \leq \varepsilon$ within*

$$T^\star = \frac{2d}{K}\frac{L}{\mu}\log\left(\frac{F(x_0) - F(x^\star)}{\varepsilon}\right),$$

iterations under which $B^\star = \lceil cT^\star \rceil$ bits are sent.

b) ***Convex:*** *If $F(\cdot)$ is convex and L-smooth, then the algorithm reaches $F(x_T) - F(x^\star) \leq \varepsilon$ within*

$$T^\star = \frac{2d}{K}\frac{LR^2}{\varepsilon},$$

iterations under which $B^\star = \lceil cT^\star \rceil$ bits are sent. Here, $R \geq \|x_k - x^\star\|$ for all k.

c) ***Non-convex:*** *If the whole objective function $F(\cdot)$ is L-smooth, then the algorithm reaches $\min_{k\in[0,T-1]} \|\nabla F(x_k)\|^2 \leq \varepsilon$ within*

$$T^\star = \frac{2d}{K}\frac{L[F(x_0) - F(x_T)]}{\varepsilon},$$

iterations under which $B^\star = \lceil cT^\star \rceil$ bits are sent.

Proof. See Appendix 3.B.5. □

From Corollary 3.1 and 3.2, iteration and communication complexities of GD with different BREQs (3.5) for non-convex problems are summarized in Table 3.1. On the one hand, the deterministic sparsification with $K \geq \sqrt{d}$ achieves the lower iteration complexity than the deterministic ternary quantization, and deterministic sparsification and quantization. On the other hand, if $K_{\max} := \max_k |\text{supp}(Q(\nabla f(x_k)))| < \min(\sqrt{d}, 2d/[\log_2(d) + 1])$, then the deterministic sparsification and quantization provides the lower communication complexity than the deterministic sparsification and deterministic ternary quantization. We validate this observation in our numerical experiments in the next section.

3.3.2 Experimental Results

We evaluated the performance of GD with different BREQs on a least-squares problem. This problem is on the form (3.1) with $F(x) = (1/2)\|Ax - b\|^2$ where $A \in \mathbb{R}^{m\times d}$ and $b \in \mathbb{R}^m$. Test instances with $m = 1000$ and $d = 800$ were created as follows: each element of A was drawn from $\mathcal{U}(0, 1)$ and each element of b was set to be the sign of a random number drawn from $\mathcal{N}(0, 1)$. We normalized each row of A by its Euclidean norm and computed the Lipschitz constant as $L = \lambda_{\max}(A^T A)$. The compressed gradient iterations were

Quantizer	γ	$T^\star$	$B^\star$
Deterministic sparsification	$\frac{1}{L}$	$\frac{d}{K}\nu$	$\lceil d(\log_2(d)+b)\nu \rceil$
Deterministic ternary quantization	$\frac{1}{dL}$	$d\nu$	$\lceil K_{\max}(2d+b)\nu \rceil$
Deterministic sparsification and quantization	$\frac{1}{\sqrt{d}L}$	$\sqrt{d}\nu$	$\lceil K_{\max} C_D \nu \rceil$

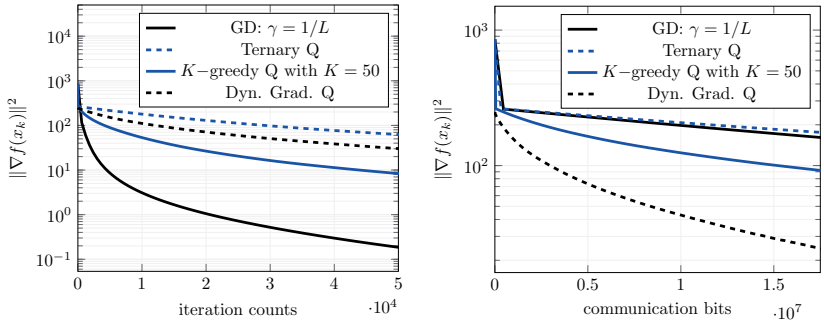
Table 3.1: iteration and communications complexity of compressed gradient descent for non-convex problems, where $C_D = K_{\max}(\log_2(d)+1)+b$, $K_{\max} = \max_k K_k$, $K_k = |\text{supp}(Q(\nabla f(x_k)))|$ and $\nu = 2L\varepsilon_0/\varepsilon$.

initialized from $x_0 = \mathbf{0}$, and we assumed that real numbers were represented by $b = 64$ bits. The step sizes under different gradient compression algorithms are tuned according to Table 3.1.

From Figure 3.1a, GD has the fastest convergence towards the optimum in terms of iteration counts, since all gradient compression techniques introduce an information loss. Furthermore, since $K \geq \sqrt{d}$ in this case, GD with the deterministic sparsification provides lower iteration complexity to reach the ϵ-accuracy than GD using other compressors. However, GD with compression tends to have better performance than GD in terms of the number of communicated bits; see Figure 3.1b. The exception is the deterministic ternary quantization, which is uniformly worse than its alternatives, possibly due to its small theoretically justified step size. Figure 3.1b indicates that GD with the deterministic sparsification and quantization attains the best communication complexity among GD with other compressors.

3.4 Convergence Analysis under URQ

In this section, we provide a unified convergence analysis of gradient descent under URQ. We show that the unbiasedness property of URQ can be used to obtain theoretical guarantees for both full and partial gradient architectures. Our results show explicit dependency of convergence on compression accuracy and asynchrony.



(a) Iteration counts (b) communicated bits

Figure 3.1: The performance of gradient descent with different BREQs and full-precision gradient descent. Here, K-`greedy Q`, `Ternary Q` and `Dyn. Grad. Q` are the deterministic sparsification, the deterministic ternary quantization, and the deterministic sparsification and quantization, respectively.

3.4.1 Full Gradient Communication

To build the intuition on the effect of URQ, we study gradient descent under the full gradient communication architecture. The result is of significance for distributed optimization using dual decomposition methods, where the dual function is optimized typically by using gradient descent (see details in Section 2.6.1). Furthermore, it complements and improves on results for the BREQ in Section 3.3. In essence, the convergence of algorithms using the URQ, like the BREQ, is shown to depend on the compression accuracy. This theoretical guarantee leads to explicit formulas for the iteration and communication complexity of GD under URQ. In addition, the result establishes a baseline for the partial gradient communication architectures later.

To this end, we start by considering compressed gradient descent, which updates the solution $\{x_k\}_{k\in\mathbb{N}}$ via the following recursion

$$x_{k+1} = x_k - \gamma Q(\nabla F(x_k)), \tag{3.6}$$

where γ is the fixed, positive step size, and $Q(\cdot)$ is the URQ. Throughout this section, we derive explicit expressions for how the variance bound α of the URQ affects admissible step-sizes and guaranteed convergence times. The next result charaterizes the convergence of GD with URQ (3.6).

Theorem 3.3. *Consider Problem* (3.1)*, and the iterates* $\{x_k\}_{k\in\mathbb{N}}$ *generated by* (3.6)*. Let* $\Delta = \min(\Delta_{\text{ave}}, \Delta_{\max})$ *and* $\bar{L} = L\sqrt{n(1+\Delta)}$*. Then,*

1. ***Strongly-convex:*** *If* $\gamma = (1/\alpha)\left(2/(\mu+\bar{L})\right)$*, and Assumption 3.1, 3.2*

and 3.4 hold, then

$$\mathbf{E}\|x_k - x^\star\|^2 \leq \left(1 - \frac{1}{\alpha}\frac{4\mu\bar{L}}{(\mu+\bar{L})^2}\right)^k \|x_0 - x^\star\|^2.$$

2. ***Convex:*** *If $\gamma = 1/(\alpha\bar{L})$, and Assumption 3.1 and 3.4 hold, then*

$$\mathbf{E}\left(F(x_k) - F(x^\star)\right) \leq \frac{\alpha\bar{L}}{2(k+1)}\|x_0 - x^\star\|^2.$$

Proof. See Appendix 3.C.2. □

Theorem 3.3 quantifies how the convergence guarantees depend on α. If the worker node sends the exact gradient, *i.e.* $Q(\nabla F(x_k)) = \nabla F(x_k)$, $\alpha = 1$. Also, this theorem with $\alpha = 1$ recovers the convergence rate of full-precision gradient descent for strongly-convex problems with $\gamma = 2/(\mu+\bar{L})$ and convex problems with $\gamma = 1/\bar{L}$, presented in [112, 73]. If the quantizer produces a less accurate vector (larger α), then we must decrease the step size γ to guarantee numerical stability at the cost of higher iteration and communication complexity to reach the target solution accuracy.

To illustrate this, notice that one naive encoding of a vector processed by the URQ requires $c(\log_2 d + B)$ bits: $\log_2 d$ bits to represent each index and B bits to represent the corresponding vector entry of c non-zero values. Hence, Theorem 3.3 yields the following iteration and communication complexity that all are penalized by the URQ precision α.

Corollary 3.3. *Consider Problem* (3.1)*, and the iterates $\{x_k\}_{k\in\mathbb{N}}$ generated by* (3.6)*. Let $\Delta = \min(\Delta_{\text{ave}}, \Delta_{\max})$, $\bar{L} = L\sqrt{n(1+\Delta)}$, B be the number of bits required to encode a single vector entry, and $\mathbf{E}\{\|Q(v)\|_0\} \leq c$. Then,*

1. ***Strongly-convex:*** *If $\gamma = (1/\alpha)\left(2/(\mu+\bar{L})\right)$, and Assumption 3.1, 3.2 and 3.4 hold, then the algorithm in* (3.6) *reaches $\mathbf{E}\|x_T - x^\star\|^2 \leq \varepsilon$ within*

$$T^\star = \alpha\frac{(\mu+\bar{L})^2}{4\mu\bar{L}}\log\left(\frac{\|x_0 - x^\star\|^2}{\varepsilon}\right)$$

iterations, under which

$$B^\star = \left\lceil (\log_2 d + B)\, c \cdot \alpha\frac{(\mu+\bar{L})^2}{4\mu\bar{L}}\log\left(\frac{\|x_0 - x^\star\|^2}{\varepsilon}\right)\right\rceil$$

bits are sent.

2. ***Convex:*** *If $\gamma = 1/(\alpha\bar{L})$, and Assumption 3.1 and 3.4 hold, then the algorithm in* (3.6) *reaches* $\mathbf{E}\left(F(x_T) - F(x^\star)\right) \leq \varepsilon$ *within*

$$T^\star = \frac{\alpha\bar{L}}{2} \cdot \frac{\|x_0 - x^\star\|^2}{\varepsilon}$$

iterations, under which

$$B^\star = \left\lceil \left(\log_2 d + B\right) c \cdot \frac{\alpha\bar{L}}{2} \cdot \frac{\|x_0 - x^\star\|^2}{\varepsilon} \right\rceil$$

bits are sent.

Proof. See Appendix 3.C.3. □

Next, we conclude this section by studying the following compressed IAG algorithm: given an initial point x_0 and a fixed, positive step size γ

$$x_{k+1} = x_k - \gamma Q\left(\sum_{i=1}^{n} \nabla F^i(x_{k-\tau_k^i})\right). \tag{3.7}$$

The iteration accounts for heterogeneous worker delays, but performs a centralized compression of the sum of staled gradients. We include the result here to highlight how the introduction of heterogeneous delays affect our convergence guarantees, and consider it as an intermediate step towards the more practical architectures studied in the next section. Note that compressed IAG (3.7) with $\tau_k^i = 0$ is Algorithm (3.6). We now state the result:

Theorem 3.4. *Consider Problem* (3.1)*, and the iterates $\{x_k\}_{k\in\mathbb{N}}$ generated by* (3.7) *with $\tau_k^i \leq \tau$ and $0 < \gamma < \bar{\gamma}$ for positive scalars τ, $\bar{\gamma}$. Let $\bar{L} = L\sqrt{n(1+\Delta)}$ and $\Delta = \min(\Delta_{\text{ave}}, \Delta_{\max})$.*

1. ***Strongly-convex:*** *If Assumption 3.1, 3.4 and 3.2 hold, and*

$$\bar{\gamma} = \min\left(\frac{\mu}{\sqrt{\alpha}\tau\bar{L}^2}, \frac{1}{\alpha\bar{L}}\right),$$

then

$$\mathbf{E}[F(x_k) - F(x^\star)] \leq \left(1 - \mu\gamma + \frac{\bar{L}^4\gamma^3\tau^2\alpha}{\mu}\right)^{k/(1+2\tau)} \big(F(x_0) - F(x^\star)\big).$$

2. ***Non-convex:*** *If Assumption 3.1 and 3.4 hold, and*

$$\bar{\gamma} = \frac{1}{\sqrt{1 + 8\left(1 + \beta(1+\theta)\right)\tau(\tau+1)}} \frac{2}{\bar{L}},$$

and $\beta < 1/\left(2(1+1/\theta)\right)$ *for* $\theta > 0$*, then*

$$\min_{l\in[0,k]} \mathbf{E}\|\nabla F(x_l)\|^2 \leq \frac{1}{a}\frac{1}{k+1}\left(F(x_0) - F(x^\star)\right),$$

where $a = \gamma/2 - \gamma\beta(1+1/\theta)$.

Proof. See Appendix 3.C.4. □

From Theorem 3.4, the admissible step-size and corresponding convergence speed all depend on the delay bound τ and the compression bound α. For strongly convex problems, the upper bound on the step-size in Theorem 3.4-1 is smaller than the corresponding result in Theorem 3.3-1. If the quantizer produces the exact output ($Q(v) = v$), then the compressed IAG algorithm (3.7) coincides with the IAG algorithm (3.3) for strongly convex optimization. If we let $\alpha = 1$, $\mu/\bar{L} \leq \tau$, and $\gamma = 0.5\bar{\gamma}$, then the IAG iteration (3.3) from Theorem 3.4-1 satisfies

$$F(x_k) - F(x^\star) \leq \left(1 - \frac{1}{8}\frac{1}{1+2\tau}\frac{\mu^2}{\tau\bar{L}^2}\right)^k \left(F(x_0) - F(x^\star)\right)$$

where the inequality follows from the fact that $(1-x)^a \leq 1-ax$ for $x, a \in [0,1]$. Thus, our step-size is more than three times larger than the one derived in [27], which results in corresponding improvements in convergence factors.

From Theorem 3.4 we can next characterize the associated ε-convergence times and expected information exchange from the worker to the master for running the compressed IAG algorithm (3.7).

Corollary 3.4. *Consider Problem* (3.1)*, and the iterates* $\{x_k\}_{k\in\mathbb{N}}$ *generated by* (3.7) *with* $\tau_k^i \leq \tau$ *and* $0 < \gamma < \bar{\gamma}$ *for positive scalars* τ, $\bar{\gamma}$*. Let* $\bar{L} = L\sqrt{n(1+\Delta)}$, $\Delta = \min(\Delta_{\text{ave}}, \Delta_{\max})$*, B be the number of bits required to encode a single vector entry, and* $\mathbf{E}\{\|Q(v)\|_0\} \leq c$.

1. ***Strongly-convex:*** *If Assumption 3.1, 3.4 and 3.2 hold, and*

$$\bar{\gamma} = \min\left(\frac{\mu}{\sqrt{\alpha}\tau\bar{L}^2}, \frac{1}{\alpha\bar{L}}\right),$$

then the algorithm (3.7) *reaches* $\mathbf{E}[F(x_T) - F(x^\star)] \leq \epsilon$ *within*

$$T^\star = (1+2\tau)\frac{\mu}{\gamma\left(\mu^2 - \bar{L}^4\gamma^2\tau^2\alpha\right)}\log\left(\frac{F(x_0) - F(x^\star)}{\varepsilon}\right)$$

iterations, under which

$$B^\star = \left\lceil (\log_2 d + B)c \cdot (1+2\tau)\frac{\mu}{\gamma\left(\mu^2 - \bar{L}^4\gamma^2\tau^2\alpha\right)}\log\left(\frac{F(x_0) - F(x^\star)}{\varepsilon}\right)\right\rceil$$

bits are sent.

2. ***Non-convex:*** *If Assumption 3.1 and 3.4 hold, and*

$$\bar{\gamma} = \frac{1}{\sqrt{1 + 8\,(1 + \beta(1+\theta))\,\tau(\tau+1)}} \frac{2}{\bar{L}},$$

and $\beta < 1/\left(2(1+1/\theta)\right)$ *for* $\theta > 0$, *then the algorithm* (3.7) *reaches* $\min_{l \in [0,T]} \mathbf{E}\|\nabla F(x_l)\|^2 \leq \epsilon$ *within*

$$T^\star = \frac{1}{\gamma/2 - \gamma\beta(1+1/\theta)} \frac{F(x_0) - F(x^\star)}{\epsilon}$$

iterations, under which

$$B^\star = \left\lceil (\log_2 d + B)c \cdot \frac{1}{\gamma/2 - \gamma\beta(1+1/\theta)} \frac{F(x_0) - F(x^\star)}{\epsilon} \right\rceil$$

bits are sent.

Proof. See Appendix 3.C.5. □

3.4.2 Partial Gradient Communication

To solve scalable real-world problems effectively, it is natural to apply compression strategies on gradient descent under the partial gradient communication architecture. Even though convergence rate results of algorithms using the BREQ can be obtained, they often require restrictive conditions on problems. For instance, the vast majority of existing works often makes the uniformly bounded assumption on the norm of the objective function gradient. Although this assumption is valid for a certain class of problems, it is always violated for strongly-convex problems.

In this section, we rather study the effect of the URQ on distributed gradient descent algorithms. The unbiased property of the URQ, unlike the BREQ, allows us to establish theoretical guarantees without restrictive assumptions. Analogous to Section 3.4.1, the convergence rate is shown to depend explicitly on the URQ accuracy and the communication delay due to asynchrony.

Before studying these effect on the asynchronous algorithm, we consider its synchronous version called D-QGD. In each iteration of this algorithm, the master waits for the compressed gradient sent by every worker and maintains the iterates $\{x_k\}$ according to

$$x_{k+1} = x_k - \gamma \sum_{i=1}^{n} Q(\nabla F^i(x_k)). \tag{3.8}$$

Since URQs are random and modify the gradient vectors and their support, the sparsity patterns of the quantized gradients are time-varying and can be

characterized by the quantities

$$\begin{aligned} \Delta_{\max}^k &= \max_{i\in[1,n]} \left\{ \sum_{j=1,j\neq i}^{n} \mathbf{1}\left\{\operatorname{supp}(Q(a^i)) \cap \operatorname{supp}(Q(a^j)) \neq \emptyset\right\} \right\} \\ \Delta_{\text{ave}}^k &= \frac{1}{n}\sum_{i=1}^{n} \left\{ \sum_{j=1,j\neq i}^{n} \mathbf{1}\{\operatorname{supp}(Q(a^i)) \cap \operatorname{supp}(Q(a^j)) \neq \emptyset\} \right\}. \end{aligned} \tag{3.9}$$

A limitation with these quantities is that they cannot be computed off-line. However, since compression reduces the support of vectors, i.e. $\operatorname{supp}(Q(a^i)) \subset \operatorname{supp}(a^i)$, it always holds that $\Delta_{\max}^k \leq \Delta_{\max}$ and $\Delta_{\text{ave}}^k \leq \Delta_{\text{ave}}$.

The next lemma enables us to benefit from sparsity in our analysis.

Lemma 3.5. *Under Assumption 3.4, for $k \geq 0$*

$$\left\| \sum_{i=1}^{n} Q(\nabla F^i(x_k)) \right\|^2 \leq \sigma_k \sum_{i=1}^{n} \left\| Q(\nabla F^i(x_k)) \right\|^2,$$

where

$$\sigma_k = \min\left(\sqrt{n(1+\Delta_{\text{ave}}^k)}, 1+\Delta_{\max}^k \right).$$

Moreover,

$$\sigma_k \leq \sigma = \min\left(\sqrt{n(1+\Delta_{\text{ave}})}, 1+\Delta_{\max} \right).$$

Proof. See Appendix 3.C.6. □

Notice that Lemma 3.5 quantifies the combined impact of data sparsity and compression. We have $\sigma_k = 1$ if the quantized gradients are completely sparse (their support sets do not overlap), whereas $\sigma_k = n$ if the quantized gradients are completely dense (all support sets overlap).

Now, we state the convergence rate result.

Theorem 3.5. *Consider the optimization problem* (3.1)*, and iterates $\{x_k\}_{k\in\mathbb{N}}$ generated by* (3.8) *with $\gamma = 1/\left(L\alpha(1+\theta)\sigma\right)$ for a positive scalar θ. Let $\sigma = \min\left(\sqrt{n(1+\Delta_{\text{ave}})}, 1+\Delta_{\max}\right)$.*

1. ***Strongly-convex:*** *If Assumption 3.1, 3.2 and 3.4 hold, then*

$$\mathbf{E}\|x_k - x^\star\|^2 \leq (1-\mu\gamma)^k \|x_0 - x^\star\|^2 + \frac{1}{\mu\theta L}\sum_{i=1}^{n} \|\nabla F^i(x^\star)\|^2.$$

2. ***Convex:*** *If Assumption 3.1 and 3.4 hold, then*

$$\mathbf{E}(F(\bar{x}_k) - F(x^\star)) \leq \frac{1}{\gamma}\frac{1}{k}\|x_0 - x^\star\|^2 + \frac{1}{\theta L}\sum_{i=1}^{n}\|\nabla F^i(x^\star)\|^2,$$

where $\bar{x}_k = (1/k)\sum_{l=0}^{k-1} x_l$.

Proof. See Appendix 3.C.7. □

Theorem 3.5 establishes a convergence of D-QGD towards the optimum with a residual error depending on th step-size parameter θ, problem parameters μ, L, and the Euclidean norm of a component gradient at the optimum $\|\nabla F^i(x^\star)\|$. The step-size and convergence rate of D-QGD are impacted by the sparsity measure σ and the compression accuracy parameter α. In particular, a larger value of θ allows for the larger step-size and better convergence factor, at the cost of a larger residual error.

For simplicity of notation and applicability of the results, we formulated Theorem 3.5 in terms of σ, not σ_k (the proof, however, can also provide convergence guarantees in terms of σ_k). The result is conservative in the sense that compression increases sparsity of the gradients, which should translate into larger step-sizes. To evaluate the degree of conservatism, we carry out Monte Carlo simulations on the data sets described in Table 3.3. We indeed note that σ_k is significantly smaller than σ, as shown in Table 3.2.

Unlike many finite-time convergence results for distributed SGD, Theorem 3.5 do not require bounded gradients and can therefore handle strongly-convex problems such as ℓ_2-regularized losses. In addition, general results such as [22, Theorem 3.4] can lead to extremely small step-sizes and associated slow convergence. To verify this, we generated a random least-squares problem with dimension 1000×100 and used the ternary quantizer with $s = 1$ (i.e. $\beta = 10$). We set the number of iterations to $T = 2000$ and let the number of workers (mini-batch groups) equal to 10. We then generate random data matrices for two scenarios: one dense ($\sigma = n$) and one block diagonal ($\sigma = 1$). Our analysis suggest a step-size of 1000 times larger than the step-size allowed by [22, Theorem 3.4] in the sparse data examples, and a factor 100 larger in the dense data examples. Figure 3.2 suggests that the convergence using the step-size in Theorem 3.5-2 gives dramatic improvements in convergence over [22, Theorem 3.4].

Q-IAG Method

IAG is the popular first-order algorithm implemented on the partial gradient communication architecture. For heterogeneous and communication-limited environment, we study the compressed IAG algorithm which reduces communication bandwidth by both asynchronous regimes and compression operators.

Data Set	σ/n	$\mathbf{E}\{\sigma_k\}/m$		
		GS	TQ	LP
RCV1-train	0.83	0.66	0.07	0.42
real-sim	0.8278	0.58	0.06	0.37
GenDense	1	1	0.7	1

Table 3.2: Empirical evaluations of σ_k and σ when we use gradient sparsifier (GS) with $p^j = 0.5$ for $j \in [1, d]$, ternary quantizer (TQ), and low-precision quantizer (LP) with $s = 4$.

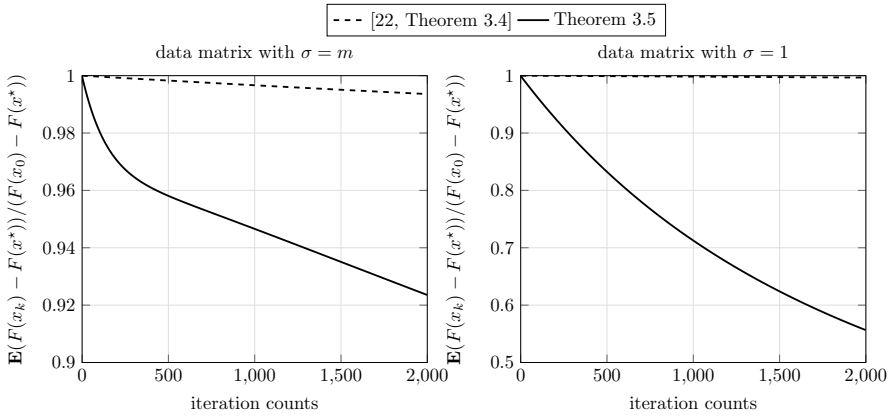


Figure 3.2: Convergence of distributed quantized gradient methods (3.8) using the ternary quantizer with $s = 1$ for least-squares problems over the randomly generated data sets with dimension 1000×100. The initial point $x_0 = x^\star + 10\eta$ where $x^\star$ is the optimum and η is a Gaussian noise with zero mean and unit variance.

In each iteration of this algorithm, called Q-IAG, the master node receives compressed gradients from some worker nodes, while reusing the staled gradients from the rest. Then, Q-IAG updates the iterate x_k according to

$$x_{k+1} = x_k - \gamma \sum_{i=1}^{n} Q\left(\nabla F^i(x_{k-\tau_k^i})\right), \tag{3.10}$$

where γ is the constant step size, and $Q(\cdot)$ is the URQ.

By Assumption 3.4, $\operatorname{supp}(Q(\nabla F^i(x_{k-\tau_k^i}))) = \operatorname{supp}(Q(a^i))$, and thus the sparsity measures defined (3.9) will be used to strengthen our analysis. Now, we present the result which are analogous to Theorem 3.5.

Theorem 3.6. *Consider the optimization problem* (3.1)*, and the iterates* $\{x_k\}_{k\in\mathbb{N}}$ *generated by* (3.10) *with* $\tau_k^i \leq \tau$ *for all* i, k *and* $0 < \gamma < \bar{\gamma}$*. Let* $\bar{L} = L\sqrt{n(1+\Delta)}$*,* $\sigma = \min\left(\sqrt{n(1+\Delta_{\text{ave}})}, 1+\Delta_{\max}\right)$*, and* $\Delta = \min(\Delta_{\text{ave}}, \Delta_{\max})$*.*

1. ***Strongly-convex:*** *If Assumption 3.1, 3.4 and 3.2 hold, and*

$$\bar{\gamma} = \frac{2\mu}{1 + n\sigma\alpha L^2\left(2\bar{L}^2\tau^2 + (1+\theta)\right)}$$

where $\theta > 0$*, then*

$$\mathbf{E}\|x_k - x^\star\|^2 \leq (p+q)^{k/(1+2\tau)}\|x_0 - x^\star\|^2 + e/(1-p-q),$$

where

$$\begin{aligned} p &= 1 - 2\mu\gamma + \gamma^2 \\ q &= 2n\sigma\alpha L^2\gamma^2\bar{L}^2\tau^2 + (1+\theta)\gamma^2 n\alpha\sigma L^2 \\ e &= \left(2n\alpha\gamma^2\bar{L}^2\tau^2 + (1+1/\theta)\gamma^2\sigma\alpha\right)\sum_{i=1}^{n}\left\|\nabla F^i(x^\star)\right\|^2. \end{aligned}$$

2. ***Non-convex with bounded gradients:*** *If Assumption 3.1, 3.4 and 3.3 hold, and*

$$\bar{\gamma} = \frac{1}{1+\sqrt{1+8\tau(\tau+1)}}\frac{2}{\bar{L}},$$

then

$$\min_{l\in[0,k]}\mathbf{E}\|\nabla F(x_l)\|^2 \leq \frac{2}{\gamma}\frac{1}{k+1}\left(F(x_0) - F(x^\star)\right) + 2\beta\sigma nC^2.$$

Proof. See Appendix 3.C.8. □

Unlike the result for the compressed IAG algorithm (3.7), Theorem 3.6 can only guarantee that Q-IAG (3.10) converges towards the optimum with the residual error depending on problem parameters μ, L, C, n, the sparsity pattern σ, the tuning parameters γ, θ, the precision of compression α, and the Euclidean norm of a component gradient at the optimum $\|\nabla F^i(x^\star)\|$.

In addition, setting $\theta = 1$ and $\gamma = 0.5\bar{\gamma}$ in Theorem 3.6-1 results in the following convergence rate of Q-IAG for strongly-convex problems

$$\mathbf{E}\|x_k - x^\star\|^2 \leq \left(1 - \frac{\mu^2}{1 + 2n\sigma\alpha L^2\left(\bar{L}^2\tau^2 + 1\right)}\right)^{k/(1+2\tau)}\|x_0 - x^\star\|^2 + E,$$

where

$$E = 2\mu \frac{n\alpha\bar{L}^2\tau^2 + \sigma\alpha}{1 + 2n\sigma\alpha L^2(\bar{L}^2\tau^2 + 1)} \sum_{i=1}^{n} \|\nabla F^i(x^\star)\|^2.$$

Thus, the convergence rate and step-size for (3.10) depend on the delay bound τ and the URQ accuracy parameter α. In particular, the convergence factor is penalized roughly by $\mu^2/(\alpha\bar{L}^4\tau^2)$ when individual workers compress their gradient information. In the absence of the worker asynchrony ($\tau = 0$), the upper bound on the step-size becomes $\mu/(n\sigma\alpha L^2)$, which is smaller than the step-size allowed by Theorem 3.5-1 with $\theta = 1$.

3.4.3 Simulation Results

We consider the empirical risk minimization problem (3.1) with component loss functions on the form of

$$F^i(x) = \frac{1}{2\rho}\|A_i x - b_i\|^2 + \frac{\sigma}{2}\|x\|^2,$$

where $A_i \in \mathbb{R}^{p\times d}$ and $b_i \in \mathbb{R}^p$. Data samples $(a^1, b^1), \ldots, (a^m, b^m)$ were split among n workers. Hence, $m = np$. The experiments were done using both synthetic and real-world data sets as shown in Table 3.3. Each data sample a^i is then normalized by its own Euclidean norm. We evaluated the performance of the distributed gradient algorithms (3.6)-(3.10) using the stochastic sparsification, the stochastic sparsification and quantization, and the stochastic ternary quantization. We set $n = 3$, $x_0 = \mathbf{1}$, set $\sigma = 1$, and set ρ equal to the total number of data samples according to Table 3.3. In addition, GenDense from Table 3.3 generated the dense data set such that each element of the data matrices A_i is randomly drawn from a uniform random number between 0 and 1, and each element of the class label vectors b_i is the sign of a zero-mean Gaussian random number with unit variance. For the stochastic sparsification, we assumed that vector elements are represented by 64 bits (IEEE doubles) while the low-precision quantizer only requires $1 + \log_2(s)$ bits to encode each vector entry. For the distributed algorithms, we have used $\tau = n$.

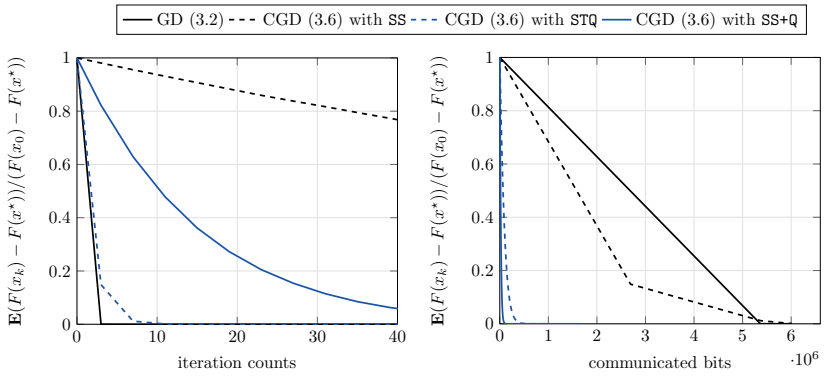
Data Set	Type	Samples	Dimension
RCV1-train	sparse	23149	47236
real-sim	sparse	72309	20958
covtype	dense	581012	54
GenDense	dense	40000	1000

Table 3.3: Summary of synthetic and benchmark data used in our experiments.

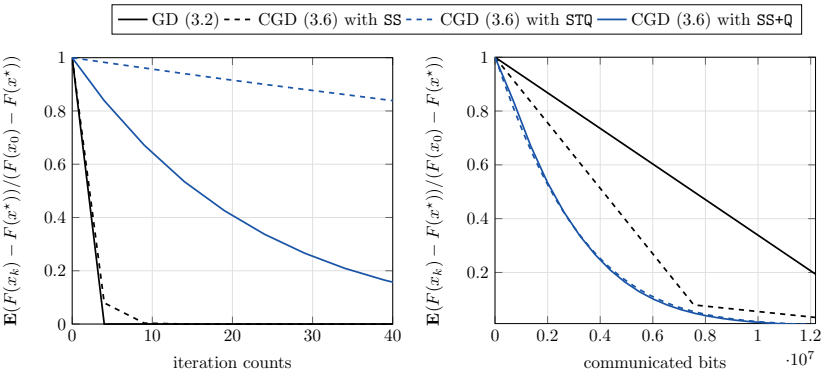
Figure 3.3 shows the trade-off between the convergence in terms of iteration counts and the number of communicated bits. Gradient descent with the

URQs (3.6) has the slower convergence rate with respect to iteration counts than full-precision gradient descent. The situation is reversed if we judge the convergence relative to the number of communicated bits. In this case, the 9-bit stochastic sparsification and quantization ($s = 10$) makes the fastest progress per information bit, followed by the stochastic ternary quantization. To reach the sub-optimality at $\varepsilon = 0.2$ on the real-sim and covtype data set, the full gradient descent requires more bits in the order of magnitude than the stochastic sparsification and quantization.

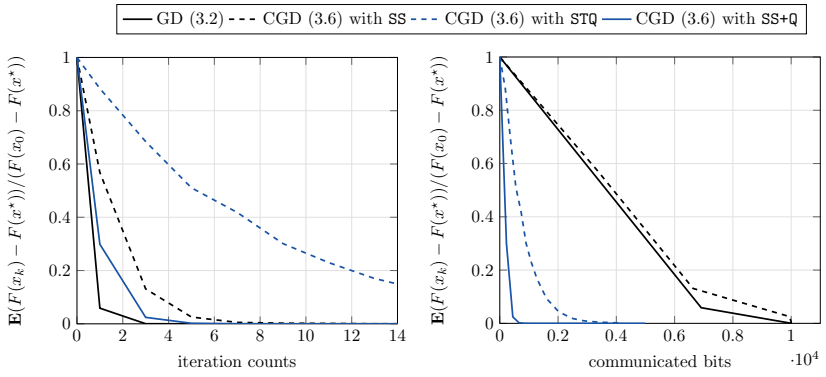
The corresponding results for the compressed IAG algorithm according to Equation (3.10) in the asynchronous parameter server setting are shown in Figure 3.4. The results are qualitatively similar: sending the gradient vectors in higher precision yields the fastest convergence but can be extremely wasteful in terms of communication load. The stochastic sparsification and quantization allows us to make a gentle trade-off between the two objectives, having both a rapid and communication-efficient convergence. In particular, the results from covtype show that a fast convergence in terms of both iteration counts and communications load for the stochastic sparsification and quantization with the higher number of quantization levels.



(a) real-sim

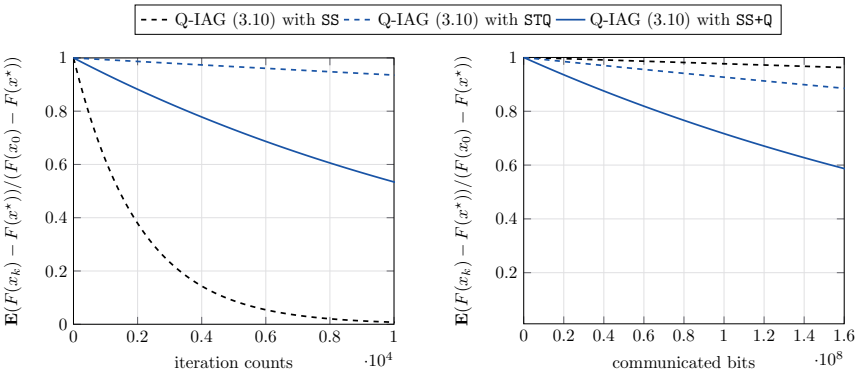


(b) RCV1-train

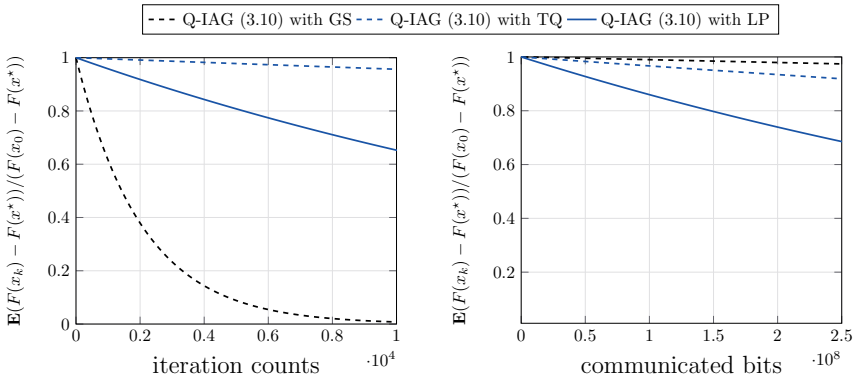


(c) covtype

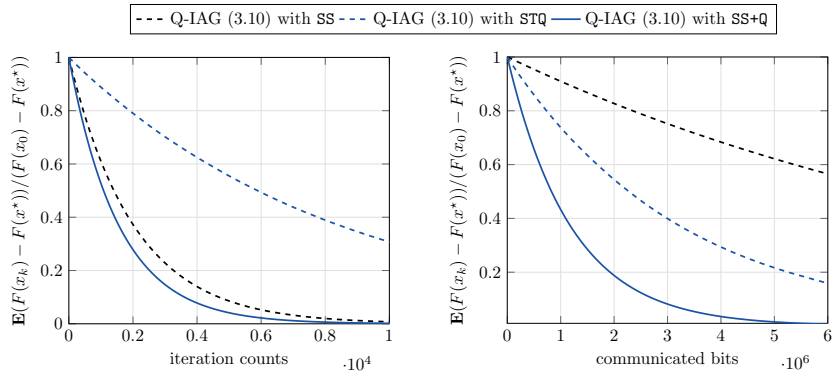
Figure 3.3: Convergence of compressed gradient descent algorithms (3.6) against iteration counts and communicated bits using different URQs over benchmark data. Here, SS, STQ and SS+Q are the stochastic sparsification with $p^i = 0.5$ for $i \in [1, d]$, the stochastic ternary quantization and the stochastic sparsification and quantization with $s = 10$.



(a) real-sim



(b) RCV1-train



(c) covtype

Figure 3.4: Convergence of Q-IAG (3.10) against iteration counts and communicated bits using different URQs over benchmark data. Here, SS, STQ and SS+Q are the stochastic sparsification with $p^i = 0.5$ for $i \in [1, d]$, the stochastic ternary quantization and the stochastic sparsification and quantization with $s = 10$.

Appendix

3.A Derivation of BREQ parameters

Deterministic sparsification

Clearly,

$$\|Q_G^K(v)\|^2 = \sum_{i \in I(v)} |v^i|^2 \leq \|v\|_2^2$$

so $\beta = 1$ is a valid estimate. To estimate α, we write $Q_G^K(v) = \sum_{i \in I(v)} e^i v^i$, where v^i is the i^{th} element of $v \in \mathbb{R}^d$; $I(v)$ collects K indices corresponding elements of v with largest absolute value; and $e^i \in \{0,1\}^d$ with only 1 at component $i \in I(v)$ and zeros elsewhere. Then

$$\langle v, Q_G^K(v) \rangle = \langle v, \sum_{i \in I(g)} e^i v^i \rangle = \sum_{i \in I(v)} |v^i|^2 = \|Q_G^K(v)\|^2.$$

Introducing $I^c(v)$ as the complement of the set $I(v)$, we note that for every $j \in I^c(v)$,

$$\left|v^j\right|^2 \leq \min_{i \in I(v)} |v^i|^2 \leq \frac{1}{K} \sum_{i \in I(v)} |v^i|^2.$$

As $|I(v)| = K$ and $|I^c(v)| = d - K$

$$\|v\|^2 \;\; = \sum_{i \in I(v)} |v^i|^2 + \sum_{j \in I^c(v)} |v^j|^2 \leq (1 + \tfrac{d-K}{K}) \sum_{i \in I(v)} |v^i|^2 = \tfrac{d}{K} \|Q_G^K(v)\|^2,$$

which implies the inequality (3.4) and that

$$\langle v, Q_G^K(v) \rangle \geq (K/d) \|v\|^2$$

Hence, $\alpha = K/d$ is a valid estimate.

Deterministic quantization

Since $\|v\|_1 = \sum_{i=1}^d v^i \operatorname{sgn}(v^i)$ and $\|v\|_1 \geq \|v\|$

$$\langle v, Q_T(v) \rangle = \|v\| \|v\|_1 \geq \|v\|^2$$

so $\alpha = 1$ is valid. Next, $\|Q_T(v)\|^2 = |\operatorname{supp}(Q_T(v))| \cdot \|v\|^2 \leq d\|v\|^2$, confirms the bounds for β.

Deterministic sparsification and quantization

By Definition 3.4, we have

$$\langle v, Q_D(v)\rangle = \langle v, \|v\| \sum_{i\in I(v)} \operatorname{sgn}(v^i)e^i\rangle = \|v\| \sum_{i\in I(v)} v^i \operatorname{sgn}(v^i) = \|v\| \sum_{i\in I(v)} |v^i|$$

By the construction of $I(v)$, we thus conclude that

$$\langle v, Q_D(v)\rangle \geq \|v\|^2.$$

In addition, $\|Q_D(v)\|^2 = |I(v)| \cdot \|v\|^2$. From [22, Lemma F.1], we know that $|I(v)| \leq \sqrt{d}$. This confirms the proposed bounds on α and β.

3.B Proofs of main results for BREQ

3.B.1 Lemma 3.6

This lemma establishes our unified theoretical results of compressed gradient descent with the BREQ compression and the fixed step-size.

Lemma 3.6. *Consider the optimization problem* (3.1)*, where the whole objective function $F(\cdot)$ is L-smooth. Then, the iterates $\{x_k\}_{k\in\mathbb{N}}$ generated by* (3.5) *with the BREQ according to Definition 3.1 satisfy*

$$F(x_{k+1}) \leq F(x_k) - \gamma\left(\alpha - \frac{L\beta\gamma}{2}\right)\|\nabla F(x_k)\|^2.$$

Proof. From the definition of the Lipschitz continuity of $\nabla F(\cdot)$ and (3.5), we have:

$$F(x_{k+1}) \leq F(x_k) - \gamma\langle \nabla F(x_k), Q(\nabla F(x_k))\rangle + \frac{L\gamma^2}{2}\|Q(\nabla F(x_k))\|^2.$$

Applying two inequalities of BREQ from Definition 3.1 into the main result completes the proof. □

3.B.2 Proof of Theorem 3.1

By using Lemma 3.6, we can prove the results for Theorem 3.1.

Proof of Theorem 3.1-a)

By the strong convexity assumption of $F(\cdot)$,

$$\|\nabla F(x_k)\|^2 \geq 2\mu\left(F(x_k) - F(x^\star)\right).$$

Applying this inequality into the one in Lemma 3.6 and using $\gamma = \alpha/(\beta L)$ yields

$$F(x_{k+1}) - F(x^\star) \leq \rho \left(F(x_k) - F(x^\star)\right), \tag{3.11}$$

where $\rho = 1 - \Gamma/\kappa$, $\Gamma = \alpha^2/(2\beta)$ and $\kappa = L/\mu$. Suppose that $\rho \in (0, 1)$. Then, by the recursion of the inequality, we obtain the result.

Proof of Theorem 3.1-b)

We start by assuming that there exists a finite positive constant R such that $\|x_k - x^\star\| \leq R$ where $\{x_k\}$ is generated by (3.5). This assumption is commonly stated; see e.g., [73]. By the convexity of the objective function $F(\cdot)$, we have:

$$F(x_k) - F(x^\star) \leq \langle \nabla F(x_k), x_k - x^\star \rangle.$$

By Cauchy-Schwarz's inequality,

$$F(x_k) - F(x^\star) \leq \|\nabla F(x_k)\| \|x_k - x^\star\| \leq R \|\nabla F(x_k)\|. \tag{3.12}$$

Denote $V_k = F(x_k) - F(x^\star)$. Plugging this inequality into the one in Lemma 3.6 and using $\gamma = \alpha/(\beta L)$, we get

$$V_{k+1} \leq V_k - \Omega V_k^2, \tag{3.13}$$

where $\Omega = \alpha^2/(2\beta L)$. Using this inequality, we have

$$\frac{1}{V_{k+1}} - \frac{1}{V_k} \geq \Omega \frac{V_k}{V_{k+1}} \geq \Omega,$$

where we reach the last inequality by the fact that $V_{k+1} \leq V_k$ from Lemma 3.6 with $\gamma = \alpha/(\beta L)$. By the recursion,

$$\frac{1}{V_T} \geq \frac{1}{V_0} + T\Omega.$$

Since $V_0 \geq 0$, the proof is complete.

Proof of Theorem 3.1-c)

Summing the inequality in Lemma 3.6 with $\gamma = \alpha/(\beta L)$ over $k = 0, 1, \ldots, T-1$ yields

$$\sum_{k=0}^{T-1} \|\nabla F(x_k)\|^2 \leq \frac{2\beta L}{\alpha^2} \left(F(x_0) - F(x_T)\right).$$

Using the fact that $\min_{k \in [0, T-1]} \|\nabla F(x_k)\|^2 \leq (1/T) \sum_{k=0}^{T-1} \|\nabla F(x_k)\|^2$, we complete the proof.

3.B.3 Proof of Corollary 3.1

Proof of Corollary 3.1-a)

Let $\Omega = \alpha^2/(2\beta L)$ and $\varepsilon_0 = F(x_0) - F(x^\star)$. From Theorem 3.1-a), the number of iterations T required for the algorithm satisfies

$$T \log(1 - \mu\Omega) \leq -\log\left(\frac{\varepsilon_0}{\varepsilon}\right)$$

to reach $F(x_T) - F(x^\star) \leq \varepsilon$. By proper manipulations and by the fact that $-1/\log(1-x) \leq 1/x$ for $0 < x \leq 1$, we obtain

$$T \leq \frac{1}{\mu\Omega} \log\left(\frac{\varepsilon_0}{\varepsilon}\right).$$

If c is the number of bits required to encode one compressed vector, then the total communication need to reach the ϵ-accuracy is $B = \lceil cT \rceil$.

Proof of Corollary 3.1-b)

Let $\Omega = \alpha^2/(2\beta L)$ and $\varepsilon_0 = R^2$ where $R \geq \|x_k - x^\star\|$ for all k. From Theorem 3.1-b), the number of iterations T required for the algorithm satisfies

$$\frac{1}{T}\frac{1}{\Omega}R^2 \leq \varepsilon$$

to reach $F(x_T) - F(x^\star) \leq \varepsilon$. By proper manipulations, we obtain

$$T \leq \frac{1}{\Omega}\frac{\varepsilon_0}{\varepsilon}.$$

If c is the number of bits required to encode one compressed vector, then the total communication need to reach the ϵ-accuracy is $B = \lceil cT \rceil$.

Proof of Corollary 3.1-c)

Let $\Omega = \alpha^2/(2\beta L)$ and $\varepsilon_0 = F(x_0) - F(x^\star)$. From Theorem 3.1-c), the number of iterations T required for the algorithm satisfies

$$\frac{1}{T}\frac{1}{\Omega}\varepsilon_0 \leq \varepsilon$$

to reach $\min_{k\in[0,T-1]} \|\nabla F(x_k)\|^2 \leq \varepsilon$. By proper manipulations, we obtain

$$T \leq \frac{1}{\Omega}\frac{\varepsilon_0}{\varepsilon}.$$

If c is the number of bits required to encode one compressed vector, then the total communication need to reach the ϵ-accuracy is $B = \lceil cT \rceil$.

3.B.4 Proof of Theorem 3.2

In this section, we provide the unified results of gradient descent with deterministic sparsification. We begin by stating the following lemma which is useful in our analysis.

Lemma 3.7. *Consider the optimization problem* (3.1)*, where the whole objective function $F(\cdot)$ is L-smooth. Suppose that $\gamma \leq 2/L$. Then, the iterates $\{x_k\}_{k\in\mathbb{N}}$ generated by* (3.5) *with deterministic sparsification satisfy*

$$F(x_{k+1}) \leq F(x_k) - \left(\gamma - \frac{L\gamma^2}{2}\right)\frac{K}{d}\|\nabla F(x_k)\|^2.$$

Proof. From the definition of the Lipschitz continuity of $\nabla F(\cdot)$ and (3.5), and by the fact $\langle v, Q_G^K(v)\rangle = \|Q_G^K(v)\|^2$,

$$F(x_{k+1}) \leq F(x_k) - \left(\gamma - \frac{L\gamma^2}{2}\right)\|Q_G^K(\nabla F(x_k))\|^2.$$

By the fact that $\|Q_G^K(\nabla F(x_k))\|^2 \geq (K/d)\|\nabla F(x_k)\|^2$ and that $\gamma \leq 2/L$, we have the result. □

Proof of Theorem 3.2-a)

Applying the strong convexity assumption (3.11) into the one in Lemma 3.7 with $\gamma = 1/L$ yields

$$F(x_{k+1}) - F(x^\star) \leq \rho\left(F(x_k) - F(x^\star)\right),$$

where $\rho = 1-(2d/K)(\mu/L)$. Since $\rho \in (0,1)$, by the recursion of the inequality, we obtain the result.

Proof of Theorem 3.2-b)

By the convex property of the whole objective function $F(\cdot)$, we have the inequality according to (3.12). Plugging this inequality into the one in Lemma 3.7 with $\gamma = 1/L$ yields the inequality (3.13) with $\Omega = K/(2dL)$. Using the same arguments as in Theorem 3.1, we complete the proof.

Proof of Theorem 3.2-c)

Summing the inequality in Lemma 3.6 with $\gamma = 1/L$ over $k = 0, 1, \ldots, T-1$ yields

$$\sum_{k=0}^{T-1}\|\nabla F(x_k)\|^2 \leq \frac{2dL}{K}\left(F(x_0) - F(x_T)\right).$$

Using the fact that $\min_{k\in[0,T-1]}\|\nabla F(x_k)\|^2 \leq (1/T)\sum_{k=0}^{T-1}\|\nabla F(x_k)\|^2$, we complete the proof.

3.B.5 Proof of Corollary 3.2

From Theorems 3.2-a), Theorems 3.2-b) and Theorems 3.2-c), we follow similar proof arguments in Corollary 3.1 to obtain the result.

3.C Proof of main results for URQ

3.C.1 Proof of Lemma 3.4

For $x, y \in \mathbb{R}^d$, $\|x+y\|^2 = \|x\|^2 + \|y\|^2 + 2\langle x, y\rangle$. Together with the fact that $F(x) = \sum_{i=1}^n F^i(x)$, this property implies that

$$
\begin{aligned}
&\|\nabla F(x) - \nabla F(y)\|^2 \\
&= \left\|\nabla F^1(x) - \nabla F^1(y)\right\|^2 + \sum_{i=2}^n \left\langle \nabla F^1(x) - \nabla F^1(y), \nabla F^i(x) - \nabla F^i(y)\right\rangle \\
&\quad + \left\|\nabla F^2(x) - \nabla F^2(y)\right\|^2 + \sum_{i=1, i\neq 2}^n \left\langle \nabla F^2(x) - \nabla F^2(y), \nabla F^i(x) - \nabla F^i(y)\right\rangle \\
&\quad + ... + \|\nabla F^n(x) - \nabla F^n(y)\|^2 + \sum_{i=1}^{n-1} \left\langle \nabla F^n(x) - \nabla F^n(y), \nabla F^i(x) - \nabla F^i(y)\right\rangle \\
&= \sum_{i=1}^n \left\|\nabla F^i(x) - \nabla F^i(y)\right\|^2 \\
&\quad + \sum_{i=1}^n \sum_{j=1, j\neq i}^n \left\langle \nabla F^i(x) - \nabla F^i(y), \nabla F^j(x) - \nabla F^j(y)\right\rangle e_{i,j},
\end{aligned}
$$

where $e_{i,j} = 1$ if $\text{supp}(\nabla F^i(x)) \cap \text{supp}(\nabla F^j(x)) \neq \emptyset$ and 0 otherwise. By Cauchy-Schwarz's inequality, we have

$$
\begin{aligned}
&\|\nabla F(x) - \nabla F(y)\|^2 \\
&\leq \sum_{i=1}^n \left\|\nabla F^i(x) - \nabla F^i(y)\right\|^2 \\
&\quad + \sum_{i=1}^n \sum_{j=1, j\neq i}^n \left\|\nabla F^i(x) - \nabla F^i(y)\right\| \cdot \left\|\nabla F^j(x) - \nabla F^j(y)\right\| e_{i,j}.
\end{aligned}
$$

The Lipschitz continuity of the gradients of component functions $F^i(\cdot)$ implies that

$$
\begin{aligned}
\|\nabla F(x) - \nabla F(y)\|^2 &\leq L^2 \left(n + \sum_{i=1}^n \sum_{j=1, j\neq i}^n e_{i,j}\right) \|x-y\|^2 \\
&= L^2 n \left(1 + \Delta_{\text{ave}}\right) \|x-y\|^2,
\end{aligned}
$$

Notice that the sparsity pattern of $\nabla F^i(\cdot)$ can be found using the data matrix A, [113].

Next, we can tighten the bound using the maximum conflict degree $\Delta_{\max}$. By Cauchy-Schwarz's inequality and by the Lipschitz gradient assumption of $F^i(\cdot)$, we have

$$\begin{aligned}\|\nabla F(x) - \nabla F(y)\|^2 &\leq L^2 \left(n + \sum_{i=1}^{n} \sum_{j=1, j\neq i}^{n} e_{i,j} \right) \|x - y\|^2 \\ &\leq L^2 n \left(1 + \Delta_{\max}\right) \|x - y\|^2,\end{aligned}$$

where the last inequality derives from the definition of the maximum conflict graph degree. In conclusion,

$$\bar{L}^2 = L^2 n \left(1 + \Delta\right),$$

where $\Delta = \min(\Delta_{\text{ave}}, \Delta_{\max})$.

3.C.2 Proof of Theorem 3.3

Proof of Theorem 3.3-1.

Using the distance between the iterates $\{x_k\}_{k\in\mathbb{N}}$ and the optimum $x^\star$, we have

$$\|x_{k+1} - x^\star\|^2 = \|x_k - x^\star\|^2 - 2\gamma\langle Q(\nabla F(x_k)), x_k - x^\star\rangle + \gamma^2 \|Q(\nabla F(x_k))\|^2.$$

Taking the expectation with respect to all the randomness in the algorithm yields

$$\begin{aligned}\mathbf{E}\|x_{k+1} - x^\star\|^2 &= \mathbf{E}\|x_k - x^\star\|^2 - 2\gamma \mathbf{E}\langle \nabla F(x_k), x_k - x^\star\rangle + \gamma^2 \mathbf{E}\|Q(\nabla F(x_k))\|^2 \\ &\leq \mathbf{E}\|x_k - x^\star\|^2 - 2\gamma \mathbf{E}\langle \nabla F(x_k), x_k - x^\star\rangle + \gamma^2 \alpha \mathbf{E}\|\nabla F(x_k)\|^2,\end{aligned}$$

where the inequality follows from the second property in Definition 3.5. Denote $V_k = \mathbf{E}\|x_k - x^\star\|^2$. It follows from [73, Theorem 2.1.12] that

$$V_{k+1} \leq \rho V_k + \left(-2\gamma \frac{1}{\mu + \bar{L}} + \gamma^2 \alpha \right) \mathbf{E}\|\nabla F(x_k)\|^2,$$

where $\rho = 1 - 2\gamma \frac{\mu \bar{L}}{\mu + \bar{L}}$. If $-2\gamma/(\mu + \bar{L}) + \gamma^2 \alpha \leq 0$, or equivalently

$$\gamma \in \left(0, \frac{2}{\alpha(\mu + \bar{L})}\right],$$

then $\rho \in [0, 1)$ for $\alpha \geq 1$, and the second term on the right-hand side of the above inequality is non-positive. Therefore, $V_{k+1} \leq \rho V_k$ implies $V_k \leq \rho^k V_0$.

Proof of Theorem 3.3-2.

Denote $V_k = \mathbf{E}\|x_k - x^\star\|^2$ and $\bar{L} = L\sqrt{n(1+\Delta)}$. Following the proof in Theorem 3.3-1., we have

$$V_{k+1} \leq V_k - 2\gamma\mathbf{E}\langle \nabla F(x_k), x_k - x^\star\rangle + \gamma^2\alpha\mathbf{E}\|\nabla F(x_k)\|^2.$$

By the property of Lipschitz continuity of $\nabla F(x)$, we have:

$$\langle \nabla F(x_k), x_k - x^\star\rangle \geq F(x_k) - F(x^\star) + \frac{1}{2\bar{L}}\|\nabla F(x_k)\|^2,$$

and by assuming that $\gamma \leq 1/(\bar{L}\alpha)$, we get

$$V_{k+1} \leq V_k - 2\gamma\mathbf{E}\left(F(x_k) - F(x^\star)\right).$$

After the manipulation, we have:

$$2\sum_{k=0}^{T}\gamma\mathbf{E}\left(F(x_k) - F(x^\star)\right) \leq V_0 - V_{T+1}. \tag{3.14}$$

Again from the Lipschitz gradient assumption of $F(\cdot)$, we have

$$F(x_{k+1}) \leq F(x_k) - \gamma\langle \nabla F(x_k), Q(\nabla F(x_k))\rangle + \frac{\bar{L}\gamma^2}{2}\|Q(\nabla F(x_k))\|^2.$$

Taking the expectation over all random variables yields

$$\mathbf{E}F(x_{k+1}) \leq \mathbf{E}F(x_k) - \left(\gamma - \frac{\bar{L}\alpha\gamma^2}{2}\right)\|\nabla F(x_k)\|^2,$$

where we reach the inequality by properties stated in Definition 3.5. Due to the fact that $\gamma \leq 1/(\bar{L}\alpha)$ and the non-negativity of the Euclidean norm, we can conclude that $\mathbf{E}F(x_{k+1}) \leq \mathbf{E}F(x_k)$. From (3.14),

$$2\gamma(T+1)\mathbf{E}\left(F(x_T) - F(x^\star)\right) \leq \mathbf{E}\|x_0 - x^\star\|^2 - \mathbf{E}\|x_{T+1} - x^\star\|^2.$$

By proper manipulations and by letting $\gamma = 1/(\bar{L}\alpha)$, we obtain the result.

3.C.3 Proof of Corollary 3.3

From Theorems 3.3-1. and 3.3-2., we follow similar proof arguments in Corollary 3.1 to obtain the upper bound of iteration complexity $T^\star$. Also note that the number of bits required to code the vector is at most $(\log_2 d + B)\,c$ bits in each iteration, where B is the number bits required to encode a single vector entry. We thus reach the upper bound of communication complexity $B^\star$.

3.C.4 Proof of Theorem 3.4

Proof of Theorem 3.4-1.

Denote $g_k = \sum_{i=1}^n \nabla F^i(x_{k-\tau_k^i})$. Let us first introduce two main lemmas which are instrumental to our main analysis.

Lemma 3.8. *Consider the iterates generated by* (3.7)*. For* $k \in \mathbb{N}_0$*,*

$$\|g_k\|^2 \leq \frac{2\bar{L}^2}{\mu} \max_{s\in[k-\tau,k]} F(x_s) - F(x^\star),$$

where $\bar{L} = L\sqrt{n(1+\Delta)}$ *and* $\Delta = \min(\Delta_{\text{ave}}, \Delta_{\max})$*.*

Proof. Since $\nabla f(x^\star) = 0$, we have

$$\|g_k\|^2 = \left\| \sum_{i=1}^n \nabla F^i(x_{k-\tau_k^i}) - \nabla F^i(x^\star) \right\|^2.$$

Following the proof of Lemma 3.4 with $x = x_{k-\tau_k^i}$ and $y = x^\star$ yields

$$\|g_k\|^2 \leq \bar{L}^2 \max_{s\in[k-\tau,k]} \|x_s - x^\star\|^2,$$

where $\bar{L} = L\sqrt{n(1+\Delta)}$ and $\Delta = \min(\Delta_{\text{ave}}, \Delta_{\max})$. The result also uses the fact that $\|x_{k-\tau_k^i} - x^\star\| \leq \max_{s\in[k-\tau,k]} \|x_s - x^\star\|$. Since $F(x) - F(x^\star) \geq (\mu/2)\|x - x^\star\|^2$, for any x, we complete the proof. □

Lemma 3.9. *The sequence* $\{x_k\}$ *generated by* (3.7) *satisfies*

$$\mathbf{E}\|\nabla F(x_k) - g_k\|^2 \leq \frac{2\gamma^2\bar{L}^4\tau^2\alpha}{\mu} \max_{s\in[k-2\tau,k]} F(x_s) - F(x^\star),$$

for $k \in \mathbb{N}_0$*, where* $\bar{L} = L\sqrt{n(1+\Delta)}$ *and* $\Delta = \min(\Delta_{\text{ave}}, \Delta_{\max})$*.*

Proof. By the definition of g_k,

$$\|\nabla F(x_k) - g_k\|^2 = \left\| \sum_{i=1}^n \nabla F^i(x_k) - \nabla F^i(x_{k-\tau_k^i}) \right\|^2.$$

Following the proof of Lemma 3.4 with $x = x_k$ and $y = x_{k-\tau_k^i}$ yields

$$\|\nabla F(x_k) - g_k\|^2 \leq \bar{L}^2 \max_{i\in[1,n]} \left\| x_k - x_{k-\tau_k^i} \right\|^2,$$

where $\bar{L} = L\sqrt{n(1+\Delta)}$ and $\Delta = \min(\Delta_{\text{ave}}, \Delta_{\max})$. We also reach the result by the fact that $\left\|x_k - x_{k-\tau_k^i}\right\| \leq \max_{i\in[1,n]} \left\|x_k - x_{k-\tau_k^i}\right\|$. Next, by the fact that $\|\sum_{i=1}^N x_i\|^2 \leq N\sum_{i=1}^N \|x_i\|^2$ for $x_i \in \mathbb{R}^d$ and $N \in \mathbb{N}$, and that $0 < \tau_i^k \leq \tau$ for all i, k

$$\|\nabla F(x_k) - g_k\|^2 \leq \bar{L}^2 \max_{i\in[1,n]} \tau_k^i \sum_{j=k-\tau_k^i}^{k-1} \|x_{j+1} - x_j\|^2 \leq \bar{L}^2\gamma^2\tau \sum_{j=k-\tau}^{k-1} \|Q(g_j)\|^2 .$$

Taking the expectation with respect to the randomness yields

$$\mathbf{E}\|\nabla F(x_k) - g_k\|^2 \leq \gamma^2\bar{L}^2\tau\alpha \sum_{j=k-\tau}^{k-1} \|g_j\|^2 .$$

It follows from Lemma 3.8 that

$$\begin{aligned}\mathbf{E}\|\nabla F(x_k) - g_k\|^2 &\leq \frac{2\gamma^2\bar{L}^4\tau\alpha}{\mu} \sum_{j=k-\tau}^{k-1} \max_{s\in[j-\tau,j]} F(x_s) - F(x^\star) \\ &\leq \frac{2\gamma^2\bar{L}^4\tau^2\alpha}{\mu} \max_{s\in[k-2\tau,k]} F(x_s) - F(x^\star).\end{aligned}$$

□

We now prove Theorem 3.4-2. Since the entire cost function $F(\cdot)$ has Lipschitz continuous gradient with constant $\bar{L}$, we have

$$F(x_{k+1}) - F(x^\star) \leq F(x_k) - F(x^\star) - \gamma\langle Q(g_k), \nabla F(x_k)\rangle + \frac{\gamma^2\bar{L}}{2}\|Q(g_k)\|^2.$$

Taking the expectation with respect to the randomness and using the second property in Definition 3.5, we obtain

$$\mathbf{E}[F(x_{k+1}) - F(x^\star)] \leq \mathbf{E}[F(x_k) - F(x^\star)] - \gamma\mathbf{E}[\langle g_k, \nabla F(x_k)\rangle] + \frac{\gamma^2\alpha\bar{L}}{2}\mathbf{E}[\|g_k\|^2].$$

If $\gamma\alpha\bar{L} \leq 1$, then $\gamma^2\alpha\bar{L} \leq \gamma$, which implies that

$$\mathbf{E}[F(x_{k+1}) - F(x^\star)] \leq \mathbf{E}[F(x_k) - F(x^\star)] - \gamma\mathbf{E}[\langle g_k, \nabla F(x_k)\rangle] + \frac{\gamma}{2}\mathbf{E}[\|g_k\|^2].$$

Using $g_k = g_k - \nabla F(x_k) + \nabla F(x_k)$ and $F(x) - F(x^\star) \leq (1/(2\mu))\|\nabla F(x)\|^2$ for any x, we have

$$\mathbf{E}[F(x_{k+1}) - F(x^\star)] \leq (1-\gamma\mu)\mathbf{E}[F(x_k) - F(x^\star)] + \frac{\gamma}{2}\mathbf{E}[\|g_k - \nabla F(x_k)\|^2],$$

It follows from Lemma 3.9 that

$$V_{k+1} \leq pV_k + q \max_{s\in[k-2\tau,k]} V_s,$$

where $V_k = \mathbf{E}[F(x_k) - F(x^\star)]$, $p = 1 - \gamma\mu$ and $q = \gamma^3\bar{L}^4\tau^2\alpha/\mu$. According to Lemma 1 of [114], if $p + q < 1$, or, equivalently, $\gamma < \mu/(\bar{L}^2\tau\sqrt{\alpha})$, then $V_k \leq (p+q)^{k/(1+2\tau)}V_0$. This completes the proof.

Proof of Theorem 3.4-2.

Define $g_k = \sum_{i=1}^n \nabla F^i(x_{k-\tau_k^i})$. Let us introduce three main lemmas which are instrumental in our main analysis.

Lemma 3.10. *The sequence* $\{x_k\}$ *generated by* (3.7) *satisfies*

$$\|\nabla F(x_k) - g_k\|^2 \leq \bar{L}^2\gamma^2\tau \sum_{j=k-\tau}^{k-1} \|Q(g_j)\|^2,$$

where $\bar{L} = L\sqrt{n(1+\Delta)}$ *and* $\Delta = \min(\Delta_{\text{ave}}, \Delta_{\max})$.

Proof. Following the proof in Lemma 3.9 yields the result. □

Lemma 3.11. *The sequence* $\{x_k\}$ *generated by* (3.7) *satisfies*

$$\mathbf{E}\|\nabla F(x_k) - Q(g_k)\|^2 \leq \alpha_1\mathbf{E}\|\nabla F(x_k) - g_k\|^2 + \alpha_2\mathbf{E}\|\nabla F(x_k)\|^2,$$

where $\alpha_1 = 2(1+\beta(1+\theta))$, $\alpha_2 = 2\beta(1+1/\theta)$, *and* $\theta > 0$.

Proof. We start by deriving the upper bound of $\mathbf{E}\|g_k - Q(g_k)\|^2$. By the property stating that $\mathbf{E}\|Q(v)-v\|^2 \leq \beta\|v\|^2$ and by the fact that $\nabla F(x^\star) = 0$, we have:

$$\mathbf{E}\|g_k - Q(g_k)\|^2 \leq \beta(1+\theta)\|g_k - \nabla F(x_k)\|^2 + \beta(1+1/\theta)\|\nabla F(x_k)\|^2,$$

where the inequality derives from the fact that $\|x+y\|^2 \leq (1+\theta)\|x\|^2 + (1+1/\theta)\|y\|^2$ for $x, y \in \mathbb{R}^d$ and $\theta > 0$.

Now, we are ready to derive the upper bound of $\mathbf{E}\|\nabla F(x_k) - Q(g_k)\|^2$. By the fact that $\|\sum_{i=1}^N x_i\|^2 \leq N\sum_{i=1}^N \|x_i\|^2$ for $x_i \in \mathbb{R}^d$ and $N \in \mathbb{N}$, we have

$$\|\nabla F(x_k) - Q(g_k)\|^2 \leq 2\|\nabla F(x_k) - g_k\|^2 + 2\|g_k - Q(g_k)\|^2.$$

Taking the expectation over the randomness and then plugging the upper bound of $\mathbf{E}\|g_k - Q(g_k)\|^2$ into the result yield the result. □

Lemma 3.12. *Suppose that non-negative sequences* $\{V_k\}, \{w_k\}$, *and* $\{\Theta_k\}$ *satisfying the following inequality*

$$V_{k+1} \leq V_k - a\Theta_k - bw_k + c\sum_{j=k-\tau}^{k} w_j, \tag{3.15}$$

where $a, b, c > 0$. *Further suppose that* $b - c(\tau+1) \geq 0$ *and* $w_k = 0$ *for* $k < 0$. *Then,*

$$\frac{1}{K+1}\sum_{k=0}^{K}\Theta_k \leq \frac{1}{a}\frac{1}{K+1}(V_0 - V_{K+1}).$$

Proof. Summing (3.15) from $k = 0$ to $k = K$ yields

$$\sum_{k=0}^{K} V_{k+1} \leq \sum_{k=0}^{K} V_k - a\sum_{k=0}^{K}\Theta_k - b\sum_{k=0}^{K} w_k + c\sum_{k=0}^{K}\sum_{j=k-\tau}^{k} w_j,$$

or equivalently due to the telescopic series

$$\begin{aligned}
a\sum_{k=0}^{K}\Theta_k &\leq (V_0 - V_{K+1}) - b\sum_{k=0}^{K} w_k + c\sum_{k=0}^{K}\sum_{j=k-\tau}^{k} w_j \\
&= (V_0 - V_{K+1}) - b\sum_{k=0}^{K} w_k \\
&\quad + c(w_{-\tau} + w_{-\tau+1} + \ldots + w_0) \\
&\quad + c(w_{-\tau+1} + w_{-\tau+2} + \ldots + w_0 + w_1) + \ldots \\
&\quad + c(w_{-\tau+K} + w_{-\tau+K+1} + \ldots + w_0 + w_1 + \ldots + w_K) \\
&\leq (V_0 - V_{K+1}) - b\sum_{k=0}^{K} w_k + c(\tau+1)\sum_{k=0}^{K} w_k \\
&\leq V_0 - V_{K+1},
\end{aligned}$$

where the second inequality comes from the fact that $w_k \geq 0$ for $k \geq 0$. In addition, the last inequality follows from the assumption that $b - c(\tau+1) \geq 0$. Then, we obtain the result. □

Now, we are ready to derive the convergence rate. From the definition of the Lipschitz continuity of the gradient of the function $F(\cdot)$ and from the fact that $2\langle x, y\rangle = \|x\|^2 + \|y\|^2 - \|x - y\|^2$ for any $x, y \in \mathbb{R}^d$, we have

$$\begin{aligned}
F(x_{k+1}) - F(x^\star) \leq F(x_k) - F(x^\star) - \frac{\gamma}{2}\|\nabla F(x_k)\|^2 - \left(\frac{\gamma}{2} - \frac{\gamma^2\bar{L}}{2}\right)\|Q(g_k)\|^2 \\
+ \frac{\gamma}{2}\|\nabla F(x_k) - Q(g_k)\|^2,
\end{aligned}$$

where $\bar{L} = L\sqrt{n(1+\Delta)}$ and $\Delta = \min(\Delta_{\text{ave}}, \Delta_{\max})$. Denote $V_k = \mathbf{E}F(x_k) - F(x^\star), \Theta_k = \mathbf{E}\|\nabla F(x_k)\|^2$, and $w_k = \mathbf{E}\|Q(g_k)\|^2$. Next, taking the expectation over the randomness, and then plugging the inequality from Lemma 3.10 and 3.11 yield

$$V_{k+1} \leq V_k - \alpha_1\Theta_k - \alpha_2 w_k + \alpha_3 \sum_{j=k-\tau}^{k-1} w_j,$$

where $\alpha_1 = \gamma/2 - \gamma\beta(1+1/\theta)$, $\alpha_2 = 0.5(\gamma - \bar{L}\gamma^2)$ and $\alpha_3 = \gamma\left(1+\beta(1+\theta)\right)\bar{L}^2\gamma^2\tau$, and $\bar{L} = L\sqrt{n(1+\Delta)}$ and $\Delta = \min(\Delta_{\text{ave}}, \Delta_{\max})$. Next, we apply Lemma 3.12. Notice that $\|Q(g_k)\| = \|x_{k+1} - x_k\|/\gamma$, which implies that $w_k = 0$ if $k < 0$. Therefore,

$$\frac{1}{K+1}\sum_{k=0}^{K}\Theta_k \leq \frac{1}{a}\frac{1}{K+1}(V_0 - V_{K+1}),$$

which means that

$$\min_{k\in[0,K]} \mathbf{E}\|\nabla F(x_k)\|^2 \leq \frac{1}{a}\frac{1}{K+1}\left(F(x_0) - F(x^\star)\right) - \frac{1}{a}\frac{1}{K+1}\left(\mathbf{E}F(x_k) - F(x^\star)\right).$$

To ensure the validity of the result, we must determine γ and β to satisfy three conditions, *i.e.* $a > 0$, $b > 0$ and $b - c(\tau+1) \geq 0$. The first criterion implies that $\beta < 1/\left(2(1+1/\theta)\right)$, and the last two criteria yield the admissible range of the step size γ. The second criterion implies that $\gamma < 1/\bar{L}$, and the equivalence of the last criterion is $0.5 - 0.5\bar{L}\gamma - \left(1+\beta(1+\theta)\right)\bar{L}^2\tau(\tau+1)\gamma^2 \geq 0$. Therefore, let $\gamma = 1/\big(\bar{L}(1+\omega)\big)$ where $\omega > 0$, and plugging the expression into the inequality yields $\omega^2 + \omega - 2\psi \geq 0$, where $\psi = \left(1+\beta(1+\theta)\right)\tau(\tau+1)$. Therefore, $\omega \geq 0.5\left(-1 + \sqrt{1+8\psi}\right)$, and the admissible step-size is $\gamma < 2/(\alpha\bar{L})$ with $\alpha = \sqrt{1 + 8\left(1+\beta(1+\theta)\right)\tau(\tau+1)}$.

3.C.5 Proof of Corollary 3.4

From Theorems 3.4-1. and 3.4-2., we follow similar proof arguments in Corollary 3.1 to obtain the upper bound of iteration complexity $T^\star$. Also note that the number of bits required to code the vector is at most $\left(\log_2 d + B\right)c$ bits in each iteration, where B is the number bits required to encode a single vector entry. We thus reach the upper bound of communication complexity $B^\star$.

3.C.6 Proof of Lemma 3.5

Denote $s_i^k = \text{ supp}(Q(a^i))$. By Assumption 3.4, the definition of the Euclidean norm and Cauchy-Schwarz's inequality,

$$\left\|\sum_{i=1}^{n} Q(\nabla F^i(x_k))\right\|^2 \leq \sum_{i=1}^{n}\|Q(\nabla F^i(x_k))\|^2 + T,$$

where $T = \sum_{i=1}^{n}\sum_{j=1,j\neq i}^{n}\|Q(\nabla F^i(x_k))\|\|Q(\nabla F^j(x_k))\|\mathbf{1}\left(s_i^k \cap s_j^k \neq \emptyset\right)$. For simplicity, let $e_{i,j} = \mathbf{1}\left(s_i^k \cap s_j^k \neq \emptyset\right)$ and $\Delta_i = \sum_{j=1,j\neq i}^{n} e_{i,j}$. Therefore, we define the maximum conflict degree $\Delta_{\max}^k = \max_{i\in[1,n]}\Delta_i$ and the average conflict degree $\Delta_{\text{ave}}^k = (\sum_{i=1}^{n}\Delta_i)/n$. Now, we bound the left-hand side by using two different data sparsity measures. First, we bound T by using the maximum conflict degree $\Delta_{\max}^k$. By the fact that $2ab \leq a^2 + b^2$ for $a, b \in \mathbb{R}$, we have

$$\begin{aligned} T &\leq \frac{1}{2}\sum_{i=1}^{n}\sum_{j=1,j\neq i}^{n}\left(\|Q(\nabla F^i(x_k))\|^2 + \|Q(\nabla F^j(x_k))\|^2\right)e_{i,j} \\ &\leq \Delta_{\max}^k\sum_{i=1}^{n}\|Q(\nabla F^i(x_k))\|^2. \end{aligned}$$

Therefore, $\|\sum_{i=1}^{n} Q(\nabla f_i(x_k))\|^2 \leq (1+\Delta_{\max}^k)\sum_{i=1}^{n}\|Q(\nabla F^i(x_k))\|^2$. Next, we bound the left-hand side by using the average conflict degree Δ_{ave}^k. By Cauchy-Schwarz's inequality, we get:

$$\begin{aligned} \left\|\sum_{i=1}^{n} Q(\nabla F^i(x_k))\right\|^2 &= \sum_{i=1}^{n}\|Q(\nabla F^i(x_k)\|\sum_{j=1}^{n}\|Q(\nabla F^j(x_k))\|e_{i,j} \\ &\leq \sum_{i=1}^{n}\|Q(\nabla F^i(x_k))\|\sqrt{\sum_{j=1}^{n}\|Q(\nabla F^j(x_k))\|^2\sum_{j=1}^{n}e_{i,j}^2} \\ &\leq \sqrt{\sum_{i=1}^{n}\|Q(\nabla F^i(x_k))\|^2}\sqrt{\sum_{j=1}^{n}\|Q(\nabla F^j(x_k))\|^2}\sqrt{\sum_{i=1}^{n}\sum_{j=1}^{n}e_{i,j}^2} \\ &\leq \sqrt{\sum_{i=1}^{n}\sum_{j=1}^{n}e_{i,j}}\sum_{i=1}^{n}\|Q(\nabla F^i(x_k))\|^2 \\ &= \sqrt{n(1+\Delta_{\text{ave}}^k)}\sum_{i=1}^{n}\|Q(\nabla F^i(x_k))\|^2. \end{aligned}$$

In conclusion,

$$\left\|\sum_{i=1}^{n} Q(\nabla F^i(x_k))\right\|^2 \leq \sigma_k\sum_{i=1}^{n}\|Q(\nabla F^i(x_k))\|^2,$$

where $\sigma_k = \min\left(\sqrt{n(1+\Delta_{\text{ave}}^k)}, 1+\Delta_{\max}^k\right)$.

3.C.7 Proof of Theorem 3.5

Proof of Theorem 3.5-1.

Since the component functions are convex and have L-Lipschitz continuous gradients,

$$\|\nabla F^i(x) - \nabla F^i(y)\|^2 \leq L\langle \nabla F^i(x) - \nabla F^i(y), x - y\rangle \qquad \forall x, y \in \mathbb{R}^d. \quad (3.16)$$

By Young's inequality and (3.16),

$$\begin{aligned}\|\nabla F^i(x_k)\|^2 \leq (1+\theta)L\langle \nabla F^i(x_k) - \nabla F^i(x^\star), x_k - x^\star\rangle \\ +(1+1/\theta)\|\nabla F^i(x^\star)\|^2.\end{aligned} \quad (3.17)$$

We use the distance between the iterates $\{x_k\}_{k\in\mathbb{N}}$ and the optimum $x^\star$ to analyze the convergence:

$$\begin{aligned}\|x_{k+1} - x^\star\|^2 \leq \|x_k - x^\star\|^2 - 2\gamma \left\langle \sum_{i=1}^n Q(\nabla F^i(x_k)), x_k - x^\star \right\rangle \\ + \gamma^2\sigma_k \sum_{i=1}^n \left\| Q(\nabla F^i(x_k)) \right\|^2,\end{aligned}$$

where the inequality comes from Lemma 3.5. Since all machines have the same quantizers with the same parameters, we have $\mathbf{E}\left\|Q(\nabla F^i(x_k))\right\|^2 \leq \alpha\mathbf{E}\|\nabla F^i(x_k)\|^2$. Therefore, taking the expectation over all random variables yields

$$\begin{aligned}\mathbf{E}\|x_{k+1} - x^\star\|^2 &\leq \mathbf{E}\|x_k - x^\star\|^2 - 2\gamma\mathbf{E}\left\langle \nabla F(x_k), x_k - x^\star\right\rangle \\ &\quad + \gamma^2\sigma_k\alpha\sum_{i=1}^n \mathbf{E}\left\|\nabla F^i(x_k)\right\|^2 \\ &\leq \mathbf{E}\|x_k - x^\star\|^2 - 2\gamma\mathbf{E}\left\langle \nabla F(x_k) - \nabla f(x^\star), x_k - x^\star\right\rangle \\ &\quad + \gamma^2\sigma\alpha L(1+\theta)\mathbf{E}\langle \nabla F(x_k) - \nabla F(x^\star), x_k - x^\star\rangle \\ &\quad + \gamma^2\sigma\alpha(1+1/\theta)\sum_{i=1}^n \mathbf{E}\|\nabla F^i(x^\star)\|^2,\end{aligned}$$

where the last inequality comes from (3.17), $\nabla F(x) = \sum_{i=1}^n \nabla F^i(x)$, $\nabla F(x^\star) = 0$, and $\sigma_k \leq \sigma$. Now, let $\gamma = 1/\left(L\alpha(1+\theta)\sigma\right)$. Then, by strong convexity of $F(\cdot)$, we have:

$$\mathbf{E}\|x_{k+1} - x^\star\|^2 \leq \rho\mathbf{E}\|x_k - x^\star\|^2 + \gamma\frac{1}{\theta L}\sum_{i=1}^n \mathbf{E}\left\|\nabla F^i(x^\star)\right\|^2,$$

where $\rho = 1 - \mu\gamma$. Since $\rho \in (0,1)$, we have $V_k \leq \rho^k V_0 + \bar{e}$ where $\bar{e} = e/(1-\rho)$ or equivalently

$$\mathbf{E}\|x_k - x^\star\|^2 \leq (1-\mu\gamma)^k \|x_0 - x^\star\|^2 + \frac{1}{\mu\theta L}\sum_{i=1}^n \left\|\nabla F^i(x^\star)\right\|^2.$$

Proof of Theorem 3.5-2.

Following the proof in Theorem 3.5-1., we reach:

$$\begin{aligned}\mathbf{E}\|x_{k+1} - x^\star\|^2 \leq \mathbf{E}\|x_k - x^\star\|^2 &- 2\gamma\mathbf{E}\left\langle \nabla F(x_k) - \nabla F(x^\star), x_k - x^\star\right\rangle \\ &+ \gamma^2\sigma\alpha L(1+\theta)\mathbf{E}\langle \nabla F(x_k) - \nabla F(x^\star), x_k - x^\star\rangle \\ &+ \gamma^2\sigma\alpha(1+1/\theta)\sum_{i=1}^n \mathbf{E}\|\nabla F^i(x^\star)\|^2.\end{aligned}$$

Now, let $\gamma = 1/\left(L\alpha(1+\theta)\sigma\right)$. Then, we have:

$$\begin{aligned}\mathbf{E}\|x_{k+1} - x^\star\|^2 &\leq \mathbf{E}\|x_k - x^\star\|^2 - \gamma\mathbf{E}\left\langle \nabla F(x_k) - \nabla F(x^\star), x_k - x^\star\right\rangle \\ &\quad + \gamma\frac{1}{\theta L}\sum_{i=1}^n \mathbf{E}\left\|\nabla F^i(x^\star)\right\|^2 \\ &\leq \mathbf{E}\|x_k - x^\star\|^2 - \gamma\mathbf{E}(F(x_k) - F(x^\star)) + \gamma\frac{1}{\theta L}\sum_{i=1}^n \mathbf{E}\left\|\nabla F^i(x^\star)\right\|^2,\end{aligned}$$

where we reach the last inequality by the convexity of $F(\cdot)$, i.e.

$$\langle \nabla F(x_k), x_k - x^\star\rangle \geq F(x_k) - F(x^\star).$$

Denote $\bar{x}_T = \sum_{k=0}^{T-1} x_k/T$. Due to the convexity of the objective function f, $F(\bar{x}_T) \leq \sum_{k=0}^{T-1} F(x_k)/T$. By the manipulation, we have

$$\mathbf{E}(F(x_T) - F(x^\star)) \leq \frac{1}{\gamma}\frac{1}{T}\|x_0 - x^\star\|^2 + \frac{1}{\theta L}\sum_{i=1}^n \mathbf{E}\|\nabla F^i(x^\star)\|^2,$$

where we reach the last inequality by the telescopic series and by the non-negativity of the Euclidean norm.

3.C.8 Proof of Theorem 3.6

Proof of Theorem 3.6-1.

Denote $g_k = \sum_{i=1}^n \nabla F^i(x_{k-\tau_k^i})$. Before deriving the convergence rate, we introduce an essential lemma for our main analysis.

Lemma 3.13. *Let* $\bar{L} = L\sqrt{n(1+\Delta)}$, *and* $\Delta = \min(\Delta_{\text{ave}}, \Delta_{\max})$. *Consider the IAG update* (3.10) *with the URQ according to Definition 3.5. Then,*

$$\mathbf{E}\left\|e_k\right\|^2 \leq 2m\sigma\alpha L^2\gamma^2\bar{L}^2\tau^2 \max_{s\in[k-2\tau,k]}\left\|x_s - x^\star\right\|^2 + 2m\alpha\gamma^2\bar{L}^2\tau^2\sum_{i=1}^n\left\|\nabla F^i(x^\star)\right\|^2$$

where $\sigma = \min\left(\sqrt{n(1+\Delta_{\text{ave}})}, 1+\Delta_{\max}\right)$, *and* $g_k = \sum_{i=1}^n \nabla F^i(x_{k-\tau_k^i})$.

Proof. Denote $g_k = \sum_{i=1}^n \nabla F^i(x_{k-\tau_k^i})$ and $e_k = \nabla F(x_k) - g_k$. Following the proof of Lemma 3.4 with $x = x_k$ and $y = x_{k-\tau_k^i}$ yields

$$\left\|e_k\right\|^2 \leq \bar{L}^2 \max_{i\in[1,n]}\left\|x_k - x_{k-\tau_k^i}\right\|^2,$$

where $\bar{L} = L\sqrt{n(1+\Delta)}$, and $\Delta = \min(\Delta_{\text{ave}}, \Delta_{\max})$. Next, by the bounded delay assumption and (3.10),

$$\left\|e_k\right\|^2 \leq \bar{L}^2 \max_{i\in[1,n]}\tau_k^i \sum_{j=k-\tau_k^i}^{k-1}\left\|x_{j+1} - x_j\right\|^2 \leq \gamma^2\bar{L}^2\tau\sum_{j=k-\tau}^{k-1}\left\|\sum_{i=1}^n Q(\nabla F^i(x_{j-\tau_j^i}))\right\|^2.$$

On the other hand, by Lemma 3.5 and the second property of Definition 3.5,

$$\mathbf{E}\left\|\sum_{i=1}^n Q(\nabla F^i(x_{j-\tau_j^i}))\right\|^2 \leq \sigma\sum_{i=1}^n \mathbf{E}\left\|Q(\nabla F^i(x_{j-\tau_j^i}))\right\|^2 \leq \sigma\alpha\sum_{i=1}^n\left\|\nabla F^i(x_{j-\tau_j^i})\right\|^2.$$

Using the inequality $\|x+y\|^2 \leq 2\|x\|^2+2\|y\|^2$ with $x = \nabla F^i(x_{j-\tau_j^i}) - \nabla F^i(x^\star)$ and $y = \nabla F^i(x^\star)$ and using the Lipschitz continuity assumption for gradient of each F^i, we have

$$\begin{aligned}\mathbf{E}\left\|\sum_{i=1}^n Q(\nabla F^i(x_{j-\tau_j^i}))\right\|^2 &\leq 2\sigma\alpha\sum_{i=1}^n L^2\left\|x_{j-\tau_j^i} - x^\star\right\|^2 + 2\sigma\alpha\sum_{i=1}^n\left\|\nabla F^i(x^\star)\right\|^2 \\ &\leq 2n\sigma\alpha L^2 \max_{s\in[j-\tau,j]}\left\|x_s - x^\star\right\|^2 + 2n\alpha\sum_{i=1}^n\left\|\nabla F^i(x^\star)\right\|^2,\end{aligned}$$

where the last inequality by the bounded delay assumption. Hence, plugging this result into the upper bound of e_k yields the result. □

We now prove Theorem 3.6-1. Using the definition of the Euclidean norm with (3.10) and taking the expectation over all the random variables yields

$$\begin{aligned}\mathbf{E}\left\|x_{k+1} - x^\star\right\|^2 = \mathbf{E}\left\|x_k - x^\star\right\|^2 - 2\gamma\mathbf{E}\left\langle\sum_{i=1}^n \nabla F^i(x_{k-\tau_k^i}), x_k - x^\star\right\rangle \\ + \gamma^2\mathbf{E}\left\|\sum_{i=1}^n Q\left(\nabla F^i(x_{k-\tau_k^i})\right)\right\|^2.\end{aligned}$$

Using the second property in Definition 3.5 and Lemma 3.5 due to Assumption 3.4, we get

$$\begin{aligned}\mathbf{E}\left\|x_{k+1}-x^\star\right\|^2 &\le \mathbf{E}\left\|x_k-x^\star\right\|^2 - 2\gamma\mathbf{E}\left\langle\nabla F(x_k), x_k-x^\star\right\rangle\\ &\quad + 2\gamma\mathbf{E}\left\langle g_k-\nabla F(x_k), x_k-x^\star\right\rangle\\ &\quad + \gamma^2\sigma\alpha\sum_{i=1}^n\mathbf{E}\left\|\nabla F^i(x_{k-\tau_k^i})\right\|^2\\ &\le \mathbf{E}\left\|x_k-x^\star\right\|^2 - 2\gamma\mathbf{E}\left\langle\nabla F(x_k), x_k-x^\star\right\rangle\\ &\quad + \mathbf{E}\left\|g_k-\nabla F(x_k)\right\|^2 + \gamma^2\mathbf{E}\left\|x_k-x^\star\right\|^2\\ &\quad + (1+\theta)\gamma^2\sigma\alpha\sum_{i=1}^n\mathbf{E}\left\|\nabla F^i(x_{k-\tau_k^i})-\nabla F^i(x^\star)\right\|^2\\ &\quad + (1+1/\theta)\gamma^2\sigma\alpha\sum_{i=1}^n\mathbf{E}\left\|\nabla F^i(x^\star)\right\|^2\end{aligned}$$

where $\sigma = \min\left(\sqrt{n(1+\Delta_{\text{ave}})}, 1+\Delta_{\max}\right)$ $g_k = \sum_{i=1}^n \nabla F^i(x_{k-\tau_k^i})$. The second inequality follows from Cauchy-Schwarz's inequality and from the fact that $\|x+y\|^2 \le (1+\theta)\|x\|^2 + (1+1/\theta)\|y\|^2$ for $x, y \in \mathbb{R}^d$ and $\theta > 0$. Due to the Lipschitz continuity assumption of $\nabla F^i(\cdot)$, we get

$$\begin{aligned}\mathbf{E}\left\|x_{k+1}-x^\star\right\|^2 &\le (1+\gamma^2)\mathbf{E}\left\|x_k-x^\star\right\|^2 - 2\gamma\mathbf{E}\left\langle\nabla F(x_k), x_k-x^\star\right\rangle\\ &\quad + \mathbf{E}\left\|g_k-\nabla F(x_k)\right\|^2\\ &\quad + (1+\theta)\gamma^2 n\alpha\sigma L^2\mathbf{E}\left\|x_{k-\tau_k^i}-x^\star\right\|^2\\ &\quad + (1+1/\theta)\gamma^2\sigma\alpha\sum_{i=1}^n\mathbf{E}\left\|\nabla F^i(x^\star)\right\|^2.\end{aligned}$$

It follows from Lemma 3.13 that

$$\begin{aligned}\mathbf{E}\left\|x_{k+1}-x^\star\right\|^2 &\le (1+\gamma^2)\mathbf{E}\left\|x_k-x^\star\right\|^2 - 2\gamma\mathbf{E}\left\langle\nabla F(x_k), x_k-x^\star\right\rangle\\ &\quad + 2n\sigma\alpha L^2\gamma^2\bar{L}^2\tau^2\max_{s\in[k-2\tau,k]}\left\|x_s-x^\star\right\|^2\\ &\quad + 2n\alpha\gamma^2\bar{L}^2\tau^2\sum_{i=1}^n\left\|\nabla F^i(x^\star)\right\|^2\\ &\quad + (1+\theta)\gamma^2 n\alpha\sigma L^2\max_{s\in[k-\tau,k]}\mathbf{E}\left\|x_s-x^\star\right\|^2\\ &\quad + (1+1/\theta)\gamma^2\sigma\alpha\sum_{i=1}^n\mathbf{E}\left\|\nabla F^i(x^\star)\right\|^2.\end{aligned}$$

Denote $V_k = \mathbf{E}\|x_k - x^\star\|^2$. Due to the fact that $\nabla F(x^\star) = 0$ and the property of the strong convexity assumption of $F(\cdot)$, it holds for $x, y \in \mathbb{R}^d$ that

$\langle \nabla f(x) - \nabla f(y), x - y \rangle \geq \mu \|x - y\|^2$, using this inequality with $x = x_k, y = x^\star$ and yields

$$V_{k+1} \leq pV_k + q \max_{s \in [k-2\tau, k]} V_s + e,$$

where

$$\begin{aligned} p &= 1 - 2\mu\gamma + \gamma^2 \\ q &= 2n\sigma\alpha L^2\gamma^2\bar{L}^2\tau^2 + (1+\theta)\gamma^2 n\alpha\sigma L^2 \\ e &= \left(2n\alpha\gamma^2\bar{L}^2\tau^2 + (1+1/\theta)\gamma^2\sigma\alpha\right) \sum_{i=1}^{n} \mathbf{E}\left\|\nabla F^i(x^\star)\right\|^2. \end{aligned}$$

From Lemma 1 of [114], $p + q < 1$ implies that

$$\gamma < \frac{2\mu}{1 + n\sigma\alpha L^2\left(2\bar{L}^2\tau^2 + (1+\theta)\right)}.$$

Then, this implies that $V_k \leq (p+q)^{k/(1+2\tau)} V_0 + e/(1-p-q)$.

Proof of Theorem 3.6-2.

Denote $\tilde{g}_k = \sum_{i=1}^n Q\left(\nabla F^i(x_{k-\tau_k^i})\right)$ and $g_k = \sum_{i=1}^n \nabla F^i(x_{k-\tau_k^i})$. Before deriving the convergence rate, we introduce the lemmas which are instrumental in our main analysis.

Lemma 3.14. *The sequence* $\{x_k\}$ *generated by* (3.10) *satisfies*

$$\|g_k - \nabla F(x_k)\|^2 \leq \bar{L}^2\gamma^2\tau \sum_{j=k-\tau}^{k-1} \|\tilde{g}_j\|^2.$$

Proof. Following the proof in Lemma 3.9 yields the result. □

Lemma 3.15. *The sequence* $\{x_k\}$ *generated by* (3.10) *under Assumption 3.4 satisfies*

$$\|g_k - \tilde{g}_k\|^2 \leq \sigma \sum_{i=1}^{n} \left\|\nabla F^i(x_{k-\tau_k^i}) - Q\left(\nabla F^i(x_{k-\tau_k^i})\right)\right\|^2,$$

where $\sigma = \min\left(\sqrt{n(1+\Delta_{\text{ave}})}, 1 + \Delta_{\max}\right)$.

Proof. The proof arguments follow those in Lemma 3.5 with replacing $Q(\nabla F^i(x_k))$ with $\nabla F^i(x_{k-\tau_k^i}) - Q\left(\nabla F^i(x_{k-\tau_k^i})\right)$. Also, note that

$$\text{supp}\left(\nabla F^i(x_{k-\tau_k^i}) - Q\left(\nabla F^i(x_{k-\tau_k^i})\right)\right) \subset \text{supp}(\nabla F^i(x_{k-\tau_k^i})),$$

and Assumption 3.4 implies that $\text{supp}(\nabla F^i(x_{k-\tau_k^i}))$ can be computed from the data directly. □

Lemma 3.16. *The sequence* $\{x_k\}$ *generated by* (3.10) *under Assumption 3.4 and 3.3 satisfies*

$$\mathbf{E}\|\tilde{g}_k - \nabla F(x_k)\|^2 \leq 2\bar{L}^2\gamma^2\tau \sum_{j=k-\tau}^{k-1} \mathbf{E}\|\tilde{g}_j\|^2 + 2\beta\sigma nC^2.$$

Proof. By the fact that $\|x+y\|^2 \leq 2\|x\|^2 + 2\|y\|^2$, we have:

$$\begin{aligned}\|\tilde{g}_k - \nabla F(x_k)\|^2 &\leq 2\|g_k - \nabla F(x_k)\|^2 + 2\|g_k - \tilde{g}_k\|^2 \\ &\leq 2\bar{L}^2\gamma^2\tau \sum_{j=k-\tau}^{k-1} \|\tilde{g}_j\|^2 + 2\|g_k - \tilde{g}_k\|^2,\end{aligned}$$

where the last inequality follows from Lemma 3.14. Next, taking the expectation of the inequality from Lemma 3.15 over the randomness yields

$$\begin{aligned}\mathbf{E}\|g_k - \tilde{g}_k\|^2 &\leq \sigma \sum_{i=1}^n \mathbf{E}\left\|\nabla F^i(x_{k-\tau_k^i}) - Q\left(\nabla F^i(x_{k-\tau_k^i})\right)\right\|^2 \\ &\leq \beta\sigma \sum_{i=1}^n \mathbf{E}\left\|\nabla F^i(x_{k-\tau_k^i})\right\|^2,\end{aligned}$$

where we reach the last inequality by the second property of the URQ, *i.e.* $\mathbf{E}\|Q(v) - v\|^2 \leq \beta\mathbf{E}\|v\|^2$. Next, taking the expectation over the randomness yields

$$\begin{aligned}\mathbf{E}\|\tilde{g}_k - \nabla f(x_k)\|^2 &\leq 2\bar{L}^2\gamma^2\tau \sum_{j=k-\tau}^{k-1} \mathbf{E}\|\tilde{g}_j\|^2 + 2\mathbf{E}\|g_k - \tilde{g}_k\|^2 \\ &\leq 2\bar{L}^2\gamma^2\tau \sum_{j=k-\tau}^{k-1} \mathbf{E}\|\tilde{g}_j\|^2 + 2\beta\sigma \sum_{i=1}^n \mathbf{E}\left\|\nabla F^i(x_{k-\tau_k^i})\right\|^2 \\ &\leq 2\bar{L}^2\gamma^2\tau \sum_{j=k-\tau}^{k-1} \mathbf{E}\|\tilde{g}_j\|^2 + 2\beta\sigma nC^2,\end{aligned}$$

where the last inequality results from Assumption 3.3.

□

Lemma 3.17. *Assume that non-negative sequences* $\{V_k\}, \{w_k\}$, *and* $\{\Theta_k\}$ *satisfying the following inequality*

$$V_{k+1} \leq V_k - a\Theta_k - bw_k + c \sum_{j=k-\tau}^{k} w_j + e, \tag{3.18}$$

where $a, b, c, e > 0$. *Further suppose that* $b - c(\tau+1) \geq 0$ *and* $w_k = 0$ *for* $k < 0$. *Then,*

$$\frac{1}{K+1}\sum_{k=0}^{K}\Theta_k \leq \frac{1}{a}\frac{1}{K+1}(V_0 - V_{K+1}) + \frac{1}{a}e.$$

Proof. Following the proof in Lemma 3.12 yields the result. □

Now, we are ready to derive the convergence rate. From the Lipschitz continuity assumption of the gradient of f and from the fact that $2\langle x, y\rangle = \|x\|^2 + \|y\|^2 - \|x-y\|^2$ for $x, y \in \mathbb{R}^d$,

$$f(x_{k+1}) - f^\star \leq F(x_k) - F(x^\star) - \frac{\gamma}{2}\|\nabla F(x_k)\|^2 - \alpha\|\tilde{g}_k\|^2 + \frac{\gamma}{2}\|\tilde{g}_k - \nabla F(x_k)\|^2,$$

where $\alpha = \gamma/2 - \bar{L}\gamma^2/2$. Taking the expectation over the randomness and using Lemma 3.16 yields

$$\mathbf{E}F(x_{k+1}) - F(x^\star) \leq \mathbf{E}F(x_k) - F(x^\star) - \frac{\gamma}{2}\mathbf{E}\|\nabla F(x_k)\|^2 - \alpha\mathbf{E}\|\tilde{g}_k\|^2 + \bar{L}^2\gamma^3\tau\sum_{j=k-\tau}^{k-1}\mathbf{E}\|\tilde{g}_j\|^2 + \gamma\beta\sigma nC^2$$

Applying Lemma (3.17) with $V_k = \mathbf{E}F(x_k) - F(x^\star)$, $\Theta_k = \mathbf{E}\|\nabla F(x_k)\|^2$, $w_k = \mathbf{E}\|\tilde{g}_k\|^2$, $e = \gamma\beta\sigma nC^2$, $a = \gamma/2$, $b = \alpha$, and $c = \bar{L}^2\tau\gamma^3$ yields the result.

$$\min_{k\in[0,K]}\mathbf{E}\|\nabla F(x_k)\|^2 \leq \frac{V_0 - V_K}{a(K+1)} + \frac{1}{a}e.$$

where $V_k = F(x_k) - F(x^\star)$. Note that $w_k = 0$ for $k < 0$ since $\mathbf{E}\|\tilde{g}_k\|^2 = \mathbf{E}\|x_{k+1} - x_k\|^2/\gamma^2$. Lastly, we need to find the admissible range of the step-size which guarantees the convergence. The following criteria must be satisfied: $b > 0$ and $b - c(\tau+1) \geq 0$. The first criterion implies that $\gamma < 1/\bar{L}$. The second criterion implies that $0.5\gamma - 0.5\bar{L}\gamma^2 - \bar{L}^2\tau(\tau+1)\gamma^3 \geq 0$. Lastly, let $\gamma = 1/(\bar{L}+\omega)$ for $\omega > 0$ and plugging the expression into the result yields $\omega^2 + \bar{L}\omega - 2\bar{L}^2\tau(\tau+1) \geq 0$, and therefore $\omega \geq \left(-1 + \sqrt{1 + 8\tau(\tau+1)}\right)\bar{L}/2$. Thus, we can conclude that the admissible range of the step-size is $\gamma < (2/L)(1/\beta)$, where $\beta = 1 + \sqrt{1 + 8\tau(\tau+1)}$.

Chapter 4

Error-compensated Gradient Methods

Large-scale and data-intensive problems in machine learning, signal processing, and control are typically solved by parallel/distributed optimization algorithms. These algorithms achieve high performance by splitting the computation load between multiple nodes that cooperatively determine the optimal solution. In the process, much of the algorithm complexity is shifted from the computation to the coordination. This means that the communication can easily become the main bottleneck of the algorithms, making it expensive to exchange full precision information especially when the decision vectors are large and dense. For example, in training state-of-the-art deep neural network models with millions of parameters such as AlexNet, ResNet and LSTM communication can account for up to 80% of overall training time, [22].

To reduce the communication overhead in large-scale optimization much recent literature has focused on algorithms that compress the communicated information. Some successful examples of such compression strategies are *sparsification*, where some elements of information are set to be zero [103, 115] and *quantization*, where information is reduced to a low-precision representation [22, 82]. Algorithms that compress information in this manner have been extensively analyzed under both centralized and decentralized architectures, [22, 115, 82, 116, 117, 118, 119, 120, 121, 122]. These algorithms are theoretically shown to converge to approximate optimal solutions with an accuracy that is limited by the compression precision. Even though compression schemes reduce the number of communicated bits in practice, they often lead to significant performance degradation in terms of both solution accuracy and convergence times, [29, 28].

To mitigate these negative effects of information compression on optimization algorithms, serveral error compensation strategies have been proposed [28, 123, 124, 125]. In essence, error compensation corrects for the accumulation of many consecutive compression errors by keeping a memory of previous errors. Even though very coarse compressors are used, optimization algorithms using error compensation often display the same practical performance as algorithms using full-precision information, [28, 123].

Motivated by these encouraging experimental observations, we provide novel theoretical insights in error compensation on gradient descent with compression schemes. Our key results quantify accuracy gains of error compensation on strongly convex quadratic problems. The improvements in solution

accuracy are particularly significant on ill-conditioned problems. Furthermore, we provide strong theoretical guarantees of stochastic gradient methods using both Hessian-free and Hessian-aided error compensation on the network with multiple nodes. In essence, we show that both compensation schemes with proper tuning parameters can achieve arbitrarily high solution accuracy. We also prove that the Hessian-aided error compensation, unlike the Hessian-free compensation, avoids accumulation of compression errors. Numerical experiments confirm the superior performance of Hessian-aided error compensation over other schemes. In addition, the experiments indicate that error compensation with a diagonal Hessian approximation achieves similar performance improvements as using the full Hessian.

4.1 Motivation and Prelimiary Results

In this section, we motivate our study of error-compensated gradient methods. We give an overview of distributed optimization algorithms based on communicating gradient information in § 4.1.1 and describe a general form of gradient compressors, covering most existing ones, in § 4.1.2. Later in § 4.2 we illustrate the limits of directly compressing the gradient, motivating the need for the error-compensated gradient methods studied in this paper.

4.1.1 Distributed Optimization

Distributed optimization algorithms have enabled us to solve large-scale and data-intensive problems in a wide range of application areas such as smart grids, wireless networks, and statistical learning. Many distributed optimization algorithms build on gradient methods and can be categorized based on whether they use a) full gradient communication or b) partial gradient communication (see detailed discussions in § 2.6). The full gradient algorithms solve problems on the form

$$\underset{x}{\text{minimize}} \quad F(x), \tag{4.1}$$

by the standard gradient descent iterations

$$x_{k+1} = x_k - \gamma \nabla F(x_k), \tag{4.2}$$

communicating the full gradient $\nabla F(x^k)$ in every iteration. Such a communication pattern usually appears in dual decomposition methods where $F(\cdot)$ is a dual function associated with some large-scale primal problem; we illustrate this in § 2.6.1. The partial gradient algorithms are used to solve separable optimization problems on the form

$$\underset{x}{\text{minimize}} \quad F(x) = \frac{1}{n} \sum_{i=1}^{n} F^i(x), \tag{4.3}$$

by gradient descent

$$x_{k+1} = x_k - \frac{\gamma}{n} \sum_{i=1}^{n} \nabla F^i(x_k), \tag{4.4}$$

and distributing the gradient evaluation on n nodes, each responsible for evaluating one of the partial gradients $\nabla F^i(x)$; see § 2.6.2). Clearly, full gradient communication is a special case of partial gradient communication with $n = 1$. However, considering the full gradient communication algorithms separately will enable us to get stronger results in that case.

4.1.2 Gradient Compression

We consider the following class of gradient compressors.

Definition 4.1. The operator $Q : \mathbb{R}^d \to \mathbb{R}^d$ is an ϵ-*compressor* if there exists a positive constant ϵ such that

$$\|Q(v) - v\| \leq \epsilon, \quad \forall v \in \mathbb{R}^d.$$

Definition 5.1 only requires bounded magnitude of the compression errors. A small value of ϵ corresponds to high accuracy. At the extreme when $\epsilon = 0$, we have $Q(v) = v$. An ϵ-compressor does not need to be unbiased (in constrast to those considered in [22, 117]) and is allowed to have a quantization error arbitrarily larger than magnitude of the original vector (in constrast to [126, Definition 2.1] and [127, Assumption A]). Definition 5.1 covers most popular compressors in machine learning and signal processing appplications, which substantiates the generality of our results later in the paper. One common example is the rounding quantizer, where each element of a full-precision vector $v_i \in \mathbb{R}$ is rounded to the closet point in a grid with resolution level $\Delta > 0$

$$[Q_{dr}(v)]^i = \text{sign}(v^i) \cdot \Delta \cdot \left\lfloor \frac{|v^i|}{\Delta} + \frac{1}{2} \right\rfloor. \tag{4.5}$$

This rounding quantizer is a ϵ-compressor with $\epsilon = d \cdot \Delta^2/4$, [128, 129, 130, 131]. In addition, if gradients are bounded, the sign compressor [28], the deterministic sparsification [115], and the deterministic sparsification and quantization [22, 115] are all ϵ- compressors.

4.2 The Limits of Direct Gradient Compression

To reduce communication overhead in distributed optimization, it is most straightforward to compress the gradients directly. The goal of this section is to illustrate the limits of this approach, which motivates our gradient correction mechanisms for compression algorithms in § 4.3.

4.2.1 Full Gradient Communication and Quadratic Case

A major drawback with direct gradient compression is that it leads to error accumulation. To illustrate why this happens we start by considering convex quadratic objectives

$$F(x) = \frac{1}{2}x^T Hx + b^T x. \tag{4.6}$$

Gradient descent using compressed gradients reduces to

$$x_{k+1} = x_k - \gamma Q\left(\nabla F(x_k)\right), \tag{4.7}$$

which can be equivalently expressed as

$$x_{k+1} = \overbrace{(I - \gamma H)}^{:=A_\gamma} x_k - \gamma b + \gamma\left(\nabla f(x_k) - Q\left(\nabla F(x_k)\right)\right). \tag{4.8}$$

Hence,

$$\begin{aligned} x_{k+1} - x^\star = A_\gamma^{k+1}(x_0 - x^\star) \\ + \gamma \sum_{j=0}^{k} A_\gamma^{k-j}\left(\nabla F(x_j) - Q\left(\nabla F(x_j)\right)\right). \end{aligned} \tag{4.9}$$

where $x^\star$ is the optimal solution and the equality follows from the fact that $Hx^\star + b = 0$. The final term of Equation (4.9) describes how the compression errors from every iteration accumulate. We show how error compensation helps to remove this accumulation in § 4.3. Even though the error accumulates, the compression error will remain bounded if the matrix A_γ is stable (which can be achieved by a sufficiently small step-size), as illustrated in the following theorem.

Theorem 4.1. *Consider the optimization problem over the objective function* (4.6) *where H is positive definite and let μ and L be the smallest and largest eigenvalues of H, respectively. Then, the iterates $\{x_k\}_{k\in\mathbb{N}}$ generated by* (4.7) *satisfy*

$$\|x_k - x^\star\| \leq \rho^k \|x_0 - x^\star\| + \frac{1}{\mu}\epsilon,$$

where

$$\rho = \begin{cases} 1 - 1/\kappa & \text{if} \quad \gamma = 1/L \\ 1 - 2/(\kappa + 1) & \text{if} \quad \gamma = 2/(\mu + L) \end{cases},$$

and $\kappa = L/\mu$ is the condition number of H.

Proof. See Appendix 4.B.1. □

Theorem 4.1 shows that the iterates of the compressed gradient descent in Equation (4.7) converge linearly to solution with residual error ϵ/μ. The theorem recovers the results of classical gradient descent when $\epsilon = 0$.

We show in § 4.2.3 that this upper bound is tight. With our error-compensated method as presented in § 4.3 we can achieve arbitrarily high solution accuracy even for fixed $\epsilon > 0$ and $\mu > 0$. These results can be generalized to include partial gradient communication, stochastic, and non-convex optimization problems as we show next.

4.2.2 Partial Gradient Communication

We now study direct gradient compression in the partial gradient communication architecture. We focus on the distributed compressed stochastic gradient descent algorithm (D-CSGD)

$$x_{k+1} = x_k - \gamma \frac{1}{n} \sum_{i=1}^{n} Q(g^i(x_k)), \tag{4.10}$$

where each $g^i(x)$ is a partial stochastic gradient sent by worker node i to the central node. We assume that the stochastic gradient preserves the unbiasedness and bounded variance assumptions, i.e.

$$\mathbf{E} g^i(x) = \nabla F^i(x), \quad \text{and} \tag{4.11}$$

$$\mathbf{E}\|g^i(x) - \nabla F^i(x)\|^2 \leq \sigma^2, \quad \forall x \in \mathbb{R}^d. \tag{4.12}$$

Notice that unlike [88, Assumption 1] and [132, Assumption 3] this condition only requires similarity between the local gradient and its stochastic oracle but allows for arbitrary differences between the whole data and local data distributions. In the deterministic case, we have the following rate result analogous to Theorem 4.1.

Theorem 4.2. *Consider the optimization problem* (4.3) *where* $F^i(\cdot)$ *are* L*-smooth and* $F(\cdot)$ *is* μ*-strongly convex. Suppose that* $g^i(x_k) = \nabla F^i(x_k)$*. If* $Q(\cdot)$ *is the* ϵ*-compressor and* $\gamma = 2/(\mu + L)$ *then*

$$\|x_k - x^\star\| \leq \rho^k \|x_0 - x^\star\| + \frac{1}{\mu}\epsilon.$$

where $\rho = 1 - 2/(\kappa + 1)$ *and* $\kappa = L/\mu$*.*

Proof. See Appendix 4.B.2. □

Furthermore, in the general stochastic case we have the following result.

Theorem 4.3. *Consider the optimization problem* (4.3) *where each* $F^i(\cdot)$ *is* L*-smooth, and the iterates* $\{x_k\}_{k\in\mathbb{N}}$ *generated by* (4.10) *under the assumption that the underlying partial stochastic gradients* $g^i(x_k)$ *satisfies the unbiased and bounded variance assumptions in Equation* (4.11) *and* (4.12)*. Assume that* $Q(\cdot)$ *is the* ϵ*-compressor and* $\gamma < 1/(3L)$*.*

a) (non-convex problems) Then,

$$\min_{l\in[0,k]} \mathbf{E}\|\nabla F(x_l)\|^2 \leq \frac{1}{k+1}\frac{2}{\gamma}\frac{1}{1-3L\gamma}\left(F(x_0)-F(x^\star)\right) + \frac{3L}{1-3L\gamma}\gamma\sigma^2 + \frac{1+3L\gamma}{1-3L\gamma}\epsilon^2. \tag{4.13}$$

b) (strongly-convex problems) If $F(\cdot)$ *is also* μ*-strongly convex, then*

$$\mathbf{E}\left(F(\bar{x}_k)-F(x^\star)\right) \leq \frac{1}{k+1}\frac{1}{2\gamma}\frac{1}{1-3L\gamma}\|x_0-x^\star\|^2 + \frac{3}{1-3L\gamma}\gamma\sigma^2 + \frac{1}{2}\frac{1/\mu+3\gamma}{1-3L\gamma}\epsilon^2, \tag{4.14}$$

where $\bar{x}_k = \sum_{l=0}^k x_l/(k+1)$*.*

Proof. See Appendix 4.B.3 □

Theorem 4.3 establishes a sub-linear convergence of D-CSGD toward the optimum with a residual error depending on the stochastic gradient noise σ, compression ϵ, problem parameters μ, L and the step-size γ. In particular, the residual error consists of two terms. The first term comes from the stochastic gradient noise σ^2 and decreases in proportion to the step-size. The second term arises from the precision of the compression ϵ, and cannot diminish towards zero no matter how small we choose the step-size. In fact, it can be bounded by noting that

$$\frac{1+3L\gamma}{1-3L\gamma} > 1 \quad \text{and} \quad \frac{1}{2}\frac{1/\mu+3\gamma}{1-3L\gamma} > \frac{1}{2\mu},$$

for all $\gamma \in (0, 1/(3L))$. This means that the upper bound in Equation (4.13) cannot become smaller than ϵ^2 and the upper bound in Equation (4.14) cannot become smaller than $\epsilon^2/(2\mu)$.

4.2.3 Limits of Direct Compression: Lower Bound

We now show that the bounds derived above are tight.

Example 4.1. *Consider the scalar optimization problem*

$$\underset{x}{\text{minimize}} \quad \frac{\mu}{2}x^2.$$

and the iterates generated by the CGD algorithm

$$x_{k+1} = x_k - \gamma Q(F'(x_k)) = x_k - \gamma\mu Q(x_k), \tag{4.15}$$

where $Q(\cdot)$ is the ϵ-compression (see Definition 5.1)

$$Q(z) = \begin{cases} z - \epsilon\frac{z}{|z|} & \text{if } z \neq 0 \\ \epsilon & \text{otherwise.} \end{cases}$$

If $\gamma \in (0, 1/\mu]$ and $|x_0| > \epsilon$ then for all $k \in \mathbb{N}$ we have $x^\star = 0$ and

$$\begin{aligned} |x_{k+1} - x^\star| =& |x_k - \gamma Q(F'(x_k))| \\ =& (1-\mu\gamma)|x_k| + \gamma\epsilon \\ =& (1-\mu\gamma)^{k+1}|x_0| + \epsilon\gamma\sum_{i=0}^{k}(1-\mu\gamma)^i \\ =& (1-\mu\gamma)^{k+1}|x_0| + \epsilon\gamma\frac{1-(1-\mu\gamma)^{k+1}}{\mu\gamma} \\ =& (1-\mu\gamma)^{k+1}(|x_0| - \epsilon) + \epsilon/\mu \\ \geq& \epsilon/\mu, \end{aligned}$$

where we have used that $x^\star = 0$. In addition,

$$F(\bar{x}_k) - F(x^\star) = \frac{\mu}{2}\frac{1}{k+1}\sum_{i=0}^{k}|x_i|^2 \geq \frac{1}{2\mu}\epsilon^2,$$

where $\bar{x}_k = \sum_{i=0}^{k} x_i/(k+1)$.

The above example shows that the ϵ-compressor cannot achieve accuracy better than ϵ/μ and $\epsilon^2/(2\mu)$ in terms of $\|x_k - x^\star\|^2$ and $F(\bar{x}_k) - F(x^\star)$, respectively. These lower bounds match the upper bound in Theorem 4.1, and the upper bound (4.14) in Theorem 4.3 if the step-size is sufficiently small. However, in the next section we show the surprising fact that an arbitrarily good solution accuracy can be obtained with ϵ-compressor and any $\epsilon > 0$ if we include a simple correction step in the optimization algorithms.

4.3 Error Compensated Gradient Compression

In this section we illustrate how error compensation can avoid the accumulation of previous compression errors in compressed gradient methods. We first introduce our error compensation mechanism and illustrate its benefits on quadratic problems, as shown next.

4.3.1 Error Compensation: Algorithm and Illustrative Example

To introduce the error compensation algorithm and show how it avoids the accumulation of compression errors, we again consider the quadradic problem

$$F(x) = \frac{1}{2}x^T Hx + b^T x.$$

The basic idea of the error compensation scheme is to compute the compression error in each iteration and compensate for it in the next search direction. For quadratic problem and full gradient descent, the iterations can be written as

$$\begin{aligned} x_{k+1} &= x_k - \gamma Q(\nabla F(x_k) + A_\gamma e_k) \\ e_{k+1} &= \underbrace{\nabla F(x_k) + A_\gamma e_k}_{\text{Input to Compressor}} - \underbrace{Q(\nabla F(x_k) + A_\gamma e_k)}_{\text{Output from Compressor}}. \end{aligned} \tag{4.16}$$

with $e_0 = 0$ and $A_\gamma = I - \gamma H$. This algorithm is similar to the direct gradient compression in Equation (4.7). However, the main difference is that we have introduced the memory term e_k in the gradient update. The term e_k is essentially the *compression error*, the difference between the compression input and output. To see how the error correction is helpful, consider the *gradient error*

$$c_k = \underbrace{\nabla F(x_k)}_{\text{True Gradient}} - \underbrace{Q(\nabla F(x_k) + A_\gamma e_k)}_{\text{Approximated Gradient Step}}.$$

The compression error can then be re-written as

$$e_{k+1} = c_k + A_\gamma e_k,$$

which reduces to

$$e_k = \sum_{j=0}^{k-1} A_\gamma^{k-1-j} c_j.$$

With this in mind, we can re-write the algorithm step as

$$x_{k+1} = A_\gamma x_k - \gamma b + \gamma c_k$$

and establish that

$$\begin{aligned} x_{k+1} - x^\star &= A_\gamma^{k+1}(x_0 - x^\star) + \gamma \sum_{i=0}^{k} A_\gamma^{k-i} c_i \\ &= A_\gamma^{k+1}(x_0 - x^\star) + \gamma e_{k+1}. \end{aligned} \tag{4.17}$$

Notice that here the residual error depends only on the latest compression error e_{k+1}, instead of the accumulation of previous compression errors as in

Equation (4.9). In particular, $\|e_{k+1}\| \leq \epsilon$ if $Q(\cdot)$ is an ϵ-compressor and we do not accumulate compression errors. This means that we can recover high solution accuracy given proper step-size tuning. We illustrate this in the following theorem.

Theorem 4.4. *Consider the optimization problem over objective function* (4.6) *where H is positive definite, and let μ and L be the smallest and largest eigenvalues of H, respectively. Then, the iterates $\{x_k\}_{k\in\mathbb{N}}$ generated by* (4.16) *with $A_\gamma = I - \gamma H$ and $e_0 = 0$ satisfy*

$$\|x_k - x^\star\| \leq \rho^k \|x_0 - x^\star\| + \gamma\epsilon,$$

where

$$\rho = \begin{cases} 1 - 1/\kappa & \text{if} \quad \gamma = 1/L \\ 1 - 2/(\kappa+1) & \text{if} \quad \gamma = 2/(\mu+L) \end{cases},$$

and $\kappa = L/\mu$.

Proof. See Appendix 4.B.4. □

Theorem 4.4 implies that error-compensated gradient descent has linear convergence rate and can attain arbitrarily high solution accuracy by decreasing the step-size. Comparing with Theorem 4.1, we note that error compensation attains lower residual error than direct compression if we insist on maintaining the same convergence rate. In particular, error compensation in Equation (4.16) with $\gamma = 1/L$ and $\gamma = 2/(\mu + L)$ reduces compression error κ and $(\kappa + 1)/2$, respectively. Hence, the benefit is especially pronounced for ill-conditioned problems [133]. Finally, Figure 4.1 shows that our worst-case bound in Theorem 4.4 is empirically shown to be tight for least-squares problems over synthetic data sets.

We next generalize these results to stochastic gradient methods under partial gradient communication architectures.

4.3.2 Partial Gradient Communication

For optimization with partial gradient communication, the natural generalization of error-compensated gradient algorithms consists of the following steps: at each iteration in parallel, worker nodes compute their local stochastic gradients $g^i(x)$ and add a local error compensation term e^i before applying the ϵ-compressor. The master node waits for all compressed gradients and updates the decision vector by

$$x_{k+1} = x_k - \gamma \frac{1}{n} \sum_{i=1}^{n} Q(g^i(x_k) + A_k^i e_k^i), \tag{4.18}$$

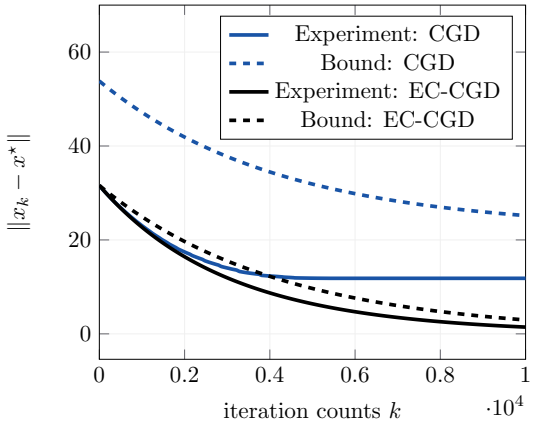


Figure 4.1: The performance of CGD (4.7) and EC-CGD (4.16) with their theoretical bounds presented in Theorems 4.1 and 4.4 for least-squares problems over synthetic data sets with $40,000$ data points and $1,000$ problem variables. Here, we set the step-size $\gamma = 1/L$ and the initial point $x_0 = 0$.

while each worker i updates its memory e^i according to

$$e^i_{k+1} = g^i(x_k) + A^i_k e^i_k - Q(g^i(x_k) + A^i_k e^i_k). \tag{4.19}$$

Here, γ is the fixed, positive step-size and $A^i_k \in \mathbb{R}^{d\times d}$ is the weighted matrix. The compression schemes can be divided into two categories: Hessian-fee and Hessian-aided error compensation schemes. We provide theoretical details of these compensation algorithms separately in the next section.

4.3.2.1 Hessian-Free Error Compensation

Stochastic gradient descent with Hessian-free error compensation updates the solution according to Equation (4.18) and (4.19) with $A^i_k = I$. These error compensation algorithms were empirically shown to have almost comparable convergence performance as full-precision algorithms [28, 123]. Motivated by these practical observations, we provide the first theoretical analysis of these compensated algorithms with the deterministic sparsification (according to Definition 3.2). In essence, we show explicit formulas of step-size and convergence rate under mild conditions as shown in the following theorem.

Theorem 4.5. *Consider the optimization problem* (4.3) *where each* $F^i(\cdot)$ *is* L*-smooth, and the iterates* $\{x_k\}_{k\in\mathbb{N}}$ *generated by* (4.18) *with* $A^i_k = I$*. Assume that* $Q(\cdot)$ *is the deterministic sparsification according to Definition 3.2 which*

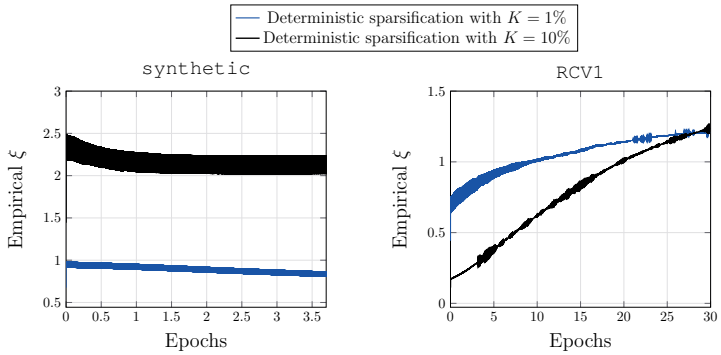


Figure 4.2: Validating Condition (4.20) on least-squares problems.

satifies $\|Q(v) - v\| \leq \alpha\|v\|$ *with* $\alpha = \sqrt{1 - K/d}$, *and*

$$\left\| Q\left(\frac{1}{n}\sum_{i=1}^{n} v_k^i\right) - \frac{1}{n}\sum_{i=1}^{n} Q\left(v_k^i\right) \right\| \leq \xi \left\| \frac{1}{n}\sum_{i=1}^{n} g^i(x_{k-1}) \right\|, \tag{4.20}$$

where $v_k^i = g^i(x_{k-1}) + e_{k-1}^i$ *and* ξ *is a positive constant. Further suppose that stochastic gradients* $g_i(x)$ *are unbiased and satisfy* $\mathbf{E}\|\sum_{i=1}^n g^i(x)/n\|^2 \leq M^2$ *for all* $x \in \mathbb{R}^d$. *If* $\gamma \leq 1/(\beta L)$ *where* $\beta = 2(\alpha + \xi)^2 \sum_{l=0}^{k-1} (2\alpha^2)^l$ *for a fixed count* $k \in \mathbb{N}$, *then*

$$\min_{l \in [0,k]} \mathbf{E}\|\nabla F(x_l)\|^2 \leq \frac{2}{\gamma(k+1)} \left(F(x_0) - F(x^\star)\right) + \gamma L M^2. \tag{4.21}$$

Proof. See Appendix 4.B.5. □

This theorem shows the sub-linear convergence of algorithms with Hessian-free error compensation toward the optimum with the residual error. The step-size tuning is affected by the compression accuracy α, while the residual term depends on the norms of stochastic gradients M^2. Also note that we can obtain the solution with high accuracy. As the step-size decreases, the residual term becomes arbitrarily small.

Even though Condition (4.20) seems limited, its validity can be shown on the experimental results. We validate this assumption on the least-squares problems on synthetic data and benchmark data RCV1 [134]. In particular, we generated the dense synthetic data set with 4000 samples and 2000 features, while we used RCV1 with 20242 samples and 47236 features. Figure 4.2 indicates that this condition appears to hold with relatively low, stable values of the constant ξ.

4.3.2.2 Hessian-Aided Error Compensation

Motivated by the result from Section 4.3.1, we consider stochastic gradient descent with Hessian-aided error compensation. In essence, we study the convergence results of these error-compensated algorithms according to (4.18) and (4.19) with the weighted matrix defined by

$$A_k^i = I - \gamma H_k^i \tag{4.22}$$

where H_k^i is either a deterministic or stochastic version of the Hessian $\nabla^2 F^i(x_k)$. In this section, we define the stochastic Hessian in analogous way as the stochastic gradient as follows:

$$\mathbf{E}[H_k^i] = \nabla^2 F^i(x_k), \quad \text{and} \tag{4.23}$$

$$\mathbf{E}\|H_k^i - \nabla^2 F^i(x_k)\|^2 \leq \sigma_H^2. \tag{4.24}$$

Notice that H_k^i is a local information of worker i. In real implementations, each worker can form the stochastic Hessian and the stochastic gradient independently at random. In the deterministic case the algorithm has similar convergence properties as the error compensation for the quadradic problems studied above.

Theorem 4.6. *Consider the optimization problem* (4.3) *where* $F^i(\cdot)$ *are* L*-smooth and* $f(\cdot)$ *is* μ*-strongly convex. Suppose that* $g^i(x_k) = \nabla F^i(x_k)$ *and* $H_k^i = \nabla^2 F^i(x_k)$*. If* $Q(\cdot)$ *is the* ϵ*-compressor and* $\gamma = 2/(\mu + L)$ *then*

$$\|x_k - x^\star\| \leq \rho^k \|x_0 - x^\star\| + \gamma \epsilon C,$$

where $\rho = 1 - 2/(\kappa + 1)$*,* $\kappa = L/\mu$*,* $C = 1 + \gamma L(\kappa + 1)$*.*

Proof. See Appendix 4.B.6. □

The theorem shows that the conclusions from the quadratic case can be extended to general strongly-convex functions and multiple nodes. In particular, the algorithm converges linearly to an approximately optimal solution with higher precision as the step-size γ decreases. We now illustrate the results in the general stochastic case.

Theorem 4.7. *Consider the optimization problem* (4.3) *where each* $F^i(\cdot)$ *is* L*-smooth, and the iterates* $\{x_k\}_{k\in\mathbb{N}}$ *generated by* (4.18) *with* $\gamma_k = \gamma$ *and* A_k^i *defined by Equation* (4.22)*, under the assumptions of stochastic gradients* $g^i(x_k)$ *in Equation* (4.11) *and* (4.12)*, and stochastic Hessians* H_k^i *in Equation* (4.23) *and* (4.24)*. Assume that* $Q(\cdot)$ *is an* ϵ*-compressor and that* $e_0^i = 0$ *for all* $i \in [1, n]$*.*

a) (non-convex problems) If $\gamma < 1/(3L)$*, then*

$$\min_{l\in[0,k]} \mathbf{E}\|\nabla F(x_l)\|^2 \leq \frac{1}{k+1}\frac{2}{\gamma}\frac{1}{1-3L\gamma}(F(x_0)-F(x^\star)) + \frac{3L}{1-3L\gamma}\gamma\sigma^2 + \frac{\alpha_2}{1-3L\gamma}\gamma^2\epsilon^2,$$

where $\alpha_2 = L^2 + (2+6L\gamma)(\sigma_H^2 + L^2)$.

b) (strongly-convex problems) If $F(\cdot)$ *is also* $\mu-$*strongly convex, and* $\gamma < (1-\beta)/(3L)$ *with* $0<\beta<1$*, then*

$$\mathbf{E}\left(F(\bar{x}_k) - F(x^\star)\right) \leq \frac{1}{k+1}\frac{1}{2\gamma}\frac{1}{1-\beta-3L\gamma}\|x_0 - x^\star\|^2 + \frac{3}{2}\frac{1}{1-\beta-3L\gamma}\gamma\sigma^2 + \frac{1}{2}\frac{\alpha_1}{1-\beta-3L\gamma}\gamma^2\epsilon^2,$$

where $\alpha_1 = \mu + L/\beta + (4/\mu + 6\gamma)\left(\sigma_H^2 + L^2\right)$ *and* $\bar{x}_k = \sum_{l=0}^k x_l/(k+1)$.

Proof. See Appendix 4.B.7. □

The theorem establishes the sub-linear convergence of the Hessian-aided error-compensation method at the rate $\mathcal{O}(1/k)$ toward the optimum with a residual error. Like Theorem 4.3 for direct gradient compression, the residual error consists of two terms. The first residual term depends on the stochastic gradient noise σ^2 and the second term depends on the precision of the compression ϵ. The first term can be made arbitrary small by decreasing the step-size $\gamma > 0$, similarly as in Theorem 4.3. However, unlike in Theorem 4.3, here we can make the second residual term arbitrarily small by decreasing $\gamma > 0$. In particular, for a fixed $\epsilon > 0$, the second residual term goes to zero at the rate $\mathcal{O}(\gamma^2)$. This means that in absence of gradient noise ($\sigma^2 = 0$) we can get an arbitrarily high solution accuracy even when the compression resolution ϵ is fixed.

4.3.3 Comparison to Hessian-Free Error Compensation

We now compare the Hessian-aided error compensation algorithms to the Hessian-free error compensation algorithms, which was analyzed by the other recent works [124, 125, 126, 127, 135, 136]. Unlike the Hessian-aided compensation, the Hessian-free compensation schemes keep in memory the sum (or weighted sum) of all previous compression errors. To illustrate this, we consider the centralized algorithms for quadratic problems in Section 4.3.1. Then, we can write the centralized Hessian-free compensation algorithms in the form of Equation (4.18) and (4.19) (in Section 4.3.2) by setting $n = 1$ and $A_k^1 = \alpha \cdot I$ with $\alpha \in (0, 1]$. If we perform the similar convergence study as

we did for the centralized Hessian-aided compensation algorithm for quadratic problems in Section 4.3.1, then we have

$$x_{k+1} - x^\star = A_\gamma^{k+1}(x_0 - x^\star) + \gamma e_{k+1} + \gamma \sum_{l=0}^{k} A_\gamma^{k-l} B_{\alpha,\gamma} e_l$$

where $A_\gamma = I - \gamma H$ and $B_{\alpha,\gamma} = (1-\alpha)I - \gamma H$. The final term shows that these Hessian-free compensation schemes do not remove the accumulated quantization errors, even though they have been shown to outperform direct compression. However, the Hessian-aided error compensation does avoid all of the accumulated error, as shown in Section 4.3.1. This example shows why the Hessian-based compensation schemes can be more effective than the Hessian-free compensation schemes.

We validate the superior performance of Hessian-based error compensation over existing schemes in Section 4.4. To reduce computing and memory requirements, we propose a Hessian approximation (using only the diagonal elements of the Hessian). Error compensation with this approximation is shown to have comparable performance to using the full Hessian.

4.3.4 Algorithm Complexity and Hessian Approximation

The Hessian-aided compensation scheme improves the iteration complexity of compressed gradient methods, both of the methods that use direct compression and error-compensation. This reduces the number of gradient transmissions, which in turn makes our compression more communication efficient than the existing Hessian-free compensation schemes. However, the improved communication complexity comes at the price of additional computations, since the Hessian-aided compensation uses the second-order information. In particular, from Equations (4.18) and (4.19) with $A_k^i = I - \gamma H_k^i$, computing the compressed gradient at each node requires $\mathcal{O}(d^2)$ arithmetic operations to multiply the Hessian matrix by the compression error. On the other hand, direct compression and existing Hessian-free compensation methods require only $\mathcal{O}(d)$ operations to compute the compressed gradient. Thus, the Hessian-aided compensation methods are more communication efficient than existing Hessian-free compensation schemes but achieves that by additional computations at nodes each iteration. We can improve the computational efficiency of the Hessian-aided error-compensation by using computationally efficient Hessian approximations. For example, the Hessian can be approximated by using only its diagonal elements. This reduces the computation of each compression to $\mathcal{O}(d)$ operations, comparable to existing schemes. We show in the next section that this approach gives good results on both convex and non-convex problems. Alternatively, we might use standard Hessian approximations from quasi-Newton methods such as BFGS [31], [32] or update the Hessian less frequently.

4.4 Numerical Results

In this section, we validate the superior convergence properties of Hessian-aided error compensation compared to the state-of-the-art. We also show that error compensation with a diagonal Hessian approximation shares many benefits with the full Hessian algorithm. In particular, we evaluate the compensation schemes on centralized stochastic gradient descent and distributed gradient descent for the minimization problem (4.3) with

$$F^i(x) = \sum_{j=1}^{m} \ell(x; z_j^i, y_j^i), \quad \text{for} \quad i = 1, \dots, n. \tag{4.25}$$

Here, $(z_1^i, y_1^i), \dots, (z_m^i, y_m^i)$ are data samples associated with the component function $F^i(x)$, with feature vectors $z_j^i \in \mathbb{R}^d$ and associated class labels $y_j^i \in \{-1, 1\}$. In all simulations, we normalized each data sample by its Euclidean norm and used the initial point $x_0 = 0$. In plot legends, `Non-EC` denotes the compressed gradient method (4.10), while `EC-I`, `EC-H` and `EC-diag-H` are compensation methods governed by the iteration described in Equations (4.18) and (4.19) with $A_k^i = I$, $A_k^i = I - \gamma H_k^i$ and $A_k^i = I - \gamma \text{diag}(H_k^i)$, respectively. Here, H_k^i is the Hessian matrix associated with the stochastic gradient $g^i(x_k)$ and $\text{diag}(H_k^i)$ is a matrix with the diagonal entries of H_i^k on its diagonal and zeros elsewhere. Thus, `EC-I` is the existing Hessian-free error compensation scheme in the literature, `EC-H` denotes the Hessian-aided error compensation, and `EC-diag-H` is the error compensation using a diagonal Hessian approximation. Throughout the experiments, we also set the same step-size for all compression algorithms. In particular, $\gamma = 2/(\mu + L)$ and $0.1/L$ for linear least-squares regression over synthetic data and benchmark data (i.e. `mushrooms`, `a9a` and `w8a`), while $\gamma = 1/(60\sqrt{3}L)$ for non-convex robust linear regression.

4.4.1 Linear Least Squares Regression

We consider the least-squares regression problem (4.3) with each component function on the form (4.25) with

$$\ell(x; z_j^i, y_j^i) = (\langle z_j^i, x \rangle - y_j^i)^2/2.$$

Clearly, $F^i(x)$ is strongly convex and smooth with parameters μ^i and L^i, denoting the smallest and largest eigenvalues, respectively, of the matrix $A^i = \sum_{j=1}^m z_j^i (z_j^i)^T$. For this problem, the objective function $F(\cdot)$ is thus μ-strongly convex and L-smooth with $\mu = \min_{i \in [1,n]} \mu^i$ and $L = \max_{i \in [1,n]} L^i$, respectively.

We evaluated full gradient methods with three Hessian-aided compensation variants; `EC-H`, `EC-diag-H` and `EC-BFGS-H`. Here, `EC-BFGS-H` is the compensation update where the full Hessian H_k is approximated using the BFGS

update [137]. Figure 4.1 suggests that the worst-case bound from Theorem 4.4 is tight for error compensated methods with EC−H. In addition, compensation updates that approximate Hessians by using only the diagonal elements and by the BFGS method perform worse than the full Hessian compensation scheme.

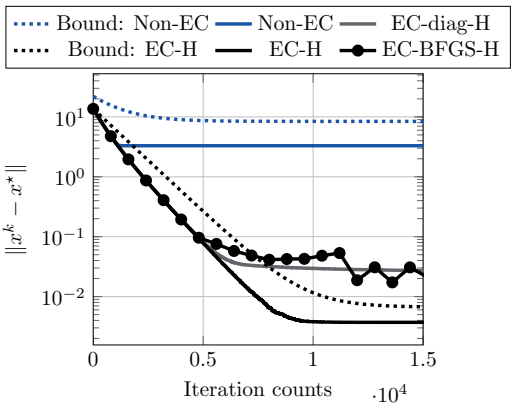


Figure 4.1: Comparisons of D-CSGD and D-EC-CSGD with one worker node and with $g_i(x;\xi_i) = \nabla f_i(x)$ for the least-squares problems over synthetic data with $4,000$ data points and 400 features, when the deterministic rounding quantizer with $\Delta = 1$ is applied. We set the step-size $\gamma = 2/(\mu + L)$.

We begin by considering the deterministic rounding quantizer (4.5). Figure 4.2 shows that compressed SGD cannot reach a high solution accuracy, and its performance deteriorates as the quantization resolution decreases (the compression is coarser). Error compensation, on the other hand, achieves a higher solution accuracy and is surprisingly insensitive to the amount of compression.

Figures 4.3 and 4.4 evaluate error compensation the binary (sign) compressor on several data sets in both single and multi-node settings. Almost all variants of error compensation achieve higher solution accuracy after a sufficiently large number of iterations. In particular, EC−H outperforms the other error compensation schemes in terms of high convergence speed and low residual error guarantees for centralized SGD and distributed GD. In addition, EC−diag−H has almost the same performance as EC−H.

4.4.2 Non-convex Robust Linear Regression

Next, we consider the robust linear regression problem [138, 139], with the component function (4.25) and

$$\ell(x; z_j^i, y_j^i) = (\langle z_j^i, x\rangle - y_j^i)^2/(1 + (\langle z_j^i, x\rangle - y_j^i)^2).$$

Here, $F^i(x)$ is smooth with $L^i = \sum_{j=1}^m \|z_j^i\|^2/(6\sqrt{3})$, and thus $F(x)$ is smooth with the parameter $L = \max_{i\in[1,n]} L^i$.

We consider the binary (sign) compressor and evaluated many compensation algorithms on different data sets; see Figures 4.5 and 4.6, Compared with direct compression, almost all error compensation schemes improve the solution accuracy, and `EC-H` provides a higher accurate solution than `EC-diag-H` and `EC-I` for both centralized and distributed architectures. Error compensation using diagonal elements of full Hessian (without any noise) also provides stronger performance than `EC-I` (see Figure 4.6). However, as the Hessian becomes stochastic, the advantage of the diagonal compensation diminishes, and its performance becomes similar to existing compensation schemes, as shown in Figure 4.5.

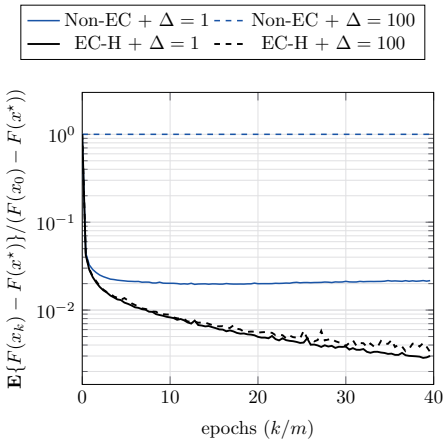


Figure 4.2: Comparisons of D-CSGD and D-EC-CSGD with one worker node using different compensation schemes for the least-squares problems over `a3a` when the deterministic rounding quantizer is applied. Here, the step-size $\gamma = 0.1/L$, the mini-batch size $b = |\mathcal{D}|/20$, where $|\mathcal{D}|$ is the total number of data samples.

4.4.3 Step-size tuning without the Lipschitz constant

In this paper we considered constant step-sizes that rely on the Lipschitz constant L. For some loss functions it is non-trivial to compute L. Therefore, it is worthwhile to illustrate the benefits of the Hessian-aided error-compensation for other step-size schemes that do not rely on the knowledge of L. We consider two step-size schemes: a) diminishing step-sizes [36, 141, 142] and line search techniques [51, 143, 144].

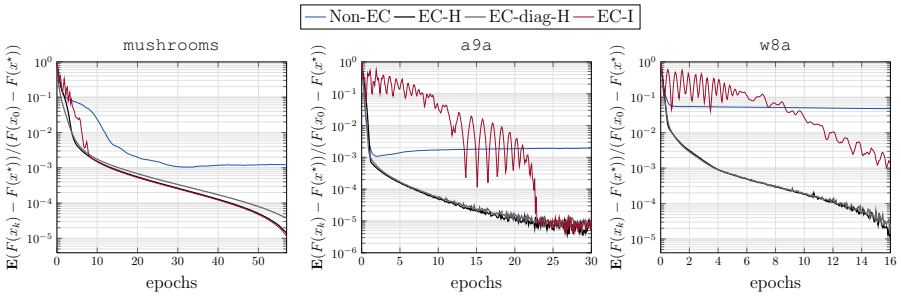


Figure 4.3: Comparisons of D-CSGD and D-EC-CSGD with one worker node using different compensation schemes for the least-squares problems over bench-marking data sets from [140] when the binary (sign) compression is applied. We set the step-size $\gamma = 0.1/L$, and the mini-batch size $b = |\mathcal{D}|/10$, where $|\mathcal{D}|$ is the total number of data samples.

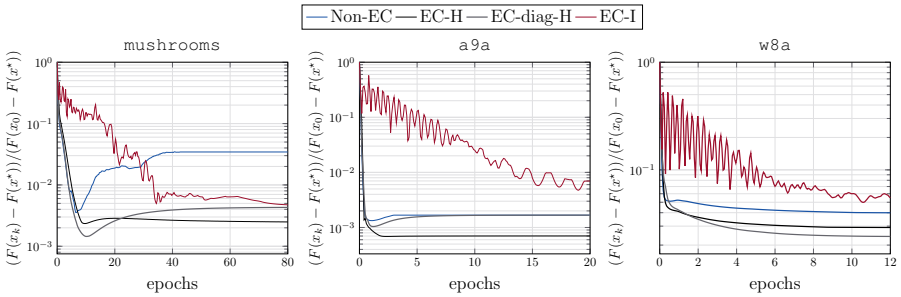


Figure 4.4: Comparisons of D-CSGD and D-EC-CSGD with $g^i(x) = \nabla F^i(x)$ using different compensation schemes for the least-squares problems over bench-marking data sets from [140] when the binary (sign) compression is applied. We set the step-size $\gamma = 0.1/L$, and 5 worker nodes.

We run the compensation method in (4.18) with $n = 1$. The diminishing step-size scheme is $\gamma_k = 1/(a + k)$ where $a > 0$. The step-sizes evaluated by the line search rule are generated as follows. We start with an initial Lipschitz estimate L_k. We then double L_k until the following inequality is satisfied

$$\tilde{F}(x_k - \gamma_k g_Q(x_k)) \leq \tilde{F}(x_k) - \gamma_k \langle g(x_k), g_Q(x_k) \rangle + \frac{\gamma_k^2 L_k}{2} \|g_Q(x_k)\|^2 \quad (4.26)$$

where

$$g_Q(x_k) = Q(g(x_k) + (I - \gamma_k H_k) e_k),$$

$\gamma_k = 1/(\theta L_k)$, and $\tilde{F}(x_k)$ and $g(x_k)$ is the stochastic function and gradient, respectively, based on the random subset of the local data at iteration k.

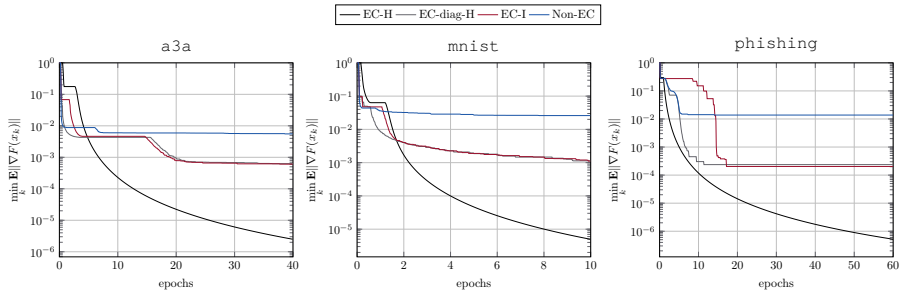


Figure 4.5: Comparisons of D-CSGD and D-EC-CSGD with one worker node using different compensation schemes for non-convex robust linear regression over bench-marking data sets from [140] when the binary (sign) compression is applied. We set the step-size $\gamma = 1/(60\sqrt{3}L)$, and the mini-batch size $b = |\mathcal{D}|/10$, where $|\mathcal{D}|$ is the total number of data samples.

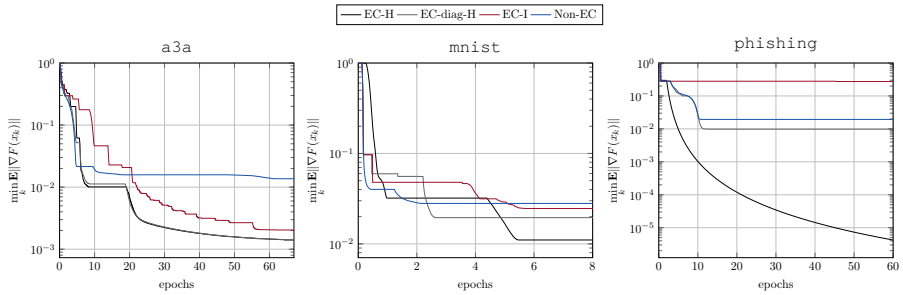


Figure 4.6: Comparisons of D-CSGD and D-EC-CSGD with $g^i(x) = \nabla F^i(x)$ using different compensation schemes for non-convex robust linear regression over bench-marking data sets from [140] when the binary (sign) compression is applied. We set the step-size $\gamma = 1/(60\sqrt{3}L)$, and 5 worker nodes.

We illustrate the results in Figure 4.7. We use the binary (sign) compressor and consider the linear least-squares regression problem over `a9a` as in Section 4.4.1. We let a and θ be the mini-batch size and 10, respectively. Figure 4.7 shows that Hessian information improves both the speed and solution accuracy of error compensation methods with both the diminishing step-size and the step-size based on line search procedures. These results show that our Hessian-aided error compensation can be beneficial for other time-varying step-sizes and step-sizes that do not rely on knowledge of L.

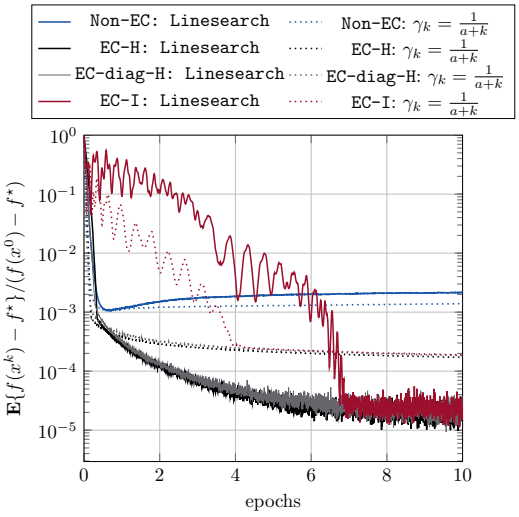


Figure 4.7: Convergence performance of all compensation methods with one worker for the least-squares problem over `a9a` when the binary compression is applied. We evaluated these methods with (a) the line search (in solid lines) and (b) the diminishing step-size $\gamma_k = 1/(a+k)$, where $\theta = 10$, $a = b$ and the mini-batch size $b = |\mathcal{D}|/10$ with $|\mathcal{D}|$ is the total number of data samples.

Appendix

4.A Review of Useful Lemmas

This section states lemmas which are instrumental in our convergence analysis.

Lemma 4.1. *For $x_i \in \mathbb{R}^d$ and a natural number N,*

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

Lemma 4.2. *For $x, y \in \mathbb{R}^d$ and a positive scalar θ,*

$$\|x+y\|^2 \leq (1+\theta)\|x\|^2 + (1+1/\theta)\|y\|^2.$$

Lemma 4.3. *For $x, y \in \mathbb{R}^d$ and a positive scalar θ,*

$$2\langle x, y\rangle \leq \theta\|x\|^2 + (1/\theta)\|y\|^2.$$

Lemma 4.4 ([73])**.** *Assume that $F(x)$ is convex and $L-$smooth, and the optimimum is denoted by $x^\star$. Then,*

$$\|\nabla F(x)\|^2 \leq 2L(F(x) - F(x^\star)), \quad \textit{for } x \in \mathbb{R}^d. \tag{4.27}$$

4.B Proof of Main Results

4.B.1 Proof of Theorem 4.1

The algorithm in Equation (4.7) can be written as

$$x_{k+1} = x_k - \gamma(\nabla F(x_k) + e_k),$$

where $e_k = Q(\nabla F(x_k)) - \nabla F(x_k)$. Using that $\nabla F(x^\star) = Hx^\star - b = 0$ we have

$$x_{k+1} - x^\star = (I - \gamma H)(x_k - x^\star) - \gamma e_k,$$

or equivalently

$$x_k - x^\star = (I - \gamma H)^k (x_0 - x^\star) - \gamma \sum_{i=0}^{k-1} (I - \gamma H)^{k-1-i} e_i. \tag{4.28}$$

By the triangle inequality and the fact that for a symmetric matrix $I - \gamma H$ we have

$$\|(I - \gamma H)x\| \leq \rho\|x\| \quad \text{for all } x \in \mathbb{R}^d.$$

where

$$\rho = \max_{i\in\{1,2,\ldots,d\}} |\lambda_i(I - \gamma H)| = \max_{i\in\{1,2,\ldots,d\}} |1 - \gamma\lambda_i(H)|.$$

We thus have

$$\|x_k - x^\star\| \leq \rho^k\|x_0 - x^\star\| + \gamma\sum_{i=0}^{k-1}\rho^{k-1-i}\epsilon.$$

In particular, when $\gamma = 1/L$ then $\rho = 1 - 1/\kappa$ meaning that

$$\|x_k - x^\star\| \leq (1 - 1/\kappa)^k \|x_0 - x^\star\| + \frac{1}{L}\sum_{i=0}^{k-1}(1 - 1/\kappa)^{k-1-i}\,\epsilon,$$

where $\kappa = L/\mu$. Since $1 - 1/\kappa \in (0, 1)$ we have

$$\sum_{i=0}^{k-1}(1 - 1/\kappa)^{k-1-i} \leq \sum_{i=0}^{\infty}(1 - 1/\kappa)^i = \kappa,$$

which implies

$$\|x_k - x^\star\| \leq (1 - 1/\kappa)^k \|x_0 - x^\star\| + \frac{1}{\mu}\epsilon.$$

Similarly, when $\gamma = 2/(\mu + L)$ then $\rho = 1 - 2/(\kappa + 1)$ and

$$\|x_k - x^\star\| \leq (1 - 2/(\kappa + 1))^k \|x_0 - x^\star\| + \frac{2}{\mu + L}\sum_{i=0}^{k-1}(1 - 2/(\kappa + 1))^{k-1-i}\,\epsilon.$$

Since $1 - 2/(\kappa + 1) \in (0, 1)$ we have

$$\sum_{i=0}^{k-1}(1 - 2/(\kappa + 1))^{k-1-i} \leq \sum_{i=0}^{\infty}(1 - 2/(\kappa + 1))^i = (\kappa + 1)/2.$$

This means that

$$\|x_k - x^\star\| \leq \left(\frac{\kappa - 1}{\kappa + 1}\right)^k \|x_0 - x^\star\| + \frac{1}{\mu}\epsilon.$$

4.B.2 Proof of Theorem 4.2

We can rewrite the direct compression algorithm (4.10) equivalently as Equation (4.29) with $\eta_k = 0$. By the triangle inequality for the Euclidean norm,

$$\|x_{k+1} - x^\star\| \leq \|x_k - x^\star - \gamma \nabla f(x_k)\| + \gamma \frac{1}{n} \sum_{i=1}^{n} \|e_k^i\|.$$

If $\gamma = 2/(\mu + L)$, by the fact that $f(\cdot)$ is L-smooth and that $\|e_k^i\| \leq \epsilon$

$$\|x_{k+1} - x^\star\| \leq \rho \|x_k - x^\star\| + \gamma\epsilon,$$

where $\rho = 1 - 2/(\kappa + 1)$ and $\kappa = L/\mu$. By recursively applying this main inequality, we have

$$\|x_{k+1} - x^\star\| \leq \rho^k \|x_0 - x^\star\| + \frac{\gamma}{1-\rho}\epsilon.$$

By rearranging the terms, we complete the proof.

4.B.3 Proof of Theorem 4.3

We can write the algorithm in Equation (4.10) equivalently as

$$x_{k+1} = x_k - \gamma \left(\nabla F(x_k) + \eta_k + e_k \right), \tag{4.29}$$

where

$$\eta_k = \frac{1}{n} \sum_{i=1}^{n} \left[g^i(x_k) - \nabla F^i(x_k) \right], \quad \text{and}$$
$$e_k = \frac{1}{n} \sum_{i=1}^{n} \left[Q\left(g^i(x_k) \right) - g^i(x_k) \right].$$

By Lemma 4.1, the bounded gradient assumption, and the definition of the ϵ-compressor we have

$$\mathbf{E}\|\eta_k\|^2 \leq \frac{1}{n} \sum_{i=1}^{n} \mathbf{E}\|g^i(x_k) - \nabla F^i(x_k)\|^2 \leq \sigma^2, \text{ and} \tag{4.30}$$

$$\|e_k\|^2 \leq \frac{1}{n} \sum_{i=1}^{n} \|Q\left(g^i(x_k) \right) - g^i(x_k)\|^2 \leq \epsilon^2. \tag{4.31}$$

Proof of Theorem 4.3-a)

By the Lipschitz smoothness assumption on $F(\cdot)$ and Equation (4.29) we have

$$F(x_{k+1}) \leq F(x_k) - \gamma \langle \nabla F(x_k), \nabla F(x_k) + \eta_k + e_k \rangle + \frac{L\gamma^2}{2} \|\nabla F(x_k) + \eta_k + e_k\|^2.$$

Due to the unbiased property of the stochastic gradient (i.e. $\mathbf{E}\eta_k = 0$), taking the expectation and applying Lemma 4.1, and Equation (4.30) and (4.31) yields

$$\mathbf{E}F(x_{k+1}) \leq \mathbf{E}F(x_k) - \left(\gamma - \frac{3L\gamma^2}{2}\right)\mathbf{E}\|\nabla F(x_k)\|^2 + \gamma\mathbf{E}\langle\nabla F(x_k), -e_k\rangle + \frac{3L\gamma^2}{2}(\sigma^2 + \epsilon^2).$$

Next, applying Lemma 4.3 with $x = \nabla F(x_k)$, $y = -e_k$ and $\theta = 1$ into the main result yields

$$\mathbf{E}F(x_{k+1}) \leq \mathbf{E}F(x_k) - \left(\frac{\gamma}{2} - \frac{3L\gamma^2}{2}\right)\mathbf{E}\|\nabla F(x_k)\|^2 + T,$$

where $T = (1+3L\gamma)\gamma\epsilon^2/2 + 3L\gamma^2\sigma^2/2$. By rearranging the terms and recalling that $\gamma < 1/(3L)$ we get

$$\mathbf{E}\|\nabla F(x_k)\|^2 \leq \frac{2}{\gamma}\frac{1}{1-3L\gamma}\left(\mathbf{E}F(x_k) - \mathbf{E}F(x_{k+1}) + T\right).$$

Using the fact that

$$\min_{l\in[0,k]}\mathbf{E}\|\nabla F(x_l)\|^2 \leq \frac{1}{k+1}\sum_{l=0}^{k}\mathbf{E}\|\nabla F(x_l)\|^2$$

and the cancelations of telescopic series we get

$$\min_{l\in[0,k]}\mathbf{E}\|\nabla F(x_l)\|^2 \leq \frac{1}{k+1}\frac{2}{\gamma}\frac{1}{1-3L\gamma}\left(\mathbf{E}F(x_0) - \mathbf{E}F(x_{k+1})\right) + \frac{2}{\gamma}\frac{1}{1-3L\gamma}T.$$

We can now conclude the proof by noting that $F(x^\star) \leq F(x)$ for all $x \in \mathbb{R}^d$

Proof of Theorem 4.3-b)

From the definition of the Euclidean norm and Equation (4.29),

$$\|x_{k+1} - x^\star\|^2 = \|x_k - x^\star\|^2 - 2\gamma\langle\nabla F(x_k) + \eta_k + e_k, x_k - x^\star\rangle + \gamma^2\|\nabla F(x_k) + \eta_k + e_k\|^2. \tag{4.32}$$

By taking the expected value on both sides and using the unbiasedness of the stochastic gradient, i.e., that $\mathbf{E}\eta_k = 0$, and Lemma 4.1 and Equation (4.30) and (4.31) to get the bound

$$\mathbf{E}\|\nabla F(x_k) + \eta_k + e_k\|^2 \leq 3\mathbf{E}\|\nabla F(x_k)\|^2 + 3(\sigma^2 + \epsilon^2)$$

we have

$$\mathbf{E}\|x_{k+1} - x^\star\|^2 \leq \mathbf{E}\|x_k - x^\star\|^2 - 2\gamma\mathbf{E}\langle\nabla F(x_k) + e_k, x_k - x^\star\rangle + 3\gamma^2\mathbf{E}\|\nabla F(x_k)\|^2 + 3\gamma^2(\sigma^2 + \epsilon^2).$$

Applying Equation (2.4) with $x = x_k$ and $y = x^\star$ and using Lemma 4.4 with $x = x_k$ we have

$$\mathbf{E}\|x_{k+1} - x^\star\|^2 \leq (1 - \mu\gamma)\mathbf{E}\|x_k - x^\star\|^2 - 2(\gamma - 3L\gamma^2)\mathbf{E}[F(x_k) - F(x^\star)] + 2\gamma\mathbf{E}\langle e_k, x^\star - x_k\rangle + 3\gamma^2(\sigma^2 + \epsilon^2).$$

From Lemma 4.3 with $\theta = \mu$ and Equation (4.31), we have

$$2\gamma\langle e_k, x^\star - x_k\rangle \leq \mu\gamma\|x_k - x^\star\|^2 + \epsilon^2\gamma/\mu,$$

which yields

$$\mathbf{E}\|x_{k+1} - x^\star\|^2 \leq \mathbf{E}\|x_k - x^\star\|^2 - 2\gamma(1 - 3L\gamma)\mathbf{E}[F(x_k) - F(x^\star)] + T,$$

where $T = \gamma(1/\mu + 3\gamma)\epsilon^2 + 3\gamma^2\sigma^2$. By rearranging the terms and recalling that $\gamma < 1/3L$ we get

$$\mathbf{E}\left(F(x_k) - F(x^\star)\right) \leq \frac{1}{2\gamma}\frac{1}{1 - 3L\gamma}\left(\mathbf{E}\|x_k - x^\star\|^2 - \mathbf{E}\|x_{k+1} - x^\star\|^2 + T\right).$$

Define $\bar{x}_k = \sum_{l=0}^k x_l/(k+1)$. By the convexity of $F(\cdot)$ and from the cancelations of the telescopic series we have

$$\begin{aligned}\mathbf{E}\left(F(\bar{x}_k) - F(x^\star)\right) &\leq \frac{1}{k+1}\sum_{l=0}^{k}\mathbf{E}\left(F(x_l) - F(x^\star)\right) \\ &\leq \frac{1}{k+1}\frac{1}{2\gamma}\frac{1}{1 - 3L\gamma}\|x_0 - x^\star\|^2 + \frac{1}{2\gamma}\frac{1}{1 - 3L\gamma}T.\end{aligned}$$

Hence, the proof is complete.

4.B.4 Proof of Theorem 4.4

We can write the algorithm in Equation (4.16) equivalently as

$$x_{k+1} = x_k - \gamma(\nabla F(x_k) - c_k),$$

where

$$\begin{aligned}c_k &= \nabla f(x_k) - Q(\nabla F(x_k) + A_\gamma e_k), \quad \text{and} \\ e_{k+1} &= c_k + A_\gamma e_k\end{aligned}$$

and $A_\gamma = I - \gamma H$. Following similar line of arguments as in the proof of Theorem 4.1 we obtain

$$x_k - x^\star = A_\gamma^k(x_0 - x^\star) + \gamma \sum_{i=0}^{k-1} A_\gamma^{k-1-i} c_i.$$

By using that $e_k = \sum_{i=0}^{k-1} A_\gamma^{k-1-i} c_i$ and $e_0 = 0$ we get that

$$x_k - x^\star = A_\gamma^k(x_0 - x^\star) + \gamma e_k.$$

Since A_γ is symmetric, by the triangle inequality and the fact that $\|e_k\| \leq \epsilon$ (since e_k is the compression error) we have

$$\|x_k - x^\star\| \leq \rho^k \|x_0 - x^\star\| + \gamma\epsilon,$$

where $\rho = \max_{i\in[1,d]} |1-\gamma\lambda_i|$. Now following similar arguments as used in the proof of Theorem 4.1 If $\gamma = 1/L$ then $\rho = 1 - 1/\kappa$. Since $1 - 1/\kappa \in (0,1)$ we have

$$\|x_k - x^\star\| \leq \left(1 - \frac{1}{\kappa}\right)^k \|x_0 - x^\star\| + \frac{1}{L}\epsilon.$$

If $\gamma = 2/(\mu + L)$ then $\rho = 1 - 2/(\kappa + 1)$. Since $1 - 2/(\kappa + 1) \in (0,1)$ we have

$$\|x_k - x^\star\| \leq \left(\frac{\kappa - 1}{\kappa + 1}\right)^k \|x_0 - x^\star\| + \frac{2}{\mu + L}\epsilon.$$

4.B.5 Proof of Theorem 4.5

We can rewrite Equation (4.18) with $A_k^i = I$ equivalently as

$$\tilde{x}_{k+1} = \tilde{x}_k - \gamma \frac{1}{n} \sum_{i=1}^n g^i(x_k), \tag{4.33}$$

where

$$\tilde{x}_k = x_k - \gamma \frac{1}{n} \sum_{i=1}^n e_k^i. \tag{4.34}$$

Before proving the main result, we prove one lemma which is instrumental to our analysis.

Lemma 4.5. *Consider the sequences $\{x_k\}$ and $\{\tilde{x}_k\}$ generated by Equation (4.18) and (4.33), respectively. Assume that the stochastic gradient satisfies*

$$\mathbf{E}\left\|\frac{1}{n}\sum_{i=1}^n g^i(x)\right\|^2 \leq M^2$$

for all $x \in \mathbb{R}^d$, and that $Q(\cdot)$ is the K-greedy quantizer satisfying

$$\left\| Q\left(\frac{1}{n}\sum_{i=1}^{n} g^i(x_k) + e_k^i\right) - \frac{1}{n}\sum_{i=1}^{n} Q\left(g^i(x_k) + e_k^i\right) \right\| \leq \xi \left\| \frac{1}{n}\sum_{i=1}^{n} g^i(x_k) \right\|,$$

for $k \geq 0$ and some positive constant ξ. Then, for fixed $k \geq 0$

$$\mathbf{E}\|\tilde{x}_k - x_k\|^2 \leq \gamma^2 \cdot 2(\alpha+\xi)^2 M^2 \sum_{l=0}^{k-1} (2\alpha^2)^l,$$

where $\alpha = \sqrt{1-K/d}$.

Proof. From Lemma 4.1 and Equation (4.33), the Euclidean distance between x_k and $\tilde{x}_k$ satisfies

$$\|\tilde{x}_k - x_k\|^2 \leq \gamma^2 \left\| \frac{1}{n}\sum_{i=1}^{n} e_k^i \right\|^2. \tag{4.35}$$

To bound this distance, we need to determine the upper-bound of $\left\|\sum_{i=1}^{n} e_k^i/n\right\|$. From the compensation update according to Equation (4.19) with $A_k^i = I$,

$$\begin{aligned}
\left\| \frac{1}{n}\sum_{i=1}^{n} e_k^i \right\| &= \left\| \frac{1}{n}\sum_{i=1}^{n} g^i(x_{k-1}) + e_{k-1}^i - Q\left(g^i(x_{k-1}) + e_{k-1}^i\right) \right\| \\
&\leq \left\| \frac{1}{n}\sum_{i=1}^{n} g^i(x_{k-1}) + e_{k-1}^i - Q\left(\frac{1}{n}\sum_{i=1}^{n} g^i(x_{k-1}) + e_{k-1}^i\right) \right\| \\
&\quad + \left\| Q\left(\frac{1}{n}\sum_{i=1}^{n} g^i(x_{k-1}) + e_{k-1}^i\right) - \frac{1}{n}\sum_{i=1}^{n} Q\left(g^i(x_{k-1}) + e_{k-1}^i\right) \right\|,
\end{aligned}$$

where the last inequality comes from Cauchy-Schwarz's inequality. Assume that $Q(\cdot)$ is K-greedy quantizer which satifies $\|Q(v) - v\| \leq \alpha\|v\|$ with $\alpha = \sqrt{1-K/d}$, and

$$\left\| Q\left(\frac{1}{n}\sum_{i=1}^{n} v_k^i\right) - \frac{1}{n}\sum_{i=1}^{n} Q\left(v_k^i\right) \right\| \leq \xi \left\| \frac{1}{n}\sum_{i=1}^{n} g^i(x_{k-1}) \right\|,$$

where $v_k^i = g^i(x_{k-1}) + e_{k-1}^i$ and ξ is a positive constant. We then have

$$\begin{aligned}
\left\| \frac{1}{n}\sum_{i=1}^{n} e_k^i \right\| &\leq \alpha \left\| \frac{1}{n}\sum_{i=1}^{n} g^i(x_{k-1}) + e_{k-1}^i \right\| + \xi \left\| \frac{1}{n}\sum_{i=1}^{n} g^i(x_{k-1}) \right\| \\
&\leq (\alpha+\xi) \left\| \frac{1}{n}\sum_{i=1}^{n} g^i(x_{k-1}) \right\| + \alpha \left\| \frac{1}{n}\sum_{i=1}^{n} e_{k-1}^i \right\|.
\end{aligned}$$

Therefore,

$$\left\|\frac{1}{n}\sum_{i=1}^{n} e_k^i\right\|^2 \le 2(\alpha+\xi)^2\left\|\frac{1}{n}\sum_{i=1}^{n} g^i(x_{k-1})\right\|^2 + 2\alpha^2\left\|\frac{1}{n}\sum_{i=1}^{n} e_{k-1}^i\right\|^2,$$

or equivalently

$$\left\|\frac{1}{n}\sum_{i=1}^{n} e_k^i\right\|^2 \le 2(\alpha+\xi)^2\sum_{l=0}^{k-1}(2\alpha^2)^{k-1-l}\left\|\frac{1}{n}\sum_{i=1}^{n} g^i(x_{l-1})\right\|^2.$$

Further suppose that $\mathbf{E}\|\sum_{i=1}^n g^i(x)/n\|^2 \le M^2$ for all $x \in \mathbb{R}^d$. Then, taking the expectation over the inequality yields

$$\mathbf{E}\left\|\frac{1}{n}\sum_{i=1}^{n} e_k^i\right\|^2 \le 2(\alpha+\xi)^2 M^2\sum_{l=0}^{k-1}(2\alpha^2)^l.$$

Using this inequality and taking the expectation over (4.35), we complete the proof. □

Now, we prove the main result. By the Lipschitz smoothness of the whole objective function $f(\cdot)$ and (4.33),

$$F(\tilde{x}_{k+1}) \le F(\tilde{x}_k) - \gamma\left\langle \nabla F(\tilde{x}_k), \frac{1}{n}\sum_{i=1}^{n} g^i(x_k)\right\rangle + \frac{\gamma^2 L}{2}\left\|\frac{1}{n}\sum_{i=1}^{n} g^i(x_k)\right\|^2.$$

Taking the expectation and using the fact that $F(x) = \sum_{i=1}^n F^i(x)/n$ yield

$$\mathbf{E}F(\tilde{x}_{k+1}) \le \mathbf{E}F(\tilde{x}_k) - \gamma\mathbf{E}\left\langle \nabla F(\tilde{x}_k), \nabla F(x_k)\right\rangle + \frac{\gamma^2 L}{2}\mathbf{E}\left\|\frac{1}{n}\sum_{i=1}^{n} g^i(x_k)\right\|^2.$$

By the fact that $2\langle x, y\rangle = \|x\|^2 + \|y\|^2 - \|x-y\|^2$ for $x, y \in \mathbb{R}^d$ and that $F(\cdot)$ is L-smooth,

$$\begin{aligned}\mathbf{E}F(\tilde{x}_{k+1}) \le \mathbf{E}F(\tilde{x}_k) &- \frac{\gamma}{2}\mathbf{E}\|\nabla F(\tilde{x}_k)\|^2 - \frac{\gamma}{2}\mathbf{E}\left\|\nabla F(x_k)\right\|^2 \\ &+ \frac{\gamma L^2}{2}\mathbf{E}\left\|\tilde{x}_k - x_k\right\|^2 + \frac{\gamma^2 L}{2}\mathbf{E}\left\|\frac{1}{n}\sum_{i=1}^{n} g^i(x_k)\right\|^2.\end{aligned}$$

Assume that stochastic gradients satisfy $\mathbf{E}\left\|\sum_{i=1}^n g^i(x)/n\right\|^2 \le M^2$ for all $x \in \mathbb{R}^d$. Using Lemma 4.1 and (4.5), we have

$$\begin{aligned}\mathbf{E}F(\tilde{x}_{k+1}) \le \mathbf{E}F(\tilde{x}_k) &- \frac{\gamma}{2}\mathbf{E}\|\nabla F(\tilde{x}_k)\|^2 - \frac{\gamma}{2}\mathbf{E}\left\|\nabla F(x_k)\right\|^2 \\ &+ \frac{\gamma^3 L^2}{2}\cdot 2(\alpha+\xi)^2 M^2\sum_{l=0}^{k-1}(2\alpha^2)^l + \frac{\gamma^2 L}{2}M^2.\end{aligned}$$

If $\gamma \leq 1/(\beta L)$ where $\beta = 2(\alpha+\xi)^2 \sum_{l=0}^{k-1}(2\alpha^2)^l$, then

$$\mathbf{E}F(\tilde{x}_{k+1}) \leq \mathbf{E}F(\tilde{x}_k) - \frac{\gamma}{2}\mathbf{E}\|\nabla F(\tilde{x}_k)\|^2 - \frac{\gamma}{2}\mathbf{E}\left\|\nabla F(x_k)\right\|^2 + \gamma^2 LM^2,$$

or equivalently

$$\mathbf{E}\left\|\nabla F(x_k)\right\|^2 \leq \frac{2}{\gamma}\left[\mathbf{E}F(\tilde{x}_k) - \mathbf{E}F(\tilde{x}_{k+1})\right] - \mathbf{E}\|\nabla F(\tilde{x}_k)\|^2 + \gamma LM^2.$$

Since $\|x\|^2 \geq 0$ for $x \in \mathbb{R}^d$ and $\min_{l\in[0,k]} \mathbf{E}\|\nabla F(x_l)\|^2 \leq \sum_{l=0}^{k} \mathbf{E}\|\nabla F(x_l)\|^2/(k+1)$, we have

$$\min_{l\in[0,k]} \mathbf{E}\|\nabla F(x_l)\|^2 \leq \frac{2}{\gamma(k+1)}\left[\mathbf{E}F(\tilde{x}_0) - \mathbf{E}F(\tilde{x}_{k+1})\right] + \gamma LM^2.$$

By the cancelations in the telescopic series and by the fact that $F(x) \geq F(x^\star)$ for $x \in \mathbb{R}^d$ and $\tilde{x}_0 = x_0$, we complete the proof.

4.B.6 Proof of Theorem 4.6

We can rewrite the error compensation algorithm (4.18) with $g^i(x_k) = \nabla F^i(x_k)$ and $A_k^i = I - \gamma\nabla^2 F^i(x_k)$ equivalently as Equation (4.36) with $\eta_k = 0$. By the triangle inequality for the Euclidean norm,

$$\begin{aligned}\|\tilde{x}_{k+1} - x^\star\| \leq \|\tilde{x}_k - x^\star - \gamma\nabla F(\tilde{x}_k)\| + \gamma\|\nabla F(\tilde{x}_k) - \nabla F(x_k)\| \\ + \gamma^2 \frac{1}{n}\sum_{i=1}^{n} \|\nabla^2 F^i(x_k) e_k^i\|.\end{aligned}$$

If $\gamma = 2/(\mu + L)$, by the fact that $F(\cdot)$ is L-smooth and that $\tilde{x}_k - x_k = -\gamma\sum_{i=1}^n e_k^i/n$

$$\|\tilde{x}_{k+1} - x^\star\| \leq \rho\|\tilde{x}_k - x^\star\| + \gamma^2 L\epsilon + \gamma^2 \frac{1}{n}\sum_{i=1}^{n} \|\nabla^2 F^i(x_k) e_k^i\|,$$

where $\rho = 1 - 2/(\kappa+1)$ and $\kappa = L/\mu$. Since each $F^i(\cdot)$ is L-smooth, $\nabla^2 F^i(x) \preceq LI$ for $x \in \mathbb{R}^d$, and $\|e_k^i\| \leq \epsilon$, we have

$$\|\tilde{x}_{k+1} - x^\star\| \leq \rho\|\tilde{x}_k - x^\star\| + 2\gamma^2 L\epsilon.$$

By recursively applying this main inequality,

$$\|\tilde{x}_k - x^\star\| \leq \rho^k\|\tilde{x}_0 - x^\star\| + \frac{2\gamma^2 L}{1-\rho}\epsilon.$$

Using the triangle inequality and the fact that $\|x_k - \tilde{x}_k\| \leq \gamma\epsilon$, we can conclude that

$$\|x_k - x^\star\| \leq \|\tilde{x}_k - x^\star\| + \|x_k - \tilde{x}_k\| \leq \rho^k\|\tilde{x}_0 - x^\star\| + \left(\frac{2\gamma^2 L}{1-\rho} + \gamma\right)\epsilon.$$

Since $\tilde{x}_0 = x_0$, the proof is complete.

4.B.7 Proof of Theorem 4.7

We can write the algorithm in Equation (4.18) equivalently as

$$\tilde{x}_{k+1} = \tilde{x}_k - \gamma \left[\nabla F(x_k) + \eta_k\right] - \gamma \frac{1}{n}\sum_{i=1}^{n}(A_k^i - I)e_k^i, \tag{4.36}$$

where

$$\tilde{x}_k = x_k - \gamma \frac{1}{n}\sum_{i=1}^{n} e_k^i, \quad \text{and}$$

$$\eta_k = \frac{1}{n}\sum_{i=1}^{n}\left[g^i(x_k) - \nabla F^i(x_k)\right].$$

By Lemma 4.1, the bounded gradient assumption and by the definition of the ϵ-compressor, it can be proved that

$$\mathbf{E}\|\eta_k\|^2 \leq \sigma^2, \quad \text{and} \tag{4.37}$$

$$\|x_k - \tilde{x}_k\|^2 \leq \gamma^2 \sum_{i=1}^{n}\|e_k^i\|^2/n \leq \gamma^2\epsilon^2. \tag{4.38}$$

Proof of Theorem 4.7-a)

Before deriving the main result we prove two lemmas that are need in our analysis.

Lemma 4.6. *Assume that $\|e_k^i\| \leq \epsilon$, and that the Hessian H_k^i satisfies the unbiased and bounded variance assumptions described in Equation* (4.23) *and* (4.24)*. If $\nabla^2 f_i(x) \preccurlyeq LI$ for $x \in \mathbb{R}^d$, then*

$$\mathbf{E}\left\|\gamma\frac{1}{n}\sum_{i=1}^{n} H_k^i e_k^i\right\|^2 \leq 2\gamma^2(\sigma_H^2 + L^2)\epsilon^2, \quad \text{for } k \in \mathbb{N}. \tag{4.39}$$

Proof. By Lemma 4.1, we have

$$\mathbf{E}\left\|\gamma\frac{1}{n}\sum_{i=1}^{n} H_k^i e_k^i\right\|^2 \leq 2\gamma^2\frac{1}{n}\sum_{i=1}^{n}\left(\mathbf{E}\|[H_k^i - \nabla^2 F^i(x_k)]e_k^i\|^2 + \mathbf{E}\|\nabla^2 F^i(x_k)e_k^i\|^2\right).$$

Since $H_k^i - \nabla^2 F^i(x_k)$ is symmetric, using Equation (4.24), and the fact that $\nabla^2 F^i(x_k) \preccurlyeq L \cdot I$ and that $\|e_k^i\| \leq \epsilon$ yields the result. □

Lemma 4.7. *If $F(\cdot)$ is strongly convex, then for $\theta_1 > 0$*

$$\begin{aligned} -\langle \nabla F(x_k), \tilde{x}_k - x^\star\rangle \leq & -(F(x_k) - F(x^\star)) - \frac{\mu}{4}\|\tilde{x}_k - x^\star\|^2 \\ & + \frac{1}{2}\left(\mu + \frac{1}{\theta_1}\right)\|\tilde{x}_k - x_k\|^2 + \frac{\theta_1}{2}\|\nabla F(x_k)\|^2. \end{aligned} \tag{4.40}$$

Proof. By using the strong convexity inequality in Equation (2.4) with $x = x_k$ and $y = x^\star$ we have

$$-\langle \nabla F(x_k), x_k - x^\star \rangle \leq -(F(x_k) - F(x^\star)) - \frac{\mu}{2}\|x_k - x^\star\|^2.$$

Using the fact that $\|x+y\|^2 \leq 2\|x\|^2 + 2\|y\|^2$ with $x = x_k - x^\star$ and $y = \tilde{x}_k - x_k$, we have

$$-\|x_k - x^\star\|^2 \leq -\frac{1}{2}\|\tilde{x}_k - x^\star\|^2 + \|x_k - \tilde{x}_k\|^2.$$

Combining these inequalities yields

$$-\langle \nabla F(x_k), x_k - x^\star \rangle \leq -(F(x_k) - F(x^\star)) - \frac{\mu}{4}\|\tilde{x}_k - x^\star\|^2 + \frac{\mu}{2}\|x_k - \tilde{x}_k\|^2. \tag{4.41}$$

Next, by Lemma 4.3

$$-\langle \nabla F(x_k), \tilde{x}_k - x_k \rangle \leq \frac{1}{2\theta_1}\|x_k - \tilde{x}_k\|^2 + \frac{\theta_1}{2}\|\nabla F(x_k)\|^2, \tag{4.42}$$

for $\theta_1 > 0$. Summing Equation (4.41) and (4.42) completes the proof. □

Now, we prove the main result. By using the L-smoothness of $F(\cdot)$ and Equation (4.36) with A_k^i defined by Equation (4.22) we have

$$\begin{aligned} F(\tilde{x}_{k+1}) \leq F(\tilde{x}_k) - \gamma\langle \nabla F(\tilde{x}_k), \nabla F(x_k) + \eta_k \rangle + \gamma \left\langle \nabla F(\tilde{x}_k), \gamma \frac{1}{n}\sum_{i=1}^n H_k^i e_k^i \right\rangle \\ + \frac{L\gamma^2}{2}\left\| \nabla F(x_k) + \eta_k - \gamma\frac{1}{n}\sum_{i=1}^n H_k^i e_k^i \right\|^2. \end{aligned}$$

By the unbiased property of the stochastic gradient in Equation (4.11), and by applying Lemma 4.3 with $\theta = 1$ and Lemma 4.1 we get

$$\begin{aligned} \mathbf{E}F(\tilde{x}_{k+1}) \leq \mathbf{E}F(\tilde{x}_k) - \gamma\mathbf{E}\langle \nabla F(\tilde{x}_k), \nabla F(x_k) \rangle + \left(\frac{\gamma}{2} + \frac{3L\gamma^2}{2}\right)\mathbf{E}\|\nabla F(\tilde{x}_k)\|^2 \\ + \frac{3L\gamma^2}{2}\mathbf{E}\|\eta_k\|^2 + \left(\frac{\gamma}{2} + \frac{3L\gamma^2}{2}\right)\mathbf{E}\left\| \gamma\frac{1}{n}\sum_{i=1}^n H_k^i e_k^i \right\|^2. \end{aligned}$$

Since each $F^i(\cdot)$ is L-smooth, $\nabla^2 F^i(x) \preceq L \cdot I$ for $x \in \mathbb{R}^d$. Applying the bounds in Equation (4.37) and (4.39) yields

$$\begin{aligned} \mathbf{E}F(\tilde{x}_{k+1}) \leq \mathbf{E}F(\tilde{x}_k) - \gamma\mathbf{E}\langle \nabla F(\tilde{x}_k), \nabla F(x_k) \rangle \\ + \left(\frac{\gamma}{2} + \frac{3L\gamma^2}{2}\right)\mathbf{E}\|\nabla F(x_k)\|^2 + T, \end{aligned}$$

where $T = 3L\gamma^2\sigma^2/2 + (1+3L\gamma)(\sigma_H^2 + L^2)\gamma^3\epsilon^2$. Using that

$$-2\langle x, y\rangle = -\|x\|^2 - \|y\|^2 + \|x-y\|^2 \quad \text{for all } x, y \in \mathbb{R}^d$$

we have

$$\begin{aligned}\mathbf{E}F(\tilde{x}_{k+1}) \leq \mathbf{E}F(\tilde{x}_k) - \left(\frac{\gamma}{2} - \frac{3L\gamma^2}{2}\right)\mathbf{E}\|\nabla F(x_k)\|^2 \\ + \frac{\gamma}{2}\mathbf{E}\|\nabla F(\tilde{x}_k) - \nabla F(x_k)\|^2 + T.\end{aligned}$$

By the Lipschitz continuity assumption of $\nabla F(\cdot)$, and by (4.38),

$$\mathbf{E}F(\tilde{x}_{k+1}) \leq \mathbf{E}F(\tilde{x}_k) - \left(\frac{\gamma}{2} - \frac{3L\gamma^2}{2}\right)\mathbf{E}\|\nabla F(x_k)\|^2 + \bar{T},$$

where $\bar{T} = T + L^2(\gamma^3/2)\epsilon^2$. By rearranging the terms and recalling that $\gamma < 1/(3L)$ we get

$$\mathbf{E}\|\nabla F(x_k)\|^2 \leq \frac{2}{\gamma}\frac{1}{1-3L\gamma}\left(\mathbf{E}F(\tilde{x}_k) - \mathbf{E}F(\tilde{x}_{k+1}) + \bar{T}\right).$$

Since $\min_{l\in[0,k]}\mathbf{E}\|\nabla F(x_l)\|^2 \leq \sum_{l=0}^k \mathbf{E}\|\nabla F(x_l)\|^2/(k+1)$, we have

$$\min_{l\in[0,k]}\mathbf{E}\|\nabla F(x_l)\|^2 \leq \frac{1}{k+1}\frac{2}{\gamma}\frac{1}{1-3L\gamma}\left(\mathbf{E}F(\tilde{x}_0) - \mathbf{E}F(\tilde{x}_{k+1})\right) + \frac{2}{\gamma}\frac{1}{1-3L\gamma}\bar{T}.$$

By the fact that $e_0 = 0$ (i.e. $\tilde{x}_0 = x_0$), that $F(x) \geq F(x^\star)$ for $x \in \mathbb{R}^d$ we complete the proof.

Proof of Theorem 4.7-b)

From Equation (4.36) with A_k^i defined by Equation (4.22) we have

$$\begin{aligned}\|\tilde{x}_{k+1} - x^\star\|^2 = \|\tilde{x}_k - x^\star\|^2 - 2\gamma\langle\nabla F(x_k) + \eta_k, \tilde{x}_k - x^\star\rangle \\ + 2\gamma\left\langle\gamma\frac{1}{n}\sum_{i=1}^n H_k^i e_k^i, \tilde{x}_k - x^\star\right\rangle \\ + \gamma^2\left\|\nabla F(x_k) + \eta_k - \gamma\frac{1}{n}\sum_{i=1}^n H_k^i e_k^i\right\|^2.\end{aligned}$$

By the unbiasedness of the stochastic gradient described in Equation (4.11), by Lemma 4.1, by Lemma 4.3 with $\theta = \mu/2$ and by the bound in Equation (4.37) we have

$$\begin{aligned}\mathbf{E}\|\tilde{x}_{k+1} - x^\star\|^2 \leq \left(1 + \frac{\mu\gamma}{2}\right)\mathbf{E}\|\tilde{x}_k - x^\star\|^2 - 2\gamma\mathbf{E}\langle\nabla F(x_k), \tilde{x}_k - x^\star\rangle \\ + \left(\frac{2\gamma}{\mu} + 3\gamma^2\right)\mathbf{E}\left\|\gamma\frac{1}{n}\sum_{i=1}^n H_k^i e_k^i\right\|^2 + 3\gamma^2\mathbf{E}\|\nabla F(x_k)\|^2 + 3\gamma^2\sigma^2.\end{aligned}$$

Since each $F^i(\cdot)$ is L-smooth, $\nabla^2 F^i(x) \preceq L \cdot I$ for $x \in \mathbb{R}^d$, we can apply Lemma 4.7. From Equation (4.38) in Lemma 4.6 and Equation (4.40) in Lemma 4.7 with $\theta_1 = \beta/L$ we have

$$\begin{aligned}\mathbf{E}\|\tilde{x}_{k+1} - x^\star\|^2 \leq \mathbf{E}\|\tilde{x}_k - x^\star\|^2 - 2\gamma\mathbf{E}\left(F(x_k) - F(x^\star)\right) \\ + \left(\frac{\beta\gamma}{L} + 3\gamma^2\right)\mathbf{E}\|\nabla F(x_k)\|^2 + \bar{T},\end{aligned}$$

where

$$\bar{T} = \left(\mu + \frac{L}{\beta} + \left(\frac{4}{\mu} + 6\gamma\right)(\sigma_H^2 + L^2)\right)\gamma^3\epsilon^2 + 3\gamma^2\sigma^2.$$

By Lemma 4.4, we have

$$\mathbf{E}\|\tilde{x}_{k+1} - x^\star\|^2 \leq \mathbf{E}\|\tilde{x}_k - x^\star\|^2 - 2\alpha\gamma\mathbf{E}\left(F(x_k) - F(x^\star)\right) + \bar{T}.$$

where $\alpha = 1 - \beta - 3L\gamma$. By recalling that $\gamma < (1-\beta)/(3L)$ and $\beta \in (0,1)$ then

$$\mathbf{E}\left(F(x_k) - F(x^\star)\right) \leq \frac{1}{2\alpha\gamma}\left(\mathbf{E}\|\tilde{x}_k - x^\star\|^2 - \mathbf{E}\|\tilde{x}_{k+1} - x^\star\|^2 + \bar{T}\right).$$

Define $\bar{x}_k = \sum_{l=0}^k x_l/(k+1)$. By the convexity of $F(\cdot)$ and the cancelations in the telescopic series we have

$$\begin{aligned}\mathbf{E}\left(F(\bar{x}_k) - F(x^\star)\right) &\leq \frac{1}{k+1}\sum_{l=0}^{k}\mathbf{E}\left(F(x_l) - F(x^\star)\right) \\ &\leq \frac{1}{k+1}\frac{1}{2\alpha\gamma}\mathbf{E}\|\tilde{x}_0 - x^\star\|^2 + \frac{1}{2\alpha\gamma}\bar{T}.\end{aligned}$$

By the fact that $e_0^i = 0$ (i.e. $\tilde{x}_0 = x_0$), the proof is complete.

Chapter 5

Federated Learning with Error-compensated Compression

Federated learning (FL) has become a popular distributed machine learning framework as it allows multiple nodes to optimize problem parameters without sharing local data [145, 90, 146]. However, communication easily becomes a major bottleneck in FL because the nodes need to iteratively share models which often have millions of parameters [147, 148, 124]. This easily renders FL algorithms infeasible in high-dimensional settings, especially if the nodes are power-constrained or communicate over bandwidth-restricted networks [90, 149, 22].

A popular approach to limit the communication overhead in FL is to increase the local computation on the devices. This is typically done by having the devices perform additional deterministic, stochastic, or proximal gradient updates locally before communicating its local parameters [91, 92, 94]. Experimental results have shown that this approach does accelerate convergence, but it also causes the algorithm to converge to a sub-optimal solution. In fact, it is easily shown that increasing the number of local gradient/proximal updates can make the algorithm converge to an increasingly sub-optimal solution [150]. To mitigate this problem, [150] developed `FedSplit`, an FL algorithm based on operator splitting. `FedSplit` adapts Peaceman-Rachford splitting [60] for solving distributed problems, and it enjoys fast linear convergence toward the *exact* optimal solution.

Communication efficiency can also be improved by reducing the number of bits exchanged per transmission round. This can, e.g., be done by compressing the information using, for example, *sparsification* [103, 115] or *quantization* [22, 82, 97]. Even though these compression operators have improved the communication efficiency of distributed optimization algorithms, they generally suffer in terms of solution accuracy. However, recent work has illustrated how the solution accuracy of compressed algorithms can be improved by error-compensation, exploiting a feedback mechanism based on previous compression errors. Error-compensation is shown to enable algorithms with even coarse compression to retain the convergence rate of the full-precision algorithm, and also obtain highly accurate solutions [124, 151, 152].

In this paper we propose `Eco-FedSplit`, a fast communication efficient federated algorithm that does not sacrifice solution accuracy. By using both operator splitting schemes and error-compensated compression, our algorithm guarantees global linear convergence towards a high-accuracy solution. Our

key results prove that error-compensated compression has far superior performance to naive compression, and fully eliminates the accumulation of compression errors. In particular, we demonstrate that error compensation enables our algorithm to obtain an arbitrary solution accuracy as we decrease its tuning parameter λ. Finally, our numerical experiments confirm strong performance of `Eco-FedSplit` on logistic regression problems over benchmark data sets.

5.1 λ-`FedSplit`

Federated learning is a distributed framework for solving finite-sum optimization problems on the form

$$\underset{x\in\mathbb{R}^d}{\text{minimize}} \quad F(x) = \frac{1}{n}\sum_{i=1}^{n} F^i(x), \tag{5.1}$$

where each $F^i(\cdot)$ is an objective function privately known by worker node i. Typically, a master node updates the solution, based on the local variables aggregated from all the worker nodes. In particular, for a given initial value x_0, the master performs the following update

$$x_{k+1} = (1-\lambda)x_k + \lambda\frac{1}{n}\sum_{i=1}^{n} \mathcal{T}^p_{\gamma F^i}(x_k), \tag{5.2}$$

where k is the iteration index and γ is the positive, fixed learning rate. The parameter $\lambda \in [0,1]$ provides the master node flexibility to put some weights on its own previous iterations. This degree of freedom is not typical in federated learning algorithms. However, we show that for error-compensated compression, λ allows us to balance a trade-off between the solution accuracy and convergence rate. In particular, decreasing λ will improve the solution accuracy at the cost of more iterations.

At each iteration k, worker i performs p local iterations by applying the local operator $\mathcal{T}_{\gamma F^i}(\cdot)$ p-times on the parameter x_k. The benchmark federated learning algorithms `FedAvg` [91] and `FedProx` [94] are covered by Equation (5.2) with $\mathcal{T}_{\gamma F^i}(x) = x - \gamma\nabla F^i(x)$ and with $\mathcal{T}_{\gamma F^i}(x) = \mathbf{prox}_{\gamma F^i}(x)$, respectively, and $\lambda = 1$. However, `FedAvg` and `FedProx` do not generally converge to the global optimum, even in the deterministic case that we study. In particular, the stationary/fixed points of these algorithms can be distant from the optimal solution even on simple problems, as shown in [150].

`FedSplit` [150] was recently developed to find the exact optimal solution of the problem. This algorithm relies on classical splitting schemes, and has the following iteration. Each worker i updates its local vector z^i_k based on its private function $F^i(\cdot)$ via:

$$z^i_{k+1} = \mathbf{refl}_{\gamma F^i}(2\bar{z}_k - z^i_k). \tag{5.3}$$

Then, the master aggregates the local vectors from all workers, and performs the following update

$$x_{k+1} = (1-\lambda)x_k + \lambda \bar{z}_{k+1}, \tag{5.4}$$

where $\bar{z}_k = \sum_{i=1}^n z_k^i/n$ and $\lambda \in [0,1]$ is the tuning parameter. Note that when $n = 1$, `FedSplit` reduces to Douglas-Rachford splitting [153] for $\lambda = 1/2$ and to Peaceman-Rachford splitting [60] for $\lambda = 1$.

The linear convergence of `FedSplit` for $\lambda = 1$ is proved in [150]. For the purposes of this paper, it will be useful to have the freedom to consider any $\lambda \in (0,1]$. We begin by establishing a similar linear convergence result in this case.

Theorem 5.1. *Consider `FedSplit` in* (5.3) *and* (5.4) *to solve Problem* (5.1)*, where each $f_i(\cdot)$ is μ-strongly convex and L-smooth. If $\gamma = 1/\sqrt{\mu L}$ and $x_0 = z_0^1 = \ldots = z_0^n$, then*

$$\mathbf{w}_{k+1} \leq A^k \mathbf{w}_0, \quad \textit{where} \quad A := \begin{bmatrix} 1-\lambda & \rho/\sqrt{n} \\ 0 & \rho \end{bmatrix}, \tag{5.5}$$

$\lambda \in (0,1]$*,* $\rho = 1 - 2/(\sqrt{L/\mu}+1)$*, and*

$$\mathbf{w}_k = \begin{bmatrix} \|x_k - x^\star\| \\ \|z_k - z^\star\| \end{bmatrix} \tag{5.6}$$

for $z = (z^1, z^2, \ldots, z^n) \in \mathbb{R}^{d\cdot n}$*.*

Proof. See Appendix 5.A. □

Theorem 5.1 shows that `FedSplit` with $\lambda \in (0,1]$ converges linearly towards the exact optimum, unlike federated learning algorithms such as `FedAvg` and `FedProx`. When $\lambda = 1$, our result recovers the `FedSplit` convergence presented in Theorem 1 with $b = 0$ in [150]. However, the convergence rate of `FedSplit` is also affected by the tuning parameter λ. In particular, decreasing λ slows down the convergence rate of `FedSplit`. However, we show that when we compress the local variables with error-compensation, decreasing λ allows us to achieve better solution accuracy.

5.2 Direct Compression - Limitations

To limit the communication in `FedSplit`, we need to compress the communicated information. Since in `FedSplit` z^i is the variable that is communicated by each worker, a natural compressed version of `FedSplit` is to have the master perform the following update at each iteration k:

$$x_{k+1} = (1-\lambda)x_k + \lambda \bar{z}_{k+1}, \tag{5.7}$$

where $\bar{z}_{k+1}$ is updated according to

$$\begin{aligned} \bar{z}_{k+1} &= \frac{1}{n}\sum_{i=1}^{n} Q(z^i_{k+1}), \quad \text{and} \\ z^i_{k+1} &= \mathbf{refl}_{\gamma F^i}(2\bar{z}_k - z^i_k), \end{aligned} \tag{5.8}$$

where $Q(\cdot)$ is a compression operator. To make the conclusions of our analysis broad we consider a general class compressors.

Definition 5.1 (Bounded Error Compression)**.** The operator $Q : \mathbb{R}^d \to \mathbb{R}^d$ is an *ϵ-compressor* if there exists a positive constant ϵ such that $\|Q(v) - v\| \leq \epsilon$ for all $v \in \mathbb{R}^d$.

For compression operators to satisfy Definition 5.1, they need only to be bounded error magnitude. Here, $\epsilon > 0$ quantifies the precision of the compression operator. In particular, a compression with a large ϵ is coarse/imprecise and a compression with a small ϵ is precise (e.g. $Q(v) = v$ in the extreme case when $\epsilon = 0$). Most popular compressors in machine learning such as the sign compression [97], the Top-K sparsification [115] and the sparsification together with quantization [22] are in fact the ϵ-compressor if the full-precision vector has a bounded norm.

Our next result characterizes the linear convergence of `FedSplit` with the ϵ-compressor in Equation (5.7).

Theorem 5.2. *Consider compressed `FedSplit` in* (5.7) *and* (5.8) *to solve Problem* (5.1)*, where each $F^i(\cdot)$ is μ-strongly convex and L-smooth. Also, let $\mathbf{refl}_{I_E}(z) = 2\bar{z}\cdot\mathbf{1} - z$ be non-expansive. If $\gamma = 1/\sqrt{\mu L}$ and $x_0 = z^1_0 = \ldots = z^n_0$, then*

$$\mathbf{w}_{k+1} \leq A\mathbf{w}_k + \lambda\epsilon \cdot [1, 0]^T, \tag{5.9}$$

where $\lambda \in (0, 1]$, $\rho = 1 - 2/(\sqrt{L/\mu} + 1)$, and where A and $\mathbf{w}_k$ are defined in (5.5) *and* (5.6)*. In addition, we have*

$$\limsup_{k\to\infty} \|x_k - x^\star\| \leq \epsilon.$$

Proof. See Appendix 5.B. □

Theorem 5.2 establishes a convergence bound on the compressed `FedSplit`. In particular, the convergence rate in Equation (5.9) has two parts. The first part is the linear rate to the exact optimal solution, similar to the non-compressed `FedSplit` in Theorem 5.1. The second part is a compression error depending on $\epsilon > 0$ that is not improved as the algorithm progresses. Note that even if we decrease λ the upper bound on $\lim_{k\to\infty} ||x_k - x^\star||$ is not improved.

This is because decreasing λ increases the eigenvalues of the matrix A. In particular, the fixed point of the linear dynamical system in Equation (5.9) is

$$\vec{w}^{\texttt{Fix}} = (I - A)^{-1} \begin{bmatrix} \lambda\epsilon & 0 \end{bmatrix}^T = \begin{bmatrix} \epsilon & 0 \end{bmatrix}^T.$$

This means that no matter how we choose γ or λ, we cannot improve the solution accuracy of compressed `FedSplit`.

The upper bound in Theorem 5.2 is tight as we show next.

Proposition 5.1. *There exists a federated learning problem on the form of* (5.1) *such that* $\lim_{k\to\infty} ||x_k - x^\star|| = \epsilon$.

We can prove the proposition with the following example.

Example 5.1 (Lower Bound)**.** Consider Problem in (5.1) with $i = 1$, $d = 1$, and $F^1(x) = (\mu/2)x^2$. Let x_k, $\bar{z}_k$, and z_k^i be the iterates of compressed `FedSplit` given in (5.7) and (5.8) where $Q(\cdot)$ is the ϵ-compression

$$Q(z) = \begin{cases} z - \epsilon \cdot z/|z| & \text{if } z \neq 0 \\ \epsilon & \text{otherwise.} \end{cases}$$

Suppose also that $z^k = x^k$, $\lambda \in (0, 1]$, $\gamma > 0$ and $x_0 > \theta\epsilon$ where $\theta = 1 + 1/(\mu\gamma)$. Then for all $k \in \mathbb{N}$, we have that

$$\textbf{\textit{refl}}_{\gamma F}(2x_k - z_k) = \frac{1}{1 + \mu\gamma} x_k,$$

and

$$\begin{aligned} |x_{k+1} - x^\star| &= \rho|x_k| + \lambda\epsilon \\ &= \rho^{k+1}|x_0| + \lambda\epsilon \sum_{i=0}^{k} \rho^i \\ &= \rho^{k+1}|x_0| + \lambda\epsilon \frac{1 - \rho^{k+1}}{1 - \rho} \\ &= \rho^{k+1}\left(|x_0| - \theta\epsilon\right) + \theta\epsilon \\ &\geq \epsilon, \end{aligned}$$

where $\rho = 1 - \lambda + \lambda/(1 + \mu\gamma)$ and $x^\star = 0$. The last inequality follows from the fact that $\theta \geq 1$.

These results show that the solution accuracy of compressed `FedSplit` can never be better than the compression precision ϵ. In the next section we illustrate how we can achieve arbitrarily good solution accuracy no matter how large the compression precision ϵ is.

5.3 Eco-FedSplit

We now illustrate how error compensation enables FedSplit to achieve significant solution accuracy improvements. We call this algorithm Eco-FedSplit, and describe it as follows. The master receives the compensated vectors from all worker nodes, and performs the iteration

$$x_{k+1} = (1-\lambda)x_k + \lambda \bar{z}_{k+1}, \tag{5.10}$$

while each worker updates the compression error e_k^i by the following iteration: given $e_0^i = 0$ for $i = 1, 2, \ldots, n$,

$$\begin{aligned} \bar{z}_{k+1} &= \frac{1}{n}\sum_{i=1}^{n} Q(z_{k+1}^i + (1-\lambda)e_k^i), \\ z_{k+1}^i &= \mathbf{refl}_{\gamma F^i}(2\bar{z}_k - z_k^i), \quad \text{and} \\ e_{k+1}^i &= (z_{k+1}^i + (1-\lambda)e_k^i) - Q(z_{k+1}^i + (1-\lambda)e_k^i). \end{aligned} \tag{5.11}$$

To understand why Eco-FedSplit achieves higher solution accuracy than compressed FedSplit, we compare the closed form of both algorithms. On the one hand, compressed FedSplit (5.7) has the following equivalent closed-form

$$x_k = (1-\lambda)^k x_0 + \lambda \sum_{l=0}^{k} (1-\lambda)^{k-l}(\tilde{z}_l + \tilde{e}_l),$$

where $\tilde{z}_k = (1/n)\sum_{i=1}^n z_k^i$, $\tilde{e}_k = (1/n)\sum_{i=1}^n e_k^i$ and $e_k^i = Q(z_k^i) - z_k^i$ for $i = 1, 2, \ldots, n$. On the other hand, Eco-FedSplit (5.10) can be written equivalently as

$$x_k = (1-\lambda)^k x_0 + \lambda \sum_{l=0}^{k} (1-\lambda)^{k-l}\tilde{z}_l + \lambda \tilde{e}_k.$$

Notice that Eco-FedSplit fully avoids the accumulations of previous compression errors. In fact, our algorithm can recover the solution with high accuracy by properly adjusting the tuning parameter λ. We illustrate this below:

Theorem 5.3. *Consider Eco-FedSplit in* (5.10) *and* (5.11) *to solve Problem* (5.1)*, where each $F^i(\cdot)$ is μ-strongly convex and L-smooth. Also, let $\mathbf{refl}_{I_E}(z) = 2\bar{z}\cdot\mathbf{1} - z$ be non-expansive. If $\gamma = 1/\sqrt{\mu L}$ and $x^0 = z_0^1 = \ldots = z_0^n$, then*

$$\mathbf{w}_{k+1} \leq A\mathbf{w}_k + \lambda^2 \epsilon \cdot [1, 0]^T \tag{5.12}$$

where $\lambda \in (0, 1]$, $\rho = 1 - 2/(\sqrt{L/\mu} + 1)$, and where A and $\mathbf{w}_k$ are defined in (5.5) *and* (5.6)*. In addition,*

$$\limsup_{k\to\infty} \|x_k - x^\star\| \leq \lambda \cdot \epsilon.$$

Proof. See Appendix 5.C. □

Theorem 5.3 shows that `Eco-FedSplit` converges to the neighborhood of the optimal solution with the same linear rate as the full-precision and compressed `FedSplit`. Similarly as Theorem 5.2 for compressed `FedSplit`, the convergence bound has the residual term due to the precision of compression ϵ. Compared to compressed `FedSplit`, this upper bound on $\lim_{k\to\infty} ||x_k - x^\star||$ for `Eco-FedSplit` can be made arbitrarily small by reducing λ. For instance, `Eco-FedSplit` obtains the approximate solution with higher accuracy than compressed `FedSplit` when $\lambda < 1$, and with the same accuracy as full-precision `FedSplit` when λ is close to zero. `Eco-FedSplit` with the small λ guarantees significant solution improvements at the cost of the slower convergence rate. This highlights the trade-off between the solution accuracy and the convergence speed for `Eco-FedSplit`, similarly for error-compensated distributed gradient algorithms in [152].

5.4 Experimental Results

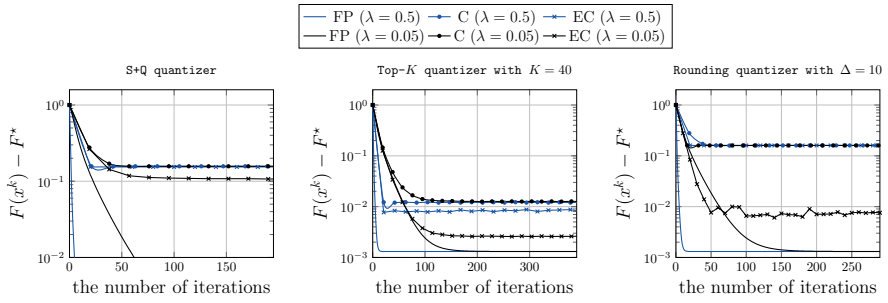
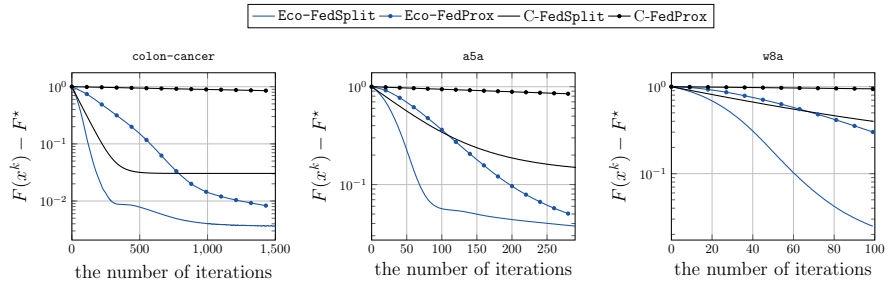
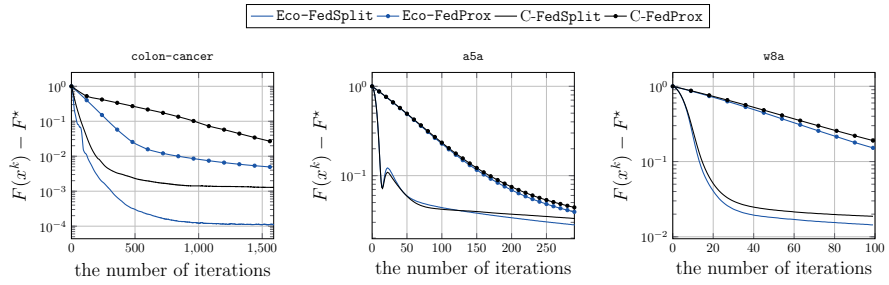


Figure 5.1: Convergence of `FedSplit` with full precision (FP), compression (C) and error compensation (EC) on logistic regression (5.13) with $\sigma = 10^{-2}$ over the synthetic data with $m = 10,000$ and $d = 100$. We evaluated the algorithm using the sparsification with quantization (left), the deterministic sparsification with $T = 40$ (middle), and the rounding quantizer with $\Delta = 10$ (right).

We illustrate the superior convergence properties of `Eco-FedSplit` compared to other compressed federated algorithms, in terms of both the speed and solution accuracy. We evaluated the performance of all communication-efficient federated learning methods on the regularized logistic regression prob-



(a) Sign quantizer



(b) Deterministic sparsification with $T = 0.3 \cdot d$

Figure 5.2: Convergence of `FedSplit` and `FedProx` with compression (C) and error compensation (Eco).

lem

$$\underset{x\in\mathbb{R}^d}{\text{minimize}} \quad \sum_{i=1}^{m} \log(1 + \exp(-b^i \cdot \langle a^i, x\rangle)) + \frac{\sigma}{2}\|x\|^2, \tag{5.13}$$

where σ is a positive regularization parameter, and $(a^1, b^1), (a^2, b^2), \ldots, (a^m, b^m)$ is the collection of data points with $a^i \in \mathbb{R}^d$ is the feature vector and $b^i \in \{-1, 1\}$ is its associated class label. Throughout all the experiments, we normalized each feature vector by its Euclidean norm ($\|a^i\| = 1$ for $i = 1, 2, \ldots, m$), and set 5 workers. We implemented `FedProx` and `FedSplit`, and popular compressors in Python: the rounding quantizer [154, Equation (15)], the sign quantizer [29, Section 6.1], the deterministic sparsification [104, Definition 2], and the sparsification together with quantization [104, Definition 4]. The proximal operators were also solved by using `CVXPY` with the `SCS` solver using a convergence tolerance at $\epsilon = 10^{-2}$.

5.4.1 Synthetic Data

We study the effect of tuning parameters on the performance of federated learning methods over synthetic data with $m = 10,000$ and $d = 100$. We randomly generated each element of the true solution $x^\star$ and of each feature vector a^i according to $\mathcal{N}(0,1)$, and then we set the class label via $b^i = \text{sign}\left(\text{sigmoid}(\langle a^i, x^\star\rangle)\right)$, where the sigmoid function is

$$\text{sigmoid}(z) = 1/(1+\exp(-z)).$$

We also set $\sigma = 10^{-2}$ and $\gamma = 0.1$.

Figure 5.1 suggests that the convergence speeds of all `FedSplit` algorithms decrease when we lower λ. No matter how small λ is, compressed `FedSplit` cannot obtain the solution with high accuracy. Unlike direct compression, error compensation enables `FedSplit` to gain significant accuracy improvements even if the compressors is very coarse. Especially when $\lambda = 0.05$ and the rounding quantizer with $\Delta = 10$ is used, the solution accuracy from `Eco-FedSplit` is higher than compressed `FedSplit` by more than an order of magnitude (see Figure 5.1). While saving communication costs, error-compensated compression also obtains higher accurate solutions than direct compression.

5.4.2 Benchmark Data

We compared the performance of direct compression and error-compensated compression on `Fedsplit` and `Fedprox`. over benchmark LIBSVM data [155]. We also set $\sigma = 10^{-1}$, $\lambda = 10^{-2}$, and $\gamma = 0.06, 0.1$ and 5 for `a5a`, `w8a` and `colon-cancer`, respectively.

`FedSplit` has faster convergence than `FedProx` when both direct and error-compensated compression are used. Compared to direct compression, error compensation gains significant accuracy improvements for `FedProx` and `FedSplit` (see Figure 5.2a and 5.2b). `Eco-FedSplit` outperforms other federated learning methods, in terms of both the convergence speed and the solution accuracy. When training over `colon-cancer`, `Eco-FedSplit` obtains a higher accurate solution than `Eco-FedProx` by an order of magnitude for the sign quantizer and two orders of magnitude for the deterministic sparsification with $T = 0.3 \cdot d$. To reach target accuracy at 10^{-1}, iteration counts for error-compensated `FedProx` require more than twice those for `Eco-FedSplit` to solve the problems on `a5a` and `w8a`. Also notice that compressed `FedSplit` still outperforms compressed `FedProx`. In particular, compressed `FedSplit` reaches the optimal solution with higher accuracy than compressed `FedProx` by more than an order of magnitude to train over `colon-cancer`, as suggested in Figure 5.2a and 5.2b.

Appendix

5.A Proof of Theorem 1

If $\lambda \in (0, 1]$, then by the triangle inequality of the Euclidean norm and by the iteration (5.4)

$$\|x_{k+1} - x^\star\| \leq (1-\lambda)\|x_k - x^\star\| + \lambda\|\bar{z}_{k+1} - \bar{z}^\star\|, \tag{5.14}$$

where the inequality follows from the fixed point assumption, i.e. the point $x^\star = \bar{z}^\star = (1/n)\sum_{i=1}^n (z^i)^\star$ satisfies the relation

$$\mathbf{prox}_{\gamma F^i}(2\bar{z}^\star - (z^i)^\star) = \bar{z}^\star, \quad \text{for } i = 1, 2, \ldots, n.$$

Next, define the vector $z = (z^1, z^2, \ldots, z^n) \in \mathbb{R}^{d\cdot n}$. Let $F^i(\cdot)$ be μ-strongly convex and L-smooth, let $\mathbf{refl}_{I_E}(z) = 2\bar{z}\cdot\mathbf{1} - z$ be non-expansive, and let $\gamma = 1/\sqrt{\mu L}$. Since

$$\|\bar{z}_{k+1} - \bar{z}^\star\| \leq \frac{1}{\sqrt{n}}\|z_k - z^\star\|$$

by Equation (36) with $r^{(t)} = 0$ from [150], we then have

$$\|\bar{z}_k - \bar{z}^\star\| \leq \frac{\rho}{\sqrt{n}}\|z_k - z^\star\| \tag{5.15}$$

where $\rho = 1 - 2/(\sqrt{\kappa}+1)$ and $\kappa = L/\mu$. Plugging this result into (5.14) yields

$$\|x_{k+1} - x^\star\| \leq (1-\lambda)\|x_k - x^\star\| + \frac{\rho}{\sqrt{n}}\|z_k - z^\star\|.$$

By rearranging the terms we get

$$\mathbf{w}_{k+1} \leq A\mathbf{w}_k := \begin{bmatrix} 1-\lambda & \rho/\sqrt{n} \\ 0 & \rho \end{bmatrix} \mathbf{w}_k,$$

where

$$\mathbf{w}_k = \begin{bmatrix} \|x_k - x^\star\| \\ \|z_k - z^\star\| \end{bmatrix}.$$

By recursively applying this inequality, we can prove that

$$\mathbf{w}_k \leq A^k \mathbf{w}_0.$$

Following [156], we guarantee the convergence of the main inequality if $\|A\| < 1$ or equivalently $\det(I - A) > 0$, i.e. we needs to satisfy the condition:

$$\lambda(1-\rho) > 0.$$

This stability condition is thus satisfied when $\rho \in (0, 1)$ and $\lambda > 0$. Finally, we complete the proof.

5.B Proof of Theorem 2

Compressed `FedSplit` in Equation (5.7) can be expressed equivalently as

$$x_{k+1} = (1-\lambda)x_k + \lambda \bar{z}_{k+1} + \lambda e_k, \tag{5.16}$$

where

$$\begin{aligned} e_k &= \frac{1}{n}\sum_{i=1}^{n}[Q(z_{k+1}^i) - z_{k+1}^i], \\ \bar{z}_k &= \frac{1}{n}\sum_{i=1}^{n} Q(z_k^i), \quad \text{and} \\ z_k^i &= \mathbf{refl}_{\gamma F^i}(2\bar{z}_k - z_k^i) \quad \text{for} \quad i = 1, 2, \ldots, n. \end{aligned}$$

Note that from Definition 5.1 we can easily show that

$$\|e_k\| \leq \frac{1}{n}\sum_{i=1}^{n}\|Q(z_{k+1}^i) - z_{k+1}^i\| \leq \epsilon.$$

If $\lambda \in (0, 1]$, then by the triangle inequality of the Euclidean norm and by the update (5.16) we have

$$\|x_{k+1} - x^\star\| \leq (1-\lambda)\|x_k - x^\star\| + \lambda\|\bar{z}_k - \bar{z}^\star\| + \lambda\epsilon, \tag{5.17}$$

where the inequality follows from the fixed point assumption, i.e. the point $x^\star = \bar{z}^\star = (1/n)\sum_{i=1}^{n}(z^i)^\star$ satisfies the relation

$$\mathbf{refl}_{\gamma F^i}(2\bar{z}^\star - (z^i)^\star) = \bar{z}^\star, \quad \text{for } i = 1, 2, \ldots, n.$$

Next, define the vector $z = (z^1, z^2, \ldots, z^n) \in \mathbb{R}^{d\cdot n}$. Let $F^i(\cdot)$ be μ-strongly convex and L-smooth, let $\mathbf{refl}_{I_E}(z) = 2\bar{z}\cdot\mathbf{1} - z$ be non-expansive, and let $\gamma = 1/\sqrt{\mu L}$. Since

$$\|\bar{z}_{k+1} - \bar{z}^\star\| \leq \frac{1}{\sqrt{n}}\|z_k - z^\star\|$$

by Equation (36) with $r^{(t)} = 0$ from [150], we then have Eq. (5.15). By applying this result into (5.17), we have

$$\|x_{k+1} - x^\star\| \leq (1-\lambda)\|x_k - x^\star\| + \frac{\rho}{\sqrt{n}}\|z_k - z^\star\| + \lambda\epsilon.$$

By rearranging the terms we get

$$\mathbf{w}_{k+1} \leq A\mathbf{w}_k + \lambda \begin{bmatrix} \epsilon \\ 0 \end{bmatrix},$$

where

$$A = \begin{bmatrix} 1-\lambda & \rho/\sqrt{n} \\ 0 & \rho \end{bmatrix}, \text{ and } \mathbf{w}_k = \begin{bmatrix} \|x_k - x^\star\| \\ \|z_k - z^\star\| \end{bmatrix}.$$

By following [156], the convergence of this main inequality is guaranteed if $\|A\| < 1$ or equivalently $\det(I - A) > 0$, i.e. the following condition must be satisfied

$$\lambda(1-\rho) > 0.$$

We thus can guarantee the convergence since $\rho \in (0, 1)$ and $\lambda > 0$. In addition,

$$\begin{aligned} \limsup_{k\to\infty} \mathbf{w}_k &\leq (I-A)^{-1}\lambda \begin{bmatrix} \epsilon \\ 0 \end{bmatrix} \\ &= \frac{\lambda}{\lambda(1-\rho)} \begin{bmatrix} 1-\rho & \rho/\sqrt{n} \\ 0 & \lambda \end{bmatrix} \begin{bmatrix} \epsilon \\ 0 \end{bmatrix} \\ &= \begin{bmatrix} \epsilon \\ 0 \end{bmatrix}. \end{aligned}$$

5.C Proof of Theorem 3

Compensated `FedSplit` algorithm in Equation (5.10) and (5.11) can be similarly rewritten on the form

$$\tilde{x}_{k+1} = (1-\lambda)\tilde{x}_k + \lambda \tilde{z}_{k+1}, \tag{5.18}$$

where

$$\begin{aligned} \tilde{z}_k &= \frac{1}{n}\sum_{i=1}^n z_k^i, \\ \tilde{x}_k &= x_k + \lambda \frac{1}{n}\sum_{i=1}^n e_k^i, \\ \bar{z}_{k+1} &= \frac{1}{n}\sum_{i=1}^n Q(z_{k+1}^i + (1-\lambda)e_k^i), \quad \text{and} \\ z_{k+1}^i &= \mathbf{refl}_{\gamma F^i}(2\bar{z}_k - z_k^i). \end{aligned}$$

If $\lambda \in (0, 1]$, then from Definition 5.1 we can easily show that $\|e_k^i\| \leq \epsilon$ for all i, k, and

$$\|\tilde{x}_k - x_k\| \leq \frac{\lambda}{n}\sum_{i=1}^n \left\|e_k^i\right\| \leq \lambda\epsilon. \tag{5.19}$$

Thus, we complete the proof. Next, by the triangle inequality of the Euclidean norm and by the update (5.18) we have

$$\|\tilde{x}_{k+1} - x^\star\| \leq (1-\lambda)\|\tilde{x}_k - x^\star\| + \lambda\|\tilde{z}_k - \tilde{z}^\star\|, \tag{5.20}$$

where the inequality follows from the fixed point assumption, i.e. the point $x^\star = \bar{z}^\star = \tilde{z}^\star = (1/n)\sum_{i=1}^n (z^i)^\star$ satisfies the relation

$$\mathbf{refl}_{\gamma F^i}(2\bar{z}^\star - (z^i)^\star) = \tilde{z}^\star, \quad \text{for } i = 1, 2, \ldots, n.$$

Next, define the vector $z = (z^1, z^2, \ldots, z^n) \in \mathbb{R}^{d\cdot n}$. Let $F^i(\cdot)$ be μ-strongly convex and L-smooth, let $\mathbf{refl}_{I_E}(z) = 2\bar{z}\cdot\mathbf{1} - z$ be non-expansive, and let $\gamma = 1/\sqrt{\mu L}$. Since

$$\|\tilde{z}_{k+1} - \tilde{z}^\star\| \leq \frac{1}{\sqrt{n}}\|z_k - z^\star\|$$

by Equation (36) with $r^{(t)} = 0$ from [150], we then have Eq. (5.15). By plugging this result into (5.20), we have

$$\|\tilde{x}_{k+1} - x^\star\| \leq (1-\lambda)\|\tilde{x}_k - x^\star\| + \frac{\rho}{\sqrt{n}}\|z_k - z^\star\|.$$

By rearranging the terms we get

$$\tilde{\mathbf{w}}_{k+1} \leq A\tilde{\mathbf{w}}_k,$$

where

$$A = \begin{bmatrix} 1-\lambda & \rho/\sqrt{n} \\ 0 & \rho \end{bmatrix}, \text{ and } \tilde{\mathbf{w}}_k = \begin{bmatrix} \|\tilde{x}_k - x^\star\| \\ \|z_k - z^\star\| \end{bmatrix}.$$

By recursively applying this inequality, we can prove that

$$\tilde{\mathbf{w}}_k \leq A^k\tilde{\mathbf{w}}_0. \tag{5.21}$$

Following [156], we guarantee the convergence if $\|A\| < 1$ or equivalently $\det(I - A) > 0$, i.e. we needs to satisfy the condition

$$\lambda(1-\rho) > 0.$$

This stability condition is hence satisfied, since $\rho \in (0,1)$ and $\lambda > 0$. Next, by the triangle inequality of the Euclidean norm and by (5.19)

$$\|x_k - x^\star\| \leq \|\tilde{x}_k - x^\star\| + \|\tilde{x}_k - x_k\| \leq \|\tilde{x}_k - x^\star\| + \lambda\epsilon.$$

Since $e_0^i = 0$ for all $i = 1, 2, \ldots, n$, we can easily show that $\tilde{x}_0 = x_0$, and that the main inequality (5.21) can be equivalently expressed as

$$\mathbf{w}_k \leq A^k\mathbf{w}_0 + \lambda\begin{bmatrix} \epsilon \\ 0 \end{bmatrix},$$

where

$$\mathbf{w}_k = \begin{bmatrix} \|x_k - x^\star\| \\ \|z_k - z^\star\| \end{bmatrix}.$$

Furthermore, we can prove that

$$\limsup_{k\to\infty} \|x_k - x^\star\| \leq \lambda\epsilon.$$

Part II

Flexible Tuning Framework

Chapter 6

Adaptive Compression Framework

The vast size of modern machine learning is shifting the focus on optimization and learning algorithms from centralized to distributed architectures. State-of-the-art models are now typically trained using multiple CPUs or GPUs, and data is increasingly being collected and processed in networks of resource-constrained devices, e.g., IoT devices, smart phones, or wireless sensors. This trend is shifting the bottleneck from the computation to the communication. The shift is particularly striking when learning is performed on energy-constrained devices that communicate over shared wireless channels. Indeed, distributed training is often communication bound since the associated optimization algorithms hinge on frequent transmissions of gradients between nodes. These gradients are typically huge: it is not uncommon for state-of-the-art models to have millions of parameters. To get a sense of the corresponding communication cost, transmitting a single gradient or stochastic gradient using single precision (32 bits per entry) requires 40 MB for a model with 10 million parameters. If we use 4G communications, this means that we can expect to transmit roughly one gradient per second. The huge communication load easily overburdens training even on loosely interconnected clusters and may render federated learning on some IoT or edge devices infeasible.

To alleviate the communication bottleneck, much recent research has focused on compressed gradient methods. These methods achieve communication efficiency by using only the most informative parts of the gradients at each iteration. We may, for example, sparsify the gradient, *i.e.* use only the most significant entries at each iteration and set the rest to be zero [22, 124, 151, 95, 96, 104, 30]. We may also quantize the gradient elements or do some mix of quantization and sparsification [22, 117, 82, 30, 31, 128, 157, 158].

Several of the cited papers have demonstrated huge communication improvements for specific training problems. However, these communication benefits are often realized after a careful tuning of the compression level before training, *e.g.* the number of elements to keep when sparsifying the gradient. We cannot expect there to be a universally good compressor that works well on all problems, as shown by the worst-case communication complexity of any optimization methods in [75]. There is generally a delicate problem-specific balance between compressing too much or too little. Trying to strike the right balance by hyper-parameter tuning is expensive and the resulting tuning parameters will be problem-specific. Moreover, most existing compression schemes are agnostic of the disparate communication costs for different technologies. In contrast, our proposed on-line mechanism adapts the compression

level to each gradient information and each platform-specific communication cost.

Contributions: We consider deterministic and stochastic gradient methods in distributed settings where compressed gradients are communicated at every iteration. We propose a flexible framework for an on-line adaption of the gradient compression level to the problem data and communication technology used. This Communication-aware Adaptive Tuning (CAT) optimally adjusts the compression of each communicated gradient by maximizing the ratio between the guaranteed objective function improvement and the communication cost. The communication cost can be easily adjusted to the technology used, either by analytical models of the communication protocols or through empirical measurements. We illustrate these ideas on three state-of-the-art compression schemes: a) sparsification, b) sparsification with quantization, and c) stochastic sparsification. In all cases, we first derive descent lemmas specific to the compression, relating the function improvement to the tuning parameter. Using these results we can find the tuning that optimizes the communication efficiency measured in the descent direction relative to the given communication costs. Even though most of our theoretical results are for a single node case, we illustrate the efficiency of CAT to all three compression schemes in large-scale experiments in multi-node settings. For the stochastic sparsification we also prove convergence for stochastic gradient methods in the multi-node settings.

Related work on federated learning: Another approach to improve communication efficiency is to increase local computations in hope of reducing the number of iterations, and thereby the number of communication rounds. This is typically done by letting the nodes perform multiple gradient or proximal gradient updates locally before communicating their updated parameters, see, e.g., [91, 93, 92, 159, 160],[94]. By adapting the number of local updates within a communication time interval [161], it is possible to find a good balance between the communication savings and suboptimality guarantees of the solution. In contrast, we focus on adaptive compression, where our CAT framework strikes this balance by adjusting the compression level online, e.g. by optimizing the transmitted bits per iteration. In general, these two adaptive strategies can be combined to obtain further savings without compromising the convergence guarantees. In other words, CAT can be seen as a complement to existing adaptive schemes, addressing another dimension of communication efficiency.

6.1 Background

The main focus of this paper is empirical risk minimization

$$\underset{x\in\mathbb{R}^d}{\text{minimize}}\ F(x) = \frac{1}{|\mathcal{D}|}\sum_{z\in\mathcal{D}} L(x;z)$$

where $\mathcal{D}$ is a set of data points, x is the model parameters and $L(\cdot)$ is a loss function.

6.1.1 Gradient Compression

Consider the standard compressed gradient iteration

$$x_{k+1} = x_k - \gamma_k Q_T(\nabla F(x_k)). \tag{6.1}$$

Here, $Q_T(\cdot)$ is a compression operator and T is a parameter that controls the compression level. The goal is to achieve communication efficiency by using only the most significant information. One of the simplest compression schemes is to sparsify the gradient, *i.e.* we let

$$[Q_T(g)]^j = \begin{cases} g^j & \text{if } j \in I_T(g) \\ 0 & \text{otherwise.} \end{cases} \tag{6.2}$$

where $I_T(g)$ is the index set for the T components of g with largest magnitude. The following combination of sparsification and quantization has been shown to give good practical performance [22]:

$$[Q_T(g)]^j = \begin{cases} ||g||\texttt{sign}(g^j) & \text{if } i \in I_T(g) \\ 0 & \text{otherwise.} \end{cases} \tag{6.3}$$

In this case, we communicate only the gradient magnitude and the sparsity pattern of the gradient. It is sometimes advantageous to use stochastic sparsification. Rather than sending the top T entries of each gradient, we then send T components on average. We can achieve this by setting

$$[Q_{T,p}\,(g)]^j = (g^j/p^j)\xi^j, \tag{6.4}$$

where $\xi^j \sim \texttt{Bernouli}(p^j)$ and $T = \sum_{j=1}^{d} p^j$. Ideally, we would like p^j to represent the magnitude of g^j, so that p^j should be large if $|g^j|$ is large relative to the other entries. There are many heuristic methods to choose p^j. For example, if we set $p^j = |g^j|/\|g\|_q$ with $q = 2$, $q = \infty$, and $q \in (0, \infty]$, then we get, respectively, the stochastic sparsifications in [22] with $s = 1$, the *TernGrad* in [95], and ℓ_q-quantization in [96]. We can also optimize p to reduce the variance error, see [96].

Experimental results have shown huge communication savings by compressed gradient methods in large-scale machine learning [162, 163]. Nevertheless, we can easily create pedagogical examples where they are no more communication efficient than full gradient descent. For sparsification, consider the function $F(x) = ||x||^2/2$. Then, gradient descent with the step-size $\gamma = 1$ converges in one iteration, and thus communicates only d floating points

(one for each element of $\nabla F(x) \in \mathbb{R}^d$) to reach any ϵ-accuracy. On the other hand, T-sparsified gradient descent (where T divides d) needs d/T iterations, which implies d communicated floating points in total. In fact, the sparsified method is even worse since it requires additional $d\log(d)$ communicated bits to indicate the sparsity pattern.

This example shows that the benefits of sparsification cannot be seen on worst-case problems, and that traditional worst-case analysis (*e.g.* [104]) is unable to guarantee the improved communication complexity. Rather, sparsification is useful for exploiting the structure that appears in real-world problems. The key in exploiting this structure is to choose T properly at each iteration. In this paper we describe how to choose T dynamically to optimize the communication efficiency of sparsification.

6.1.2 Communication Cost: Bits, Packets, Energy and Beyond

The compressors discussed above have a tuning parameter T, which controls the sparsity budget of compressed gradient descent. Our goal is to tune T adaptively to optimize the communication efficiency. To explain how this is done, we first need to discuss how to model the communication cost. Let $C(T)$ denote the communication cost per iteration as a function of T, e.g., the total number of transmitted bits. Then, $C(T)$ consists of payload (actual data) and communication overhead. The payload is the number of bits required to communicate the compressed gradient. For the sparsification in Eq. (6.2) and the quantization in Eq. (6.3), the payload consumes, respectively,

$$\begin{aligned} P^{\mathtt{S}}(T) &= T \times (\lceil \log_2(d) \rceil + \mathtt{FPP}) \quad \text{bits} \quad \text{and} \\ P^{\mathtt{SQ}}(T) &= \mathtt{FPP} + T \times \lceil \log_2(d) \rceil \quad \text{bits}, \end{aligned} \tag{6.5}$$

where the $\log_2(d)$ factor comes from indicating T indices in the d-dimensional gradient vector. Here $\mathtt{FPP}$ is our floating point precision, e.g., $\mathtt{FPP} = 32$ or $\mathtt{FPP} = 64$ for, respectively, single or double precision floating-points. Our simplest communication model accounts only for the payload, i.e., $C(T) = P(T)$. We call this the *payload model*. In real-world networks, however, each communication also includes the overhead and set-up costs. A more realistic model is therefore affine $C(T) = c_1 P(T) + c_0$, where $P(T)$ is the payload. Here c_0 is the communication overhead, while c_1 is the cost of transmitting a single payload byte. For example, if we just count transmitted bits ($c_1 = 1$), then a single UDP packet transmitted over the Ethernet requires an overhead of $c_0 = 54 \times 8$ bits and can have a payload of up to 1472 bytes. In the wireless standard IEEE 802.15.4, the overhead ranges from 23-82 bytes, leaving $51-110$ bytes of payload before the maximum packet size of 133 bytes is reached [164]. Another possibility is to use a *packet model*, *i.e.* to have a fixed cost per packet

$$C(T) = c_1 \times \lceil P(T)/P_{\max} \rceil + c_0, \tag{6.6}$$

where $P_{\max}$ is the number of payload bits per packet. The term $\lceil P(T)/P_{\max} \rceil$ counts the number of packets required to send the $P(T)$ payload bits, c_1 is the cost per packet, and c_0 is the cost of initiating the communication. These are just two examples; ideally, $C(T)$ should be tailored to the specific communication standard used, and possibly even estimated from system measurements.

6.1.3 Key Idea: Communication-Aware Adaptive Tuning (CAT)

When communicating the compressed gradients, we would like to use each bit as efficiently as possible. In optimization terms, we would like the objective function improvement for each communicated bit to be as large as possible. In other words, we want to maximize the ratio

$$\texttt{Efficiency}(T) = \frac{\texttt{Improvement}(T)}{C(T)}, \tag{6.7}$$

where $\texttt{Improvement}(T)$ is the improvement in the objective function when we use T-sparsification as the given compressor. We will demonstrate how the value of $\texttt{Improvement}(T)$ can be obtained from novel descent lemmas and derive dynamic sparsification policies which, at each iteration, find the T that optimizes $\texttt{Efficiency}(T)$. We derive optimal T-values for the three compressors and two communication models introduced above. However, the idea is general and can be used to improve the communication efficiency for many other compression algorithms.

We begin by describing how our CAT framework can be applied to sparsified gradient methods. To this end, the following lemma introduces a useful measure $\alpha(T)$ of function value improvement:

Lemma 6.1. *Suppose that $F : \mathbb{R}^d \to \mathbb{R}$ is (possibly non-convex) L-smooth and $\gamma = 1/L$. Then for any $x, x^+ \in \mathbb{R}^d$ with $x^+ = x - \gamma Q_T(\nabla F(x))$ we have*

$$F(x^+) \leq F(x) - \frac{\alpha(T)}{2L}||\nabla F(x)||^2,$$

where $\alpha(T) = ||Q_T(\nabla F(x))||^2/||\nabla F(x)||^2$. Moreover, there are L-smooth functions $F(\cdot)$ for which the inequality is tight for every $T = 1, \ldots, d$.

This lemma is in the category of descent lemmas, which are standard tools to study the convergence for convex and non-convex functions. In fact, Lemma 6.1 generalizes the standard descent lemma for L-smooth functions (see, e.g., in Proposition A.24 in [165]). In particular, if the gradient $\nabla F(x)$ is T-sparse (or $T = d$) then Lemma 6.1 gives the standard descent

$$F(x^+) \leq F(x) - \frac{1}{2L}||\nabla F(x)||^2.$$

In the next subsection, we will use Lemma 6.1 to derive novel convergence rate bounds for sparsified gradient methods, extending many standard results for gradient descent. First, however, we will use $\texttt{Improvement}(T) = \alpha(T)$ to define the following CAT mechanism for dynamic sparsification:

$$\textbf{Step 1: } T_k = \operatorname*{argmax}_{T \in [1,d]} \frac{\alpha_k(T)}{C(T)}, \tag{6.8}$$

$$\textbf{Step 2: } x_{k+1} = x_k - \frac{1}{L} Q_{T_k}(\nabla F(x_k)). \tag{6.9}$$

The algorithm first finds the sparsity budget T_k that optimizes the communication efficiency defined in (6.7), and then performs a standard sparsification using this value of T_k. Since $\|\nabla F(x)\|^2/2L$ is independent of T, we can maximize efficiency by maximizing $\alpha(T)/C(T)$.

To find T^i at each iteration we need to solve the maximization problem in Eq. (6.9). This problem has one dimension, and even a brute force search would be feasible in many cases. As the next two results show, however, the problem has a favourable structure that allows the maximization to be solved very efficiently. The first result demonstrates that the descent always increases with T and is bounded.

Lemma 6.2. *For $g \in \mathbb{R}^d$ the function $\alpha(T) = ||Q_T(g)||^2/||g||^2$ is increasing and concave when extended to the continuous interval $[0, d]$. Moreover, $\alpha(T) \geq T/d$ for all $T \in \{0, \dots d\}$ and there exists an L-smooth function such that*

$$||Q_T(\nabla f(x))||^2/||\nabla f(x)||^2 = T/d \quad \textit{for all } x \in \mathbb{R}^d.$$

Lemma 6.2 results in many consequences in the next section, but first we make another observation:

Proposition 6.1. *Let $\alpha(T)$ be increasing and concave. If $C(T) = \tilde{c}_1 T + c_0$, then $\alpha(T)/C(T)$ is quasi-concave and has a unique maximum on $[0, d]$. When $C(T) = \tilde{c}_1 \lceil T/\tau_{\max} \rceil + c_0$, on the other hand, $\alpha(T)/C(T)$ attains its maximum for a T which is an integer multiple of $\tau_{\max}$.*

This proposition shows that the optimization in Eq. (6.9) is easy to solve. For the affine communication model, one can simply sort the elements in the decreasing order, initialize $T = 1$ and increase T until $\alpha(T)/C(T)$ decreases. In the packet model, the search for the optimal T is even more efficient, as one can increase T in steps of $\tau_{\max}$.

6.1.4 Dynamic Sparsification Benefits in Theory and Practice

Although the CAT framework applies to general communication costs, it is instructive to see what our results say about the communication complexity,

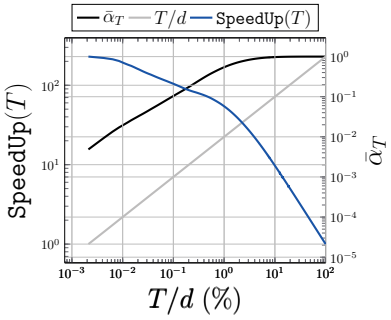
	Deterministic Sparsification			Stochastic Sparsification		
Upper-Bound	μ-convex	convex	nonconvex	μ-convex	convex	nonconvex
No-Compression	$A_\epsilon^{\mathtt{SC}}$	$A_\epsilon^{\mathtt{C}}$	$A_\epsilon^{\mathtt{NC}}$	$B_\epsilon^{\mathtt{SC}}$	$B_\epsilon^{\mathtt{C}}$	$B_\epsilon^{\mathtt{NC}}$
Data-Dependent	$\frac{1}{\bar{\alpha}_T} \cdot A_\epsilon^{\mathtt{SC}}$	$\frac{1}{\bar{\alpha}_T} \cdot A_\epsilon^{\mathtt{C}}$	$\frac{1}{\bar{\alpha}_T} \cdot A_\epsilon^{\mathtt{NC}}$	$\frac{1}{\bar{\omega}_T} \cdot B_\epsilon^{\mathtt{SC}}$	$\frac{1}{\bar{\omega}_T} \cdot B_\epsilon^{\mathtt{C}}$	$\frac{1}{\bar{\omega}_T} \cdot B_\epsilon^{\mathtt{NC}}$
Worst-Case	$\frac{d}{T} \cdot A_\epsilon^{\mathtt{SC}}$	$\frac{d}{T} \cdot A_\epsilon^{\mathtt{C}}$	$\frac{d}{T} \cdot A_\epsilon^{\mathtt{NC}}$	$\frac{d}{T} \cdot B_\epsilon^{\mathtt{SC}}$	$\frac{d}{T} \cdot B_\epsilon^{\mathtt{C}}$	$\frac{d}{T} \cdot B_\epsilon^{\mathtt{NC}}$

Table 6.1: Iteration complexity for T-sparsified (stochastic) gradient descent. We prove these results and analogous results for $S + Q$ compression in the Appendix. For Deterministic Sparsification, $A_\epsilon^{\mathtt{SC}}$, $A_\epsilon^{\mathtt{C}}$, and $A_\epsilon^{\mathtt{NC}}$ are standard upper bounds for gradient descent on iteration counts needed to achieve ϵ-accuracy for, respectively, strongly-convex, convex, and non-convex problems. In particular, $A_\epsilon^{\mathtt{SC}} = \kappa \log(\epsilon_0/\epsilon)$, $A_\epsilon^{\mathtt{C}} = 2LR^2/\epsilon$, $A_\epsilon^{\mathtt{NC}} = 2L\epsilon_0/\epsilon$, where $\kappa = L/\mu$, $\epsilon_0 = F(x^0) - F^\star$, and R is a constant such that $||x^i - x^\star|| \leq R$ (for some optimizer $x^\star$). For Stochastic Sparsification, $B_\epsilon^{\mathtt{SC}}$, $B_\epsilon^{\mathtt{C}}$, and $B_\epsilon^{\mathtt{NC}}$ are standard upper bounds for multi-node stochastic gradient descent on iteration counts needed to achieve ϵ-accuracy (in expected value) for, respectively, strongly-convex, convex, and non-convex problems. In particular, $B_\epsilon^{\mathtt{SC}} = 2(1 + 2\sigma^2/(\mu\epsilon L))(A_\epsilon^{\mathtt{SC}} + \delta)$, $B_\epsilon^{\mathtt{C}} = 2(1 + 2\sigma^2/(\epsilon L))A_\epsilon^{\mathtt{C}}$, and $B_\epsilon^{\mathtt{NC}} = 2(1 + 2\sigma^2/\epsilon)A_\epsilon^{\mathtt{NC}}$, where $\delta = \kappa \log(2)$ and σ^2 is a variance bound of stochastic gradients. The ϵ-accuracy is measured in $\mathbf{E}[F(x) - F(x^\star)]$ for convex problems, and in $\mathbf{E}\|\nabla F(x)\|^2$ otherwise.

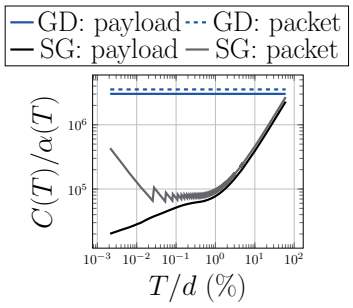
i.e., the number of bits that needs to be communicated to guarantee that a solution is found within an ϵ-accuracy. Table 1 compares the iteration complexity of Gradient Descent (GD) in row 1 and T-Sparsified Gradient Descent (T-SGD) in rows 2 and 3 with constant T for strongly-convex, convex, and non-convex problems. The results for gradient descent are well-known and found in, e.g., [166], while the worst-case analysis is from [104]. The results for T-sparsified gradient descent are derived using Lemma 6.1 instead of the standard descent lemma; see proofs in the supplementary materials.

Comparing rows 1 and 3 in the table, we see that the worst-case analysis does not guarantee any improvements in the number of communicated floating points. Although T-SGD only communicates T out of d gradient entries in each round, we need to perform d/T times more iterations with T-SGD than with SGD, so both of these approaches will need to communicate the same number of floating points. In fact, T-SGD will be worse in terms of communicated bits since it requires $T\lceil \log_2(d) \rceil$ additional bits per iteration to indicate the sparsity pattern.

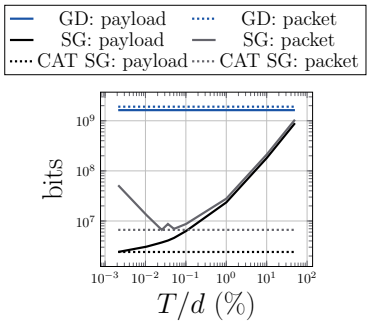
Let us now turn our attention to our novel analysis shown in row 2 of Table



(a) $\bar{\alpha}_T$ and speedup



(b) Hypothetical: Communications to reach ϵ-accuracy



(c) Experiments: Communications (in bits) to reach ϵ-accuracy

Figure 6.1: CAT sparsified gradient descent on the `RCV1` data set.

1. Here, the parameter $\bar{\alpha}_T$ is a lower bound on $\alpha_k(T)$ over every iteration, that is

$$\alpha_k(T) \geq \bar{\alpha}_T \quad \text{for all } k.$$

Unfortunately, $\bar{\alpha}_T$ is not useful for algorithm development: we know from Lemma 6.2 that it can be as low as T/d, and it is not easy to compute a tight data-dependent bound off-line, since $\bar{\alpha}_T$ depends on the iterates produced by the algorithm. However, $\bar{\alpha}_T$ explains why gradient sparsification is communication efficient. In practice, only few top entries cover the majority of the gradient energy, so $\alpha_k(T)$ grows rapidly for small values of T and is much larger than T/d.

To illustrate the benefits of sparsification, let us look at the concrete example of logistic regression on the standard benchmark data set `RCV1` (with $d = 47,236$ and $697,641$ data points). Figure 6.1a depicts $\bar{\alpha}_T$ computed after running 1000 iterations of gradient descent and compares it to the worst

case bound T/d. The results show a dramatic difference between these two measures. We quantify this difference by the ratio

$$\texttt{SpeedUp}(T) = \frac{d}{T} \Bigg/ \frac{1}{\bar{\alpha}_T} = \frac{\bar{\alpha}_T}{T/d}.$$

Note that this measure is the ratio between rows 2 and 3 in Table 1, and hence tells us the hypothetical speedup by sparsification, i.e., the ratio between the number of communicated floating points needed by GD and T-SGD to reach ϵ-accuracy. The figure shows a dramatic speedup; for small values of T, the speed-up is of three orders of magnitude (we confirm this in experiments below).

Interestingly, the speedup decreases with T and is maximized at $T = 1$. This happens because doubling T doubles the number of communicated bits, while the additional descent is often less significant. Thus, an increase in T worsens communication efficiency. This suggests that we should always take $T = 1$ if the communication efficiency in terms of bits is optimized without considering overhead. In the context of the dynamic algorithm in Eq. (6.9), this leads to the following result:

Proposition 6.2. *Consider the dynamic sparsified gradient algorithm in Eq.* (6.9) *with $C(T) = P^{\texttt{S}}(T)$ given by Eq. (6.5). Then, the maximization problem* (6.9) *has the solution $T_k = 1$ for all k.*

Figures 6.1b and 6.1c depict, respectively, the hypothetical and true values of the total number of bits needed to reach an ϵ-accuracy for different communication models. In particular, Figure 6.1b depicts the ratio $C(T)/\bar{\alpha}_T$ (compare with Table 1) and Figure 6.1c depicts the experimental results of running T-SGD for different values of T. We consider: a) the payload model with $C(T) = P^{\texttt{S}}(T)$ (dashed lines) and b) the packet model in Eq. (6.6) with $c_1 = 128$ bytes, $c_0 = 64$ bytes and $P_{\max} = 128$ bytes (solid lines). In both cases, the floating point precision is $\texttt{FPP} = 64$. We compare the results with GD (blue lines) with payload $d \times \texttt{FPP}$ bits per iteration. As expected, if we ignore overheads, then $T = 1$ is optimal and the improvement compared to GD are of three orders of magnitude. For the packet model, there is a delicate balance between choosing T too small and too big. For general communication models it is difficult to find the right value of T *a priori*, and the costs of choosing a bad T can be of many orders of magnitude. To find a good T we could do a hyper-parameter search, e.g. by first estimating $\bar{\alpha}_T$ from data and then by using it to find the optimal T. However, this will be expensive and moreover $\bar{\alpha}_T$ might not be a good estimate of $\alpha_k(T)$ we get at each iteration. In contrast, our CAT framework finds the optimal T at each iteration without any hyper-parameter optimization. In Figure 6.1c we show the number of communicated bits needed to reach ϵ-accuracy with our algorithm. The results show that for both communication models, our algorithm achieves the same communication efficiency as if we would choose the optimal T.

6.2 Dynamic Sparsification + Quantization

We now describe how our CAT framework can improve the communication efficiency of compressed gradient methods that use sparsification combined with quantization, i.e., using $Q_T(\cdot)$ in Equation (6.3). As before, our goal is to choose T_k dynamically by maximizing the communication efficiency per iteration defined in (6.7). This selection can be performed based on the following descent lemma.

Lemma 6.3. *Suppose that $F : \mathbb{R}^d \to \mathbb{R}$ is (possibly non-convex) L-smooth. Then for any $x, x^+ \in \mathbb{R}^d$ with $x^+ = x - \gamma Q_T(\nabla F(x))$ where $Q_T(\cdot)$ is as defined in Eq. (6.3) and $\gamma = \sqrt{\beta(T)}/(\sqrt{T}L)$ then*

$$F(x^+) \leq F(x) - \frac{\beta(T)}{2L}||\nabla F(x)||^2,$$

where $\beta(T) = \langle \nabla F(x), Q_T(\nabla F(x))\rangle^2 / \big(T \cdot \|\nabla F(x)\|^4\big)$.

Since this compression operator affects the descent differently from sparsification, this lemma differs from Lemma 6.1, e.g, in terms of the step-size and descent measure ($\beta(T)$ vs. $\alpha(T)$). Unlike $\alpha(T)$ in Lemma 6.1, $\beta(T)$ does not converge to 1 as T goes to d. In fact, $\beta(T)$ is not even an increasing function, and $Q_T(g)$ does not converge to g when T increases. Nevertheless, $\langle \nabla F(x), Q_T(\nabla F(x))\rangle^2$ is non-negative, increasing and concave. Under the affine communication model, $T \times C(T) = \tilde{c}_0 T^2 + c_1 T$ is non-negative and convex, which implies that $\beta(T)/C(T)$ is quasi-concave. The optimal T can then be efficiently found similarly to what was done for the CAT-sparsification. Therefore, Lemma 6.3 allows us to apply the CAT framework for this compression. In particular, with

$$\beta_k(T) = \frac{\langle \nabla F(x_k), Q_T(\nabla F(x_k))\rangle^2}{T \cdot \|\nabla F(x_k)\|_2^4}$$

we get the algorithm

$$\textbf{Step 1: } T_k = \operatorname*{argmax}_{T \in [1,d]} \frac{\beta_k(T)}{C(T)} \tag{6.10}$$

$$\textbf{Step 2: } \gamma_k = \frac{\sqrt{\beta_k(T_k)}}{\sqrt{T_k}L} \tag{6.11}$$

$$\textbf{Step 3: } x_{k+1} = x_k - \gamma_k Q_{T_k}(\nabla F(x_k)). \tag{6.12}$$

The algorithm optimizes T_k, based on each gradient and the actual communication cost. Note that [22] proposes a dynamic mechanism that chooses T_k so that $I_{T_k}(g_k)$ is the smallest subset such that $\sum_{j \in I_{T_k}(g_k)} |g_k^j| \geq \|g^j\|$. However, this heuristic has no clear connection to the descent or consideration for

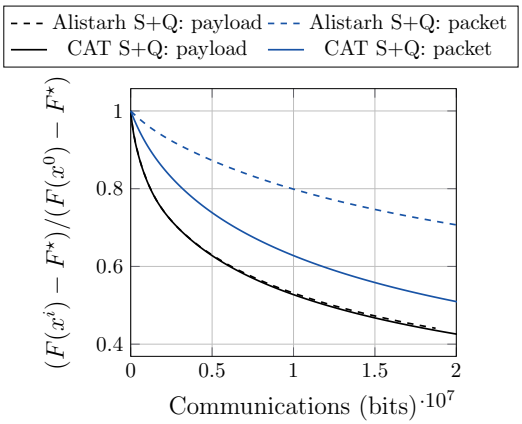


Figure 6.1: CAT sparsification + quantization on the `RCV1` data set.

communication costs. Our experiments show that our framework outperforms this heuristic in both running time and communication efficiency.

We compared CAT to the dynamic tuning introduced in [22]. In Figure 6.1, algorithms with both tuning rules are comparable if we only account for the payload in Equation (6.5). However, the heuristic rule in [22] is agnostic to the actual communication model $C(T)$ in Equation (6.6) with $c_1 = 128$ bytes, $c_0 = 64$ bytes and $P_{\max} = 128$ bytes. The blue lines show that the CAT is roughly two times more communication efficient than the dynamic tuning rule in [22] for the packet communication model.

6.3 Dynamic Stochastic Sparsification: Stochastic Gradient & Multiple Nodes

We finally illustrate how the CAT framework can improve the communication efficiency of stochastic sparsification. Our goal is to choose T_k and p_k dynamically for the stochastic sparsification in Eq. (6.4) to maximize the communication efficiency per iteration. To this end, we need the following descent lemma, similarly to the ones we proved for deterministic sparsifications in the last two sections.

Lemma 6.4. *Suppose that $F : \mathbb{R}^d \to \mathbb{R}$ is (possibly non-convex) L-smooth. Then for any $x, x^+ \in \mathbb{R}^d$ with $x^+ = x - \gamma Q_{T,p}(\nabla F(x))$ where $Q_{T,p}(\cdot)$ is defined in (6.4) and $\gamma = \omega_p(T)/L$ we have*

$$\mathbf{E}F(x^+) \leq \mathbf{E}F(x) - \frac{\omega_p(T)}{2L}\mathbf{E}||\nabla F(x)||^2,$$

where $\omega_p(T) = ||\nabla F(x)||^2/\mathbf{E}||Q_{T,p}(\nabla F(x))||^2$.

Similarly as before, we optimize the descent and the communication efficiency by maximizing, respectively, $\omega_p(T)$ and $\omega_p(T)/C(T)$. For a given T, the $p^\star$ minimizing $\omega_p(T)$ can be found efficiently, see [96] and our discussion in the supplementary materials. In this paper we always use $p^\star$ and omit p in $Q_T(\cdot)$ and $\omega(T)$. We can now use our CAT framework to optimize the communication efficiency. If we set $\omega_k(T)=||\nabla F(x_k)||^2/\mathbf{E}||Q_{T,p}(\nabla F(x_k))||^2$ we get the dynamic algorithm:

$$\textbf{Step 1: } T_k = \operatorname*{argmax}_{B\in[1,d]} \frac{\omega_k(T)}{C(T)} \tag{6.13}$$

$$\textbf{Step 2: } \gamma_k = \frac{\omega_k(T_k)}{L} \tag{6.14}$$

$$\textbf{Step 3: } x_{k+1} = x_k - \gamma_k Q_{T_k}(\nabla F(x_k)). \tag{6.15}$$

This algorithm can maximize communication efficiency by finding the optimal sparsity budget T to the one-dimensional problem. This can be solved efficiently since the sparsification parameter $\omega(T)$ has properties that are similar to $\alpha(T)$ for deterministic sparsification. Like Lemma 6.2 for deterministic sparsification, the following result shows that $\omega(T)$ is increasing with the budget $T \in [1, d]$ and is lower-bounded by T/d.

Lemma 6.5. *For any vector $g \in \mathbb{R}^d$ the function $\omega(T) = \|g\|^2/\|Q_{T,p}(g)\|^2$ is increasing over $T \in [1, d]$. Moreover, $\omega(T) \geq T/d$ for all $T \in [1, d]$, where we obtain the equality when $p^j = T/d$ for all $j \in [1, d]$.*

This lemma leads to many consequences for $\omega(T)$, analogous to $\alpha(T)$. For instance, by following proof arguments in Proposition 6.1, $\omega(T)/C(T)$ attains its maximum for a T which is an integer multiple of $\tau_{\max}$ when $C(T) = \tilde{c}_1\lceil T/\tau_{\max}\rceil + c_0$.

Furthermore, stochastic sparsification has some favorable properties that allow us to generalize our theoretical results to stochastic gradient methods and to multi-node settings. Suppose that we have n nodes that wish to solve the minimization problem with $F(x) = \sum_{i=1}^n F^i(x)/n$ where $F^i(\cdot)$ is kept by node $i \in [1, n]$. Then, we may solve the problem by distributed compressed gradient descent

$$x_{k+1} = x_k - \gamma \frac{1}{n} \sum_{i=1}^n Q_{T_k^i}\left(g^i(x_k; \xi_k^i)\right), \tag{6.16}$$

where $Q(\cdot)$ is the stochastic sparsifier and $g^i(x; \xi^i)$ is a stochastic gradient with respect to $\nabla F^i(x)$ at x. We assume that $g_j(x; \xi_j)$ is unbiased and satisfies a bounded variance assumption, i.e. $\mathbf{E}_\xi g^i(x; \xi^i) = \nabla F^i(x)$ and $\mathbf{E}_\xi \|g^i(x; \xi^i) - \nabla F(x)\|^2 \leq \sigma^2$. The expectation is with respect to a local data distribution

at node i. These conditions are standard to analyze randomized first-order algorithms in machine learning [132, 88].

We can derive a descent lemma for Algorithm (6.16) similarly as Lemma 6.4 for the single-node sparsification method (see Theorem 3 in the Appendix). This means that we easily prove similar data-dependent convergence results as we did for deterministic sparsification in Table 1. To illustrate this, suppose that for a given T there is $\bar{\omega}_T$ satisfying $\omega_k^i(T) \geq \bar{\omega}_T$ where $\omega_k^i(T) = ||\nabla F^i(x_k)||^2/\mathbf{E}||Q_T(\nabla F^i(x_k))||^2$. Then, the iteration complexity of Algorithm (6.16) is as given in the right part of Table 1. The parameter $\bar{\omega}_T$ captures the sparsification gain, similarly as $\bar{\alpha}_T$ did for deterministic sparsification. In the worst case there is no communication improvement of sparsification compared to sending full gradients, but when $\bar{\omega}_T$ is large the communication improvement can be significant.

6.4 Experimental Results

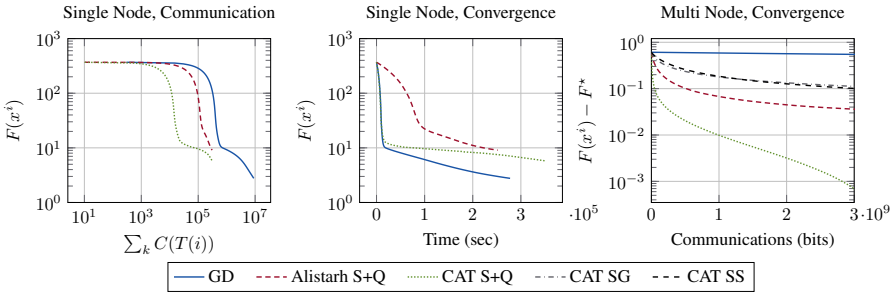


Figure 6.1: Performance of CAT frameworks on three compressors for solving logistic regression problems. We used the `URL` data set in the single-node (one master/one worker) architecture and `RCV1` in the multi-node (one master/four worker) setting.

Experiment 1 (single node). We evaluate the performance of our CAT framework for dynamic sparsification and quantization (S+Q) in the single-master, single-worker setup on the `URL` data set with 2.4 million data points and 3.2 million features. The master node, located 500 km away from the worker node, is responsible for maintaining the decision variables based on the gradient information received from the worker node. The nodes communicate with each other over a 1000 Mbit Internet connection using the `ZMQ` library. We implemented vanilla gradient descent (GD), Alistarh's S+Q [22] and CAT- S+Q using the C++ library `POLO` [87]. We first set `FPP` $= 32$ and measure the communication cost in a wall-clock time. After obtaining a linear fit to the measured communication cost (see the supplementary materials for

details), we ran 30,000 iterations and with step-size according to Lemma 6.3. Figure 6.1 shows the loss improvement with respect to the total communication cost (leftmost) and wall-clock time (middle). We observe that CAT S+Q outperforms GD and Alistarh's S+Q up to two orders and one order of magnitude, respectively, in communication efficiency. In terms of wall-clock time, CAT S+Q takes 26% (respectively, 39%) more time to finish the full 30,000 iterations than that of GD (respectively, Alistarh's S+Q). Note, however, that CAT S+Q achieves an order of magnitude loss improvement in an order of magnitude shorter time, and the loss value is always lower in CAT S+Q than that in Alistarh's S+Q. Such a performance is desirable in most of the applications (e.g., hyper-parameter optimizations and day-ahead market-price predictions) that do not impose a strict upper bound on the iteration counts but rather on the wall-clock time of the algorithm.

Experiment 2 (MPI - multiple nodes): We evaluate the performance of our CAT tuning rules on deterministic sparsification (SG), stochastic sparsification (SS), and sparsification with quantization (S+Q) in a multi-node setting on `RCV1`. We compare the results to gradient descent and Alistarh's S+Q [22]. We implement all algorithms in Julia, and run them on 4 nodes using MPI, splitting the data evenly between the nodes. In all cases we use the packet communication model (6.6) with $c_1 = 576$ bytes, $c_0 = 64$ bytes and $P_{\max} = 512$ bytes. The right-most plot in Figure 6.1 shows that our CAT S+Q outperforms all other compression schemes. In particular, CAT is roughly 6 times more communication efficient than the dynamic rule in [22] for the same compression scheme (compare number of bits needed to reach $\epsilon = 0.4$).

Additional Experiments for Stochastic Sparsification and Error Compensation. We include additional simulations that illustrate generality and efficiency of our CAT frameworks on stochastic sparsification, and on error compensation (see details e.g. in [124, 151, 125, 152]). We evaluated the performance for logistic regression problems on the `RCV1` data set. The packet communication model in Equation (6.6) also has $c_1 = 128$ bytes, $c_0 = 64$ bytes and $P_{\max} = 128$ bytes.

For stochastic sparsification, we compared traditional `ATOMO` called `SS` [96], against our CAT tuning with `ATOMO` named `CAT-SS`. The communications are averaged over three Monte Carlo runs. Figure 6.2 shows that stochastic sparsification has the same conclusions as deterministic sparsification. In the simplest payload model it is best to choose T small. However, for the packet model we need to carefully tune T so that it is neither too big nor too small. Our CAT rule adaptively finds the best value of T in both cases.

Next, we showed that our CAT algorithms can achieve faster convergence acceleration by using error compensation schemes. For deterministic sparsification, the error-compensated CAT algorithm is 2 times more communication efficient to achieve $F(x_k) = 0.4$ than the direct compression CAT algorithm (see Figure 6.3). This highlights high flexibility of our CAT framework to improve efficiency of any compressed algorithms.

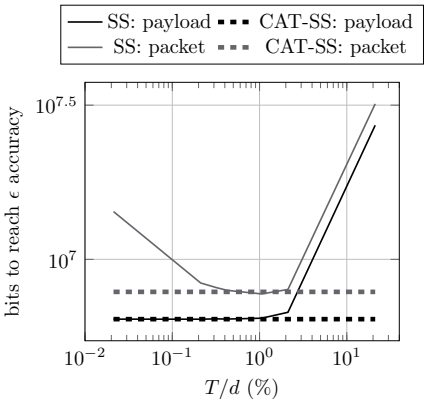


Figure 6.2: Expected communicated bits to reach ϵ-accuracy for gradient descent with stochastic sparsification.

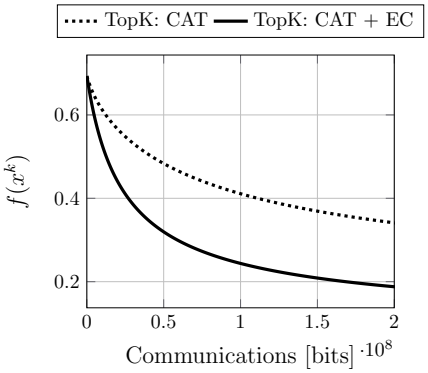


Figure 6.3: CAT sparsification with and without error compensation on `RCV1`.

Appendix

6.A Proofs of Lemmas and Propositions

6.A.1 Proof of Lemma 6.1

By the L-smoothness of $F(\cdot)$ and the iterate $x^+ = x - \gamma Q_T(\nabla F(x))$ where $x^+, x \in \mathbb{R}^d$, from Lemma 1.2.3. of [166] we have

$$F(x^+) \leq F(x) - \gamma \left\langle \nabla F(x), Q_T(\nabla F(x)) \right\rangle + \frac{L\gamma^2}{2} \|Q_T(\nabla F(x))\|^2.$$

It can be verified that

$$\langle g, Q_T(g) \rangle = ||Q_T(g)||^2$$

for all $g \in \mathbb{R}^d$ and, therefore, if $\gamma = 1/L$ then we have

$$F(x^+) \leq F(x) - \frac{1}{2L} \|Q_T(\nabla F(x))\|^2.$$

By the definition of $\alpha(T)$ we have $\|Q_T(\nabla F(x))\|^2 = \alpha(T)\|\nabla F(x)\|^2$, which yields the result.

Next, we prove that there exist L-smooth functions $F(\cdot)$ where the inequality is tight. Consider $F(x) = L\|x\|^2/2$. Then, F is L-smooth, and also satisfies

$$\begin{aligned} F(x - \gamma Q_T(\nabla F(x))) &= \frac{L}{2} \|x - \gamma Q_T(Lx)\|^2 \\ &= \frac{L}{2}\|x\|^2 - \gamma \langle Lx, Q_T(Lx) \rangle + \frac{L\gamma^2}{2} \|Q_T(Lx)\|^2. \end{aligned}$$

Since $\langle g, Q_T(g) \rangle = \|Q_T(g)\|^2$ by the definition $Q_T(\cdot)$ and $\gamma = 1/L$, we have

$$F(x - \gamma Q_T(\nabla F(x))) = F(x) - \frac{1}{2L} \|Q_T(Lx)\|^2.$$

Since $\nabla F(x) = Lx$, by the definition of $\alpha(T)$

$$\alpha(T) = \frac{\|Q_T(Lx)\|^2}{\|Lx\|^2} = \frac{\sum_{i \in I_T} x_i^2}{\sum_{i=1}^d x_i^2},$$

where I_T is the index set of T elements with the highest absolute magnitude. Therefore,

$$F(x - \gamma Q_T(\nabla F(x))) = F(x) - \frac{\alpha(T)}{2L} \|\nabla F(x)\|^2.$$

6.A.2 Proof of Lemma 6.2

Take $g \in \mathbb{R}^d$ and, without the loss of generality, we let $|g^1| \geq |g^2| \geq \ldots \geq |g^d|$ and $g^i \in \mathbb{R}$ (otherwise we may re-order g). To prove that $\alpha(T)$ is increasing we rewrite the definition of $\alpha(T)$ equivalently as

$$\alpha(T) = \sum_{j=1}^{T} (g^j)^2 / \|g\|^2, \quad \text{for} \quad T \in [0, d].$$

Notice that $\alpha(T) = 0$ when $T = 0$. Clearly, $\alpha(T)$ is also increasing with $T \in [1, d]$ since each term of the sum $\sum_{j=1}^{T} (g^j)^2$ is increasing.

We prove that $\alpha(T)$ is concave by recalling the slope of $\alpha(T)$

$$\frac{d}{dT}\alpha(T) = (g^M)^2 / \|g\|^2,$$

for $T \in (M-1, M)$ and $M = 1, 2, \ldots, d$. Since $|g^1| \geq |g^2| \geq \ldots \geq |g^d|$, the slope of $\alpha(T)$ has a non-increasing slope when T increases. Therefore, $\alpha(T)$ is concave.

We prove the second statement by writing $\|g\|^2$ on the form of

$$\|g\|^2 = \sum_{j \in I_T(g)} (g^j)^2 + \sum_{j \in I_{T^c}(g)} (g^j)^2,$$

where I_{T^c} is the index set of $d - T$ elements with lowest absolute magnitude. Applying the fact that $(g^j)^2 \leq \min_{l \in I_T(g)} (g^l)^2$ for $j \in I_{T^c}(g)$ and that $\min_{l \in I_T(g)} (g^l)^2 \leq (1/T) \sum_{l \in I_T(g)} (g^l)^2$ into the main inequality, we have

$$\|g\|^2 \leq \left(1 + \frac{d-T}{T}\right) \sum_{j \in I_T(g)} (g^j)^2.$$

By the definition of $Q_T(g)$, we get

$$\alpha(T) \geq T/d.$$

Finally, we prove the last statement by setting $F(x) = (1/2) x^T A x$ where $A = (L/d)\mathbf{1}\mathbf{1}^T$.Then $F(\cdot)$ is L-smooth and its gradient is

$$\nabla F(x) = \bar{x}\mathbf{1},$$

where

$$\bar{x} = \frac{1}{d} \sum_{j=1}^{d} x^j.$$

Therefore, $\|Q_T(\nabla F(x))\|^2 = (T/d)\|\nabla F(x)\|^2$.

6.A.3 Proof of Proposition 6.1

The ratio between a non-negative concave function $\alpha(T)$ and a positive affine function $C(T)$ is quasi-concave and semi-strictly quasi-concave [167, 168], meaning that every local maximal point is globally maximal.

Next, we consider $\alpha(T)/C(T)$ when $C(T) = \tilde{c}_1\lceil T/\tau_{\max}\rceil + c_0$. If $T \in ((c-1)\tau_{\max}, c\tau_{\max}]$, then $\alpha(T) \leq \alpha(c\tau_{\max})$ due to monotonicity of $\alpha(\cdot)$ and $C(T) = C(c\tau_{\max})$, meaning that $\alpha(T)/C(T) \leq \alpha(c\tau_{\max})/C(c\tau_{\max})$. This implies that $T = c\tau_{\max}$ maximizes $\alpha(T)/C(T)$ for $T \in ((c-1)\tau_{\max}, c\tau_{\max}]$ and that we can obtain the maximum of $T = c\tau_{\max}$ for some integers c.

6.A.4 Proof of Proposition 6.2

Take $g \in \mathbb{R}^d$ and, without the loss of generality, we let $|g^1| \geq |g^2| \geq \ldots \geq |g^d|$ and $g^j \in \mathbb{R}$ (otherwise we may re-order g). Since $C(T) = C \cdot T$ where $C = \lceil \log_2(d) \rceil + \texttt{FPP}$, we have

$$T_k = \underset{T\in[1,d]}{\operatorname{argmax}} \frac{\alpha_k(T)}{C(T)} = \underset{T\in[1,d]}{\operatorname{argmax}} \frac{\sum_{j=1}^{T}(g^j)^2}{C \cdot T}.$$

Since $\sum_{j=1}^{T}(g^j)^2/T \leq (g^1)^2$, the solution from Equation (6.9) is $T_k = 1$ for all k.

6.A.5 Proof of Lemma 6.3

By using the L-smoothness of $F(\cdot)$ (Lemma 1.2.3. of [166]) and the iterate $x^+ = x - \gamma Q_T(\nabla F(x))$ where $x^+, x \in \mathbb{R}^d$, we have

$$F(x^+) \leq F(x) - \gamma \left\langle \nabla F(x), Q_T(\nabla F(x)) \right\rangle + \frac{L\gamma^2}{2} \|Q_T(\nabla F(x))\|^2.$$

If $Q_T(\nabla F(x))$ has T non-zero elements, then we can easily prove that

$$\left\langle \nabla F(x), Q_T(\nabla F(x)) \right\rangle = \sqrt{T\beta(T)} \cdot \|\nabla F(x)\|^2, \quad \text{and}$$
$$\|Q_T(\nabla F(x))\|^2 = T \cdot \|\nabla F(x)\|^2,$$

where $\beta(T)$ is defined as

$$\beta(T) = \frac{1}{T} \frac{\left\langle \nabla F(x), Q_T(\nabla F(x)) \right\rangle^2}{\|\nabla F(x)\|_2^4}.$$

Plugging these equations into the above inequality yields

$$F(x^+) \leq F(x) - \left(\gamma\sqrt{T\beta(T)} - \frac{TL\gamma^2}{2} \right) \|\nabla F(x)\|^2.$$

Setting $\gamma = \sqrt{\beta(T)}/(\sqrt{T}L)$ completes the proof.

6.A.6 Proof of Lemma 6.4

By using the L-smoothness of $F(\cdot)$ (Lemma 1.2.3. of [166]) and the iterate $x^+ = x - \gamma Q_{T,p}(\nabla F(x))$ where $x^+, x \in \mathbb{R}^d$, we have

$$F(x^+) \leq F(x) - \gamma \langle \nabla F(x), Q_{T,p}(\nabla F(x)) \rangle + \frac{L\gamma^2}{2} \|Q_{T,p}(\nabla F(x))\|^2.$$

Since $\omega_p(T)$ is defined by

$$\omega_p(T) = \frac{||\nabla F(x)||^2}{\mathbf{E}||Q_{T,p}(\nabla F(x))||^2},$$

taking the expectation, and using the unbiased property of $Q_{T,p}(\cdot)$ we get

$$\mathbf{E}F(x^+) \leq \mathbf{E}F(x) - \left(\gamma - \frac{L\gamma^2}{2\omega_p(T)}\right) \mathbf{E}\|\nabla F(x)\|^2.$$

Now taking $\gamma = \omega_p(T)/L$ concludes the complete the proof.

6.A.7 Proof of Lemma 6.5

Consider $Q_{T,p}(\cdot)$ in Equation (6.4). Then,

$$\mathbf{E}\|Q_{T,p}(g)\|^2 = \sum_{j=1}^{d} \frac{1}{p^j} (g^j)^2.$$

Here, we assume without the loss of generality that each element of $g \in \mathbb{R}^d$ is g^j such that $|g^1| \geq |g^2| \geq \ldots \geq |g^d|$ (otherwise we may re-order g). Therefore,

$$\omega(T) = \frac{\sum_{j=1}^{d} (g^j)^2}{\sum_{j=1}^{d} \frac{1}{p^j} (g^j)^2}.$$

To ensure the high sparsity budget T of the compressed gradient $Q_{T,p}(g)$, probabilities must also have high values in some coordinates (some p^j are close to one). Therefore, $\omega(T)$ is increasing over the sparsity budget $T \in [1, d]$.

Next, we assume that $Q_{T,p}(\cdot)$ in Equation (6.4) has $p^j = T/d$ for all $j \in [1, d]$. Then,

$$\mathbf{E}\|Q_{T,p}(g)\|^2 = \frac{d}{T}\|g\|^2.$$

Plugging this result into the main definition, we have $\omega(T) = T/d$. Since we assign p that minimizes $\omega_p(T) = \|g\|^2/\mathbf{E}\|g\|^2$, $\omega_p(T) \geq T/d$.

6.B Iteration Complexities of Adaptive Compressors

In this section, we provide the iteration complexities of gradient descent (6.1) with three main compressors: deterministic sparsification (6.2), dynamic sparsification together with quantization (6.3) , and stochastic sparsification (6.4).

6.B.1 Analysis for Deterministic Sparsification

We provide theoretical convergence guarantees for gradient descent using deterministic sparsification.

Theorem 6.1. *Consider the minimization problem over the function $F(x)$ and the iterates $\{x_k\}_{k\in\mathbb{N}}$ generated by gradient descent with dynamic sparsification in Equation* (6.9)*. Suppose that there exists $\bar{\alpha}_T \in [0,1]$ such that $\alpha_k(T) \geq \bar{\alpha}_T \geq T/d$ for all k. Set $\epsilon_0 = F(x_0) - F(x^\star)$. Then,*

1. ***Non-convex:*** *If F is L-smooth, then we find $\min_{l\in[0,k-1]} \|\nabla F(x_l)\|^2 \leq \epsilon$ in*

$$k = \frac{1}{\bar{\alpha}_T}\frac{2L\epsilon_0}{\epsilon} \quad \textit{iterations.}$$

2. ***Convex:*** *If F is also convex and there exists a positive constant R such that $\|x_k - x^\star\| \leq R$, then we find $F(x_k) - F(x^\star) \leq \epsilon$ in*

$$k = \frac{1}{\bar{\alpha}_T}\frac{2LR^2}{\epsilon} \quad \textit{iterations.}$$

3. ***Strongly-convex:*** *If F is also μ-strongly convex, then we find $F(x_k) - F(x^\star) \leq \epsilon$ in*

$$k = \frac{1}{\bar{\alpha}_T}\kappa \log\left(\frac{\epsilon_0}{\epsilon}\right) \quad \textit{iterations.}$$

Proof. See Appendix 6.C. □

6.B.2 Analysis for Dynamic Sparsification together with Quantization

We prove convergence rate of gradient descent with dynamic sparsification together with quantization.

Theorem 6.2. *Consider the minimization problem over the function $F(x)$ and the iterates $\{x_k\}_{k\in\mathbb{N}}$ generated by gradient descent with sparsification together with quantization in Equation* (6.10)*. Suppose that there exist constants $T \in (0,d]$ and $\bar{\beta}_T \in (0,\infty)$ such that $T_k \leq T$ and $\beta_k(T_k) \geq \bar{\beta}_T$ for all i, respectively. Set $\epsilon_0 = F(x_0) - F(x^\star)$. Then,*

1. ***Non-convex:*** *If F is L-smooth, then we find* $\min_{l\in[0,k-1]}\|\nabla F(x_l)\|^2 \leq \epsilon$ *in*

$$k = \frac{1}{\bar{\beta}_T}\frac{2L\epsilon_0}{\epsilon} \quad \textit{iterations.}$$

2. ***Convex:*** *If F is also convex and there exists a positive constant R such that $\|x_k - x^\star\| \leq R$, then we find $F(x_k) - F(x^\star) \leq \epsilon$ in*

$$k = \frac{1}{\bar{\beta}_T}\frac{2LR^2}{\epsilon} \quad \textit{iterations.}$$

3. ***Strongly-convex:*** *If F is also μ-strongly convex, then we find $F(x_k) - F(x^\star) \leq \epsilon$ in*

$$k = \frac{1}{\bar{\beta}_T}\kappa\log\left(\frac{\epsilon_0}{\epsilon}\right) \quad \textit{iterations.}$$

Proof. See Appendix 6.D. □

6.B.3 Analysis for Stochastic Sparsification

We prove iteration complexities of stochastic gradient descent with stochastic sparsification in the multi-node setting.

Theorem 6.3. *Consider the minimization problem over the function $F(x) = \sum_{i=1}^n F^i(x)$, where each $F^i(\cdot)$ is L-smooth. Let the iterates $\{x_k\}_{k\in\mathbb{N}}$ generated by Algorithm* (6.16)*, and suppose that there exists $\bar{\omega}_T$ such that $\omega_k^i(T_k^i) \geq \bar{\omega}_T$ where ω_k^i is the sparsification parameter of node i at iteration k. Set $\epsilon_0^F = F(x_0) - F(x^\star)$ and $\epsilon_0^X = \|x_0 - x^\star\|$. Then,*

1. ***Non-convex*** *If*

$$\gamma = \frac{\bar{\omega}_T}{2L}\frac{1}{2\sigma^2/\epsilon + 1}$$

then we find $\min_{l\in[0,k-1]}\mathbf{E}\|\nabla F(x_l)\|^2 \leq \epsilon$ *in*

$$k = \frac{2}{\bar{\omega}_T}\left(1 + \frac{2\sigma^2}{\epsilon}\right)\cdot\frac{2L\epsilon_0^F}{\epsilon} \quad \textit{iterations.}$$

2. ***Convex*** *If F is convex and*

$$\gamma = \frac{\bar{\omega}_T}{2}\frac{1}{2\sigma^2/\epsilon + L}$$

then we find $\mathbf{E}\left[F(\sum_{l=0}^{k-1} x_l/k) - F^\star\right] \leq \epsilon$

$$k = \frac{2}{\bar{\omega}_T}\left(1 + \frac{2\sigma^2}{\epsilon L}\right)\frac{2L\epsilon_0^X}{\epsilon} \quad \textit{iterations.}$$

3. ***Strongly-convex*** *If F is μ-strongly convex and*

$$\gamma = \frac{\bar{\omega}_T}{2} \frac{1}{2\sigma^2/(\mu\epsilon) + L}$$

then we find $\mathbf{E}[F(x_k) - F(x^\star)] \leq \epsilon$ in

$$k = \frac{2}{\bar{\omega}_T} \kappa \left(1 + \frac{2\sigma^2}{\mu\epsilon L}\right) \log\left(\frac{2\epsilon_0^F}{\epsilon}\right) \quad \textit{iterations.}$$

Proof. See Appendix 6.E. □

6.C Iteration Complexities for Deterministic Sparsification

In this section, we derive iteration complexities of gradient descent with deterministic sparsification.

Proof of Theorem 6.1-1

By recursively applying the inequality from Lemma 6.1 with $x^+ = x_{k+1}$ and $x = x_k$, we have

$$\min_{l \in [0,k-1]} \|\nabla F(x_l)\|^2 \leq \frac{2L}{k} \sum_{l=0}^{k-1} \frac{1}{\alpha_k(T)} [F(x_l) - F(x_{l+1})]$$

where the inequality follows from the fact that $\min_{l\in[0,k-1]} \|\nabla F(x_l)\|^2 \leq \sum_{l=0}^{k-1} \|\nabla F(x_k)\|^2/k$. If there exists $\bar{\alpha}_T$ such that $\alpha_k(T) \geq \bar{\alpha}_T$, then

$$\min_{l \in [0,k-1]} \|\nabla F(x_l)\|^2 \leq \frac{2L}{k\bar{\alpha}_T} [F(x_0) - F(x_i)].$$

Since $F(x) \geq F(x^\star)$ for $x \in \mathbb{R}^d$,

$$\min_{l \in [0,k-1]} \|\nabla F(x_l)\|^2 \leq \frac{2L}{k\bar{\alpha}_T} \epsilon_0,$$

where $\epsilon_0 = F(x_0) - F(x^\star)$. This means that to reach target solution accuracy at ϵ (i.e. $\min_{l\in[0,k-1]} \|\nabla F(x_l)\|^2 \leq \epsilon$), the sparsified gradient method (6.1) needs at most

$$k \geq \frac{1}{\bar{\alpha}_T} \frac{2L\epsilon_0}{\epsilon} \quad \text{iterations.}$$

We also recover the iteration complexities of the sparsified gradient method and of classical full gradient method when we let $\bar{\alpha}_T = T/d$ and $\bar{\alpha}_T = 1$, respectively.

Proof of Theorem 6.1-2

Before deriving the result, we introduce one useful lemma.

Lemma 6.6. *The non-negative sequence* $\{V_k\}_{k\in\mathbb{N}}$ *generated by*

$$V_{k+1} \leq V_k - qV_k^2, \quad \textit{for} \quad q > 0 \tag{6.17}$$

satisfies

$$\frac{1}{V_k} \geq \frac{1}{V_0} + kq. \tag{6.18}$$

Proof. By the fact that $x^2 \geq 0$ for $x \in \mathbb{R}$, clearly $V_{k+1} \leq V_k$. By the proper manipulation, we rearrange the terms in Equation (6.17) as follows:

$$\frac{1}{V_{ki+1}} - \frac{1}{V_k} \geq q\frac{V_k}{V_{k+1}} \geq q,$$

where the last inequality follows from the fact that $V_{k+1} \leq V_k$. By the recursion, we complete the proof. □

By Lemma 6.1 with $x^+ = x_{k+1}$ and $x = x_k$, we have

$$F(x_{k+1}) \leq F(x_k) - \frac{\alpha_k(T)}{2L}\|\nabla F(x_k)\|^2$$

Since F is convex, i.e.

$$F(x) - F(x^\star) \leq \langle \nabla F(x), x - x^\star \rangle, \text{ for } x \in \mathbb{R}^d$$

by Cauchy-Scwartz's inequality and assuming that the iteration satisfies $\|x - x^\star\| \leq R$ for $R > 0$ and $x \in \mathbb{R}^d$,

$$\|\nabla F(x)\| \geq \frac{1}{R}\left[F(x) - F(x^\star)\right].$$

Plugging this inequality into the main result, we have

$$V_{k+1} \leq V_k - \frac{\alpha_k(T)}{2LR^2}V_k^2,$$

where $V_k = F(x_k) - F(x^\star)$. If there exists $\bar{\alpha}_T$ such that $\alpha_k(T) \geq \bar{\alpha}_T$, then by Lemma 6.6 and by using the fact that $V_0 \geq 0$

$$V_k \leq \frac{1}{\bar{\alpha}_T}\frac{2LR^2}{k}.$$

To reach $F(x_k) - F(x^\star) \leq \epsilon$, the sparsified gradient methods needs the number of iterations i satisfying

$$k \geq \frac{1}{\bar{\alpha}_T}\frac{2LR^2}{\epsilon}.$$

We also recover the iteration complexities of the sparsified gradient method and of classical full gradient method when we let $\bar{\alpha}_T = T/d$ and $\bar{\alpha}_T = 1$, respectively.

Proof of Theorem 6.1-3

By Lemma 6.1 with $x^+ = x_{k+1}$ and $x = x_k$, we have

$$F(x_{k+1}) \leq F(x_k) - \frac{\alpha_k(T)}{2L}\|\nabla F(x_k)\|^2.$$

Since F is μ-strongly convex, i.e. $\|\nabla F(x)\|^2 \geq 2\mu[F(x) - F(x^\star)]$ for $x \in \mathbb{R}^d$, applying this inequality into the main result we have

$$F(x_{k+1}) - F(x^\star) \leq \left(1 - \frac{\mu\alpha_k(T)}{L}\right)[F(x_k) - F(x^\star)].$$

If there exists $\bar{\alpha}_T$ such that $\alpha_k(T) \geq \bar{\alpha}_T$, then by the recursion we get

$$F(x_k) - F(x^\star) \leq \left(1 - \frac{\mu\alpha_T}{L}\right)^k \epsilon_0,$$

where $\epsilon_0 = F(x_0) - F(x^\star)$. To reach $F(x_k) - F(x^\star) \leq \epsilon$, the sparsified gradient methods requires the number of iterations k satisfying

$$\left(1 - \frac{\mu\alpha_T}{L}\right)^k \epsilon_0 \leq \epsilon.$$

Taking the logarithm on both sides of the inequality and using the fact that $-1/\log(1-x) \leq 1/x$ for $0 < x \leq 1$, we have

$$k \geq \frac{1}{\bar{\alpha}_T}\kappa \log\left(\frac{\epsilon_0}{\epsilon}\right).$$

We also recover the iteration complexities of the sparsified gradient method and of classical full gradient method when we let $\bar{\alpha}_T = T/d$ and $\bar{\alpha}_T = 1$, respectively.

6.D Iteration Complexities for S+Q

In this section, we prove the iteration complexities of gradient descent using dynamic sparsification together with quantization.

Proof of Theorem 6.2-1

Suppose that there exist T and $\bar{\beta}_T$ such that $T_k \leq T$ and $\beta_k(T_k) \geq \bar{\beta}_T$ for all k, respectively. Applying Lemma 6.3 with $x^+ = x_{k+1}$ and $x = x_k$ and using the fact that

$$\min_{l \in [0,k-1]} \|\nabla F(x_l)\|^2 \leq \frac{1}{k}\sum_{l=0}^{k-1} \|\nabla F(x_l)\|^2,$$

we then have

$$\min_{l\in[0,k-1]} \|\nabla F(x_l)\|^2 \leq \frac{2L}{\bar{\beta}_T \cdot k} \sum_{l=0}^{k-1} [F(x_l) - F(x_{l+1})].$$

Next, by the cancellations of the telescopic series, and by the fact that $F(x) \geq F(x^\star)$ for $x \in \mathbb{R}^d$,

$$\min_{l\in[0,k-1]} \|\nabla F(x_l)\|^2 \leq \frac{2L}{\bar{\beta}_T k} \epsilon_0,$$

where $\epsilon_0 = F(x_0) - F(x^\star)$. To reach $\min_{l\in[0,k-1]} \|\nabla F(x_l)\|^2 \leq \epsilon$, gradient descent using sparsification and quantization needs at most

$$k \geq \frac{2L}{\bar{\beta}_T} \frac{\epsilon_0}{\epsilon} \quad \text{iterations.}$$

Proof of Theorem 6.2-2

Since $\beta_k(T_k) \geq \bar{\beta}_T$ for all k, applying Lemma 6.3 with $x^+ = x_{k+1}$ and $x = x_k$ we have

$$[F(x_{k+1}) - F(x^\star)] \leq [F(x_k) - F(x^\star)] - \frac{\bar{\beta}_T}{2L} \|\nabla F(x_k)\|^2.$$

Since F is convex, i.e.

$$F(x) - F(x^\star) \leq \langle \nabla F(x), x - x^\star \rangle, \text{ for } x \in \mathbb{R}^d$$

by Cauchy-Scwartz's inequality and assuming that the iteration satisfies $\|x - x^\star\| \leq R$ for $R > 0$ and $x \in \mathbb{R}^d$,

$$\|\nabla F(x)\| \geq \frac{1}{R} \left[F(x) - F(x^\star)\right].$$

Plugging this inequality into the main result, we have

$$V_{k+1} \leq V_k - \frac{\bar{\beta}_T}{2LR^2} V_k^2,$$

where $V_k = F(x_k) - F(x^\star)$. Applying Lemma 6.6 with $V_0 \geq 0$ into this inequality we get

$$V_k \leq \frac{1}{\bar{\beta}_T} \frac{2LR^2}{k}.$$

To reach $F(x_k) - F(x^\star) \leq \epsilon$, gradient descent using sparsification with quantization needs the number of iterations k satisfying

$$k \geq \frac{1}{\bar{\beta}_T} \frac{2LR^2}{\epsilon}.$$

Proof of Theorem 6.2-3

Since $\beta_k(T_k) \geq \bar{\beta}_T$ for all k, applying Lemma 6.3 with $x^+ = x_{k+1}$ and $x = x_k$ we have

$$F(x_{k+1}) \leq F(x_k) - \frac{\bar{\beta}_T}{2L}\|\nabla F(x_k)\|^2.$$

Since F is μ-strongly convex, i.e. $\|\nabla F(x)\|^2 \geq 2\mu[F(x) - F(x^\star)]$ for $x \in \mathbb{R}^d$, applying this inequality into the main result we have

$$F(x_{k+1}) - F(x^\star) \leq \left(1 - \frac{\mu\bar{\beta}_T}{L}\right)[F(x_k) - F(x^\star)].$$

By the recursion, we get

$$F(x_k) - F(x^\star) \leq \left(1 - \frac{\mu\bar{\beta}_T}{L}\right)^k \epsilon_0,$$

where $\epsilon_0 = F(x_0) - F(x^\star)$. To reach $F(x_k) - F(x^\star) \leq \epsilon$, the sparsified gradient methods requires the number of iterations k satisfying

$$\left(1 - \frac{\mu\bar{\beta}_T}{L}\right)^k \epsilon_0 \leq \epsilon.$$

Taking the logarithm on both sides of the inequality and using the fact that $-1/\log(1-x) \leq 1/x$ for $0 < x \leq 1$, we have

$$k \geq \frac{1}{\bar{\beta}_T}\kappa \log\left(\frac{\epsilon_0}{\epsilon}\right).$$

6.E Iteration Complexities for Distributed Stochastic Sparsified Gradient

We prove the iteration complexities of multi-node stochastic gradient descent with stochastic sparsification in Equation (6.16). We begin by introducing three useful lemmas for our analysis.

Lemma 6.7. *Let $\{x_k\}_{k\in\mathbb{N}}$ be the iterates generated by Algorithm* (6.16) *and suppose that there exists $\bar{\omega}_T$ such that $\omega_k^i(T_k^i) \geq \bar{\omega}_T$ where ω_k^i is the sparsification parameter of node i at iteration k. Then,*

$$\mathbf{E}\left\|\frac{1}{n}\sum_{i=1}^n Q_{T_k^i}\left(g^i(x_k;\xi_k^i)\right)\right\|^2 \leq \frac{2}{\bar{\omega}_T}\left(\mathbf{E}\|\nabla F(x_k)\|^2 + \sigma^2\right).$$

Proof. Since

$$\mathbf{E}\|Q_{T_k^i}(g^i(x_k;\xi_k^i))\|^2 = \|g^i(x_k;\xi_k^i)\|^2/\omega_k^i(T_k^i),$$

by using Cauchy-Scwartz's inequality and by the fact that $\omega_k^i(T_k^i) \geq \bar{\omega}_T$ where ω_k^i is the sparsification level of node i at iteration k, we have

$$\mathbf{E}\left\|\frac{1}{n}\sum_{i=1}^n Q_{T_k^i}\left(g^i(x_k;\xi_k^i)\right)\right\|^2 \leq \frac{1}{n\bar{\omega}_T}\sum_{i=1}^n \mathbf{E}\left\|g^i(x_k;\xi_k^i)\right\|^2.$$

After utilizing the inequality $\|x+y\|^2 \leq 2\|x\|^2 + 2\|y\|^2$ with $x = g^i(x_k;\xi_k^i) - \nabla F(x_k)$ and $y = \nabla F(x_k)$,

$$\mathbf{E}\left\|\frac{1}{n}\sum_{i=1}^n Q_{T_k^i}\left(g^i(x_k;\xi_k^i)\right)\right\|^2 \leq \frac{2}{n\bar{\omega}_T}\sum_{i=1}^n \left(T + \mathbf{E}\|\nabla F(x_k)\|^2\right).$$

where $T = \mathbf{E}\left\|g^i(x_k;\xi_k^i) - \nabla F(x_k)\right\|^2$. By the variance-bounded assumption (i.e. $\mathbf{E}\|g^i(x;\xi^i) - \nabla F(x)\|^2 \leq \sigma^2$), we complete the proof.

□

Lemma 6.8. *Suppose that each component function $F^i(\cdot)$ is L-smooth. Let $\{x_k\}_{k\in\mathbb{N}}$ be the iterates generated by Algorithm* (6.16) *and assume that there exists $\bar{\omega}_T$ such that $\omega_k^i(T_k^i) \geq \bar{\omega}_T$ where ω_k^i is the sparsification parameter of node i at iteration k. Then,*

$$\mathbf{E}F(x_{k+1}) \leq \mathbf{E}F(x_k) + (L/\bar{\omega}_T)\gamma^2\sigma^2 - \left(\gamma - (L/\bar{\omega}_T)\gamma^2\right)\mathbf{E}\|\nabla F(x_k)\|^2.$$

Proof. By Cauchy-Scwartz's inequality and the fact that $F(x) = \sum_{i=1}^n F^i(x)/n$, we can easily show that $F(\cdot)$ is also L-smooth. From the smoothness assumption of $F(\cdot)$ (Lemma 1.2.3. of [166]) and Equation (6.16),

$$F(x_{k+1}) \leq F(x_k) - \gamma\left\langle \nabla F(x_k), g_k\right\rangle + \frac{L\gamma^2}{2}\|g_k\|^2$$

where $g_k = (1/n)\sum_{i=1}^n Q_{T_k^i}\left(g^i(x_k;\xi_k^i)\right)$. Taking the expectation, and using Lemma 6.7 and the unbiased properties of stochastic gradient $g^i(\cdot)$ and stochastic sparsification $Q_T(\cdot)$, we complete the proof. □

Lemma 6.9. *Suppose that each component function $F^i(\cdot)$ is L-smooth and F is convex. Let $\{x_k\}_{k\in\mathbb{N}}$ be the iterates generated by Algorithm* (6.16) *and assume that there exists $\bar{\omega}_T$ such that $\omega_k^i(T_k^i) \geq \bar{\omega}_T$ where ω_k^i is the sparsification parameter of node i at iteration k. Then,*

$$\begin{aligned}\mathbf{E}\|x_{k+1} - x^\star\|^2 \leq{}& \mathbf{E}\|x_k - x^\star\|^2 + 2\gamma^2\sigma^2/\bar{\omega}_T \\ &- 2(\gamma - L\gamma^2/\bar{\omega}_T)\mathbf{E}\left\langle \nabla F(x_k), x_k - x^\star\right\rangle.\end{aligned}$$

Proof. From the definition of the Euclidean norm and Equation (6.16), we have

$$\|x_{k+1} - x^\star\|^2 = \|x_k - x^\star\|^2 - 2\gamma \langle g_k, x_k - x^\star \rangle + \gamma^2 \|g_k\|^2,$$

where $g_k = (1/n) \sum_{i=1}^n Q_{T_k^i}\left(g^i(x_k; \xi_k^i)\right)$. Taking the expectation, and using Lemma 6.7 and the unbiased properties of stochastic gradient $g^i(\cdot)$ and stochastic sparsification $Q_T(\cdot)$, we have

$$\begin{aligned}\mathbf{E}\|x_{k+1} - x^\star\|^2 \leq \mathbf{E}\|x_k - x^\star\|^2 + (2/\bar{\omega}_T)\gamma^2\sigma^2 - 2\gamma \mathbf{E} \langle \nabla F(x_k), x_k - x^\star \rangle \\ + (2/\bar{\omega}_T)\gamma^2 \mathbf{E}\|\nabla F(x_k)\|^2\end{aligned}$$

Since $F(\cdot)$ is L-smooth, i.e. for $x \in \mathbb{R}^d$

$$\|\nabla F(x) - \nabla F(y)\|^2 \leq L \langle \nabla F(x) - \nabla F(y), x - y \rangle,$$

applying this inequality with $x = x_k$ and $y = x^\star$ into the main result and recalling that $\nabla F(x^\star) = 0$ we complete the proof. □

Now, we prove the main results for Algorithm (6.16).

Proof of Theorem 6.3-1.

If $\gamma < \bar{\omega}_T/L$, rearranging the terms from Lemma 6.7, we get

$$\mathbf{E}\|\nabla F(x_k)\|^2 \leq \frac{1}{\gamma - (L/\bar{\omega}_T)\gamma^2} \left(\mathbf{E}F(x_k) - \mathbf{E}F(x_{k+1})\right) + \frac{(L/\bar{\omega}_T)\gamma}{1 - (L/\bar{\omega}_T)\gamma}\sigma^2.$$

Since $\min_{l \in [0,k-1]} \mathbf{E}\|\nabla F(x_l)\|^2 \leq \sum_{l=0}^{k-1} \mathbf{E}\|\nabla F(x_l)\|^2/i$, we obtain

$$\min_{l \in [0,k-1]} \mathbf{E}\|\nabla F(x_l)\|^2 \leq \frac{\mathbf{E}F(x_0) - \mathbf{E}F(x^k)}{k[\gamma - (L/\bar{\omega}_T)\gamma^2]} + T,$$

where $T = \sigma^2 \cdot (L/\bar{\omega}_T)\gamma/[1 - (L/\bar{\omega}_T)\gamma]$. By the fact that $F(x) \geq F(x^\star)$ for $x \in \mathbb{R}^d$, we have

$$\min_{l \in [0,k-1]} \mathbf{E}\|\nabla F(x_l)\|^2 \leq \frac{1}{k[\gamma - (L/\bar{\omega}_T)\gamma^2]}\epsilon_0 + T,$$

where $\epsilon_0 = F(x_0) - F(x^\star)$. If the step-size is

$$\gamma = \frac{\bar{\omega}_T}{2L} \frac{1}{2\sigma^2/\epsilon + 1},$$

then clearly $\gamma < \bar{\omega}/L$ and

$$\frac{(L/\bar{\omega}_T)\gamma}{1 - (L/\bar{\omega}_T)\gamma}\sigma^2 \leq \frac{\epsilon}{2}.$$

From Lemma 6.8, Algorithm (6.16) reaches $\min_{l\in[0,k-1]} \mathbf{E}\|\nabla F(x_l)\|^2 \leq \epsilon$ for the number of iterations k which fulfills

$$\frac{1}{k}\frac{2L\epsilon_0}{\bar{\omega}_T}(2\sigma^2/\epsilon+1)\cdot\frac{4\sigma^2/\epsilon+2}{4\sigma^2/\epsilon+1}\leq\frac{\epsilon}{2}.$$

Since $\left(4\sigma^2/\epsilon+2\right)/\left(4\sigma^2/\epsilon+1\right)\leq 2$, the main condition can be rewritten equivalently as

$$k\geq\frac{4}{\bar{\omega}_T}\left(1+\frac{2\sigma^2}{\epsilon}\right)\cdot\frac{2L\epsilon_0}{\epsilon}.$$

Proof of Theorem 6.3-2.

If $\gamma<\bar{\omega}_T/L$, by Lemma 6.9 and by the convexity of F, i.e. $\langle\nabla F(x),x-x^\star\rangle\geq F(x)-F(x^\star)$ for $x\in\mathbb{R}^d$ we get

$$\begin{aligned}\mathbf{E}\|x_{k+1}-x^\star\|^2\leq{}&\mathbf{E}\|x_k-x^\star\|^2+2\gamma^2\sigma^2/\bar{\omega}_T\\&-2(\gamma-L\gamma^2/\bar{\omega}_T)\mathbf{E}[F(x_k)-F(x^\star)]\end{aligned}$$

By rearranging the terms and using the fact that F is convex, i.e. $F(\sum_{l=0}^{k-1}x_l)\leq\sum_{l=0}^{k-1}F(x_l)$, we then have

$$\begin{aligned}\mathbf{E}\left[F\left(\frac{1}{k}\sum_{l=0}^{k-1}x_l\right)-F(x^\star)\right]&\leq\frac{1}{k}\sum_{l=0}^{k-1}\mathbf{E}[F(x_l)-F(x^\star)]\\&\leq\frac{1}{k}\frac{1}{2\gamma}\frac{\epsilon_0}{1-(L/\bar{\omega}_T)\gamma}+\frac{\gamma\sigma^2/\bar{\omega}_T}{1-(L/\bar{\omega}_T)\gamma},\end{aligned}$$

where $\epsilon_0=\|x_0-x^\star\|^2$. The last inequality follows from the cancellations of the telescopic series the fact that $\|x\|^2\geq 0$ for $x\in\mathbb{R}^d$. If the step-size is

$$\gamma=\frac{\bar{\omega}_T}{2}\frac{1}{2\sigma^2/\epsilon+L},$$

then clearly $\gamma<\bar{\omega}_T/L$ and

$$\frac{\gamma\sigma^2/\bar{\omega}_T}{1-(L/\bar{\omega}_T)\gamma}\leq\frac{\epsilon}{2}.$$

To reach $\mathbf{E}\left[F\left(\sum_{l=0}^{k-1}x_l/k\right)-F(x^\star)\right]\leq\epsilon$, Algorithm (6.16) needs the number of iterations k satisfying

$$\frac{1}{k}\frac{1}{\bar{\omega}_T}\frac{2(2\sigma^2/\epsilon+L)^2}{4\sigma^2/\epsilon+L}\epsilon_0\leq\frac{\epsilon}{2}.$$

Since $\left(4\sigma^2/\epsilon+2L\right)/\left(4\sigma^2/\epsilon+L\right)\leq 2$, the main condition can be rewritten equivalently as

$$k\geq\frac{2}{\bar{\omega}_T}\left(1+\frac{2\sigma^2}{\epsilon L}\right)\frac{2L\epsilon_0}{\epsilon}$$

Proof of Theorem 6.3-3.

If $\gamma < \bar{\omega}_T/L$, then by Lemma 6.8 and by the strong convexity of $F(\cdot)$, i.e. $\|\nabla F(x)\|^2 \geq 2\mu[F(x) - F(x^\star)]$ for $x \in \mathbb{R}^d$ we have

$$\mathbf{E}[F(x_{k+1}) - F(x^\star)] \leq \rho \mathbf{E}[F(x_k) - F(x^\star)] + \frac{L}{\bar{\omega}_T}\gamma^2\sigma^2,$$

where

$$\rho = 1 - 2\mu\left(\gamma - \frac{L}{\bar{\omega}_T}\gamma^2\right).$$

By applying the inequality recursively, we get

$$\mathbf{E}[F(x_{k+1}) - F(x^\star)] \leq \rho^k \epsilon_0 + \frac{L}{2\mu\bar{\omega}_T}\frac{\gamma}{1 - L\gamma/\bar{\omega}_T}\sigma^2,$$

where $\epsilon_0 = F(x_0) - F(x^\star)$. If the step-size is

$$\gamma = \frac{\bar{\omega}_T}{2}\frac{1}{2\sigma^2/(\mu\epsilon) + L},$$

then clearly $\gamma < \bar{\omega}_T/L$ and

$$\frac{L}{2\mu\bar{\omega}_T}\frac{\gamma}{1 - L\gamma/\bar{\omega}_T}\sigma^2 \leq \frac{\epsilon}{2}.$$

To reach $\mathbf{E}\|x_k - x^\star\|^2 \leq \epsilon$, Algorithm (6.16) needs the number of iterations k which satisfies

$$\left(1 - \frac{\mu}{2}\cdot\frac{\bar{\omega}_T(4\sigma^2/(\mu\epsilon) + L)}{(2\sigma^2/(\mu\epsilon) + L)^2}\right)^k \epsilon_0 \leq \frac{\epsilon}{2}$$

Taking the logarithm on both sides, and utilizing the fact that $-1/\log(1-x) \leq 1/x$ for $0 < x \leq 1$, we have

$$k \geq \frac{2}{\bar{\omega}_T\mu}\frac{\left(2\sigma^2/(\mu\epsilon) + L\right)^2}{4\sigma^2/(\mu\epsilon) + L}\log\left(\frac{2\epsilon_0}{\epsilon}\right).$$

Since $\left(4\sigma^2/(\mu\epsilon) + 2L\right)/\left(4\sigma^2/(\mu\epsilon) + L\right) \leq 2$, the main condition can be rewritten equivalently as

$$k \geq \frac{2}{\bar{\omega}_T}\kappa\left(1 + \frac{2\sigma^2}{\mu\epsilon L}\right)\log\left(\frac{2\epsilon_0}{\epsilon}\right).$$

6.F Discussions on Optimizing Parameters of Stochastic Sparsification

In this section, we show how to tune the parameters of stochastic sparsification p to maximize the descent direction. In optimization formulation, for a fixed sparsity budget T we obtain the optimal probabilities $p^\star$ by solving the following problem [96]

$$\begin{aligned} &\underset{p\in[0,1]^d}{\text{maximize}} \quad \omega_p(T) \\ &\text{subject to} \quad \sum_{j=1}^{d} p^j = T. \end{aligned} \tag{6.19}$$

Here, $\omega_p(T)$ can be rewritten as

$$\omega_p(T) = \frac{\|g\|^2}{\mathbf{E}\|Q_{T,p}(g)\|^2} = \frac{\|g\|^2}{\sum_{j=1}^{d}(g^j)^2/p^j}.$$

This minimization problem (6.19) has the optimal solution

$$p^\star = ((p^1)^\star, (p^2)^\star, \ldots, (p^d)^\star),$$

which is on the form [96]

$$(p^i)^\star = \begin{cases} 1 & \text{if } i = 1, \ldots, n_s \\ |\lambda^i|(s-n_s)/\sum_{j=n_s+1}^{n}|\lambda^j| & \text{if } i = n_s+1, \ldots, n \end{cases}$$

where $\lambda = [\lambda^1, \ldots, \lambda^d]^T$ is the vector mapped from the gradient g such that $|\lambda^1| \geq \ldots \geq |\lambda^d|$, and n_s is selected such that $(p^i)^\star$ is bounded above by 1. This observation results in many algorithms to compute the optimal probabilities (see e.g., Algorithm 1 in [96]).

6.G Descent Lemma for Multi-node Gradient Methods with Stochastic Sparsification

We include a descent lemma for distributed compressed gradient methods with stochastic sparsification in Equation (6.16), which are analogous to single-node gradient methods using stochastic sparsification.

Lemma 6.10. *Consider the problem of minimizing $F(x) = \sum_{i=1}^{n} F^i(x)/n$ where each $F^i(\cdot)$ is possibly non-convex and L-smooth. Furthermore, let $\omega(T) = (\sum_{i=1}^{n} 1/[n\omega^i(T^i)])^{-1}$, where $\omega^i(T^i)$ is the sparsification parameter of node i. Suppose that*

$$x^+ = x - \gamma \frac{1}{n}\sum_{i=1}^{n} Q_{T^i}(g^i(x;\xi^i)), \tag{6.20}$$

where each $g^i(x;\xi^i)$ is unbiased and has variance with respect to $\nabla F(x)$ bounded by σ^2. If $\gamma = \omega(T)/(2L)$, then

$$\mathbf{E}F(x^+) \leq \mathbf{E}F(x) - \frac{\omega(T)}{4L}\mathbf{E}\|\nabla F(x)\|^2 + \frac{\omega(T)}{4L}\sigma^2.$$

Proof. By Cauchy-Schwartz's inequality, we can show $F(x) = \sum_{i=1}^n F^i(x)/n$ is also L-smooth. Using the smoothness assumption (Lemma 1.2.3. of [166]) and Equation (6.16), we have

$$F(x^+) \leq F(x) - \frac{\gamma}{n}\sum_{i=1}^n \langle \nabla F(x), Q_{T^i}(g^i(x;\xi^i))\rangle + \frac{L\gamma^2}{2n}\sum_{i=1}^n \|Q_{T^i}(g^i(x;\xi^i))\|^2.$$

Since $\omega^i(T^i)$ is defined by

$$\omega^i(T^i) = \frac{||g^i(x;\xi^i)||^2}{\mathbf{E}||Q_{T^i}(g(x;\xi^i))||^2},$$

we can easily show that

$$\frac{1}{n}\sum_{i=1}^n \|Q_{T^i}(g^i(x;\xi^i))\|^2 \leq \frac{2}{\omega(T)}\|\nabla F(x)\|^2 + \frac{2}{\omega(T)}\sigma^2, \tag{6.21}$$

where $\omega(T) = (\sum_{i=1}^n 1/[n\omega^i(T^i)])^{-1}$ and $\omega^i(T^i)$ is the sparsification parameter of node i. This result follows from using the inequality $\|x+y\|^2 \leq 2\|x\|^2 + 2\|y\|^2$ with $x = g^i(x;\xi^i) - \nabla F(x)$ and $y = \nabla F(x)$, and from the fact that $\mathbf{E}\|g^i(x;\xi^i) - \nabla F(x)\|^2 \leq \sigma^2$.

Next, by taking the expectation and then using unbiased property of stochastic gradients, the fact that $F(x) = \sum_{i=1}^n F^i(x)/n$, and Inequality (6.21), we have

$$\mathbf{E}F(x^+) \leq \mathbf{E}F(x) - \left(\gamma - L\gamma^2/\omega(T)\right)\mathbf{E}\|\nabla F(x)\|^2 + L\gamma^2\sigma^2/\omega(T).$$

Choosing $\gamma = \omega(T)/(2L)$, we complete the proof.

□

6.H Additional Experiments on Logistic Regression over URL

In this section, we provide additional simulations of logistic regression problems over the URL data set with dimension $d = 3.2\cdot 10^6$. We fit a linear communication model based on measurements, and benchmark our CAT framework and the heuristic from [22] on gradient descent with dynamic sparsification together with quantization in the single-master, single-worker architecture.

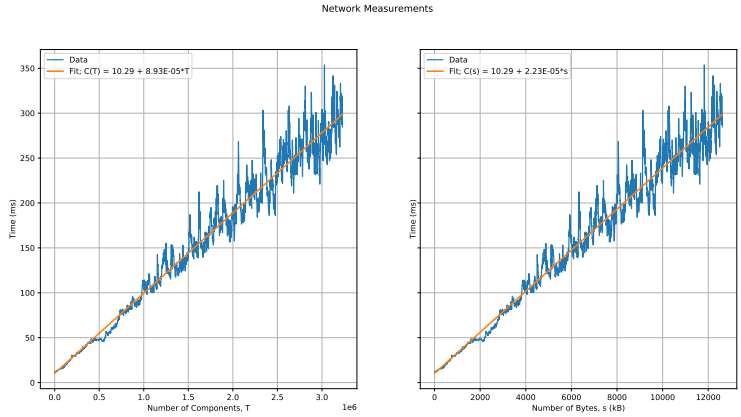


Figure 6.H.1: Time for communicating the vector with sparsity budget T (left) and its associated bytes (right).

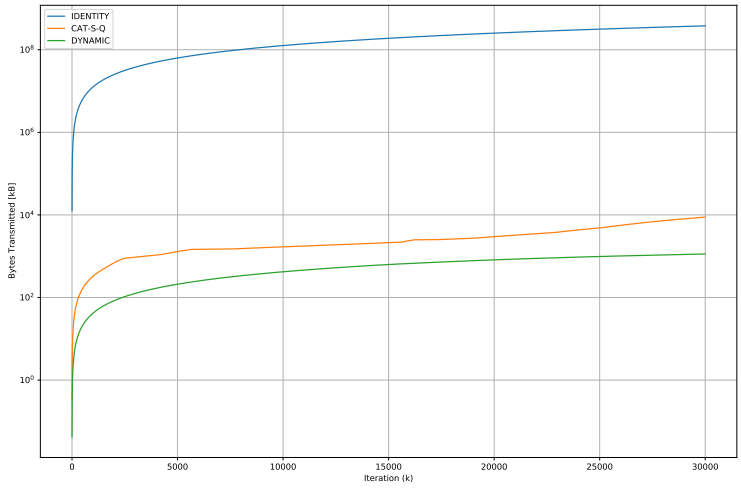


Figure 6.H.2: Gradient transmission (bytes) in each iteration k for full gradient descent (identity), CAT S+Q, and Alistarh's S+Q (dynamic).

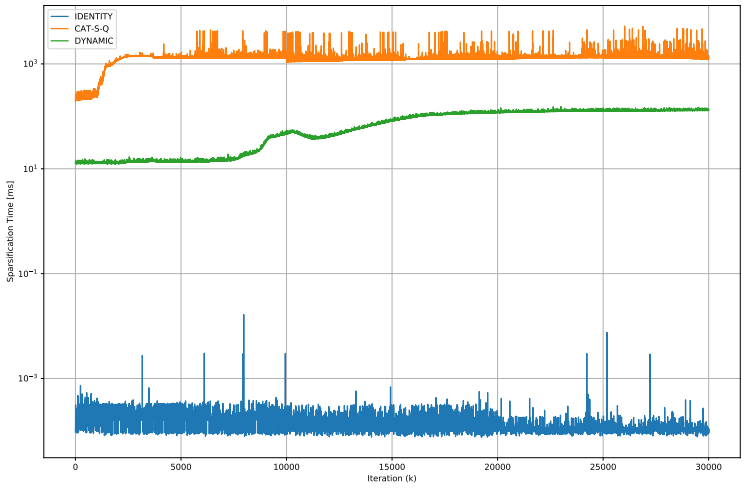


Figure 6.H.3: Sparsification time in each iteration k for full gradient descent (identity), CAT S+Q, and Alistarh's S+Q (dynamic).

In Figure 6.H.1 each blue data point represents the average of ten measurements of the end-to-end transmission time of a sparsified gradient with sparsity budget $T = \{1000, 2000, 3000, \ldots, d\}$. The orange lines demonstrate that an affine communication model is able to capture the communication cost. In retrospect, the affine behaviour should be expected, since we use the `ZMQ` library which initiates TCP communication once, and then reuses the communication together with buffers to optimize message transmission. Our ability to capture the communication cost (time) with an affine model indicates CAT that the framework could provide near-optimal performance in terms of communication time. Similarly for energy-constrained applications in IoT devices we can indeed investigate how the energy spent is related to the information transmitted. Utilizing this characteristics, our CAT framework can communicate information efficiently with low energy costs.

To illustrate how the CAT framework reduces communication cost and wall-clock time to reach target solution accuracy, we compared CAT S+Q and Alistarh's S+Q (dynamic), against full gradient descent. From Figure 6.H.2, CAT S+Q and Alistarh's S+Q reduce communication costs by 4 and 5 orders of magnitudes, respectively, compared to full gradient descent. However, we also observe that sparsification time of CAT S+Q is higher than Alistarh's S+Q by roughly an order of magnitude, as shown in Figure 6.H.3. Interest-

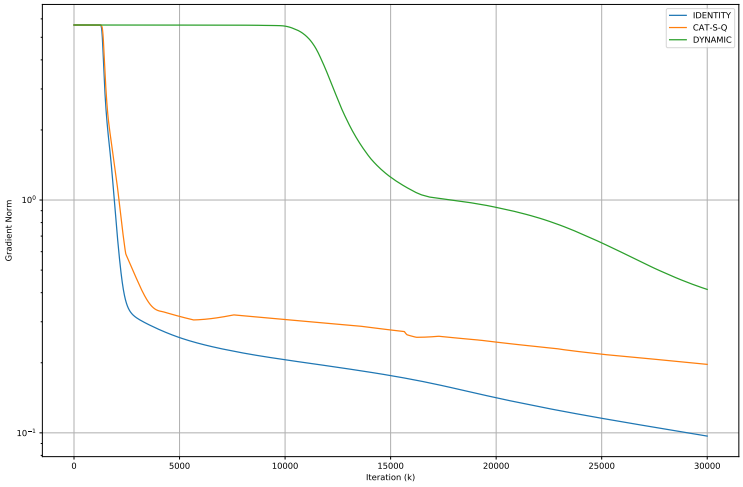


Figure 6.H.4: Convergence performance in the gradient norm $\|\nabla F(x^k)\|$ for full gradient descent (identity), CAT S+Q, and Alistarh's S+Q (dynamic).

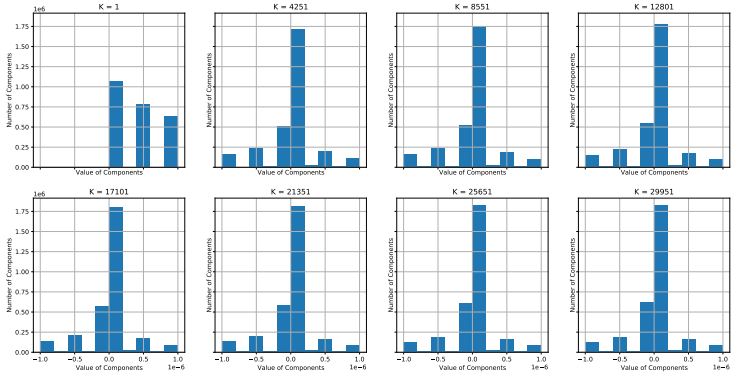


Figure 6.H.5: Histogram of the gradient elements when CAT S+Q is run for $30,000$ iterations.

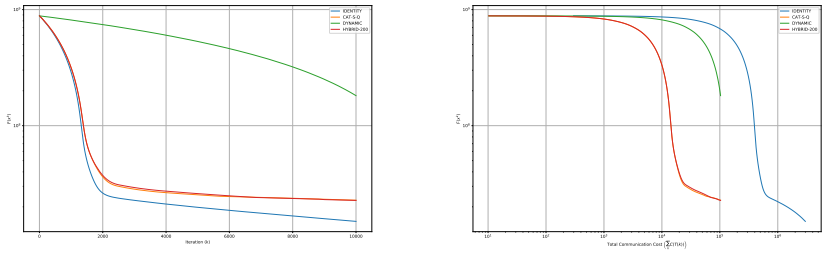


Figure 6.H.6: Performance with respect to iteration counts (left) and to communication costs (right) for different algorithms. We evaluated full gradient descent (identity), CAT S+Q, Alistarh's S+Q (dynamic) and the hybrid algorithm that uses CAT framework for every 200 iterations (hybrid-200).

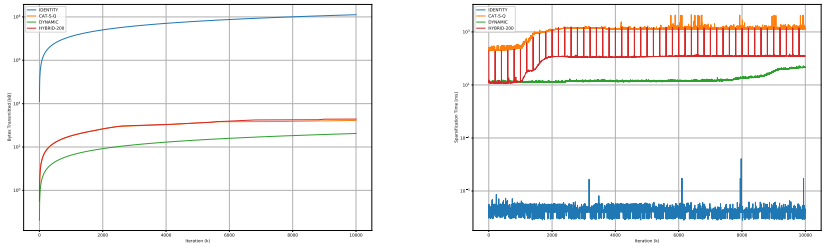


Figure 6.H.7: Gradient transmission in bytes (left) and sparsification time in each iteration (right) for different algorithms. We evaluated full gradient descent (identity), CAT S+Q, Alistarh's S+Q (dynamic) and the hybrid algorithm that uses CAT framework for every 200 iterations (hybrid-200).

ingly, the sparsification times for both algorithms increase as iteration counts grow. This happens because the sorting strategy in C++ leads to a worse performance especially when the gradient elements become more homogeneous. After running CAT S+Q for $30,000$ iterations, we observe in Figure 6.H.5 that gradient information after iteration $K = 4251$ has very similar histograms. In this case, sorting from scratch at each iteration is inefficient.

Based on these observations, we propose to let the CAT framework update the sparsity budget every S iterations (rather than every iteration) and then keep the sparsity budget fixed in the iterates between CAT recalculations. Figure 6.H.6 shows that this hybrid heuristic with $S = 200$ (i.e. rather infrequent updates) achieves only slightly worse loss function convergence with respect to iteration count and communication cost. From Figure 6.H.7, despite almost

the same data transmission size in each iteration, CAT S+Q reduces the time to sparsify the gradients by roughly an order of magnitude.

Chapter 7

Improved Step-Size Schedules for Noisy Gradient Methods

Optimization problems cover applications in many fields, including signal processing, machine learning and control. These problems are always relying on more and more data, both in terms of a number of data points and decision variables. To solve such large-scale optimization problems quickly, parallel and distributed optimization have become the methods of choice. Distributed optimization algorithms rely on splitting the computation load between multiple nodes to cooperatively compute the optimal solution. This makes the computational burden lighter, but at the same time it introduces some coordination challenges in the algorithms.

One of the most popular distributed algorithms is *stochastic gradient descent* (SGD). The idea of SGD is to approximate the gradients from a few random data points, and therefore the algorithm requires small memory footprint and low per-iteration cost. Even though SGD reduces computation times, the main parameter for guaranteeing fast convergence is the *step-size*. Several approaches of selecting the step-size have been proposed in recent years. Using fixed step-sizes on SGD guarantees the convergence toward a sub-optimal solution. To ensure the convergence toward the exact optimum, one common approach is to use decreasing step-sizes. The idea is to use large step-sizes initially, when the gradient is large compared to the noise. Then, decreasing the step-sizes as the algorithm progresses enables high solution accuracy. However, the key challenge of this approach is to decide how to decrease the step-sizes to ensure optimal convergence behaviors. Many decreasing step-size schedules have been studied and shown to improve the convergence of SGD in both theory and practice. These ideas were elegantly analyzed by Polyak (see [112, chapter 4]), and have been polished by many authors since then, e.g., [169, 170, 171, 172, 173, 174, 38, 142].

Although SGD has been extensively studied, recent optimization algorithms using gradients with noise have arisen for solving state-of-the-art learning problems. However, the benefits of diminishing step-sizes have not been reaped in these noisy gradient algorithms. For instance, before the gradients are communicated in these algorithms, they are injected with different types of noise, e.g., due to compression, coding schemes and function evaluations. Compressed gradient algorithms use compression operators to lower the precision of the gradients, thus reducing the costs of transmitting the gradients among the machines connected with a power-limited digital chan-

nel [22, 97, 175, 95, 82, 116]. Stochastic coding algorithms introduce redundancy of (stochastic) gradient computations among the machines to alleviate the impact of stragglers (or the slowest workers) on the convergence performance [45, 176, 177]. Zeroth-order algorithms that estimate the gradients from function values are popular in solving problems with black-box objective functions [178, 179] or deep learning problems for finding adversarial examples [66, 68, 180].

Contributions: The goal of this paper is to provide a theoretical justification on the benefit of using diminishing step-sizes for noisy gradient algorithms in general. Our analysis is based on two classes of systems that capture how the step-sizes influence the convergence behaviors of many algorithms. Our key results prove that the diminishing schedules enable the algorithms to converge at the optimal rate. We illustrate how these schedules can be used to improve the convergence performance of three popular noisy gradient algorithms: stochastic compression algorithms, stochastic coding algorithms and zeroth-order optimization algorithms. Finally, numerical experiments on stochastic compression algorithms validate the improved performance of using the diminishing step-sizes, in terms of solution accuracy.

Notation: We let g_k^i be a stochastic gradient with respect to the objective function $F^i(x_k)$, which is unbiased if $\mathbf{E}[g_k^i] = \nabla F^i(x_k)$ and which satisfies a bounded gradient and bounded variance property if, respectively,

$$\mathbf{E}\|g_k^i\|^2 \leq C^2, \quad \text{and} \tag{7.1}$$

$$\mathbf{E}\|g_k^i - \nabla F^i(x_k)\|^2 \leq \sigma^2. \tag{7.2}$$

7.1 Step-size Lemmas for Perturbed Sequences

In this section, we show how to tune the optimal step-size schedules for two classes of systems that are commonly used to analyze the performance of noisy gradient algorithms.

7.1.1 Contractive System With Noise

When the deterministic algorithm defines a contraction, noise often enters in a way that allows the iterates to be described by a non-negative sequence $\{V_k\}$ that satisfies

$$V_{k+1} \leq (1 - A\gamma_k)V_k + \gamma_k^2 B, \quad \text{for} \quad k \in \mathbb{N}. \tag{7.3}$$

Here, V_k is the quantity that we want to converge (e.g. objective function value or iterate distance to optimum), γ_k is a step size or learning rate, and A, B are positive scalars. The step-size γ_k is the key parameter that we can tune to influence the convergence speed of the algorithm. Ideally, we would like to choose γ_k optimally to ensure that the convergence is as fast as possible. If we know A and B and can monitor V_k, then the γ_k that ensures the fastest

decrease of V_k can be found analytically by minimizing the right hand side of Eq. (7.3) with respect to γ_k. By noticing that

$$g(\gamma) = (1 - A\gamma)V_k + \gamma^2 B$$

is convex function, we see that the optimal solution is $\gamma_k = AV_k/(2B)$. By plugging this step-size into Eq. (7.3),

$$V_{k+1} \leq V_k - \frac{A^2}{4B} V_k^2.$$

Applying Eq. (17) from [112, Lemma 6] with $\alpha_k = A^2/(4B)$ and $p = 1$, we can therefore prove that

$$V_k \leq \frac{V_0}{1 + A^2 V_0 k/(4B)} \leq \frac{4B}{A^2 k}. \tag{7.4}$$

This fastest $\mathcal{O}(1/k)$ rate needs the step-size to monitor V_k. This step-size cannot, in general, be implemented in practice. Ideally, we would like to choose the step-size without this knowledge while still attaining similar convergence behavior. By using Eq. (7.4), γ_k is shown to satisfy

$$\gamma_k = \frac{AV_k}{2B} \leq \frac{2A^{-1}}{k}.$$

We could therefore hope that setting $\gamma_k = 2A^{-1}/k$ would result in a convergence rate proportional to $4B/(A^2 k)$. This is indeed possible in general, as shown in the following result.

Lemma 7.1. *Consider the system in Eq.* (7.3) *and choose the step-size* $\gamma_k = \min\{\gamma, \alpha/(k+1)\}$ *where* $\gamma \in (0, 1/A)$ *and* $\alpha > 0$. *Let* $k^* = \max\{0, \alpha/\gamma - 1\}$, $V_0^* = (k^*+2)^{A\alpha}((1-A\gamma)^{(k^*+1)}V_0 + B\gamma/A)$, $\nu = (1 + 1/(k^*+2))^2$, *and* $k > k^*$. *Then,*

1. *if* $1/A < \alpha < 2/A$, *then*

$$V_k \leq \frac{B\alpha^2\nu}{(A\alpha - 1)} \frac{1}{(k+1)} + \frac{V_0^*}{(k+1)^{A\alpha}} + \frac{B\alpha^2\nu}{(k+1)^2}. \tag{7.5}$$

2. *if* $\alpha = 2/A$, *then*

$$V_k \leq \frac{4B}{A^2(k+1)} + \frac{V_0^* + 2\ln(k+1) + 2}{(k+1)^2}. \tag{7.6}$$

Proof. See Appendix 7.A.1.1. □

Lemma 7.1 establishes the $\mathcal{O}(1/k)$ convergence rate for V_k, and hence for any algorithm which produces the iterates that satisfy (7.3). The step-size schedule ensuring this convergence rate does not need to monitor V_k or to know B from (7.3). Nevertheless, as shown in Eq. (7.6), for $\alpha = 2/A$ we get

$$V_k \leq \frac{4B}{A^2(k+1)} + o\left(\frac{1}{k}\right),$$

which is comparable to the rate in Eq. (7.4) obtained by setting $\gamma_k = AV_k/(2B)$. Lemma 7.1 can also be applied to obtain the best known complexity bounds of some optimization algorithms. For instance, consider stochastic gradient descent for strongly convex problems, which satisfies Eq. (7.3) with $V_k = \mathbf{E}\|x_k - x^\star\|^2$, $A = \mu$ and $B = 2\max_{i\in[1,n]} \mathbf{E}\|\nabla F^i(x^\star)\|^2$. Lemma 7.1 will then give us the $\mathcal{O}(1/k)$ rate, which is comparable to the best known results in, e.g., [172, Section 3], [142] or [112, Chapter 4]. Furthermore, Lemma 7.1 extends directly to a number of other popular noisy gradient algorithms, and can be used to improve their performance in both theory and practice (see Section 7.2).

7.1.2 Non-expansive system with noise

Many iterative algorithms define non-expansive operators. In the presence of noise, they can often be described by non-negative sequences $\{V_k\}$ and $\{W_k\}$ that satisfy

$$V_{k+1} \leq V_k - \gamma_k W_k + \gamma_k^2 B, \quad \text{for} \quad k \in \mathbb{N}. \tag{7.7}$$

Here, W_k is the quantity that we want to converge, and V_k is typically a related quantity that is used to bound the convergence rate. In the applications that we consider, $W_k \leq cV_k$ for some positive constant c, so it makes sense to choose γ_k to minimize the right-hand side of (7.7). This suggests the step-size policy $\gamma_k = W_k/(2B)$ and that

$$V_{k+1} \leq V_k - \frac{1}{4B} W_k^2$$

Summing $k+1$ consecutive inequalities of this form, dividing by $1/(k+1)^2$ and taking square roots yields the bound

$$\frac{1}{k+1}\sum_{l=0}^{k} W_l \leq \frac{2\sqrt{B}\sqrt{V_0 - V_{k+1}}}{\sqrt{k+1}}. \tag{7.8}$$

The above argument has some limitations: we minimize an upper bound of W_{k+1}, disregard the fact that V_{k+1} must be non-negative, and obtain a step-size policy which makes the impractical assumption that we can monitor W_k. Nevertheless, the next result shows that the same rate is attainable with a step-size on the form $\gamma_k = \alpha/\sqrt{k+1}$:

Lemma 7.2. *Consider the system in Eq.* (7.7) *and has the step-size* $\gamma_k = \alpha/\sqrt{k+1}$ *for* $\alpha > 0$. *If* $V_k \leq R^2$, *then*

$$\frac{1}{k+1}\sum_{l=0}^{k} W_l \leq \frac{R^2/\alpha + \alpha B}{\sqrt{k+1}}. \tag{7.9}$$

Proof. See Appendix 7.A.1.2. □

Lemma 7.2 proves the $\mathcal{O}(1/\sqrt{k})$ rate in W_k for any iterates that satisfy Eq. (7.7) with $V_k \leq R^2$. The $\mathcal{O}(1/\sqrt{k})$ step-size ensuring this rate does not require the knowledge of W_k or B. In addition, from Lemma 7.2 with $\alpha = R/\sqrt{B}$ and $R > 0$, the system in Eq. (7.7) with $V_k \leq R^2$ would satisfy:

$$\frac{1}{k+1}\sum_{l=0}^{k} W_l \leq \frac{2\sqrt{B}}{\sqrt{k+1}}R.$$

This rate at $2\sqrt{B}R/\sqrt{k+1}$ matches the one in Eq. (7.8) when $V_k \leq R^2$. Also notice that the system in Eq. (7.7) with $V_k \leq R^2$ is satisfied by projection algorithms for problems over bounded, closed and convex constraints. For instance, if the system has $V_k = \mathbf{E}\|x_k - x^\star\|^2$ and the constraint set $X \subset \mathbb{R}^d$ with diameter $D = \max_{x_1,x_2\in X}\|x_1 - x_2\|$, then $V_k \leq R^2$ where $R = D$.

For the system in Eq. (7.7) in the absence of the assumption that $V_k \leq R^2$, the next lemma proves that choosing the diminishing step-size can guarantee the near $\mathcal{O}(1/\sqrt{k})$ rate.

Lemma 7.3. *Consider the system in Eq.* (7.7) *and has the step-size* $\gamma_k = \alpha/(k+1)^\beta$ *for* $\alpha > 0$ *and* $\beta \in [0, 1)$.

1. *If* $\beta = 1/2$, *then for all* $k \in \mathbb{N}$

$$\frac{1}{k+1}\sum_{l=0}^{k} W_l \leq \frac{V_0 - V_{k+1}}{\alpha\sqrt{k+1}} + B\alpha\frac{1+\ln(k+1)}{\sqrt{k+1}}. \tag{7.10}$$

2. *If* $\beta = 1/2 + \xi$ *and* $\xi \in (0, 1/2)$, *then for all* $k \in \mathbb{N}$

$$\frac{1}{k+1}\sum_{l=0}^{k} W_l \leq \frac{V_0 - V_{k+1}}{\alpha(k+1)^{1/2-\xi}} + B\alpha\frac{1+1/(2\xi)}{(k+1)^{1/2-\xi}}. \tag{7.11}$$

Proof. See Appendix 7.A.1.3. □

This lemma provides the near $\mathcal{O}(1/\sqrt{k})$ rate for any system in Eq. (7.7) without the condition that $V_k \leq R^2$. The system using $\gamma_k = \alpha/(k+1)^\beta$ with $\beta = 1/2$ and $\beta = 1/2 + \xi$ for $\xi \in (0, 1/2)$ enjoys the $\mathcal{O}(\ln(k)/k^{1/2})$ and

$\mathcal{O}(1/k^{1/2-\xi})$ rate, respectively. These rates are thus slightly slower than the $\mathcal{O}(1/\sqrt{k})$ rate from Lemma 7.2.

Similarly to Lemma 7.1, this result can be used to recover some of the best known complexity bounds of some noisy gradient algorithms. For example, stochastic gradient descent for convex and L-smooth problems satisfies Eq. (7.7) with $V_k = \mathbf{E}\|x_k - x^\star\|^2$, $W_k = \mathbf{E}[F(x_k) - F(x^\star)]$ and $B = 2\max_{i\in[1,n]} \mathbf{E}\|\nabla F^i(x^\star)\|^2$. By the L-smoothness assumption the algorithm also satisfies $W_k \leq cV_k$ with $c = L/2$. Thus, Lemma 7.3 guarantees an $\mathcal{O}(\ln(k)/k^{1/2})$ rate comparable to [181, Corollary 4.2], and an $\mathcal{O}(1/k^{1/2-\xi})$ rate similar to [182, Corollary 2.4], respectively.

Next, we will show that our convergence results can be used to improve the convergence guarantees and practical performance of several other noisy gradient algorithms.

7.2 Applications

In this section, we will illustrate how our step-size results can be used to improve the complexity bounds of three noisy gradient algorithms: stochastic gradient compression, stochastic coding and zeroth-order algorithms. Throughout this paper, we consider problems on the form:

$$\min_{x\in\mathbb{R}^d} \; P(x) := F(x) + h(x), \tag{7.12}$$

where $F(x)$ is the average of many component functions, i.e.

$$F(x) = \frac{1}{n}\sum_{i=1}^{n} F^i(x) \tag{7.13}$$

and $h(x)$ is a convex but possibly non-smooth function. We also impose the following assumptions.

Assumption 7.1. Each $F^i(x)$ is L-smooth, i.e. there exists a positive constant L such that $\forall x, y \in \mathbb{R}^d$

$$\|\nabla F^i(x) - \nabla F^i(y)\| \leq L\|x - y\|.$$

Assumption 7.1 implies that $F(x)$ is also L-smooth.

Assumption 7.2. The function $F(x)$ is μ-strongly convex, i.e. there exists a positive constant μ such that $\forall x, y \in \mathbb{R}^d$

$$F(y) \geq F(x) + \langle \nabla F(x), y - x\rangle + \frac{\mu}{2}\|y - x\|^2.$$

7.2.1 Proximal Stochastic Compression Algorithms

When gradient algorithms are distributed in an attempt to deal with larger problem instances, communication of gradient information between nodes quickly becomes the bottleneck. This is particularly apparent in neural network training, where state-of-the art models can have millions of parameters. In the training of neural network models such as AlexNet, VGG and LSTM, gradient communication can account for up to 80% of the total running time [22]. To reduce this communication overhead, gradients are often compressed before transmission. In particular, we consider proximal stochastic compression algorithms (`Prox-SComp`) that update the iterate x_k via:

$$x_{k+1} = \mathbf{prox}_{\gamma_k h}\left(x_k - \frac{\gamma_k}{n}\sum_{i=1}^{n} Q(g_k^i)\right). \tag{7.14}$$

Here, γ_k is a positive step-size and g_k^i is the stochastic gradient with respect to the local objective function $F^i(x_k)$. Also, the operator $Q : \mathbb{R}^d \to \mathbb{R}^d$ is the unbiased random quantizer, which satisfies unbiased and variance-bounded properties, i.e. there exists a positive scalar q such that for all $x \in \mathbb{R}^d$

$$\mathbf{E}[Q(x)] = x, \quad \text{and} \quad \mathbf{E}\|Q(x)\|^2 \leq q\|x\|^2. \tag{7.15}$$

The variance-bounded condition implies high precision of the compressor when q is close to 1. Examples of the stochastic compressors include stochastic sparsification:

$$[Q_p(v)]_i = (v_i/p_i)\xi_i, \quad \forall i \in \{1, 2, \ldots, d\} \tag{7.16}$$

where $\xi_i \sim \text{Bernouli}(p_i)$. The stochastic sparsification in Eq. (7.16) satisfies the unbiased and variance-bounded conditions in Eq. (7.15) with $q = 1/p_{\min}$ and $p_{\min} = \min_{i\in[1,d]} p^i$ [30]. There are many choices to tune p_i. For instance, we obtain `QSGD` in [22] with $s = 1$, `TernGrad` in [95] and the ℓ_q-quantizer in [96], when we set $p^i = |v^i|/\|v\|_r$ with $r = 2$, $r = \infty$, and $r \in (0, \infty]$, respectively. The probabilities p_i can also be fine-tuned adaptively to maximize the solution accuracy [96], or the communication efficiency [183].

The next result demonstrates how the convergence results in Section 7.1 can be used to suggest effective step-size policies for `Prox-SComp`.

Theorem 7.1. *Consider the sequence $\{x_k\}_{k\in\mathbb{N}}$ generated by `Prox-SComp` in Eq. (7.14) for solving Problem (7.12) under Assumption 7.1.*

1. *(Non-convex problems) If $\mathbf{E}[P_{1/\bar{\rho}}(x_k) - P_{1/\bar{\rho}}(x^\star)] \leq R^2$, g_i^k satisfies Eq. (7.1), and $\gamma_k = (\bar{\rho} - L)^{-1}/\sqrt{k+1}$ with $\bar{\rho} > L$, then*

$$\min_{l\in[0,k]} \mathbf{E}\|\nabla P_{1/\bar{\rho}}(x_l)\|^2 \leq \frac{2\bar{\rho}^2R^2 + \bar{\rho}qC^2}{(\bar{\rho} - L)\sqrt{k+1}}.$$

2. *(Strongly convex problems) If Assumption 7.2 also holds, g_i^k satisfies Eq. (7.2), and $\gamma_k = \min\{\gamma, \alpha/(k+1)\}$ with $\gamma = (3q+1)^{-1}/(2L)$ and $\alpha = 2/\mu$, then*

$$\mathbf{E}\|x_k - x^\star\|^2 \leq \frac{4B}{\mu^2(k+1)} + \frac{V_0^* + 2\ln(k+1) + 2}{(k+1)^2},$$

for $k > k^*$. Here, k^* and V_0^* are defined in Lemma 7.1-2) with $A = \mu$ and $B = 3q[\sigma^2 + \sum_{i=1}^n \mathbf{E}\|\nabla F^i(x^\star)\|^2/n]$.

Proof. See Appendix 7.A.3. □

This theorem characterizes the impact of the compression level q and the problem parameters μ, L on the convergence rate for `Prox-SComp`. The algorithms enjoy the $\mathcal{O}(k^{-1})$ and $\mathcal{O}(k^{-1/2})$ rate for strongly convex and nonconvex problems, respectively. Diminishing step-sizes attaining these rates can improve existing convergence results for state-of-the-art compression algorithms that use often fixed step-size schedules [22, 183, 117]. To show this, consider strongly convex problems. On the one hand, from Theorem 7.1-2), `Prox-SComp` with $\gamma_k = \min\{\gamma, 2\mu^{-1}/(k+1)\}$ and $\gamma = (3q+1)^{-1}/(2L)$ reaches the ϵ-accurate solution by running at least

$$\max\left(\frac{16C_2/\mu^2 + 4}{\epsilon}, \sqrt{\frac{2V_0^* + 4}{\epsilon}}\right) \quad \text{iterations.}$$

On the other hand, from [183, Theorem 3] `Prox-SComp` with $h(x) = 0$ using the fixed step-size $\gamma_k = 0.5/(q(\beta + L))$ and $\beta = 2\sigma^2/(\mu\epsilon)$ requires at least

$$\frac{2qL}{\mu}\log\left(\frac{2\epsilon_0}{\epsilon}\right) + \frac{4q\sigma^2}{\mu^2}\cdot\frac{1}{\epsilon}\log\left(\frac{2\epsilon_0}{\epsilon}\right) \quad \text{iterations}$$

where $\epsilon_0 = F(x_0) - F(x^\star)$. The algorithm using our proposed diminishing step-size ensures the exact optimum unlike [117, Theorem 5.2], and obtains the $\mathcal{O}(1/\epsilon)$ iteration complexity that is lower than the $\mathcal{O}((1/\epsilon)\log(1/\epsilon))$ complexity of the algorithm without the proximal operator that uses the fixed step-size by [183, Theorem 3-3.]. The benefit of using the diminishing step-size over the fixed one is also confirmed by our numerical experiments in Section 7.3.

7.2.2 Proximal Stochastic Coding Algorithms

When distributed algorithms are executed in a synchronous manner, their performance is determined by the slowest nodes, sometimes referred to as *stragglers*. To mitigate the impact of stragglers, various coding schemes have been introduced [45, 176, 177]. These schemes assign redundant data across the

nodes, and enable the synchronous methods to approximate the full gradient aggregated only from non-straggling nodes [45, 176, 177]. In some scenarios, these coding algorithms can attain a significantly faster convergence rate than traditional synchronous algorithms.

To this end, we consider proximal stochastic coding algorithms (`Prox-SCod`). At each iteration k of the algorithms, each node computes the coded gradient

$$G_j = \sum_{i=1}^{n} \frac{D_{ij}}{\bar{\alpha}_i} g_k^i,$$

where g_k^i is the stochastic gradient with respect to $F^i(x_k)$. Here, $\bar{\alpha}_i = \sum_{j=1}^{m} D_{ij}$ and $D \in \mathbb{R}^{n\times m}$ is an assignment matrix of data points to nodes, i.e. $D_{ij} \neq 0$ if data point i is available on node j for $i = 1, 2, \ldots, n$ and $j = 1, 2, \ldots, m$, and $D_{ij} = 0$ otherwise. Then, the server aggregates the gradients from the nodes with probability $1 - p$, and updates the iterate x_k according to:

$$x_{k+1} = \mathbf{prox}_{\gamma_k h}\left(x_k - \frac{\gamma_k}{n(1-p)} \sum_{j=1}^{m} w_j G_j\right). \tag{7.17}$$

Here, $w \in \mathbb{R}^m$ represents a decoding coefficient vector where $w_j = 1$ if the server receives the gradient from node j, and 0 otherwise. `Prox-SCod` in Eq. (7.17) can be expressed equivalently as:

$$x_{k+1} = \mathbf{prox}_{\gamma_k h}\left(x_k - \frac{\gamma_k}{n(1-p)} \sum_{i=1}^{n} \frac{\alpha_i}{\bar{\alpha}_i} g_k^i\right), \tag{7.18}$$

where $\alpha_i = \sum_{j=1}^{m} D_{ij} w_j$. Also notice that the coded gradient information is unbiased, i.e.

$$\mathbf{E}\left\{\frac{1}{n(1-p)} \sum_{i=1}^{n} \frac{\alpha_i}{\bar{\alpha}_i} g_k^i\right\} = \nabla F(x_k).$$

Similarly to `Prox-SComp`, we can obtain the convergence results for `Prox-SCod` by using the lemmas in Section 7.1.

Theorem 7.2. *Consider the sequence $\{x_k\}_{k\in\mathbb{N}}$ generated by `Prox-SCod` in Eq. (7.18) for Problem (7.12) under Assumption 7.1. Denote $\beta = m \max_i \alpha_i$ and $\alpha_i = \sum_{j=1}^{m} D_{ij}^2 / (\sum_{j=1}^{m} D_{ij})^2$ for the matrix $D \in \mathbb{R}^{m\times n}$.*

1. *(Non-convex problems) If $\mathbf{E}[P_{1/\bar{\rho}}(x_k) - P_{1/\bar{\rho}}(x^\star)] \leq R^2$, g_i^k satisfies Eq. (7.1) and $\gamma_k = (\bar{\rho} - L)^{-1}/\sqrt{k+1}$ with $\bar{\rho} > L$, then*

$$\min_{l\in[0,k]} \mathbf{E}\|\nabla P_{1/\bar{\rho}}(x_l)\|^2 \leq \frac{2\bar{\rho}^2 R^2 + \bar{\rho}\beta(1-p)^{-1}C^2}{(\bar{\rho} - L)\sqrt{k+1}}.$$

2. *(Strongly convex problems) If Assumption 7.2 also holds,* g_i^k *satisfies Eq.* (7.2)*, and* $\gamma_k = \min\{\gamma, \alpha/(k+1)\}$ *with* $\gamma = (3\beta/(1-p)+1)^{-1}/(2L)$ *and* $\alpha = 2/\mu$*, then*

$$\mathbf{E}\|x_k - x^\star\|^2 \leq \frac{4B}{\mu^2(k+1)} + \frac{V_0^* + 2\ln(k+1) + 2}{(k+1)^2},$$

for $k > k^*$*. Here,* k^* *and* V_0^* *are defined in Lemma 7.1-2) with* $A = \mu$ *and* $B = (3\beta/(1-p))[\sigma^2 + \sum_{i=1}^n \mathbf{E}\|\nabla F^i(x^\star)\|^2/n]$.

Proof. See Appendix 7.A.4. □

Theorem 7.2 shows dependency of the convergence rate for `Prox-SCod` on the stochastic coding scheme β and the communication model p. From Theorem 7.2-2), `Prox-SCod` obtains the same $\mathcal{O}(k^{-1})$ rate as the stochastic coding algorithms without the proximal operator for strongly convex problems by Theorem 2 and 4 in [47]. Furthermore, to the best of our knowledge, Theorem 7.2-1) is the first result establishing the $\mathcal{O}(k^{-1/2})$ rate for the proximal stochastic coding algorithms using the decaying step-size on non-convex problems.

7.2.3 Proximal Zeroth-Order Algorithms

In some applications, gradients are very expensive to compute or even infeasible to obtain. This is the case when the objective function is computed from simulations, or in model-free approaches to decision-making where we only observe the objective function value for the actions that we take. In these applications it is common to use zeroth-order optimization techniques that estimate directional derivatives using finite differences. Here, we will consider proximal zeroth-order algorithms [178] which progress as follows:

$$x_{k+1} = \mathbf{prox}_{\gamma_k h}\left(x_k - \gamma_k G_{\tau_k}(x_k)\right) \quad \text{for } k \in \mathbb{N} \tag{7.19}$$

Here,

$$G_{\tau_k}(x_k) = \frac{F(x_k + \tau_k u_k) - F(x_k)}{\tau_k} u_k$$

is the finite difference estimation of the derivative in the direction $u_k \in \mathbb{R}^d$ generated from the Gaussian distribution with zero mean and unit variance, with a positive scalar τ_k. One can prove (cf. [178, 179]) that the finite difference derivative $G_\tau(x)$ satisfies

$$\mathbb{E}_u[G_\tau(x)] = \nabla F_\tau(x), \quad \text{and} \tag{7.20}$$

$$\mathbb{E}_u\|G_\tau(x)\|^2 \leq \frac{\tau^2 L^2 (d+6)^3}{2} + 2(d+4)\|\nabla F(x)\|^2, \tag{7.21}$$

where $F_\tau(x) = \mathbb{E}_u[F(x + \tau u)]$.

Although zeroth-order algorithms have been studied extensively (e.g. in [178, 179, 184]), very few convergence results exist for proximal zeroth-order algorithms. By applying the step-size lemmas in Section 7.1, we provide the following convergence results for these algorithms.

Theorem 7.3. *Consider the sequence $\{x_k\}_{k\in\mathbb{N}}$ generated by the proximal zeroth-order algorithms (7.19) with $\tau_k = \eta\gamma_k$ and $\eta > 0$ for Problem (7.12) under Assumption 7.1 and the condition that $\|\nabla F(x)\|^2 \leq C^2$ for $x \in \mathbb{R}^d$.*

1. *(Non-convex problems) If $\mathbf{E}[P_{1/\bar{\rho}}(x_k) - P_{1/\bar{\rho}}(x^\star)] \leq R^2$, and choose $\gamma_k = (2\bar{\rho}\sqrt{d+6})^{-1}/(k+1)^{1/2}$ with $\bar{\rho} \in (L - 1/2, 2L + 1]$, then*

$$\min_{l\in[0,k]} \mathbf{E}\|\nabla P_{1/\bar{\rho}}(x_l)\|^2 \leq \frac{\bar{\rho}}{\bar{\rho} - L - 1/2} \frac{R^2/\alpha + \alpha^2\tilde{C}}{\sqrt{k+1}},$$

   where $\tilde{C} = \eta^2L^2(d+3)^2/(16\bar{\rho}) + \eta^2L^2(d+6)^2/(8\bar{\rho}^2) + (2d+9)C^2$.

2. *(Strongly convex problems) If Assumption 7.2 also holds, and choose $\gamma_k = \min\{\gamma, \alpha/(k+1)\}$ with $\gamma = 1/[(d+6)L]$ and $\alpha = 2/\mu$, then*

$$\mathbf{E}\|x_k - x^\star\|^2 \leq \frac{4B}{\mu^2(k+1)} + \frac{V_0^* + 2\ln(k+1) + 2}{(k+1)^2},$$

   for $k > k^*$. Here, k^* and V_0^* are defined in Lemma 7.1-2) with $A = \mu$ and $B = \eta^2L(d+3)^2/(4\mu) + \eta^2(d+6) + 2(2d+9)C^2$.

Proof. See Appendix 7.A.5. □

From this theorem, proximal zeroth-order algorithms (7.19) with $\tau_k = \eta\gamma_k$ and $\eta > 0$ enjoy the $\mathcal{O}(k^{-1})$ and $\mathcal{O}(k^{-1/2})$ rate for strongly convex and non-convex problems, respectively. Furthermore, the methods for non-convex deterministic problems from Theorem 7.3-1) have tighter convergence bounds than the counterparts for non-convex stochastic problems by [185, Theorem 1]. We can illustrate this by setting $\phi_{1/\bar{\rho}}(x_0) - \phi_\star \leq R^2$, and $\alpha_k = \alpha(k+1)^{-1/2}$ for $k \in \mathbb{N}$ into Eq. (15) from [185, Theorem 1], which yields

$$\min_{l\in[0,k]} \mathbf{E}\|\nabla P_{1/\bar{\rho}}(x_l)\|^2 \leq \frac{2\bar{\rho}}{\bar{\rho} - L} \frac{R^2/\alpha + C_1(1 + \ln(k+1))}{\sqrt{k+1}},$$

for some positive scalars C. By using the step-size $\gamma_k = \alpha(k+1)^{-1/2}$ for $\alpha > 0$, the $\mathcal{O}(k^{-1/2})$ rate from Theorem 7.3-1) is faster than the $\mathcal{O}(\ln(k)k^{-1/2})$ rate from [185, Theorem 1].

In addition, from Theorem 7.3-1), the proximal zeroth-order algorithms (7.19) with $\gamma_k = (2\bar{\rho}\sqrt{d+6})^{-1}(k+1)^{-1/2}$ and $\eta = 1/\sqrt{d+6}$ for non-convex problems can reach the ϵ-accuracy within

$$\mathcal{O}\left(\frac{d+6}{\epsilon^2}\right) \quad \text{iterations.}$$

This $\mathcal{O}(d/\epsilon^2)$ complexity is the same as the complexity for the zeroth-order methods without the proximal operator on non-convex problems by Nesterov [178, Case 1. in Section 7].

7.3 Experimental Results

We now illustrate the convergence rate results for the suggested step-size schemes. In particular, we focus on `Prox-SComp` given in (7.14) in Section 7.2.1 with the `QSGD` quantizer [22] with quantization level s. We consider both strongly-convex logistic regression and non-convex robust linear regression problems. The results are averaged over three Monte Carlo runs using the `RCV1` data set [140] (with 20242 data points and 47236 features), which is uniformly distributed across $n = 3$ workers. We set the initial solution $x_0 = \mathbf{0}$ and the mini-batch size $b = 2$ for each worker. The compression parameter for `QSGD` is $q = 1 + \min(d/s^2, \sqrt{d}/s)$ where d is the problem dimension, similar as in [22, 117].

7.3.1 Strongly-convex Regularized Logistic Regression

Consider the strongly-convex regularized logistic regression problem, that is Problem (7.12) with $h(x) = \lambda_1\|x\|_1$ and

$$F^i(x) = \frac{1}{m}\sum_{j=1}^{m} \log(1 + \exp(-y_j^i \cdot \langle z_j^i, x\rangle) + \frac{\lambda_2}{2}\|x\|^2.$$

Here, $z_j^i \in \mathbb{R}^d$ is a feature vector for sample $j \in [1, m]$ at worker i with its associated label $y_j^i \in \{-1, 1\}$. We set the quantization level $s = 100$, and regularization parameters $\lambda_1 = 10^{-5}$ and $\lambda_2 = 10^{-4}$. We compared our proposed diminishing step-size from Theorem 7.1-2) against the fixed step-size $\gamma_k = (3q+1)^{-1}/(2L)$ and the step-size $\gamma_k = 1/(L \cdot k)$ proposed in [112, Theorem 3].

From Figure 7.1, `Prox-SComp` (7.14) with the step-size in Theorem 7.1-2) outperforms that with other step-sizes in terms of solution accuracy. At $k = 6 \cdot 10^5$, the step-size in Theorem 7.1-2) achieves the accuracy 5 times higher than the step-size $\gamma_k = (3q+1)^{-1}/(2L)$ and $5 \cdot 10^3$ times better than the step-size $\gamma_k = 1/(L \cdot k)$ from the step-size $\gamma_k = 1/(L \cdot k)$ in [112, Theorem 3].

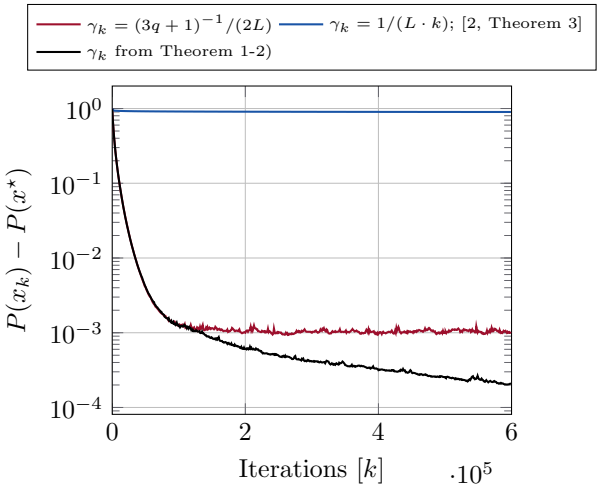


Figure 7.1: The performance of stochastic compression algorithms using different step-sizes and `QSGD` with 100 quantization levels for solving strongly-convex logistic regression on `RCV1`.

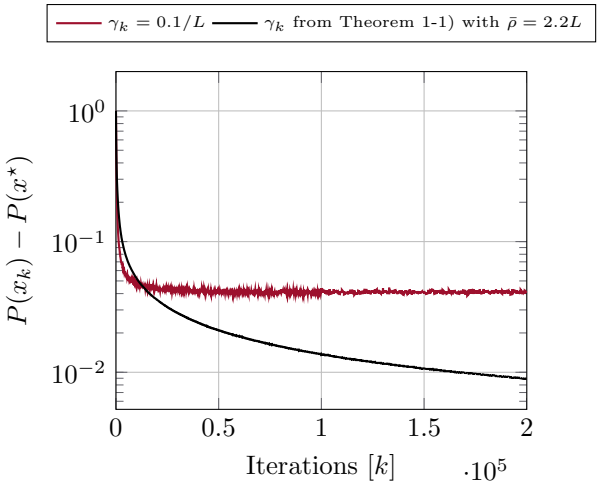


Figure 7.2: The performance of stochastic compression algorithms using different step-sizes and `QSGD` with 50 quantization levels for solving non-convex robust linear regression on `RCV1`.

7.3.2 Non-convex Robust Linear Regression

Next, consider the non-convex robust linear regression problem [138, 139], i.e. Problem (7.12) with $h(x) = \lambda_1 \|x\|_1$ and

$$F^i(x) = \frac{1}{m} \sum_{j=1}^{m} (\langle z_i^j, x \rangle - y_i^j)^2 / (1 + (\langle z_i^j, x \rangle - y_i^j)^2).$$

Here, we set $s = 50$ and $\lambda_1 = 10^{-5}$, and compared the proposed step-size from Theorem 7.1-1) with $\bar{\rho} = 2.2L$ against the fixed step-size $\gamma_k = 0.1/L$.

Figure 7.2 indicates a higher solution accuracy from `Prox-SComp` (7.14) using the step-size from Theorem 7.1-1) with $\bar{\rho} = 2.2L$ than the fixed step-size $\gamma_k = 0.1/L$. At $k = 2 \cdot 10^5$, the the accuracy attained by the fixed step-size $\gamma_k = 0.1/L$ is 5 times worse than the step-size from Theorem 7.1-1) with $\bar{\rho} = 2.2L$.

Appendix

7.A Proofs

7.A.1 Proof of Central Lemmas

In this section, we prove Lemmas 7.1, 7.2 and 7.3 for analyzing the convergence results of two main systems that characterize the performance of proximal noisy gradient algorithms.

7.A.1.1 Proof of Lemma 7.1

Let $k^* = \max\{0, \alpha/\gamma - 1\}$. We can show that $\gamma_k = \gamma$ for $0 \leq k \leq k^*$, while $\gamma_k = \alpha/(k+1)$ for $k > k^*$. For $0 \leq k \leq k^*$, applying the step size $\gamma_k = \gamma$ in Eq. (7.3) recursively gives

$$V_{k^*+1} \leq (1 - A\gamma)^{(k^*+1)} V_0 + \frac{B\gamma}{A}. \tag{7.22}$$

For $k > k^*$, plugging the step-size $\gamma_k = \alpha/(k+1)$ into Eq. (7.3) yields

$$V_{k+1} \leq \left(1 - \frac{A\alpha}{k+1}\right) V_k + \frac{B\alpha^2}{(k+1)^2}.$$

By applying the inequality recursively and utilizing $1 + x \leq \exp(x)$ for $x \in \mathbb{R}$,

$$\begin{aligned} V_{k+1} \leq \exp\left(-A\alpha \sum_{l=k^*+1}^{k} \frac{1}{l+1}\right) V_{k^*+1} \\ + B\alpha^2 \sum_{l=k^*+1}^{k} \frac{1}{(l+1)^2} \exp\left(-A\alpha \sum_{i=l+1}^{k} \frac{1}{i+1}\right). \end{aligned} \tag{7.23}$$

Since $1/(s+1)$ is decreasing in s, we have

$$\sum_{i=l+1}^{k} \frac{1}{i+1} \geq \int_{i=l+1}^{k+1} \frac{di}{i+1} = \ln(k+2) - \ln(l+2).$$

Plugging these inequalities into Eq. (7.23) yields

$$V_{k+1} \leq \frac{(k^*+2)^{A\alpha} V_{k^*+1}}{(k+2)^{A\alpha}} + \frac{B\alpha^2}{(k+2)^{A\alpha}} \sum_{l=k^*+1}^{k} \frac{(l+2)^{A\alpha}}{(l+1)^2}.$$

By Eq. (7.22) and the above inequality, for $k > k^*$

$$V_{k+1} \leq \frac{V_0^*}{(k+2)^{A\alpha}} + \frac{B\alpha^2}{(k+2)^{A\alpha}} \sum_{l=k^*+1}^{k} \frac{(l+2)^{A\alpha}}{(l+1)^2}, \tag{7.24}$$

where $V_0^* = (k^*+2)^{A\alpha}((1-A\gamma)^{(k^*+1)}V_0 + B\gamma/A)$. Hence, for $A\alpha > 1$

$$\begin{aligned}
\sum_{l=k^*+1}^{k} \frac{(l+2)^{A\alpha}}{(l+1)^2} &\leq \nu \sum_{l=k^*+1}^{k} (l+2)^{A\alpha-2} \\
&\leq \nu \int_{l=k^*+1}^{k} (l+2)^{A\alpha-2} dl + \nu(k+2)^{A\alpha-2} \\
&\leq \frac{\nu(k+2)^{A\alpha-1}}{A\alpha-1} + \nu(k+2)^{A\alpha-2},
\end{aligned}$$

where $\nu = (1 + 1/(k^*+2))^2$. Incorporating the above inequality into Eq. (7.24), we have

$$V_{k+1} \leq \frac{V_0^*}{(k+2)^{A\alpha}} + \frac{B\alpha^2\nu}{(A\alpha-1)} \frac{1}{(k+2)} + \frac{B\alpha^2\nu}{(k+2)^2}.$$

Finally, if we choose $\alpha = 2/A$, then Eq. (7.24) can be improved by the follow procedure:

$$\begin{aligned}
V_{k+1} &\leq \frac{V_0^*}{(k+2)^2} + \frac{B\alpha^2}{(k+2)^2} \sum_{l=k^*+1}^{k} \left(1 + \frac{2}{l+1} + \frac{1}{(l+1)^2}\right) \\
&\leq \frac{V_0^*}{(k+2)^2} + \frac{B\alpha^2}{(k+2)^2} \left(k - k^* + 3 + 2\int_{l=k^*+1}^{k+1} \frac{dl}{l} + \int_{l=k^*+1}^{k+1} \frac{dl}{l^2}\right) \\
&\leq \frac{V_0^*}{(k+2)^2} + \frac{B\alpha^2(k+3+2\ln(k+1)+\frac{1}{k^*+1}-k^*)}{(k+2)^2} \\
&\leq \frac{V_0^* + 2\ln(k+1) + 2}{(k+2)^2} + \frac{B\alpha^2}{k+2}.
\end{aligned}$$

We thus complete the proof.

7.A.1.2 Proof of Lemma 7.2

The recursion (7.7) can be equivalently expressed as

$$\frac{1}{\gamma_k} W_k \leq \frac{1}{\gamma_k^2}(V_k - V_{k+1}) + B. \tag{7.25}$$

By summing (7.25) over $l = 0, 1, \ldots, k$,

$$\sum_{l=0}^{k} \frac{1}{\gamma_l} W_l \leq \frac{1}{\gamma_0^2} V_0 + \left(\frac{1}{\gamma_1^2} - \frac{1}{\gamma_0^2} \right) V_1 + \left(\frac{1}{\gamma_2^2} - \frac{1}{\gamma_1^2} \right) V_2 + \ldots + \left(\frac{1}{\gamma_k^2} - \frac{1}{\gamma_{k-1}^2} \right) V_k - \frac{1}{\gamma_k^2} V_{k+1} + B(k+1). \tag{7.26}$$

Since $\alpha > 0$ and $\eta \geq 1$, γ_k is monotonically decreasing in k, i.e. $1/\gamma_{k+1}^2 \geq 1/\gamma_k^2$ for all $k \geq 0$. By the fact that $V_k \leq R^2$ for all $k \geq 0$ and by the consequence of the telescopic series, we have from (7.26) that

$$\sum_{l=0}^{k} \frac{1}{\gamma_l} W_l \leq \frac{1}{\gamma_k^2} R^2 - \frac{1}{\gamma_k^2} V_{k+1} + B(k+1).$$

Next, since V_k, W_k are non-negative functions and γ_k is decreasing in k, by re-arranging the terms we have

$$\frac{1}{k+1} \sum_{l=0}^{k} W_l \leq \frac{1}{\gamma_k (k+1)} R^2 + \gamma_k B. \tag{7.27}$$

Finally, by plugging $\gamma_k = \alpha/\sqrt{k+1}$ into (7.27),

$$\frac{1}{k+1} \sum_{l=0}^{k} W_l \leq \frac{1}{\alpha\sqrt{k+1}} R^2 + \frac{\alpha B}{\sqrt{k+1}}.$$

7.A.1.3 Proof of Lemma 7.3

We can rewrite the recursion (7.7) equivalently as:

$$\gamma_k W_k \leq (V_k - V_{k+1}) + \gamma_k^2 B. \tag{7.28}$$

By summing (7.7) over $l = 0, 1, \ldots, k$, by the monotonically decreasing function γ_k in k, and by the non-negativity of W_k,

$$\gamma_k \sum_{l=0}^{k} W_l \leq \sum_{l=0}^{k} \gamma_l W_l \leq (V_0 - V_{k+1}) + B \sum_{l=0}^{k} \gamma_l^2. \tag{7.29}$$

If $\gamma_k = \alpha/(k+1)^\beta$ for $\alpha > 0$ and $\beta \in (0, 1]$, then by re-arranging the terms

$$\frac{1}{k+1} \sum_{l=0}^{k} W_l \leq \frac{V_0 - V_{k+1}}{\alpha (k+1)^{1-\beta}} + \frac{B\alpha}{(k+1)^{1-\beta}} \sum_{l=0}^{k} \frac{1}{(l+1)^{2\beta}}.$$

Finally, if $\beta = 1/2$, then $1/(l+1)$ is decreasing in l and

$$\sum_{l=0}^{k} \frac{1}{(l+1)^{2\beta}} \leq 1 + \int_{l=0}^{k} \frac{1}{l+1} dl = 1 + \ln(k+1),$$

and therefore we reach (7.10).

If $\beta = 1/2 + \xi$ for $\xi \in (0, 1/2)$, then

$$\sum_{l=0}^{k} \frac{1}{(l+1)^{2\beta}} \leq 1 + \int_{l=0}^{k} \frac{1}{(l+1)^{1+2\xi}} dl \leq 1 + \frac{1}{2\xi}.$$

and thus we obtain (7.11).

7.A.2 Useful Lemmas for Applications

In this section, we provide useful lemmas for our analysis. We first prove one lemma, which states the property of the Moreau envelope $P_{1/\bar{\rho}}(x)$.

Lemma 7.4. *Consider the problem of minimizing $P(x) = F(x) + h(x)$ where $F(\cdot)$ is L-smooth and $h(\cdot)$ is convex. Then, for $\bar{\rho} > 0$*

$$\frac{1}{2\bar{\rho}} \|\nabla P_{1/\bar{\rho}}(x)\|^2 \leq P_{1/\bar{\rho}}(x) - P_{1/\bar{\rho}}(x^\star). \tag{7.30}$$

where $x^\star = argmin_{x \in \mathbb{R}^d} P(x)$.

Proof. Let $x^\star = \operatorname{argmin}_{x \in \mathbb{R}^d} P(x)$ and $\hat{x} = \mathbf{prox}_{P/\bar{\rho}}(x)$. Then,

$$P_{1/\bar{\rho}}(x) - P(\hat{x}) = \frac{\rho}{2} \|x - \hat{x}\|^2 = \frac{1}{2\bar{\rho}} \|\nabla P_{1/\bar{\rho}}(x)\|^2.$$

Next, by the fact that $P(\hat{x}) \geq P(x^\star)$,

$$P_{1/\bar{\rho}}(x) - P(x^\star) \geq \frac{1}{2\bar{\rho}} \|\nabla P_{1/\bar{\rho}}(x)\|^2.$$

Since $x^\star = \mathbf{prox}_{P/\bar{\rho}}(x^\star)$, i.e.

$$x^\star = \operatorname*{argmin}_{x \in \mathbb{R}^d} \left[P(x) + \frac{\bar{\rho}}{2} \|x - x^\star\|^2 \right],$$

we have $P(x^\star) = P_{1/\bar{\rho}}(x^\star)$. Finally, using this fact we complete the proof. □

Next, we derive the upper-bounds of $\mathbf{E}\|e_k\|^2$ for proximal stochastic compression and stochastic coding methods under the gradient-bounded and variance-bounded assumptions.

Lemma 7.5. *Consider the iterates $\{x_k\}_{k\in\mathbb{N}}$ generated by the* `Prox-SComp` *in Eq.* (7.14) *or equivalently* (7.38).

1. *If $\mathbf{E}\|g_k^i\|^2 \le C^2$, then $\mathbf{E}\|e_k\|^2 \le \tilde{C}$ where $\tilde{C} = qC^2$.*
2. *If $\mathbf{E}\|g_k^i - \nabla F^i(x_k)\|^2 \le \sigma^2$, then $\mathbf{E}\|e_k\|^2 \le C_1T + C_2$ where $C_1 = 3q$, $C_2 = 3q[\sigma^2 + \sum_{i=1}^n \mathbf{E}\|\nabla F^i(x^\star)\|^2]/n$, and $T = \sum_{i=1}^n \mathbf{E}\|\nabla F^i(x_k) - \nabla F^i(x^\star)\|^2/n$.*

Proof. Recalling the definition of e_k in (7.38), and using the fact that $\mathbf{E}\|\xi - \mathbf{E}\xi\|^2 = \mathbf{E}\|\xi\|^2 - \|\mathbf{E}\xi\|^2$ for any random vector $\xi \in \mathbb{R}^d$ and that $\|x\|^2 \ge 0$, we have

$$\mathbf{E}\|e_k\|^2 \le \mathbf{E}\left\|\frac{1}{n}\sum_{i=1}^n Q(g_k^i)\right\|^2.$$

Since $\mathbf{E}\|Q(v)\|^2 \le q\|v\|^2$ for $v \in \mathbb{R}^d$, we next have

$$\mathbf{E}\|e_k\|^2 \le \frac{q}{n}\sum_{i=1}^n \mathbf{E}\left\|g_k^i\right\|^2. \tag{7.31}$$

If $\mathbf{E}\|g_k^i\|^2 \le C^2$, then by (7.31) we prove the first statement.

If $\mathbf{E}\|g_k^i - \nabla F^i(x_k)\|^2 \le \sigma^2$, then by utilizing the fact that $\|x+y+z\|^2 \le 3\|x\|^2 + 3\|y\|^2 + 3\|z\|^2$ with $x = g_k^i - \nabla F^i(x_k)$, $y = \nabla F^i(x_k) - \nabla F^i(x^\star)$ and $z = \nabla F^i(x^\star)$ we prove the second statement. □

Lemma 7.6. *Consider the iterates $\{x_k\}_{k\in\mathbb{N}}$ generated by* `Prox-SCod` *in Eq.* (7.18) *or equivalently* (7.39). *Let $\beta = m\max_i \tau_i$ and $\tau_i = \sum_{j=1}^m D_{ij}^2/(\sum_{j=1}^m D_{ij})^2$ for the data assignment matrix $D \in \mathbb{R}^{m\times n}$.*

1. *If $\mathbf{E}\|g_k^i\|^2 \le C^2$, then $\mathbf{E}\|e_k\|^2 \le \tilde{C}$ where $\tilde{C} = \beta C^2/(1-p)$.*
2. *If $\mathbf{E}\|g_k^i - \nabla F^i(x_k)\|^2 \le \sigma^2$, then $\mathbf{E}\|e_k\|^2 \le C_1T + C_2$ where $C_1 = 3\beta/(1-p)$, $C_2 = 3\beta[\sigma^2 + \sum_{i=1}^n \mathbf{E}\|\nabla F^i(x^\star)\|^2/n]/(1-p)$, and $T = \sum_{i=1}^n \mathbf{E}\|\nabla F^i(x_k) - \nabla F^i(x^\star)\|^2/n$.*

Proof. Recalling the definition of e_k in (7.39), and using the fact that $\mathbf{E}\|\xi - \mathbf{E}\xi\|^2 = \mathbf{E}\|\xi\|^2 - \|\mathbf{E}\xi\|^2$ for any random vector $\xi \in \mathbb{R}^d$ and that $\|x\|^2 \ge 0$,

$$\mathbf{E}\|e_k\|^2 \le \frac{1}{n(1-p)^2}\sum_{i=1}^n \mathbf{E}\left\{\frac{\alpha_i^2}{\bar{\alpha}_i^2}\|g_k^i\|^2\right\},$$

where $\alpha_i = \sum_{j=1}^m D_{ij}w_j$ and $\bar{\alpha}_i = \sum_{j=1}^m D_{ij}$. Since w_j is 1 with probability $1-p$, i.e.

$$\mathbf{E}\alpha_i^2 \le m\sum_{j=1}^m \mathbf{E}[D_{ij}^2 w_j^2] = m(1-p)\sum_{j=1}^m D_{ij}^2,$$

we have

$$\mathbf{E}\|e_k\|^2 \leq \frac{m}{n(1-p)} \max_i \left\{ \frac{\beta_i}{\bar{\alpha}_i^2} \right\} \sum_{i=1}^{n} \mathbf{E}\|g_k^i\|^2, \tag{7.32}$$

where $\beta_i = \sum_{j=1}^{m} D_{ij}^2$.

If $\mathbf{E}\|g_k^i\|^2 \leq C^2$, then by (7.32) we prove the first statement.

If $\mathbf{E}\|g_k^i - \nabla F^i(x_k)\|^2 \leq \sigma^2$, then by utilizing the fact that $\|x+y+z\|^2 \leq 3\|x\|^2 + 3\|y\|^2 + 3\|z\|^2$ with $x = g_k^i - \nabla F^i(x_k)$, $y = \nabla F^i(x_k) - \nabla F^i(x^\star)$ and $z = \nabla F^i(x^\star)$ we prove the second statement. □

From Lemma 7.5 and 7.6, the problems under the gradient-bounded assumption implies $\mathbf{E}\|e_k\|^2 \leq \tilde{C}$ with a positive constant $\tilde{C}$, while the problems under the variance-bounded condition implies $\mathbf{E}\|e_k\|^2 \leq C_1 \sum_{i=1}^{n} \mathbf{E}\|\nabla F^i(x_k) - \nabla F^i(x^\star)\|^2/n + C_2$ with positive constants C_1, C_2.

By using the assumption that $\mathbf{E}\|e_k\|^2 \leq \tilde{C}$ for a positive constant $\tilde{C}$, the next lemma establishes the main inequality for deriving the results for proximal noisy gradient algorithms on non-convex problems.

Lemma 7.7. *Consider the problem of minimizing $P(x) = F(x) + h(x)$ where $F(\cdot)$ is L-smooth and $h(\cdot)$ is convex. Then, the iterates $\{x_k\}_{k\in\mathbb{N}}$ by the proximal noisy gradient algorithm: Given $x_0 \in \mathbb{R}^d$*

$$x_{k+1} = \mathbf{prox}_{\gamma_k h}\left(x_k - \gamma_k[\nabla F(x_k) + e_k]\right), \tag{7.33}$$

where $\mathbf{E}[e_k] = 0$ and $\mathbf{E}\|e_k\|^2 \leq \tilde{C}$ for $\tilde{C} \geq 0$. If $\gamma_k \leq 1/(\bar{\rho} - L)$ and $\bar{\rho} > L$, then

$$\mathbf{E}[P_{1/\bar{\rho}}(x_{k+1})] \leq \mathbf{E}[P_{1/\bar{\rho}}(x_k)] + \frac{\bar{\rho}}{2}\gamma_k^2\tilde{C} - \frac{\gamma_k(\bar{\rho}-L)}{2\bar{\rho}}\mathbf{E}\|\nabla P_{1/\bar{\rho}}(x_k)\|^2. \tag{7.34}$$

Proof. Since $P_{1/\bar{\rho}}(x) = \min_y\{P(y) + (\bar{\rho}/2)\|y-x\|^2\}$,

$$P_{1/\bar{\rho}}(x_{k+1}) \leq P(\hat{x}_k) + \frac{\bar{\rho}}{2}\|\hat{x}_k - x_{k+1}\|^2,$$

where $\hat{x}_k = \mathbf{prox}_{P/\bar{\rho}}(x_k)$. By taking the expectation,

$$\mathbf{E}[P_{1/\bar{\rho}}(x_{k+1})] \leq \mathbf{E}[P(\hat{x}_k)] + \frac{\bar{\rho}}{2}\mathbf{E}\|\hat{x}_k - x_{k+1}\|^2. \tag{7.35}$$

To complete the proof, we need to find the upper-bound for $\mathbf{E}\|\hat{x}_k - x_{k+1}\|^2$. From the definition of the Euclidean norm, from Lemma [186, Lemma 3.2] (i.e. $\hat{x}_k = \mathbf{prox}_{\gamma_k h}(\gamma_k\bar{\rho}x_k - \gamma_k\nabla F(\hat{x}_k) + (1-\gamma_k\bar{\rho})\hat{x}_k)$), and by the non-expansive property of the proximal operator,

$$\|x_{k+1} - \hat{x}_k\|^2 \leq \|\delta_k(x_k - \hat{x}_k) - \gamma_k[\nabla F(x_k) - \nabla F(\hat{x}_k)] - \gamma_k e_k\|^2,$$

where $\delta_k = 1-\gamma_k\bar{\rho}$. Next, taking the expectation, using the fact that $\mathbf{E}[e_k] = 0$ and expanding the terms yield

$$\begin{aligned}\mathbf{E}\|x_{k+1} - \hat{x}_k\|^2 \leq \delta_k^2\mathbf{E}\|x_k - \hat{x}_k\|^2 + \gamma_k^2\mathbf{E}\|\nabla F(x_k) - \nabla F(\hat{x}_k)\|^2 \\ - 2\gamma_k\delta_k\mathbf{E}\langle x_k - \hat{x}_k, \nabla F(x_k) - \nabla F(\hat{x}_k)\rangle + \gamma_k^2\mathbf{E}\|e_k\|^2.\end{aligned}$$

Since $F(x)$ is L-smooth, we can conclude that

$$\mathbf{E}\|x_{k+1} - \hat{x}_k\|^2 \leq \beta_k\mathbf{E}\|x_k - \hat{x}_k\|^2 + \gamma_k^2\mathbf{E}\|e_k\|^2,$$

where $\beta_k = \delta_k^2 + 2\gamma_k\delta_k L + \gamma_k^2L^2$. Next, if $\gamma_k \leq 1/(\bar{\rho} - L)$ with $\bar{\rho} > L$, then $\beta_k \leq 1 - \gamma_k(\bar{\rho} - L)$. Since $\mathbf{E}\|e_k\|^2 \leq \tilde{C}$ for a positive constant $\tilde{C}$,

$$\mathbf{E}\|x_{k+1} - \hat{x}_k\|^2 \leq \mathbf{E}\|x_k - \hat{x}_k\|^2 + \gamma_k^2\tilde{C} - \gamma_k(\bar{\rho} - L)\mathbf{E}\|x_k - \hat{x}_k\|^2.$$

Plugging this inequality into Eq. (7.35) yields

$$\begin{aligned}\mathbf{E}[P_{1/\bar{\rho}}(x_{k+1})] \leq \mathbf{E}\left[P(\hat{x}_k) + \frac{\bar{\rho}}{2}\|\hat{x}_k - x_k\|^2\right] + \frac{\bar{\rho}}{2}\gamma_k^2\tilde{C} \\ - \frac{\bar{\rho}\gamma_k(\bar{\rho} - L)}{2}\mathbf{E}\|x_k - \hat{x}_k\|^2.\end{aligned}$$

Since $\hat{x}_k = \mathbf{prox}_{P/\bar{\rho}}(x_k)$, we can prove that

$$P_{1/\bar{\rho}}(x_k) = P(\hat{x}_k) + (\bar{\rho}/2)\|\hat{x}_k - x_k\|^2,$$

and that

$$\mathbf{E}[P_{1/\bar{\rho}}(x_{k+1})] \leq \mathbf{E}\left[P_{1/\bar{\rho}}(x_k)\right] + \frac{\bar{\rho}}{2}\gamma_k^2\tilde{C} - \frac{\bar{\rho}\gamma_k(\bar{\rho} - L)}{2}\mathbf{E}\|x_k - \hat{x}_k\|^2.$$

Finally, since $\nabla P_{1/\bar{\rho}}(x) = \bar{\rho}(x - \mathbf{prox}_{P/\bar{\rho}}(x))$, we obtain (7.34).

□

Finally, by utilizing the assumption that $\mathbf{E}\|e_k\|^2 \leq C_1\sum_{i=1}^n \mathbf{E}\|\nabla F^i(x_k) - \nabla F^i(x^\star)\|^2/n + C_2$ for positive constants C_1, C_2, the next lemma proves the convergence of proximal noisy gradient algorithms for strongly convex problems.

Lemma 7.8. *Consider the problem of minimizing $P(x) = F(x) + h(x)$ where $F(\cdot)$ is μ-strongly convex and L-smooth, and $h(\cdot)$ is convex, and the proximal noisy gradient algorithm: Given $x_0 \in \mathbb{R}^d$*

$$x_{k+1} = \mathbf{prox}_{\gamma_k h}\left(x_k - \gamma_k[\nabla F(x_k) + e_k]\right), \tag{7.36}$$

where $\mathbf{E}[e_k] = 0$ and $\mathbf{E}\|e_k\|^2 \leq C_1\sum_{i=1}^n \mathbf{E}\|\nabla F^i(x_k) - \nabla F^i(x^\star)\|^2/n + C_2$ for positive constants C_1, C_2. If $\gamma_k \leq (C_1+1)^{-1}/L$, then

$$\mathbf{E}\|x_{k+1} - x^\star\|^2 \leq (1 - A\gamma_k)\,\mathbf{E}\|x_k - x^\star\|^2 + \gamma_k^2 B, \tag{7.37}$$

where $A = \mu$ and $B = C_2$.

Proof. Since $x^\star = \operatorname{argmin}_x P(x)$, we can show that $x^\star = \mathbf{prox}_{\gamma_k h}(x^\star - \gamma_k \nabla F(x^\star))$. By the definition of the Euclidean norm and the non-expansive property of the proximal operator,

$$\|x_{k+1} - x^\star\|^2 \leq \|x_k - x^\star - \gamma_k[\nabla F(x_k) + e_k - \nabla F(x^\star)]\|^2.$$

Next, taking the expectation and using $\mathbf{E}[e_k] = 0$

$$\begin{aligned}\mathbf{E}\|x_{k+1} - x^\star\|^2 \leq \mathbf{E}\|x_k - x^\star\|^2 + \gamma_k^2 \mathbf{E}\|e_k\|^2 - 2\gamma_k \mathbf{E}\langle \nabla F(x_k) - \nabla F(x^\star), x_k - x^\star\rangle \\ + \gamma_k^2 \mathbf{E}\|\nabla F(x_k) - \nabla F(x^\star)\|^2.\end{aligned}$$

If $\mathbf{E}\|e_k\|^2 \leq C_1 \sum_{i=1}^n \mathbf{E}\|\nabla F^i(x_k) - \nabla F^i(x^\star)\|^2/n + C_2$ for positive constants C_1, C_2, then

$$\begin{aligned}\mathbf{E}\|x_{k+1} - x^\star\|^2 \leq \mathbf{E}\|x_k - x^\star\|^2 + \gamma_k^2 C_2 - 2\gamma_k \mathbf{E}\langle \nabla F(x_k) - \nabla F(x^\star), x_k - x^\star\rangle \\ + \gamma_k^2 \mathbf{E}\|\nabla F(x_k) - \nabla F(x^\star)\|^2 \\ + \frac{\gamma_k^2 C_1}{n} \sum_{i=1}^n \mathbf{E}\|\nabla F^i(x_k) - \nabla F^i(x^\star)\|^2.\end{aligned}$$

Next, by the co-coercivity of $F^i(\cdot)$, i.e. for all $x, y \in \mathbb{R}^d$

$$\|\nabla F^i(x) - \nabla F^i(y)\|^2 \leq L\langle \nabla F^i(x) - \nabla F^i(y), x - y\rangle,$$

and by the fact that $F(x) = \sum_{i=1}^n F^i(x)/n$, we have

$$\mathbf{E}\|x_{k+1} - x^\star\|^2 \leq \mathbf{E}\|x_k - x^\star\|^2 + \gamma_k^2 C_2 - 2\gamma_k \beta_k \mathbf{E}\langle \nabla F(x_k) - \nabla F(x^\star), x_k - x^\star\rangle,$$

where $\beta_k = 1 - \gamma_k(C_1 + 1)L/2$. Next, by using the strong convexity of $F(\cdot)$, i.e. for all $x, y \in \mathbb{R}^d$

$$\langle \nabla F(x) - \nabla F(y), x - y\rangle \geq \mu\|x - y\|^2,$$

we get

$$\mathbf{E}\|x_{k+1} - x^\star\|^2 \leq \rho_k \mathbf{E}\|x_k - x^\star\|^2 + \gamma_k^2 C_2,$$

where $\rho_k = 1 - 2\mu\gamma_k\beta_k$. Finally, if $\gamma_k \leq (C_1 + 1)^{-1}/L$, then $\rho_k \leq 1 - \mu\gamma_k$. We thus complete the proof.

□

7.A.3 Proof of Theorem 7.1

We can write `Prox-SComp` in Eq. (7.14) equivalently as

$$x_{k+1} = \mathbf{prox}_{\gamma_k h}(x_k - \gamma_k[\nabla F(x_k) + e_k]), \tag{7.38}$$

where

$$e_k = \frac{1}{n} \sum_{i=1}^n Q(g_k^i) - \nabla F(x_k).$$

Here, e_k satisfies $\mathbf{E}[e_k] = 0$ and now we prove the results.

Proof of Theorem 7.1-1)

Since $\mathbf{E}\|g_k^i\|^2 \leq C^2$ and $\mathbf{E}[e_k] = 0$, Lemma 7.5-1) holds. Lemma 7.4 and Lemma 7.7 with $\tilde{C} = qC^2$ imply that `Prox-SComp` for Problem (7.12) under Assumption 7.1 satisfies (7.7) with $V_k = \mathbf{E}[P_{1/\bar{\rho}}(x_k) - P_{1/\bar{\rho}}(x^\star)]$, $W_k = (\bar{\rho} - L)\mathbf{E}\|\nabla P_{1/\bar{\rho}}(x_k)\|^2/(2\bar{\rho})$ and $B = \bar{\rho}qC^2/2$, for $\bar{\rho} > L$. Also note that `Prox-SComp` satisfies $W_k \leq cV_k$ with $c = 1/(\bar{\rho} - L)$. Finally, if $0 \leq V_k \leq R^2$, then from Lemma 7.2 with $\alpha = \bar{\rho} - L$ we complete the proof.

Proof of Theorem 7.1-2)

Since $\mathbf{E}\|g_k^i - \nabla F(x_k)\|^2 \leq \sigma^2$ and $\mathbf{E}[e_k] = 0$, Lemma 7.5-2) holds. Lemma 7.8 with $C_1 = 3q$ and $C_2 = 3q[(1/n)\sum_{i=1}^n \mathbf{E}\|\nabla F^i(x^\star)\|^2 + \sigma^2]$ implies that `Prox-SComp` for Problem (7.12) under Assumptions 7.1 and 7.2 satisfies (7.3) with $V_k = \mathbf{E}\|x_k - x^\star\|^2$, $A = \mu$, and $B = C_2$. Finally, if $\alpha = 2/A$, from Lemma 7.1-2) we complete the proof.

7.A.4 Proof of Theorem 7.2

We can express `Prox-SCod` in Eq. (7.18) equivalently as

$$x_{k+1} = \mathbf{prox}_{\gamma_k h}(x_k - \gamma_k[\nabla F(x_k) + e_k]), \tag{7.39}$$

where

$$e_k = \frac{1}{n(1-p)} \sum_{i=1}^n \frac{\alpha_i}{\bar{\alpha}_i} g_k^i - \nabla F(x_k).$$

Here, e_k satisfies $\mathbf{E}[e_k] = 0$ and now we prove the results.

Proof of Theorem 7.2-1)

Since $\mathbf{E}\|g_k^i\|^2 \leq C^2$ and $\mathbf{E}[e_k] = 0$, Lemma 7.6-1) holds. Lemma 7.4 and Lemma 7.7 with $\tilde{C} = \beta C^2/(1-p)$ imply that `Prox-SCod` for Problem (7.12) under Assumption 7.1 satisfies (7.7) with $V_k = \mathbf{E}[P_{1/\bar{\rho}}(x_k) - P_{1/\bar{\rho}}(x^\star)]$, $W_k = (\bar{\rho} - L)\mathbf{E}\|\nabla P_{1/\bar{\rho}}(x_k)\|^2/(2\bar{\rho})$ and $B = \bar{\rho}\beta C^2/[2(1-p)]$, for $\bar{\rho} > L$. Also note that `Prox-SCod` satisfies $W_k \leq cV_k$ with $c = 1/(\bar{\rho} - L)$. Finally, if $0 \leq V_k \leq R^2$, then from Lemma 7.2 with $\alpha = \bar{\rho} - L$ we complete the proof.

Proof of Theorem 7.2-2)

Since $\mathbf{E}\|g_k^i - \nabla F(x_k)\|^2 \leq \sigma^2$ and $\mathbf{E}[e_k] = 0$, Lemma 7.5-2) holds. Lemma 7.8 with $C_1 = 3\beta/(1-p)$ and $C_2 = 3\beta[(1/n)\sum_{i=1}^n \mathbf{E}\|\nabla F^i(x^\star)\|^2 + \sigma^2]/(1-p)$ implies that `Prox-SCod` for Problem (7.12) under Assumptions 7.1 and 7.2 satisfies (7.3) with $V_k = \mathbf{E}\|x_k - x^\star\|^2$, $A = \mu$, and $B = C_2$. Finally, if $\alpha = 2/A$, from Lemma 7.1-2) we complete the proof.

7.A.5 Proof of Theorem 7.3

We derive the main results for the proximal zeroth-order algorithms (7.19) for Problem (7.12).

7.A.5.1 Proof of Theorem 7.3-1)

Since $P_{1/\bar{\rho}}(x) = \min_y\{P(y) + (\bar{\rho}/2)\|y - x\|^2\}$,

$$P_{1/\bar{\rho}}(x_{k+1}) \leq P(\hat{x}_k) + \frac{\bar{\rho}}{2}\|\hat{x}_k - x_{k+1}\|^2,$$

where $\hat{x}_k = \mathbf{prox}_{P/\bar{\rho}}(x_k)$. By taking the expectation,

$$\mathbf{E}[P_{1/\bar{\rho}}(x_{k+1})] \leq \mathbf{E}[P(\hat{x}_k)] + \frac{\bar{\rho}}{2}\mathbf{E}\|\hat{x}_k - x_{k+1}\|^2. \tag{7.40}$$

To complete the proof, we need to find the upper-bound for $\mathbf{E}\|\hat{x}_k - x_{k+1}\|^2$. From the definition of the Euclidean norm, from Lemma [186, Lemma 3.2] (i.e. $\hat{x}_k = \mathbf{prox}_{\gamma_k h}(\gamma_k\bar{\rho}x_k - \gamma_k\nabla F(\hat{x}_k) + (1 - \gamma_k\bar{\rho})\hat{x}_k)$), and by the non-expansive property of the proximal operator,

$$\|x_{k+1} - \hat{x}_k\|^2 \leq \|\delta_k(x_k - \hat{x}_k) - \gamma_k[G_{\tau_k}(x_k) - \nabla F(\hat{x}_k)]\|^2,$$

where $\delta_k = 1 - \gamma_k\bar{\rho}$. Next, by taking the expectation, expanding the terms and using the fact that $\mathbf{E}[G_{\tau_k}(x_k)] = \nabla F_{\tau_k}(x_k)$ where $F_\tau(x) = \mathbb{E}_u[F(x + \tau u)]$,

$$\begin{aligned}\mathbf{E}\|x_{k+1} - \hat{x}_k\|^2 \leq \delta_k^2\mathbf{E}\|x_k - \hat{x}_k\|^2 + \gamma_k^2\mathbf{E}\|G_{\tau_k}(x_k) - \nabla F(\hat{x}_k)\|^2 \\ - 2\delta_k\gamma_k\mathbf{E}\langle x_k - \hat{x}_k, \nabla F(x_k) - \nabla F(\hat{x}_k)\rangle \\ + 2\delta_k\gamma_k\mathbf{E}\langle x_k - \hat{x}_k, \nabla F_{\tau_k}(x_k) - \nabla F(x_k)\rangle.\end{aligned}$$

Since the fact that $F(x)$ is L-smooth implies that $F(x)$ is L-weakly convex [186], i.e.

$$\langle \nabla F(x) - \nabla F(y), x - y\rangle \geq -L\|x - y\|^2,$$

and since we have by Cauchy-Schwarz's inequality

$$2\langle x_k - \hat{x}_k, \nabla F_{\tau_k}(x_k) - \nabla F(x_k)\rangle \leq \|x_k - \hat{x}_k\|^2 + \|\nabla F_{\tau_k}(x_k) - \nabla F(x_k)\|^2,$$

then

$$\begin{aligned}\mathbf{E}\|x_{k+1} - \hat{x}_k\|^2 \leq \beta_k\mathbf{E}\|x_k - \hat{x}_k\|^2 + \gamma_k^2\mathbf{E}\|G_{\tau_k}(x_k) - \nabla F(\hat{x}_k)\|^2 \\ + \delta_k\gamma_k\mathbf{E}\|\nabla F_{\tau_k}(x_k) - \nabla F(x_k)\|^2,\end{aligned}$$

where $\beta_k = \delta_k^2 + (2L + 1)\delta_k\gamma_k$. Next, from Eq. (27) of [178, Lemma 3], i.e. $\|\nabla F_\tau(x) - \nabla F(x)\| \leq \tau^2L^2(d + 3)^3/4$,

$$\begin{aligned}\mathbf{E}\|x_{k+1} - \hat{x}_k\|^2 \leq \beta_k\mathbf{E}\|x_k - \hat{x}_k\|^2 + \frac{\delta_k\gamma_k\tau_k^2L^2(d + 3)^3}{4} \\ + \gamma_k^2\mathbf{E}\|G_{\tau_k}(x_k) - \nabla F(\hat{x}_k)\|^2.\end{aligned}$$

To complete the proof, we need to derive the bound for $\mathbf{E}\|G_{\tau_k}(x_k)-\nabla F(\hat{x}_k)\|^2$. If $\|\nabla F(x)\|^2 \leq C^2$, then

$$\begin{aligned}\mathbf{E}\|G_{\tau_k}(x_k) - \nabla F(\hat{x}_k)\|^2 &\leq 2\mathbf{E}\|G_{\tau_k}(x_k)\|^2 + 2C^2 \\ &\leq \tau_k^2 L^2(d+6)^3 + 2(2d+9)C^2,\end{aligned}$$

where we reach the first inequality by the fact that $\|x+y\|^2 \leq 2\|x\|^2+2\|y\|^2$ for $x, y \in \mathbb{R}^d$, and the second inequality by Lemma 4 from [178]. Next, plugging this result into the main inequality yields

$$\begin{aligned}\mathbf{E}\|x_{k+1} - \hat{x}_k\|^2 \leq \beta_k \mathbf{E}\|x_k - \hat{x}_k\|^2 + \frac{\delta_k \gamma_k \tau_k^2 L^2 (d+3)^3}{4} \\ + \gamma_k^2[\tau_k^2 L^2(d+6)^3 + 2(2d+9)C^2].\end{aligned}$$

If $\bar{\rho} \in (L - 1/2, 2L + 1]$, then

$$\begin{aligned}\mathbf{E}\|x_{k+1} - \hat{x}_k\|^2 \leq \mathbf{E}\|x_k - \hat{x}_k\|^2 - 2\gamma_k(\bar{\rho} - L - 1/2)\mathbf{E}\|x_k - \hat{x}_k\|^2 \\ + \gamma_k^2[\tau_k^2 L^2(d+6)^3 + 2(2d+9)C^2] + \frac{\delta_k \gamma_k \tau_k^2 L^2 (d+3)^3}{4}.\end{aligned}$$

If $\gamma_k = (2\bar{\rho}\sqrt{d+6})^{-1}/\sqrt{k+1}$ and $\tau_k = \eta\gamma_k$ for $\eta > 0$, then $\gamma_k \leq 1/(2\bar{\rho}\sqrt{d+6})$ and

$$\mathbf{E}\|x_{k+1} - \hat{x}_k\|^2 \leq \mathbf{E}\|x_k - \hat{x}_k\|^2 + \gamma_k^2 \tilde{C} - 2\gamma_k(\bar{\rho} - L - 1/2)\mathbf{E}\|x_k - \hat{x}_k\|^2,$$

where $\tilde{C} = \eta^2 L^2 (d+3)^2/(8\bar{\rho}) + \eta^2 L^2 (d+6)^2/(4\bar{\rho}^2) + 2(2d+9)C^2$. Next, substituting this inequality into Eq. (7.40), we have

$$\begin{aligned}\mathbf{E}[P_{1/\bar{\rho}}(x_{k+1})] \leq \mathbf{E}\left[P(\hat{x}_k) + \frac{\bar{\rho}}{2}\|\hat{x}_k - x_k\|^2\right] \\ + \frac{\bar{\rho}}{2}\gamma_k^2 \tilde{C} - \bar{\rho}\gamma_k(\bar{\rho} - L - 1/2)\mathbf{E}\|x_k - \hat{x}_k\|^2.\end{aligned}$$

Since $\hat{x}_k = \mathbf{prox}_{P/\bar{\rho}}(x_k)$, i.e. $P_{1/\bar{\rho}}(x_k) = P(\hat{x}_k) + (\bar{\rho}/2)\|\hat{x}_k - x_k\|^2$,

$$\mathbf{E}[P_{1/\bar{\rho}}(x_{k+1})] \leq \mathbf{E}\left[P_{1/\bar{\rho}}(x_k)\right] + \frac{\bar{\rho}}{2}\gamma_k^2 \tilde{C} - \bar{\rho}\gamma_k(\bar{\rho} - L - 1/2)\mathbf{E}\|x_k - \hat{x}_k\|^2.$$

Next, by the fact that $\nabla P_{1/\bar{\rho}}(x) = \bar{\rho}(x - \mathbf{prox}_{P/\bar{\rho}}(x))$ and Lemma 7.4, the proximal zeroth-order algorithms (7.19) for Problem (7.12) under Assumption 7.1 satisfies (7.7) with $V_k = \mathbf{E}[P_{1/\bar{\rho}}(x_k) - P_{1/\bar{\rho}}(x^\star)]$, $W_k = (\bar{\rho} - L - 1/2)\mathbf{E}\|\nabla P_{1/\bar{\rho}}(x_k)\|^2/\bar{\rho}$ and $B = \bar{\rho}\tilde{C}/2$, for $\bar{\rho} \in (L - 1/2, 2L + 1]$. Also note that Algorithm (7.19) satisfy $W_k \leq cV_k$ with $c = 1/(2\bar{\rho} - 2L - 1)$. Finally, if $0 \leq V_k \leq R^2$, then from Lemma 7.2 with $\alpha \geq 1/(2\bar{\rho})$ we complete the proof.

7.A.5.2 Proof of Theorem 7.3-2)

From the definition of the Euclidean norm and from Eq. (7.19), and also by the fact that $x^\star = \mathbf{prox}_{\gamma_k h}(x^\star - \gamma_k \nabla F(x^\star))$ and by the non-expansive property of $\mathbf{prox}_{\gamma_k h}(y)$,

$$r_{k+1}^2 \leq r_k^2 + \gamma_k^2 \|G_{\tau_k}(x_k) - \nabla F(x^\star)\|^2 - 2\gamma_k \langle G_{\tau_k}(x_k) - \nabla F(x^\star), x_k - x^\star \rangle,$$

where $r_k = \|x_k - x^\star\|$. Next, by taking the expectation and using the fact that $\mathbf{E}[G_{\tau_k}(x_k)] = \nabla F_{\tau_k}(x_k)$, where $F_\tau(x) = \mathbb{E}_u[F(x + \tau u)]$

$$\begin{aligned}\mathbf{E}[r_{k+1}^2] \leq \mathbf{E}[r_k^2] &+ \gamma_k^2 \mathbf{E}\|G_{\tau_k}(x_k) - \nabla F(x^\star)\|^2 \\ &- 2\gamma_k \mathbf{E}\langle \nabla F_{\tau_k}(x_k) - \nabla F(x^\star), x_k - x^\star \rangle.\end{aligned}$$

By using the fact that $\|x + y\|^2 \leq 2\|x\|^2 + 2\|y\|^2$ for $x, y \in \mathbb{R}^d$ and by re-arranging the terms,

$$\begin{aligned}\mathbf{E}[r_{k+1}^2] \leq \mathbf{E}[r_k^2] &+ 2\gamma_k^2 \mathbf{E}\|G_{\tau_k}(x_k)\|^2 + 2\gamma_k^2 \|\nabla F(x^\star)\|^2 \\ &- 2\gamma_k \mathbf{E}\langle \nabla F(x_k) - \nabla F(x^\star), x_k - x^\star \rangle \\ &- 2\gamma_k \mathbf{E}\langle \nabla F_{\tau_k}(x_k) - \nabla F(x_k), x_k - x^\star \rangle.\end{aligned}$$

Next, since $F(x)$ is μ-strongly convex, i.e.

$$\langle \nabla F(x_k) - \nabla F(x^\star), x_k - x^\star \rangle \geq \mu \|x_k - x^\star\|^2,$$

and

$$\begin{aligned}\gamma_k \mathbf{E}\langle \nabla F(x_k) - \nabla F_{\tau_k}(x_k), x_k - x^\star \rangle \leq \frac{\mu\gamma_k}{2} &\mathbf{E}\|x_k - x^\star\|^2 \\ &+ \frac{\gamma_k}{2\mu} \mathbf{E}\|\nabla F(x_k) - \nabla F_{\tau_k}(x_k)\|^2,\end{aligned}$$

we get

$$\begin{aligned}\mathbf{E}[r_{k+1}^2] \leq (1 - \mu\gamma_k)\mathbf{E}[r_k^2] &+ \frac{\gamma_k}{\mu} \mathbf{E}\|\nabla F(x_k) - \nabla F_{\tau_k}(x_k)\|^2 \\ &+ 2\gamma_k^2 \mathbf{E}\|G_{\tau_k}(x_k)\|^2 + 2\gamma_k^2 \|\nabla F(x^\star)\|^2.\end{aligned}$$

Next, from Eq. (27) and (29) in [178], we obtain

$$\begin{aligned}\mathbf{E}[r_{k+1}^2] \leq (1 - \mu\gamma_k)\mathbf{E}[r_k^2] &+ \gamma_k \beta_k \\ &+ 4\gamma_k^2 (d+4) \mathbf{E}\|\nabla F(x_k)\|^2 + 2\gamma_k^2 \|\nabla F(x^\star)\|^2,\end{aligned}$$

where $\beta_k = \tau_k^2 L^2 (d+3)^3/(4\mu) + \gamma_k \tau_k^2 L^2 (d+6)^3$.

If $\|\nabla F(x)\|^2 \leq C^2$, then

$$\mathbf{E}[r_{k+1}^2] \leq (1 - \mu\gamma_k)\mathbf{E}[r_k^2] + \gamma_k \beta_k + 2\gamma_k^2 (2d+9) C^2.$$

If $\gamma_k = 1/[L(d+6)]$ and $\tau_k = \eta\gamma_k$ for $\eta > 0$, then $\beta_k \leq \gamma_k[\eta^2 L(d+3)^2/(4\mu) + \eta^2(d+6)]$ and

$$\mathbf{E}[r_{k+1}^2] \leq (1-\mu\gamma_k)\mathbf{E}[r_k^2] + \gamma_k^2\tilde{\beta},$$

where $\tilde{\beta} = 0.25\eta^2 L(d+3)^2\mu^{-1} + \eta^2(d+6) + 2(2d+9)C^2$. To conclude, the proximal zeroth-order algorithms (7.19) for Problem (7.12) under Assumptions 7.1 and 7.2 satisfies (7.3) with $V_k = \mathbf{E}\|x_k - x^\star\|^2$, $A = \mu$, and $B = \tilde{\beta}$. Finally, if $\alpha = 2/A$, then from Lemma 7.1-2) we complete the proof.

Chapter 8
Conclusion and Future Outlook

In this thesis, we have explored first-order algorithms for communication efficient learning. In particular, we have developed theoretical frameworks for analyzing and improving convergence behaviors of algorithms using compressed information on two types of communication architectures: full gradient communication (or the single-node setting) and partial gradient communication (or the multiple-node setting). In the first part, we have provided unified convergence analysis frameworks for first-order algorithms using direct compression and error compensation. In the second part, we have designed flexible parameter-tuning frameworks that enable the algorithms to attain strong convergence performance and high communication efficiency.

Next, we briefly summarize the contributions and provide a future outlook.

Summary

In Chapter 3, we investigated how various compression schemes impact iteration and communication complexity of the first-order algorithms. In particular, we presented the unified convergence analysis for several families of compression algorithms under full and/or partial gradient communication. The tuning parameters and convergence rates have explicit formulas for how compression accuracy and staleness bounds due to asynchrony affect the expected time to reach an ϵ-accurate solution. These results allow us to characterize the trade-off between iteration and communication complexity of compressed first-order algorithms.

To improve the performance of compression algortithms, we explored error compensation strategies in Chapter 4. In essence, we provided a theoretical justification on why error compensation can avoid all accumulated compression errors in compressed gradient methods, and provide significant solution accuracy gains on ill-conditioned quadratic problems. In addition, we established strong convergence guarantees of decentralized stochastic gradient methods using Hessian-free and Hessian-aided error compensation. We have presented the first theoretical analysis of sparsified methods with the Hessian-free compensation scheme for non-convex optimization. We have also shown the effect of Hessian-aided compensation on stochastic gradient methods on both convex and nonconvex problems. Compared to direct compression methods, the error compensation methods exhibit superior convergence performance in terms of speed and solution accuracy, as illustrated numerically on

classification problems using large benchmark data-sets in machine learning. Next, we extended a theoretical support for error compensation to a class of federated learning algorithms called `Eco-FedSplit`, and demonstrated significantly improved communication efficiency and solution accuracy in Chapter 5. Unlike direct compression, error-compensated compression helps `FedSplit` to converge towards the approximate solution with arbitrarily high accuracy as we reduce the value of the federated averaging parameter λ. The superior performance of `Eco-FedSplit` compared to other compressed algorithms in federated learning was illustrated empirically on logistic regression problems on several benchmark data-sets.

In Chapter 6, we proposed a communication-aware adaptive tuning that optimizes the communication-efficiency of gradient sparsification. The adaptive tuning relies on a data-dependent measure of an objective function improvement, and adapts the compression level to maximize the descent per communicated bit. Unlike existing heuristics, our tuning rules are guaranteed to reduce communication in realistic communication models. In particular, our rules are more communication-efficient when communication overhead or packet transmissions are accounted for. We experimentally showed how CAT can be used to optimize both transmission time and communicated bits. Moreover, we illustrated the promise of CAT in multi-node settings with all considered compressions. From the theoretical point of view , we demonstrated that worst-case analysis of gradient compression does not guarantee *any* advantages or provide insight into why compression improves communication efficiency. In contrast, our problem/data dependent convergence results that demonstrate these benefits. Our theoretical results cover multi-node settings if the compression is stochastic and unbiased, and single-node settings if the compression is deterministic. To the best of our knowledge, all analytical results on multi-node compression use either such assumptions or assumptions on the similarity of the nodes' loss functions.

In addition to the adaptive tuning strategies for different compressionl levels, we investigated step-size policies for contractive and non-expansive iterations under noise in Chapter 7. We derived closed-loop policies that maximize the expected progress per iteration, and proposed diminishing step-size schedules with matching convergence rates. Our theoretical results suggest that decaying step-sizes enable these iterations to converge at an optimal $\mathcal{O}(1/k)$ and $\mathcal{O}(1/\sqrt{k})$ rate, respectively. We then applied the results to a number of noisy proximal gradient algorithms that were so far lacking a complete theoretical understanding. Numerical experiments validated our results.

Future Outlook

Now, we outline the following possible future research directions.

Adaptive step-size schedules for high-performance systems

A step-size is a critical tuning parameter that controls convergence speed of general distributed and federated optimization algorithms. In Chapter 7, we illustrated the benefits of using diminishing step-sizes over fixed step-sizes for general noisy gradient descent. However, these diminishing step-size schedules may lead to too slow convergence in practice since they decrease the step-size too early. To improve the performance, a number of adaptive step decay policies [187, 188, 189, 190] have been recently proposed. The key idea of these approaches is to start with a large step-size and then reduce it after the algorithm reaches stationarity detected by statistical tests (e.g. by optimization analysis tools [187, 188] or by the characterizations of stationary stochastic processes [189, 190]). However, these policies improve the convergence only for stochastic gradient algorithms, and theory confirming the results for general noisy gradient algorithms (e.g. compression algorithms, delayed gradient algorithms, zeroth-order optimization algorithms, etc.) are currently lacking. It would be interesting to explore and develop analysis tools that can be used as a guideline to fine-tune step-size schedules to maximize the convergence rate, thus saving communication costs.

Problem-dependent complexity of compression algorithms

In Chapter 6 we provided data-dependent complexity and automatic tuning strategies for centralized compression algorithms. However, theoretical results for general distributed and federated compression algorithms are lacking, and current tuning strategies need the gradient to compute the optimal compression level at every iteration. Therefore, one possible extension is to explore the statistical distributions of gradients in machine learning models (see e.g., in [191]). This could potentially allow us to better characterize closed-form data-dependent complexity and find the almost optimal compression level before training. Another interesting direction is to explore an average-case analysis framework that is used to analyze the exact convergence behaviors for stochastic gradient methods [192, 193, 194]. This framework may enable us to characterize tight convergence bounds for stochastic compression methods, which can lead to tunable parameters (the step-size and compression level) that guarantee near-optimal convergence in practice. These directions are still under its infancy and it would be interesting to investigate whether state-of-the-art tools can be developed in building fast compression algorithms with optimally tuned hyparameters.

Resource-aware distributed communication strategies

Throughout the thesis, we have explored how to tune hyperparameters (i.e. step-sizes and compression levels) for communication-efficient optimization al-

gorithms. These hyperparameters are however tuned separately according to different aims (e.g. to maximize the convergence rate [160] or communication efficiency, e.g., in Chapter 5). Algorithms using all of these parameter tunings that are optimized individually may attain sub-optimal convergence performance and communication efficiency. This necessitates the development of a systematic framework to tune interrelated hyperparameters for general distributed and federated optimization algorithms [195, 196]. However, this research direction is currently underexplored. It would be interesting to develop a unified framework for fine-tuning key hyperparameters simultaneously to attain high convergence speed, solution accuracy, and communication efficiency. In this direction, we need to develop performance and efficiency measures, and then to investigate unified formulation to optimize the hyperparameters jointly to satisfy these measures.

Compression methods for improved differentially-private training

State-of-the-art distributed and federated systems are susceptible to privacy leakage by recent inference attacks [197, 198]. To address this issue, many privacy-preserving strategies have been proposed. In addition to introducing additive noise, one recent strategy is compressing transmitted information to guarantee differential privacy on optimization algorithms in privacy analysis [199, 200]. All stated privacy-preserving approaches however lead to a trade-off between high privacy and fast convergence speed. This necessitates the development of privacy-aware optimization algorithms that can strike the optimal balance to attain high privacy and convergence performance. It would be interesting to investigate if and how well-known compression strategies (such as error compensation in Chapter 4 and 5, and dithered compression [201]) can guarantee high privacy and communication efficiency without sacrificing convergence performance.

References

[1] L. Bottou, F. E. Curtis, and J. Nocedal, "Optimization methods for large-scale machine learning," *Siam Review*, vol. 60, no. 2, pp. 223–311, 2018.

[2] A. Robicquet, A. Sadeghian, A. Alahi, and S. Savarese, "Learning social etiquette: Human trajectory understanding in crowded scenes," in *European conference on computer vision*, pp. 549–565, Springer, 2016.

[3] V. Panayotov, G. Chen, D. Povey, and S. Khudanpur, "Librispeech: an ASR corpus based on public domain audio books," in *2015 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 5206–5210, IEEE, 2015.

[4] T. Bertin-Mahieux, D. P. Ellis, B. Whitman, and P. Lamere, "The million song dataset," in *Proceedings of the 12th International Conference on Music Information Retrieval (ISMIR 2011)*, 2011.

[5] G.-Y. Wei, D. Brooks, *et al.*, "Benchmarking TPU, GPU, and CPU platforms for deep learning," *arXiv preprint arXiv:1907.10701*, 2019.

[6] J. H. Park, S. Kim, J. Lee, M. Jeon, and S. H. Noh, "Accelerated training for cnn distributed deep learning through automatic resource-aware layer placement," *arXiv preprint arXiv:1901.05803*, 2019.

[7] J. Dongarra *et al.*, "MPI: a message-passing interface standard version 3.0," *High Performance Computing Center Stuttgart (HLRS)*, 2013.

[8] J. Dean and S. Ghemawat, "Mapreduce: simplified data processing on large clusters," *Communications of the ACM*, vol. 51, no. 1, pp. 107–113, 2008.

[9] M. Zaharia, R. S. Xin, P. Wendell, T. Das, M. Armbrust, A. Dave, X. Meng, J. Rosen, S. Venkataraman, M. J. Franklin, *et al.*, "Apache spark: a unified engine for big data processing," *Communications of the ACM*, vol. 59, no. 11, pp. 56–65, 2016.

[10] A. Aytekin and M. Johansson, "Harnessing the power of serverless runtimes for large-scale optimization," *arXiv preprint arXiv:1901.03161*, 2019.

[11] A. Aytekin, *Asynchronous First-Order Algorithms for Large-Scale Optimization: Analysis and Implementation*. PhD thesis, KTH Royal Institute of Technology, 2019.

[12] L. Dagum and R. Menon, "Openmp: an industry standard api for shared-memory programming," *IEEE computational science and engineering*, vol. 5, no. 1, pp. 46–55, 1998.

[13] J. Protic, M. Tomasevic, and V. Milutinovic, "Distributed shared memory: Concepts and systems," *IEEE Parallel & Distributed Technology: Systems & Applications*, vol. 4, no. 2, pp. 63–71, 1996.

[14] E. Gabriel, G. E. Fagg, G. Bosilca, T. Angskun, J. J. Dongarra, J. M. Squyres, V. Sahay, P. Kambadur, B. Barrett, A. Lumsdaine, *et al.*, "Open mpi: Goals, concept, and design of a next generation mpi implementation," in *European Parallel Virtual Machine/Message Passing Interface Users' Group Meeting*, pp. 97–104, Springer, 2004.

[15] P. Hintjens, *ZeroMQ: messaging for many applications.* " O'Reilly Media, Inc.", 2013.

[16] T. Chilimbi, Y. Suzue, J. Apacible, and K. Kalyanaraman, "Project adam: Building an efficient and scalable deep learning training system," in *11th {USENIX} Symposium on Operating Systems Design and Implementation ({OSDI} 14)*, pp. 571–582, 2014.

[17] M. Abadi, A. Agarwal, P. Barham, E. Brevdo, Z. Chen, C. Citro, G. S. Corrado, A. Davis, J. Dean, M. Devin, *et al.*, "Tensorflow: Large-scale machine learning on heterogeneous distributed systems," *arXiv preprint arXiv:1603.04467*, 2016.

[18] R. Gu, S. Yang, and F. Wu, "Distributed machine learning on mobile devices: A survey," *arXiv preprint arXiv:1909.08329*, 2019.

[19] A. Ghoting, R. Krishnamurthy, E. Pednault, B. Reinwald, V. Sindhwani, S. Tatikonda, Y. Tian, and S. Vaithyanathan, "SystemML: declarative machine learning on mapreduce," in *2011 IEEE 27th International Conference on Data Engineering*, pp. 231–242, IEEE, 2011.

[20] X. Meng, J. Bradley, B. Yavuz, E. Sparks, S. Venkataraman, D. Liu, J. Freeman, D. Tsai, M. Amde, S. Owen, *et al.*, "Mllib: Machine learning in apache spark," *The Journal of Machine Learning Research*, vol. 17, no. 1, pp. 1235–1241, 2016.

[21] A. Sergeev and M. Del Balso, "Horovod: fast and easy distributed deep learning in tensorflow," *arXiv preprint arXiv:1802.05799*, 2018.

[22] 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*, pp. 1709–1720, 2017.

[23] Y. Zhang, N. Suda, L. Lai, and V. Chandra, "Hello edge: Keyword spotting on microcontrollers," *arXiv preprint arXiv:1711.07128*, 2017.

[24] K. Bhardwaj, C.-Y. Lin, A. Sartor, and R. Marculescu, "Memory-and communication-aware model compression for distributed deep learning inference on iot," *ACM Transactions on Embedded Computing Systems (TECS)*, vol. 18, no. 5s, p. 82, 2019.

[25] T. Chen, G. Giannakis, T. Sun, and W. Yin, "Lag: Lazily aggregated gradient for communication-efficient distributed learning," in *Advances in Neural Information Processing Systems*, pp. 5050–5060, 2018.

[26] A. Aytekin, H. R. Feyzmahdavian, and M. Johansson, "Analysis and implementation of an asynchronous optimization algorithm for the parameter server," *arXiv preprint arXiv:1610.05507*, 2016.

[27] M. Gurbuzbalaban, A. Ozdaglar, and P. A. Parrilo, "On the convergence rate of incremental aggregated gradient algorithms," *SIAM Journal on Optimization*, vol. 27, no. 2, pp. 1035–1048, 2017.

[28] F. Seide, H. Fu, J. Droppo, G. Li, and D. Yu, "1-bit stochastic gradient descent and its application to data-parallel distributed training of speech dnns," in *Fifteenth Annual Conference of the International Speech Communication Association*, 2014.

[29] 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*, pp. 3252–3261, PMLR, 2019.

[30] J. Wangni, J. Wang, J. Liu, and T. Zhang, "Gradient sparsification for communication-efficient distributed optimization," in *Advances in Neural Information Processing Systems*, pp. 1299–1309, 2018.

[31] C. Zhu, S. Han, H. Mao, and W. J. Dally, "Trained ternary quantization," *arXiv preprint arXiv:1612.01064*, 2016.

[32] S. Boyd and L. Vandenberghe, *Convex optimization*. Cambridge university press, 2004.

[33] Y. Nesterov, "Introductory lectures on convex programming volume i: Basic course," *Lecture notes*, vol. 3, no. 4, p. 5, 1998.

[34] B. T. Polyak, "Introduction to optimization. optimization software," *Inc., Publications Division, New York*, vol. 1, 1987.

[35] D. P. Bertsekas, "Incremental gradient, subgradient, and proximal methods for convex optimization: A survey," *Optimization for Machine Learning*, vol. 2010, no. 1-38, p. 3, 2011.

[36] L. Bottou, "Large-scale machine learning with stochastic gradient descent," in *Proceedings of COMPSTAT'2010*, pp. 177–186, Springer, 2010.

[37] X. Qian, P. Richtarik, R. Gower, A. Sailanbayev, N. Loizou, and E. Shulgin, "SGD with arbitrary sampling: General analysis and improved rates," in *International Conference on Machine Learning*, pp. 5200–5209, 2019.

[38] R. M. Gower, N. Loizou, X. Qian, A. Sailanbayev, E. Shulgin, and P. Richtárik, "SGD: General analysis and improved rates," in *International Conference on Machine Learning*, pp. 5200–5209, 2019.

[39] S. Khirirat, "Randomized first-order methods for convex optimization: Improved convergence rate bounds and experimental evaluations," 2016.

[40] O. Dekel, R. Gilad-Bachrach, O. Shamir, and L. Xiao, "Optimal distributed online prediction using mini-batches," *Journal of Machine Learning Research*, vol. 13, no. Jan, pp. 165–202, 2012.

[41] R. Johnson and T. Zhang, "Accelerating stochastic gradient descent using predictive variance reduction," in *Advances in neural information processing systems*, pp. 315–323, 2013.

[42] A. Defazio, F. Bach, and S. Lacoste-Julien, "SAGA: A fast incremental gradient method with support for non-strongly convex composite objectives," in *Advances in neural information processing systems*, pp. 1646–1654, 2014.

[43] Z. Charles and D. Papailiopoulos, "Gradient coding using the stochastic block model," in *2018 IEEE International Symposium on Information Theory (ISIT)*, pp. 1998–2002, IEEE, 2018.

[44] N. Raviv, I. Tamo, R. Tandon, and A. G. Dimakis, "Gradient coding from cyclic mds codes and expander graphs," *IEEE Transactions on Information Theory*, vol. 66, no. 12, pp. 7475–7489, 2020.

[45] R. Tandon, Q. Lei, A. G. Dimakis, and N. Karampatziakis, "Gradient coding: Avoiding stragglers in distributed learning," in *International Conference on Machine Learning*, pp. 3368–3376, 2017.

[46] C. Karakus, Y. Sun, S. Diggavi, and W. Yin, "Straggler mitigation in distributed optimization through data encoding," in *Advances in Neural Information Processing Systems*, pp. 5434–5442, 2017.

[47] R. Bitar, M. Wootters, and S. El Rouayheb, "Stochastic gradient coding for straggler mitigation in distributed learning," *IEEE Journal on Selected Areas in Information Theory*, vol. 1, no. 1, pp. 277–291, 2020.

[48] K. Lee, M. Lam, R. Pedarsani, D. Papailiopoulos, and K. Ramchandran, "Speeding up distributed machine learning using codes," *IEEE Transactions on Information Theory*, vol. 64, no. 3, pp. 1514–1529, 2017.

[49] G. B. Passty, "Ergodic convergence to a zero of the sum of monotone operators in hilbert space," *Journal of Mathematical Analysis and Applications*, vol. 72, no. 2, pp. 383–390, 1979.

[50] D. Davis and W. Yin, "Faster convergence rates of relaxed peaceman-rachford and ADMM under regularity assumptions," *Mathematics of Operations Research*, vol. 42, no. 3, pp. 783–805, 2017.

[51] A. Beck and M. Teboulle, "A fast iterative shrinkage-thresholding algorithm for linear inverse problems," *SIAM journal on imaging sciences*, vol. 2, no. 1, pp. 183–202, 2009.

[52] L. Xiao and T. Zhang, "A proximal stochastic gradient method with progressive variance reduction," *SIAM Journal on Optimization*, vol. 24, no. 4, pp. 2057–2075, 2014.

[53] J. C. Duchi and Y. Singer, "Efficient learning using forward-backward splitting.," in *NIPS*, vol. 22, pp. 495–503, 2009.

[54] X. Wang, S. Wang, and H. Zhang, "Inexact proximal stochastic gradient method for convex composite optimization," *Computational Optimization and Applications*, vol. 68, no. 3, pp. 579–618, 2017.

[55] A. Nitanda, "Stochastic proximal gradient descent with acceleration techniques," *Advances in Neural Information Processing Systems*, vol. 27, pp. 1574–1582, 2014.

[56] J. Konečnỳ, J. Liu, P. Richtárik, and M. Takáč, "Mini-batch semi-stochastic gradient descent in the proximal setting," *IEEE Journal of Selected Topics in Signal Processing*, vol. 10, no. 2, pp. 242–255, 2015.

[57] N. D. Vanli, M. Gurbuzbalaban, and A. Ozdaglar, "Global convergence rate of proximal incremental aggregated gradient methods," *SIAM Journal on Optimization*, vol. 28, no. 2, pp. 1282–1300, 2018.

[58] W. Peng, H. Zhang, and X. Zhang, "Nonconvex proximal incremental aggregated gradient method with linear convergence," *Journal of Optimization Theory and Applications*, vol. 183, no. 1, pp. 230–245, 2019.

[59] J. Douglas and H. H. Rachford, "On the numerical solution of heat conduction problems in two and three space variables," *Transactions of the American mathematical Society*, vol. 82, no. 2, pp. 421–439, 1956.

[60] D. W. Peaceman and H. H. Rachford, Jr, "The numerical solution of parabolic and elliptic differential equations," *Journal of the Society for industrial and Applied Mathematics*, vol. 3, no. 1, pp. 28–41, 1955.

[61] S. Liu, S. P. Chepuri, M. Fardad, E. Maşazade, G. Leus, and P. K. Varshney, "Sensor selection for estimation with correlated measurement noise," *IEEE Transactions on Signal Processing*, vol. 64, no. 13, pp. 3509–3522, 2016.

[62] A. O. Hero and D. Cochran, "Sensor management: Past, present, and future," *IEEE Sensors Journal*, vol. 11, no. 12, pp. 3064–3075, 2011.

[63] S. Liu, J. Chen, P.-Y. Chen, and A. Hero, "Zeroth-order online alternating direction method of multipliers: Convergence analysis and applications," in *International Conference on Artificial Intelligence and Statistics*, pp. 288–297, PMLR, 2018.

[64] J. Snoek, H. Larochelle, and R. P. Adams, "Practical bayesian optimization of machine learning algorithms," *Advances in neural information processing systems*, vol. 25, 2012.

[65] J. Bergstra, R. Bardenet, Y. Bengio, and B. Kégl, "Algorithms for hyper-parameter optimization," *Advances in neural information processing systems*, vol. 24, 2011.

[66] P.-Y. Chen, H. Zhang, Y. Sharma, J. Yi, and C.-J. Hsieh, "Zoo: Zeroth order optimization based black-box attacks to deep neural networks without training substitute models," in *Proceedings of the 10th ACM workshop on artificial intelligence and security*, pp. 15–26, 2017.

[67] N. Papernot, P. McDaniel, I. Goodfellow, S. Jha, Z. B. Celik, and A. Swami, "Practical black-box attacks against machine learning," in *Proceedings of the 2017 ACM on Asia conference on computer and communications security*, pp. 506–519, 2017.

[68] W. Hu and Y. Tan, "Generating adversarial malware examples for black-box attacks based on GAN," *arXiv preprint arXiv:1702.05983*, 2017.

[69] A. Ilyas, L. Engstrom, A. Athalye, and J. Lin, "Black-box adversarial attacks with limited queries and information," in *International Conference on Machine Learning*, pp. 2137–2146, PMLR, 2018.

[70] J. A. Nelder and R. Mead, "A Simplex Method for Function Minimization," *The Computer Journal*, vol. 7, pp. 308–313, 01 1965.

[71] C. Audet and J. E. Dennis, "Mesh adaptive direct search algorithms for constrained optimization," *SIAM Journal on Optimization*, vol. 17, no. 1, pp. 188–217, 2006.

[72] B. Polyak, *Introduction to Optimization*. 07 2020.

[73] Y. Nesterov, *Introductory lectures on convex optimization: A basic course*, vol. 87. Springer Science & Business Media, 2013.

[74] J. Nocedal and S. J. Wright, *Numerical optimization*. Springer, 1999.

[75] J. N. Tsitsiklis and Z.-Q. Luo, "Communication complexity of convex optimization," *Journal of Complexity*, vol. 3, no. 3, pp. 231–243, 1987.

[76] N. Li, L. Chen, and S. H. Low, "Optimal demand response based on utility maximization in power networks," in *2011 IEEE power and energy society general meeting*, pp. 1–8, IEEE, 2011.

[77] R. Madan and S. Lall, "Distributed algorithms for maximum lifetime routing in wireless sensor networks," *IEEE Transactions on wireless communications*, vol. 5, no. 8, pp. 2185–2193, 2006.

[78] S. H. Low and D. E. Lapsley, "Optimization flow control. i. basic algorithm and convergence," *IEEE/ACM Transactions on Networking (TON)*, vol. 7, no. 6, pp. 861–874, 1999.

[79] M. Chiang, S. H. Low, A. R. Calderbank, and J. C. Doyle, "Layering as optimization decomposition: A mathematical theory of network architectures," *Proceedings of the IEEE*, vol. 95, no. 1, pp. 255–312, 2007.

[80] D. P. Palomar and M. Chiang, "A tutorial on decomposition methods for network utility maximization," *IEEE Journal on Selected Areas in Communications*, vol. 24, no. 8, pp. 1439–1451, 2006.

[81] C. Zhao, U. Topcu, N. Li, and S. Low, "Design and stability of load-side primary frequency control in power systems," *IEEE Transactions on Automatic Control*, vol. 59, no. 5, pp. 1177–1189, 2014.

[82] S. Magnússon, C. Enyioha, N. Li, C. Fischione, and V. Tarokh, "Convergence of limited communications gradient methods," *IEEE Transactions on Automatic Control*, 2017.

[83] S. Magnússon, C. Enyioha, K. Heal, N. Li, C. Fischione, and V. Tarokh, "Distributed resource allocation using one-way communication with applications to power networks," in *2016 Annual Conference on Information Science and Systems (CISS)*, pp. 631–636, IEEE, 2016.

[84] J. Chen, X. Pan, R. Monga, S. Bengio, and R. Jozefowicz, "Revisiting distributed synchronous SGD," *arXiv preprint arXiv:1604.00981*, 2016.

[85] S. Zhang, C. Zhang, Z. You, R. Zheng, and B. Xu, "Asynchronous stochastic gradient descent for DNN training," in *2013 IEEE International Conference on Acoustics, Speech and Signal Processing*, pp. 6660–6663, IEEE, 2013.

[86] S. Dutta, G. Joshi, S. Ghosh, P. Dube, and P. Nagpurkar, "Slow and stale gradients can win the race: Error-runtime trade-offs in distributed SGD," in *International Conference on Artificial Intelligence and Statistics*, pp. 803–812, PMLR, 2018.

[87] A. Aytekin, M. Biel, and M. Johansson, "POLO: a policy-based optimization library," *arXiv preprint arXiv:1810.03417*, 2018.

[88] X. Lian, Y. Huang, Y. Li, and J. Liu, "Asynchronous parallel stochastic gradient for nonconvex optimization," in *Advances in Neural Information Processing Systems*, pp. 2737–2745, 2015.

[89] Y. Ma, F. Rusu, and M. Torres, "Stochastic gradient descent on highly-parallel architectures," *arXiv preprint arXiv:1802.08800*, 2018.

[90] B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y Arcas, "Communication-efficient learning of deep networks from decentralized data," in *Artificial Intelligence and Statistics*, pp. 1273–1282, PMLR, 2017.

[91] X. Li, K. Huang, W. Yang, S. Wang, and Z. Zhang, "On the convergence of FedAvg on Non-IID data," in *International Conference on Learning Representations*, 2020.

[92] S. U. Stich, "Local SGD converges fast and communicates little," in *International Conference on Learning Representations*, 2019.

[93] A. Khaled, K. Mishchenko, and P. Richtárik, "First analysis of local GD on heterogeneous data," *arXiv preprint arXiv:1909.04715*, 2019.

[94] T. Li, A. K. Sahu, M. Zaheer, M. Sanjabi, A. Talwalkar, and V. Smith, "Federated optimization in heterogeneous networks," *Proceedings of Machine Learning and Systems*, vol. 2, pp. 429–450, 2020.

[95] W. Wen, C. Xu, F. Yan, C. Wu, Y. Wang, Y. Chen, and H. Li, "Terngrad: Ternary gradients to reduce communication in distributed deep learning," in *Advances in neural information processing systems*, pp. 1509–1519, 2017.

[96] H. Wang, S. Sievert, Z. Charles, D. Papailiopoulos, and S. Wright, "Atomo: Communication-efficient learning via atomic sparsification," *arXiv preprint arXiv:1806.04090*, 2018.

[97] J. Bernstein, Y.-X. Wang, K. Azizzadenesheli, and A. Anandkumar, "signSGD: Compressed optimisation for non-convex problems," pp. 560–569, 2018.

[98] R. Zhang and J. Kwok, "Asynchronous distributed ADMM for consensus optimization," in *International Conference on Machine Learning*, pp. 1701–1709, 2014.

[99] M. Li, D. G. Andersen, A. J. Smola, and K. Yu, "Communication efficient distributed machine learning with the parameter server," in *Advances in Neural Information Processing Systems*, pp. 19–27, 2014.

[100] D. Blatt, A. O. Hero, and H. Gauchman, "A convergent incremental gradient method with a constant step size," *SIAM Journal on Optimization*, vol. 18, no. 1, pp. 29–51, 2007.

[101] P. Tseng and S. Yun, "Incrementally updated gradient methods for constrained and regularized optimization," *J. Optimization Theory and Applications*, vol. 160, no. 3, pp. 832–853, 2014.

[102] A. Ben-Tal and A. Nemirovski, *Lectures on modern convex optimization: analysis, algorithms, and engineering applications*, vol. 2. Siam, 2001.

[103] J. Wangni, J. Wang, J. Liu, and T. Zhang, "Gradient sparsification for communication-efficient distributed optimization," in *Advances in Neural Information Processing Systems*, vol. 31, 2018.

[104] S. Khirirat, M. Johansson, and D. Alistarh, "Gradient compression for communication-limited convex optimization," in *2018 IEEE Conference on Decision and Control (CDC)*, pp. 166–171, IEEE, 2018.

[105] C. M. De Sa, C. Zhang, K. Olukotun, and C. Ré, "Taming the wild: A unified analysis of hogwild-style algorithms," in *Advances in neural information processing systems*, pp. 2674–2682, 2015.

[106] M. F. Balcan, A. Blum, S. Fine, and Y. Mansour, "Distributed learning, communication complexity and privacy," in *Conference on Learning Theory*, pp. 26–1, 2012.

[107] J. N. Tsitsiklis and Z.-Q. Luo, "Communication complexity of convex optimization," *Journal of Complexity*, vol. 3, no. 3, pp. 231–243, 1987.

[108] Y. Arjevani and O. Shamir, "Communication complexity of distributed convex learning and optimization," in *Advances in neural information processing systems*, pp. 1756–1764, 2015.

[109] A. Bellet, Y. Liang, A. B. Garakani, M.-F. Balcan, and F. Sha, "A distributed frank-wolfe algorithm for communication-efficient sparse learning," in *Proceedings of the 2015 SIAM International Conference on Data Mining*, pp. 478–486, SIAM, 2015.

[110] J. Nutini, M. Schmidt, I. Laradji, M. Friedlander, and H. Koepke, "Coordinate descent converges faster with the gauss-southwell rule than random selection," in *International Conference on Machine Learning*, pp. 1632–1641, 2015.

[111] E. J. Candès and B. Recht, "Exact matrix completion via convex optimization," *Foundations of Computational mathematics*, vol. 9, no. 6, p. 717, 2009.

[112] B. T. Polyak, "Introduction to optimization. translations series in mathematics and engineering," *Optimization Software*, 1987.

[113] S. Khirirat, H. R. Feyzmahdavian, and M. Johansson, "Mini-batch gradient descent: Faster convergence under data sparsity," in *2017 IEEE 56th Annual Conference on Decision and Control (CDC)*, pp. 2880–2887, IEEE, 2017.

[114] A. Aytekin, H. R. Feyzmahdavian, and M. Johansson, "Asynchronous incremental block-coordinate descent," in *Communication, Control, and Computing (Allerton), 2014 52nd Annual Allerton Conference on*, pp. 19–24, IEEE, 2014.

[115] Sarit Khirirat, Mikael Johansson, and Dan Alistarh, "Gradient compression for communication-limited convex optimization," in *2018 IEEE Conference on Decision and Control (CDC)*, pp. 166–171, Dec 2018.

[116] S. Magnússon, H. Shokri-Ghadikolaei, and N. Li, "On maintaining linear convergence of distributed learning and optimization under limited communication," in *2019 53rd Asilomar Conference on Signals, Systems, and Computers*, pp. 432–436, IEEE, 2019.

[117] S. Khirirat, H. R. Feyzmahdavian, and M. Johansson, "Distributed learning with compressed gradients," *arXiv preprint arXiv:1806.06573*, 2018.

[118] A. Koloskova, S. Stich, and M. Jaggi, "Decentralized stochastic optimization and gossip algorithms with compressed communication," in *International Conference on Machine Learning*, pp. 3478–3487, PMLR, 2019.

[119] T. T. Doan, S. T. Maguluri, and J. Romberg, "Fast convergence rates of distributed subgradient methods with adaptive quantization," *IEEE Transactions on Automatic Control*, 2020.

[120] A. Reisizadeh, H. Taheri, A. Mokhtari, H. Hassani, and R. Pedarsani, "Robust and communication-efficient collaborative learning," in *Advances in Neural Information Processing Systems*, pp. 8386–8397, 2019.

[121] X. Zhang, J. Liu, Z. Zhu, and E. S. Bentley, "Compressed distributed gradient descent: Communication-efficient consensus over networks," in *IEEE INFOCOM 2019-IEEE Conference on Computer Communications*, pp. 2431–2439, IEEE, 2019.

[122] A. Reisizadeh, A. Mokhtari, H. Hassani, and R. Pedarsani, "An exact quantized decentralized gradient descent algorithm," *IEEE Transactions on Signal Processing*, vol. 67, no. 19, pp. 4934–4947, 2019.

[123] N. Strom, "Scalable distributed DNN training using commodity GPU cloud computing," in *Sixteenth Annual Conference of the International Speech Communication Association*, 2015.

[124] D. Alistarh, T. Hoefler, M. Johansson, N. Konstantinov, S. Khirirat, and C. Renggli, "The convergence of sparsified gradient methods," in *Advances in Neural Information Processing Systems*, pp. 5977–5987, 2018.

[125] 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*, pp. 5325–5333, PMLR, 2018.

[126] S. U. Stich, J.-B. Cordonnier, and M. Jaggi, "Sparsified SGD with memory," in *Advances in Neural Information Processing Systems*, pp. 4452–4463, 2018.

[127] S. Praneeth Karimireddy, Q. Rebjock, S. U. Stich, and M. Jaggi, "Error Feedback Fixes SignSGD and other Gradient Compression Schemes," *arXiv preprint arXiv:1901.09847*, 2019.

[128] M. G. Rabbat and R. D. Nowak, "Quantized incremental algorithms for distributed optimization," *IEEE Journal on Selected Areas in Communications*, vol. 23, no. 4, pp. 798–808, 2005.

[129] S. Zhu, M. Hong, and B. Chen, "Quantized consensus ADMM for multi-agent distributed optimization," in *Acoustics, Speech and Signal Processing (ICASSP), 2016 IEEE International Conference on*, pp. 4134–4138, IEEE, 2016.

[130] H. Li, S. De, Z. Xu, C. Studer, H. Samet, and T. Goldstein, "Training quantized nets: A deeper understanding," in *Advances in Neural Information Processing Systems*, pp. 5811–5821, 2017.

[131] C. De Sa, M. Leszczynski, J. Zhang, A. Marzoev, C. R. Aberger, K. Olukotun, and C. Ré, "High-accuracy low-precision training," *arXiv preprint arXiv:1803.03383*, 2018.

[132] H. R. Feyzmahdavian, A. Aytekin, and M. Johansson, "An asynchronous mini-batch algorithm for regularized stochastic optimization," *IEEE Transactions on Automatic Control*, vol. 61, no. 12, pp. 3740–3754, 2016.

[133] S. Khirirat, S. Magnússon, and M. Johansson, "Convergence bounds for compressed gradient methods with memory based error compensation," in *ICASSP 2019-2019 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 2857–2861, IEEE, 2019.

[134] D. D. Lewis, Y. Yang, T. G. Rose, and F. Li, "Rcv1: A new benchmark collection for text categorization research," *Journal of machine learning research*, vol. 5, no. Apr, pp. 361–397, 2004.

[135] H. Tang, C. Yu, X. Lian, T. Zhang, and J. Liu, "Doublesqueeze: Parallel stochastic gradient descent with double-pass error-compensated compression," in *International Conference on Machine Learning*, pp. 6155–6165, PMLR, 2019.

[136] H. Tang, X. Lian, S. Qiu, L. Yuan, C. Zhang, T. Zhang, and J. Liu, "DeepSqueeze: Decentralization meets error-compensated compression," *arXiv preprint arXiv:1907.07346*, 2019.

[137] J. Nocedal and S. Wright, *Numerical optimization*. Springer Science & Business Media, 2006.

[138] P. Xu, F. Roosta, and M. W. Mahoney, "Newton-type methods for non-convex optimization under inexact hessian information," *Mathematical Programming*, pp. 1–36, 2019.

[139] A. E. Beaton and J. W. Tukey, "The fitting of power series, meaning polynomials, illustrated on band-spectroscopic data," *Technometrics*, vol. 16, no. 2, pp. 147–185, 1974.

[140] C.-C. Chang and C.-J. Lin, "Libsvm: a library for support vector machines," *ACM transactions on intelligent systems and technology (TIST)*, vol. 2, no. 3, p. 27, 2011.

[141] A. Rakhlin, O. Shamir, and K. Sridharan, "Making gradient descent optimal for strongly convex stochastic optimization," in *Proceedings of the 29th International Coference on International Conference on Machine Learning*, pp. 1571–1578, 2012.

[142] L. Nguyen, P. H. Nguyen, M. Dijk, P. Richtárik, K. Scheinberg, and M. Takác, "SGD and hogwild! convergence without the bounded gradients assumption," in *International Conference on Machine Learning*, pp. 3750–3758, PMLR, 2018.

[143] M. Schmidt, N. Le Roux, and F. Bach, "Minimizing finite sums with the stochastic average gradient," *Mathematical Programming*, vol. 162, no. 1-2, pp. 83–112, 2017.

[144] S. Vaswani, A. Mishkin, I. Laradji, M. Schmidt, G. Gidel, and S. Lacoste-Julien, "Painless stochastic gradient: Interpolation, line-search, and convergence rates," in *Advances in Neural Information Processing Systems*, pp. 3732–3745, 2019.

[145] J. Konečnỳ, H. B. McMahan, D. Ramage, and P. Richtárik, "Federated optimization: Distributed machine learning for on-device intelligence," *arXiv preprint arXiv:1610.02527*, 2016.

[146] T. Li, A. K. Sahu, A. Talwalkar, and V. Smith, "Federated learning: Challenges, methods, and future directions," *IEEE Signal Processing Magazine*, vol. 37, no. 3, pp. 50–60, 2020.

[147] J. Azar, A. Makhoul, M. Barhamgi, and R. Couturier, "An energy efficient IoT data compression approach for edge machine learning," *Future Generation Computer Systems*, vol. 96, pp. 168–175, 2019.

[148] M. A. Razzaque, C. Bleakley, and S. Dobson, "Compression in wireless sensor networks: A survey and comparative evaluation," *ACM Transactions on Sensor Networks (TOSN)*, vol. 10, no. 1, pp. 1–44, 2013.

[149] D. Rothchild, A. Panda, E. Ullah, N. Ivkin, I. Stoica, V. Braverman, J. Gonzalez, and R. Arora, "FetchSGD: Communication-efficient federated learning with sketching," in *International Conference on Machine Learning*, pp. 8253–8265, PMLR, 2020.

[150] R. Pathak and M. J. Wainwright, "Fedsplit: an algorithmic framework for fast federated optimization," in *Advances in Neural Information Processing Systems*, vol. 33, 2020.

[151] S. U. Stich, J.-B. Cordonnier, and M. Jaggi, "Sparsified SGD with memory," in *Advances in Neural Information Processing Systems*, pp. 4447–4458, 2018.

[152] S. Khirirat, S. Magnússon, and M. Johansson, "Compressed gradient methods with hessian-aided error compensation," *IEEE Transactions on Signal Processing*, vol. 69, pp. 998–1011, 2020.

[153] J. Eckstein, *Splitting methods for monotone operators with applications to parallel optimization*. PhD thesis, Massachusetts Institute of Technology, 1989.

[154] S. Zhu and B. Chen, "Quantized consensus by the ADMM: Probabilistic versus deterministic quantizers," *IEEE Transactions on Signal Processing*, vol. 64, no. 7, pp. 1700–1713, 2015.

[155] C.-C. Chang and C.-J. Lin, "LIBSVM: A library for support vector machines," *ACM Transactions on Intelligent Systems and Technology*, vol. 2, pp. 27:1–27:27, 2011. Software available at http://www.csie.ntu.edu.tw/ cjlin/libsvm.

[156] S. Pu and A. Nedić, "Distributed stochastic gradient tracking methods," *Mathematical Programming*, pp. 1–49, 2020.

[157] T. Vogels, S. P. Karimireddy, and M. Jaggi, "Practical low-rank communication compression in decentralized deep learning," *Advances in Neural Information Processing Systems*, vol. 33, pp. 14171–14181, 2020.

[158] S. Magnússon, H. Shokri-Ghadikolaei, and N. Li, "On maintaining linear convergence of distributed learning and optimization under limited communication," *IEEE Transactions on Signal Processing*, vol. 68, pp. 6101–6116, 2020.

[159] H. Yu, S. Yang, and S. Zhu, "Parallel restarted SGD with faster convergence and less communication: Demystifying why model averaging works for deep learning," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 33, pp. 5693–5700, 2019.

[160] J. Wang and G. Joshi, "Cooperative SGD: A unified framework for the design and analysis of local-update sgd algorithms," *Journal of Machine Learning Research*, vol. 22, no. 213, pp. 1–50, 2021.

[161] J. Wang and G. Joshi, "Adaptive communication strategies to achieve the best error-runtime trade-off in local-update SGD," in *Proceedings of Machine Learning and Systems* (A. Talwalkar, V. Smith, and M. Zaharia, eds.), vol. 1, pp. 212–229, 2019.

[162] S. Shi, Q. Wang, K. Zhao, Z. Tang, Y. Wang, X. Huang, and X. Chu, "A distributed synchronous SGD algorithm with global top-k sparsification for low bandwidth networks," *arXiv preprint arXiv:1901.04359*, 2019.

[163] S. Shi, K. Zhao, Q. Wang, Z. Tang, and X. Chu, "A convergence analysis of distributed sgd with communication-efficient gradient sparsification," in *Proceedings of the Twenty-Eighth International Joint Conference on Artificial Intelligence, IJCAI-19*, pp. 3411–3417, 7 2019.

[164] A. Kozłowski and J. Sosnowski, "Analysing efficiency of IPv6 packet transmission over 6LoWPAN network," in *Photonics Applications in Astronomy, Communications, Industry, and High Energy Physics Experiments 2017* (R. S. Romaniuk and M. Linczuk, eds.), pp. 456 – 467, SPIE, 2017.

[165] D. P. Bertsekas, *Nonlinear programming: 2nd Edition*. Athena Scientific, 1999.

[166] Y. Nesterov, *Lectures on convex optimization*, vol. 137. Springer, 2018.

[167] S. Schaible, *Fractional programming*, pp. 605–608. Boston, MA: Springer US, 2013.

[168] M. Avriel, W. E. Diewert, S. Schaible, and I. Zang, *Generalized concavity*. Society for Industrial and Applied Mathematics, 2010.

[169] E. Moulines and F. R. Bach, "Non-asymptotic analysis of stochastic approximation algorithms for machine learning," in *Advances in Neural Information Processing Systems*, pp. 451–459, 2011.

[170] A. Nemirovski, A. Juditsky, G. Lan, and A. Shapiro, "Robust stochastic approximation approach to stochastic programming," *SIAM Journal on optimization*, vol. 19, no. 4, pp. 1574–1609, 2009.

[171] H. Robbins and S. Monro, "A stochastic approximation method," *The annals of mathematical statistics*, pp. 400–407, 1951.

[172] A. Rakhlin, O. Shamir, and K. Sridharan, "Making gradient descent optimal for strongly convex stochastic optimization," in *Proceedings of the 29th International Coference on International Conference on Machine Learning*, pp. 1571–1578, 2012.

[173] S. Lacoste-Julien, M. Schmidt, and F. Bach, "A simpler approach to obtaining an o (1/t) convergence rate for the projected stochastic subgradient method," *arXiv preprint arXiv:1212.2002*, 2012.

[174] O. Shamir and T. Zhang, "Stochastic gradient descent for non-smooth optimization: Convergence results and optimal averaging schemes," in *International conference on machine learning*, pp. 71–79, 2013.

[175] A. Albasyoni, M. Safaryan, L. Condat, and P. Richtárik, "Optimal gradient compression for distributed and federated learning," *arXiv preprint arXiv:2010.03246*, 2020.

[176] C. Karakus, Y. Sun, S. N. Diggavi, and W. Yin, "Redundancy techniques for straggler mitigation in distributed optimization and learning.," *Journal of Machine Learning Research*, vol. 20, no. 72, pp. 1–47, 2019.

[177] M. Ye and E. Abbe, "Communication-computation efficient gradient coding," *arXiv preprint arXiv:1802.03475*, 2018.

[178] Y. Nesterov and V. Spokoiny, "Random gradient-free minimization of convex functions," *Foundations of Computational Mathematics*, vol. 17, no. 2, pp. 527–566, 2017.

[179] S. Ghadimi and G. Lan, "Stochastic first-and zeroth-order methods for nonconvex stochastic programming," *SIAM Journal on Optimization*, vol. 23, no. 4, pp. 2341–2368, 2013.

[180] S.-M. Moosavi-Dezfooli, A. Fawzi, O. Fawzi, and P. Frossard, "Universal adversarial perturbations," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 1765–1773, 2017.

[181] A. Khaled, O. Sebbouh, N. Loizou, R. M. Gower, and P. Richtárik, "Unified analysis of stochastic gradient methods for composite convex and smooth optimization," *arXiv preprint arXiv:2006.11573*, 2020.

[182] O. Sebbouh, R. M. Gower, and A. Defazio, "Almost sure convergence rates for stochastic gradient descent and stochastic heavy ball," in *Conference on Learning Theory*, pp. 3935–3971, PMLR, 2021.

[183] S. Khirirat, S. Magnússon, A. Aytekin, and M. Johansson, "A flexible framework for communication-efficient machine learning," *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 35, pp. 8101–8109, May 2021.

[184] H. Ye, Z. Huang, C. Fang, C. J. Li, and T. Zhang, "Hessian-aware zeroth-order optimization for black-box adversarial attack," *arXiv preprint arXiv:1812.11377*, 2018.

[185] V. Kungurtsev and F. Rinaldi, "A zeroth order method for stochastic weakly convex optimization," *Computational Optimization and Applications*, pp. 1–23, 2021.

[186] D. Davis and D. Drusvyatskiy, "Stochastic model-based minimization of weakly convex functions," *SIAM Journal on Optimization*, vol. 29, no. 1, pp. 207–239, 2019.

[187] S. Pesme, A. Dieuleveut, and N. Flammarion, "On convergence-diagnostic based step sizes for stochastic gradient descent," in *International Conference on Machine Learning*, pp. 7641–7651, PMLR, 2020.

[188] M. Sordello, H. He, and W. Su, "Robust learning rate selection for stochastic optimization via splitting diagnostic," *arXiv preprint arXiv:1910.08597*, 2019.

[189] H. Lang, L. Xiao, and P. Zhang, "Using statistics to automate stochastic optimization," *Advances in Neural Information Processing Systems*, vol. 32, pp. 9540–9550, 2019.

[190] P. Zhang, H. Lang, Q. Liu, and L. Xiao, "Statistical adaptive stochastic gradient methods," *arXiv preprint arXiv:2002.10597*, 2020.

[191] B. Chmiel, L. Ben-Uri, M. Shkolnik, E. Hoffer, R. Banner, and D. Soudry, "Neural gradients are near-lognormal: improved quantized and sparse training," *arXiv preprint arXiv:2006.08173*, 2020.

[192] C. Paquette, K. Lee, F. Pedregosa, and E. Paquette, "SGD in the large: Average-case analysis, asymptotics, and stepsize criticality," in *Conference on Learning Theory*, pp. 3548–3626, PMLR, 2021.

[193] C. Paquette and E. Paquette, "Dynamics of stochastic momentum methods on large-scale, quadratic models," *Advances in Neural Information Processing Systems*, vol. 34, 2021.

[194] M. Velikanov and D. Yarotsky, "Explicit loss asymptotics in the gradient descent training of neural networks," in *Thirty-Fifth Conference on Neural Information Processing Systems*, 2021.

[195] M. Khodak, R. Tu, T. Li, L. Li, M.-F. Balcan, V. Smith, and A. Talwalkar, "Federated hyperparameter tuning: Challenges, baselines, and connections to weight-sharing," *arXiv preprint arXiv:2106.04502*, 2021.

[196] Z. Wang, H. Xu, J. Liu, H. Huang, C. Qiao, and Y. Zhao, "Resource-efficient federated learning with hierarchical aggregation in edge computing," in *IEEE INFOCOM 2021-IEEE Conference on Computer Communications*, pp. 1–10, IEEE, 2021.

[197] M. Fredrikson, S. Jha, and T. Ristenpart, "Model inversion attacks that exploit confidence information and basic countermeasures," in *Proceedings of the 22nd ACM SIGSAC conference on computer and communications security*, pp. 1322–1333, 2015.

[198] R. Shokri, M. Stronati, C. Song, and V. Shmatikov, "Membership inference attacks against machine learning models," in *2017 IEEE Symposium on Security and Privacy (SP)*, pp. 3–18, IEEE, 2017.

[199] F. Farokhi, "Gradient sparsification can improve performance of differentially-private convex machine learning," in *2021 60th IEEE Conference on Decision and Control (CDC)*, pp. 1695–1700, IEEE, 2021.

[200] T. Li, Z. Liu, V. Sekar, and V. Smith, "Privacy for free: Communication-efficient learning with differential privacy using sketches," *arXiv preprint arXiv:1911.00972*, 2019.

[201] A. Abdi and F. Fekri, "Nested dithered quantization for communication reduction in distributed training," *arXiv preprint arXiv:1904.01197*, 2019.