

Optimization methods for collaborative learning

Présentée le 30 septembre 2021

Faculté informatique et communications
Laboratoire d'apprentissage automatique et d'optimisation
Programme doctoral en informatique et communications

pour l'obtention du grade de Docteur ès Sciences

par

Sai Praneeth Reddy KARIMIREDDY

Acceptée sur proposition du jury

Prof. P. Thiran, président du jury
Prof. M. Jaggi, directeur de thèse
Prof. M. Jordan, rapporteur
Prof. M. Mohri, rapporteur
Prof. D. Kuhn, rapporteur

to my *ammamma* and *tatayya*, their playfulness, their stubbornness.

Acknowledgments

I will be eternally grateful to my parents, my partner, and advisor Martin. Their constant support and encouragement forms the foundation upon which this work is built. Nothing I can say will be enough. Other mentors such as my undergrad advisor Sandeep Sen and college teacher Ramesh Babu were also instrumental in setting me on this path. I am especially grateful to all my amazing collaborators: Sebastian, Thijs, Tao, Lie, Anastasia, Annie, Felix, Satyen, Mehryar, Theertha, Sashank, Jingzhao, Andreas, Seungyeon, Francesco, and Haihao. Thank you all for taking a chance on working with me, for sharing your passion and insights, and for teaching me most of my technical skills. But perhaps the more important skills I learnt, and certainly the experiences I most cherish, were due my friends—the joys and pitfalls of being. Thank you. Jennifer and all the other EDIC staff have been an absolute godsend. You have my sincere gratitude for bending the rules where possible and for your wholehearted assistance where not. Finally, I found both the Swiss people and the government to be warm and welcoming. Thank you for sharing your beautiful country with me. However, I wish it could be shared more freely with more people.

Lausanne, September 13, 2021

S.P.K.

Abstract

A traditional machine learning pipeline involves collecting massive amounts of data centrally on a server and training models to fit the data. However, increasing concerns about the privacy and security of user's data, combined with the sheer growth in the data sizes has incentivized looking beyond such traditional centralized approaches. Collaborative learning (which encompasses distributed, federated, and decentralized learning) proposes instead for a network of data holders to collaborate together to train models without transmitting any data. This new paradigm minimizes data exposure, but inherently faces some fundamental challenges. In this thesis, we bring to bear the framework of stochastic optimization to formalize and develop new algorithms for these challenges. This serves not only to develop novel solutions, but also to test the utility of the *optimization lens* in modern deep learning.

We study three fundamental problems. Firstly, collaborative training replaces a one-time transmission of raw data with repeated rounds of communicating partially trained models. However, this quickly runs against bandwidth constraints when dealing with large models. We propose to solve this bandwidth constraint using **compressed communication**. Next, collaborative training leverages the computation power of the data holders directly. However, this is not as reliable as using a data center with only a subset of them available at any given time. Thus, we require new algorithms which can efficiently utilize **unreliable local computation** of the data holders. Finally, collaborative training allows any data holder to participate in the training process, without being able to inspect their data or local computation. This may potentially open the system to malicious or faulty agents who seek to derail the training. We develop algorithms with **Byzantine robustness** which are guaranteed to be resilient to such attackers.

Keywords: Machine learning, optimization, collaborative learning, distributed learning, federated learning, Byzantine robustness.

Résumé

Une procédure d'apprentissage automatique traditionnelle implique la collecte de quantités massives de données de manière centralisée sur un serveur et l'apprentissage de modèles à partir de ces données. Cependant, les inquiétudes croissantes concernant la confidentialité et la sécurité des données des utilisateurs, combinées à l'augmentation de la taille des données, ont incité à regarder au-delà de ces approches centralisées traditionnelles. L'apprentissage collaboratif (qui englobe l'apprentissage distribué, fédéré et décentralisé) propose plutôt à un réseau de détenteurs de données de collaborer ensemble pour former des modèles sans transmettre de données. Ce nouveau paradigme minimise l'exposition des données, mais est confronté à des défis fondamentaux. Dans cette thèse, nous utilisons l'optimisation stochastique pour formaliser et développer de nouveaux algorithmes afin de relever ces défis. Cela sert non seulement à développer de nouvelles solutions, mais aussi à tester l'utilité des notions classiques d'optimisation dans l'apprentissage profond moderne.

Nous étudions trois problèmes fondamentaux. Premièrement, l'apprentissage collaboratif remplace une transmission unique de données brutes par des cycles répétés de communication de modèles partiellement entraînés. Cependant, ces communications se heurtent rapidement aux contraintes de bande passante lorsqu'il s'agit de grands modèles. Nous proposons de résoudre cette contrainte via la **compression des communications**. Deuxièmement, l'apprentissage collaboratif exploite directement la puissance de calcul des détenteurs de données. Avec seulement un sous-ensemble d'entre eux disponible à un moment donné, cela n'est pas aussi fiable que d'utiliser un centre de données. Ainsi, nous avons besoin de nouveaux algorithmes capables d'utiliser efficacement **le calcul local et peu fiable** des détenteurs de données. Enfin, l'apprentissage collaboratif permet à tout détenteur de données de participer au processus d'apprentissage, sans avoir connaissance des données ou des calculs locaux. Cela peut potentiellement exposer le système à des agents malveillants ou défectueux qui cherchent à faire échouer l'apprentissage. Nous développons des algorithmes à la **robustesse Byzantine** garantissant leur résistance à de tels attaquants.

Mots clés : Machine learning, optimisation, apprentissage collaboratif, apprentissage distribué, apprentissage fédéré, robustesse Byzantine.

Contents

Acknowledgements	i
Abstract (English/Français)	iii
1 Introduction	1
I Compressed communication	7
2 Fixing gradient compression using error feedback	9
2.1 Preface	9
2.2 Introduction	10
2.3 Significance and related work	11
2.4 Counterexamples for signSGD	13
2.5 Convergence with error feedback	15
2.5.1 Setup	15
2.5.2 Rate of convergence	16
2.5.3 Convergence of EF-SIGNSGD	18
2.6 Effect of SignSGD on generalization	19
2.6.1 Distance to gradient span	20
2.6.2 Simulations	21
2.7 Experiments	21
2.7.1 Experimental setup	22
2.7.2 Results	22
2.8 Conclusion	24
3 PowerSGD: Practical Gradient Compression	25
3.1 Preface	25
3.2 Introduction	26
3.3 Related work	27
3.4 Method	28
3.5 Theoretical Analysis of POWERSGD	30
3.5.1 Single/multi worker equivalence	31
3.5.2 Convergence of multi-worker error feedback with momentum	31
3.6 Ablation study of POWERSGD	32
	vii

Contents

3.6.1	Effect of error feedback	32
3.6.2	Effect of warm-start	33
3.6.3	Effect of varying the rank	34
3.7	Empirical evaluation of POWERSGD	34
3.7.1	Comparison with other compressors	35
3.7.2	Scalability of POWERSGD	36
3.7.3	Other tasks and methods	37
3.8	Conclusion	38
II	Unreliable local computation	39
4	SCAFFOLD: Stochastic Controlled Averaging for Federated Learning	41
4.1	Preface	41
4.2	Introduction	42
4.3	Setup	44
4.4	Convergence of FedAvg	45
4.4.1	Rate of convergence	45
4.4.2	Lower bounding the effect of heterogeneity	46
4.5	SCAFFOLD algorithm	47
4.6	Convergence of SCAFFOLD	49
4.7	Usefulness of local steps	51
4.8	Experiments	52
4.8.1	Setup	53
4.8.2	Simulated results	53
4.8.3	EMNIST results	53
4.9	Conclusion	55
5	Mime: mimicking centralized algorithms in cross-device federated learning	57
5.1	Preface	57
5.2	Introduction	58
5.3	Problem setup	60
5.4	How momentum can help reduce client drift	61
5.5	Mime framework	63
5.6	Theoretical analysis of Mime	65
5.6.1	Convergence with a generic base optimizer	65
5.6.2	Circumventing server-only lower bounds	67
5.7	Proof sketch	69
5.8	Experimental analysis on real world datasets	70
5.8.1	Setup	70
5.8.2	Ablation and comparative study	71
5.8.3	Large scale comparison with equal server and client communication	72
5.9	Conclusion	74

III Byzantine Robustness	75
6 Learning from History for Byzantine Robust Optimization	77
6.1 Preface	77
6.2 Introduction	78
6.3 Related work	79
6.4 Brittleness of existing aggregation rules	81
6.5 Necessity of using history	83
6.6 Robust aggregation	85
6.6.1 Anatomy of a robust aggregator	85
6.6.2 Robust aggregation via centered clipping	86
6.7 Robust optimization using momentum	88
6.7.1 Rate of convergence	88
6.7.2 Improved convergence using MVR	90
6.8 Experiments	90
6.8.1 Failure of “middle seekers”	91
6.8.2 Impact of momentum on robust aggregation rules	92
6.8.3 Stability of Centered Clip	92
6.9 Conclusion	92
7 Byzantine-Robust Learning on Heterogeneous Datasets via Resampling	95
7.1 Preface	95
7.2 Introduction	96
7.3 Related work	97
7.4 Attacks against existing aggregation schemes	99
7.4.1 Failure on imbalanced data without Byzantine workers	99
7.4.2 Mimic attack on balanced data	100
7.5 Constructing an agnostic robust aggregator using resampling	101
7.5.1 Resampling algorithm	102
7.5.2 Agnostic robust aggregation	102
7.6 Robust non-iid optimization using a robust aggregator	104
7.6.1 Algorithm description	105
7.6.2 Convergence rates	105
7.6.3 Lower bounds and the challenge of heterogeneity	106
7.6.4 Circumventing lower bounds using overparameterization	107
7.7 Experiments	108
7.7.1 Resampling against the attacks on non-iid data	108
7.7.2 Resampling against general Byzantine attacks	109
7.7.3 Resampling hyperparameter	110
7.7.4 Discussion	110
7.8 Conclusion	111

8 Conclusion	113
Appendices	117
9 Appendix for Error Feedback	119
9.1 Additional Experiments	119
9.1.1 Convergence under sparse noise	119
9.1.2 Description of models and datasets	120
9.1.3 Learning rate tuning	120
9.1.4 Experiments with Resnet	121
9.1.5 Experiments with VGG	121
9.1.6 Data generation process (Section 2.6.2)	123
9.2 Missing Proofs	123
9.2.1 Proof of counter-example (Theorem I)	124
9.2.2 Proof of bounded error (Lemma 3)	124
9.2.3 Proof of non-convex convergence of EF-SIGNSGD (Theorem II)	125
9.2.4 Proof of convex convergence of EF-SIGNSGD (Theorem III)	126
9.2.5 Proof relating linear span of gradients to pseudo-inverse (Lemma 9)	127
10 Appendix for PowerSGD	129
10.1 Discussions of convergence	129
10.1.1 Eigen compression	129
10.1.2 Subspace iteration	130
10.1.3 Proof that Orthogonalization is a linear operation (Remark 27)	131
10.1.4 Proof of Single/multi worker equivalence (Lemma 3.5.1)	131
10.1.5 Proof of convergence (Theorem V)	132
10.2 Cluster specifications	136
10.3 Convergence curves	137
10.4 Language Modeling with Transformers	139
10.5 The need for error feedback	141
10.6 Network parameters	142
10.7 Compressor implementation details	143
10.7.1 Random Block	143
10.7.2 Random K	143
10.7.3 Sign+Norm	144
10.7.4 Top K	144
10.7.5 Signum	145
10.7.6 Atomo	146
10.7.7 Best-approximation POWERSGD	146
10.8 Performance optimizations	147
10.9 Learning rate tuning	147
11 Appendix for SCAFFOLD	149

11.1 Related work and significance	149
11.2 Technicalities	150
11.2.1 Additional definitions	150
11.2.2 Some technical lemmas	151
11.3 Properties of convex functions	154
11.4 Convergence of FEDAVG	155
11.4.1 Bounding heterogeneity	156
11.4.2 Rates of convergence (Theorem VI)	157
11.4.3 Proof of convergence	158
11.4.4 Lower bound for FEDAVG (Theorem VII)	161
11.5 Convergence of SCAFFOLD	164
11.5.1 Convergence of SCAFFOLD FOR CONVEX FUNCTIONS (THEOREM VIII)	166
11.5.2 Convergence of SCAFFOLD FOR NON-CONVEX FUNCTIONS (THEOREM VIII)	173
11.6 Usefulness of local steps (Theorem IX)	179
11.6.1 Additional notation and assumptions	180
11.6.2 Lemmas tracking errors	181
11.6.3 Lemmas showing progress	183
12 Appendix for Mime	187
12.1 Related work and significance	187
12.2 Technicalities	188
12.2.1 Additional definitions	189
12.2.2 Some technical lemmas	189
12.3 Properties of convex functions	192
12.4 Convergence of FEDAVG	193
12.4.1 Bounding heterogeneity	194
12.4.2 Rates of convergence (Theorem VI)	195
12.4.3 Proof of convergence	196
12.4.4 Lower bound for FEDAVG (Theorem VII)	199
12.5 Convergence of SCAFFOLD	202
12.5.1 Convergence of SCAFFOLD FOR CONVEX FUNCTIONS (THEOREM VIII)	204
12.5.2 Convergence of SCAFFOLD FOR NON-CONVEX FUNCTIONS (THEOREM VIII)	211
12.6 Usefulness of local steps (Theorem IX)	217
12.6.1 Additional notation and assumptions	218
12.6.2 Lemmas tracking errors	219
12.6.3 Lemmas showing progress	221
13 Appendix for Byzantine robust learning using history	225
13.1 Convergence of momentum SGD	225
13.2 Proof of Theorem XV - Failure of permutation-invariant methods	228
13.3 Proof of Theorem XVI (Limits of robust aggregation)	229
13.4 Proof of Theorem XVII- Robustness of iterative clipping	229
13.4.1 Single iteration with good starting point	230

Contents

13.4.2 Robustness starting from arbitrary point	232
13.5 Proof of Theorem XIX - Byzantine-Robust Convergence	234
13.6 Proof of Theorem XX (Momentum based variance reduction)	237
13.7 Additional Experiments	241
13.7.1 Experiment setups	241
13.7.2 Exploring local steps between aggregations	243
13.7.3 Comparison with (Allen-Zhu et al., 2021)	243
14 Appendix for heterogeneous Byzantine robust learning using resampling	247
14.1 Experiment setup and additional experiments	248
14.1.1 Experiment setup	248
14.1.2 Additional experiments	250
14.2 Implementing the mimic attack	256
14.3 Constructing a robust aggregator using resampling	257
14.3.1 Supporting lemmas	257
14.3.2 Proofs of robustness	259
14.4 Lower bounds on non-iid data (Proof of Theorem XXIII)	262
14.5 Convergence of robust optimization on non-iid data (Theorems XXII and XXIV)	263
Bibliography	271
Curriculum Vitae	293

1 Introduction

Modern machine learning has enjoyed immense success stories such as using CNNs for skin cancer classification (Esteva et al., 2017), GPT-3 for natural language understanding (Brown et al., 2020), AlphaFold for predicting protein folding (Senior et al., 2020), etc. However, all of these achievements have been predicated on collecting large amounts of user data and training immense models in a centralized datacenter (aka centralized training). For example, the GPT-3 model has 175 billion parameters and is trained on 500 billion words crawled from the internet (Crawl, 2020). Further improvement in prediction performance seems to necessitate using even bigger models and training datasets (Kaplan et al., 2020). However, this raises serious questions about the data being utilized in such centralized training—who owns it? was it obtained with consent? how is it being stored? etc.

Collaborative learning presents an alternative training approach where multiple data holders collaborate with each other to train a machine learning model on their combined datasets, without ever communicating their raw data. This is a more democratic approach to machine learning with data holders retaining their ownership rights. The data holders could be edge devices such as mobile phones, organizations such as hospitals, or even companies within a consortium. They want to learn a joint objective on their sensitive datasets such as training a speech recognition model from voice recordings on mobile phones, or learning cancer markers from genomic data by hospitals, or identifying money laundering from bank transactions by a bank consortium. To encapsulate a wide variety of such use cases, we propose the following broad definition of collaborative learning:

Collaborative learning is a machine learning paradigm where multiple data holders collaborate in solving a machine learning problem. This can occur either under the coordination of a central server (in which case we refer to it as federated learning), or via direct peer to peer communication (called decentralized learning). The training utilizes the local compute of the data holders (called clients or workers) instead of a central datacenter. Further, their raw data is stored locally and not exchanged or transferred; instead, only focused updates intended for immediate aggregation are communicated.

We note three important characteristics of collaborative learning: i) the data never leaves the data holders—thus they retain full ownership throughout the process, ii) data can only be used in the training if the data holder actively chooses to participate in the process thus ensuring consent, and finally iii) the training objective is defined right at the start, ensuring that the data usage is tied to a specific use case. The latter is especially important since meaningful consent of data usage needs to be tied to a concrete use-case. This principle also underlies the core ‘data minimization’ requirement in GDPR legislation (ICO, 2021).

A real world analogy for the collaborative training setting vs. the centralized training setting is as follows¹. Suppose my land lord decides to install a video camera in my living room. They

¹This example is inspired by a talk of Blaise Agüera y Arcas.

tell me the video feed is being fed into a machine learning model in the cloud which detects movement and turns off the lights when the room is not being used. Contrast this setup with a simple motion detector which can also achieve the same goal of detecting motion and turning off/on the lights. With the former approach, I have no idea where the video recording is being stored, how long it is stored for, how securely it is being stored, or for what other purposes the recordings are being used for. In the latter setup, I can be guaranteed that no data leaves my house and further no data other than what is absolutely required for the task at hand is being collected. Consent to smart lights can not be construed to imply consenting to indefinite and undisclosed storage of recordings. This is similar to the collaborative training setting—consent of participation is tied to a specific machine learning objective, with no movement of raw data.

While collaborative learning is undoubtedly a better paradigm than the centralized approach, it comes with a host of challenges. In this thesis, we examine three fundamental roadblocks which need to be overcome for wide scale adoption: i) compressing the inter-client communication during training so that model sizes are not limited by the bandwidth, ii) training algorithms which can efficiently leverage the often unreliable local computation available on the clients, and iii) designing robust systems which can withstand potentially malicious attackers or simply buggy clients who may seek to derail convergence.

Compressed communication. In collaborative learning, each client iteratively partially trains a model using its local data, which is then communicated to a server or directly to other clients. However, as we noted, modern machine learning models can be immense with their growth in size outpacing even Moore’s law (AI, 2018). In such cases, communication (more specifically bandwidth) becomes the bottleneck. A simple way to alleviate this would be to use lossy compression and only send approximations of the update. E.g. we can send only the signs of the parameter updates (called SignSGD), or only the k largest parameters by magnitude (called Top- k). However, we will see that a naive application of such methods is problematic. We show counter-examples (both theoretical and real world) where they fail to converge, and further even if they converge they may not generalize. We then demonstrate (theoretically and empirically) that using error-feedback, i.e. incorporating the error made by the compression operator into the next step, overcomes both these issues.

Then, we incorporate real world system constraints into the error-feedback framework to come up with a practical highly performant algorithm. This is achieved by designing a novel low rank compressor which uses power iterations and combining it with modern optimizers such as momentum and Adam. Crucially, our compressor involves only matrix multiplications (hence can leverage GPUs for fast encoding/decoding), and is also linear (hence is compatible with all-reduce). We show that our method works ‘out of the box’ without requiring additional hyper-parameter tuning, and is ready for adoption in practice.

Unreliable local computation. A second core feature of collaborative learning is that most of the training occurs directly on the clients themselves using local computational resources.

However, only a few clients will likely be available at any given time. Factors such as time zones would likely dictate when clients are available—e.g. mobile phones will only be available when they are charging and not being used (typically in the night), and hospitals may only participate during working hours when the system can be supervised. Bad network (data) connections may further cause unforeseen dropouts. Thus, the key challenge we face is to design algorithms which can efficiently leverage the local compute power of the (some-time) available clients.

We show that FedAvg (the de facto standard method for federated learning) suffers from ‘client-drift’ when the clients are heterogeneous (non-iid), resulting in unstable and slow convergence. Instead, we propose a new algorithm (SCAFFOLD) which uses control variates (variance reduction) to correct for the ‘client-drift’ in its local updates. We show (theoretically and empirically) that SCAFFOLD is unaffected by heterogeneity or unavailability of clients. We further propose a more practical algorithmic framework (Mime). Mime i) mitigates client drift similar to SCAFFOLD, but also ii) adapts an arbitrary centralized optimization algorithm such as momentum and Adam to FL in a principled manner. We theoretically prove that Mime is provably *faster than any centralized method*—the first such result, and also perform a thorough experimental evaluation.

Byzantine robustness. The third facet of collaborative learning is that no one can inspect any data holder’s raw data or even local processing. This may open up the system to potentially malicious (or simply faulty) participants who may derail the training. Byzantine robustness seeks to develop algorithms which are resilient to such attackers. Such robustness also ensures that no single data holder has a disproportionately large control on the output of the collaborative training.

First, we examine existing aggregation rules and show realistic examples where they fail to converge even in the *absence of any Byzantine attackers*. This, we show, is because traditional definitions of robustness do not suffice for our setting. We introduce a more fine-grained definition of robust aggregator and give a new *iterative clipping* procedure which satisfies it. Our procedure is efficient, and compatible with secure aggregation and all-reduce.

Secondly, we show that even if the aggregation rules may succeed in limiting the influence of the attackers in a single round, the attackers can couple their attacks across time eventually leading to divergence of any memory-less systems. This implies that in order to ensure Byzantine robustness, it is necessary to profile workers using past updates. We then show that simply incorporation *local momentum* is sufficient to overcome such time-coupled attacks. This is the first provably robust method for the standard stochastic optimization setting.

Finally, we study the effect of data heterogeneity. To handle such settings, we propose a simple resampling scheme that adapts existing robust algorithms to heterogeneous datasets at a negligible computational cost. We demonstrate (theoretically and experimentally) that combining resampling with existing robust algorithms is effective against challenging attacks. Our work also shows that having over-parameterized models, when combined with robust

aggregation rules, is very beneficial for heterogeneous Byzantine robust optimization.

Organization. The thesis consists of three parts, each dealing with one of the problem described above. Each part then consists of two chapters, the first of which presents a largely theoretical investigation into the problem and proposing solutions to a possibly simplified version. The second chapter then takes a more practical look, trying to incorporate real world system constraints and characteristics yielding algorithms which are actually useful in real world collaborative learning. Each of these chapter maps to a paper written by the author (with other collaborators). Chapters also start with a preface, consisting of a summary of the work and list of author contributions using the CRediT framework ([Brand et al., 2015](#)). The appendices (which contain any missing proofs, additional experiments, etc.) for all the chapter are collected at the end in the fourth part.

Compressed communication **Part I**

2 Fixing gradient compression using error feedback

2.1 Preface

Contribution and sources. This chapter is largely based on (Karimireddy et al., 2019), with improved theoretical rates using techniques from (Stich and Karimireddy, 2020). Ideation, theory, experiment design, and most of the writing was done by the author. The detailed individual contributions are listed below using the CRediT framework (Brand et al., 2015):

SPK (author): Conceptualization, Methodology, Formal analysis, Software (20%), Writing – original draft preparation (90%)

Quentin Rebjock: Software (80%), Writing – original draft preparation (10%)

Sebastian Stich: Writing – review and editing

Martin Jaggi: Writing – review and editing, Administration, Supervision.

Summary. As machine learning models and datasets grow bigger outpacing Moore’s law, distributed training using multiple workers (GPUs / TPUs) is fast becoming a necessity. However, this leads to communication between the different workers (instead of computation or storage) becoming the bottleneck. Gradient compression methods are popular strategies to overcome such communication bottlenecks.

In this chapter, we examine these techniques through the lens of stochastic optimization and show that a naive application of communication compression (e.g. SIGNSGD or Top-k) may not work. We show counter-examples (both theoretical and real world) where they fail to converge, and further even if they converge they may not generalize. We then demonstrate (theoretically and empirically) that using error-feedback, i.e. incorporating the error made by the compression operator into the next step, overcomes both these issues.

Since our initial work, numerous follow up papers (cf. Xu et al. (2020) for a survey) have repeatedly corroborated our conclusions—using error-feedback with biased compressors retains the train and test accuracy while using only a fraction of the communication.

2.2 Introduction

Stochastic optimization algorithms (Bottou, 2010) which are amenable to large-scale parallelization, taking advantage of massive computational resources (Krizhevsky et al., 2012; Dean et al., 2012) have been at the core of significant recent progress in deep learning (Schmidhuber, 2015; LeCun et al., 2015). One such example is the SIGNSGD algorithm and its variants, c.f. (Seide et al., 2014; Bernstein et al., 2018, 2019).

To minimize a continuous (possibly) non-convex function $f: \mathbb{R}^d \rightarrow \mathbb{R}$, the classic stochastic gradient algorithm (SGD) (Robbins and Monro, 1951) performs iterations of the form

$$\mathbf{x}_{t+1} := \mathbf{x}_t - \gamma \mathbf{g}_t, \quad (\text{SGD})$$

where $\gamma \in \mathbb{R}$ is the step-size (or learning-rate) and \mathbf{g}_t is the stochastic gradient such that $\mathbb{E}[\mathbf{g}_t] = \nabla f(\mathbf{x}_t)$.

Methods performing updates only based on the *sign* of each coordinate of the gradient have recently gaining popularity for training deep learning models (Seide et al., 2014; Carlson et al., 2015; Wen et al., 2017; Balles and Hennig, 2018; Bernstein et al., 2018; Zaheer et al., 2018; Liu et al., 2018). For example, the update step of SIGNSGD is given by:

$$\mathbf{x}_{t+1} := \mathbf{x}_t - \gamma \text{sign}(\mathbf{g}_t). \quad (\text{SIGNSGD})$$

Such sign-based algorithms are particularly interesting since they can be viewed through two lenses: as i) approximations of adaptive gradient methods such as ADAM (Balles and Hennig, 2018), and also a ii) communication efficient gradient compression scheme (Seide et al., 2014). However, we show that a severe handicap of sign-based algorithms is that they *do not* converge in general. To substantiate this claim, we present in this work simple convex counter-examples where SIGNSGD cannot converge. The main reasons being that the sign operator loses information about, i.e. ‘forgets’, i) the magnitude, as well as ii) the direction of \mathbf{g}_t . We present an elegant solution that provably fixes these problems of SIGNSGD, namely algorithms with *error-feedback*.

Error-feedback. We demonstrate that the aforementioned problems of SIGNSGD can be fixed by i) scaling the signed vector by the norm of the gradient to ensure the magnitude of the gradient is not forgotten, and ii) locally storing the difference between the actual and compressed gradient, and iii) adding it back into the next step so that the correct direction is not forgotten. We call our ‘fixed’ method EF-SIGNSGD (Algorithm 1).

In Algorithm 1, \mathbf{e}_t denotes the accumulated error from all quantization/compression steps. This residual error is added to the gradient step $\gamma \mathbf{g}_t$ to obtain the corrected direction \mathbf{p}_t . When compressing \mathbf{p}_t , the signed vector is again scaled by $\|\mathbf{p}_t\|_1$ and hence does not lose information about the magnitude. Note that our algorithm does not introduce any additional parameters and requires only the step-size γ .

Algorithm 1 EF-SIGNSGD (SIGNSGD with Error-Feedb.)

```

1: Input: learning rate  $\gamma$ , initial iterate  $\mathbf{x}_0 \in \mathbb{R}^d$ ,  $\mathbf{e}_0 = \mathbf{0}$ 
2: for  $t = 0, \dots, T - 1$  do
3:    $\mathbf{g}_t := \text{stochasticGradient}(\mathbf{x}_t)$ 
4:    $\mathbf{p}_t := \gamma \mathbf{g}_t + \mathbf{e}_t$   $\triangleright$  error correction
5:    $\Delta_t := (\|\mathbf{p}_t\|_1 / d) \text{sign}(\mathbf{p}_t)$   $\triangleright$  compression
6:    $\mathbf{x}_{t+1} := \mathbf{x}_t - \Delta_t$   $\triangleright$  update iterate
7:    $\mathbf{e}_{t+1} := \mathbf{p}_t - \Delta_t$   $\triangleright$  update residual error
8: end for

```

Our contributions. We show that naively using biased gradient compression schemes (such as e.g. SIGNSGD) can lead to algorithms which may not generalize or even converge. We show both theoretically and experimentally that simply adding error-feedback solves such problems and recovers the performance of full SGD, thereby saving on communication costs. We state our results for SIGNSGD to ease our exposition but our positive results are valid for general compression schemes, and our counterexamples extend to SIGNSGD with momentum, multiple worker settings, and even other biased compression schemes. More specifically our contributions are:

1. We construct a simple convex non-smooth counterexample where SIGNSGD cannot converge, even with the full batch (sub)-gradient and tuning the step-size. Another counterexample for a wide class of smooth convex functions proves that SIGNSGD with stochastic gradients cannot converge with batch-size one.
2. We prove that by incorporating error-feedback, SIGNSGD—as well as any other gradient compression schemes—always converge. Further, our theoretical analysis for non-convex smooth functions recovers the same rate as SGD, i.e. we get compression *for free*.
3. We show that our algorithm EF-SIGNSGD which incorporates error-feedback approaches the linear span of the past gradients. Therefore, unlike SIGNSGD, EF-SIGNSGD converges to the max-margin solution in over-parameterized least-squares. This provides a theoretical justification for why EF-SIGNSGD can be expected to have better generalization.
4. We show extensive experiments on CIFAR10 and CIFAR100 using Resnet and VGG architectures demonstrating that EF-SIGNSGD indeed significantly outperforms SIGNSGD, and matches SGD both on test as well as train datasets while reducing communication by a factor of $\sim 30\times$.

2.3 Significance and related work

Relation to adaptive methods. Introduced in (Kingma and Ba, 2014), ADAM has gained immense popularity as the algorithm of choice for adaptive stochastic optimization for its perceived lack of need for parameter-tuning. However since, the convergence (Reddi et al.,

(2018) as well the generalization performance (Wilson et al., 2017) of such adaptive algorithms has been called into question. Understanding when ADAM performs poorly and providing a principled ‘fix’ for these cases is crucial given its importance as the algorithm of choice for many researchers. It was recently noted by Balles and Hennig (2018) that the behavior of ADAM is in fact identical to that of SIGNSGD with *momentum*. More formally, the SIGNSGDM algorithm (referred to as ‘signum’ by Bernstein et al. (2018, 2019)) adds momentum to the SIGNSGD update as:

$$\begin{aligned} \mathbf{m}_{t+1} &:= \mathbf{g}_t + \beta \mathbf{m}_t \\ \mathbf{x}_{t+1} &:= \mathbf{x}_t - \gamma \text{sign}(\mathbf{m}_{t+1}). \end{aligned} \tag{SIGNSGDM}$$

for parameter $\beta > 0$. This connection between signed methods and fast stochastic algorithms is not surprising since sign-based gradient methods were first studied as a way to speed up SGD (Riedmiller and Braun, 1993). Given their similarity, understanding the behavior of SIGNSGD and SIGNSGDM can help shed light on the convergence of ADAM.

Relation to gradient compression methods. As the size of the models keeps getting bigger, the training process can often take days or even weeks (Dean et al., 2012). This process can be significantly accelerated by massive parallelization (Li et al., 2014; Goyal et al., 2017). However, at these scales communication of the gradients between the machines becomes a bottleneck hindering us from making full use of the impressive computational resources available in today’s data centers (Chilimbi et al., 2014; Seide et al., 2014; Strom, 2015). A simple solution to alleviate this bottleneck is to compress the gradient and reduce the number of bits transmitted. While the analyses of such methods have largely been restricted to *unbiased* compression schemes (Alistarh et al., 2017; Wen et al., 2017; Wang et al., 2018), biased schemes which perform extreme compression practically perform much better (Seide et al., 2014; Strom, 2015; Lin et al., 2018)—often *without any loss* in convergence or accuracy. Of these, (Seide et al., 2014; Strom, 2015; Wen et al., 2017) are all sign-based compression schemes. Interestingly, all the practical works (Seide et al., 2014; Strom, 2015; Lin et al., 2018) use some form of error-feedback.

Error-feedback. The idea of error-feedback was, as far as we are aware, first introduced in 1-bit SGD (Seide et al., 2014; Strom, 2015). The algorithm 1-bit SGD is very similar to our EF-SIGNSGD algorithm, but tailored for the specific recurrent network studied there. (Wu et al., 2018) analyze unbiased compressors with a form of error-feedback involving two additional hyper-parameters and restricted to quadratic functions. Though not presented as such, the ‘momentum correction’ used in (Lin et al., 2018) is a variant of error-feedback. However the error-feedback is not on the vanilla SGD algorithm, but on SGD with momentum. As far as we are aware, Karimireddy et al. (2018a); Lu et al. (2020) were the first to provide an analysis of the error feedback mechanism. They use it to design inexact accelerated methods. Stich et al. (2018) utilized similar techniques to analyze error feedback with compressed stochastic gradients (they call it ‘memory’) in the strongly convex case. Our convergence results can be seen as a further extension to the non-convex, and non-smooth convex cases.

Generalization of deep learning methods. Deep networks are almost always over-parameterized and are known to be able to fit arbitrary data and always achieve zero training error (Zhang et al., 2017). This ability of deep networks to generalize well on real data, while simultaneously being able to fit arbitrary data has recently received a lot of attention (e.g. Soudry et al. (2018); Dinh et al. (2017); Zhang et al. (2018); Arpit et al. (2017); Kawaguchi et al. (2017)). SIGNSGD and ADAM are empirically known to generalize worse than SGD (Wilson et al., 2017; Balles and Hennig, 2018). A number of recent papers try close this gap for ADAM. Luo et al. (2019) show that by bounding the adaptive step-sizes in ADAM leads to closing the generalization gap. They require new hyper-parameters on top of ADAM to adaptively tune these bounds on the step-sizes. Chen and Gu (2019) interpolate between SGD and ADAM using a new hyper-parameter p and show that tuning this can recover performance of SGD. Zaheer et al. (2018) introduce a new adaptive algorithm which is closer to Adagrad (Duchi et al., 2011). Similarly, well-tuned ADAM (where all the hyper-parameters and not just the learning rate are tuned) is also known to close the generalization gap (Gugger and Howard, 2018). In all of these algorithms, new hyper-parameters are introduced which essentially control the effect of the adaptivity. Thus they require additional tuning while the improvement upon traditional SGD is questionable. We are not aware of other work bridging the generalization gap in sign-based methods.

2.4 Counterexamples for signSGD

In this section we study the limitations of SIGNSGD. Under benign conditions—for example if i) the function f is smooth, and ii) the stochastic noise is gaussian or an extremely large batch-size is used (equal to the total number of iterations)—the algorithm can be shown to converge (Bernstein et al., 2018, 2019). However, we show that SIGNSGD does not converge under more standard assumptions. We demonstrate this first on a few pedagogic examples and later also for realistic and general sum-structured loss functions.

If we use a fixed step-size $\gamma \geq 0$, SIGNSGD does not converge even for simple one-dimensional linear functions. **Counterexample 1.** For $x \in \mathbb{R}$ consider the constrained problem

$$\min_{x \in [-1, 1]} [f(x) := \frac{1}{4}x],$$

with minimum at $x^* = -1$. Assume stochastic gradients are given as (note that $f(x) = \frac{1}{4}(4x - x - x - x)$)

$$g = \begin{cases} 4, & \text{with prob. } \frac{1}{4} \\ -1, & \text{with prob. } \frac{3}{4} \end{cases} \quad \text{with} \quad \mathbb{E}[g] = \nabla f(x).$$

For SGD with any step-size γ ,

$$\mathbb{E}_t[f(x_{t+1})] = \frac{1}{4}(x_t - \gamma \mathbb{E}[g]) = f(x_t) - \frac{\gamma}{16}.$$

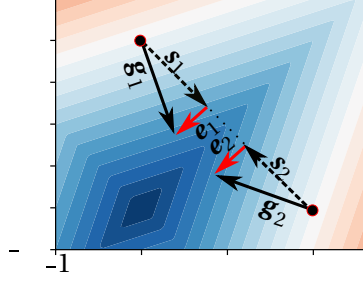


Figure 2.1 – The gradients \mathbf{g} (in solid black), signed gradient direction $\mathbf{s} = \text{sign}(\mathbf{g})$ (in dashed black), and the error \mathbf{e} (in red) are plotted for $\epsilon = 0.5$. SIGNSGD moves only along $\mathbf{s} = \pm(1, -1)$ while the error \mathbf{e} is ignored.

On the other hand, for SIGNSGD with any fixed γ ,

$$\mathbb{E}_t[f(x_{t+1})] = \frac{1}{4}(x_t - \gamma \mathbb{E}[\text{sign}(g)]) = f(x_t) + \frac{\gamma}{8},$$

i.e. the objective function increases in expectation for $\gamma \geq 0$.

Remark 1. In the above example, we exploit that the sign operator loses track of the magnitude of the stochastic gradient. Also note that our noise is bimodal. The counter-examples for the convergence of ADAM (Reddi et al., 2018; Luo et al., 2019) also use similar ideas. Such examples were previously noted for SIGNSGD by (Bernstein et al., 2019).

In the example above the step-size γ was fixed. However increasing batch-size or tuning the step-size may still allow convergence. Next we show that even with adaptive step-sizes (e.g. decreasing, or adaptively chosen optimal step-sizes) SIGNSGD does not converge. This even holds if the full (sub)-gradient is available (non-stochastic case).

Counterexample 2. For $\mathbf{x} \in \mathbb{R}^2$ consider the following non-smooth convex problem with $\mathbf{x}^* = (0, 0)^\top$:

$$\min_{\mathbf{x} \in \mathbb{R}^2} \left[f(\mathbf{x}) := \epsilon |x_1 + x_2| + |x_1 - x_2| \right],$$

for parameter $0 < \epsilon < 1$ and subgradient

$$\mathbf{g}(\mathbf{x}) = \text{sign}(x_1 + x_2) \cdot \epsilon \begin{pmatrix} 1 \\ 1 \end{pmatrix} + \text{sign}(x_1 - x_2) \begin{pmatrix} 1 \\ -1 \end{pmatrix}.$$

See Fig. 2.1. The iterates of SIGNSGD started at $\mathbf{x}_0 = (1, 1)^\top$ lie along the line $x_1 + x_2 = 2$. Note that for any \mathbf{x} s.t. $x_1 + x_2 > 0$, $\text{sign}(\mathbf{g}(\mathbf{x})) = \pm(1, -1)^\top$, and hence $x_1 + x_2$ remains constant among the iterations of SIGNSGD. Consequently, for any step-size sequence γ_t , $f(\mathbf{x}_t) \geq f(\mathbf{x}_0)$.

Remark 2. In this example, we exploit the fact that the sign operator is a biased approximation of the gradient—it consistently ignores the direction $\mathbf{e} = \epsilon(1, 1)^\top$ (see Fig 2.1). Tuning the step-size would not help either.

Algorithm 2 EF-SGD (Compr. SGD with Error-Feedback)

```

1: Input: learning rate  $\gamma$ , compressor  $\mathcal{C}(\cdot)$ ,  $\mathbf{x}_0 \in \mathbb{R}^d$ 
2: Initialize:  $\mathbf{e}_0 = \mathbf{0} \in \mathbb{R}^d$ 
3: for  $t = 0, \dots, T-1$  do
4:    $\mathbf{g}_t := \text{stochasticGradient}(\mathbf{x}_t)$ 
5:    $\mathbf{p}_t := \gamma \mathbf{g}_t + \mathbf{e}_t$  ▷ error correction
6:    $\Delta_t := \mathcal{C}(\mathbf{p}_t)$  ▷ compression
7:    $\mathbf{x}_{t+1} := \mathbf{x}_t - \Delta_t$  ▷ update iterate
8:    $\mathbf{e}_{t+1} := \mathbf{p}_t - \Delta_t$  ▷ update residual error
9: end for
    
```

One might wonder if the smooth-case is easier. Unfortunately, the previous example can easily be extended to show that SIGNSGD with stochastic gradients may not converge even for smooth functions.

Counterexample 3. For $\mathbf{x} \in \mathbb{R}^2$ consider the following least-squares problem with $\mathbf{x}^* = (0, 0)^\top$:

$$\min_{\mathbf{x} \in \mathbb{R}^2} [f(\mathbf{x}) := (\langle \mathbf{a}_1, \mathbf{x} \rangle)^2 + (\langle \mathbf{a}_2, \mathbf{x} \rangle)^2], \text{ where}$$

$$\mathbf{a}_{1,2} := \pm(1, -1) + \epsilon(1, 1),$$

for parameter $0 < \epsilon < 1$ and stochastic gradient $\mathbf{g}(\mathbf{x}) = \nabla_{\mathbf{x}}(\langle \mathbf{a}_1, \mathbf{x} \rangle)^2$ with prob. $\frac{1}{2}$ and $\mathbf{g}(\mathbf{x}) = \nabla_{\mathbf{x}}(\langle \mathbf{a}_2, \mathbf{x} \rangle)^2$ with prob. $\frac{1}{2}$. The stochastic gradient is then either $e\mathbf{a}_1$ or $e\mathbf{a}_2$ for some scalar e . Exactly as in the non-smooth case, for $\mathbf{x}_0 = (1, 1)^\top$, the sign of the gradient $\text{sign}(\mathbf{g}) = \pm(1, -1)$. Hence SIGNSGD with any step-size sequence remains stuck along the line $x_1 + x_2 = 2$ and $f(\mathbf{x}_t) \geq f(\mathbf{x}_0)$ a.s.

We can generalize the above counter-example to arbitrary dimensions and loss functions.

Theorem I. Suppose that scalar loss functions $\{l_i: \mathbb{R} \rightarrow \mathbb{R}\}_{i=1}^n$ and data-points $\{\mathbf{a}_i\}_{i=1}^n \in \mathbb{R}^d$ for $d \geq 2$ satisfy: i) $f(\mathbf{x}) := \sum_{i=1}^n l_i(\langle \mathbf{a}_i, \mathbf{x} \rangle)$ has a unique optimum at \mathbf{x}^* , and ii) there exists $\mathbf{s} \in \{-1, 1\}^d$ such that $\text{sign}(\mathbf{a}_i) = \pm \mathbf{s}$ for all i . Then SIGNSGD with batch-size 1 and stochastic gradients $\mathbf{g}(\mathbf{x}) = \nabla_{\mathbf{x}} l_i(\langle \mathbf{a}_i, \mathbf{x} \rangle)$ for i chosen uniformly at random does not converge to \mathbf{x}^* a.s. for any adaptive sequence of step-sizes, even with random initialization.

2.5 Convergence with error feedback

We show the rather surprising result that incorporating error-feedback is sufficient to ensure that the algorithm converges at a rate which matches that of SGD. In this section we consider a general gradient compression scheme.

2.5.1 Setup

We generalize the notion of a compressor from (Stich et al., 2018).

Assumption A (Compressor). *An operator $\mathcal{C} : \mathbb{R}^d \rightarrow \mathbb{R}^d$ is a δ -approximate compressor over \mathcal{Q} for $\delta \in (0, 1]$ if*

$$\|\mathcal{C}(\mathbf{x}) - \mathbf{x}\|_2^2 \leq (1 - \delta)\|\mathbf{x}\|_2^2, \quad \forall \mathbf{x} \in \mathcal{Q}.$$

Note that $\delta = 1$ implies that $\mathcal{C}(\mathbf{x}) = \mathbf{x}$. Examples of compressors include: i) the sign operator, ii) top- k which selects k coordinates with the largest absolute value while zero-ing out the rest (Lin et al., 2018; Stich et al., 2018), iii) k -PCA which approximates a matrix \mathbf{X} with its top k eigenvectors (Wang et al., 2018). Randomized compressors satisfying the assumption in expectation are also allowed.

We employ standard assumptions of smoothness of the loss function and the variance of the stochastic gradient.

Assumption B (Smoothness). *A function $f : \mathbb{R}^d \rightarrow \mathbb{R}$ is L -smooth if for all $\mathbf{x}, \mathbf{y} \in \mathbb{R}^d$ the following holds:*

$$|f(\mathbf{y}) - (f(\mathbf{x}) + \langle \nabla f(\mathbf{x}), \mathbf{y} - \mathbf{x} \rangle)| \leq \frac{L}{2} \|\mathbf{y} - \mathbf{x}\|_2^2.$$

Assumption C (Variance bound). *For any \mathbf{x} , our query for a stochastic gradient returns \mathbf{g} such that*

$$\mathbb{E}[\mathbf{g}] = \nabla f(\mathbf{x}), \quad \mathbb{E}\|\mathbf{g} - \nabla f(\mathbf{x})\|_2^2 \leq \sigma^2, \quad \text{and} \quad \|\nabla f(\mathbf{x})\|_2^2 \leq M^2.$$

We now state a key lemma that shows that the residual errors maintained in Algorithm 2 do not accumulate too much.

Lemma 3 (Error is bounded). *Given that Assumptions A and C hold for all t . Then at any iteration t of EF-SGD, the norm of the error \mathbf{e}_t in Algorithm 2 is bounded:*

$$\mathbb{E}\|\mathbf{e}_t\|_2^2 \leq \frac{2(1-\delta)}{\delta^2} \gamma^2 (2M^2 + \delta\sigma^2), \quad \forall t \geq 0.$$

If $\delta = 1$, then $\|\mathbf{e}_t\| = 0$ and the error is zero as expected.

2.5.2 Rate of convergence

Given these assumptions, we can formally state our theorem followed by a sketch of the proof.

Theorem II (Non-convex convergence of EF-SGD). *Let $\{\mathbf{x}_t\}_{t \geq 0}$ denote the iterates of Algorithm 2 for any step-size $\gamma \in [0, 4\delta / L(\delta + 4(\sqrt{1-\delta}))]$. Under Assumptions A, B, and C,*

$$\min_{t \in [T]} \mathbb{E}[\|\nabla f(\mathbf{x}_t)\|_2^2] \leq \frac{2f_0}{\gamma(T+1)} + 6\gamma L\sigma^2,$$

with $f_0 := f(\mathbf{x}_0) - f^*$.

Proof Sketch. Intuitively, the condition that $\mathcal{C}(\cdot)$ is a δ -approximate compressor implies that at each iteration a δ -fraction of the gradient information is sent. The rest is added to \mathbf{e}_t to be

transmitted later. Eventually, all the gradient information is transmitted—albeit with a delay which depends on δ . Thus EF-SGD can intuitively be viewed as a delayed gradient method. If the function is smooth, the gradient does not change quickly and so the delay does not significantly matter.

More formally, consider the error-corrected sequence $\tilde{\mathbf{x}}_t$ which represents \mathbf{x}_t with the ‘delayed’ information added:

$$\tilde{\mathbf{x}}_t := \mathbf{x}_t - \mathbf{e}_t.$$

It satisfies the recurrence

$$\tilde{\mathbf{x}}_{t+1} = \mathbf{x}_t - \mathbf{e}_{t+1} - \mathcal{C}(\mathbf{p}_t) = \mathbf{x}_t - \mathbf{p}_t = \tilde{\mathbf{x}}_t - \gamma \mathbf{g}_t.$$

If \mathbf{x}_t was exactly equal to $\tilde{\mathbf{x}}_t$ (i.e. there was zero ‘delay’), then we could proceed with the standard proof of SGD. We instead rely on Lemma 3 which shows $\tilde{\mathbf{x}}_t \approx \mathbf{x}_t$ and on the smoothness of f which shows $\nabla f(\mathbf{x}_t) \approx \nabla f(\tilde{\mathbf{x}}_t)$. \square

Remark 4. If we substitute $\gamma = \min\left(\frac{4\delta}{L(\delta+4(\sqrt{1-\delta}))}, \sqrt{\frac{f_0}{L\sigma^2(T+1)}}\right)$ in Theorem II, we get

$$\min_{t \in [T]} \mathbb{E}[\|\nabla f(\mathbf{x}_t)\|^2] \leq \mathcal{O}\left(\sqrt{\frac{Lf_0\sigma^2}{T+1}} + \frac{Lf_0}{\delta(T+1)}\right)$$

In the above rate, the compression factor δ only appears in the higher order $\mathcal{O}(1/T)$ term. For comparison, SGD under the exact same assumptions achieves

$$\min_{t \in [T]} \mathbb{E}[\|\nabla f(\mathbf{x}_t)\|^2] \leq \mathcal{O}\left(\sqrt{\frac{Lf_0\sigma^2}{T+1}} + \frac{Lf_0}{(T+1)}\right).$$

This means that after $T \geq \mathcal{O}(1/\delta)$ iterations the second term becomes negligible and the rate of convergence catches up with full SGD—this is usually true after just the first few epochs. Thus we prove that compressing the gradient does not change the asymptotic rate of SGD.

Remark 5. The use of error-feedback was motivated by our counter-examples for biased compression schemes. However our rates show that even if using an unbiased compression (e.g. QSGD (Alistarh et al., 2017)), using error-feedback gives significantly better rates. Suppose we are given an unbiased compressor $cU(\cdot)$ such that $\mathbb{E}[cU(\mathbf{x})] = \mathbf{x}$ and $\mathbb{E}[\|cU(\mathbf{x})\|_2^2] \leq k\|\mathbf{x}\|^2$. Then without feedback, using standard analysis (e.g. (Alistarh et al., 2017)) the algorithm converges k times slower:

$$\min_{t \in [T]} \mathbb{E}[\|\nabla f(\mathbf{x}_t)\|^2] \leq \mathcal{O}\left(\sqrt{\frac{kLf_0\sigma^2}{T+1}} + \frac{Lf_0}{(T+1)}\right).$$

Instead, if we use $\mathcal{C}(\mathbf{x}) = \frac{1}{k}cU(\mathbf{x})$ with error-feedback, we would achieve

$$\min_{t \in [T]} \mathbb{E}[\|\nabla f(\mathbf{x}_t)\|^2] \leq \mathcal{O}\left(\sqrt{\frac{Lf_0\sigma^2}{T+1}} + \frac{kLf_0}{(T+1)}\right),$$

thereby pushing the dependence on k into the higher order $\mathcal{O}(1/T)$ term.

Our counter-examples showed that biased compressors may not converge for non-smooth functions. Below we prove that adding error-feedback ensures convergence under standard assumptions even for non-smooth functions.

Theorem III (Non-smooth convergence of EF-SGD). *Let $\{\mathbf{x}_t\}_{t \geq 0}$ denote the iterates of Algorithm 2 for any step-size $\gamma > 0$ and define $\bar{\mathbf{x}}_t = \frac{1}{T} \sum_{t=0}^T \mathbf{x}_t$. Given that f is convex and Assumptions A and C hold,*

$$\mathbb{E}[f(\bar{\mathbf{x}}_t)] - f^* \leq \frac{\|\mathbf{x}_0 - \mathbf{x}^*\|^2}{2\gamma(T+1)} + \gamma(M^2 + \sigma^2) \left(\frac{1}{2} + \frac{2\sqrt{1-\delta}}{\delta} \right).$$

Remark 6. By picking the optimal $\gamma = \mathcal{O}(1/\sqrt{T})$, we see that

$$\mathbb{E}[f(\bar{\mathbf{x}}_t)] - f^* \leq \frac{\|\mathbf{x}_0 - \mathbf{x}^*\| \sqrt{M^2 + \sigma^2}}{2\sqrt{T+1}} \sqrt{1 + \frac{4\sqrt{1-\delta}}{\delta}}.$$

For comparison, the rate of convergence under the same assumptions for SGD is

$$\mathbb{E}[f(\bar{\mathbf{x}}_t)] - f^* \leq \frac{\|\mathbf{x}_0 - \mathbf{x}^*\| \sqrt{M^2 + \sigma^2}}{2\sqrt{T+1}}.$$

For non-smooth functions, unlike in the smooth case, the compression quality δ appears directly in the leading term of the convergence rate. This is to be expected since we can no longer assume that $\nabla f(\bar{\mathbf{x}}_t) \approx \nabla f(\mathbf{x}_t)$, which formed the crux of our argument for the smooth case.

Remark 7. Consider the top-1 compressor which just picks the coordinate with the largest absolute value, and zeroes out everything else. It is obvious that top-1 is a $\frac{1}{d}$ -approximate compressor (cf. (Stich et al., 2018, Lemma A.1)). Running EF-SGD with \mathcal{C} as top-1 results in a greedy coordinate algorithm. This is the first result we are aware which shows the convergence of a greedy-coordinate type algorithm on non-smooth functions.

2.5.3 Convergence of EF-SIGNSGD

What do our proven rates imply for EF-SIGNSGD (Algorithm 1), the method of our interest here?

Lemma 8 (Compressed sign). *The operator $\mathcal{C}(\mathbf{v}) := \frac{\|\mathbf{v}\|_1}{d} \text{sign}(\mathbf{v})$ is a*

$$\phi(\mathbf{v}) = \frac{\|\mathbf{v}\|_1^2}{d\|\mathbf{v}\|_2^2}$$

compressor.

We refer to the quantity $\phi(\mathbf{v})$ as the density of \mathbf{v} . If the vector \mathbf{v} had only one non-zero element, the value of δ for EF-SIGNSGD could be as bad as $1/d$. However, in deep learning the

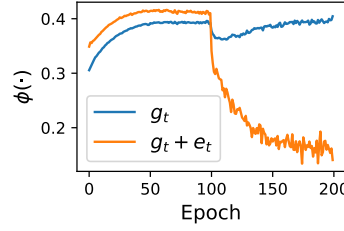


Figure 2.2 – The density $\phi(\cdot)$ for the stochastic gradients \mathbf{g}_t and the error-corrected stochastic gradients $\mathbf{g}_t + \mathbf{e}_t$ for VGG19 on CIFAR10 and batchsize 128 (See Sec. 2.7). Minimum value of $\phi(\mathbf{g}_t + \mathbf{e}_t)$ is greater than 0.13.

gradients are usually dense and hence $\phi(\mathbf{v})$ is much larger (see Fig. 2.2). Note that for our convergence rates, it is not the density of the gradient \mathbf{g}_t which matters but the density of the error-corrected gradient $\mathbf{g}_t + \mathbf{e}_t$. **Faster convergence than SGD?** (Kingma and Ba, 2014) and (Bernstein et al., 2018) note that different coordinates of the stochastic gradient \mathbf{g} may have different variances. In standard SGD, the learning rate γ would be reduced to account for the *maximum* of these coordinate-wise variances since otherwise the path might be dominated by the noise in these sparse coordinates. Instead, using coordinate-wise learning-rates like ADAM does, or using only the coordinate-wise sign of \mathbf{g} as SIGNSGD does, might mitigate the effect of such ‘bad’ coordinates by effectively scaling down the noisy coordinates. This is purported to be the reason why ADAM and SIGNSGD can be faster than SGD on train dataset.

In EF-SIGNSGD, the noise from the ‘bad’ coordinates gets accumulated in the error-term \mathbf{e}_t and is not forgotten or scaled down. Thus, if there are ‘bad’ coordinates whose variance slows down convergence of SGD, EF-SIGNSGD should be similarly slow. Confirming this, in a toy experiment with sparse noise (Appendix 9.1.1), SGD and EF-SIGNSGD converge at the same *slower* rate, whereas SIGNSGD is faster. However, our real world experiments contradict this—even with the feedback, EF-SIGNSGD is consistently *faster* than SGD, SIGNSGD, and SIGNSGDM on training data. Thus the coordinate-wise variance adaption explanation proposed by (Bernstein et al., 2018; Kingma and Ba, 2014) does not explain the faster convergence of EF-SIGNSGD, and is probably an incomplete explanation of why sign based methods or adaptive methods are faster than SGD!

2.6 Effect of SignSGD on generalization

So far our discussion has mostly focused on the convergence of the methods i.e. their performance on training data. However for deep-learning, we actually care about their performance on test data i.e. their generalization. It has been observed that the optimization algorithm being used significantly impacts the properties of the optima reached (Im et al., 2016; Li et al., 2018a). For instance, ADAM and SIGNSGD are known to generalize poorly compared with SGD (Wilson et al., 2017; Balles and Hennig, 2018).

The proposed explanation for this phenomenon is that in an over-parameterized setting, SGD reaches the ‘max-margin’ solution whereas ADAM and SIGNSGD do not (Zhang et al., 2017; Wilson et al., 2017), and (Balles and Hennig, 2018). As with the issues in convergence, the issues of SIGNSGD with generalization also turn out to be related to the biased nature of the sign operator. We explore how error-feedback may also alleviate the issues with generalization for any compression operator.

2.6.1 Distance to gradient span

Like (Zhang et al., 2017; Wilson et al., 2017), we consider an over-parameterized least-squares problem

$$\min_{\mathbf{x} \in \mathbb{R}^d} \left[f(\mathbf{x}) := \|\mathbf{A}\mathbf{x} - \mathbf{y}\|_2^2 \right],$$

where $\mathbf{A} \in \mathbb{R}^{n \times d}$ for $d > n$ is the data matrix and $\mathbf{y} \in \{-1, 1\}^n$ is the set of labels. The set of solutions $X^* := \{\mathbf{x} : f(\mathbf{x}) = 0\}$ of this problem forms a subspace in \mathbb{R}^d . Of particular interest is the solution with smallest norm:

$$\operatorname{argmin}_{\mathbf{x} \in X^*} \|\mathbf{x}\|^2 = \mathbf{A}^\dagger \mathbf{y} = \mathbf{A}^\top (\mathbf{A}\mathbf{A}^\top)^{-1} \mathbf{y},$$

as this corresponds to the *maximum margin* solution in the dual.

Maximizing margin is known to have a regularizing effect and is said to improve generalization (Valiant, 1984; Cortes and Vapnik, 1995).

The key property that SGD (with or without momentum) trivially satisfies is that the iterates always lie in the linear span of the gradients.

Lemma 9. *Given any over-parameterized least-squares problem, suppose that the iterates of the algorithm always lie in the linear span of the gradients and it converges to a 0 loss solution. This solution corresponds to the minimum norm/maximum margin solution.*

If we instead use a biased compressor (e.g. SIGNSGD), it is clear that the iterate may not lie in the span of the gradients. In fact it is easy to construct examples where this happens for SIGNSGD (Balles and Hennig, 2018), as well as top- k sparsification (Gunasekar et al., 2018), perhaps explaining the poor generalization of these schemes. Error-feedback is able to overcome this issue as well.

Theorem IV. *Suppose that we run Algorithm 2 for t iterations starting from $\mathbf{x}_0 = \mathbf{0}$. Let $\mathbf{G}_t = [\mathbf{g}_0^\top, \dots, \mathbf{g}_{t-1}^\top] \in \mathbb{R}^{d \times t}$ denote the matrix of the stochastic gradients and let $\Pi_{\mathbf{G}_t} : \mathbb{R}^n \rightarrow \operatorname{Im}(\mathbf{G}_t)$ denote the projection onto the range of \mathbf{G}_t .*

$$\|\mathbf{x}_t - \Pi_{\mathbf{G}_t}(\mathbf{x}_t)\|_2^2 \leq \|\mathbf{e}_t\|_2^2.$$

Here \mathbf{e}_t is the error as defined in Algorithm 2. The theorem follows directly from observing

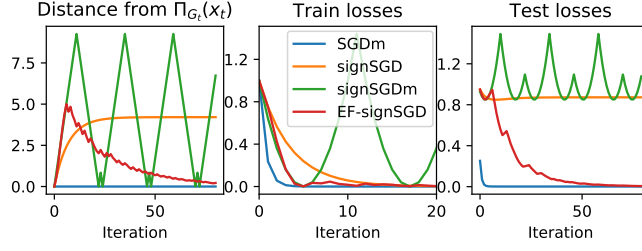


Figure 2.3 – Left shows the distance of the iterate from the linear span of the gradients $\|\mathbf{x}_t - \Pi_{G_t}(\mathbf{x}_t)\|$. The middle and the right plots show the train and test loss. SIGNSGD and SIGNSGDM have a high distance to the span, and do not generalize (test loss is higher than 0.8). Distance of EF-SIGNSGD to the linear span (and the test loss) goes to 0.

that $(\mathbf{x}_{t+1} + \mathbf{e}_{t+1}) = (\mathbf{x}_0 + \sum_{i=0}^t \gamma \mathbf{g}_i)$, and hence lies in the linear span of the gradients.

Remark 10. *Theorem IV along with Lemma 3 implies that the iterates of Algorithm 2 are always close to the linear span of the gradients.*

$$\|\mathbf{x}_t - \Pi_{G_t}(\mathbf{x}_t)\|_2^2 \leq \frac{4\gamma^2(1-\delta)}{\delta^2} \max_{i \in [t]} \|\mathbf{g}_i\|^2.$$

This distance further reduces as the algorithm progresses since the step-size γ is typically reduced.

2.6.2 Simulations

We compare the generalization of the four algorithms with full batch gradient: i) SGD ii) SIGNSGD, iii) SIGNSGDM, and iv) EF-SIGNSGD. The data is generated as in (Wilson et al., 2017) and is randomly split into test and train. The step-size γ and (where applicable) the momentum parameter β are tuned to obtain fastest convergence, but the results are representative across parameter choices.

In all four cases (Fig. 2.3), the train loss quickly goes to 0. The distance to the linear span of gradients is quite large for SIGNSGD and SIGNSGDM. For EF-SIGNSGD, exactly as predicted by our theory, it first increases to a certain limit and then goes to 0 as the algorithm converges. The test error, almost exactly corresponding to the distance $\|\mathbf{x}_t - \Pi_{G_t}(\mathbf{x}_t)\|$, goes down to 0. SIGNSGDM oscillates significantly because of the momentum term, however the conclusion remains unchanged—the best test error is higher than 0.8. This indicates that using error-feedback might result in generalization performance comparable with SGD.

2.7 Experiments

We run experiments on deep networks to test the validity of our insights. The results confirm that i) EF-SIGNSGD with error feedback always outperforms the standard SIGNSGD and SIGNSGDM, ii) the generalization gap of SIGNSGD and SIGNSGDM vs. SGD gets larger for

smaller batch sizes, iii) the performance of EF-SIGNSGD on the other hand is much closer to SGD, and iv) SIGNSGD behaves erratically when using small batch-sizes.

2.7.1 Experimental setup

All our experiments used the PyTorch framework (Paszke et al., 2019) on the CIFAR-10/100 dataset (Krizhevsky et al., 2009). Each experiment is repeated three times and the results are reported in Fig 2.4. Additional details and experiments can be found in Appendix 9.1.

Algorithms: We experimentally compared the following four algorithms: i) SGDM which is SGD with momentum, ii) (scaled)SIGNSGD with step-size scaled by the L_1 -norm of the gradient, iii) SIGNSGDM which is SIGNSGD using momentum, and iv) EF-SIGNSGD (Alg. 1). The scaled SIGNSGD performs the update

$$\mathbf{x}_{t+1} := \mathbf{x}_t - \gamma \frac{\|\mathbf{g}_t\|_1}{d} \text{sign}(\mathbf{g}_t).$$

We chose to include this in our experiments since we wanted to isolate the effects of error-feedback from that of scaling. Further we drop the unscaled SIGNSGD from our discussion here since it was observed to perform worse than the scaled version. As is standard in compression schemes (Seide et al., 2014; Lin et al., 2018; Wang et al., 2018), we apply our compression layer-wise. Thus the net communication for the (scaled) SIGNSGD and EF-SIGNSGD is $\sum_{i=1}^l (d_i + 32)$ bits where d_i is the dimension of layer i , and l is the total number of layers. If the total number of parameters is much larger than the number of layers, then the cost of the extra $32l$ bits is negligible—usually the number of parameters is three orders of magnitude more than the number of layers.

All algorithms are run for 200 epochs. The learning-rate is decimated at 100 epochs and then again at 150 epochs. The initial learning rate is tuned manually (see Appendix 9.1) for all algorithms using batch-size 128. For the smaller batch-sizes, the learning-rate is proportionally reduced as suggested in (Goyal et al., 2017). The momentum parameter β (where applicable) was fixed to 0.9 and weight decay was left to the default value of 5×10^{-4} .

Models: We use the VGG+BN+Dropout network on CIFAR-10 (VGG19) from (Simonyan and Zisserman, 2014) and Resnet+BN+Dropout network (Resnet18) from (He et al., 2016a). We adopt the standard data augmentation scheme and preprocessing scheme (He et al., 2016a,b). Our code builds upon on an open source library.¹

2.7.2 Results

The results of the experiments for Resnet18 on Cifar100 are shown in Fig. 2.4 and Table 2.1. Results for VGG19 on Cifar10 are also similar and can be found in the Appendix. We make four main observations:

¹github.com/kuangliu/pytorch-cifar

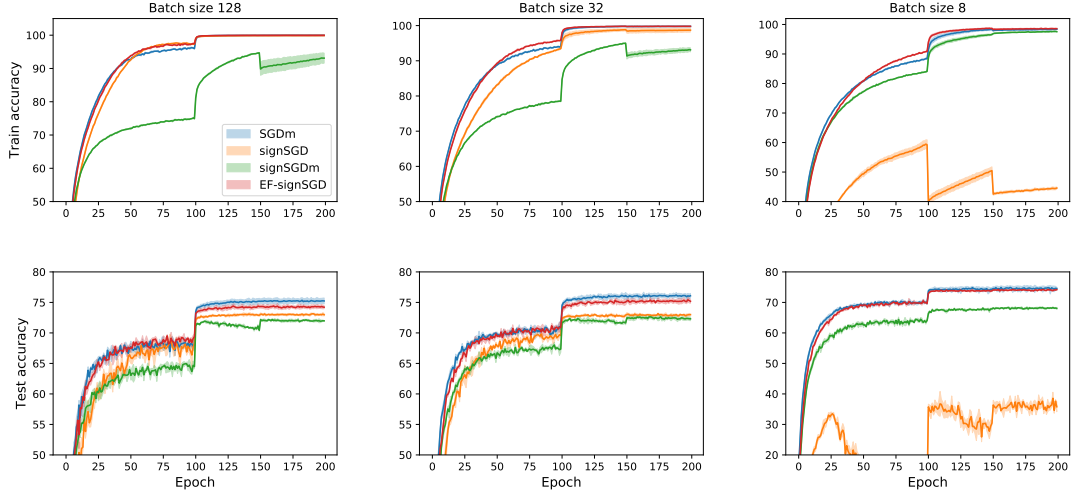


Figure 2.4 – Experimental results showing the train and test accuracy percentages on CIFAR-100 using Resnet18 for different batch-sizes. The solid curves represent the mean value and shaded region spans one standard deviation obtained over three replications. Note that the scale of the y-axis varies across the plots. EF-SIGNSGD consistently and significantly outperforms the other sign-based methods, closely matching the performance of SGDM.

Batch-size	SGDM	SIGNSGD	SIGNSGDM	EF-SIGNSGD
128	75.35	-2.21	-3.15	-0.92
32	76.22	-3.04	-3.57	-0.79
8	74.91	-36.35	-6.6	-0.64

Table 2.1 – Generalization gap on CIFAR-100 using Resnet18 for different batch-sizes. For SGDM we report the best mean test accuracy percentage, and for the other algorithms we report their difference to the SGDM accuracy (i.e. the generalization gap). EF-SIGNSGD has a much smaller gap.

EF-SIGNSGD is faster than SGDM on train. On the train dataset, both the accuracy and the losses (Fig. 9.2) is better for EF-SIGNSGD than for SGD for all batch-sizes and models (Fig. 9.3). In fact even SIGNSGD is faster than SGDM on the train dataset on VGG19 (Fig. 9.3) for batch-size 128. As we note in Section 2.5.3, the result that the scaled sign methods are also faster than SGD (and in fact faster than even the without feedback algorithms) overturns previously understood intuition (cf. (Kingma and Ba, 2014; Bernstein et al., 2018)) for why SIGNSGD and other adaptive methods outperform SGD—i.e. restricting the effect of a some ‘bad’ coordinates with high variance may not be the main reason why sign based methods are faster than SGD on train.

EF-SIGNSGD almost matches SGDM on test. On the test dataset (Table 2.1), the accuracy and the loss is much closer to SGD than the other sign methods across batch-sizes and models (Tables 9.2, 9.3). The generalization gap (both in accuracy and loss) reduces with decreasing batch-size. We believe this is because the learning-rate was scaled proportional to the batch-size and hence smaller learning-rates lead to smaller generalization gap, as was theo-

retically noted in Remark 10.

SIGNSGD performs poorly for small batch-sizes. The performance of SIGNSGD is always worse than EF-SIGNSGD indicating that scaling is insufficient and that error-feedback is crucial for performance. Further all metrics (train and test, loss and accuracy) increasingly become worse as the batch-size decreases indicating that SIGNSGD is indeed a brittle algorithm. In fact for batch-size 8, the algorithm becomes extremely unstable.

SIGNSGDM performs poorly on some datasets and for smaller batch-sizes. We were surprised that the training performance of SIGNSGDM is significantly worse than even SIGNSGD on CIFAR-100 for batch-sizes 128 and 32. On CIFAR-10, on the other hand, SIGNSGDM manages to be faster than SGDM (though still slower than EF-SIGNSGD). We believe this may be due to SIGNSGDM being sensitive to the weight-decay parameter as was noted in (Bernstein et al., 2018). We do not tune the weight-decay parameter and leave it to its default value for all methods (including EF-SIGNSGD). Further the generalization gap of SIGNSGDM gets worse for decreasing batch-sizes with a significant 6.6% drop in accuracy for batch-size 8.

2.8 Conclusion

We study the effect of biased compressors on the convergence and generalization of stochastic gradient algorithms for non-convex optimization. We have shown that biased compressors if naively used can lead to bad generalization, and even non-convergence. We then show that using error-feedback all such adverse effects can be mitigated. Our theory and experiments indicate that using error-feedback, our compressed gradient algorithm EF-SGD enjoys the same rate of convergence as original SGD—thereby giving compression *for free*. We believe this should have a large impact in the design of future compressed gradient schemes for distributed and decentralized learning. Further, given the relation between sign-based methods and ADAM, we believe that our results will be relevant for better understanding the performance and limitations of adaptive methods. Finally, in this work we only consider the single worker case. Developing a practical and scalable algorithm for multiple workers is a fruitful direction for future work.

* Acknowledgements. We are grateful to Tao Lin, Thijs Vogels, and Negar Foroutan for their help with the experiments. We also thank Jean-Baptiste Cordonnier, Konstantin Mishchenko, Jeremy Bernstein, and anonymous reviewers for their suggestions which helped improve our writeup.

3 PowerSGD: Practical Gradient Compression

3.1 Preface

Contribution and sources. This chapter is an extension of (Vogels et al., 2019). We additionally analyze here the convergence of error feedback when combined with momentum. Initial idea, theory, and design of the experiments was carried by the author. Thijs Vogels coded and ran all the experiments, and did a large portion of the writing. The excellent visualizations are also by TV. Detailed individual contributions:

SPK (author): Conceptualization, Methodology (50%), Formal analysis, Writing (30%)

Thijs Vogels: Software, Data visualization, Methodology (50%), Writing (70%)

Martin Jaggi: Writing – review and editing, Administration, Supervision .

Summary. Gradient compression was introduced to alleviate the communication bottleneck and speed up distributed training. However, upon testing in realistic settings, we find that most compression techniques are flawed: i) they suffer a large drop in accuracy, or ii) are *slower* than vanilla SGD in terms of the wall-clock time.

In this chapter, we incorporate real world system constraints into the error-feedback framework, reducing communication by (100×) and wall-clock time by (2×). This is achieved by designing a novel low rank compressor which uses power iterations and combining it with modern optimizers such as momentum and Adam. Crucially, our compressor involves only matrix multiplications (hence can leverage GPUs for fast encoding/decoding), and is also linear (hence is compatible with all-reduce). We show that our method works ‘out of the box’ without requiring additional hyper-parameter tuning, and is ready for adoption in practice.

Since its introduction, our method remains the state of the art for practical communication compression (cf. Agarwal et al. (2021) for a recent evaluation) and has found wide-spread adoption. It is the default compression method in PyTorch (DDP), the most popular deep learning framework, and was also a crucial part of training DALLE (Ramesh et al., 2021) where PowerSGD was used to train a 1 billion parameter model with 64 GPUs.

3.2 Introduction

Synchronous data-parallel SGD is the most common method for accelerating training of deep learning models (Dean et al., 2012; Iandola et al., 2016; Goyal et al., 2017). Because the gradient vectors of such models can be large, the time required to share those gradients across workers limits the scalability of deep learning training (Seide et al., 2014; Iandola et al., 2016; Lin et al., 2018).

Previous work proposes lossy gradient compression as a solution to this issue. Notable examples include replacing the coordinates of the gradient with only their sign (Seide et al., 2014; Carlson et al., 2015; Bernstein et al., 2018, 2019; Karimireddy et al., 2019), quantizing the individual coordinates (Alistarh et al., 2017; Wen et al., 2017), and low-rank approximation of the gradient (Wang et al., 2018). While these works demonstrate speedups over full-precision SGD in some settings, we find that their speedups vanish with a fast network and highly optimized communication backend, even on commodity hardware. Some prior work also suffers from degraded test accuracy compared to SGD. We combine three observations to fix these issues: i) Linear compressor operators achieve scalability by enabling aggregation using all-reduce. ii) Error feedback ensures convergence with general biased compressors. iii) Low-rank updates enable aggressive compression without sacrificing quality.

First, we explore the properties of various gradient compression schemes for SGD and identify which ones are crucial for high scalability. In particular, we note that currently proposed gradient compressors are not linear. Their compressed messages cannot be added up hierarchically, unlike raw gradients. This prevents current compressed SGD algorithms from aggregating gradients using an efficient *reduce* operation and instead require a *gather* operation. Current deep learning frameworks rely either solely or predominantly on all-reduce, which is key to why regular SGD scales well with fast communication hardware (cf. Awan et al., 2018; Panda et al., 2019).

Secondly, it was recently shown that using error feedback (i.e. storing the difference between the computed and compressed gradient, and reinserting it at the next iteration) improves both convergence and generalization for compression schemes (Karimireddy et al., 2019). This can enable general biased gradient compression schemes to reach the target test accuracy.

Thirdly, there is growing evidence that the generalization ability of modern over-parameterized deep learning models is related to low-rankedness (Arora et al., 2018; Martin and Mahoney, 2018; Collins et al., 2018). Using a low-rank update (as we do) can be viewed as implicitly performing spectral regularization (Gunasekar et al., 2018) and hence can be expected to have good generalization properties (Yoshida and Miyato, 2017). Further, (Wang et al., 2018) show that the eigenspectrum of the stochastic gradients for deep learning models decays, suggesting that a rank-based schemes can get away with aggressive compression without sacrificing convergence.

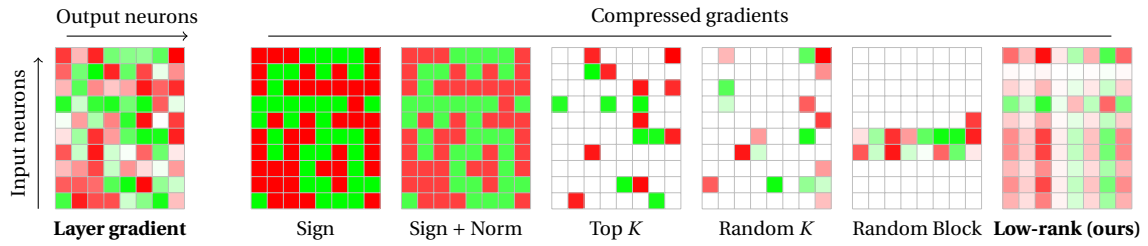


Figure 3.1 – Compression schemes compared in this paper. Left: Interpretation of a layer’s gradient as a matrix. Coordinate values are color coded (**positive**, **negative**). Right: The output of various compression schemes on the same input. Implementation details are in Appendix 10.7.

In this work, we design POWERSGD with the above observations in mind. POWERSGD computes a low-rank approximation of the gradient using a generalized *power* iteration (known as subspace iteration (Stewart and Miller, 1975)). The approximation is computationally lightweight, avoiding any prohibitively expensive Singular Value Decomposition. To improve the quality of the efficient approximation, we *warm-start* the power iteration by reusing the approximation from the previous optimization step. Using all-reduce gradient aggregation, we empirically demonstrate that POWERSGD achieves wall-clock speedups over regular SGD in a 16-GPU setting, even with the optimized NCCL communication backend on a fast network (and is the only algorithm to do so.) By compressing gradients more than 120 \times , we reduce communication time (including coding and decoding) by 54% for RESNET18 on CIFAR10 and by 90% for an LSTM on WIKITEXT-2. End-to-end wall-clock training time to full test quality is reduced by 24% for RESNET18 and by 55% for the LSTM.

3.3 Related work

Gradient compression. A variety of compression schemes (Figure 3.1) have been proposed: (Alistarh et al., 2017) and (Wen et al., 2017) quantize each gradient coordinate; (Seide et al., 2014; Carlson et al., 2015; Bernstein et al., 2018, 2019) and (Karimireddy et al., 2019) replace each coordinate of the gradient with its sign; (Lin et al., 2018; Stich et al., 2018) and (Wangni et al., 2018) use the largest few coordinates; and (Konečný et al., 2016) and (Wang et al., 2018) use a low-rank approximation.

Spectral Atomo by (Wang et al., 2018) is perhaps the closest to our work. It performs importance sampling of the gradient’s singular vectors and is an unbiased compression scheme. It requires, however, a full Singular Value Decomposition every iteration and is hence computationally impractical.

Commutative compression and addition. (Yu et al., 2018) stress that commutability of compression with gradient addition enables efficient aggregation with *ring all-reduce*. Most compressors, however, lack this property. Yu et al. utilize temporally-consistent correlations between gradients coordinates to compress them linearly. POWERSGD has a similar property that we call ‘linearity’.

Error feedback. First introduced in (Seide et al., 2014) and analyzed in (Stich et al., 2018) for the convex case, error feedback involves computing the difference between a worker’s gradient and the compressed gradient (i.e. *error*) and adding it back to the next gradient (*feedback*). (Karimireddy et al., 2019) and (Stich and Karimireddy, 2020) further develop and generalize the framework of error feedback with improved rates. In the non-convex setting, (Karimireddy et al., 2019) show that error feedback is crucial both for convergence and generalization when using biased compressors (e.g. sign or top- K). In general, biased compression schemes equipped with error feedback tend to out-perform their unbiased counterparts. The practical algorithm by (Lin et al., 2018) is also as an approximate top- K compressor with error feedback.

Low-rank methods. Recent works argue that in modern over-parameterized deep networks, the final model learnt has a ‘low stable rank’ (Martin and Mahoney, 2018; Li et al., 2018c). This can partially explain their impressive generalization properties despite being substantially overparameterized (Arora et al., 2018). Adding explicit spectral regularization has shown to further improve the performance of such models (Mazumder et al., 2010; Yoshida and Miyato, 2017). Using a low-rank update (as we do) can be viewed as implicitly performing a similar regularization (Gunasekar et al., 2018). If the target matrices are known to be exactly low-ranked (instead of just low stable rank), (Yurtsever et al., 2017) show that it is sometimes possible to converge to the optima using low rank approximations of the gradients without the need for error feedback.

3.4 Method

In data-parallel optimization of machine learning models, a number of W workers share the same model parameters $\mathbf{x} \in \mathbb{R}^d$. They iteratively update \mathbf{x} by computing independent stochastic gradients, aggregating these gradients by averaging¹, and updating the model parameters based on this aggregate.

POWERSGD compression We approximate each layer in the model independently. The parameters of fully-connected layers (dense matrix multiplication) and their gradients have an inherent matrix structure. The parameters of convolutional layers can be naturally interpreted as fully-connected layers applied repeatedly over a 2D grid of inputs. Practically, this amounts to flattening input and kernel dimensions in the 4D gradient tensors. Neural networks also contain bias vectors, but these typically constitute a tiny fraction of the parameter space and can be aggregated uncompressed.

For each parameter’s gradient $M \in \mathbb{R}^{n \times m}$, the aim of rank- r matrix approximation is to find matrices $P \in \mathbb{R}^{n \times r}$ and $Q \in \mathbb{R}^{m \times r}$ such that PQ^\top approximates M well. POWERSGD uses a single step of subspace iteration—*power* iteration generalized to $r > 1$ —to compute such an approximation. This involves performing one right multiplication, one left multiplica-

¹(Bernstein et al., 2019) propose Signum which aggregates 1-bit gradients by majority voting instead of averaging.

Algorithm 3 Rank- r POWERSGD compression

```

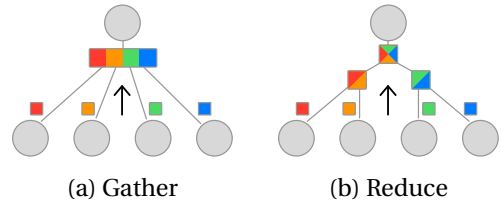
1: The update vector  $\Delta_w$  is treated as a list of tensors corresponding to individual model
   parameters. Vector-shaped parameters (biases) are aggregated uncompressed. Other pa-
   rameters are reshaped into matrices. The functions below operate on such matrices in-
   dependently. For each matrix  $M \in \mathbb{R}^{n \times m}$ , a corresponding  $Q \in \mathbb{R}^{m \times r}$  is initialized from an
   iid standard normal distribution.
2: function COMPRESS+AGGREGATE(update matrix  $M \in \mathbb{R}^{n \times m}$ , previous  $Q \in \mathbb{R}^{m \times r}$ )
3:    $P \leftarrow MQ$ 
4:    $P \leftarrow \text{ALL REDUCE MEAN}(P)$   $\triangleright$  Now,  $P = \frac{1}{W}(M_1 + \dots + M_W)Q$ 
5:    $\hat{P} \leftarrow \text{ORTHOGONALIZE}(P)$   $\triangleright$  Orthonormal columns
6:    $Q \leftarrow M^\top \hat{P}$ 
7:    $Q \leftarrow \text{ALL REDUCE MEAN}(Q)$   $\triangleright$  Now,  $Q = \frac{1}{W}(M_1 + \dots + M_W)^\top \hat{P}$ 
8:   return the compressed representation  $(\hat{P}, Q)$ .
9: end function
10: function DECOMPRESS( $\hat{P} \in \mathbb{R}^{n \times r}$ ,  $Q \in \mathbb{R}^{m \times r}$ )
11:   return  $\hat{P}Q^\top$ 
12: end function

```

tion, and an orthogonalization. We use the Gram-Schmidt procedure to orthogonalize our matrices since they have very few columns (1–4), and this is the most expensive part of the compression procedure. Further, we ‘warm-start’ the subspace iteration by reusing the approximation computed at the previous step. With the inclusion of warm-start, a *single* step of subspace iteration yields a factorization $M \sim PQ^\top$ with the same performance as the best rank- r approximation from an expensive Singular Value Decomposition.

Efficient aggregation between workers In data-parallel optimization, we want to approximate the *average* of the worker’s gradients. Suppose POWERSGD operates on a list of corresponding gradients $[M_1 \dots M_W]$ from W workers. Both occurrences of M in the algorithm are a (linear) matrix multiplication followed by a (linear) mean reduction over workers. This introduces a practical invariance: execution on 1 worker with batch size $B \times W$ is equivalent to execution on W workers with batch size B each. We call this property ‘linearity’. Refer to Appendix 3.5.1 for more details.

An important benefit of the POWERSGD’s linearity is that it can be implemented using the **all-reduce** protocol as opposed to needing a gather operation. To illustrate the difference, suppose that we want to compute the sum of W matrices $\sum_{i=1}^W M_i$ for $W = 4$. The all-reduce method can use associativity of addition to rewrite the computation as $(M_1 + M_2) + (M_3 + M_4)$. This enables a divide-and-conquer approach and allows the summation task to be split over multiple workers, as illustrated on the right. With W workers, both the computation and the communication time scale as $\mathcal{O}(\log W)$ for all-reduce, compared to $\mathcal{O}(W)$ for all-gather.



In addition to improved scaling, all-reduce communication is preferred over a parameter-server setting because it avoids *double compression*. With a parameter server, both the ‘clients \rightarrow server’ and ‘server \rightarrow clients’ communication have to be compressed (Bernstein et al., 2019; Seide et al., 2014). We avoid this by merging compression and aggregation into one step.

Error-feedback SGD Since the POWERSGD scheme is biased (i.e. compressing and decompressing a random gradient does not yield the original in expectation), we use error feedback (Seide et al., 2014; Karimireddy et al., 2019). Our version of error feedback (Algorithm 4) extends the original by introducing post-compression *momentum*. This simple extension allows us to reuse the same learning rate and hyper-parameters as those tuned for SGD with momentum.

Algorithm 4 Distributed Error-feedback SGD with Momentum

```

1: hyperparameters: learning rate  $\gamma$ , momentum parameter  $\lambda$ 
2: initialize model parameters  $\mathbf{x} \in \mathbb{R}^d$ , momentum  $\mathbf{m} \leftarrow \mathbf{0} \in \mathbb{R}^d$ , replicated across workers
3: at each worker  $w = 1, \dots, W$  do
4:   initialize memory  $\mathbf{e}_w \leftarrow \mathbf{0} \in \mathbb{R}^d$ 
5:   for each iterate  $t = 0, \dots$  do
6:     Compute a stochastic gradient  $\mathbf{g}_w \in \mathbb{R}^d$ .
7:      $\Delta_w \leftarrow \mathbf{g}_w + \mathbf{e}_w$  ▷ Incorporate error-feedback into update
8:      $\mathcal{C}(\Delta_w) \leftarrow \text{COMPRESS}(\Delta_w)$ 
9:      $\mathbf{e}_w \leftarrow \Delta_w - \text{DECOMPRESS}(\mathcal{C}(\Delta_w))$  ▷ Memorize local errors
10:     $\mathcal{C}(\Delta) \leftarrow \text{AGGREGATE}(\mathcal{C}(\Delta_1), \dots, \mathcal{C}(\Delta_W))$  ▷ Exchange gradients
11:     $\Delta' \leftarrow \text{DECOMPRESS}(\mathcal{C}(\Delta))$  ▷ Reconstruct an update  $\in \mathbb{R}^d$ 
12:     $\mathbf{m} \leftarrow \lambda \mathbf{m} + \Delta'$ 
13:     $\mathbf{x} \leftarrow \mathbf{x} - \gamma (\Delta' + \mathbf{m})$ 
14:   end for
15: end at

```

3.5 Theoretical Analysis of POWERSGD

The proof of convergence of EF-SGD with momentum can be derived by incorporating a few key changes to the proof of (Karimireddy et al., 2019): i) we are in a multi-worker setting, and ii) we incorporate the techniques introduced by (Ghadimi and Lan, 2016) to handle the additional *momentum*. Further, $\|\cdot\|^2$ unless otherwise specified is always the standard euclidean norm for vectors, and is the *Frobenius* norm for matrices.

Suppose that we want to minimize a continuous (possibly) non-convex function $f: \mathbb{R}^d \rightarrow \mathbb{R}$:

$$f^\star = \min_{\mathbf{x} \in \mathbb{R}^d} f(\mathbf{x}).$$

The classic stochastic gradient algorithm (SGD) (Robbins and Monro, 1951) when adapted to

the distributed optimization setting performs iterations of the form

$$\mathbf{x}_{t+1} := \mathbf{x}_t - \gamma \mathbf{g}_t, \text{ where} \quad (3.1)$$

$$\mathbf{g}_t = \frac{1}{W} \sum_{w=1}^W \mathbf{g}_{t,w} \quad \text{and} \quad \mathbb{E}[\mathbf{g}_t] = \nabla f(\mathbf{x}_t).$$

Here $\gamma \in \mathbb{R}$ is the step-size (or learning-rate) and $\mathbf{g}_{t,w}$ is the stochastic gradient computed by the w th worker for $w \in \{1, \dots, W\}$ workers.

Now EF-SGD (Algorithm 4) when run on the W workers with step-size γ and momentum parameter λ can be rewritten making the dependence on iteration t explicit as follows:

$$\begin{aligned} \Delta'_t &= \text{DECOMPRESS}(\text{COMPRESS}(\mathbf{g}_t + \mathbf{e}_t)), \\ \mathbf{m}_{t+1} &= \Delta'_t + \lambda \mathbf{m}_t, \\ \mathbf{x}_{t+1} &= \mathbf{x}_t - \gamma(\Delta'_t + \mathbf{m}_{t+1}), \text{ and} \\ \mathbf{e}_{t+1} &= (\mathbf{g}_t + \mathbf{e}_t) - \Delta'_t. \end{aligned} \quad (3.2)$$

3.5.1 Single/multi worker equivalence

The difference between the update as written in (3.2) and Algorithm 4 is that the error computation and compression is performed on the *aggregated* gradient \mathbf{g}_t instead of on the individual workers' gradients $\mathbf{g}_{t,w}$. While in general these are not equivalent, the linearity of POWERSGD ensures that these are indeed equivalent. This implies that POWERSGD has the neat property that the algorithm is equivalent if run on W workers or a single worker with a larger batch-size. This does not hold for most other schemes (e.g. sign based compression schemes, QSGD, etc.).

Lemma 11 (Equivalence of single worker and multi worker updates). *The updates in POWERSGD (i.e. Algorithm 4 using Compressor 3) are equivalent to the updates (3.2).*

3.5.2 Convergence of multi-worker error feedback with momentum

With Lemma 3.5.1, we can restrict our analysis to updates (3.2) and forget about the difference between the single worker and multi-worker case.

Theorem V (Non-convex convergence of EF-SGD). *Let us run T steps of Algorithm 4 with momentum parameter $\lambda \in [0, 1)$ and effective step-size $\tilde{\gamma} \leq \frac{1}{2L}$. Then the following holds given standard assumptions:*

$$\frac{1}{T+1} \sum_{t=0}^T \mathbb{E}[\|\nabla f(\mathbf{x}_t)\|^2] \leq \frac{4(f(\mathbf{x}_0) - f^*)}{\tilde{\gamma}(T+1)} + \frac{4\tilde{\gamma}\sigma^2}{W} + \tilde{\gamma}^2 C,$$

where

$$\tilde{\gamma} = \gamma \left(1 + \frac{1}{1-\lambda}\right), \quad C = 8L^2 B^2 \left(\frac{4(1-\delta)}{\delta} + \frac{\lambda}{(1-\lambda)(2-\lambda)} \right),$$

and f^* is the optimum value of f .

Note that the condition on the ‘effective’ steps-size $\tilde{\gamma} \leq \frac{1}{2L}$ implies a condition on the true step-size $\gamma \leq \frac{2-\lambda}{2L(2-\lambda)}$.

Remark 12 (No dependence on δ or λ). *The asymptotic rate does not depend on either the compression quality δ , nor on the momentum term λ . With an appropriate choice of the step-size, we obtain a rate of*

$$\frac{1}{T+1} \sum_{t=0}^T \mathbb{E}[\|\nabla f(\mathbf{x}_t)\|^2] \leq \mathcal{O} \left(\sqrt{\frac{Lf_0\sigma^2}{W(T+1)}} + \frac{C}{T+1} \right).$$

Since δ and λ only affect the faster C/T term, they have no bearing on the asymptotic rate of the algorithm. This is similar to what was noted by (Karimireddy et al., 2019) when using EF-SGD without the momentum, and matches the rate of vanilla SGD.

Remark 13 (Scaling with workers W). *The rate in Theorem V also shows that if we increase the number of workers W , we get a linear decrease in the variance. As long as the first term is dominant (i.e. T is large enough), this implies linear speed-up with increasing number of workers.*

3.6 Ablation study of POWERSGD

In this section, we consider different aspects of POWERSGD in isolation and hope to empirically understand: i) the effect of using error feedback, ii) the effect of ‘warm-start’, and iii) the trade-off between test accuracy and compression rate with varying approximation rank.

3.6.1 Effect of error feedback

Using error-feedback SGD as a base algorithm for POWERSGD has two advantages. First, it enables our use of a biased compressor. Secondly, EF-SGD improves convergence and obtains better test accuracy (Karimireddy et al., 2019).

To illustrate the improved test accuracy, we compare POWERSGD—a biased compressor with error feedback—against an unbiased low-rank approximation. To approximate a matrix $M \in \mathbb{R}^{n \times m}$, the unbiased rank- r approximator samples a random matrix $U \in \mathbb{R}^{m \times r}$ such that $\mathbb{E}[UU^\top] = I_m$ and outputs (MU, U) as the low-rank approximation. This scheme is unbiased since

$$\mathbb{E}[(MU)U^\top] = M\mathbb{E}[UU^\top] = MI = M.$$

POWERSGD is the natural biased counterpart of this unbiased scheme. Table 3.1 demon-

3.6. Ablation study of POWERSGD

Table 3.1 – Rank-based compression with and without error feedback. The biased POWERSGD outperforms an unbiased linear rank- r compressor on test accuracy.




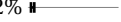
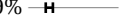


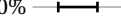
Algorithm	Test accuracy	Data/epoch
SGD	94.3% 	1023 MB
Rank-1 POWERSGD	93.6% 	4 MB
Rank-2 POWERSGD	94.4% 	8 MB
Unbiased Rank 1	71.2% 	3 MB
Unbiased Rank 2	75.9% 	4 MB

Table 3.2 – Best rank-2 approximation vs. POWERSGD. Warm-start improves test accuracy, even matching the performance of the best rank-2 approximation.

Algorithm	Test accuracy
Best approximation	94.4% 
Warm start (default)	94.4% 
Without warm start	94.0% 

strates that our biased approximator with error feedback outperforms the unbiased operator on image classification.

3.6.2 Effect of warm-start

POWERSGD does not compute the best rank- r approximation of a gradient matrix, but uses a cheaper, low-fidelity approximation based on power iteration. Comparing the time per batch of POWERSGD and Spectral Atomo in Table 3.6, we see the importance of avoiding a Singular Value Decomposition. With gradients shaped as in POWERSGD, computing the SVD of a stochastic gradient takes 673ms, the equivalent of computing 6 mini-batch gradients. In contrast, one full step of rank-2 POWERSGD, including communication between 16 workers, takes only 105ms.

Given that we only use a single step of power iteration, the quality of the approximation suffers—compare the test accuracy of ‘without warm start’ and ‘best approximation’ in Table 3.2. A key feature of POWERSGD is the *warm start* strategy which reuses previously computed matrix approximations to initialize the power iteration algorithm. If the matrix on which we perform power iteration remains constant, then this recovers the best rank- r approximation (see Theorem XXV in the Appendix). We argue that this strategy sometimes makes sense even if the underlying matrices are varying.

Suppose we approximate the sequence of gradient matrices $\{M_t\}$ at timesteps t . At timestep t , we leverage the previous factorization $M_{t-1} \approx P_{t-1}Q_{t-1}^\top$. If $M_t \approx M_{t-1}$ then we would benefit from reusing P_{t-1} and Q_{t-1} as our starting point. While this is unlikely to be true, if M_t and M_{t-1} are stochastic approximations of the full gradient, we can expect that $\mathbb{E}[M_t] \approx \mathbb{E}[M_{t-1}]$ since the function is smooth and we only take small update steps. The result is akin to Oja’s algorithm for *stochastic power iteration* (Oja, 1982), and hence could result in an improved approximation quality. As we show empirically in Table 3.2, this ‘warm starting’ strategy is sufficient to close the gap in test accuracy between POWERSGD and the much more expensive best rank- r approximation.

Chapter 3. PowerSGD: Practical Gradient Compression

Table 3.3 – POWERSGD with varying rank. With sufficient rank, POWERSGD accelerates training of a RESNET18 and an LSTM by reducing communication, achieving test quality on par with regular SGD in the same number of iterations. The time per batch includes the forward/backward pass (constant). See Section 3.7 for the experimental setup.

Image classification — RESNET18 on CIFAR10					
Algorithm	Test accuracy	Data sent per epoch		Time per batch	
SGD	94.3%	1023 MB	(1×)	312 ms	+0%
Rank 1	93.6%	4 MB	(243×)	229 ms	−26%
Rank 2	94.4%	8 MB	(136×)	239 ms	−23%
Rank 4	94.5%	14 MB	(72×)	260 ms	−16%

Language modeling — LSTM on WIKITEXT-2					
Algorithm	Test perplexity	Data sent per epoch		Time per batch	
SGD	91	7730 MB	(1×)	300 ms	+0%
Rank 1	102	25 MB	(310×)	131 ms	−56%
Rank 2	93	38 MB	(203×)	141 ms	−53%
Rank 4	91	64 MB	(120×)	134 ms	−55%

Default experimental setting

Dataset	CIFAR10
Architecture	RESNET18
Number of workers	16
Backend	NCCL (fastest in PyTORCH)
Batch size	128 × number of workers
Momentum	0.9
Learning rate	Tuned for 16 workers — 0.1 × 16 for SGD. Scaled linearly by the number of workers
LR decay	/10 at epoch 150 and 250
LR warmup	Linearly within 5 epochs, starting from the single-worker LR
# Epochs	300
Weight decay	10 ^{−4} , 0 for BatchNorm parameters
Repetitions	3, with varying seeds
Error bars	min — max

3.6.3 Effect of varying the rank

















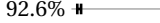



POWERSGD allows users to choose the rank of its gradient approximations. The trade-off between approximation quality and compression, decompression and transfer cost is explored in Table 3.3. In both the image classification and language modeling tasks we explore, the test quality achieved by POWERSGD grows with increasing rank. In both cases, it reaches a quality that is as good, or even slightly better than regular SGD.

3.7 Empirical evaluation of POWERSGD

This section demonstrates the practicality of POWERSGD for distributed optimization of deep neural networks. We show that the compression scheme of POWERSGD i) is fast and matches test performance of SGD, ii) scales well with increasing workers even with a sub-optimal communication backend, and iii) significantly reduces training time for larger models.

3.7. Empirical evaluation of POWERSGD

Table 3.4 – Comparing different compression operators for Error-feedback SGD in a unified setting; running 300 epochs of Error-feedback SGD with Momentum (Algorithm 4) with a learning rate tuned for full-precision SGD on 16 GPUs for CIFAR10. Note that the variations of POWERSGD with ranks 2 and 7 strike the best balance between the achieved test accuracy and time per batch (total time for forward, backward, compression, decompression, and gradient aggregation).

		Test accuracy	Sent/epoch	All-reduce	Time/batch
No compression		94.3% 	1023 MB	✓	312 ms 
Medium	Rank 7	94.6% 	24 MB	✓	285 ms 
	Random Block	93.3% 	24 MB	✓	243 ms 
	Random K	94.0% 	24 MB	✓	540 ms 
	Sign+Norm	93.9% 	32 MB	×	429 ms 
	Top K	94.4% 	32 MB	×	444 ms 
High	Rank 2	94.4% 	8 MB	✓	239 ms 
	Random Block	87.8% 	8 MB	✓	240 ms 
	Random K	92.6% 	8 MB	✓	534 ms 
	Top K	93.6% 	8 MB	×	411 ms 

Most of the analysis is performed on CIFAR10, in the setting described in the table on the right. We verify the generality of POWERSGD by an additional evaluation of an LSTM for language modeling on WIKITEXT-2. We use 16 GPUs on 8 machines, connected through a fast (10Gbit/s) network. To obtain meaningful timings, we have aimed to optimize all compared optimizers to a similar level. We provide a list of our performance optimizations in Appendix 10.8. Throughout these results, we tune the learning rate for full-precision SGD, and use the *same* parameters for POWERSGD and other compression algorithms that use error feedback with momentum. Learning rates for the compared-to Spectral Atomo (Wang et al., 2018) and Signum (Bernstein et al., 2019) were separately tuned cf. Appendix 10.9.

3.7.1 Comparison with other compressors

Error feedback in compressed optimization enables the use of a multitude of compression schemes, including biased ones. The potential compression operators illustrated in Figure 3.1 are compared in Table 3.4. We evaluate compressors based on the test accuracy achieved and the total time taken to process one mini-batch. The former is a holistic measure of the accuracy of the compression operator, and the latter is the net time required for a forward pass, backward pass, gradient compression and decompression and gradient communication. We study two compression regimes—medium and high.

At around $32\times$ compression, achieved by sign-based methods, all compression schemes (other than Random Block) achieve test accuracy close to full-precision SGD. This implies that all schemes in this regime (other than Random Block) obtain a good-enough compression quality. At high compression ($128\times$), POWERSGD particularly stands out as the only method to achieve the target test accuracy.

Table 3.5 – Breakdown of time spent (in seconds) in one iteration of RESNET18 training. Because POWERSGD (Rank 2) uses all-reduce, time spent encoding/decoding gradients is constant.

■ Forward pass, ■ Backward pass, ■ Gradient exchange, ■ Encoding and decoding.

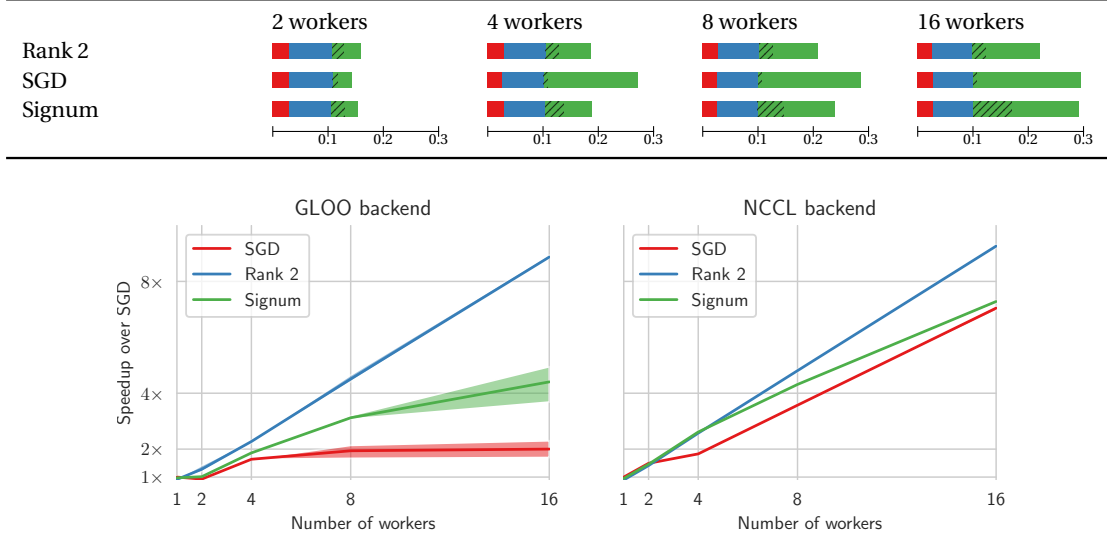


Figure 3.3 – Scaling of POWERSGD on CIFAR10 compared to full-precision SGD and Signum (Bernstein et al., 2019) on two communication backends. The batch size increases linearly with the number of workers. We compare training time for one epoch to 1-worker SGD. Note that the faster NCCL backend used throughout benefits the baselines more than our method.

In both the medium and high compression settings, the only schemes to be faster than full-precision SGD are POWERSGD and Random Block. Note that both are simple linear schemes and hence support all-reduce. While Random K also supports all-reduce, the overhead for random memory access during both the compression and decompression stages is substantial, making it slower overall than SGD. Thus, on modern GPU-enabled infrastructure, POWERSGD, which relies on matrix multiplication, is faster and much more accurate than the other compression schemes.

3.7.2 Scalability of POWERSGD

Here we investigate how POWERSGD scales with an increasing number of workers, shedding light on what we can expect if we use a significantly larger number of workers. Additionally, we investigate how these results depend on the choice of communication backend. We benchmark POWERSGD against SGD and Signum (signSGD with majority vote) from (Bernstein et al., 2019) which we believe is the current state-of-the-art for distributed algorithms.

Table 3.5 provides a detailed breakdown of the time spent for each mini-batch (i.e. one step) into the forward pass, backward pass, gradient exchange (communication), and compres-

3.7. Empirical evaluation of POWERSGD

Table 3.6 – Results on CIFAR10.

Contrary to rank-2 Spectral Atomo (Wang et al., 2018) and Signum (Bernstein et al., 2019), POWERSGD achieves the same test accuracy as full-precision SGD within the default epoch budget.


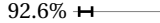


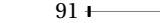
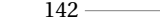
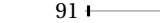
Algorithm	Test accuracy	Data/epoch	Time per batch	
SGD	94.3% 	1023 MB	312 ms	+0%
Atomo	92.6% 	113 MB	948 ms	+204%
Signum	93.6% 	32 MB	301 ms	−3%
Rank 2	94.4% 	8 MB	239 ms	−23%

Table 3.7 – In language modeling, rank-4 POWERSGD achieves the target test accuracy and provides a significant speedup over SGD.

Algorithm	Test perplexity	Data/epoch	Time per batch	
SGD	91 	7730 MB	300 ms	+0%
Signum	142 	242 MB	424 ms	+41%
Rank 4	91 	64 MB	134 ms	−55%

sion/decompression. The time spent in the forward and backward pass is constant across all algorithms and numbers of workers. Since both SGD and POWERSGD use all-reduce, the gradient communication time (solid green in Table 3.5) scales gracefully with increasing number of workers. Signum—which uses all-gather instead of all-reduce—has a steeper increase. It has comparable time to POWERSGD for 4 workers but becomes more expensive for 16 workers.

There is another, more subtle, consequence of all-reduce vs. all-gather on the decoding times. In all-reduce, the *aggregation* step and the *communication* step happen simultaneously. Each worker receives a pre-aggregated gradient, making the cost of decompression independent of the number of workers. On the other hand, in all-gather, a worker receives W compressed gradients that need to be individually decompressed and aggregated (either using majority vote or averaging). The time for decompression with all-gather therefore scales linearly with number of workers. This shows when comparing the hatched regions in Table 3.5. This observation speaks to the importance of the reduce operation for scalability.

We next study two different backends—the more optimized NCCL and the slower GLOO. All three methods scale reasonably well with the optimized NCCL backend, although Signum has a slope less than 1 in the log-log plot, indicating sub-linear scaling. On the slower GLOO backend, POWERSGD is notably the only method that retains excellent scaling due to its high compression rate.

3.7.3 Other tasks and methods

In Table 3.6, we compare POWERSGD against the state-of-the-art compressed optimization algorithms Signum and Spectral Atomo. The cost of performing a full SVD at each step renders Spectral Atomo impractical in a high-performance setting, especially considering that it fails to match the test accuracies of the other methods. Signum performs much better, prov-

ing a minor speedup over SGD. POWERSGD is the fastest and most accurate of the compared methods.

The advantage of POWERSGD truly shows when using really large models, i.e. where the communication actually becomes a bottleneck. To verify this, we run Signum, full-precision SGD, and POWERSGD to train an LSTM on a language modeling task which has a substantially larger model size than RESNET18 (see Appendix 10.6). To match the test score of full-precision SGD, we needed to use a rank-4 approximation (see Section 3.6.3). POWERSGD reduces communication by 90% and the overall running time by 55%, while Signum becomes slower than full-precision SGD and also obtains a worse test score.

Convergence curves on test accuracy corresponding to Tables 3.3, 3.6 and 3.7 are provided in Appendix 10.3. In those figures, you can read our improvements in time-to-accuracy for any target accuracy. We also provide a case study on using PowerSGD for a novel task (language modeling with transformers on WIKITEXT-2) and more workers (32) on the public cloud in Appendix 10.4.

3.8 Conclusion

Gradient compression is a promising approach to tackling the communication bottleneck in synchronous distributed optimization. Thus far, however, it has not found widespread adoption because existing compression schemes either run slower than SGD with optimized all-reduce gradient aggregation, or more importantly do not reach the same test performance. We see POWERSGD as the first practical gradient compression method, and believe it is ready for adaptation in practice.

The key to the practicality of POWERSGD is its linear compression scheme that is cheap to compute and allows for all-reduce gradient aggregation, while simultaneously matching the test performance of full-precision SGD. This speedup gained over SGD actually *increases* for larger models such as those commonly found in NLP. Further, as a result of our modifications to the error feedback algorithm, POWERSGD is a plug-in replacement for SGD with momentum, avoiding the need for additional hyper-parameter tuning. We expect that these properties of POWERSGD will enable training of even larger models with even more workers than what is possible with full-precision SGD.

Acknowledgements

We thank Alp Yurtsever and Tao Lin for valuable discussions and the reviewers for their feedback. This project was supported by SNSF grant 200021_175796, as well as a Google Focused Research Award.

Unreliable local computation Part II

4 SCAFFOLD: Stochastic Controlled Averaging for Federated Learning

4.1 Preface

Contribution and sources. This chapter reproduces (Karimireddy et al., 2020b). The theory, experiments, and writing were carried out mostly by the author. Satyen Kale provided crucial help in the running of the experiments. Detailed individual contributions:

SPK (author): Conceptualization, Methodology, Formal analysis, Software, Writing – original draft preparation

Satyen Kale: Supervision, Administration, Writing – review and editing, Running Experiments

Mehryar Mohri: Supervision, Administration, Writing – review and editing

Sashank Reddi: Supervision, Writing – review and editing

Sebastian Stich: Writing – review and editing

Ananda Theertha Suresh: Writing – review and editing .

Summary. Increasing concerns about scalability and data privacy have lead to the popularity of federated learning where training is performed directly on the edge devices without any transmission of data. However, this comes with very high latency and unreliability—which cannot be alleviated by communication compression. Instead, popular methods (such as FEDAVG) use multiple local updates between communication rounds on the available devices to compensate for the high latencies. However, in spite of recent research efforts, its performance is not well understood.

Using the framework of stochastic optimization, we fully characterize the convergence of FEDAVG and prove that it suffers from ‘client-drift’ when the data is heterogeneous (non-iid), resulting in unstable and slow convergence. As a solution, we propose a new algorithm (SCAFFOLD) which uses control variates (variance reduction) to correct for the ‘client-drift’ in its local updates. We show (theoretically and empirically) that SCAFFOLD is unaffected by heterogeneity or unavailability of clients.

The concept of client drift has since become an accepted phenomenon, with SCAFFOLD inspiring a large number of follow up solutions (cf. (Wang et al., 2021) for a survey).

4.2 Introduction

Federated learning has emerged as an important paradigm in modern large-scale machine learning. Unlike in traditional centralized learning where models are trained using large datasets stored in a central server (Dean et al., 2012; Iandola et al., 2016; Goyal et al., 2017), in federated learning, the training data remains distributed over a large number of clients, which may be phones, network sensors, hospitals, or alternative local information sources (Konečný et al., 2016; McMahan et al., 2017; Mohri et al., 2019; Kairouz et al., 2019). A centralized model (referred to as server model) is then trained without ever transmitting client data over the network, thereby ensuring a basic level of privacy. In this work, we investigate stochastic optimization algorithms for federated learning.

The key challenges for federated optimization are 1) dealing with unreliable and relatively slow network connections between the server and the clients, 2) only a small subset of clients being available for training at a given time, and 3) large heterogeneity (non-iid-ness) in the data present on the different clients (Konečný et al., 2016). The most popular algorithm for this setting is FEDAVG (McMahan et al., 2017). FEDAVG tackles the communication bottleneck by performing multiple local updates on the available clients before communicating to the server. While it has shown success in certain applications, its performance on heterogeneous data is still an active area of research (Li et al., 2018b; Yu et al., 2019b; Li et al., 2019c; Haddadpour and Mahdavi, 2019; Khaled et al., 2020). We prove that indeed such heterogeneity has a large effect on FEDAVG—it introduces a *drift* in the updates of each client resulting in slow and unstable convergence. Further, we show that this client-drift persists even if full batch gradients are used and all clients participate throughout the training.

As a solution, we propose a new Stochastic Controlled Averaging algorithm (SCAFFOLD) which tries to correct for this client-drift. Intuitively, SCAFFOLD estimates the update direction for the server model (\mathbf{c}) and the update direction for each client \mathbf{c}_i .¹ The difference ($\mathbf{c} - \mathbf{c}_i$) is then an estimate of the client-drift which is used to correct the local update. This strategy successfully overcomes heterogeneity and converges in significantly fewer rounds of communication. Alternatively, one can see heterogeneity as introducing ‘client-variance’ in the updates across the different clients and SCAFFOLD then performs ‘client-variance reduction’ (Schmidt et al., 2017; Johnson and Zhang, 2013; Defazio et al., 2014). We use this viewpoint to show that SCAFFOLD is relatively unaffected by client sampling.

Finally, while accommodating heterogeneity is important, it is equally important that a method can take advantage of similarities in the client data. We prove that SCAFFOLD indeed has such a property, requiring fewer rounds of communication when the clients are more similar.

Contributions. We summarize our main results below.

- We derive tighter convergence rates for FEDAVG than previously known for convex and

¹We refer to these estimates as *control variates* and the resulting correction technique as stochastic controlled averaging.

Table 4.1 – Summary of notation used in the paper

$N, S,$ and i	total num., sampled num., and index of clients
R, r	number, index of communication rounds
K, k	number, index of local update steps
\mathbf{x}^r	aggregated server model after round r
$\mathbf{y}_{i,k}^r$	i th client's model in round r and step k
$\mathbf{c}^r, \mathbf{c}_i^r$	control variate of server, i th client after round r

non-convex functions with client sampling and heterogeneous data.

- We give matching lower bounds to prove that even with no client sampling and full batch gradients, FEDAVG can be slower than SGD due to client-drift.
- We propose a new Stochastic Controlled Averaging algorithm (SCAFFOLD) which corrects for this client-drift. We prove that SCAFFOLD is at least as fast as SGD and converges for arbitrarily heterogeneous data.
- We show SCAFFOLD can additionally take advantage of similarity between the clients to further reduce the communication required, proving the advantage of taking local steps over large-batch SGD for the first time.
- We prove that SCAFFOLD is relatively unaffected by the client sampling obtaining variance reduced rates, making it especially suitable for federated learning.

Finally, we confirm our theoretical results on simulated and real datasets (extended MNIST by [Cohen et al. \(2017\)](#)).

Related work. For identical clients, FEDAVG coincides with parallel SGD analyzed by ([Zinkevich et al., 2010](#)) who proved asymptotic convergence. [Stich \(2019a\)](#) and, more recently [Stich and Karimireddy \(2019\)](#); [Patel and Dieuleveut \(2019\)](#); [Khaled et al. \(2020\)](#), gave a sharper analysis of the same method, under the name of local SGD, also for identical functions. However, there still remains a gap between their upper bounds and the lower bound of [Woodworth et al. \(2018\)](#). The analysis of FEDAVG for heterogeneous clients is more delicate due to the afore-mentioned client-drift, first empirically observed by [Zhao et al. \(2018\)](#). Several analyses bound this drift by assuming bounded gradients ([Wang et al., 2019](#); [Yu et al., 2019b](#)), or view it as additional noise ([Khaled et al., 2020](#)), or assume that the client optima are ϵ -close ([Li et al., 2018b](#); [Haddadpour and Mahdavi, 2019](#)). In a concurrent work, ([Liang et al., 2019](#)) propose to use variance reduction to deal with client heterogeneity but still show rates slower than SGD and do not support client sampling. Our method SCAFFOLD can also be seen as an improved version of the distributed optimization algorithm DANE by ([Shamir et al., 2014](#)), where a fixed number of (stochastic) gradient steps are used in place of a proximal point update. A more in-depth discussion of related work is given in Appendix 12.1. We summarize the complexities of different methods for heterogeneous clients in Table 4.2.

Chapter 4. SCAFFOLD: Stochastic Controlled Averaging for Federated Learning

Table 4.2 – Number of communication rounds required to reach ϵ accuracy for μ strongly convex and non-convex functions (log factors are ignored). Set $\mu = \epsilon$ for general convex rates. (G, B) bounds gradient dissimilarity (A1), and δ bounds Hessian dissimilarity (A2). Our rates for FEDAVG are more general and tighter than others, even matching the lower bound. However, SGD is still faster ($B \geq 1$). SCAFFOLD does not require any assumptions, is faster than SGD, and is robust to client sampling. Further, when clients become more similar (small δ), SCAFFOLD converges even faster.

Method	Strongly convex	Non-convex	Sampling	Assumptions
SGD (large batch)	$\frac{\sigma^2}{\mu N K \epsilon} + \frac{1}{\mu}$	$\frac{\sigma^2}{N K \epsilon^2} + \frac{1}{\epsilon}$	×	–
FedAvg				
(Li et al., 2019c)	$\frac{\sigma^2}{\mu^2 N K \epsilon} + \frac{G^2 K}{\mu^2 \epsilon}$	–	×	$(G, 0)$ -BGD
(Yu et al., 2019b)	–	$\frac{\sigma^2}{N K \epsilon^2} + \frac{G^2 N K}{\epsilon}$	×	$(G, 0)$ -BGD
(Khaled et al., 2020)	$\frac{\sigma^2 + G^2}{\mu N K \epsilon} + \frac{\sigma + G}{\mu \sqrt{\epsilon}} + \frac{N B^2}{\mu}$	–	×	(G, B) -BGD
Ours (Thm. VI) ¹	$\frac{M^2}{\mu S K \epsilon} + \frac{G}{\mu \sqrt{\epsilon}} + \frac{B^2}{\mu}$	$\frac{M^2}{S K \epsilon^2} + \frac{G}{\epsilon^{3/2}} + \frac{B^2}{\epsilon}$	✓	(G, B) -BGD
Lower-bound (Thm. VII)	$\Omega(\frac{\sigma^2}{\mu N K \epsilon} + \frac{G}{\sqrt{\mu \epsilon}})$?	×	$(G, 1)$ -BGD
FedProx (Li et al., 2018b) ²	$\frac{B^2}{\mu}$	$\frac{B^2}{\epsilon}$ (weakly convex)	✓	$\sigma = 0, (0, B)$ -BGD
DANE (Shamir et al., 2014) ^{2,3}	$\frac{\delta^2}{\mu^2}$	–	×	$\sigma = 0, \delta$ -BHD
VRL-SGD (Liang et al., 2019)	–	$\frac{N \sigma^2}{K \epsilon^2} + \frac{N}{\epsilon}$	×	–
SCAFFOLD				
Theorem VIII	$\frac{\sigma^2}{\mu S K \epsilon} + \frac{1}{\mu} + \frac{N}{S}$	$\frac{\sigma^2}{S K \epsilon^2} + \frac{1}{\epsilon} (\frac{N}{S})^{\frac{2}{3}}$	✓	–
Theorem IX ³	$\frac{\sigma^2}{\mu N K \epsilon} + \frac{1}{\mu K} + \frac{\delta}{\mu}$	$\frac{\sigma^2}{N K \epsilon^2} + \frac{1}{K \epsilon} + \frac{\delta}{\epsilon}$	×	δ -BHD

¹ $M^2 := \sigma^2 + K(1 - \frac{S}{N})G^2$. Note that $\frac{M^2}{S} = \frac{\sigma^2}{N}$ when no sampling ($S = N$).

² proximal point method i.e. $K \gg 1$.

³ proved only for quadratic functions.

4.3 Setup

We formalize the problem as minimizing a sum of stochastic functions, with only access to stochastic samples:

$$\min_{\mathbf{x} \in \mathbb{R}^d} \left\{ f(\mathbf{x}) := \frac{1}{N} \sum_{i=1}^N (f_i(\mathbf{x}) := \mathbb{E}_{\zeta_i} [f_i(\mathbf{x}; \zeta_i)]) \right\}.$$

The functions f_i represents the loss function on client i . All our results can be easily extended to the weighted case.

We assume that f is bounded from below by f^* and f_i is β -smooth. Further, we assume $g_i(\mathbf{x}) := \nabla f_i(\mathbf{x}; \zeta_i)$ is an unbiased stochastic gradient of f_i with variance bounded by σ^2 . For some results, we assume $\mu \geq 0$ (strong) convexity. Note that σ only bounds the variance *within* clients. We also define two non-standard terminology below.

(A1) (G, B) -BGD or bounded gradient dissimilarity: there exist constants $G \geq 0$ and $B \geq 1$ such that

$$\frac{1}{N} \sum_{i=1}^N \|\nabla f_i(\mathbf{x})\|^2 \leq G^2 + B^2 \|\nabla f(\mathbf{x})\|^2, \forall \mathbf{x}.$$

If $\{f_i\}$ are convex, we can relax the assumption to

$$\frac{1}{N} \sum_{i=1}^N \|\nabla f_i(\mathbf{x})\|^2 \leq G^2 + 2\beta B^2 (f(\mathbf{x}) - f^*), \forall \mathbf{x}.$$

(A2) δ -BHD or bounded Hessian dissimilarity:

$$\|\nabla^2 f_i(\mathbf{x}) - \nabla^2 f(\mathbf{x})\| \leq \delta, \forall \mathbf{x}.$$

Further, f_i is δ -weakly convex i.e. $\nabla^2 f_i(\mathbf{x}) \succeq -\delta I$.

The assumptions A1 and A2 are orthogonal—it is possible to have $G = 0$ and $\delta = 2\beta$, or $\delta = 0$ but $G \gg 1$.

4.4 Convergence of FedAvg

In this section we review FEDAVG and improve its convergence analysis by deriving tighter rates than known before. The scheme consists of two main parts: local updates to the model (4.1), and aggregating the client updates to update the server model (4.2). In each round, a subset of clients $\mathcal{S} \subseteq [N]$ are sampled uniformly. Each of these clients $i \in \mathcal{S}$ copies the current sever model $\mathbf{y}_i = \mathbf{x}$ and performs K local updates of the form:

$$\mathbf{y}_i \leftarrow \mathbf{y}_i - \eta_l g_i(\mathbf{y}_i). \quad (4.1)$$

Here η_l is the local step-size. Then the clients' updates $(\mathbf{y}_i - \mathbf{x})$ are aggregated to form the new server model using a global step-size η_g as:

$$\mathbf{x} \leftarrow \mathbf{x} + \frac{\eta_g}{|\mathcal{S}|} \sum_{i \in \mathcal{S}} (\mathbf{y}_i - \mathbf{x}). \quad (4.2)$$

4.4.1 Rate of convergence

We now state our novel convergence results for functions with bounded dissimilarity (proofs in Appendix 12.4.2).

Theorem VI. *For β -smooth functions $\{f_i\}$ which satisfy (A1), the output of FEDAVG has expected error smaller than ϵ in each of the below three cases for some values of η_l and η_g , with the following bound on R*

- μ **Strongly convex:**

$$R = \tilde{\mathcal{O}} \left(\frac{\sigma^2}{\mu K S \epsilon} + \left(1 - \frac{S}{N}\right) \frac{G^2}{\mu S \epsilon} + \frac{\sqrt{\beta} G}{\mu \sqrt{\epsilon}} + \frac{B^2 \beta}{\mu} \right),$$

• **General convex:**

$$R = \mathcal{O}\left(\frac{\sigma^2 D^2}{KS\epsilon^2} + \left(1 - \frac{S}{N}\right)\frac{G^2 D^2}{S\epsilon^2} + \frac{\sqrt{\beta}G}{\epsilon^{\frac{3}{2}}} + \frac{B^2 \beta D^2}{\epsilon}\right),$$

• **Non-convex:**

$$R = \mathcal{O}\left(\frac{\beta\sigma^2 F}{KS\epsilon^2} + \left(1 - \frac{S}{N}\right)\frac{G^2 F}{S\epsilon^2} + \frac{\sqrt{\beta}G}{\epsilon^{\frac{3}{2}}} + \frac{B^2 \beta F}{\epsilon}\right),$$

where $D := \|\mathbf{x}^0 - \mathbf{x}^*\|^2$ and $F := f(\mathbf{x}^0) - f^*$.

The exact values of η_l and η_g decrease with the number of rounds R and can be found in the proofs in the Appendix. It is illuminating to compare our rates with those of the simpler iid. case i.e. with $G = 0$ and $B = 1$. Our strongly-convex rates become $\frac{\sigma^2}{\mu SK\epsilon} + \frac{1}{\mu}$. In comparison, the best previously known rate for this case was by [Stich and Karimireddy \(2019\)](#) who show a rate of $\frac{\sigma^2}{\mu SK\epsilon} + \frac{S}{\mu}$. The main source of improvement in the rates came from the use of *two separate step-sizes* (η_l and η_g). By having a larger global step-size η_g , we can use a smaller local step-size η_l thereby reducing the client-drift while still ensuring progress. However, even our improved rates do not match the lower-bound for the identical case of $\frac{\sigma^2}{\mu SK\epsilon} + \frac{1}{K\mu}$ ([Woodworth et al., 2018](#)). We bridge this gap for quadratic functions in Section 4.7.

We now compare FEDAVG to two other algorithms FedProx by ([Li et al., 2018b](#)) (aka EASGD by ([Zhang et al., 2015](#))) and to SGD. Suppose that $G = 0$ and $\sigma = 0$ i.e. we use full batch gradients and all clients have very similar optima. In such a case, FEDAVG has a complexity of $\frac{B^2}{\mu}$ which is identical to that of FedProx ([Li et al., 2018b](#)). Thus, FedProx does not have any theoretical advantage.

Next, suppose that all clients participate (no sampling) with $S = N$ and there is no variance $\sigma = 0$. Then, the above for strongly-convex case simplifies to $\frac{G}{\mu\sqrt{\epsilon}} + \frac{B^2}{\mu}$. In comparison, extending the proof of ([Khaled et al., 2020](#)) using our techniques gives a worse dependence on G of $\frac{G^2}{\mu KN\epsilon} + \frac{G}{\mu\sqrt{\epsilon}}$. Similarly, for the non-convex case, our rates are tighter and have better dependence on G than ([Yu et al., 2019b](#)). However, simply running SGD in this setting would give a communication complexity of $\frac{\beta}{\mu}$ which is faster, and independent of similarity assumptions. In the next section we examine the necessity of such similarity assumptions.

4.4.2 Lower bounding the effect of heterogeneity

We now show that when the functions $\{f_i\}$ are distinct, the local updates of FEDAVG on each client experiences *drift* thereby slowing down convergence. We show that the amount of this client drift, and hence the slowdown in the rate of convergence, is exactly determined by the gradient dissimilarity parameter G in (A1).

We now examine the mechanism by which the client-drift arises (see Fig. 4.1). Let \mathbf{x}^* be the global optimum of $f(\mathbf{x})$ and \mathbf{x}_i^* be the optimum of each client's loss function $f_i(\mathbf{x})$. In the case of heterogeneous data, it is quite likely that each of these \mathbf{x}_i^* is far away from the other, and from the global optimum \mathbf{x}^* . Even if all the clients start from the same point \mathbf{x} , each of the \mathbf{y}_i

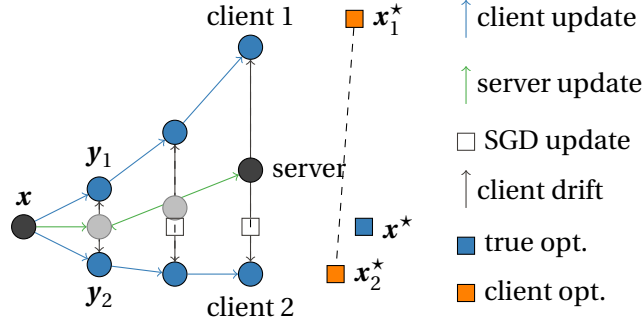


Figure 4.1 – Client-drift in FEDAVG is illustrated for 2 clients with 3 local steps ($N = 2$, $K = 3$). The local updates y_i (in blue) move towards the individual client optima x_i^* (orange square). The server updates (in red) move towards $\frac{1}{N} \sum_i x_i^*$ instead of to the true optimum x^* (black square).

will move towards their client optimum x_i^* . This means that the average of the client updates (which is the server update) moves towards $\frac{1}{N} \sum_{i=1}^N x_i^*$. This difference between $\frac{1}{N} \sum_{i=1}^N x_i^*$ and the true optimum x^* is exactly the cause of client-drift. To counter this drift, FEDAVG is forced to use much smaller step-sizes which in turn hurts convergence. We can formalize this argument to prove a lower-bound (see Appendix 12.4.4 for proof).

Theorem VII. *For any positive constants G and μ , there exist μ -strongly convex functions satisfying A1 for which FEDAVG with $K \geq 2$, $\sigma = 0$ and $N = S$ has an error*

$$f(x^r) - f(x^*) \geq \Omega\left(\frac{G^2}{\mu R^2}\right).$$

This implies that the $\frac{G}{\sqrt{\epsilon}}$ term is unavoidable even if there is no stochasticity. Further, because FEDAVG uses RKN stochastic gradients, we also have the statistical lower-bound of $\frac{\sigma^2}{\mu K N \epsilon}$. Together, these lower bounds prove that the rate derived in Theorem VI is nearly optimal (up to dependence on μ). In the next section, we introduce a new method SCAFFOLD to mitigate this client-drift.

4.5 SCAFFOLD algorithm

In this section we first describe SCAFFOLD and then discuss how it solves the problem of client-drift.

Method. SCAFFOLD has three main steps: local updates to the client model (4.3), local updates to the client control variate (4.4), and aggregating the updates (4.5). We describe each in more detail.

Along with the server model x , SCAFFOLD maintains a state for each client (client control variate c_i) and for the server (server control variate c). These are initialized to ensure that

Algorithm 5 SCAFFOLD: Stochastic Controlled Averaging for federated learning

```

1: server input: initial  $\mathbf{x}$  and  $\mathbf{c}$ , and global step-size  $\eta_g$ 
2: client  $i$ 's input:  $\mathbf{c}_i$ , and local step-size  $\eta_l$ 
3: for each round  $r = 1, \dots, R$  do
4:   sample clients  $\mathcal{S} \subseteq \{1, \dots, N\}$ 
5:   communicate  $(\mathbf{x}, \mathbf{c})$  to all clients  $i \in \mathcal{S}$ 
6:   for each client  $i \in \mathcal{S}$  in parallel do
7:     initialize local model  $\mathbf{y}_i \leftarrow \mathbf{x}$ 
8:     for  $k = 1, \dots, K$  do
9:       compute mini-batch gradient  $g_i(\mathbf{y}_i)$ 
10:       $\mathbf{y}_i \leftarrow \mathbf{y}_i - \eta_l (g_i(\mathbf{y}_i) - \mathbf{c}_i + \mathbf{c})$ 
11:    end for
12:     $\mathbf{c}_i^+ \leftarrow$  (i)  $g_i(\mathbf{x})$ , or (ii)  $\mathbf{c}_i - \mathbf{c} + \frac{1}{K\eta_l}(\mathbf{x} - \mathbf{y}_i)$ 
13:    communicate  $(\Delta \mathbf{y}_i, \Delta \mathbf{c}_i) \leftarrow (\mathbf{y}_i - \mathbf{x}, \mathbf{c}_i^+ - \mathbf{c}_i)$ 
14:     $\mathbf{c}_i \leftarrow \mathbf{c}_i^+$ 
15:  end for
16:   $(\Delta \mathbf{x}, \Delta \mathbf{c}) \leftarrow \frac{1}{|\mathcal{S}|} \sum_{i \in \mathcal{S}} (\Delta \mathbf{y}_i, \Delta \mathbf{c}_i)$ 
17:   $\mathbf{x} \leftarrow \mathbf{x} + \eta_g \Delta \mathbf{x}$  and  $\mathbf{c} \leftarrow \mathbf{c} + \frac{|\mathcal{S}|}{N} \Delta \mathbf{c}$ 
18: end for

```

$\mathbf{c} = \frac{1}{N} \sum \mathbf{c}_i$ and can safely all be initialized to 0. In each round of communication, the server parameters (\mathbf{x}, \mathbf{c}) are communicated to the participating clients $\mathcal{S} \subset [N]$. Each participating client $i \in \mathcal{S}$ initializes its local model with the server model $\mathbf{y}_i \leftarrow \mathbf{x}$. Then it makes a pass over its local data performing K updates of the form:

$$\mathbf{y}_i \leftarrow \mathbf{y}_i - \eta_l (g_i(\mathbf{y}_i) + \mathbf{c} - \mathbf{c}_i). \quad (4.3)$$

Then, the local control variate \mathbf{c}_i is also updated. For this, we provide two options:

$$\mathbf{c}_i^+ \leftarrow \begin{cases} \text{Option I.} & g_i(\mathbf{x}), \text{ or} \\ \text{Option II.} & \mathbf{c}_i - \mathbf{c} + \frac{1}{K\eta_l}(\mathbf{x} - \mathbf{y}_i). \end{cases} \quad (4.4)$$

Option I involves making an additional pass over the local data to compute the gradient at the server model \mathbf{x} . Option II instead re-uses the previously computed gradients to update the control variate. Option I can be more stable than II depending on the application, but II is cheaper to compute and usually suffices (all our experiments use Option II). The client updates are then aggregated and used to update the server parameters:

$$\begin{aligned} \mathbf{x} &\leftarrow \mathbf{x} + \frac{\eta_g}{|\mathcal{S}|} \sum_{i \in \mathcal{S}} (\mathbf{y}_i - \mathbf{x}), \\ \mathbf{c} &\leftarrow \mathbf{c} + \frac{1}{N} \sum_{i \in \mathcal{S}} (\mathbf{c}_i^+ - \mathbf{c}_i). \end{aligned} \quad (4.5)$$

This finishes one round of communication. Note that the clients in SCAFFOLD are *stateful* and retain the value of \mathbf{c}_i across multiple rounds. Further, if \mathbf{c}_i is always set to 0, then

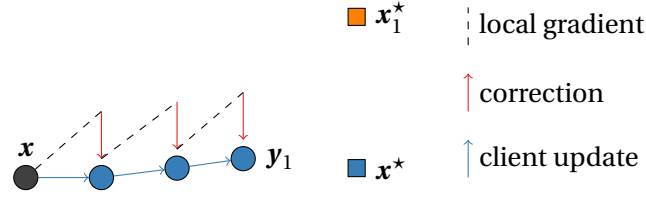


Figure 4.2 – Update steps of SCAFFOLD on a single client. The local gradient (dashed black) points to \mathbf{x}_1^* (orange square), but the correction term $(\mathbf{c} - \mathbf{c}_i)$ (in red) ensures the update moves towards the true optimum \mathbf{x}^* (black square).

SCAFFOLD becomes equivalent to FEDAVG. The full details are summarized in Algorithm 5.

Usefulness of control variates.

If communication cost was not a concern, the ideal update on client i would be

$$\mathbf{y}_i \leftarrow \mathbf{y}_i + \frac{1}{N} \sum_j \mathbf{g}_j(\mathbf{y}_i). \quad (4.6)$$

Such an update essentially computes an unbiased gradient of f and hence becomes equivalent to running FEDAVG in the iid case (which has excellent performance). Unfortunately such an update requires communicating with all clients for every update step. SCAFFOLD instead uses control variates such that

$$\mathbf{c}_j \approx \mathbf{g}_j(\mathbf{y}_i) \text{ and } \mathbf{c} \approx \frac{1}{N} \sum_j \mathbf{g}_j(\mathbf{y}_i).$$

Then, SCAFFOLD (4.3) mimics the ideal update (4.6) with

$$(\mathbf{g}_i(\mathbf{y}_i) - \mathbf{c}_i + \mathbf{c}) \approx \frac{1}{N} \sum_j \mathbf{g}_j(\mathbf{y}_i).$$

Thus, the local updates of SCAFFOLD remain synchronized and converge for arbitrarily heterogeneous clients.

4.6 Convergence of SCAFFOLD

We state the rate of SCAFFOLD without making any assumption on the similarity between the functions. See Appendix 12.5 for the full proof.

Theorem VIII. *For any β -smooth functions $\{f_i\}$, the output of SCAFFOLD has expected error smaller than ϵ for in each of the below three cases for some values of η_l and η_g , with the following bound on R*

- μ **Strongly convex**:

$$R = \tilde{\mathcal{O}}\left(\frac{\sigma^2}{\mu K S \epsilon} + \frac{\beta}{\mu} + \frac{N}{S}\right),$$

- **General convex:**

$$R = \tilde{\mathcal{O}}\left(\frac{\sigma^2 D^2}{K S \epsilon^2} + \frac{\beta D^2}{\epsilon} + \frac{N F}{S}\right),$$

- **Non-convex:**

$$R = \mathcal{O}\left(\frac{\beta \sigma^2 F}{K S \epsilon^2} + \left(\frac{N}{S}\right)^{\frac{2}{3}} \frac{\beta F}{\epsilon}\right),$$

where $D := \|\mathbf{x}^0 - \mathbf{x}^\star\|^2$ and $F := f(\mathbf{x}^0) - f^\star$.

The exact values of η_l and η_g decrease with the number of rounds R and can be found in the proofs in the Appendix. Let us first examine the rates without client sampling ($S = N$). For the strongly convex case, the number of rounds becomes $\frac{\sigma^2}{\mu N K \epsilon} + \frac{1}{\mu}$. This rate holds for arbitrarily heterogeneous clients unlike Theorem VI and further matches that of SGD with K times larger batch-size, proving that SCAFFOLD is at least as fast as SGD. These rates also match known lower-bounds for distributed optimization (Arjevani and Shamir, 2015) (up to acceleration) and are unimprovable in general. However in certain cases SCAFFOLD is provably faster than SGD. We show this fact in Section 4.7.

Now let $\sigma = 0$. Then our rates in the strongly-convex case are $\frac{1}{\mu} + \frac{N}{S}$ and $\left(\frac{N}{S}\right)^{\frac{2}{3}} \frac{1}{\epsilon}$ in the non-convex case. These exactly match the rates of SAGA (Defazio et al., 2014; Reddi et al., 2016c). In fact, when $\sigma = 0$, $K = 1$ and $S = 1$, the update of SCAFFOLD with option I reduces to SAGA where in each round consists of sampling one client f_i . Thus SCAFFOLD can be seen as an extension of variance reduction techniques for federated learning, and one could similarly extend SARA (Nguyen et al., 2017), SPIDER (Fang et al., 2018), etc. Note that standard SGD with client sampling is provably slower and converges at a sub-linear rate even with $\sigma = 0$.

Proof sketch. For simplicity, assume that $\sigma = 0$ and consider the ideal update of (4.6) which uses the full gradient $\nabla f(\mathbf{y})$ every step. Clearly, this would converge at a linear rate even with $S = 1$. FEDAVG would instead use an update $\nabla f_i(\mathbf{y})$. The difference between the ideal update (4.6) and the FEDAVG update (4.1) is $\|\nabla f_i(\mathbf{y}) - \nabla f(\mathbf{y})\|$. We need a bound on the gradient-dissimilarity as in (A1) to bound this error. SCAFFOLD instead uses the update $\nabla f_i(\mathbf{y}) - \mathbf{c}_i + \mathbf{c}$, and the difference from ideal update becomes

$$\sum_i \|\nabla f_i(\mathbf{y}) - \mathbf{c}_i + \mathbf{c} - \nabla f(\mathbf{y})\|^2 \leq \sum_i \|\mathbf{c}_i - \nabla f_i(\mathbf{y})\|^2.$$

Thus, the error is independent of how similar or dissimilar the functions f_i are, and instead only depends on the quality of our approximation $\mathbf{c}_i \approx \nabla f_i(\mathbf{y})$. Since f_i is smooth, we can expect that the gradient $\nabla f_i(\mathbf{y})$ does not change too fast and hence is easy to approximate. Appendix 12.5 translates this intuition into a formal proof.

4.7 Usefulness of local steps

In this section we investigate when and why taking local steps might be useful over simply computing a large-batch gradient in distributed optimization. We will show that when the functions across the clients share some similarity, local steps can take advantage of this and converge faster. For this we consider quadratic functions and express their similarity with the δ parameter introduced in (A2).

Theorem IX. *For any β -smooth quadratic functions $\{f_i\}$ with δ bounded Hessian dissimilarity (A2), the output of SCAFFOLD with $S = N$ (no sampling) has error smaller than ϵ in each of the following two cases with $\eta_g = 1$, some value of η_l , and R satisfying*

- **Strongly convex:**

$$R = \tilde{\mathcal{O}}\left(\frac{\beta\sigma^2}{\mu K N \epsilon} + \frac{\beta + \delta K}{\mu K}\right),$$

- **Weakly convex:**

$$R = \mathcal{O}\left(\frac{\beta\sigma^2 F}{K N \epsilon^2} + \frac{(\beta + \delta K)F}{K \epsilon}\right),$$

where we define $F := (f(\mathbf{x}^0) - f^*)$.

Here again the exact value of η_l decreases with the number of rounds R and can be found in the proofs in the Appendix. When $\sigma = 0$ and K is large, the complexity of SCAFFOLD becomes $\frac{\delta}{\mu}$. In contrast DANE, which being a proximal point method also uses large K , requires $(\frac{\delta}{\mu})^2$ rounds (Shamir et al., 2014) which is significantly slower, or needs an additional backtracking-line search to match the rates of SCAFFOLD (Yuan and Li, 2019). Further, Theorem IX is the first result to demonstrate improvement due to similarity for non-convex functions as far as we are aware.

Suppose that $\{f_i\}$ are identical. Recall that δ in (A2) measures the Hessian dissimilarity between functions and so $\delta = 0$ for this case. Then Theorem IX shows that the complexity of SCAFFOLD is $\frac{\sigma^2}{\mu K N \epsilon} + \frac{1}{\mu K}$ which (up to acceleration) matches the i.i.d. lower bound of (Woodworth et al., 2018). In contrast, SGD with K times larger batch-size would require $\frac{\sigma^2}{\mu K N \epsilon} + \frac{1}{\mu}$ (note the absence of K in the second term). Thus, for identical functions, SCAFFOLD (and in fact even FEDAVG) improves linearly with increasing number of local steps. In the other extreme, if the functions are arbitrarily different, we may have $\delta = 2\beta$. In this case, the complexity of SCAFFOLD and large-batch SGD match the lower bound of Arjevani and Shamir (2015) for the heterogeneous case.

The above insights can be generalized to when the functions are only somewhat similar. If the Hessians are δ -close and $\sigma = 0$, then the complexity is $\frac{\beta + \delta K}{\mu K}$. This bound implies that the optimum number of local steps one should use is $K = \frac{\beta}{\delta}$. Picking a smaller K increases the communication required whereas increasing it further would only waste computational resources. While this result is intuitive—if the functions are more ‘similar’, local steps are

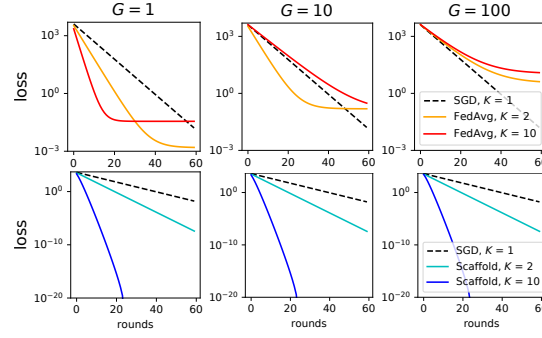


Figure 4.3 – SGD (dashed black), FedAvg (above), and SCAFFOLD (below) on simulated data. FedAvg gets worse as local steps increases with $K = 10$ (red) worse than $K = 2$ (orange). It also gets slower as gradient-dissimilarity (G) increases (to the right). SCAFFOLD significantly improves with more local steps, with $K = 10$ (blue) faster than $K = 2$ (light blue) and SGD. Its performance is identical as we vary heterogeneity (G).

more useful—Theorem IX shows that it is the similarity of the *Hessians* which matters. This is surprising since the Hessians of $\{f_i\}$ may be identical even if their individual optima \mathbf{x}_i^* are arbitrarily far away from each other and the gradient-dissimilarity (A1) is unbounded.

Proof sketch. Consider a simplified SCAFFOLD update with $\sigma = 0$ and no sampling ($S = N$):

$$\mathbf{y}_i = \mathbf{y}_i - \eta(\nabla f_i(\mathbf{y}_i) + \nabla f(\mathbf{x}) - \nabla f_i(\mathbf{x})).$$

We would ideally want to perform the update $\mathbf{y}_i = \mathbf{y}_i - \eta \nabla f(\mathbf{y}_i)$ using the full gradient $\nabla f(\mathbf{y}_i)$. We reinterpret the correction term of SCAFFOLD ($\mathbf{c} - \mathbf{c}_i$) as performing the following first order correction to the local gradient $\nabla f_i(\mathbf{y}_i)$ to make it closer to the full gradient $\nabla f(\mathbf{y}_i)$:

$$\begin{aligned} \underbrace{\nabla f_i(\mathbf{y}_i) - \nabla f_i(\mathbf{x})}_{\approx \nabla^2 f_i(\mathbf{x})(\mathbf{y}_i - \mathbf{x})} + \underbrace{\nabla f(\mathbf{x})}_{\approx \nabla f(\mathbf{y}_i) + \nabla^2 f(\mathbf{x})(\mathbf{x} - \mathbf{y}_i)} \\ \approx \nabla f(\mathbf{y}_i) + (\nabla^2 f_i(\mathbf{x}) - \nabla^2 f(\mathbf{x}))(\mathbf{y}_i - \mathbf{x}) \\ \approx \nabla f(\mathbf{y}_i) + \delta(\mathbf{y}_i - \mathbf{x}) \end{aligned}$$

Thus the SCAFFOLD update approximates the ideal update up to an error δ . This intuition is proved formally for quadratic functions in Appendix 12.6. Generalizing these results to other functions is a challenging open problem.

4.8 Experiments

We run experiments on both simulated and real datasets to confirm our theory. Our main findings are i) SCAFFOLD consistently outperforms SGD and FEDAVG across all parameter regimes, and ii) the benefit (or harm) of local steps depends on both the algorithm and the similarity of the clients data.

4.8.1 Setup

Our simulated experiments uses $N = 2$ quadratic functions based on our lower-bounds in Theorem VII. We use full-batch gradients ($\sigma = 0$) and no client sampling. Our real world experiments run logistic regression (convex) and 2 layer fully connected network (non-convex) on the EMNIST (Cohen et al., 2017). We divide this dataset among $N = 100$ clients as follows: for $s\%$ similar data we allocate to each client $s\%$ i.i.d. data and the remaining $(100 - s)\%$ by sorting according to label (cf. Hsu et al. (2019)).

We consider four algorithms: SGD, FEDAVG, SCAFFOLD and FEDPROX with SGD as the local solver (Li et al., 2018b). On each client SGD uses the full local data to compute a single update, whereas the other algorithms take 5 steps per epoch (batch size is 0.2 of local data). We always use global step-size $\eta_g = 1$ and tune the local step-size η_l individually for each algorithm. SCAFFOLD uses option II (no extra gradient computations) and FEDPROX has fixed regularization = 1 to keep comparison fair. Additional tuning of the regularization parameter may sometimes yield improved empirical performance.

4.8.2 Simulated results

The results are summarized in Fig. 4.3. Our simulated data has Hessian difference $\delta = 1$ (A2) and $\beta = 1$. We vary the gradient heterogeneity (A1) as $G \in [1, 10, 100]$. For all valued of G , FEDAVG gets slower as we increase the number of local steps. This is explained by the fact that client-drift increases as we increase the number of local steps, hindering progress. Further, as we increase G , FEDAVG continues to slow down exactly as dictated by Thms. VI and VII. Note that when heterogeneity is small ($G = \beta = 1$), FEDAVG can be competitive with SGD.

SCAFFOLD is consistently faster than SGD, with $K = 2$ being twice as fast and $K = 10$ about 5 times faster. Further, its convergence is completely unaffected by G , confirming our theory in Thm. VIII. The former observation that we do not see linear improvement with K is explained by Thm. IX since we have $\delta > 0$. This sub linear improvement is still significantly faster than both SGD and FEDAVG.








































4.8.3 EMNIST results

We run extensive experiments on the EMNIST dataset to measure the interplay between the algorithm, number of epochs (local updates), number of participating clients, and the client similarity. Table 4.3 measures the benefit (or harm) of using more local steps, Table 4.4 studies the resilience to client sampling, and Table 4.5 reports preliminary results on neural networks. We are mainly concerned with minimizing the number of *communication rounds*. We observe that

SCAFFOLD is consistently the best. Across all range of values tried, we observe that SCAFFOLD outperforms SGD, FEDAVG, and FEDPROX. The latter FEDPROX is always slower than

Chapter 4. SCAFFOLD: Stochastic Controlled Averaging for Federated Learning

Table 4.3 – Communication rounds to reach 0.5 test accuracy for logistic regression on EM-NIST as we vary number of epochs. 1k+ indicates 0.5 accuracy was not reached even after 1k rounds, and similarly an arrowhead indicates that the barplot extends beyond the table. 1 epoch for local update methods corresponds to 5 local steps (0.2 batch size), and 20% of clients are sampled each round. We fix $\mu = 1$ for FEDPROX and use variant (ii) for SCAFFOLD to ensure all methods are comparable. Across all parameters (epochs and similarity), SCAFFOLD is the fastest method. When similarity is 0 (sorted data), FEDAVG consistently gets worse as we increase the number of epochs, quickly becoming slower than SGD. SCAFFOLD initially gets worse and later stabilizes, but is always at least as fast as SGD. As similarity increases (i.e. data is more shuffled), both FEDAVG and SCAFFOLD significantly outperform SGD though SCAFFOLD is still better than FEDAVG. Further, with higher similarity, both methods benefit from increasing number of epochs.

	Epochs	0% similarity (sorted)		10% similarity		100% similarity (i.i.d.)	
		Num. of rounds	Speedup	Num. of rounds	Speedup	Num. of rounds	Speedup
SGD	1	317 	(1×)	365 	(1×)	416 	(1×)
SCAFFOLD	1	77 	(4.1×)	62 	(5.9×)	60 	(6.9×)
	5	152 	(2.1×)	20 	(18.2×)	10 	(41.6×)
	10	286 	(1.1×)	16 	(22.8×)	7 	(59.4×)
	20	266 	(1.2×)	11 	(33.2×)	4 	(104×)
FEDAVG	1	258 	(1.2×)	74 	(4.9×)	83 	(5×)
	5	428 	(0.7×)	34 	(10.7×)	10 	(41.6×)
	10	711 	(0.4×)	25 	(14.6×)	6 	(69.3×)
	20	1k+ 	(< 0.3×)	18 	(20.3×)	4 	(104×)
FEDPROX	1	1k+ 	(< 0.3×)	979 	(0.4×)	459 	(0.9×)
	5	1k+ 	(< 0.3×)	794 	(0.5×)	351 	(1.2×)
	10	1k+ 	(< 0.3×)	894 	(0.4×)	308 	(1.4×)
	20	1k+ 	(< 0.3×)	916 	(0.4×)	351 	(1.2×)

the other local update methods, though in some cases it outperforms SGD. Note that it is possible to improve FEDPROX by carefully tuning the regularization parameter (Li et al., 2018b). FEDAVG is always slower than SCAFFOLD and faster than FEDPROX.

SCAFFOLD > SGD > FedAvg for heterogeneous clients. When similarity is 0%, FEDAVG gets slower with increasing local steps. If we take more than 5 epochs, its performance is worse than SGD's. SCAFFOLD initially worsens as we increase the number of epochs but then flattens. However, its performance is always better than that of SGD, confirming that it can handle heterogeneous data.

SCAFFOLD and FedAvg get faster with more similarity, but not SGD. As similarity of the clients increases, the performance of SGD remains relatively constant. On the other hand, SCAFFOLD and FEDAVG get significantly faster as similarity increases. Further, local steps become much more useful, showing monotonic improvement with the increase in number of epochs. This is because with increasing the i.i.d.ness of the data, both the gradient and Hessian dissimilarity decrease.

Table 4.4 – Communication rounds to reach 0.45 test accuracy for logistic regression on EMNIST as we vary the number of sampled clients. Number of epochs is kept fixed to 5. SCAFFOLD is consistently faster than FEDAVG. As we decrease the number of clients sampled in each round, the increase in number of rounds is sub-linear. This slow-down is better for more similar clients.

	Clients	0% similarity		10% similarity	
SCAFFOLD	20%	143	(1.0×)	9	(1.0×)
	5%	290	(2.0×)	13	(1.4×)
	1%	790	(5.5×)	28	(3.1×)
FEDAVG	20%	179	(1.0×)	12	(1.0×)
	5%	334	(1.9×)	17	(1.4×)
	1%	1k+	(5.6+×)	35	(2.9×)

Table 4.5 – Best test accuracy after 1k rounds with 2-layer fully connected neural network (non-convex) on EMNIST trained with 5 epochs per round (25 steps) for the local methods, and 20% of clients sampled each round. SCAFFOLD has the best accuracy and SGD has the least. SCAFFOLD again outperforms other methods. SGD is unaffected by similarity, whereas the local methods improve with client similarity.

	0% similarity	10% similarity
SGD	0.766	0.764
FEDAVG	0.787	0.828
SCAFFOLD	0.801	0.842

SCAFFOLD is resilient to client sampling. As we decrease the fraction of clients sampled, SCAFFOLD, and FEDAVG only show a sub-linear slow-down. They are more resilient to sampling in the case of higher similarity.

SCAFFOLD outperforms FedAvg on non-convex experiments. We see that SCAFFOLD is better than FEDAVG in terms of final test accuracy reached, though interestingly FEDAVG seems better than SGD even when similarity is 0. However, much more extensive experiments (beyond current scope) are needed before drawing conclusions.

4.9 Conclusion

Our work studied the impact of heterogeneity on the performance of optimization methods for federated learning. Our careful theoretical analysis showed that FEDAVG can be severely hampered by *gradient dissimilarity*, and can be even slower than SGD. We then proposed a new stochastic algorithm (SCAFFOLD) which overcomes gradient dissimilarity using control variates. We demonstrated the effectiveness of SCAFFOLD via strong convergence guarantees and empirical evaluations. Further, we showed that while SCAFFOLD is always at

least as fast as SGD, it can be much faster depending on the *Hessian dissimilarity* in our data. Thus, different algorithms can take advantage of (and are limited by) different notions of dissimilarity. We believe that characterizing and isolating various dissimilarities present in real world data can lead to further new algorithms and significant impact on distributed, federated, and decentralized learning.

Acknowledgments. We thank Filip Hanzely and Jakub Konečný for discussions regarding variance reduction techniques and Blake Woodworth, Virginia Smith and Kumar Kshitij Patel for suggestions which improved the writing.

5 Mime: mimicking centralized algorithms in cross-device federated learning

5.1 Preface

Contribution and sources. This chapter reproduces (Karimireddy et al., 2020a). Ideation, theory, and writing were carried out mostly by SPK. SPK wrote an initial version of the codebase, which was significantly built upon and improved by Ananda Theertha Suresh and Satyen Kale. Their help was crucial in running the large scale experiments presented here, whereas the smaller scale experiments use the initial codebase. SK also polished the manuscript significantly. Detailed individual contributions:

SPK (author): Conceptualization, Methodology, Formal analysis, Software (33%), Writing

Martin Jaggi: Writing – review and editing

Satyen Kale: Supervision, Administration, Writing – review and editing, Software (33%)

Mehryar Mohri: Supervision, Administration, Writing – review and editing

Sashank Reddi: Writing – review and editing

Sebastian Stich: Writing – review and editing

Ananda Theertha Suresh: Supervision, Software (33%), Writing – review and editing .

Summary. While significant research has gone into developing better federated learning (FL) algorithms, most of them (including SCAFFOLD) require storing N (number of clients) copies of the model and make a large number of passes over the clients. This make them impractical for real world setting where N may be $100k+$, and we may never see the same client twice. Further, algorithmic innovations such as momentum and adaptivity are crucial for practical deep learning and need to be incorporated into FL.

In this chapter, we formulate cross-device FL as a stochastic optimization with potentially infinite clients, and propose a general algorithmic framework (MIME) for it. MIME i) mitigates client drift similar to SCAFFOLD, but also ii) adapts an arbitrary centralized optimization algorithm such as momentum and Adam to FL in a principled manner. We theoretically prove that MIME is provably *faster than any centralized method*—the first such result, and also perform a thorough experimental evaluation. Our work is open sourced at github.com/google/fedjax, and is currently being tested for real world deployment at Google.

5.2 Introduction

Federated learning (FL) is an increasingly important large-scale learning framework where the training data remains distributed over a large number of clients, which may be mobile phones or network sensors (Konečný et al., 2016; McMahan et al., 2017; Mohri et al., 2019; Kairouz et al., 2019). A server then orchestrates the clients to train a single model, here referred to as a *server model*, without ever transmitting client data over the network, thereby providing some basic levels of data privacy and security.

Two important settings are distinguished in FL (Kairouz et al., 2019, Table 1): the *cross-device* and the *cross-silo* settings. The cross-silo setting corresponds to a relatively small number of reliable clients, typically organizations, such as medical or financial institutions. In contrast, in the *cross-device* federated learning setting, the number of clients may be extremely large and include, for example, all 3.5 billion active android phones (Holst, 2019). Thus, in that setting, we may never make even a single pass over the entire clients’ data during training. The cross-device setting is further characterized by resource-poor clients communicating over a highly unreliable network. Together, the essential features of this setting give rise to unique challenges not present in the cross-silo setting. In this work, we are interested in the more challenging cross-device setting, for which we will formalize and study stochastic optimization algorithms. Importantly, recent advances in FL optimization, such as SCAFFOLD (Karimireddy et al., 2020b) or FedDyn (Acar et al., 2021), are *not anymore applicable* since they are designed for the cross-silo setting.

The problem. The de facto standard algorithm for the cross-device setting is FEDAVG (McMahan et al., 2017), which performs multiple SGD updates on the available clients before communicating to the server. While this approach can reduce the *frequency* of communication required, performing multiple steps on the same client can lead to ‘over-fitting’ to its atypical local data, a phenomenon known as *client drift* (Karimireddy et al., 2020b). This in turn leads to slower convergence and can, somewhat counter-intuitively, require *larger total communication* (Woodworth et al., 2020a). Despite significant attention received from the optimization community, the communication complexity of heterogeneous cross-device has not improved upon that of simple centralized methods, which take no local steps (aka SERVER-ONLY methods). Furthermore, algorithmic innovations such as momentum (Sutskever et al., 2013; Cutkosky and Orabona, 2019), adaptivity (Kingma and Ba, 2014; Zaheer et al., 2018; Zhang et al., 2019b), and clipping (You et al., 2017, 2019; Zhang et al., 2020) are critical to the success of deep learning applications. The lack of a theoretical understanding of the impact of multiple client steps has also hindered adapting these techniques in a principled manner into the client updates, in order to replace the vanilla SGD update of FEDAVG.

To overcome such deficiencies, we propose a new framework, MIME, that mitigates client drift and can adapt an arbitrary centralized optimization algorithm, e.g. SGD with momentum or Adam, to the federated setting. In each local client update, MIME uses global optimizer state, e.g. momentum or adaptive learning rates, and an SVRG-style correction to

mimic the updates of the centralized algorithm run on i.i.d. data. This optimizer state is computed only at the server level and kept fixed throughout the local steps, thereby avoiding overfitting to the atypical local data of any single client.

Contributions. We summarize our main results below.

- **MIME framework.** We formalize the cross-device federated learning problem, and propose a new framework MIME that can adapt arbitrary centralized algorithms to this setting.
- **Convergence result.** We prove a result showing that MIME successfully reduces client drift. We also prove that the convergence of any generic algorithm in the centralized setting translates convergence of its MIME version in the federated setting.
- **Speed-up over centralized methods.** By carefully tracking the bias introduced due to multiple local steps, we prove that MIME with momentum-based variance reduction (MVR) can beat a lower bound for centralized methods, thus breaking a fundamental barrier. This is the first such result in FL, and also the first general result showing asymptotic speed-up due to local steps.
- **Empirical validation.** We propose a simpler variant, MIMELITE, with an empirical performance similar to MIME. We report the results of thorough experimental analysis demonstrating that both MIME and MIMELITE indeed converge faster than FEDAVG.

Related work.

Analysis of FEDAVG: Much of the recent work in federated learning has focused on analyzing FEDAVG. For identical clients, FEDAVG coincides with parallel SGD, for which (Zinkevich et al., 2010) derived an analysis with asymptotic convergence. Sharper and more refined analyses of the same method, sometimes called local SGD, were provided by (Stich, 2019a), and more recently by (Stich and Karimireddy, 2019), (Patel and Dieuleveut, 2019), (Khaled et al., 2020), and (Woodworth et al., 2020b), for identical functions. Their analysis was extended to heterogeneous clients in (Wang et al., 2019; Yu et al., 2019b; Karimireddy et al., 2020b; Khaled et al., 2020; Koloskova et al., 2020). (Charles and Konečný, 2020) derived a tight characterization of FedAvg with quadratic functions and demonstrated the sensitivity of the algorithm to both client and server step sizes. Matching upper and lower bounds were recently given by (Karimireddy et al., 2020b) and (Woodworth et al., 2020a) for general functions, proving that FEDAVG can be slower than even SGD for heterogeneous data, due to the *client-drift*.

Comparison to SCAFFOLD: For the cross-silo setting where the number of clients is relatively low, (Karimireddy et al., 2020b) proposed the SCAFFOLD algorithm, which uses control-variates (similar to SVRG) to correct for client drift. However, their algorithm crucially relies on *stateful clients* which repeatedly participate in the training process. FedDyn (Acar et al., 2021) reduces the communication requirements, but also requires persistent stateful clients. In contrast, we focus on the cross-device setting where clients may be visited only once during training and where they are *stateless* (and thus SCAFFOLD and FedDyn are inapplicable). This is akin to the difference between the finite-sum (corresponding to cross-silo)

and stochastic (cross-device) settings in traditional centralized optimization (Lei and Jordan, 2017).

Comparison to FedAvg and variants: (Hsu et al., 2019) and (Wang et al., 2020c) observed that using *server momentum* significantly improves over vanilla FEDAVG. This idea was generalized by (Reddi et al., 2020), who replaced the server update with an arbitrary optimizer, e.g. Adam. However, these methods only modify the server update while using SGD for the client updates. MIME, on the other hand, ensures that every *local client update* resembles the optimizer e.g. MIME would apply momentum in every client update and not just at the server level. Beyond this, (Li et al., 2018b) proposed to add a regularizer to ensure client updates remain close. However, this may slow down convergence (cf. (Karimireddy et al., 2020b; Wang et al., 2020b)). Other orthogonal directions which can be combined with MIME include tackling computation heterogeneity, where some clients perform many more updates than others (Wang et al., 2020b), improving fairness by modifying the objective (Mohri et al., 2019; Li et al., 2019b), incorporating differential privacy (Geyer et al., 2017; Agarwal et al., 2018; Thakkar et al., 2020), Byzantine adversaries (Pillutla et al., 2019; Wang et al., 2020a; Karimireddy et al., 2021b,a), secure aggregation (Bonawitz et al., 2017; He et al., 2020), etc. We defer additional discussion to the extensive surveys by (Kairouz et al., 2019; Wang et al., 2021).

5.3 Problem setup

This section formalizes the problem of cross-device federated learning (Kairouz et al., 2019). Cross-device FL is characterized by a large number of client devices like mobile phones which may potentially connect to the server at most once. Due to their transient nature, it is not possible to store any state on the clients, precluding an algorithm like SCAFFOLD. Furthermore, each client has only a few samples, and there is wide heterogeneity in the samples across clients. Finally, communication is a major bottleneck and a key metric for optimization in this setting is the number of communication rounds.

Thus, our objective will be to minimize the following quantity within the fewest number of client-server communication rounds:

$$f(\mathbf{x}) = \mathbb{E}_{i \sim \mathcal{D}} \left[f_i(\mathbf{x}) := \frac{1}{n_i} \sum_{v=1}^{n_i} f_i(\mathbf{x}; \zeta_{i,v}) \right]. \quad (5.1)$$

Here, f_i denotes the loss function of client i and $\{\zeta_{i,1}, \dots, \zeta_{i,n_i}\}$ its local data. Since the number of clients is extremely large, while the size of each local data is rather modest, we represent the former as an expectation and the latter as a finite sum. In each round, the algorithm samples a subset of clients (of size S) and performs some updates to the server model. Due to the transient and heterogeneous nature of the clients, it is easy to see that the problem becomes intractable with arbitrarily dissimilar clients. Thus, it is **necessary** to assume bounded dissimilarity across clients.

(A1) G^2 -BGV or bounded inter-client gradient variance: there exists $G \geq 0$ such that

$$\mathbb{E}_{i \sim \mathcal{D}} [\|\nabla f_i(\mathbf{x}) - \nabla f(\mathbf{x})\|^2] \leq G^2, \forall \mathbf{x}.$$

Next, we also characterize the variance in the Hessians.

(A2) δ -BHV or bounded Hessian variance: Almost surely, the loss function of any client i satisfies

$$\|\nabla^2 f_i(\mathbf{x}; \zeta) - \nabla^2 f(\mathbf{x})\| \leq \delta, \forall \mathbf{x}.$$

This is in contrast to the usual smoothness assumption that can be stated as:

(A2*) L -smooth: $\|\nabla^2 f_i(\mathbf{x}; \zeta)\| \leq L, \forall \mathbf{x}$, a.s. for any i .

Note that if $f_i(\mathbf{x}; \zeta)$ is L -smooth then (A2) is satisfied with $\delta \leq 2L$, and hence (A2) is *weaker* than (A2*). In realistic examples we expect the clients to be similar and hence that $\delta \ll L$. In addition, we assume that $f(\mathbf{x})$ is bounded from below by f^* and is L -smooth, as is standard.

5.4 How momentum can help reduce client drift

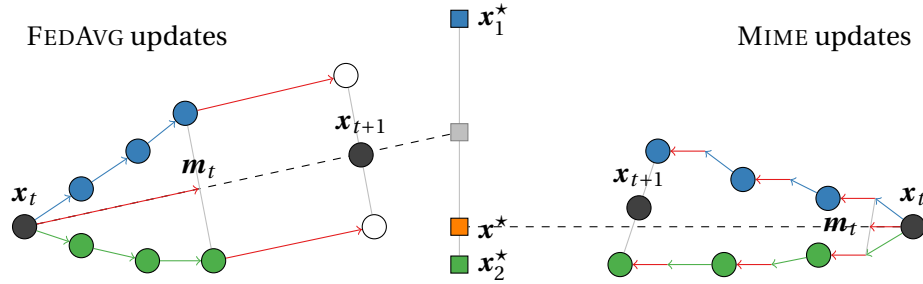


Figure 5.1 – Client-drift in FEDAVG (left) and MIME (right) is illustrated for 2 clients with 3 local steps and momentum parameter $\beta = 0.5$. The local SGD updates of FEDAVG (shown using arrows for **client 1** and **client 2**) move towards the average of client optima $\frac{\mathbf{x}_1^* + \mathbf{x}_2^*}{2}$ which can be quite different from the true **global optimum** \mathbf{x}^* . Server **momentum** only speeds up the convergence to the wrong point in this case. In contrast, MIME uses unbiased **momentum** and applies it locally at every update. This keeps the updates of MIME closer to the true **optimum** \mathbf{x}^* .

In this section we examine the tension between reducing communication by running multiple client updates each round, and degradation in performance due to client drift (Karimireddy et al., 2020b). To simplify the discussion, we assume a single client is sampled each round and that clients use full-batch gradients. **Server-only approach.** A simple way to avoid the issue of client drift is to take no local steps. We sample a client $i \sim \mathcal{D}$ and run SGD

with momentum (SGDm) with momentum parameter β and step size η :

$$\begin{aligned}\mathbf{x}_t &= \mathbf{x}_{t-1} - \eta((1 - \beta)\nabla f_i(\mathbf{x}_{t-1}) + \beta\mathbf{m}_{t-1}), \\ \mathbf{m}_t &= (1 - \beta)\nabla f_i(\mathbf{x}_{t-1}) + \beta\mathbf{m}_{t-1}.\end{aligned}\tag{5.2}$$

Here, the gradient $\nabla f_i(\mathbf{x}_t)$ is *unbiased* i.e. $\mathbb{E}[\nabla f_i(\mathbf{x}_t)] = \nabla f(\mathbf{x}_t)$ and hence we are guaranteed convergence. However, this strategy can be communication-intensive and we are likely to spend all our time waiting for communication with very little time spent on computing the gradients.

FEDAVG approach. To reduce the overall communication rounds required, we need to make more progress in each round of communication. Starting from $\mathbf{y}_0 = \mathbf{x}_{t-1}$, FEDAVG (McMahan et al., 2017) runs multiple SGD steps on the sampled client $i \sim \mathcal{D}$

$$\mathbf{y}_k = \mathbf{y}_{k-1} - \eta\nabla f_i(\mathbf{y}_{k-1}) \text{ for } k \in [K],\tag{5.3}$$

and then a pseudo-gradient $\tilde{\mathbf{g}}_t = -(\mathbf{y}_K - \mathbf{x}_t)$ replaces $\nabla f_i(\mathbf{x}_{t-1})$ in the SGDm algorithm (5.2). This is referred to as server-momentum since it is computed and applied only at the server level (Hsu et al., 2019). However, such updates give rise to *client-drift* resulting in performance worse than the naïve server-only strategy (5.2). This is because by using multiple local updates, (5.3) starts over-fitting to the local client data, optimizing $f_i(\mathbf{x})$ instead of the actual global objective $f(\mathbf{x})$. The net effect is that FEDAVG moves towards an incorrect point (see Fig 5.1, left). If K is sufficiently large, approximately

$$\begin{aligned}\mathbf{y}_K &\rightsquigarrow \mathbf{x}_i^*, \text{ where } \mathbf{x}_i^* := \underset{\mathbf{x}}{\operatorname{argmin}} f_i(\mathbf{x}) \\ \Rightarrow \mathbb{E}_{i \sim \mathcal{D}}[\tilde{\mathbf{g}}_t] &\rightsquigarrow (\mathbf{x}_t - \mathbb{E}_{i \sim \mathcal{D}}[\mathbf{x}_i^*]).\end{aligned}$$

Further, the server momentum is based on $\tilde{\mathbf{g}}_t$ and hence is also biased. Thus, it cannot correct for the client drift. We next see how a different way of using momentum can mitigate client drift.

Mime approach. FEDAVG experiences client drift because both the momentum and the client updates are biased. To fix the former, we compute momentum using only global optimizer state as in (5.2) using the sampled client $i \sim \mathcal{D}$:

$$\mathbf{m}_t = (1 - \beta)\nabla f_i(\mathbf{x}_{t-1}) + \beta\mathbf{m}_{t-1}.\tag{5.4}$$

To reduce the bias in the local updates, we will apply this unbiased momentum every step $k \in [K]$:

$$\mathbf{y}_k = \mathbf{y}_{k-1} - \eta((1 - \beta)\nabla f_i(\mathbf{y}_{k-1}) + \beta\mathbf{m}_{t-1}).\tag{5.5}$$

Note that the momentum term is kept fixed during the local updates i.e. there is no local momentum used, only global momentum is applied locally. Since \mathbf{m}_{t-1} is a moving average

Algorithm 6 Mime and MimeLite

input: initial \mathbf{x} and \mathbf{s} , learning rate η and base algorithm $\mathcal{B} = (\mathcal{U}, \mathcal{V})$
for each round $t = 1, \dots, T$ **do**
 sample subset \mathcal{S} of clients
 communicate (\mathbf{x}, \mathbf{s}) to all clients $i \in \mathcal{S}$
 communicate $\mathbf{c} \leftarrow \frac{1}{|\mathcal{S}|} \sum_{j \in \mathcal{S}} \nabla f_j(\mathbf{x})$ (only Mime)
 for each client $i \in \mathcal{S}$ **in parallel do**
 initialize local model $\mathbf{y}_i \leftarrow \mathbf{x}$
 for $k = 1, \dots, K$ **do**
 sample mini-batch ζ from local data
 $\mathbf{g}_i \leftarrow \nabla f_i(\mathbf{y}_i; \zeta) - \nabla f_i(\mathbf{x}; \zeta) + \mathbf{c}$ (Mime)
 $\mathbf{g}_i \leftarrow \nabla f_i(\mathbf{y}_i; \zeta)$ (MimeLite)
 update $\mathbf{y}_i \leftarrow \mathbf{y}_i - \eta \mathcal{U}(\mathbf{g}_i, \mathbf{s})$
 end for
 compute full local-batch gradient $\nabla f_i(\mathbf{x})$
 communicate $(\mathbf{y}_i, \nabla f_i(\mathbf{x}))$
 end for
 $\mathbf{s} \leftarrow \mathcal{V}\left(\frac{1}{|\mathcal{S}|} \sum_{i \in \mathcal{S}} \nabla f_i(\mathbf{x}), \mathbf{s}\right)$ (update optimizer state)
 $\mathbf{x} \leftarrow \frac{1}{|\mathcal{S}|} \sum_{i \in \mathcal{S}} \mathbf{y}_i$ (update server parameters)
end for

of unbiased gradients computed over multiple clients, it intuitively is a good approximation of the general direction of the updates. By taking a convex combination of the local gradient with \mathbf{m}_{t-1} , the update (5.5) is potentially also less biased. In this way MIMe combines the communication benefits of taking multiple local steps and prevents client-drift (see Fig 5.1, right). Appendix 5.7 makes this intuition precise.

5.5 Mime framework

In this section we describe how to adapt an arbitrary centralized optimizer (referred to as the “base” algorithm) which may have internal state (e.g. momentum in SGD) to the federated learning problem (5.1) while ensuring there is no client-drift.

Algorithm 6 describes our framework. We develop two variants, MIMe and MIMeLite, which consist of three components i) a base algorithm we are seeking to mimic, ii) how we compute the global (server) optimizer state, and iii) the local client updates.

Base algorithm. We assume the centralized base algorithm we are imitating can be decomposed into two steps: an *update step* \mathcal{U} which updates the parameters \mathbf{x} , and a *optimizer state update step* $\mathcal{V}(\cdot)$ which keeps track of global optimizer state \mathbf{s} . Each step of the base algorithm $\mathcal{B} = (\mathcal{U}, \mathcal{V})$ uses a gradient \mathbf{g} to update the parameter \mathbf{x} and the optimizer state \mathbf{s} as

follows:

$$\begin{aligned}\mathbf{x} &\leftarrow \mathbf{x} - \eta \mathcal{U}(\mathbf{g}, \mathbf{s}), \\ \mathbf{s} &\leftarrow \mathcal{V}(\mathbf{g}, \mathbf{s}).\end{aligned}\tag{BASEALG}$$

As an example, consider SGD with momentum (SGDm). The state in SGDm is the momentum \mathbf{m}_t . SGDm uses the following update steps:

$$\begin{aligned}\mathbf{x}_t &= \mathbf{x}_{t-1} - \eta ((1 - \beta) \nabla f_i(\mathbf{x}_{t-1}) + \beta \mathbf{m}_{t-1}), \\ \mathbf{m}_t &= (1 - \beta) \nabla f_i(\mathbf{x}_{t-1}) + \beta \mathbf{m}_{t-1}.\end{aligned}$$

Thus, SGDm can be represented in the above generic form with $\mathcal{U}(\mathbf{g}, \mathbf{s}) = (1 - \beta)\mathbf{g} + \beta\mathbf{s}$ and $\mathcal{V}(\mathbf{g}, \mathbf{s}) = (1 - \beta)\mathbf{g} + \beta\mathbf{s}$. See Appendix for how other algorithms like Adam, Adagrad, etc. can be represented in this manner. We keep the update \mathcal{U} to be linear in the gradient \mathbf{g} , whereas \mathcal{V} can be more complicated. This implies that while the parameter update step \mathcal{U} is relatively resilient to receiving a biased gradient \mathbf{g} while \mathcal{V} can be much more sensitive.

Compute optimizer state globally, apply locally. When updating the optimizer state of the base algorithm, we use only the gradient computed at the server parameters. Further, they remain fixed throughout the local updates of the clients. This ensures that these optimizer state remain unbiased and representative of the global function $f(\cdot)$. At the end of the round, the server performs

$$\begin{aligned}\mathbf{s} &\leftarrow \mathcal{V}\left(\frac{1}{|\mathcal{S}|} \sum_{i \in \mathcal{S}} \nabla f_i(\mathbf{x}), \mathbf{s}\right), \\ \nabla f_i(\mathbf{x}) &= \frac{1}{n_i} \sum_{v=1}^{n_i} \nabla f_i(\mathbf{x}; \zeta_{i,v}).\end{aligned}\tag{OPTSTATE}$$

Note that we use full-batch gradients computed at the server parameters \mathbf{x} , not client parameters \mathbf{y}_i . **Local client updates.** Each client $i \in \mathcal{S}$ performs K updates using \mathcal{U} of the base algorithm and a minibatch gradient. There are two variants possible corresponding to **MIME** and **MIMELITE** differentiated using colored boxes. Starting from $\mathbf{y}_i \leftarrow \mathbf{x}$, repeat the following K times

$$\mathbf{y}_i \leftarrow \mathbf{y}_i - \eta \mathcal{U}(\mathbf{g}_i, \mathbf{s})\tag{CLTSTEP}$$

where $\mathbf{g}_i \leftarrow \nabla f_i(\mathbf{y}_i; \zeta)$ for MIMELITE, and $\mathbf{g}_i \leftarrow \nabla f_i(\mathbf{y}_i; \zeta) - \nabla f_i(\mathbf{x}; \zeta) + \frac{1}{|\mathcal{S}|} \sum_{j \in \mathcal{S}} \nabla f_j(\mathbf{x})$ for MIME. MIMELITE simply uses the local minibatch gradient whereas MIME uses an SVRG style correction (Johnson and Zhang, 2013). This is done to reduce the noise from sampling a local mini-batch. While this correction yields faster rates in theory (and in practice for convex problems), in deep learning applications we found that MIMELITE closely matches the performance of MIME.

Finally, there are two modifications made in practical FL: we weight all averages across the clients by the number of datapoints n_i (McMahan et al., 2017), and we perform K epochs instead of K steps (Wang et al., 2020b).

5.6 Theoretical analysis of Mime

Table 5.1 summarizes the rates of MIME (highlighted in blue) and MIMELITE (highlighted in green) and compares them to SERVER-ONLY methods when using SGD, Adam and momentum methods as the base algorithms. We will first examine the convergence of MIME and MIMELITE with a generic base optimizer and show that its properties are preserved in the federated setting. We then examine a specific momentum based base optimizer, and prove that Mime and MimeLite can be asymptotically faster than the best server-only method. This is the first result to prove the usefulness of local steps and demonstrate asymptotic speed-ups.

5.6.1 Convergence with a generic base optimizer

We will prove a generic reduction result demonstrating that if the underlying base algorithm converges, and is robust to slight perturbations, then MIME and MIMELITE also preserve the convergence of the algorithm when applied to the federated setting with additional local steps.

Theorem X. Suppose that we have G^2 inter-client gradient variance (A1), L -smooth $\{f_i\}$ (A2*), and σ^2 intra-client gradient variance. Further, suppose that the updater \mathcal{U} of our base-optimizer $\mathcal{B} = (\mathcal{U}, \mathcal{V})$ satisfies i) linearity: $\mathcal{U}(\mathbf{g}_1 + \mathbf{g}_2) = \mathcal{U}(\mathbf{g}_1) + \mathcal{U}(\mathbf{g}_2)$, and ii) Lipschitzness: $\|\mathcal{U}(\mathbf{g})\| \leq B\|\mathbf{g}\|$ for some $B \geq 0$. Then, running MIME or MIMELITE with K local updates and step-size η is equivalent to running a **centralized** algorithm with step-size $\tilde{\eta} := K\eta \leq \frac{1}{2LB}$, and updates

$$\begin{aligned} \mathbf{x}_t &\leftarrow \mathbf{x}_{t-1} - \tilde{\eta} \mathcal{U}(\mathbf{g}_t + \mathbf{e}_t, \mathbf{s}_{t-1}), \text{ and} \\ \mathbf{s}_t &\leftarrow \mathcal{V}(\mathbf{g}_t, \mathbf{s}_{t-1}), \text{ where we have} \end{aligned}$$

$$\mathbb{E}_t[\mathbf{g}_t] = \nabla f(\mathbf{x}_{t-1}), \mathbb{E}_t\|\mathbf{g}_t - \nabla f(\mathbf{x}_{t-1})\|^2 \leq G^2/S, \text{ and}$$

$$\frac{1}{B^2 L^2 \tilde{\eta}^2} \mathbb{E}_t\|\mathbf{e}_t\|^2 \leq \begin{cases} \mathbb{E}_t\|\mathbf{g}_t\|^2 & \text{MIME,} \\ \mathbb{E}_t\|\mathbf{g}_t\|^2 + G^2 + \frac{\sigma^2}{K} & \text{MIMELITE.} \end{cases}$$

Here, we have proven that MIME and MIMELITE truly mimic the centralized base algorithm with very small perturbations—the magnitude of \mathbf{e}_t is $\mathcal{O}(\tilde{\eta}^2)$. The key to the result is the linearity of the parameter update step $\mathcal{U}(\cdot)$. By separating the base optimizer into a very simple parameter step \mathcal{U} and a more complicated optimizer state update step \mathcal{V} , we can ensure that commonly used algorithms such as momentum, Adam, Adagrad, and others all satisfy this property. Armed with this general reduction, we can easily obtain specific convergence results.

Corollary XI ((Mime/MimeLite) with SGD). Given that the conditions in Theorem X are satisfied, let us run T rounds with K local steps using SGD as the base optimizer and output \mathbf{x}^{out} . This output satisfies $\mathbb{E}\|\nabla f(\mathbf{x}^{out})\|^2 \leq \epsilon$ for $F := f(\mathbf{x}_0) - f^*$, $\tilde{G}^2 := G^2 + \sigma^2/K$ and

- **μ -PL inequality:** $\eta = \tilde{\mathcal{O}}\left(\frac{1}{\mu KT}\right)$, and

Chapter 5. Mime: mimicking centralized algorithms in cross-device federated learning

Table 5.1 – Number of communication rounds required to reach $\|\nabla f(\mathbf{x})\|^2 \leq \epsilon$ (log factors are ignored) with S clients sampled each round. All analyses except SCAFFOLD assume G^2 bounded gradient dissimilarity (A1). All analyses assume L -smooth losses, except MimeLiteMVR and MimeMVR, which only assume δ bounded Hessian dissimilarity (A2). Convergence of SCAFFOLD depends on the total number of clients N which is potentially infinite. FEDAVG and MIMELITE are slightly slower than the server-only methods due to additional drift terms in most cases. MIME is the fastest and either matches or improves upon the optimal statistical rates (first term in the rates). In fact, MimeMVR and MimeLiteMVR beat lower bounds for any server-only method when $\delta \ll L$.

Algorithm	Non-convex	μ -PL inequality
SCAFFOLD ^a (Karimireddy et al., 2020b)	$\left(\frac{N}{S}\right)^{\frac{2}{3}} \frac{L}{\epsilon}$	$\frac{N}{S} + \frac{L}{\mu}$
SGD		
SERVER-ONLY (Ghadimi and Lan, 2013)	$\frac{LG^2}{S\epsilon^2} + \frac{L}{\epsilon}$	$\frac{G^2}{\mu S\epsilon} + \frac{L}{\mu}$
MimeLiteSGD \equiv FedSGD ^c	$\frac{LG^2}{S\epsilon^2} + \frac{L^2 G}{\epsilon^{3/2}} + \frac{L}{\epsilon}$	$\frac{G^2}{\mu S\epsilon} + \frac{LG}{\mu\sqrt{\epsilon}} + \frac{L}{\mu}$
MimeSGD	$\frac{LG^2}{S\epsilon^2} + \frac{L}{\epsilon}$	$\frac{G^2}{\mu S\epsilon} + \frac{L}{\mu}$
ADAM		
SERVER-ONLY (Zaheer et al., 2018) ^b	$\frac{L}{\epsilon - G^2/S}$	–
MimeLiteAdam ^{bc}	$\frac{L\sqrt{S}}{\epsilon - G^2/S}$	–
MimeAdam ^b	$\frac{L}{\epsilon - G^2/S}$	–
Momentum Variance Reduction (MVR)		
SERVER-ONLY (Cutkosky and Orabona, 2019)	$\frac{LG}{\sqrt{S}\epsilon^{3/2}} + \frac{L}{\epsilon}$	–
MimeLiteMVR ^d	$\frac{\delta(G+\sigma)}{\epsilon^{3/2}} + \frac{G^2+\sigma^2}{\epsilon} + \frac{\delta}{\epsilon}$	–
MimeMVR ^d	$\frac{\delta G}{\sqrt{S}\epsilon^{3/2}} + \frac{G^2}{S\epsilon} + \frac{\delta}{\epsilon}$	–
SERVER-ONLY lower bound (Arjevani et al., 2019)	$\Omega\left(\frac{LG}{\sqrt{S}\epsilon^{3/2}} + \frac{G^2}{S\epsilon} + \frac{L}{\epsilon}\right)$	$\Omega\left(\frac{G^2}{S\epsilon}\right)$

^a Num. clients (N) can be same order as num. total rounds or even ∞ , making the bounds vacuous.

^b Adam requires large batch-size $S \geq G^2/\epsilon$ to converge (Reddi et al., 2018; Zaheer et al., 2018). Convergence of FedAdam with client sampling is unknown ((Reddi et al., 2020) only analyze with full client participation).

^c Requires $K \geq \sigma^2/G^2$ number of local updates. Typically, intra-client variance is small ($\sigma^2 \lesssim G^2$).

^d Requires $K \geq L/\delta$ number of local updates. Faster than the lower bound (and hence any SERVER-ONLY algorithm) when $\delta \ll L$ i.e. our methods can take advantage of Hessian similarity, whereas SERVER-ONLY methods cannot. In worst case, $\delta \approx L$ and all methods are comparable.

$$T = \begin{cases} \tilde{\mathcal{O}}\left(\frac{LG^2}{\mu S\epsilon} + \frac{LF}{\mu} \log\left(\frac{1}{\epsilon}\right)\right) & \text{MIME,} \\ \tilde{\mathcal{O}}\left(\frac{LG^2}{\mu S\epsilon} + \frac{L\tilde{G}}{\mu\sqrt{\epsilon}} + \frac{LF}{\mu} \log\left(\frac{1}{\epsilon}\right)\right) & \text{MIMELITE.} \end{cases}$$

- **Non-convex:** for $\eta = \mathcal{O}\left(\sqrt{\frac{FS}{L\tilde{G}^2TK^2}}\right)$, and

$$T = \begin{cases} \mathcal{O}\left(\frac{LG^2F}{S\epsilon^2} + \frac{LF}{\epsilon}\right) & \text{MIME,} \\ \mathcal{O}\left(\frac{L\tilde{G}^2F}{S\epsilon^2} + \frac{L^2\tilde{G}F}{\epsilon^{3/2}} + \frac{LF}{\epsilon}\right) & \text{MIMELITE.} \end{cases}$$

If we take a sufficient number of local steps $K \geq G^2/\sigma^2$, then we have $\tilde{G} = \mathcal{O}(G)$ in the above rates. On comparing with the rates in Table 5.1 for SERVER-ONLY SGD, we see that MIME exactly matches its rates. MIMELITE matches the asymptotic term but has a few higher order terms. Note that when using SGD as the base optimizer, MIMELITE becomes exactly the same as FEDAVG and hence has the same rate of convergence.

Corollary XII ((Mime/MimeLite) with Adam). *Suppose that the conditions in Theorem X are satisfied, and further $|\nabla_j f_i(\mathbf{x})| \leq H$ for any coordinate $j \in [d]$. Then let us run T rounds using Adam as the base optimizer with K local steps, $\beta_1 = 0$, $\epsilon_0 > 0$, $\eta \leq \epsilon_0^2/KL(H + \epsilon_0)$, and any $\beta_2 \in [0, 1]$. Output \mathbf{x}^{out} chosen randomly from $\{\mathbf{x}_1, \dots, \mathbf{x}_T\}$ satisfies $\mathbb{E}\|\nabla f(\mathbf{x}^{out})\|^2 \leq \epsilon$ for*

$$T = \begin{cases} \mathcal{O}\left(\frac{LF(H+\epsilon_0)^2}{\epsilon_0^2(\epsilon - \tilde{G}^2/S)}\right) & \text{MIME Adam,} \\ \mathcal{O}\left(\frac{LF(H+\epsilon_0)^2\sqrt{S}}{\epsilon_0^2(\epsilon - \tilde{G}^2/S)}\right) & \text{MIMELITE Adam.} \end{cases}$$

where $F := f(\mathbf{x}_0) - f^*$, $\tilde{G}^2 := G^2 + \sigma^2/K$.

Note that here ϵ_0 represents a small positive parameter used in Adam for regularization, and is different from the accuracy ϵ . Similar to the SERVER-ONLY analysis of Adam (Zaheer et al., 2018), we assume $\beta_1 = 0$ and that batch size is large enough such that $S \geq G^2/\epsilon$. A similar analysis can also be carried out for AdaGrad, and other novel variants of Adam (Liu et al., 2019).

5.6.2 Circumventing server-only lower bounds

The rates obtained above, while providing a safety-check, do not beat those of the SERVER-ONLY approach. The previous best rates for cross-device FL correspond to MimeLiteSGD which is $\mathcal{O}\left(\frac{LG^2}{S\epsilon^2} + \frac{L^2G}{\epsilon^{3/2}}\right)$ (Khaled et al., 2020; Koloskova et al., 2020; Woodworth et al., 2020a). While, using a separate server-learning rate can remove the effect of the second term (Karimireddy et al., 2019), this at best matches the rate of SERVER-ONLY SGD $\mathcal{O}\left(\frac{LG^2}{S\epsilon^2}\right)$. This is significantly slower than simply using momentum based variance reduction (MVR) as in the FL setting (SERVER-ONLY MVR) which has a communication complexity of $\mathcal{O}\left(\frac{LG}{\sqrt{S\epsilon^{3/2}}}\right)$ (Cutkosky and Orabona, 2019). Thus, even though the main reason for studying local-step methods was to improve the communication complexity, none thus far show such improvement. The above difficulty of beating SERVER-ONLY may not be surprising given the two sets of strong lower bounds known.

Necessity of local steps. Firstly, (Arjevani et al., 2019) show a gradient oracle lower bound of $\Omega(\frac{LG}{\sqrt{S}\epsilon^{3/2}})$. This matches the complexity of MVR, and hence at first glance it seems that SERVER-ONLY MVR is optimal. However, the lower bound is really only on the number of gradients computed and not on the number of clients sampled (sample complexity) (Foster et al., 2019), or number of rounds of communication required. In particular, multiple local updates which increases number of gradients computed *without needing additional communication* offers us a potential way to side-step such lower bounds. A careful analysis of the bias introduced as a result of such local steps is a key part of our analysis.

Necessity of δ -BHD. A second set of lower bounds directly study the number of communication rounds required in heterogeneous optimization (Arjevani and Shamir, 2015; Woodworth et al., 2020a). These results prove that there exist settings where local steps provide no advantage and SERVER-ONLY methods are optimal. This however contradicts real world experimental evidence (McMahan et al., 2017). As before, the disparity arises due to the contrived settings considered by the lower bounds. For distributed optimization (with full client participation) and convex quadratic objectives, δ -BHD (A2) was shown to be a sufficient (Shamir et al., 2014; Reddi et al., 2016b) and *necessary* (Arjevani and Shamir, 2015) condition to circumvent these lower bounds and yield highly performant methods. We similarly leverage δ -BHD (A2) to design novel methods which significantly extend prior results to i) all smooth non-convex functions (not just quadratics), and ii) cross-device FL with client sampling.

We now state our convergence results with momentum based variance reduction (MVR) as the base-algorithm since it is known to be optimal in the SERVER-ONLY setting.

Theorem XIII. For L -smooth f with G^2 gradient dissimilarity (A1), δ Hessian dissimilarity (A2) and $F := (f(\mathbf{x}^0) - f^*)$, let us run MVR as the base algorithm for T rounds with $K \geq L/\delta$ local steps and generate an output \mathbf{x}^{out} . This output satisfies $\mathbb{E}\|\nabla f(\mathbf{x}^{\text{out}})\|^2 \leq \epsilon$ for

- **MimeMVR**: $\eta = \mathcal{O}\left(\min\left(\frac{1}{\delta K}, \left(\frac{SF}{G^2 TK^3}\right)^{1/3}\right)\right)$, momentum $\beta = 1 - \mathcal{O}\left(\frac{\delta^2 S^{2/3}}{(TG^2)^{2/3}}\right)$, and

$$T = \mathcal{O}\left(\frac{\delta GF}{\sqrt{S}\epsilon^{3/2}} + \frac{G^2}{S\epsilon} + \frac{\delta F}{\epsilon}\right).$$

- **MimeLiteMVR**: $\eta = \mathcal{O}\left(\min\left(\frac{1}{\delta K}, \left(\frac{F}{\hat{G}^2 TK^3}\right)^{1/3}\right)\right)$, momentum $\beta = 1 - \mathcal{O}\left(\frac{\delta^2}{(T\hat{G}^2)^{2/3}}\right)$, and

$$T = \mathcal{O}\left(\frac{\delta \hat{G} F}{\epsilon^{3/2}} + \frac{\hat{G}^2}{\epsilon} + \frac{\delta F}{\epsilon}\right).$$

Here, we define $\hat{G}^2 := G^2 + \sigma^2$ and the expectation in $\mathbb{E}\|\nabla f(\mathbf{x}^{\text{out}})\|^2 \leq \epsilon$ is taken both over the sampling of the clients during the running of the algorithm, the sampling of the mini-batches in local updates, and the choice of \mathbf{x}^{out} (which is chosen randomly from the client iterates \mathbf{y}_i).

Remarkably, the rates of our methods are independent of L and only depend on δ . Thus, when $\delta \leq L$ and $\delta \leq L/s$ for MimeMVR and MimeLiteMVR, the rates beat the server only lower bound of $\Omega(\frac{LG}{\sqrt{S}\epsilon^{3/2}})$. In fact, if the Hessian variance is small and $\delta \approx 0$, our methods only need $\mathcal{O}(1/\epsilon)$ rounds to communicate. Intuitively, our results show that local steps are very useful

when heterogeneity (represented by δ) is smaller than optimization difficulty (captured by smoothness constant L).

MimeMVR uses a momentum parameter β of the order of $(1 - \mathcal{O}(TG^2)^{-2/3})$ i.e. as T increases, β asymptotically approaches 1. In contrast, previous analyses of distributed momentum (e.g. (Yu et al., 2019a)) prove rates of the form $\frac{G^2}{S(1-\beta)\epsilon^2}$, which are worse than that of standard SGD by a factor of $\frac{1}{1-\beta}$. Thus, ours is also the first result which theoretically showcases the usefulness of using large momentum in distributed and federated learning.

Our analysis is highly non-trivial and involves two crucial ingredients: i) computing the momentum at the server level to ensure that it remains unbiased and then applying it locally during every client update to reduce variance, and ii) carefully keeping track of the bias introduced via additional local steps. Our experiments (Sec. 5.8) verify our theoretical insights are indeed applicable in deep learning settings as well.

5.7 Proof sketch

In this section, we give proof sketches of the main components of Theorem XIII: i) how momentum reduces the effect of client drift, ii) how local steps can take advantage of Hessian similarity, and iii) why the SVRG correction improves constants.

Improving the statistical term via momentum. Note that the statistical (first) term in Theorem XIII without momentum ($\beta = 0$) for the convex case is $\frac{LG^2}{\mu S \epsilon}$. This is (up to constants) optimal and cannot be improved. For the non-convex case however using $\beta = 0$ gives the usual rate of $\frac{LG^2}{S \epsilon^2}$. However, this can be improved to $\left(\frac{(1+\delta)G^2 F}{S \epsilon^2}\right)^{3/4}$ using momentum. This matches a similar improvement in the centralized setting (Cutkosky and Orabona, 2019; Tran-Dinh et al., 2019) and is in fact optimal (Arjevani et al., 2019). Let us examine why momentum improves the statistical term. Assume that we sample a single client i_t in round t and that we use full-batch gradients. Also let the local client update at step k round t be of the form

$$\mathbf{y} \leftarrow \mathbf{y} - \eta \mathbf{d}_k. \quad (5.6)$$

The ideal choice of update is of course $\mathbf{d}_k^* = \nabla f(\mathbf{y})$ but however this is unattainable. Instead, MIME with momentum $\beta = 1 - a$ uses $\mathbf{d}_k^{\text{SGDm}} = \tilde{\mathbf{m}}_k \leftarrow a \nabla f_{i_t}(\mathbf{y}) + (1 - a) \mathbf{m}_{t-1}$ where \mathbf{m}_{t-1} is the momentum computed at the server. The variance of this update can then be bounded as

$$\begin{aligned} \mathbb{E} \|\tilde{\mathbf{m}}_k - \nabla f(\mathbf{y})\|^2 &\lesssim a^2 \mathbb{E} \|\nabla f_{i_t}(\mathbf{y}) - \nabla f(\mathbf{y})\|^2 + (1 - a) \mathbb{E} \|\mathbf{m}_{t-1} - \nabla f(\mathbf{y})\|^2 \\ &\approx a^2 G^2 + (1 - a) \mathbb{E} \|\mathbf{m}_{t-1} - \nabla f(\mathbf{x}_{t-2})\|^2 \approx a G^2. \end{aligned}$$

The last step follows by unrolling the recursion on the variance of \mathbf{m} . We also assumed that η is small enough that $\mathbf{y} \approx \mathbf{x}_{t-2}$. This way, momentum can reduce the variance of the update from G^2 to (aG^2) by using past gradients computed on different clients. To formalize the above sketch requires slightly modifying the momentum algorithm similar to (Cutkosky and Orabona, 2019).

Improving the optimization term via local steps. The optimization (second) term in Theorem XIII for the convex case is $\frac{\delta K + L}{\mu K}$ and for the non-convex case (with or without momentum) is $\frac{\delta K + L}{\epsilon K}$. In contrast, the optimization term of the server-only methods is L/μ and L/ϵ respectively. Since in most cases $\delta \ll L$, the former can be significantly smaller than the latter. This rate also suggests that the best choice of number of local updates is L/δ i.e. we should perform more client updates when they have more similar Hessians. This generalizes results of (Karimireddy et al., 2020b) from quadratics to all functions.

This improvement is due to a careful analysis of the *bias* in the gradients computed during the local update steps. Note that for client parameters \mathbf{y}_{k-1} , the gradient $\mathbb{E}[\nabla f_{i_t}(\mathbf{y}_{k-1})] \neq \mathbb{E}[\nabla f(\mathbf{y}_{k-1})]$ since \mathbf{y}_{k-1} was also computed using the same loss function f_{i_t} . In fact, only the first gradient computed at \mathbf{x}_{t-1} is unbiased. Dropping the subscripts k and t , we can bound this bias as:

$$\begin{aligned} \mathbb{E}[\nabla f_i(\mathbf{y}) - \nabla f(\mathbf{y})] &= \underbrace{\mathbb{E}[\nabla f_i(\mathbf{y}) - \nabla f_i(\mathbf{x})]}_{\approx \nabla^2 f_i(\mathbf{x})(\mathbf{y} - \mathbf{x})} + \underbrace{\mathbb{E}[\nabla f_i(\mathbf{x}) - \nabla f(\mathbf{y}_i)]}_{\approx \nabla^2 f(\mathbf{x})(\mathbf{x} - \mathbf{y}_i)} + \underbrace{\mathbb{E}[\nabla f_i(\mathbf{x})] - \nabla f(\mathbf{x})}_{=0 \text{ since unbiased}} \\ &\approx \mathbb{E}[(\nabla^2 f_i(\mathbf{x}) - \nabla^2 f(\mathbf{x}))(\mathbf{y}_i - \mathbf{x})] \approx \delta \mathbb{E}[(\mathbf{y}_i - \mathbf{x})]. \end{aligned}$$

Thus, the Hessian dissimilarity (A2) control the bias, and hence the usefulness of local updates. This intuition can be made formal in the Appendix.

Mini-batches via SVRG correction. In our previous discussion about momentum and local steps, we assumed that the clients compute full batch gradients and that only one client is sampled per round. However, in practice a large number (S) of clients are sampled and further the clients use mini-batch gradients. The SVRG correction reduces this within-client variance since

$$\text{Var}\left(\nabla f_i(\mathbf{y}_i; \zeta) - \nabla f_i(\mathbf{x}; \zeta) + \frac{1}{|\mathcal{S}|} \sum_{i \in \mathcal{S}} \nabla f_i(\mathbf{x})\right) \lesssim L^2 \|\mathbf{y}_i - \mathbf{x}\|^2 + \frac{G^2}{S} \approx \frac{G^2}{S}.$$

Here, we used the smoothness of $f_i(\cdot; \zeta)$ and assumed that $\mathbf{y}_i \approx \mathbf{x}$ since we don't move too far within a single round. Thus, the SVRG correction allows us to use minibatch gradients in the local updates while still ensuring that the variance is of the order G^2/S .

5.8 Experimental analysis on real world datasets

We run experiments on *natively* federated datasets to confirm our theory and accurately measure real world performance. Our main findings are i) MIME and MIMELITE consistently outperform FEDAVG, and ii) momentum and adaptivity significantly improves performance.

5.8.1 Setup

Algorithms. We consider three (meta) algorithms: FEDAVG, MIME, and MIMELITE. Each of these adapt four base optimizers: SGD, momentum, Adam, and Adagrad.

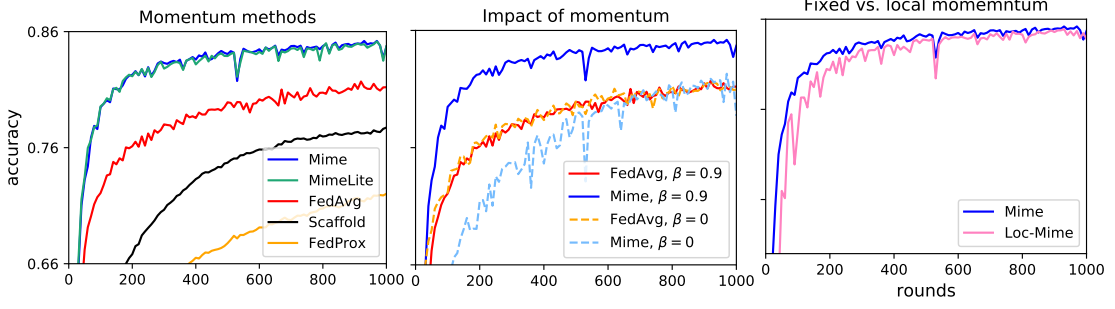


Figure 5.2 – **Mime**, **MimeLite**, **FedAvg**, **Scaffold**, **FedProx**, and **Loc-Mime** with SGD+momentum using 10 local epochs, run on EMNIST62 and a 2 hidden layer (300u-100) MLP. (Left) Mime and MimeLite are nearly identical and outperform the rest (7× faster). (Center) Mime makes better use of momentum than FedAvg, with a large increase in performance. (Right) Locally adapting momentum slows down convergence and makes it more unstable.

FEDAVG follows (Reddi et al., 2020) who run multiple epochs of SGD on each client sampled, and then aggregate the net client updates. This aggregated update is used as a pseudo-gradient in the base optimizer (called server optimizer). The learning rate for the server optimizer is fixed to 1 as in (Wang et al., 2020c). This is done to ensure all algorithms have the same number of hyper-parameters.

MIME and MIMELITE follow Algorithm 6 and also run a fixed number of epochs on the client. However, note that this requires communicating both the full local-batch gradient as well as the parameter updates doubling the communication required to be sent by the client. For a fairer comparison, we split the sampled clients in MIME and MIMELITE into two groups—the first communicates only full local-batch gradient and the latter communicates only parameter updates. Thus, all methods have **equal client communication** to the server. This variant retains the convergence guarantees up to constants (details in the Appendix). We also run Loc-MIME where instead of keeping the global optimizer state fixed, we update it locally within the client. The optimizer state is reset after the round finishes. In all methods, aggregation is weighted by the number of samples on the clients.

Datasets and models. We run five simulations on three real-world federated datasets: EMNIST62 with i) a linear classifier, ii) an MLP, and iii) a CNN, iv) a charRNN on Shakespeare, and v) an LSTM for next word prediction on StackOverflow, all accessed through Tensorflow Federated (TFF, 2020). The learning rates were individually tuned and other optimizer hyper-parameters such as β for momentum, β_1 , β_2 , ϵ_0 for Adam and AdaGrad were left to their default values, unless explicitly stated otherwise. We refer to the Appendix for additional setup details and discussion.

5.8.2 Ablation and comparative study

In order to study the different algorithms, we train a 2 hidden layer (300 μ -100) MLP on EMNIST62 with 10 local epochs for 1k rounds and use SGD+momentum (with tuned β) as the base optimizer.

Mime \approx MimeLite $>$ FedAvg $>$ SCAFFOLD $>$ FedProx. Fig. 5.2 (left) shows MIME and MIMELITE have nearly identical performance, and are about $7\times$ faster than FedAvg. This implies our strategy of applying momentum to client updates is faster than simply using server momentum. FedProx (Li et al., 2018b) uses an additional regularizer μ tuned over $[0.1, 0.5, 1]$ ($\mu = 0$ is the same as FedAvg). Regularization does not seem to reduce client drift but still slows down convergence (Wang et al., 2020b). SCAFFOLD (Karimireddy et al., 2020b) is also slower than Mime and FedAvg in this setup. This is because in cross-device setting with a large number of clients ($N = 3.4k$) means that each client is visited less than 6 times during the entire training (20 clients per round for 1k rounds). Hence, the client control variate stored is quite stale (from about 200 rounds ago) which slows down the convergence.

With momentum $>$ without momentum. Fig. 5.2 (center) examines the impact of momentum on FedAvg and Mime. Momentum slightly improves the performance of FedAvg, whereas it has a significant impact on the performance of Mime. This is also in line with our theory and confirms that Mime’s strategy of applying it locally at every client update makes better use of momentum.

Fixed $>$ locally updated optimizer state. Finally, we check how the performance of Mime changes if instead of keeping the momentum fixed throughout a round, we let it change. The latter is a way to combine global and local momentum. The momentum is reset at the end of the round ignoring the changes the clients make to it. Fig. 5.2 (right) shows that this *worsens* the performance, confirming that it is better to keep the global optimizer state fixed as predicted by our theory.

Together, the above observations validate all aspects of Mime (and MimeLite) design: compute statistics at the server level, and apply them unchanged at every client update.

5.8.3 Large scale comparison with equal server and client communication

We perform a larger scale study closely matching the setup of (Reddi et al., 2020). For both MIME and MIMELITE, only half the clients compute and transmit the updated parameters, and other half transmit the full local-batch gradients. Hence, client to server communication cost is the same for all methods for all clients. However, MIME and MIMELITE require sending additional optimization state to the clients. Hence, we also reduce the number of clients sampled in each round to ensure *sum total* of communication at each round is $40\times$ model size for EMNIST and Shakespeare experiments, and $100\times$ model size for the StackOverflow next word prediction experiment.

Since we only perform 1 local epoch, the hyper-parameters (e.g. epsilon for adaptive methods) are more carefully chosen following (Reddi et al., 2020), and MIME and MIMELITE use significantly fewer clients per round, the difference between FEDAVG and MIME is smaller here. Table 5.2 summarizes the results.

For the image classification tasks of EMNIST62 logistic and EMNIST62 CNN, Mime and MimeLite with Adam achieve the best performance. Using momentum (both with SGDm, and in Adam) significantly improves their performance. In contrast, FedAvgAdam is more unstable with worse performance. This is because FedAvg is excessively sensitive to hyperparameters (see

5.8. Experimental analysis on real world datasets

Table 5.2 – Validation % accuracies after training for 1000 rounds. Best results for each dataset is underlined and the best within each base optimizer is bolded. The number of clients sampled per round has been reduced for MIME and MIMELITE to ensure all methods have **equal client and server communication**. Final accuracies obtained by MIME and MIMELITE are competitive with FEDAVG, especially with adaptive base optimizers. FEDAVG seems unstable with Adam.

		EMNIST logistic	EMNIST CNN	Shakespeare	StackOverflow
SGD	FedAvgSGD	66.8	85.8	56.7	23.8
	MimeLiteSGD	66.8	85.8	56.7	23.8
	MimeSGD	67.4	85.3	56.1	12.5
MOMENTUM	FedAvgMom	67.4	85.7	55.4	22.2
	MimeLiteMom	67.4	86.0	49.8	19.9
	MimeMom	67.5	85.9	53.6	19.3
ADAM	FedAvgAdam	67.3	85.9	18.5	3.2
	MimeLiteAdam	68.0	86.4	54.0	21.5
	MimeAdam	68.0	86.6	54.1	22.8
ADAGRAD	FedAvgAdagrad	67.6	86.3	55.5	24.2
	MimeLiteAdagrad	66.6	85.5	56.8	23.8
	MimeAdagrad	67.4	86.3	57.1	14.7

Appendix).

We next consider the character prediction task on Shakespeare dataset, and next word prediction on StackOverflow. Here, the momentum based methods (SGDm and Adam) are slower than their non-momentum counterparts (SGD and AdaGrad). This is because the mini-batch gradients in these tasks are *sparse*, with the gradients corresponding to tokens not in the mini-batch being zero. This sparsity structure is however destroyed when using momentum or Adam. For the same reason, Mime which uses an SVRG correction also significantly increases the gradient density.

Discussion. For traditional deep learning tasks such as image classification, we observe that Mime outperforms MimeLite which in turn outperforms FedAvg. These methods are able to successfully leverage momentum to improve performance. For tasks where the client gradients are sparse, the SVRG correction used by Mime hinders performance. Adapting our techniques to work with sparse gradients (à la Yogi (Zaheer et al., 2018)) could lead to further improvements. Also, note that we reduce communication by naïvely reducing the number of participating clients per round. More sophisticated approaches to save on client communication including quantization or sparsification (Suresh et al., 2017; Alistarh et al., 2017), or even novel algorithmic innovations (Acar et al., 2021) could be explored. Further, server communication could be reduced using memory efficient optimizers e.g. AdaFactor (Shazeer and Stern, 2018) or SM3 (Anil et al., 2019).

5.9 Conclusion

Our work initiated a formal study of the cross-device federated learning problem and provided theoretically justified algorithms. We introduced a new framework MIME which overcomes the natural client-heterogeneity in such a setting, and can adapt arbitrary centralized algorithms such as Adam without additional hyper-parameters. We demonstrated the superiority of MIME via strong convergence guarantees and empirical evaluations. Further, we proved that a particular instance of our method, MimeMVR, beat centralized lower-bounds, demonstrating that additional local steps can yield asymptotic improvements for the first time. We believe our analysis will be of independent interest beyond the federated setting for understanding the sample complexity of non-convex optimization, and for yielding improved analysis of decentralized optimization algorithms.

Byzantine Robustness **Part III**

6 Learning from History for Byzantine Robust Optimization

6.1 Preface

Contribution and sources. This chapter reproduces (Karimireddy et al., 2021b). The theory, toy experiments, and writing were carried out mostly by the author. Lie He conducted most of the experiments. Detailed individual contributions:

SPK (author): Conceptualization, Methodology, Formal analysis, Writing – original draft preparation (90 %)

Lie He: Software, Writing – original draft preparation (10 %)

Martin Jaggi: Supervision, Administration, Writing – review and editing .

Summary. Collaborative learning (federated or distributed) allows any participant to retain their data and train a model in a collaborative manner. However, this also opens up the system to potentially malicious (or simply faulty) participants who seek to derail the training. Byzantine robustness seeks to develop algorithms which are resilient to such attackers. While this area has received considerable attention given its importance, we identify severe flaws in existing methods even when the data across the participants is identically distributed.

First, we show realistic examples where current state of the art robust aggregation rules fail to converge even in the absence of any Byzantine attackers. This, we show, is because traditional definitions of robustness do not suffice for our setting. We introduce a more fine-grained definition of robust aggregator and give a new *iterative clipping* procedure which satisfies it. Our procedure is efficient, and compatible with secure aggregation and all-reduce.

Secondly, we show that even if the aggregation rules may succeed in limiting the influence of the attackers in a single round, the attackers can couple their attacks across time eventually leading to divergence of any memory-less systems. This implies that in order to ensure Byzantine robustness, it is necessary to profile workers using past updates. We then show that simply incorporation *local momentum* is sufficient to overcome such time-coupled attacks. This is the first provably robust method for the standard stochastic optimization setting.

6.2 Introduction

“Those who cannot remember the past are condemned to repeat it.” – George Santayana.

Growing sizes of datasets as well as concerns over data ownership, security, and privacy have lead to emergence of new machine learning paradigms such as distributed and federated learning (Kairouz et al., 2019). In both of these settings, a central coordinator orchestrates many worker nodes in order to train a model over data which remains decentralized across the workers. While this decentralization improves scalability security and privacy, it also opens up the training process to manipulation by the workers (Lamport et al., 2019). These workers may be actively malicious trying to derail the process, or might simply be malfunctioning and hence sending arbitrary messages. Ensuring that our training procedure is robust to a small fraction of such potentially malicious agents is termed Byzantine robust learning and is the focus of the current work.

Given the importance of this problem, it has received significant attention from the community with early works including (Feng et al., 2014; Blanchard et al., 2017; Chen et al., 2017; Yin et al., 2018a). Most of these approaches replace the averaging step of distributed or federated SGD with a robust aggregation rule such as the median. However, a closer inspection reveals that these procedures are quite brittle: we show that there exist realistic scenarios where they fail to converge, even if there are *no Byzantine attackers* and the data distribution is identical across the workers (i.i.d.). This turns out to be because on their excessive sensitivity to the distribution of the noise in the gradients. The impractical assumptions made by these methods are often violated in practice, and lead to the failure of these aggregation rules.

Further, there have been recent state of the art attacks (Baruch et al., 2019; Xie et al., 2020) which empirically demonstrate a second source of failure. They show that even when current aggregation rules may succeed in limiting the influence of the attackers in any single round, they may still diverge when run for multiple rounds. We prove that this is inevitable for a wide class of methods—any aggregation rule which ignores history can be made to eventually diverge. This is accomplished by using the inherent noise in the gradients to mask small perturbations which are undetectable in a single round, but accumulate over time.

Finally, we show how to circumvent both the issues outlines above. We first describe a simple new aggregator based on iterative *centered clipping* which is much more robust to the distribution of the gradient noise. This aggregator is especially interesting since, unlike most preceding methods, it is very scalable requiring only $\mathcal{O}(n)$ computation and communication per round. Further, it is also compatible with other strategies such as asynchronous updates (Chen et al., 2016) and secure aggregation (Bonawitz et al., 2017), both of which are crucial for real world applications. Secondly, we show that the time coupled attacks can easily be overcome by using *worker momentum*. Momentum averages the updates of each worker over time, reducing the variance of the good workers and exposing the time-coupled pertur-

bations. We prove that our methods obtain optimal rates, and our theory also sheds light on the role of momentum in decreasing variance and building resilience to Byzantine workers.

Contributions. Our main results are summarized below.

- We show that most state of the art robust aggregators require strong assumptions and can fail in real settings even in the complete absence of Byzantine workers.
- We prove a strong lower bound showing that any optimization procedure which does not use history will diverge in the presence of time coupled attacks.
- We propose a simple and efficient aggregation rule based on iterative clipping and prove its performance under standard assumptions.
- We show that using momentum successfully defends against time-coupled attacks and provably converges when combined with any Byzantine robust aggregator.
- We incorporate the recent momentum based variance reduction (MVR) with Byzantine aggregators to obtain optimal rates for robust non-convex optimization.
- We perform extensive numerical experiments validating our techniques and results.

Setup. Let us formalize the robust non-convex stochastic optimization problem in the presence of a δ fraction of Byzantine workers.

Definition D (δ -robust non-convex optimization). *Given some loss function $f(\mathbf{x})$, $\epsilon > 0$, and access to n workers we want to find a stationary point \mathbf{x} such that $\mathbb{E}\|\nabla f(\mathbf{x})\|^2 \leq \epsilon$. The optimization proceeds in rounds where in every round, each worker $i \in [n]$ can compute a stochastic gradient $g_i(\mathbf{y})$ at any parameter \mathbf{y} in parallel. Then, each worker $i \in [n]$ sends some message $\mathcal{M}_{i,t}$ to the server. The server utilizes these messages to update the parameters and proceeds to the next round. During this process, we will assume that*

- The function f is L -smooth i.e. it satisfies $\|\nabla f(\mathbf{x}) - \nabla f(\mathbf{y})\| \leq L\|\mathbf{x} - \mathbf{y}\|$ for any \mathbf{x}, \mathbf{y} , and is bounded from below by f^* .
- Each worker i has access to an independent and unbiased stochastic gradient with $\mathbb{E}[g_i(\mathbf{x})|\mathbf{x}] = \nabla f(\mathbf{x})$ and variance bounded by σ^2 , $\mathbb{E}\|g_i(\mathbf{x}) - \nabla f(\mathbf{x})\|^2 \leq \sigma^2$.
- Of the n workers, at least $(1-\delta)n$ workers are good (denoted by \mathcal{G}) and will follow the protocol faithfully. The rest of the bad or Byzantine workers (denoted by \mathcal{B}) may act maliciously and can communicate arbitrary messages to the server.
- These Byzantine workers are assumed to omniscient i.e. they have access to the computations made by the rest of the good workers. However, we assume that this set of Byzantine workers \mathcal{B} remains fixed throughout the optimization process.

6.3 Related work

Robust aggregators. Distributed algorithms in the presence of Byzantine agents has a long history (Lamport et al., 2019) and is becoming increasingly important in modern distribution and federated machine learning (Kairouz et al., 2019). Most solutions involve replacing the averaging of the updates from the different machines with more robust aggregation

rules such as coordinate-wise median method (Yin et al., 2018a), geometric median methods (Blanchard et al., 2017; Chen et al., 2017; Pillutla et al., 2019), majority voting (Bernstein et al., 2018; Jin et al., 2020) etc. There have also been attempts to use recent breakthroughs in robust high-dimensional aggregators (Diakonikolas et al., 2018; Su and Xu, 2018; El-Mhamdi and Guerraoui, 2019; Data and Diggavi, 2020). However, these latter procedures are computationally expensive (quadratic in dimensions per round) and further it is unclear if the improved guarantees for mean estimation translate to improved performance in the distributed machine learning settings. Finally, for most of the above approaches, convergence guarantees when provided rely on using an extremely large batch size or strong unrealistic assumptions making them practically irrelevant.

Other more heuristic approaches propose to use a penalization or reweighting of the updates based on reputations (Peng et al., 2020; Li et al., 2019a; Fu et al., 2019; Regatti and Gupta, 2020; Rodríguez-Barroso et al., 2020). These schemes however need to trust that all workers report correct statistics. In such settings where we have full control over the workers (e.g. within a datacenter) coding theory based solutions which can correct for the mistakes have also been proposed (Chen et al., 2018b; Rajput et al., 2019; Gupta and Vaidya, 2019; Konstantinidis and Ramamoorthy, 2020; Data and Diggavi, 2020). These however are not applicable in federated learning where the data is decentralized across untrusted workers.

Time coupled attacks and defenses. Recently, two state-of-the-art attacks have been proposed which show that the state of the art Byzantine aggregation rules can be easily circumvented (Baruch et al., 2019; Xie et al., 2020). The key insight is that while the robust aggregation rules may ensure that the influence of the Byzantine workers in any single round is limited, the attackers can couple their attacks across the rounds. This way, over many training rounds the attacker is able to move weights significantly away from the desired direction and thus achieve the goal of lowering the model quality. Defending against time-coupled attacks and showing provable guarantees is one of the main concerns of this work.

It is clear that time-coupled attacks need time-coupled defenses. Closest to our work is that of Alistarh et al. (2018) who use martingale concentration across the rounds to give optimal Byzantine robust algorithms for convex functions. However, this algorithm is inherently not applicable to more general non-convex functions. The recent independent work of Allen-Zhu et al. (2021) extend the method of Alistarh et al. (2018) to non-convex functions as well. However, they assume that the noise in stochastic gradients is bounded almost surely instead of the more standard assumption that only the variance is bounded. Theoretically, such strong assumptions are unlikely to hold (Zhang et al., 2019b) and even Gaussian noise is excluded. Further, the lower-bounds of (Arjevani et al., 2019) no longer apply, and thus their algorithm may be sub-optimal. Practically, their algorithm removes suspected workers either permanently (a decision of high risk), or resets the list of suspects at each window boundary (which is sensitive to the choice of hyperparameters). Having said that, (Allen-Zhu et al., 2021) prove convergence to a local minimum instead of to a saddle point as we do here. Finally, in another independent work Mhamdi et al. (2021) empirically observe that using momentum may be

beneficial, though they provide no theoretical guarantees.

Other concerns. To deploy robust learning for real world applications, many other issues such as data heterogeneity become important (Kairouz et al., 2019; Karimireddy et al., 2020b). Robust learning algorithms which assume worker data are i.i.d. may fail in the federated learning setting (Karimireddy et al., 2021a). Numerous variations have been proposed which can handle non-iid data with varying degrees of success (Li et al., 2019a; Ghosh et al., 2019; Chen et al., 2019c; Peng et al., 2020; Data and Diggavi, 2020; ?; El-Mhamdi et al., 2020; Dong et al., 2020). Further, combining robustness with notions of privacy and security is also a crucial and challenging problem (He et al., 2020; So et al., 2020a,b; Jin et al., 2020). Such heterogeneity is especially challenging and can lead to backdoor attacks (which are orthogonal to the training attacks discussed here) (Bagdasaryan et al., 2020; Sun et al., 2019; Wang et al., 2020a) and remains an open challenge.

6.4 Brittleness of existing aggregation rules

In this section, we study the robustness of existing popular Byzantine aggregation rules. Unfortunately, we come to a surprising conclusion—most state of the art aggregators require strong non-realistic restrictions on the noise distribution. We show this frequently does not hold in practice, and present counter-examples where these aggregators fail even in the complete *absence* of Byzantine workers. State of the art aggregators such as Krum (Blanchard et al., 2017), coordinate-wise median (CW) (Yin et al., 2018a),

RFA (Pillutla et al., 2019), Bulyan (Mhamdi et al., 2018), etc. all generalize the scalar notion of the median to higher dimensions and are hence exhibit different ways of ‘middle-seeking’. At a high level, these schemes require the noise distribution to be unimodal and highly concentrated, discarding any gradients from the tail of the distribution too aggressively as ‘outliers’. We give a brief summary of these rules below. We use $[\mathbf{v}]_j$ to indicate the j th coordinate of vector \mathbf{v} .

Coordinate-wise median:

$$[\text{CM}(\mathbf{x}_1, \dots, \mathbf{x}_n)]_j = \text{median}([\mathbf{x}_1]_j, \dots, [\mathbf{x}_n]_j).$$

RFA (robust federated averaging) aka geometric median:

$$\text{RFA}(\mathbf{x}_1, \dots, \mathbf{x}_n) = \underset{\mathbf{v}}{\text{argmin}} \sum_{i=1}^n \|\mathbf{v} - \mathbf{x}_i\|_2.$$

Trimmed Mean: For each coordinate j , compute sorting Π_j which sorts the coordinate values. Compute the average after excluding (‘trimming’) δn largest and smallest values.

$$[\text{TM}(\mathbf{x}_1, \dots, \mathbf{x}_n)]_j = \frac{1}{n - 2\delta n} \sum_{i=\delta n}^{n-\delta n} [\mathbf{x}_{\Pi_j(i)}]_j.$$

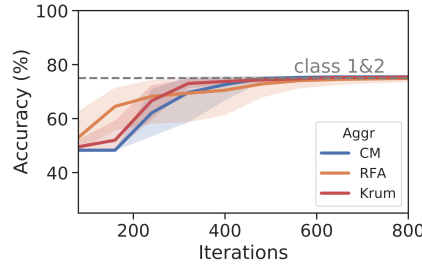


Figure 6.1 – Failure of existing methods on imbalanced MNIST dataset. Only the head classes (class 1 and 2 here) are learnt, and the rest 8 classes are ignored. See Sec. 6.8.1.

Krum: Krum tries to select a point \mathbf{x}_i which is closest to the mean after excluding $\delta n + 2$ furthest away points. Suppose that $\mathcal{S} \subset [n]$ of size at least $(n - \delta n - 2)$. Then,

$$\text{Krum}(\mathbf{x}_1, \dots, \mathbf{x}_n) = \underset{\mathbf{x}_i}{\operatorname{argmin}} \min_{\mathcal{S}} \sum_{j \in \mathcal{S}} \|\mathbf{x}_i - \mathbf{x}_j\|_2^2.$$

Counterexample 1. Let us pick n random variables ± 1 with uniform probability for some odd n . These variables have mean 0. Since n is odd, Krum, CW, Bulyan all will necessarily return either of ± 1 . This remains true even if we have infinite samples (large n), and if there are no corruptions. This simple examples illustrates the fragility of such ‘middle-seekers’ to bimodal noise.

Counterexample 2. Fig. 6.1 illustrates a more realistic example where imbalanced MNIST dataset causes a similar problem. Here, 0.5 fraction of data corresponds to class 1, 0.25 to class 2, and so on. The gradients over data of the same class are much closer than those of a different class. Hence, when we pick n i.i.d. gradients, most them will belong to class 1 or 2 with very few belonging to the rest. Thus, coordinate-wise median, geometric median and Krum always select the gradient corresponding to classes 1 or 2, ensuring that we only optimize over these classes ignoring the rest.

Counterexample 3. Middle-seekers can also fail on continuous uni-modal distributions. Consider,

$$p(x) = \begin{cases} 3x^{-4} & \text{for } x \geq 1 \\ 0 & \text{o.w.} \end{cases}$$

This power-law distribution has mean 1.5 and variance 0.75. However, since the distribution is skewed, its median is $2^{1/3} \approx 1.26$ and is smaller than the mean. This difference persists even with *infinite* samples showing that with imbalanced (i.e. skewed) distributions, coordinate-wise median, geometric median and Krum do not obtain the true optimum. Empirical evidence suggests that such heavy-tailed distributions abound in deep learning, making this setting very relevant to practice (Zhang et al., 2019b).

Theorem XIV (Failure of ‘middle-seekers’). *There exist simple*

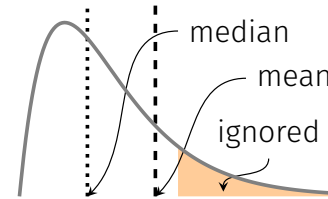


Figure 6.2 – For fat-tailed distributions, median based aggregators ignore the tail. This bias remains even if we have infinite samples.

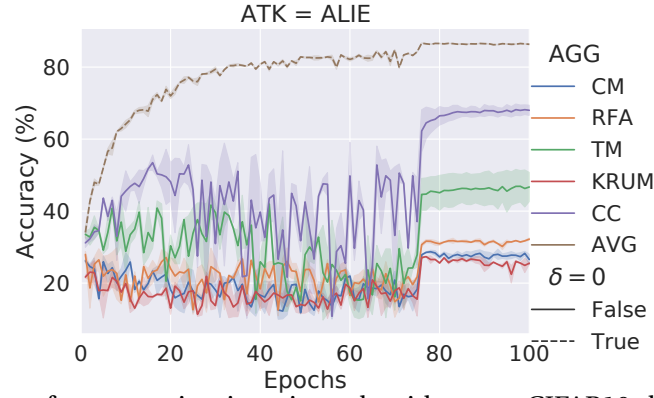


Figure 6.3 – Failure of permutation invariant algorithms on CIFAR10 dataset with (Baruch et al., 2019) attack. Comparing to simple average with no attacker (dashed lines), all robust aggregators (including centered clip) see a significant drop in accuracy against time coupled attacks. See Sec. 6.8.2.

convex stochastic optimization settings with bounded variance where traditional distributed SGD converges but coordinate-wise median, RFA, and Krum do not converge to the optimum almost surely for any number of workers and even if none of them are Byzantine.

Remark 14 (Practical usage). *Theorem XIV notes that one must be cautious while using median or Krum as aggregation rules when we suspect that our data is multi-modal (typically occurs when using small batch sizes), or if we believe our data to be heavy-tailed (typically occurs in imbalanced datasets or language tasks). These aggregators may suffice for standard image recognition tasks with large batch sizes since the noise is nearly Gaussian (Zhang et al., 2019b).*

Median based aggregators have a long and rich history in the field of robust statistics (Minsker et al., 2015). However, classically the focus of robust statistics has been to design methods which can withstand a large fraction of Byzantine workers (high *break down* point δ_{\max}) and not result in infinities (Hubert et al., 2008). It was sufficient for the output to be bounded, but the quality of the result was not a concern. The counter examples in this section exactly stem from this issue. We will later define a finer notion of a robust statistic which accounts for both the quality of the output as well as the breakdown point δ_{\max} .

6.5 Necessity of using history

Recent work (Baruch et al., 2019; Xie et al., 2020) has shown a surprising second source vulnerability for most currently popular robust aggregators. In this section we take a closer look at their attack and use our observations to make an even stronger claim—any aggregation rule which is oblivious of the past cannot converge to the optimum and retains a non-zero error even after infinite time.

The inner-product manipulation attack as defined by (Baruch et al., 2019; Xie et al., 2020) is deceptively simple. Their attacks works by hiding small Byzantine perturbations within the

variance of the good gradients. Since we only have access to noisy stochastic gradients, the aggregators fail to identify these perturbations. While this perturbation is small in any single round, these can accumulate over time. We formalize this argument into a lower bound in Theorem XVI. We show that the key reason why this attack works on algorithms such as CM, RFA, or Krum is that they are *oblivious* and do not track information from previous rounds. Thus, an attacker can couple the perturbations across time eventually leading to divergence. This is also demonstrated experimentally in Fig. 6.3.

Definition E (Permutation invariant algorithm). *Suppose we are given an instance of δ -robust optimization problem satisfying Definition D. Define the set of stochastic gradients computed by each of the n workers at some round t to be $[\tilde{\mathbf{g}}_{1,t}, \dots, \tilde{\mathbf{g}}_{n,t}]$. For a good worker $i \in \mathcal{G}$, these represent the true stochastic gradients whereas for a bad worker $j \in \mathcal{B}$, these represent arbitrary vectors. The output of any optimization algorithm ALG is a function of these gradients. A permutation-invariant algorithm is one which for any set of permutations over t rounds $\{\pi_1, \dots, \pi_t\}$, its output remains unchanged if we permute the gradients.*

$$\text{ALG} \left(\begin{bmatrix} [\tilde{\mathbf{g}}_{1,1}, \dots, \tilde{\mathbf{g}}_{n,1}] \\ \dots \\ [\tilde{\mathbf{g}}_{1,t}, \dots, \tilde{\mathbf{g}}_{n,t}] \end{bmatrix} \right) = \text{ALG} \left(\begin{bmatrix} [\tilde{\mathbf{g}}_{\pi_1(1),1}, \dots, \tilde{\mathbf{g}}_{\pi_1(n),1}] \\ \dots \\ [\tilde{\mathbf{g}}_{\pi_t(1),t}, \dots, \tilde{\mathbf{g}}_{\pi_t(n),t}] \end{bmatrix} \right)$$

Remark 15 (Memoryless methods are permutation invariant). *Any algorithm which is ‘memoryless’ i.e. uses only the computations resulting from current round is necessarily permutation-invariant since the indices corresponding to the stochastic gradient are meaningless. It is only when these stochastic gradients are tracked over multiple rounds (i.e. we use memory) do the indices carry information.*

Theorem XV (Failure of permutation-invariant methods). *Suppose we are given any permutation invariant algorithm ALG as in Definition E, $\mu \geq 0$, $\delta \in [0, 1]$, and n large enough that $\delta n \geq 4(1 + \log t)$. Then, there exists a δ -robust μ strongly-convex optimization problem satisfying Definition D, such that the output $\tilde{\mathbf{x}}_t$ of ALG after t rounds necessarily has error*

$$\mathbb{E}[f(\tilde{\mathbf{x}}_t)] - f(\mathbf{x}^*) \geq \Omega\left(\frac{\delta \sigma^2}{\mu}\right).$$

Nearly all currently popular aggregation rules, including coordinate-wise median, trimmed mean (Yin et al., 2018a), Krum (Blanchard et al., 2017), Bulyan (Mhamdi et al., 2018), RFA, geometric median (Ghosh et al., 2019), etc. are permutation invariant and satisfy Definition E. Theorem XV proves a very startling result—all of them *fail to converge* to the optimum even for strongly-convex problems. Further, as μ decreases (the problem becomes less strongly-convex), the error becomes unbounded.

Remark 16 (Fixed Byzantine workers). *The failure of permutation-invariant algorithms also illustrates the importance of assuming that the indices of Byzantine workers are fixed across rounds. If a different fraction of workers are allowed to be Byzantine each round, then the lower*

bound in Theorem XV applies to all algorithms and convergence is impossible. While it is indeed a valid concern that Byzantine workers may pretend to be someone else (or more generally perform Sybil attacks where they pretend to be multiple workers), simple mechanisms such as pre-registering all participants (perhaps using some identification) can circumvent such attacks.

There are very few methods which are not permutation invariant and are not subject to our lower bound. Examples include Byzantine SGD (Alistarh et al., 2018) which only works for convex problems, and some heuristic scoring rules such as (Regatti and Gupta, 2020). There has also been a recent independent work (Allen-Zhu et al., 2021) which utilizes history, but they have strong requirements on the noise (see Section 6.4 for why this might be an issue) and are not compatible with our problem setting. See Appendix 13.7.3 for a more detailed comparison.

6.6 Robust robust aggregation

Past work on Byzantine robust methods have had wildly varying assumptions making an unified comparison difficult. Perhaps more importantly, this lead to unanticipated failures as we saw in Sec. 6.4. In this section, we attempt to provide a standardized specification for an robust aggregator which we believe captures a wide variety of real world behavior i.e. a robust aggregator which is robust to its assumptions. We then design a simple and efficient clipping based aggregator which satisfies this notion.

6.6.1 Anatomy of a robust aggregator

Suppose that we are given an aggregation rule $\text{AGG}(\dots)$ and n vectors $\{\mathbf{x}_1, \dots, \mathbf{x}_n\}$. Among the given n vectors, let $\mathcal{G} \subseteq [n]$ be *good* (i.e. satisfy some closeness property), and the rest are Byzantine (and hence can be arbitrary). The ideal aggregator would return $\frac{1}{|\mathcal{G}|} \sum_{j \in \mathcal{G}} \mathbf{x}_j$ but this requires exactly identifying the good workers, and hence may not be possible. We will instead be satisfied if our aggregation rule approximates the ideal update up to some error.

Our notion of a robust aggregator is characterized by two quantities: δ_{\max} which denotes the breakdown point, and a constant c which determines the quality of the solution. We want an aggregator which has as large δ_{\max} and a small c .

Definition F ((δ_{\max}, c) -robust aggregator). *Suppose that for some $\delta \leq \delta_{\max} \leq 0.5$ we are given n random vectors $\mathbf{x}_1, \dots, \mathbf{x}_n$ such that a good subset $\mathcal{G} \subseteq [n]$ of size at least $|\mathcal{G}| > (1 - \delta)n$ are independent with distance bounded as*

$$\mathbb{E} \|\mathbf{x}_i - \mathbf{x}_j\|^2 \leq \rho^2,$$

for any fixed $i, j \in \mathcal{G}$. Then, define $\bar{\mathbf{x}} := \frac{1}{|\mathcal{G}|} \sum_{j \in \mathcal{G}} \mathbf{x}_j$. The, the robust aggregation rule $\text{AGG}(\mathbf{x}_1, \dots, \mathbf{x}_n)$

Algorithm 7 AGG - Centered Clipping

```

1: input:  $(\mathbf{m}_1, \dots, \mathbf{m}_n), \tau, \mathbf{v}, L$ 
2: default:  $L = 1$  and  $\mathbf{v} = \hat{\mathbf{m}}$  (previous round aggreg.)
3: for each iteration  $l = 1, \dots, L$  do
4:    $\mathbf{c}_i \leftarrow (\mathbf{m}_i - \mathbf{v}) \min\left(1, \frac{\tau}{\|\mathbf{m}_i - \mathbf{v}\|}\right)$ 
5:    $\mathbf{v} \leftarrow \mathbf{v} + \frac{1}{n} \sum_{i \in [n]} \mathbf{c}_i$ 
6: end for
7: output:  $\mathbf{v}$ 

```

outputs $\hat{\mathbf{x}}$ such that,

$$\mathbb{E}\|\hat{\mathbf{x}} - \bar{\mathbf{x}}\|^2 \leq c\delta\rho^2,$$

where the expectation is over the random variables $\{\mathbf{x}_i\}_{i \in [n]}$ and randomness in the aggregation rule AGG.

The error in Definition F is of the order $\delta\rho^2$. Thus, if $\delta = 0$ (no Byzantine workers), we recover the ideal average of the workers exactly. Further, we recover the exact average $\bar{\mathbf{x}}$ if $\rho = 0$ (no variance) since in this case all the good points are identical and are trivial to identify if they are in the majority ($\delta \leq \delta_{\max} \leq 0.5$). We demand that when the fraction of Byzantine workers is less than the breakdown point δ_{\max} , the error of the output degrades gracefully with δ .

However, the error remains positive ($\delta\rho^2$) even with infinite n and seems to indicate that having additional workers may not help. It turns out that this is unfortunately the price to pay for not knowing the good subset and is unavoidable. The following theorem is adapted from standard robust estimation lower bounds (e.g. see [Lai et al. \(2016\)](#)).

Theorem XVI (Limits of robustness). *There exist a set of n random vectors $\mathbf{x}_1, \dots, \mathbf{x}_n$ such that a good subset $\mathcal{G} \subseteq [n]$ of size at least $|\mathcal{G}| \geq (1 - \delta)n$ is i.i.d. satisfying $\mathbb{E}\|\mathbf{x}_i - \mathbf{x}_j\|^2 \leq \rho^2$, for any apriori fixed $i, j \in \mathcal{G}$. For these vectors, any aggregation rule $\hat{\mathbf{x}} = \text{AGG}(\mathbf{x}_1, \dots, \mathbf{x}_n)$ necessarily has an error*

$$\mathbb{E}\|\hat{\mathbf{x}} - \boldsymbol{\mu}\|^2 \geq \delta\rho^2.$$

Further, the error can be unbounded (∞) if $\delta \geq \frac{1}{2}$.

This establishes Definition F as the tightest notion of a robust aggregation oracle possible.

6.6.2 Robust aggregation via centered clipping

Given that most existing aggregation rules fail to satisfy Definition F, one may wonder if any such rule exists. We propose the following iterative *centered clipping* (CC) rule: starting from some point \mathbf{v}_0 , for $l \geq 0$ compute

$$\mathbf{v}_{l+1} = \mathbf{v}_l + \frac{1}{n} \sum_{i=1}^n (\mathbf{x}_i - \mathbf{v}_l) \min\left(1, \frac{\tau_l}{\|\mathbf{x}_i - \mathbf{v}_l\|}\right) \quad (\text{CC})$$

Remark 17 (Ease of implementation). *The centered clipping update is extremely simple to implement requiring $\mathcal{O}(n)$ computation and communication per step similar to coordinate-wise median. This is unlike more complicated mechanisms such as Krum or Bulyan which require $\mathcal{O}(n^2)$ computation and are hence less scalable. Further, as we will see later empirically, a single iteration of CC is often sufficient in practice. This means that the update can be implemented in an asynchronous manner (Chen et al., 2016), and is compatible with secure aggregation for federated learning (Bonawitz et al., 2017).*

We can formalize the convergence of this procedure.

Theorem XVII (Robustness of centered clipping). *Suppose that for $\delta \leq 0.1$ we are given n random vectors $\mathbf{x}_1, \dots, \mathbf{x}_n$ such that a good subset $\mathcal{G} \subseteq [n]$ of size at least $|\mathcal{G}| \geq (1 - \delta)n$ are i.i.d. with variance bounded as $\mathbb{E}\|\mathbf{x}_i - \mathbf{x}_j\|^2 \leq \rho^2$ for any fixed $i, j \in \mathcal{G}$. Then, starting from any \mathbf{v}_0 the output of centered clipping after l steps \mathbf{v}_l satisfies*

$$\mathbb{E}\|\mathbf{v}_l - \bar{\mathbf{x}}\|^2 \leq (9.7\delta)^l 3\mathbb{E}\|\mathbf{v}_0 - \bar{\mathbf{x}}\|^2 + 4000\delta\rho^2.$$

Proof Sketch. Suppose that we are given $\{\mathbf{x}_1, \dots, \mathbf{x}_n\}$ with a subset of size at most δn are bad (denoted by \mathcal{B}), and the rest are good (\mathcal{G}). Consider the following simple scenario where $\|\mathbf{x}_i\|^2 \leq \rho^2$ almost surely for any $i \in \mathcal{G}$. In such a case, a very simple aggregation rule exists: clip all values to a radius ρ and then compute the average. All the good vectors remain unchanged. The magnitude of a clipped bad vector is at most ρ and since only a δ of the vectors are bad, they can move the center by at most $\rho\delta$ ensuring that our error is $\delta^2\rho^2$. This is even better than Definition F, which only requires the error to be smaller than $\delta\rho^2$. Of course there were two aspects which over-simplified our computations in the above discussion: i) we measure the pair-wise distance $\|\mathbf{x}_i - \mathbf{x}_j\|$ between good workers instead of absolute norms, and ii) we do not have an almost sure bound, but only in expectation. \square

Corollary XVIII. *Starting from any \mathbf{v}_0 with an initial error estimate of $\mathbb{E}\|\mathbf{v}_0 - \bar{\mathbf{x}}\|^2 \leq B^2$, running CC for $l = 100\log(3B^2/\delta\rho^2)$ is a (δ_{\max}, c) -robust aggregator as per Definition F with $c = 4000$ and $\delta_{\max} = 0.1$.*

Further, if $\mathbb{E}\|\mathbf{v}_0 - \bar{\mathbf{x}}\|^2 \leq \rho^2$ then a single step of CC is a (δ_{\max}, c) -robust aggregator.

The above corollary proves that starting from *any* point \mathbf{v}_0 and running enough iterations of CC is guaranteed to provide a robust estimate. However, if we have a good starting point, we can prove a much stronger statement—that a *single* clipping step is sufficient to provide robustness. We will use this latter part in designing an efficient robust optimization scheme in the next section.

Note that we have not tried to optimize for the constants in the theorem above—there is room for improvement in bringing δ_{\max} closer to 0.5, as well as in reducing the value of c . This may need a more careful analysis, or perhaps even a new oracle. We leave such improvements for future.

Algorithm 8 Robustness using Momentum

```

1: input:  $\mathbf{x}, \eta, \beta, \text{AGG}$ 
2: initialize:  $\mathbf{m}_i \leftarrow \mathbf{0} \forall i \in [n]$ 
3: for each round  $t = 1, \dots$  do
4:   server communicates  $\mathbf{x}$  to workers
5:   for each client  $i \in \mathcal{G}$  in parallel do
6:     compute mini-batch gradient  $\mathbf{g}_i(\mathbf{x})$ 
7:     compute  $\mathbf{m}_i \leftarrow (1 - \beta)\mathbf{g}_i(\mathbf{x}) + \beta\mathbf{m}_i$ 
8:     communicate  $\mathbf{m}_i$  to server
9:   end for
10:  aggregate  $\hat{\mathbf{m}} = \text{AGG}(\mathbf{m}_1, \dots, \mathbf{m}_n)$ 
11:  update  $\mathbf{x} \leftarrow \mathbf{x} - \eta\hat{\mathbf{m}}$ 
12: end for

```

With this, we have addressed the first stumbling block and now have a robust aggregator. Next, we see how using momentum can defend against time-coupled attacks.

6.7 Robust optimization using momentum

In this section we will show that any Byzantine robust aggregator satisfying Definition F can be combined with (local) worker momentum, to obtain a Byzantine robust optimization algorithm which successfully defends against time coupled attacks. Every time step $t \geq 1$, the server sends the workers parameters \mathbf{x}_{t-1} and each good worker $i \in \mathcal{G}$ sends back $\mathbf{m}_{t,i}$ computed recursively as below starting from $\mathbf{m}_{0,i} = \mathbf{0}$

$$\mathbf{m}_{t,i} = (1 - \beta_t)\mathbf{g}_i(\mathbf{x}_{t-1}) + \beta_t\mathbf{m}_{t-1,i}. \quad (\text{WORKER})$$

The workers communicate their momentum vector to the server instead of the stochastic gradients directly since they have a much smaller variance. Byzantine workers may send arbitrary vectors to the server. The server then uses a Byzantine-resilient aggregation rule AGG such as (CC) and computes the update

$$\begin{aligned} \mathbf{m}_t &= \text{AGG}(\mathbf{m}_{t,1}, \dots, \mathbf{m}_{t,n}) \\ \mathbf{x}_t &= \mathbf{x}_{t-1} - \eta_t\mathbf{m}_t. \end{aligned} \quad (\text{SERVER})$$

Intuitively, using momentum with $\beta = (1 - \alpha)$ averages the stochastic gradients of the workers over their past $1/\alpha$ gradients. This results in a reduction of the variance of the good workers by a factor α since their noise is uncoupled. However, the variance of the time-coupled Byzantine perturbations does not reduce and becomes easy to detect.

6.7.1 Rate of convergence

Now we prove a rate of convergence of our Byzantine aggregation algorithm.

Theorem XIX (Byzantine robust SGDm). *Suppose that we are given a δ -robust problem satisfying Def. D and a (δ_{\max}, c) -robust aggregation rule satisfying Def. F for $\delta_{\max} \geq \delta$. Then, running WORKER update with step-sizes $\eta_t = \min\left(\sqrt{\frac{(f(\mathbf{x}_0) - f^*) + \frac{5c\delta}{16L}\sigma^2}{20LT\sigma^2\left(\frac{2}{n} + c\delta\right)}}, \frac{1}{8L}\right)$ and momentum parameter $\alpha_1 = 1$ and $\alpha_t = 8L\eta_{t-1}$ for $t \geq 2$ satisfies*

$$\begin{aligned} \frac{1}{T} \sum_{t=1}^T \mathbb{E} \|\nabla f(\mathbf{x}_{t-1})\|^2 \leq & \\ & 16\sqrt{\frac{\sigma^2(1 + c\delta n)}{nT}} (10L(f(\mathbf{x}_0) - f^*) + 3c\delta\sigma^2) + \\ & \frac{32L(f(\mathbf{x}_0) - f^*)}{T} + \frac{20\sigma^2(1 + c\delta n)}{nT}. \end{aligned}$$

Remark 18 (Convergence rate). *The rate of convergence in Theorem XIX is asymptotically (ignoring constants and higher order terms) of the order:*

$$\frac{1}{T} \sum_{t=1}^T \mathbb{E} \|\nabla f(\mathbf{x}_{t-1})\|^2 \lesssim \sqrt{\frac{\sigma^2}{T} \left(\frac{1}{n} + \delta\right)}.$$

First note that when $\delta = 0$ i.e. when there are no Byzantine adversaries, we recover the optimal rate of $\frac{\sigma}{\sqrt{nT}}$ which linearly scales with the number of workers n . In the presence of a δ fraction of adversaries, the rate has two terms: the first term $\frac{\sigma}{\sqrt{nT}}$ which linearly scales with the number of workers n , and a second $\frac{\sigma\sqrt{\delta}}{\sqrt{T}}$ which depends on the fraction of adversaries δ but does not improve with increasing workers. Similar phenomenon occurs in the classical robust mean estimation setting (Lai et al., 2016) and is unfortunately not possible to improve.

Our algorithm uses step-size η and momentum parameter $\alpha = (1 - \beta)$ of the order of $\sqrt{\frac{1}{nT\sigma^2} + \frac{\delta}{T\sigma^2}}$. Here δ represents the fraction of adversarial workers. When there are very few bad workers with $\delta = \mathcal{O}(\frac{1}{n})$, the momentum and the step-size parameters can remain as in the non-Byzantine case. As the number of adversaries increases, δ increases meaning we should use smaller learning rate and larger momentum. Either when using linear scaling (Goyal et al., 2017) or square-root scaling (Hoffer et al., 2017), we need to scale both the learning-rate and momentum parameters as $(\frac{1}{n} + \delta)$ instead of the traditional $\frac{1}{n}$ in the presence of a δ fraction of adversaries.

The above algorithm and convergence analysis crucially relied on the low variance of the update from the workers using worker momentum. The very high momentum ensures that the variance of the updates from the workers to the server have a variance of the order $\sqrt{\frac{\sigma^2}{nT} + \frac{\delta\sigma^2}{T}}$. Note that this variance asymptotically goes to 0 with T and is significantly smaller than the variance of the stochastic gradient σ^2 . This way, the Byzantine adversaries have very little lee-way to fool the aggregator.

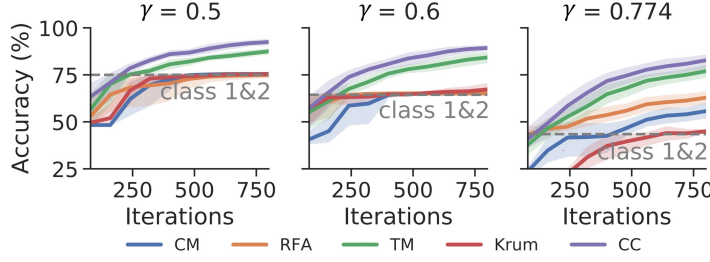


Figure 6.4 – Robust aggregation rules on imbalanced MNIST where each successive class is a γ -fraction of the previous. Centered Clip is unaffected by imbalance where as the accuracy RFA, Krum, and CM corresponds to only learning class 1 and 2 (marked by horizontal gray dashed line).

6.7.2 Improved convergence using MVR

Recently, a variation of the standard momentum, called momentum based variance reduction or MVR, was proposed by [Tran-Dinh et al. \(2019\)](#); [Cutkosky and Orabona \(2019\)](#). They show that by adding a small correction to correct for bias, we can improve SGD's $\mathcal{O}(T^{-\frac{1}{2}})$ rate of convergence to $\mathcal{O}(T^{-\frac{2}{3}})$. By combining worker momentum based variance reduction with a Byzantine robust aggregator, we can obtain a faster Byzantine robust algorithm.

Theorem XX (Byzantine robust MVR). *Suppose we are given a δ -robust Byzantine optimization problem Def. D. Let us run the MVR algorithm combined with a (δ_{\max}, c) -robust aggregation rule AGG with $\delta \leq \delta_{\max}$, step-size $\eta = \min \mathcal{O}\left(\sqrt[3]{\frac{f(\mathbf{x}_0) - f^*}{T}}, \frac{1}{4L}\right)$, and momentum parameter $\alpha = \mathcal{O}(L^2\eta^2)$. Then,*

$$\frac{1}{T} \sum_{t=1}^T \mathbb{E} \|\nabla f(\mathbf{x}_{t-1})\|^2 \lesssim \left(\frac{L\sigma\sqrt{c\delta + 1/n}}{T} \right)^{2/3}.$$

Note that Theorem XX provides a significant asymptotic speedup over the traditional momentum used in Theorem XIX and matches the lower bound of [\(Arjevani et al., 2019\)](#) when $\delta = 0$. This result highlights the versatility of our approach and the ease with which our notion of a Byzantine oracle can be combined with any state of the art optimization methods.

6.8 Experiments

In this section, we empirically demonstrate the effectiveness of CC and SGDM for Byzantine-robust learning. We refer to the baseline robust aggregation rules as RFA ([Pillutla et al., 2019](#)), coordinate-wise median (CM), trimmed mean (TM) ([Yin et al., 2018a](#)), and Krum ([Blanchard et al., 2017](#)). The inner iteration (T) of RFA is fixed to 3 as suggested in ([Pillutla et al., 2019](#)). Throughout the section, we consider the distributed training for two image classification tasks, namely MNIST ([LeCun et al., 1998](#)) on 16 nodes and CIFAR-10 ([Krizhevsky et al., 2009](#)) on 25 nodes. All experiments are repeated at least 2 times. The detailed setups are deferred to Section 13.7.1.

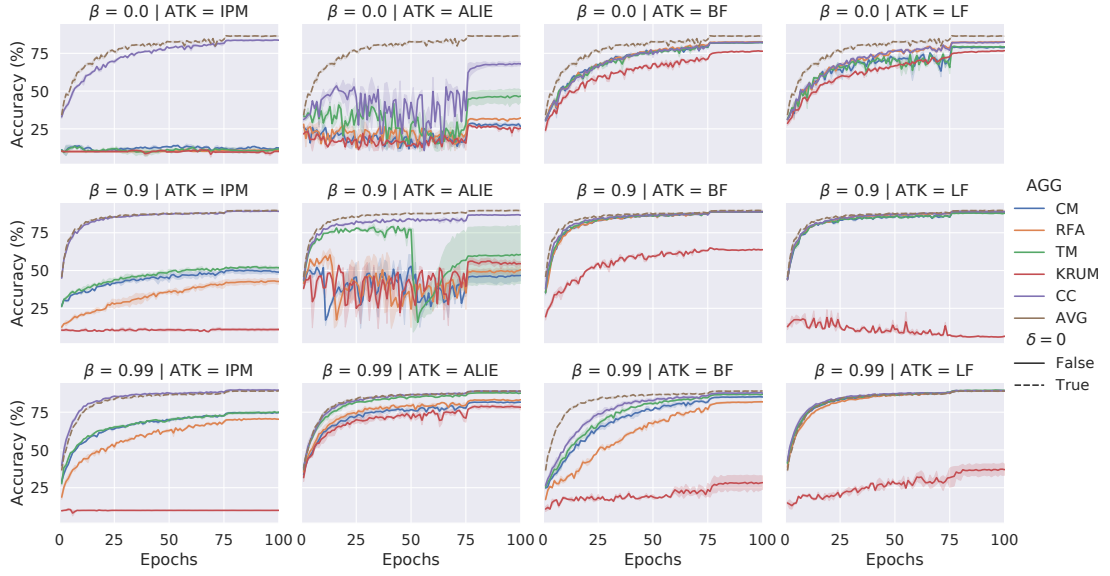


Figure 6.5 – Coordinate median (CM), Robust Federated Aggregation (RFA), Trimmed Mean (TM), Krum, and Centered Clip (CC) are tested on Cifar10 with 25 workers. Attackers run inner-product manipulation attack (IPM) (Xie et al., 2020), “a little is enough” (ALIE) (Baruch et al., 2019), bit-flipping (BF), and label-flipping (LF). IPM uses 11 Byzantine workers while others use 5. The dashed brown line is average aggregator under no attacks ($\delta = 0$). Momentum generally improves all methods, with larger momentum adding stability. Centered Clip (CC) consistently has the best performance.

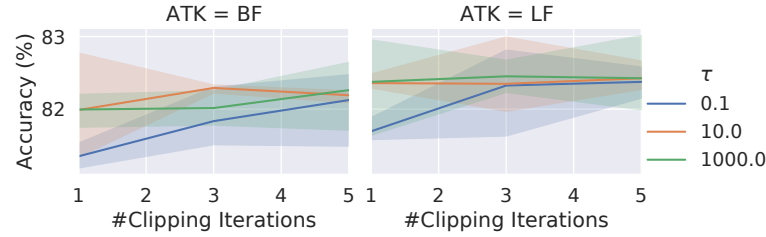


Figure 6.6 – Final test accuracy of Centered Clip as we vary clipping iterations (l) and radius (τ). It is stable across all hyper-parameters, justifying using $l = 1$ as default.

6.8.1 Failure of “middle seekers”

In this experiment, we demonstrate the challenge stated in Section 6.4 by comparing robust aggregation rules on imbalanced datasets without attackers. Imbalanced training and test MNIST dataset are created by sampling classes with exponential decay, that is $1, \gamma, \gamma^2, \dots, \gamma^{K-1}$ for classes 1 to K ($\gamma \in (0, 1]$). Then we shuffle the dataset and divide it equally into 16 nodes. The mini-batch for each node is 1.

The experimental results are presented in Fig. 6.4. For drastic decay $\gamma = 0.5$, the median and geometric median based rules can only achieve 75% accuracy which is the portion of class 1 and 2 in the data. This is a practical example of how “middle-seekers” fail. On the other hand, centered clip CC and trimmed mean have no such bound as they incorporate the gradients

from tail distributions.

6.8.2 Impact of momentum on robust aggregation rules

The traditional implementation of momentum slightly differs from (WORKER) update and uses

$$\mathbf{m}_{t,i} = \mathbf{g}_i(\mathbf{x}_{t-1}) + \beta \mathbf{m}_{t-1,i}. \quad (6.1)$$

This version is equivalent to running (WORKER) update with a re-scaled learning rate of $\eta/(1-\beta)$. Further, note that our theory predicts that the clipping radius τ should be proportional to the variance of the updates which in turn depends on the momentum parameter β . We scale τ by a factor of $(1 - \beta)$ if using (WORKER) update, and leave it constant if using update of the form (6.1).

In this experiment, we study the influence of momentum on robust aggregation rules against various attacks, including bit-flipping (BF), label-flipping (LF), little is enough (Baruch et al., 2019), and inner product manipulation (Xie et al., 2020). We train ResNet-20 (He et al., 2016a) on CIFAR-10 for 100 epochs on 25 workers where 5 of them are adversaries. For (Xie et al., 2020) we use 11 Byzantine workers to amplify the attack. The batch size per worker is set to 32 and the learning rate is 0.1 before 75th epoch and 0.01 afterwards. Note that the smaller batch size, e.g. 32, leads to larger variance among good gradients which makes the attacks in (Baruch et al., 2019; Xie et al., 2020) more challenging.

The results are presented in Fig. 6.5. Momentum generally makes the convergence faster and better for all aggregators, especially against SOTA attacks (Baruch et al., 2019; Xie et al., 2020). CC achieves best performance in almost all experiments. More specifically, it performs especially well on (Baruch et al., 2019; Xie et al., 2020) which is very close to training without attackers ($\delta = 0$).

6.8.3 Stability of Centered Clip

To demonstrate the impact of two hyperparameters τ , l of centered clip CC, we grid search τ in $[0.1, 10, 1000]$ and l in $[1, 3, 5]$. The setup is the same as in Sec. 6.8.2 and momentum is 0 to exclude its effect. The final accuracies are presented in Fig. 6.6. Centered clipping is very stable to the choice of hyperparameters, and can achieve good accuracy even without momentum.

6.9 Conclusion

The wildly disparate assumptions made in Byzantine robust learning not only makes comparison between different results impossible, but can also mask unexpected sources of failure. In this work, we strongly advocated for providing end to end convergence guarantees

under realistic assumptions. We provided well-justified notions of a Byzantine robust aggregator and formalized the Byzantine robust stochastic optimization problem. Our theoretical lens led us to a surprisingly simple yet highly effective pair of strategies: using centered clipping and worker momentum. These strategies were thoroughly tested on a variety of attacks and shown to consistently outperform all baselines.

Acknowledgment. We thank Dan Alistarh for useful comments and Eduard Gorbunov for pointing a mistake in our earlier proof of centered clipping. This work is partly supported by a Google Focused Research Award.

7 Byzantine-Robust Learning on Heterogeneous Datasets via Resampling

7.1 Preface

Contribution and sources. This chapter reproduces (Karimireddy et al., 2021a). The theory, toy experiments, and writing were carried out mostly by the author. Lie He conducted most of the experiments and came up with the initial idea for using resampling. Detailed individual contributions:

SPK (author): Conceptualization (50%), Methodology, Formal analysis, Writing – original draft preparation (70 %)

Lie He: Conceptualization (50%), Software, Writing – original draft preparation (30 %)

Martin Jaggi: Supervision, Administration, Writing – review and editing .

Summary. Algorithms for Byzantine robust distributed or federated learning typically assume that the workers are identical. In such a case, using worker momentum is sufficient to reduce the variance, and hence the inter-worker heterogeneity. However, in most real world settings the workers data is heterogeneous (non-iid).

In this chapter, we will see how to design new attacks in such settings which circumvent current defenses and lead to significant loss of performance. We then propose a simple resampling scheme that adapts existing robust algorithms to heterogeneous datasets at a negligible computational cost. We demonstrate (theoretically and experimentally) that combining resampling with existing robust algorithms is effective against challenging attacks. Our work also shows that having over-parameterized models, when combined with robust aggregation rules, is very beneficial for heterogeneous Byzantine robust optimization.

7.2 Introduction

Distributed or federated machine learning, where the data is distributed across multiple workers, has become an increasingly important learning paradigm both due to growing sizes of datasets, as well as data privacy concerns. In such a setting, the workers collaborate to train a single model without directly transmitting their training data (McMahan et al., 2017; Bonawitz et al., 2019; Kairouz et al., 2019). However, by decentralizing the training across a vast number of workers we potentially open ourselves to new security threats. Due to the presence of agents in the network which are actively malicious, or simply due to system and network failures, some workers may disobey the protocols and send arbitrary messages; such workers are also known as *Byzantine* workers (Lamport et al., 2019). Byzantine robust optimization algorithms attempt to combine the updates received from the workers using robust aggregation rules and ensure that the training is not impacted by the presence of a small number of malicious workers.

While this problem has received significant recent attention due to its importance, (Blanchard et al., 2017; Yin et al., 2018a; Alistarh et al., 2018; Karimireddy et al., 2021b), most of the current approaches assume that the data present on each different worker has identical distribution. This assumption is very unrealistic in practice and heterogeneity is inherent in distributed and federated learning (Kairouz et al., 2019). In this work, we show that existing Byzantine aggregation rules catastrophically fail with very simple attacks (or sometimes even with no attacks) in realistic settings. We carefully examine the causes of these failures, and propose a simple solution which provably solves the Byzantine resilient optimization problem under heterogeneous workers.

Concretely, our contributions in this work are summarized below

- We show that when the data across workers is heterogeneous, existing aggregation rules fail to converge, even when no Byzantine adversaries are present. We also propose a simple new attack, *mimic*, which explicitly takes advantage of data heterogeneity and circumvents median-based defenses. Together, these highlight the fragility of existing methods in real world settings.
- We then propose a simple fix - a new resampling step which can be used before any existing aggregation rule. We introduce a formal notion of a robust aggregator (ARAGG) and prove that existing methods like KRUM, coordinate-wise median (CM), and geometric median aka robust federated averaging (RFA)—though insufficient on their own—become provably robust aggregators when augmented with our resampling.
- We combine our notion of robust aggregator (ARAGG) with worker momentum to obtain optimal rates for Byzantine robust optimization with matching lower bounds. Unfortunately, our lower bounds imply that convergence to an exact optimum may not be possible due to heterogeneity. We then circumvent this lower bound and show that when heterogeneity is mild (or when the model is overparameterized), we can in fact converge to an exact optimum. This is the first result establishing convergence to the optimum for

heterogeneous Byzantine robust optimization.

- Finally, we evaluate the effect of the proposed techniques (resampling and worker momentum) against known and new attacks showcasing drastic improvement on realistic heterogeneously distributed datasets.

Setup and notations. Suppose that of the total n workers, the set of good workers is denoted by $\mathcal{G} \subseteq \{1, \dots, n\}$. Our objective is to minimize

$$f(\mathbf{x}) := \frac{1}{|\mathcal{G}|} \sum_{i \in \mathcal{G}} \{f_i(\mathbf{x}) := \mathbb{E}_{\xi_i} [F_i(\mathbf{x}; \xi_i)]\} \quad (7.1)$$

where f_i is the loss function on worker i defined over its own (heterogeneous) data distribution ξ_i . The (stochastic) gradient computed by a good worker $i \in \mathcal{G}$ over minibatch ξ_i is given as $\mathbf{g}_i(\mathbf{x}, \xi_i) := \nabla F_i(\mathbf{x}; \xi_i)$. The noise in every stochastic gradient is independent, unbiased with $\mathbb{E}_{\xi_i} [\mathbf{g}_i(\mathbf{x}, \xi_i)] = \nabla f_i(\mathbf{x})$, and has bounded variance $\mathbb{E}_{\xi_i} \|\mathbf{g}_i(\mathbf{x}, \xi_i) - \nabla f_i(\mathbf{x})\|^2 \leq \sigma^2$. Further, we assume that the data heterogeneity across the workers can be bounded as

$$\mathbb{E}_{j \sim \mathcal{G}} \|\nabla f_j(\mathbf{x}) - \nabla f(\mathbf{x})\|^2 \leq G^2, \quad \forall \mathbf{x}.$$

We write \mathbf{g}_i^t or simply \mathbf{g}_i instead of $\mathbf{g}_i(\mathbf{x}^t, \xi_i^t)$ when there is no ambiguity.

Byzantine attack model. The set of Byzantine workers $\mathcal{B} \subset [n]$ is fixed over time, with the remaining workers \mathcal{G} being good, i.e. $[n] = \mathcal{B} \uplus \mathcal{G}$. We write δ for the fraction of Byzantine workers, $|\mathcal{B}| =: f \leq \delta n$. The Byzantine workers can deviate arbitrarily from our protocol, sending any update to the server. Further, they can collude and may even know the states of all other workers.

7.3 Related work

IID defenses. There has been a significant amount of recent work on the case when all workers have identical data distributions (Blanchard et al., 2017; Chen et al., 2017; Mhamdi et al., 2018; Alistarh et al., 2018; Mhamdi et al., 2018; Yin et al., 2018a,b; Su and Xu, 2018; Damaskinos et al., 2019; Karimireddy et al., 2021b). We discuss the most pertinent of these methods next. Blanchard et al. (2017) initiated the study of Byzantine robust learning and proposed a distance-based aggregation approach KRUM which selects a worker whose gradient is very close to at least half the other workers using $\mathcal{O}(n^2)$ computation, subsequently extended in (Mhamdi et al., 2018). A different approach involves using the median and its variants—Yin et al. (Yin et al., 2018a) propose to use and analyze the coordinate-wise median (CM), and Pillutla et al. (Pillutla et al., 2019) use geometric median with a smoothed version of Weiszfeld’s algorithm. The advantage of these approaches is their linear $\mathcal{O}(n)$ computational cost. In a third approach, (Bernstein et al., 2018) propose to use the signs of gradients and then aggregate them by majority vote, however, (Karimireddy et al., 2019) show that it may fail to converge. Most recently, (Alistarh et al., 2018; Allen-Zhu et al., 2021; Mhamdi et al., 2021; Karimireddy et al., 2021b) showcase how to use past gradients to more accurately filter iid

Byzantine workers. In particular, our work builds on top of [Karimireddy et al. \(2021b\)](#) who prove that momentum combined with centered clipping can solve the iid Byzantine robust optimization. Our work can be seen as an extension of theirs to the non-iid case.

IID vs. Non-IID attacks. Many attacks have been devised for distributed training. For the iid setting, the state-of-the-art attacks are ([Baruch et al., 2019](#); [Xie et al., 2020](#)). These are time-coupled attacks where the attackers introduce a small but consistent bias at every step. This bias goes undetected in any particular round, but accumulates over time and eventually leads to divergence, so breaking most prior robust methods. ([Karimireddy et al., 2021b](#)) show that worker momentum provably overcomes such time-coupled attacks in the iid setting. In contrast, our work focuses on developing attacks (and defenses) which specifically take advantages of the non-iid setting. The non-iid setting also enables targeted backdoor attacks. These attacks are designed to take advantage of heavy-tailed data and manipulate model inference on a small specific subset of data, rather than lower the overall accuracy of training ([Bagdasaryan et al., 2020](#); [Bhagoji et al., 2018](#)). This is a challenging and open problem ([Sun et al., 2019](#); [Wang et al., 2020a](#)). However, our focus is on the overall test/train accuracy of the trained model and not on a worst-case subset as is considered by backdoor attacks.

Non-IID defenses. To the best of our knowledge, only ([Li et al., 2019a](#); [Ghosh et al., 2019](#); [Sattler et al., 2020](#); [Data and Diggavi, 2020, 2021](#)) explicitly investigate Byzantine robustness with non-iid workers. ([Li et al., 2019a](#)) proposes an SGD variant (RSA) which modifies the original objective by adding an ℓ_1 penalty, though its finite time convergence results seem incomparable to the standard SGD analysis. ([Ghosh et al., 2019](#); [Sattler et al., 2020](#)) assume that all workers belong to an apriori fixed number of clusters and use an outlier-robust clustering method to recover these clusters. If we assume that the server has the entire training dataset and can control the distribution of samples to good workers, ([Xie et al., 2019](#); [Chen et al., 2018b](#); [Rajput et al., 2019](#)) show that non-iid-ness can be overcome. Typical examples of this are centrally-coordinated parallel training of neural networks on public cloud, or volunteer computing ([Miura and Harada, 2015](#)). However, none of these methods are applicable in the standard federated learning setup we consider here. We aim to minimize the original loss function over workers while respecting the non-iid data locality, i.e. the dataset remains distributed over the workers in a non-iid fashion and there is no transfer of raw data. This is partially tackled in ([Data and Diggavi, 2020, 2021](#)) who analyze spectral methods for robust optimization. However, these methods require $\Omega(d^2)$ time (quadratic in the dimension), making them infeasible for large scale optimization.

In an independent recent work, ([Acharya et al., 2021](#)) analyze the performance of a computationally efficient variant of geometric median on non-iid data. Their novel idea is to use sparsified gradients to simplify the task of identifying the good workers. However, they do not defend against time coupled attacks (cf. ([Karimireddy et al., 2021b](#))), and their analysis neither proves convergence to the optimum nor recovers the standard rate of SGD when $\delta = 0$. In contrast, our analysis of geometric median shows i) how to get optimal robust aggregation by combining it with resampling, and ii) how worker momentum and overparameterization

give provable convergence guarantees even against time coupled attacks. Our results are also much more general, applying to many other aggregators.

7.4 Attacks against existing aggregation schemes

In this section we show that when the data across the workers is heterogeneous (non-iid), then we can design simple new attacks which take advantage of the heterogeneity, leading to the failure of existing aggregation schemes. We study three representative and widely used defenses:

Krum. For $i \neq j$, let $i \rightarrow j$ denote that \mathbf{x}_j belongs to the $n - f - 2$ closest vectors to \mathbf{x}_i . Then,

$$\text{KRUM}(\mathbf{x}_1, \dots, \mathbf{x}_n) := \underset{i}{\operatorname{argmin}} \sum_{i \rightarrow j} \|\mathbf{x}_i - \mathbf{x}_j\|^2.$$

Krum is computationally expensive, requiring $\mathcal{O}(n^2)$ work by the server (Blanchard et al., 2017). **CM.** Coordinate-wise median computes for the k th coordinate:

$$[\text{CM}(\mathbf{x}_1, \dots, \mathbf{x}_n)]_k := \text{median}([\mathbf{x}_1]_k, \dots, [\mathbf{x}_n]_k) = \underset{i}{\operatorname{argmin}} \sum_{j=1}^n |[x_i]_k - [x_j]_k|.$$

Coordinate-wise median is fast to implement requiring only $\mathcal{O}(n)$ time (Chen et al., 2017).

RFA. Robust federated averaging (RFA) computes the geometric median

$$\text{RFA}(\mathbf{x}_1, \dots, \mathbf{x}_n) := \underset{\mathbf{v}}{\operatorname{argmin}} \sum_{i=1}^n \|\mathbf{v} - \mathbf{x}_i\|_2.$$

While the geometric median has no closed form solution, (Pillutla et al., 2019) approximate it using multiple iterations of smoothed Weiszfeld algorithm, each of which requires $\mathcal{O}(n)$ computation.

7.4.1 Failure on imbalanced data without Byzantine workers

We show that when the data amongst the workers is imbalanced, existing aggregation rules *fail* even in the *absence* of any Byzantine workers. Algorithms like KRUM select workers who are *representative* of a majority of the workers by relying on statistics such as pairwise differences between the various worker updates. Our key insight is that when the data across the workers is heterogeneous, there is no single worker who is representative of the whole dataset. This is because each worker computes their local gradient over vastly different local data. Hence, for convergence it is important to not only select a good (non-Byzantine) worker, but also ensure that each of the good workers is selected with roughly equal frequency. Hence KRUM suffers a significant loss in performance with heterogeneous data, even when there are *no Byzantine workers*.

Example. Suppose that there are $2n + 1$ workers with worker i holding $(-1)^i \in \{\pm 1\}$. This means that the true mean is ≈ 0 , but KRUM, CM, and RFA will output ± 1 . This motivates our

Chapter 7. Byzantine-Robust Learning on Heterogeneous Datasets via Resampling

Table 7.1 – Test accuracy (%) with no Byzantine workers ($\delta = 0$) on imbalanced data ($\alpha = 500$).

Aggr	iid	non-iid
AVG	98.84 \pm 0.08	98.84 \pm 0.07
KRUM	98.10 \pm 0.14	82.97 \pm 3.64
CM	97.82 \pm 0.20	80.36 \pm 0.04
RFA	98.72 \pm 0.11	84.76 \pm 0.83
CCLIP	98.76 \pm 0.10	98.15 \pm 0.19

Table 7.2 – Test accuracy (%) under mimic attack with $\delta = 0.2$ fraction of Byzantine workers.

	iid	non-iid
AVG	92.6 \pm 0.4	92.6 \pm 0.4
KRUM	90.4 \pm 0.4	39.0 \pm 7.4
CM	91.0 \pm 0.3	54.2 \pm 9.6
RFA	93.1 \pm 0.3	76.4 \pm 1.6
CCLIP	93.2 \pm 0.4	85.5 \pm 0.8

next attack.

In Table 7.1, we demonstrate such failures by training on MNIST with $n = 20$ and $\delta = 0$. We construct an imbalanced dataset where each successive class has only a fraction of samples of the previous class. We defer details of the experiments to Section 14.1. As we can see in Table 7.1, KRUM, CM and RFA match the ideal performance of SGD in the iid case, but only attain around 80% accuracy in the non-iid case. This corresponds to learning only the top 2–3 classes and ignoring the rest 7–8 classes.

A similar phenomenon was observed when using batch-size 1 in the iid case by (Karimireddy et al., 2021b). However, in the iid case this can be easily overcome by increasing the batch-size. In contrast, when the data across the works is non-iid (e.g. split by class), increasing the batch-size does *not* make the worker gradients any more similar and there is a big drop in performance. Finally, note that hitherto new algorithm (CCLIP) maintains its performance both in the iid and the non-iid setting. We will explore this in more detail in Section 7.5.

7.4.2 Mimic attack on balanced data

Previously, we saw how data imbalance could lead to consistent errors in the aggregation rules, leading to significant loss in accuracy. In this section, we will propose a new attack *mimic* which specifically tries to maximize the perceived data imbalance even if the original data is balanced.

Example. Each of the good workers $i \in \mathcal{G} \subseteq [n]$ has an input a $\mathbf{x}_i \in \{\pm 1\}^d$ where each coordinate is an independent Rademacher random variable. The inputs then have mean $\mathbf{0}$ and variance $\mathbb{E}\|\mathbf{x}_i\|^2 = \rho^2 = d$. Now, the Byzantine attackers $j \in \mathcal{B}$ have dual goals: i) escape detection, and ii) increase data imbalance. For this, we propose the following simple passive attack: pick some fixed worker $i_\star \in \mathcal{G}$ (say 1) and every Byzantine worker $j \in \mathcal{B}$ outputs $\mathbf{x}_j = \mathbf{x}_1$. It is im-

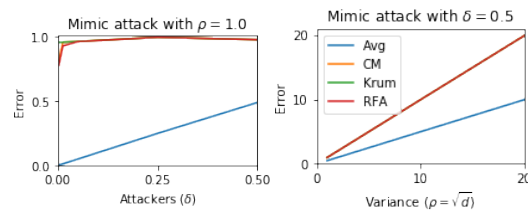


Figure 7.1 – Error with random vectors with variance $\rho^2 = d$ and δ fraction of Byzantine workers imitating a fixed good worker (say worker $1 \in \mathcal{G}$). RFA performs slightly better than CM and Krum, but all have *higher* error than simply averaging across various settings of δ and ρ .

possible to detect the attackers since they imitate an existing good worker, but still cause an imbalance in the data distribution. This serves as the intuition for our attack.

Mimic attack. All Byzantine workers pick a good worker (say i_\star) to mimic and copy its output ($\mathbf{x}_{i_\star}^t$). This inserts a consistent bias towards over-emphasizing worker i_\star and thus under-representing other workers. Since the attack simply mimics another worker, it is impossible to distinguish the Byzantine worker from a real worker and hence is cannot be filtered out. To pick i_\star , we use an initial phase ($\mathcal{J}^0 \approx 1$ epoch) to compute a direction \mathbf{z} of maximum variance of the outputs of the good workers:

$$\mathbf{z} = \operatorname{argmax}_{\|\mathbf{z}\|=1} \mathbf{z}^\top \left(\sum_{t \in \mathcal{J}_0} \sum_{i \in \mathcal{G}} (\mathbf{x}_i^t - \boldsymbol{\mu})(\mathbf{x}_i^t - \boldsymbol{\mu})^\top \right) \mathbf{z} \quad \text{where} \quad \boldsymbol{\mu} = \frac{1}{|\mathcal{G}||\mathcal{J}_0|} \sum_{i \in \mathcal{G}, t \in \mathcal{J}_0} \mathbf{x}_i^t.$$

Then we pick a worker i_\star to mimic by computing

$$i_\star = \operatorname{argmax}_{i \in \mathcal{G}} \left| \sum_{t \in \mathcal{J}_0} \mathbf{z}^\top \mathbf{x}_i^t \right|.$$

Efficient streaming algorithms for computing \mathbf{z} and i_\star are discussed in Appendix 14.2. This procedure tries to insert bias along a direction which has the largest across-worker variance.

Table 7.2 shows the effectiveness of mimic attack even when the fraction of Byzantine nodes is small (i.e. $n = 25$, $|\mathcal{B}| = 5$). Since the attackers are copying good workers, it is very challenging to detect them. Note that this attack specifically targets the non-iid nature of the data—all robust aggregators maintain their performance in the iid setting and only suffer in the non-iid setting. Their performance is in fact worse than even simply averaging. As predicted by our example, KRUM and CM have the worst performance and RFA performs slightly better (though still significantly worse than simply averaging). We will discuss the remarkable performance of CCLIP in the next section.

7.5 Constructing an agnostic robust aggregator using resampling

In Section 7.4 we demonstrated how existing aggregation rules fail in realistic non-iid scenarios, with and without attackers. In this section, we show how using resampling can provably fix such aggregation rules. The underlying reason for this failure, as we saw previously, is that the existing methods fixate on the contribution of only the most likely worker, and ignore the contributions from the rest. To overcome this issue, we propose to use resampling which ‘mixes’ the data from all the workers thereby reducing the chance of any subset of the data being consistently ignored.

Algorithm 9 Robust Aggregation (ARAGG) using resampling

- 1: **input** $\{\mathbf{x}_1, \dots, \mathbf{x}_n\}$, $s \in \mathbb{N}$, aggregation rule AGGR
 - 2: construct $\{\mathbf{v}_1, \dots, \mathbf{v}_{s \cdot n}\}$ with $\mathbf{v}_k \leftarrow \mathbf{x}_{\lceil k/s \rceil}$ // replicate s times
 - 3: pick random permutation π of $[s \cdot n]$
 - 4: compute $\mathbf{y}_i \leftarrow \frac{1}{s} \sum_{k=s(i-1)+1}^{s \cdot i} \mathbf{v}_{\pi(k)}$ for $i \in [n]$ // samples with s -replacement
 - 5: **output** $\hat{\mathbf{x}} \leftarrow \text{AGGR}(\mathbf{y}_1, \dots, \mathbf{y}_n)$ // aggregate after resampling
-

7.5.1 Resampling algorithm

Given n inputs $\mathbf{x}_1, \dots, \mathbf{x}_n$, we replicate each of the \mathbf{x}_i s times and then randomly split them into n equally sized buckets. This effectively performs *s-resampling without replacement* where each \mathbf{x}_i can be sampled at most s times. Then, the contents of each bucket are averaged to construct $\{\mathbf{y}_1, \dots, \mathbf{y}_n\}$ which are then input to an agnostic aggregator AGGR. The details are summarized in Algorithm 9. The key property of our approach is that after resampling, the resulting set of averaged $\{\mathbf{y}_1, \dots, \mathbf{y}_n\}$ are much more homogeneous (lower variance) than the original inputs. Thus, when fed into existing aggregation schemes, the chance of success increases. We formalize this in the following simple lemma.

Lemma 19 (Resampling reduces variance). *Suppose we are given n independent (but not identical) random vectors $\{\mathbf{x}_1, \dots, \mathbf{x}_n\}$ such that a good subset $\mathcal{G} \subseteq [n]$ of size at least $|\mathcal{G}| \geq n(1 - \delta)$ satisfies:*

$$\mathbb{E} \|\mathbf{x}_i - \mathbf{x}_j\|^2 \leq \rho^2, \quad \text{for any fixed } i, j \in \mathcal{G}.$$

Define $\bar{\mathbf{x}} := \frac{1}{|\mathcal{G}|} \sum_{j \in \mathcal{G}} \mathbf{x}_j$. Let the outputs after s -resampling without replacement be $\{\mathbf{y}_1, \dots, \mathbf{y}_n\}$. Then, a subset of the outputs $\tilde{\mathcal{G}} \subseteq [n]$ of size at least $|\tilde{\mathcal{G}}| \geq n(1 - \delta s)$ satisfies

$$\mathbb{E}[\mathbf{y}_i] = \mathbb{E}[\bar{\mathbf{x}}] \quad \text{and} \quad \mathbb{E} \|\mathbf{y}_i - \mathbf{y}_j\| \leq \rho^2 / s \quad \text{for any fixed } i, j \in \tilde{\mathcal{G}}.$$

The expectation in the above lemma is taken both over the random vectors as well as over the randomness of the resampling procedure.

Remark 20. Lemma 19 proves that after our resampling procedure, we are left with outputs \mathbf{y}_i which have i) pairwise variance reduced by s , and ii) potentially s times more Byzantine vectors. Hence, resampling trades off increasing influence of Byzantine inputs against having more homogeneous vectors. Using $s = 1$ simply shuffles the inputs and leaves them otherwise unchanged.

Instead of sampling without replacement as we do here, one could alternatively also try sampling with replacement. However, this does not give an almost sure bound on the influence of the Byzantine outputs. We explore this option more in the Appendix.

7.5.2 Agnostic robust aggregation

We now define what it means for an agnostic robust aggregator to succeed.

7.5. Constructing an agnostic robust aggregator using resampling

Definition G ((δ_{\max}, c) -ARAGG). Suppose we are given input $\{\mathbf{x}_1, \dots, \mathbf{x}_n\}$ of which a subset \mathcal{G} of size at least $|\mathcal{G}| > (1 - \delta)n$ for $\delta \leq \delta_{\max} \leq 0.5$ and satisfies $\mathbb{E}\|\mathbf{x}_i - \mathbf{x}_j\|^2 \leq \rho^2$. Then, the output $\hat{\mathbf{x}}$ of a Byzantine robust aggregator satisfies:

$$\mathbb{E}\|\hat{\mathbf{x}} - \bar{\mathbf{x}}\|^2 \leq c\delta\rho^2 \quad \text{where} \quad \hat{\mathbf{x}} = \text{ARAGG}_\delta(\mathbf{x}_1, \dots, \mathbf{x}_n).$$

Further, ARAGG does not need to know ρ^2 (only δ), and automatically adapts to any value ρ^2 .

We require that the robust aggregator is *agnostic* to the value of ρ^2 and automatically adjusts its output to the current ρ during training. The aggregator can take δ as an input though. This property is very useful in the context of Byzantine robust optimization since the variance ρ^2 keeps changing over the training period, whereas the fraction of Byzantine workers δ remains constant. This is a major difference from the definition used in (Karimireddy et al., 2021b).

Our robust aggregator is parameterized by $\delta_{\max} \leq 0.5$ which denotes the maximum amount of Byzantine inputs it can handle, and a constant c which determines its performance. If $\delta = 0$, i.e. when there are no Byzantine inputs, we are guaranteed to exactly recover the true average $\bar{\mathbf{x}}$. Exact recovery is also guaranteed when $\rho = 0$ since in that case it is easy to identify the good inputs since they are all equal and in majority. When both $\rho > 0$ and $\delta > 0$, we recover the average up to an additive error term.

We next show that aggregators which we saw were not robust in Section 7.4, can be made to satisfy Definition G by combining with resampling.

Theorem XXI. Suppose we are given n inputs $\{\mathbf{x}_1, \dots, \mathbf{x}_n\}$ satisfying properties in Lemma 19 for $\delta < \delta_{\max}$. Then, running Algorithm 9 with $s = \delta_{\max}/\delta$ yields the following:

- *Krum*: $\mathbb{E}\|\text{KRUM} \circ \text{RESAMPLE}(\mathbf{x}_1, \dots, \mathbf{x}_n) - \bar{\mathbf{x}}\|^2 \leq \mathcal{O}(\delta\rho^2)$ with $\delta_{\max} < 1/4$.
- *Geometric median*: $\mathbb{E}\|\text{RFA} \circ \text{RESAMPLE}(\mathbf{x}_1, \dots, \mathbf{x}_n) - \bar{\mathbf{x}}\|^2 \leq \mathcal{O}(\delta\rho^2)$ with $\delta_{\max} < 1/2$.
- *Coordinate-wise median*: $\mathbb{E}\|\text{CM} \circ \text{RESAMPLE}(\mathbf{x}_1, \dots, \mathbf{x}_n) - \bar{\mathbf{x}}\|^2 \leq \mathcal{O}(d\delta\rho^2)$ with $\delta_{\max} < 1/2$.

Thus, all the methods which we saw were broken in Section 7.4 upon combining with resampling satisfy our notion of an *agnostic* Byzantine robust aggregator (Definition G). This is because both our resampling procedures as well as the underlying aggregators are independent of ρ^2 . Further, our error is $\mathcal{O}(\delta\rho^2)$. In contrast, most other analysis only obtain an error of $\mathcal{O}(\rho^2)$ (cf. (Acharya et al., 2021)).

The error of CM depends on the dimension d which is problematic when $d \gg n$. However, we suspect this is because we measure stochasticity using Euclidean norms instead of coordinate-wise. In practice, we found that CM often outperforms KRUM, with RFA outperforming them both. Note that we select $s = \delta_{\max}/\delta$ to ensure that after resampling, we have the maximum amount of Byzantine inputs tolerated by the method with $(s\delta) = \delta_{\max}$.

Remark 21 (Centered clipping). The centered clipping aggregator (CCLIP) given a clipping radius τ and an initial guess \mathbf{v} of the average $\bar{\mathbf{x}}$ performs

Algorithm 10 Robust Optimization using Robust Aggregator

Input: ARAGG, η , β

```

1: for  $t = 1, \dots$  do
2:   for worker  $i \in [n]$  in parallel
3:      $\mathbf{g}_i \leftarrow \nabla F_i(\mathbf{x}, \xi_i)$  and  $\mathbf{m}_i \leftarrow (1 - \beta)\mathbf{g}_i + \beta\mathbf{m}_i$  // worker momentum
4:     send  $\mathbf{m}_i$  if  $i \in \mathcal{G}$ , else send * if Byzantine
5:    $\hat{\mathbf{m}} = \text{ARAGG}(\{\mathbf{m}_1, \dots, \mathbf{m}_n\})$ .
6:    $\mathbf{x} \leftarrow \mathbf{x} - \eta\hat{\mathbf{m}}$  // update params using robust aggregate
7: end for
```

$$\text{CCLIP}(\mathbf{x}_1, \dots, \mathbf{x}_n) = \mathbf{v} + \frac{1}{n} \sum_{i \in [n]} (\mathbf{x}_i - \mathbf{v}) \min(1, \tau / \|\mathbf{x}_i - \mathbf{v}\|_2).$$

Karimireddy et al. (2021b) prove that CCLIP even without resampling satisfies Definition G with $\delta_{\max} = 0.1$, and $c = \mathcal{O}(1)$. This explains its good performance on non-iid data in Section 7.4. However, CCLIP is not agnostic since it requires clipping radius τ as an input which in turn depends on ρ^2 . Devising a version of CCLIP which automatically adapts its clipping radius is an important open question. Empirically however, we observe that simple rules for setting τ work quite well—we always use $\tau = \frac{10}{1-\beta}$ in our limited experiments where β is the coefficient of momentum.

Note that Lemma 19 reduces ρ^2 by s but simultaneously increases δ by s . Thus, any robust aggregator which already satisfies Definition G when combined with resampling leaves the final error $\delta\rho^2$ unchanged i.e. resampling may not improve an already robust aggregator in theory. Empirically however, we observe that resampling always improves performance.

While we have shown how to construct a robust aggregator which satisfies some notion of a robustness, we haven't yet seen how this affects the Byzantine robust *optimization* problem. We investigate this question theoretically in the next section and empirically in Section 7.7. In fact, we will see that sometimes resampling combined with non-iid data yields better results than in the iid case.

7.6 Robust non-iid optimization using a robust aggregator

In this section, we study the problem of optimization in the presence of Byzantine workers and heterogeneity, given access to any robust aggregator satisfying Definition G. We then show that data heterogeneity makes Byzantine robust optimization especially challenging and prove lower bounds for the same. Finally, we see how mild heterogeneity, or sufficient overparameterization can circumvent these lower bounds, obtaining convergence to the optimum.

7.6.1 Algorithm description

In Section 7.5 we saw that resampling could tackle heterogeneity across the workers by reducing G^2 . However, there still remains variance σ^2 in the gradients within each worker since each worker uses stochastic gradients. To reduce the effect of this variance, we rely on worker momentum. Each worker sends their local worker momentum vector \mathbf{m}_i to be aggregated by ARAGG instead of \mathbf{g}_i directly:

$$\begin{aligned}\mathbf{m}_i^t &= \beta \mathbf{m}_i^{t-1} + (1 - \beta) \mathbf{g}_i(\mathbf{x}^{t-1}) \quad \text{for every } i \in \mathcal{G}, \\ \mathbf{x}^t &= \mathbf{x}^{t-1} - \eta \text{ARAGG}(\mathbf{m}_1^t, \dots, \mathbf{m}_n^t).\end{aligned}$$

This is a non-standard description of momentum, but is equivalent up to a rescaling of the learning rate η . Intuitively, using worker momentum \mathbf{m}_i with parameter β averages over $1/(1-\beta)$ stochastic gradients \mathbf{g}_i and reduces the effect of the within-worker-variance σ^2 (Karimireddy et al., 2021b). Note that the resulting $\{\mathbf{m}_i\}$ are *still heterogeneous* across the workers with variance G^2 and this is the key challenge we face.

7.6.2 Convergence rates

We now turn towards proving convergence rates for our resampling aggregation method Algorithm 9 based on any existing aggregator AGGR. We will assume that for any fixed $i \in \mathcal{G}$, the stochastic gradients computed satisfy

$$\mathbb{E}_{\xi_i} \|\mathbf{g}_i(\mathbf{x}) - \nabla f_i(\mathbf{x})\|^2 \leq \sigma^2 \text{ and } \mathbb{E}_{j \sim \mathcal{G}} \|\nabla f_j(\mathbf{x}) - \nabla f(\mathbf{x})\|^2 \leq G^2, \quad \forall \mathbf{x}. \quad (7.2)$$

This first condition bounds the variance of the stochastic gradient within a worker whereas the latter is a standard measure of inter-client heterogeneity in federated learning (Yu et al., 2019b; Khaled et al., 2020; Karimireddy et al., 2020b). Under these conditions, we can prove the following.

Theorem XXII. *Suppose we are given a (δ_{\max}, c) -ARAGG satisfying Definition G, and n workers of which a subset \mathcal{G} of size at least $|\mathcal{G}| \geq n(1 - \delta)$ faithfully follow the algorithm for $\delta \leq \delta_{\max}$. Further, for any good worker $i \in \mathcal{G}$ let f_i be a possibly non-convex function with L -Lipschitz gradients, and the stochastic gradients on each worker be independent, unbiased and satisfy (7.2). Then, for $F^0 := f(\mathbf{x}^0) - f^*$, the output of Algorithm 10 satisfies*

$$\frac{1}{T} \sum_{t=1}^T \mathbb{E} \|\nabla f(\mathbf{x}^{t-1})\|^2 \leq \mathcal{O} \left(c\delta G^2 + \sigma \sqrt{\frac{LF^0}{T} (c\delta + 1/n)} + \frac{LF^0}{T} \right).$$

Remark 22 (Unified proofs). Remark 21 shows that CCLIP is a robust aggregator, and Theorem XXI shows KRUM, RFA, and CM on combining with sufficient resampling are all robust aggregators satisfying Definition G. Most of these methods had no end-to-end convergence guarantees prior to our results. Thus, Theorem XXII gives the first unified analysis in both the iid and non-iid settings.

When $\delta = 0$ i.e. there are no Byzantine workers, the above rate recovers the familiar $\mathcal{O}(\frac{\sigma}{\sqrt{Tn}})$ rate which is optimal for non-convex SGD and even has linear speed-up with respect to the n workers. In contrast, all previous algorithms for non-iid data (e.g. (Data and Diggavi, 2021; Acharya et al., 2021)) do not recover convergence even when $\delta = 0$. This is also empirically reflected in Section 7.4.1, where we showed how these algorithms can fail even in the absence of Byzantine workers ($\delta = 0$).

Further, in the iid case when $G = 0$ the rate above simplifies to $\mathcal{O}(\frac{\sigma}{\sqrt{T}} \cdot \sqrt{c\delta + 1/n})$ which matches the optimal iid Byzantine robust rates of (Karimireddy et al., 2021b). In both these cases we converge to the optimum and can make the gradient arbitrarily small. However, when $\delta > 0$ and $G > 0$, Theorem XXII only shows convergence to a radius of $\mathcal{O}(\sqrt{\delta}G)$ and not to the actual optimum. We will next explore this limitation.

7.6.3 Lower bounds and the challenge of heterogeneity

Suppose worker j sends us an update which looks ‘weird’ and looks very different from the updates from the rest of the workers. This may be because worker j might be malicious and their update represents an attempted attack. On the other hand, it is also possible that worker j is good but simply has highly *non-representative data*. In the former case the update should be ignored, whereas in the latter, the update represents a valuable source of specialized data. However, it is impossible for the server to distinguish between the two situations. This is the main challenge of Byzantine robust optimization in the face of worker heterogeneity. The above argument can in fact be formalized to prove the following lower bound.

Theorem XXIII. *Given any optimization algorithm ALG, we can find n functions $\{f_1(x), \dots, f_n(x)\}$ of which at least $(1 - \delta)n$ are good (belong to \mathcal{G}), 1-smooth, μ -strongly convex functions, and satisfy $\mathbb{E}_{i \sim \mathcal{G}} \|\nabla f_i(x) - \nabla f(x)\| \leq G^2$ such that the output of ALG given access to these n function has an error at least*

$$\mathbb{E}[f(\text{ALG}(f_1, \dots, f_n)) - f^*] \geq \Omega\left(\frac{\delta G^2}{\mu}\right) \quad \text{and} \quad \mathbb{E}\|\nabla f(\text{ALG}(f_1, \dots, f_n))\|^2 \geq \Omega(\delta G^2).$$

The expectation above is over the potential randomness of the algorithm. This theorem is adapted from arguments for iid robust aggregation (Lai et al., 2016; Karimireddy et al., 2021b) and implies that it is impossible to converge to the true optimum in the presence of Byzantine workers. Note that the above lower bound is information theoretic in nature and is independent of how many gradients are computed or how long the algorithm is run.

Remark 23 (Matches lower bound). *Suppose that we satisfy the heterogeneity condition (7.2) with $G^2 > 0$ and $\sigma = 0$. Then, the rate in Theorem XXII can be simplified to $\mathcal{O}(\delta G^2 + 1/T)$. While the second term in this decays to 0 with T , the first term remains, implying that we only converge to a radius of $\sqrt{\delta}G$ around the optimum. However, this matches our lower bound result from Theorem XXIII and hence is in general unimprovable.*

This is a very strong negative result and seems to indicate that Byzantine robustness might be impossible to achieve in real world federated learning. This would be major stumbling block for deployment since the system would provably be vulnerable to attackers. We will next carefully examine the lower bound and will attempt to circumvent it.

7.6.4 Circumventing lower bounds using overparameterization

We previously saw some strong impossibility results posed by heterogeneity. In this section, we show that while indeed in the worst case being robust under heterogeneity is impossible, we may still converge to the true optimum under more realistic settings. Let us consider a different heterogeneity bound in place of (7.2):

$$\mathbb{E}_{j \sim \mathcal{G}} \|\nabla f_j(\mathbf{x}) - \nabla f(\mathbf{x})\|^2 \leq B^2 \|\nabla f(\mathbf{x})\|^2, \quad \forall \mathbf{x}. \quad (7.3)$$

Note that at the optimum \mathbf{x}^* we have $\nabla f(\mathbf{x}^*) = 0$, and hence this assumption implies that $\nabla f_j(\mathbf{x}^*) = 0$ for all $j \in \mathcal{G}$. This is satisfied if the model is *sufficiently over-parameterized* and typically holds in most realistic settings (Vaswani et al., 2019).

Theorem XXIV. *Suppose we are given a (δ_{\max}, c) -ARAGG and n workers with loss functions $\{f_1, \dots, f_n\}$ satisfying the conditions in Theorem XXII with $\delta \leq \delta_{\max}$ and (7.3) for some $B^2 < \frac{1}{3c\delta}$. Then, for $F^0 := f(\mathbf{x}^0) - f^*$, the output of Algorithm 10 satisfies*

$$\frac{1}{T} \sum_{t=1}^T \mathbb{E} \|\nabla f(\mathbf{x}^{t-1})\|^2 \leq \mathcal{O} \left(\frac{1}{1-3c\delta B^2} \cdot \left(\sigma \sqrt{\frac{LF^0}{T} (c\delta + 1/n)} + \frac{LF^0}{T} \right) \right).$$

Remark 24 (Overparameterization fixes convergence). *The rate in Theorem XXIV not only goes to 0 with T , but also matches that of the optimal iid rate of $\mathcal{O}(\frac{\sigma}{\sqrt{T}} \cdot \sqrt{c\delta + 1/n})$ (Karimireddy et al., 2021b). Thus, using a stronger bound on the heterogeneity allows us to circumvent lower bounds for the non-iid case and converge to a good solution even in the presence of Byzantine workers. This is the first result of its kind, and takes a major step towards practical Byzantine robust methods which work in realistic settings.*

In the overparameterized setting, we can be sure that we will be able to *simultaneously* optimize all worker's losses. Hence, over time the agreement between all worker's gradients increases. This in turn makes any attempts by the attackers to derail training stand out easily, especially towards the end of the training. To take advantage of this increasing closeness, we need an aggregator which automatically adapts the quality of its output as the good workers get closer. Thus, the *agnostic* robust aggregator is crucial to our overparameterized convergence result. We empirically demonstrate the effects of overparameterization in Section 14.1.2.

Remark 25 (History-less robustness for cross-device FL). *In cross-device federated learning studied in (Kairouz et al., 2019; Karimireddy et al., 2020a), there may be thousands of workers which are sampled in an online fashion. In this setting, we never see the same worker twice and so the workers cannot maintain any worker momentum. Instead, in each round*

Table 7.3 – Table 7.1 + Resampling ($s=2$).

Aggr	iid	non-iid
AVG	98.84 \pm 0.08	98.84 \pm 0.07
KRUM	98.44 \pm 0.12	97.79 \pm 0.29
CM	98.30 \pm 0.15	96.44 \pm 0.36
RFA	98.78 \pm 0.06	97.82 \pm 0.31
CCLIP	98.78 \pm 0.07	98.68 \pm 0.05

Table 7.4 – Table 7.2 + Resampling ($s=2$).

Aggr	iid	non-iid
AVG	92.6 \pm 0.4	92.6 \pm 0.4
KRUM	92.0 \pm 0.3	48.5 \pm 0.9
CM	92.2 \pm 0.3	76.1 \pm 18.4
RFA	93.3 \pm 0.3	91.3 \pm 0.3
CCLIP	93.3 \pm 0.4	91.2 \pm 0.3

the sampled workers communicate gradient updates to the server, who then performs robust aggregation using ARAGG (perhaps with additional server momentum). Theorem XXIV guarantees Byzantine robust convergence of this method as well even without worker momentum, if the local worker variance is small ($\sigma^2 = 0$) and we are in the overparameterized setting (7.3). This circumvents the impossibility results in Karimireddy et al. (2021b) who show that without overparameterization, history is necessary for convergence. Thus, we obtain Byzantine robust methods for the important cross-device federated learning setup as well.

7.7 Experiments

In this section, we demonstrate the effects of resampling on datasets distributed in a non-iid fashion. Throughout the section, we illustrate the tasks, attacks, and defenses by an example of training an MLP on a heterogeneous version of the MNIST dataset (LeCun et al., 1998). In Section 14.1, we also provide results of similar experiments on Fashion-MNIST (Xiao et al., 2017). The dataset is sorted by labels and sequentially divided into equal parts among good workers; Byzantine workers have access to the entire dataset. Implementations are based on PyTorch (Paszke et al., 2019) and will be made publicly available. We defer details of setup, implementation, and runtime to Section 14.1.

7.7.1 Resampling against the attacks on non-iid data

In Section 7.4 we have presented how heterogeneous data can lead to failure of existing robust aggregation rules. Here we apply our proposed resampling with $s = 2$ to the same aggregation rules, showing that resampling overcomes the described failures. Results are presented in Table 7.3. Comparing Table 7.3 with Table 7.1, resampling improves the aggregators' top-1 test accuracy on long-tail and non-iid dataset by 10% to 14% and allows them to learn classes at the tail distribution. For non-iid balanced dataset, resampling also greatly improves the performance of KRUM and CM and makes RFA and CCLIP close to ideal performance.

Similarly, combining aggregators with resampling also performs much better on non-iid dataset under mimic attack. In Table 7.4, RFA and CCLIP reach accuracy similar to the iid case, and the performance of KRUM, and CM are improved by 9% to 25%.

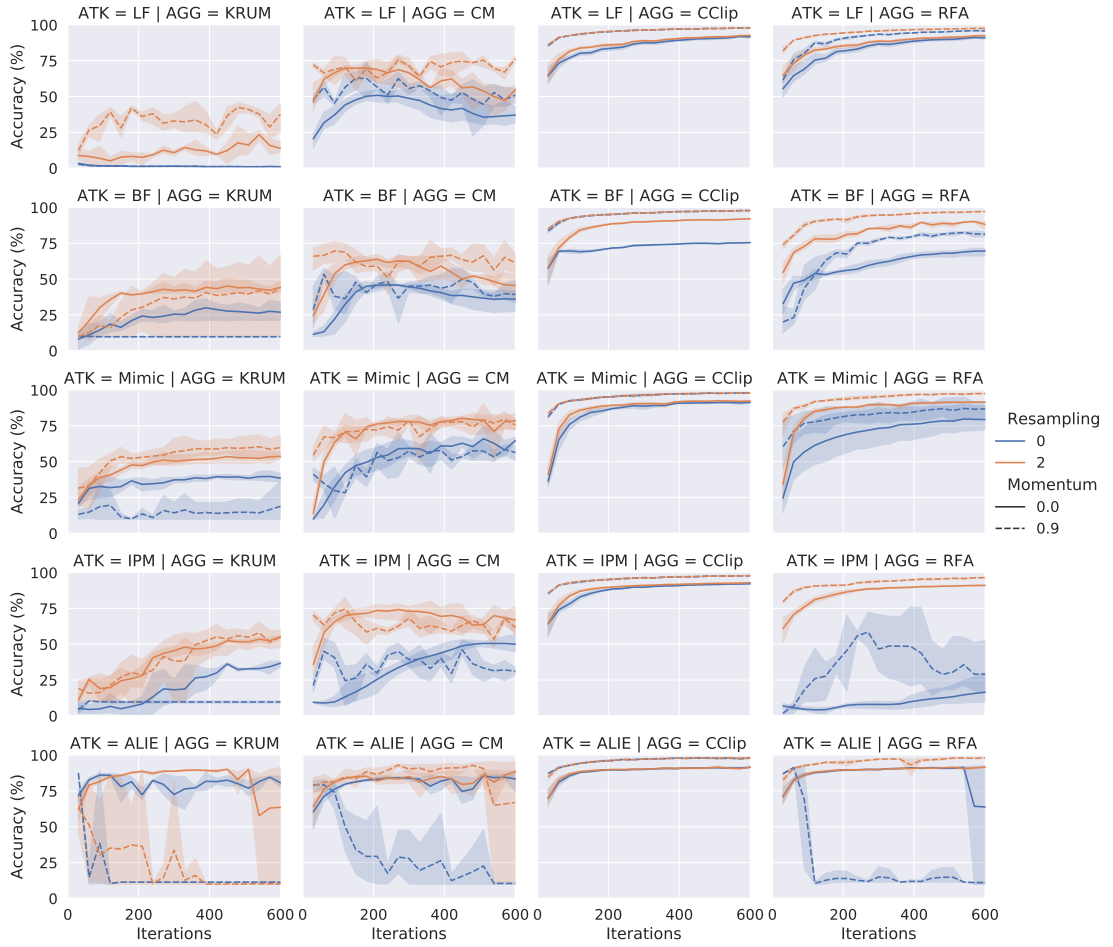


Figure 7.2 – Top-1 test accuracies of KRUM, CM, CCLIP, RFA, under 4 attacks on non-iid datasets.

7.7.2 Resampling against general Byzantine attacks

In Figure 7.2, we present thorough experiments on non-iid data over 25 workers with 5 Byzantine workers, under different attacks. In each subfigure, we compare an aggregation rule with its variant with resampling. The aggregation rules compared are KRUM (Blanchard et al., 2017), CM (Yin et al., 2018a), RFA (Pillutla et al., 2019), CCLIP (Karimireddy et al., 2021b).

Attacks. 4 different kinds of attacks are applied (one per row in the figure): bit flipping (BF), label flipping (LF), *mimic* attack, as well as inner product manipulation (IPM) attack (Xie et al., 2020) and the “a little is enough” (ALIE) attack (Baruch et al., 2019).

- **Bit flipping:** A Byzantine worker flips the sign bits and sends $-\nabla f(\mathbf{x})$ instead of $\nabla f(\mathbf{x})$ because of problems like hardware failures etc.
- **Label flipping:** The dataset on workers have corrupted labels. For the MNIST dataset, we let Byzantine workers transform labels by $\mathcal{T}(y) := 9 - y$.
- **Mimic:** Explained in Section 7.4.2.

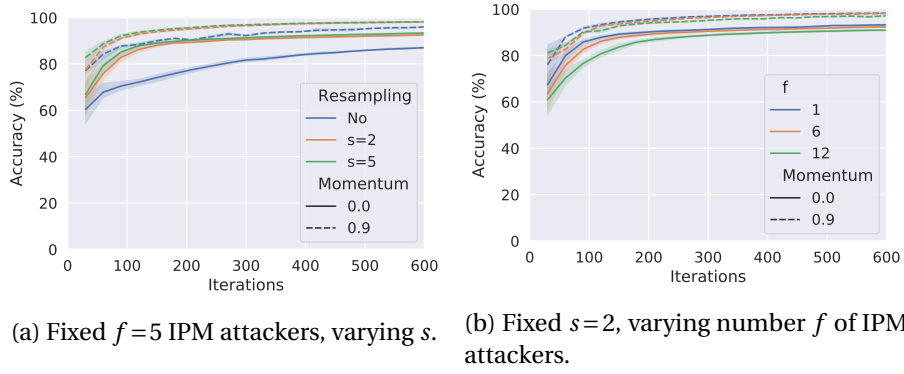


Figure 7.3 – Top-1 accuracies of CCLIP with varying f and s when training on a cluster of $n=53$ nodes

- **IPM:** The attackers send $-\frac{\epsilon}{|\mathcal{G}|} \sum_{i \in \mathcal{G}} \nabla f(\mathbf{x}_i)$ where ϵ controls the strength of the attack.
- **ALIE:** The attackers estimate the mean $\mu_{\mathcal{G}}$ and standard deviation $\sigma_{\mathcal{G}}$ of the good gradients, and send $\mu_{\mathcal{G}} - z\sigma_{\mathcal{G}}$ to the server where z is a small constant controlling the strength of the attack.

Both IPM and ALIE are the state-of-the-art attacks in the iid distributed learning setups which takes advantage of the variances among workers. These attacks are much stronger in the non-iid setup. In the last two row of Figure 7.2 we show that worker momentum and resampling reduce such variance while momentum alone is not enough. Overall, Figure 7.2 shows that resampling improves the performances of almost all aggregators under all kinds of attacks.

It is important to note that clipping radius τ of CCLIP is set to $\frac{10}{1-\beta}$ for all attacks so that we do not finetune on each specific attack. This scaling is required because CCLIP is *not agnostic*. We defer the discussion on the clipping radius and its scaling to Section 14.1.2.

7.7.3 Resampling hyperparameter

Finally we study the influence of the resampling hyperparameter s and the number of Byzantine workers f , again on the heterogeneous MNIST dataset. We use CCLIP as the base aggregator and apply IPM attack to it on non-iid data. In Figure 7.3a, we compare no resampling and resampling with $s=2,5$ and confirm that larger s gives faster convergence but $s=2$ can be good enough. The results in Figure 7.3b shows that $s=2$ still behaves well when increasing f close to 25%. The complete evaluation of the results are deferred to Section 14.1.

7.7.4 Discussion

In all our experiments, we consistently observe: i) mild resampling ($s=2$) improves performance, ii) worker momentum further stabilizes training, and finally iii) CCLIP recovers the ideal performance. Given its ease of implementation, this leads us to strongly recommend using CCLIP in practical federated learning to safeguard against actively malicious agents or passive failures. In all our experiments, using a clipping threshold of $\tau = \frac{10}{1-\beta}$ sufficed. How-

ever, given that our experiments are limited to MNIST, it is possible that τ may need to be tuned in other settings. If the default setting of τ doesn't work and tuning it is impossible, RFA combined with resampling and worker momentum also nearly recovers ideal performance and can instead be used. However, RFA has a higher computational and communication cost. Ideally, one would want to *adaptively* and automatically set the clipping radius τ so that it works in all instances without any tuning. Designing such a clipping operator as well as its large scale empirical study is left for future work.

7.8 Conclusion

Heterogeneity poses unique challenges for Byzantine robust optimization. The first challenge is that existing defenses attempt to pick a “representative” update, which may not exist in the non-iid setting. This, we showed, can be overcome by using resampling. A second more fundamental challenge is that it is difficult to distinguish between a “weird” but good worker from an actually Byzantine attacker. In fact, we proved strong impossibility results in such a setting. For this we showed how overparameterization (which is prevalent in real world deep learning) provides a solution, ensuring convergence to the optimum even in the presence of attackers. Together, our results represent a major breakthrough and yield the first practical provably Byzantine robust algorithms for the non-iid setting.

8 Conclusion

In this thesis, we have presented solutions for learning with compressed communication, using local computation of the clients, and ensuring Byzantine robustness. However these problems are far from solved and will require substantially more work before collaborative learning catches up with its centralized counterpart. Below we explore three direct extensions of the work presented in the thesis.

Direction 1. *Model-agnostic communication protocols.* Current collaborative learning methods, each user trains a model on their local data and communicates the updated parameters, which are then simply *averaged* to aggregate the information (McMahan et al., 2017). While this may suffice in convex settings, due to the highly non-convex nature of deep learning this averaging step may result in a poorly performing parameters even if the individual user's parameters were accurate (Al-Shedivat et al., 2020). Further, perhaps even more fundamentally, averaging of the parameters only makes sense if every user has the exact same model architecture. The goal of this direction is to develop novel model-independent communication protocols enabling each user to use models best suited to their individual datasets and hardware resources. Our intermediate tasks would be:

- a. *Collaborative learning using mutual knowledge transfer:* Given access to excess unlabelled data, each user can synthesize a dataset using the trained models of their peers. Retraining on this larger dataset has been proven to successfully transfer knowledge between them (Zhang et al., 2015). This step can be incorporated into the traditional federated and decentralized learning setup to enable model-independent protocols. Additionally, recent theoretical progress in understanding distillation methods (Menon et al., 2020; Mobahi et al., 2020) can be extended to incorporate collaborative training, providing end-to-end learning guarantees.
- b. *Knowledge transfer using adversarial training:* Traditional knowledge transfer protocols crucially rely on access to unlabelled data. Instead, one could directly minimize the discrepancies in the predictions between the models on adversarially chosen inputs. This mimics a similar state-of-the-art strategy in adversarial robustness (Zhang et al., 2019a). Here too, providing formal guarantees would be a desirable additional goal.

Direction 2. *Personalized collaborative learning and transfer learning.* Insisting on using the same parameters for all users in collaborative learning results typically can lead to poor performance for users with heterogeneous datasets (Hsieh et al., 2019). Further, this often leads to the performance being unequally distributed, with low-resource users having significantly worse accuracy (Yu et al., 2020). The goal of this direction is to develop principled techniques for enabling users to personalize models to their particular datasets and use cases. The intermediate tasks would be

- a. *Notions of heterogeneity.* Notions such as label discrepancy (Mansour et al., 2009) and task-diversity (Tripuraneni et al., 2020) have been used to characterize performance in heterogeneous multi-task learning and domain adaptation. These notions could be generalized to develop a precise definition of heterogeneity which can tightly characterize

the performance of collaborative learning.

- b. *Minimizing heterogeneity via personalization.* Once a precise characterization of heterogeneity is available, it can be leveraged to analyze the benefits of currently proposed algorithms (Chen et al., 2018a; Jiang et al., 2019; Mansour et al., 2020). Further, it could lead to the development of novel methods which directly minimize heterogeneity adapting techniques from discrepancy minimization (Cortes et al., 2019).

Direction 3. *Robust and secure collaborative learning framework.* Malicious agents can attempt to derail the training process as we saw, or can additionally attempt to uncover sensitive information from other participants (Nasr et al., 2019). This direction aims to develop a unified framework which protects against both such attacks, enabling privacy-sensitive users (e.g. hospitals) to embrace collaborative learning. The intermediate tasks would be

- a. *Combine robustness, secure aggregation and differential privacy.* Defenses against Byzantine adversaries (Blanchard et al., 2017; Pillutla et al., 2019) aim to ensure convergence even in the face of malicious interference, and secure aggregation with differential privacy is used to protect against information leakage (Bonawitz et al., 2017). Building upon i) our initial attempt to combine general robust aggregators with secure aggregation (He et al., 2020), ii) our recent breakthrough in designing extremely simple robust aggregators (Karimireddy et al., 2021b), and iii) novel differential privacy schemes (Pichapati et al., 2019), a unified trusted framework with end-to-end learning guarantees may be developed.
- b. *Improve robustness and privacy using personalization.* Heterogeneity amongst the users leads to increased vulnerability to malicious agents since it becomes harder to distinguish an honest but atypical user from a malicious agent. Combining results from direction 2b with 3a could provide a potential solution.

The above only represent a smattering of direct extensions we are currently exploring. However, the topic of collaborative learning is vast and largely unexplored. As advanced machine learning algorithms become more tightly integrated into our daily lives, their interactions with society as well as among themselves is giving rise to rich ecosystem. To be able to understand as well as shape such ecosystems will require combining our understanding of AI algorithms with tools from game theory, economics, and social sciences—areas well versed with studying complex human interactions. This goes well beyond the current scope of machine learning research which focuses narrowly on developing better models on individual datasets, but rather requires taking a holistic view of the emergent machine behavior. Carefully designing such machine learning systems and studying their behavior is, I believe, one of the most pressing problems we currently face and one which is just beginning to be studied.

Appendices **Part**

9 Appendix for Error Feedback

9.1 Additional Experiments

In this section we give the full experimental details and results.

9.1.1 Convergence under sparse noise

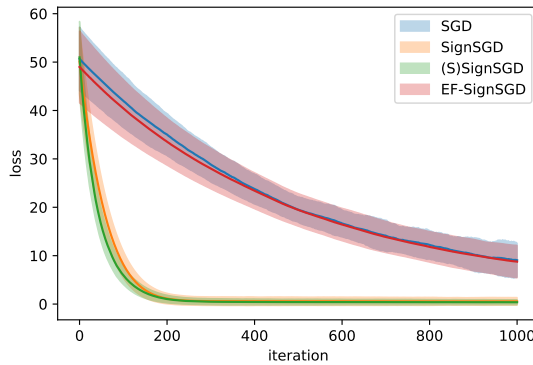


Figure 9.1 – A simple toy problem where SIGNSGD and (scaled)SIGNSGD are faster than both SGD and EF-SIGNSGD. The experiment is repeated 100 times with mean indicated by the solid line and the shaded region spans one standard deviation. As in (Bernstein et al., 2018), the loss is $f(\mathbf{x}) = \frac{1}{2}\|\mathbf{x}\|_2^2$ for $\mathbf{x} \in \mathbb{R}^{100}$, with gradient $\nabla f(\mathbf{x}) = \mathbf{x}$. The stochastic gradient is constructed by adding Gaussian noise $N(0, 100^2)$ to only the first coordinate of the gradient. The best learning-rate for SGD and EF-SIGNSGD was found to be 0.001, and for SIGNSGD and (scaled)SIGNSGD was 0.01. The conclusion of this toy experiment directly contradicts the results of our real-world experiments (Section 2.7) where EF-SIGNSGD is faster during training than both SGD and SIGNSGD. This shows that the sparse noisy coordinate explanation proposed by (Bernstein et al., 2018) is probably an incorrect explanation for the speed of sign based methods during training.

9.1.2 Description of models and datasets

The cifar dataset. The CIFAR 10 and 100 training and testing datasets was loaded using the default Pytorch torchvision api¹. Data augmentation consisting of random 32×32 crops (padding 4) and horizontal flips was performed. Both sets were normalized over each separate channel.

VGG (on CIFAR 10). We used VGG19 architecture consisting in the following layers:

64 -> 64 -> M -> 128 -> 128 -> M -> 256 -> 256 -> 256 -> 256 -> M -> 512 -> 512 -> 512 -> 512 -> M -> 512 -> 512 -> 512 -> 512 -> M

where M denotes max pool layers (kernel 2 and stride 2), and each of the number n (either of 64/128/256/512) represents a two dimensional convolution layer with n channels a kernel of 3 and a padding of 1. All of them are followed by a batch normalization layer. Everywhere, ReLU activation is used.

Resnet (on CIFAR 100). We used a standard Resnet18 architecture with one convolution followed by four blocks and one dense layer².

9.1.3 Learning rate tuning

For all the experiments, the learning rate was divided by 10 at epochs 100 and again at 150. We tuned the initial learning rate on batchsize 128. The learning rates for batchsize 32 and 8 were scaled down by 4 and 16 respectively. To tune the initial learning rate, the algorithm was run with the same constant learningrate for 100 epochs. Then the learning rate which resulted in the best (i.e. smallest) test loss is chosen. The search space of possible learning rates was taken to be 9 values equally spaced in logarithmic scale over 10^{-5} to 10^1 (inclusive).

The numbers below are rounded values (2 significant digits) of the actual learning rates:

$1.0 \times 10^{-5}, 5.6 \times 10^{-5}, 3.2 \times 10^{-4}, 1.8 \times 10^{-3}, 1.0 \times 10^{-2}, 5.6 \times 10^{-2}, 3.2 \times 10^{-1}, 1.8 \times 10^0, 1.0 \times 10^1$.

The best learning rate for each of the method is shown in table 9.1.

Algorithm	Resnet18	VGG19
SGDM	1.0×10^{-2}	1.0×10^{-2}
SIGNSGD	5.6×10^{-2}	5.6×10^{-2}
Signum	3.2×10^{-4}	5.6×10^{-5}
EF-SIGNSGD	5.6×10^{-2}	5.6×10^{-2}

Table 9.1 – The best initial learning rates for the four algorithms for batch size 128 on VGG19 (CIFAR 10 data) and Resnet18 (CIFAR 100 data).

¹<https://pytorch.org/docs/stable/torchvision/index.html>

²<https://github.com/kuangliu/pytorch-cifar/blob/master/models/resnet.py>

9.1.4 Experiments with Resnet

We report the complete results (including the losses) for Resnet in Fig. 9.2.

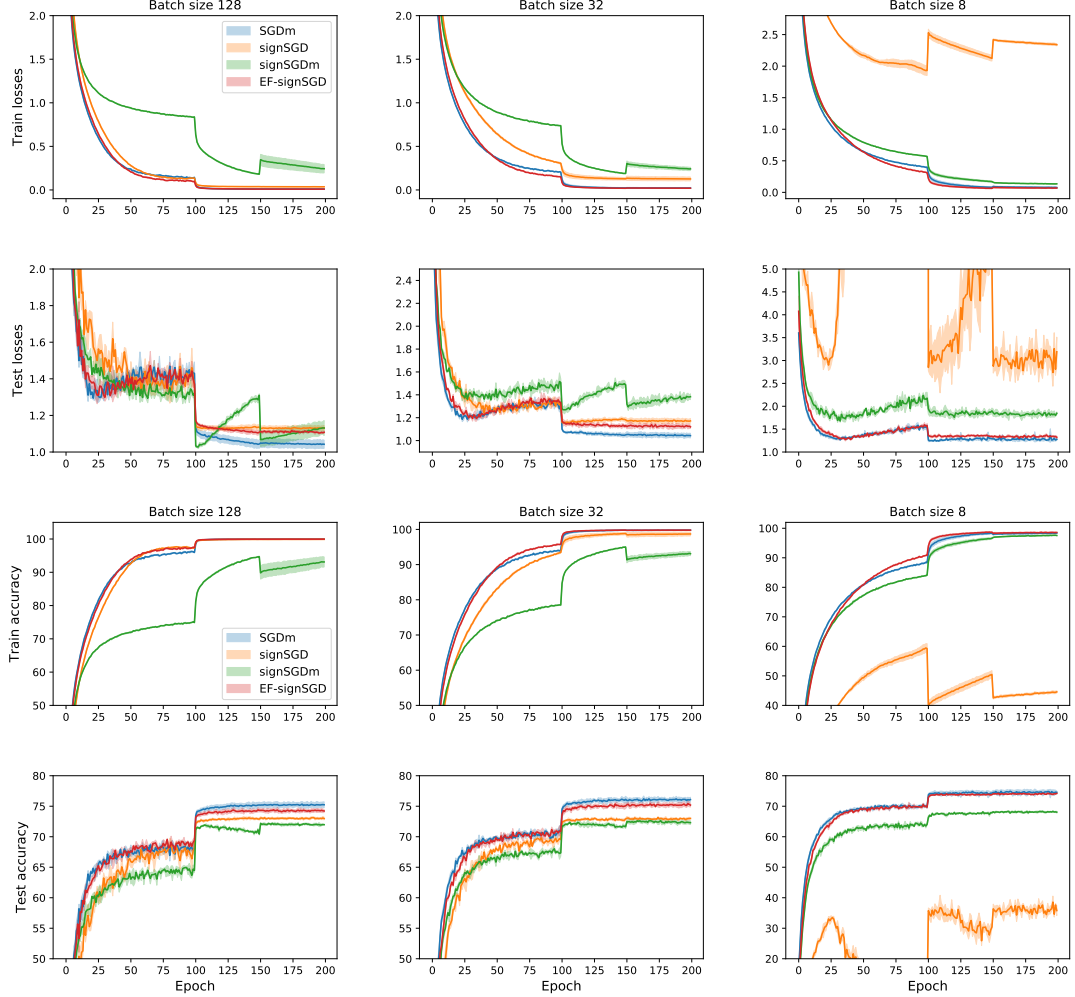


Figure 9.2 – Experimental results showing the loss values and accuracy percentages on the train and test datasets, on CIFAR-100 using Resnet18 for different batch-sizes. The solid curves represent the mean value and shaded region spans one standard deviation obtained over three repetitions. Note that the scale of the y-axis varies across the plots. The losses behave very similar to the accuracies—EF-SIGNSGD consistently and significantly outperforms the other sign-based methods, is faster than SGDM on train, and closely matches SGDM on test.

9.1.5 Experiments with VGG

We report the complete results (including the losses) for VGG in Fig. 9.3.

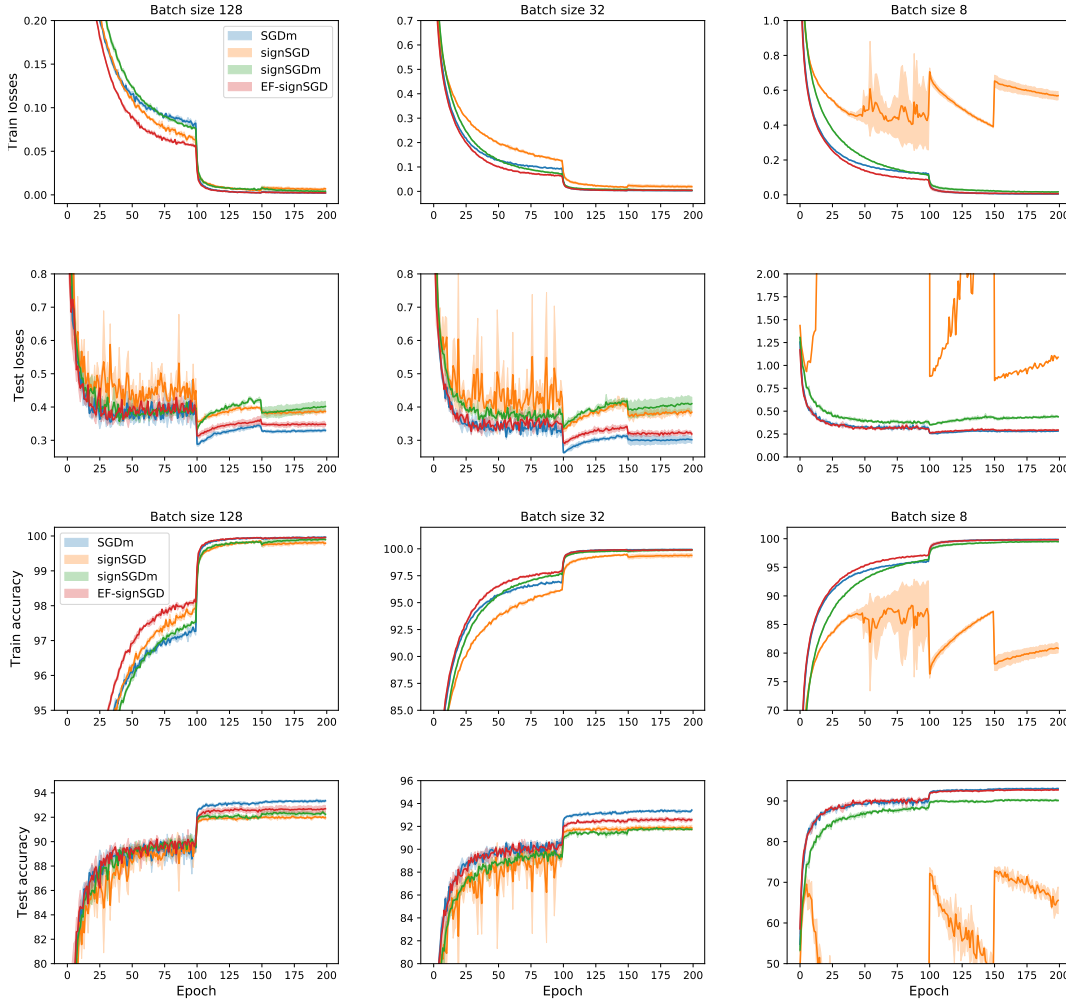


Figure 9.3 – Experimental results showing the loss values and accuracy percentages on the train and test datasets, on CIFAR-10 using VGG19 for different batch-sizes. The solid curves represent the mean value and shaded region spans one standard deviation obtained over three repetitions. Note that the scale of the y-axis varies across the plots. The plots for VGG19 behave very similarly to that of Resnet18, except that Signum performs better on the train dataset. On the test dataset, Signum and the other algorithms behave exactly as in Resnet. Here too, EF-SIGNSGD consistently and significantly outperforms the other sign-based methods, is faster than SGDM on train, and also closely matches the test performance of SGDM.

		Algorithm			
		SGDM	scaled SIGNSGD	Signum	EF-SIGNSGD
Batch size	128	75.35	-2.21	-3.15	-0.92
	32	76.22	-3.04	-3.57	-0.79
	8	74.91	-36.35	-6.6	-0.64

Table 9.2 – Generalization gap on CIFAR-100 using Resnet18 for different batch-sizes. For SGDM we report the best mean test accuracy percentage, and for the other algorithms we report their difference to the SGDM accuracy (i.e. the generalization gap). EF-SIGNSGD has a much smaller gap which decreases with decreasing batchsize. The generalization gap of Signum and SIGNSGD increases as the batchsize decreases.

		Algorithm			
		SGDM	scaled SIGNSGD	Signum	EF-SIGNSGD
Batch size	128	93.38	-1.31	-0.94	-0.68
	32	93.42	-1.49	-1.54	-0.71
	8	93.09	-20.22	-2.75	-0.27

Table 9.3 – Generalization gap on CIFAR-10 using VGG19 for different batch-sizes. For SGDM we report the best mean test accuracy percentage, and for the other algorithms we report their difference to the SGDM accuracy (i.e. the generalization gap). EF-SIGNSGD has a much smaller gap which decreases with decreasing batchsize. The generalization gap of Signum and SIGNSGD increases as the batchsize decreases.

9.1.6 Data generation process (Section 2.6.2)

The data is generated as in Section 3.3 of (Wilson et al., 2017). We fix $n = 200$ (the number of data points) and $d = 6n$ (dimension) in the below process. Each entry of the target label vector $\mathbf{y} \in \{-1, 1\}^n$ is uniformly set as -1 or 1 . Then the j th coordinate (column) of the i th data point (row) in the data matrix $\mathbf{A} \in \mathbb{R}^{n \times d}$ is filled as follows:

$$A_{i,j} = \begin{cases} y_i & j = 1, \\ 1 & j = 2, 3, \\ 1 & j = 4 + 5(i - 1), \dots, 4 + 5(i - 1) + 2(1 - y_i), \\ 0 & \text{otherwise.} \end{cases}$$

Then the data matrix \mathbf{A} and labels \mathbf{y} are randomly (and equally) split between the train and the test dataset. Hence there are 100 data points each of dimension 1200 in the test and train.

9.2 Missing Proofs

In this section we fill out the proofs of claims, lemmas, and theorems made in the main paper.

9.2.1 Proof of counter-example (Theorem 1)

The stochastic gradient at iteration t is of the form

$$\mathbf{g}_t = \mathbf{a}_{i_t} l'_{i_t}(\langle \mathbf{a}_{i_t}, \mathbf{x} \rangle).$$

This means that

$$\text{sign}(\mathbf{g}_t) = \text{sign}(\mathbf{a}_{i_t}) \cdot \text{sign}(l'_{i_t}(\langle \mathbf{a}_{i_t}, \mathbf{x} \rangle)) = \pm \mathbf{s}.$$

Thus $\mathbf{x}_{t+1} = \mathbf{x}_t \pm \gamma \mathbf{s}$ and the iterates of SIGNSGD can only move along the direction \mathbf{s} . Then, SIGNSGD can converge only if there exists $\gamma^* \in \mathbb{R}$ such that

$$\mathbf{x}_0 = \mathbf{x}^* + \gamma^* \mathbf{s}.$$

Since the measure of this set in \mathbb{R}^d for $d \geq 2$ is 0, we can conclude that SIGNSGD will not converge to \mathbf{x}^* almost surely.

9.2.2 Proof of bounded error (Lemma 3)

By definition of the error sequence,

$$\|\mathbf{e}_{t+1}\|^2 = \|\mathcal{C}(\mathbf{p}_t) - \mathbf{p}_t\|_2^2 \leq (1 - \delta) \|\mathbf{p}_t\|_2^2 = (1 - \delta) \|\mathbf{e}_t + \gamma \mathbf{g}_t\|_2^2.$$

In the inequality above we used that $\mathcal{C}(\cdot)$ is a δ -approximate compressor. Let us separate the independent stochastic noise in the above equation using

$$\mathbb{E} \|\mathbf{e}_t + \gamma \mathbf{g}_t\|_2^2 = \mathbb{E} \|\mathbf{e}_t + \gamma \nabla f(\mathbf{x}_t)\|_2^2 + \gamma^2 \mathbb{E} \|\mathbf{g}_t - \nabla f(\mathbf{x}_t)\|_2^2 \leq \mathbb{E} \|\mathbf{e}_t + \gamma \nabla f(\mathbf{x}_t)\|_2^2 + \gamma^2 \sigma^2.$$

We thus have a recurrence relation on the bound of \mathbf{e}_t . Using Young's inequality, we have that for any $\eta > 0$:

$$(1 - \delta) \mathbb{E} \|\mathbf{e}_t + \gamma \nabla f(\mathbf{x}_t)\|_2^2 + (1 - \delta) \sigma^2 \leq (1 - \delta)(1 + \eta) \|\mathbf{e}_t\|_2^2 + \gamma^2 (1 - \delta)(1 + 1/\eta) \|\nabla f(\mathbf{x}_t)\|_2^2 + (1 - \delta) \gamma^2 \sigma^2.$$

Let us pick $\eta = \frac{\delta}{2(1-\delta)}$ such that $1 + 1/\eta = (2 - \delta)/\delta \leq 2/\delta$, and $(1 - \delta)(1 + \eta) = (1 - \frac{\delta}{2})$. Plugging this in the above gives

$$\begin{aligned} \mathbb{E} \|\mathbf{e}_{t+1}\|^2 &\leq (1 - \frac{\delta}{2}) \|\mathbf{e}_t\|_2^2 + \frac{2\gamma^2(1-\delta)}{\delta} \|\nabla f(\mathbf{x}_t)\|_2^2 + (1 - \delta) \gamma^2 \sigma^2 \\ &\leq (1 - \frac{\delta}{2}) \|\mathbf{e}_t\|_2^2 + (1 - \delta) \gamma^2 (\frac{2}{\delta} M^2 + \sigma^2). \end{aligned}$$

Here on is simple algebraic computations to solve the recurrence relation above.

$$\mathbb{E} \|\mathbf{e}_{t+1}\|^2 \leq \sum_{i=0}^{\infty} [1 - \frac{\delta}{2}]^i \gamma^2 (1 - \delta) (\frac{2}{\delta} M^2 + \sigma^2) = \frac{2(1-\delta)}{\delta^2} \gamma^2 (2M^2 + \delta \sigma^2). \quad \square$$

9.2.3 Proof of non-convex convergence of EF-SIGNSGD (Theorem II)

As outlined in the proof sketch, the analysis considers the actual sequence $\{\mathbf{x}_t\}$ as an approximation to the sequence $\{\tilde{\mathbf{x}}_t\}$, where $\tilde{\mathbf{x}}_t = \mathbf{x}_t - \mathbf{e}_t$. It satisfies the recurrence

$$\tilde{\mathbf{x}}_{t+1} = \mathbf{x}_t - \mathbf{e}_{t+1} - \mathcal{C}(\mathbf{p}_t) = \mathbf{x}_t - \mathbf{p}_t = \tilde{\mathbf{x}}_t - \gamma \mathbf{g}_t.$$

Since the function f is L -smooth,

$$\begin{aligned} \mathbb{E}_t[f(\tilde{\mathbf{x}}_{t+1})] &\leq f(\tilde{\mathbf{x}}_t) + \langle \nabla f(\tilde{\mathbf{x}}_t), \mathbb{E}_t[\tilde{\mathbf{x}}_{t+1} - \tilde{\mathbf{x}}_t] \rangle + \frac{L}{2} \mathbb{E}_t[\|\tilde{\mathbf{x}}_{t+1} - \tilde{\mathbf{x}}_t\|_2^2] \\ &= f(\tilde{\mathbf{x}}_t) - \gamma \langle \nabla f(\tilde{\mathbf{x}}_t), \mathbb{E}_t[\mathbf{g}_t] \rangle + \frac{L\gamma^2}{2} \mathbb{E}_t[\|\mathbf{g}_t\|_2^2] \\ &\leq f(\tilde{\mathbf{x}}_t) - \gamma \langle \nabla f(\tilde{\mathbf{x}}_t), \nabla f(\mathbf{x}_t) \rangle + \frac{L\gamma^2\sigma^2}{2} + \frac{L\gamma^2}{2} \|\nabla f(\mathbf{x}_t)\|^2. \end{aligned}$$

In the above we need to get rid of $\nabla f(\tilde{\mathbf{x}}_t)$ since we never encounter it in the algorithm. We can do so using an alternate definition of smoothness of f :

$$\|\nabla f(\mathbf{x}) - \nabla f(\mathbf{y})\|_2 \leq L\|\mathbf{x} - \mathbf{y}\|_2.$$

Using the above with $\mathbf{x} = \mathbf{x}_t$ and $\mathbf{y} = \tilde{\mathbf{x}}_t$ we continue as

$$\begin{aligned} \mathbb{E}_t[f(\tilde{\mathbf{x}}_{t+1})] &\leq f(\tilde{\mathbf{x}}_t) - \gamma \langle \nabla f(\mathbf{x}_t), \nabla f(\mathbf{x}_t) \rangle + \frac{L\gamma^2\sigma^2}{2} + \gamma \langle \nabla f(\mathbf{x}_t) - \nabla f(\tilde{\mathbf{x}}_t), \nabla f(\mathbf{x}_t) \rangle + \frac{L\gamma^2}{2} \|\nabla f(\mathbf{x}_t)\|^2 \\ &\leq f(\tilde{\mathbf{x}}_t) - \gamma \|\nabla f(\mathbf{x}_t)\|_2^2 + \frac{L\gamma^2\sigma^2}{2} + \left(\frac{\gamma\rho}{2} + \frac{L\gamma^2}{2}\right) \|\nabla f(\mathbf{x}_t)\|_2^2 + \frac{\gamma}{2\rho} \|\nabla f(\mathbf{x}_t) - \nabla f(\tilde{\mathbf{x}}_t)\|_2^2 \\ &\leq f(\tilde{\mathbf{x}}_t) - \gamma \|\nabla f(\mathbf{x}_t)\|_2^2 + \frac{L\gamma^2\sigma^2}{2} + \left(\frac{\gamma\rho}{2} + \frac{L\gamma^2}{2}\right) \|\nabla f(\mathbf{x}_t)\|_2^2 + \frac{\gamma L^2}{2\rho} \|\mathbf{x}_t - \tilde{\mathbf{x}}_t\|_2^2 \\ &\leq f(\tilde{\mathbf{x}}_t) - \gamma \left(1 - \frac{\rho}{2} - \frac{L\gamma}{2}\right) \|\nabla f(\mathbf{x}_t)\|_2^2 + \frac{L\gamma^2\sigma^2}{2} + \frac{\gamma L^2}{2\rho} \|\mathbf{e}_t\|_2^2. \end{aligned}$$

In the second inequality follows from the mean-value inequality and holds for any $\rho > 0$. The proof of Lemma 3 gives us a recursive bound on the norm of \mathbf{e}_t as

$$\mathbb{E}\|\mathbf{e}_{t+1}\|^2 \leq (1 - \frac{\delta}{2})\|\mathbf{e}_t\|_2^2 + \frac{2\gamma^2(1-\delta)}{\delta} \|\nabla f(\mathbf{x}_t)\|_2^2 + (1 - \delta)\gamma^2\sigma^2$$

Scale this by $\frac{\gamma L^2}{\rho\delta}$ to get

$$\frac{2\gamma L^2}{2\rho\delta} \mathbb{E}\|\mathbf{e}_{t+1}\|^2 \leq \frac{2\gamma L^2}{2\rho\delta} \|\mathbf{e}_t\|_2^2 + \frac{2\gamma^3 L^2(1-\delta)}{\rho\delta^2} \|\nabla f(\mathbf{x}_t)\|_2^2 + \frac{\gamma^3 L^2(1-\delta)}{\rho\delta} \sigma^2 - \frac{\gamma L^2}{2\rho} \|\mathbf{e}_t\|_2^2.$$

Adding this to the previously derived bound on $f(\mathbf{x}_{t+1})$ gives

$$\begin{aligned} & \left(\mathbb{E}_t[f(\tilde{\mathbf{x}}_{t+1})] + \frac{2\gamma L^2}{2\rho\delta} \mathbb{E}\|\mathbf{e}_{t+1}\|^2 \right) - \left(f(\tilde{\mathbf{x}}_t) + \frac{2\gamma L^2}{2\rho\delta} \mathbb{E}\|\mathbf{e}_t\|^2 \right) \\ & \leq -\gamma \left(1 - \frac{\rho}{2} - \frac{L\gamma}{2} - \frac{2\gamma^2 L^2 (1-\delta)}{\rho\delta^2} \right) \|\nabla f(\mathbf{x}_t)\|_2^2 + \frac{L\gamma^2 \sigma^2}{2} + \frac{\gamma^3 L^2 \sigma^2}{2\rho} \frac{4(1-\delta)}{\delta} \\ & = -\gamma \left(1 - L\gamma \left(\frac{1}{2} + \frac{2(\sqrt{1-\delta})}{\delta} \right) \right) \|\nabla f(\mathbf{x}_t)\|_2^2 + L\gamma^2 \sigma^2 \left(\frac{1}{2} + 2\sqrt{1-\delta} \right). \end{aligned}$$

The last step used $\rho = \frac{2\gamma L\sqrt{1-\delta}}{\delta}$. Now, suppose that we use a step-size $\gamma \leq 4\delta/L(\delta + 4(\sqrt{1-\delta}))$. This gives $L\gamma(\frac{1}{2} + \frac{2(\sqrt{1-\delta})}{\delta}) \leq \frac{1}{4}$. Further, recall that $\mathbf{e}_0 = 0$. Thus, rearranging the terms and averaging over t gives

$$\begin{aligned} \frac{1}{T+1} \sum_{t=0}^T \|\nabla f(\mathbf{x}_t)\|_2^2 & \leq \frac{2}{\gamma} \cdot \left(\frac{1}{T+1} \sum_{t=0}^T \left(\mathbb{E} \left[f(\tilde{\mathbf{x}}_t) + \frac{2\gamma L^2}{2\rho\delta} \|\mathbf{e}_t\|^2 \right] - \mathbb{E} \left[f(\tilde{\mathbf{x}}_{t+1}) + \frac{2\gamma L^2}{2\rho\delta} \|\mathbf{e}_{t+1}\|^2 \right] \right) \right. \\ & \quad \left. + 3L\gamma^2 \sigma^2 \right) \\ & \leq \frac{2(f(\mathbf{x}_0) - f^*)}{\gamma(T+1)} + 6\gamma L\sigma^2. \end{aligned}$$

This gives the result in Theorem II.

9.2.4 Proof of convex convergence of EF-SIGNSGD (Theorem III)

As in the proof of Theorem II, we start by considering the sequence $\{\tilde{\mathbf{x}}_t\}$ where $\tilde{\mathbf{x}}_t = \mathbf{x}_t - \mathbf{e}_t$. As we saw, $\tilde{\mathbf{x}}_{t+1} = \tilde{\mathbf{x}}_t - \gamma \mathbf{g}_t$. Suppose that \mathbf{x}_t^* is an optimum solution. We will abuse notation here and use $\partial f(\mathbf{x})$ to mean any subgradient of f at \mathbf{x} .

$$\begin{aligned} \mathbb{E}_t[\|\tilde{\mathbf{x}}_{t+1} - \mathbf{x}^*\|^2] & = \mathbb{E}_t[\|\tilde{\mathbf{x}}_t - \gamma \mathbf{g}_t - \mathbf{x}^*\|^2] \\ & = \|\tilde{\mathbf{x}}_t - \mathbf{x}^*\|^2 + \gamma^2 \mathbb{E}_t[\|\mathbf{g}_t\|^2] - 2\gamma \langle \mathbb{E}_t[\mathbf{g}_t], \tilde{\mathbf{x}}_t - \mathbf{x}^* \rangle \\ & \leq \|\tilde{\mathbf{x}}_t - \mathbf{x}^*\|^2 + \gamma^2 (M^2 + \sigma^2) - 2\gamma \langle \partial f(\mathbf{x}_t), \tilde{\mathbf{x}}_t - \mathbf{x}^* \rangle. \end{aligned}$$

We do not want $\tilde{\mathbf{x}}_t$ appearing in the right side of the equation and so we will replace it with \mathbf{x}_t and use Lemma 3 to bound the error:

$$\begin{aligned} \mathbb{E}_t[\|\tilde{\mathbf{x}}_{t+1} - \mathbf{x}^*\|^2] & \leq \|\tilde{\mathbf{x}}_t - \mathbf{x}^*\|^2 + \gamma^2 (M^2 + \sigma^2) - 2\gamma \langle \partial f(\mathbf{x}_t), \mathbf{x}_t - \mathbf{x}^* \rangle + 2\gamma \langle \partial f(\mathbf{x}_t), \mathbf{x}_t - \tilde{\mathbf{x}}_t \rangle \\ & = \|\tilde{\mathbf{x}}_t - \mathbf{x}^*\|^2 + \gamma^2 (M^2 + \sigma^2) - 2\gamma \langle \partial f(\mathbf{x}_t), \mathbf{x}_t - \mathbf{x}^* \rangle - 2\gamma \langle \partial f(\mathbf{x}_t), \mathbf{e}_t \rangle \\ & \leq \|\tilde{\mathbf{x}}_t - \mathbf{x}^*\|^2 + \gamma^2 (M^2 + \sigma^2) - 2\gamma \langle \partial f(\mathbf{x}_t), \mathbf{x}_t - \mathbf{x}^* \rangle + 2\gamma \|\partial f(\mathbf{x}_t)\| \|\mathbf{e}_t\| \\ & \leq \|\tilde{\mathbf{x}}_t - \mathbf{x}^*\|^2 + \gamma^2 (M^2 + \sigma^2) - 2\gamma \langle \partial f(\mathbf{x}_t), \mathbf{x}_t - \mathbf{x}^* \rangle + \frac{2\gamma^2}{\delta} \sqrt{2(1-\delta)(2M^2 + \delta\sigma^2)} \|\partial f(\mathbf{x}_t)\|. \end{aligned}$$

We use Cauchy-Schwarz in the third step, and Lemma 3 in the last step. We will use the loose bound $\|\partial f(\mathbf{x}_t)\| \leq \sigma$. This is the key difference between the non-smooth case and the smooth (and strongly-convex) case considered in (Stich et al., 2018). In the smooth case,

the term $\|\nabla f(\mathbf{x}_t)\|^2$ can be bounded by the error $f(\mathbf{x}_t) - f^\star$. This implies that the last term in the equation above goes to 0 faster than γ^2 , allowing the asymptotic rate to not depend on the compression quality δ . However, this is not true in the non-smooth case making the dependence of the rate on δ unavoidable. We now have

$$\mathbb{E}_t[\|\tilde{\mathbf{x}}_{t+1} - \mathbf{x}^\star\|^2] \leq \|\tilde{\mathbf{x}}_t - \mathbf{x}^\star\|^2 + \gamma^2(M^2 + \sigma^2) - 2\gamma\langle \partial f(\mathbf{x}_t), \mathbf{x}_t - \mathbf{x}^\star \rangle + \frac{2\gamma^2}{\delta} M \sqrt{2(1-\delta)(2M^2 + \delta\sigma^2)}. \quad (9.1)$$

Recall that $\mathbf{e}_0 = 0$ and so $\tilde{\mathbf{x}}_0 = \mathbf{x}_0$. Rearranging the terms and averaging, we get

$$\begin{aligned} \frac{1}{T+1} \sum_{t=0}^T \mathbb{E}[\langle \partial f(\mathbf{x}_t), \mathbf{x}_t - \mathbf{x}^\star \rangle] &\leq \frac{1}{2\gamma(T+1)} \sum_{t=0}^T (\mathbb{E}[\|\tilde{\mathbf{x}}_t - \mathbf{x}^\star\|^2] - \mathbb{E}[\|\tilde{\mathbf{x}}_{t+1} - \mathbf{x}^\star\|^2]) \\ &\quad + \frac{\gamma}{2}(\sigma^2 + M^2) + \frac{2\gamma M \sqrt{(1-\delta)(M^2 + \sigma^2)}}{\delta} \\ &\leq \frac{\|\mathbf{x}_0 - \mathbf{x}^\star\|^2}{2\gamma(T+1)} + \gamma(\sigma^2 + M^2) \left(\frac{1}{2} + \frac{2M}{\delta} \sqrt{\frac{1-\delta}{M^2 + \sigma^2}} \right) \\ &\leq \frac{\|\mathbf{x}_0 - \mathbf{x}^\star\|^2}{2\gamma(T+1)} + \gamma(\sigma^2 + M^2) \left(\frac{1}{2} + \frac{2\sqrt{1-\delta}}{\delta} \right). \end{aligned}$$

To finish the proof, we have to simply use the convexity of f twice on the left hand side of the above inequality:

$$\frac{1}{T+1} \sum_{t=0}^T \mathbb{E}[\langle \partial f(\mathbf{x}_t), \mathbf{x}_t - \mathbf{x}^\star \rangle] \geq \frac{1}{T+1} \sum_{t=0}^T f(\mathbf{x}_t) - f(\mathbf{x}^\star) \geq f\left(\frac{1}{T+1} \sum_{t=0}^T \mathbf{x}_t\right) - f(\mathbf{x}^\star) = f(\tilde{\mathbf{x}}_T) - f(\mathbf{x}^\star).$$

For the standard rate of SGD in remark 6, we just set $\delta = 1$. □

9.2.5 Proof relating linear span of gradients to pseudo-inverse (Lemma 9)

Recall that $\mathbf{A} \in \mathbb{R}^{n \times d}$ for $n < d$. Assume without loss of generality that the rows of \mathbf{A} are linearly independent and hence \mathbf{A} is of rank n . The stochastic gradient for $f(\mathbf{x}) = \|\mathbf{A}\mathbf{x} - \mathbf{b}\|^2$ is of the form $\sum \alpha_i \mathbf{A}_{i,:}$ where $\mathbf{A}_{i,:}$ indicates the i th column of \mathbf{A} . If \mathbf{x}_t is in the linear span of the stochastic gradients, then there exists a vector $\boldsymbol{\alpha}_t \in \mathbb{R}^n$ such that

$$\mathbf{x}_t = \mathbf{A}^\top \boldsymbol{\alpha}_t.$$

Suppose \mathbf{x}^\star is the solution reached. Then $\mathbf{A}\mathbf{x}^\star = \mathbf{b}$ and also $\mathbf{x}^\star = \mathbf{A}^\top \boldsymbol{\alpha}^\star$ for some $\boldsymbol{\alpha}^\star$. Hence $\boldsymbol{\alpha}^\star$ must satisfy

$$\mathbf{A}\mathbf{A}^\top \boldsymbol{\alpha}^\star = \mathbf{b}.$$

Chapter 9. Appendix for Error Feedback

Since the rank of A is n , the matrix $AA^\top \in \mathbb{R}^{n \times n}$ is full-rank and invertible. This means that there exists a unique solution to α^\star and x^\star :

$$\alpha^\star = (AA^\top)^{-1}b \quad \text{and} \quad x^\star = A^\top \alpha^\star = A^\top (AA^\top)^{-1}b. \quad \square$$

10 Appendix for PowerSGD

10.1 Discussions of convergence

10.1.1 Eigen compression

Assumption H (Eigen compression). *Consider any matrix $M = g_t + e_t$ encountered during the run of Algorithm 4 such that M is of rank R . Further, suppose that $\mathcal{C}_r(M)$ is the best rank- r approximation of M i.e.*

$$\mathcal{C}_r(M) = \underset{C}{\operatorname{argmin}} \|M - C\|^2.$$

Then we assume that there exists a $\delta_{e,r} > 0$ such that

$$\|M - \mathcal{C}_r(M)\|^2 \leq (1 - \delta_{e,r}) \|M\|^2 \text{ a.s.}$$

We state the below standard fact from linear algebra.

Remark 26 (Best rank- r approximation). *Suppose we are given a matrix M of rank n whose singular value decomposition is*

$$M = \sum_{i=1}^n \sigma_i \mathbf{u}_i \mathbf{v}_i^\top,$$

where the singular-values (σ_i) are sorted in descending order. Then the best rank- r approximation of M for $r \leq n$ is

$$\mathcal{C}_r(M) = \left(\sum_{i=1}^r \sigma_i \mathbf{u}_i \mathbf{v}_i^\top \right) Q,$$

where $Q \in \mathbb{R}^{r \times r}$ is an orthogonal matrix, and further the quality of its approximation is bounded by

$$\|M - \mathcal{C}_r(M)\|^2 = \left(1 - \frac{\sum_{i=1}^r \sigma_i^2}{\sum_{i=1}^n \sigma_i^2} \right) \|M\|^2.$$

Thus if we used Algorithm 4 with exact rank- r approximation of the gradients, we would converge at rate dictated by the eigen-spectrum of the gradients. If the singular values are

‘top-heavy’ i.e. the largest r values are significantly larger than the rest, then a rank- r approximation is quite accurate. As demonstrated in (Wang et al., 2018), the eigen-spectrum of stochastic gradients in common deep learning tasks is indeed ‘top-heavy’. Thus we can expect $\delta_{e,r}$ to be bounded away from 0 even for very small r (e.g. 1 or 2). Of course computing the actual top eigenvectors of the stochastic gradients is very computationally expensive, and more-over is not linear (and hence does not support *reduce*).

10.1.2 Subspace iteration

The key innovation in POWERSGD is to use only a *single* step of subspace (or power) iteration to give a fast low rank approximation (Stewart and Miller, 1975) to the given matrix, which in our case is a stochastic gradient. However, a single step of subspace iteration in general does not result in an adequate low-rank approximation of the input matrix. To combat this, and to at the same time reduce the variance of the stochastic gradient approximation compared to the full (deterministic) gradient, we propose the *reuse* of the low-rank approximation from the previous iteration as the starting point for the current iteration. This is in spite of the target matrices which are trying to approximate are *changing*, as the parameters evolve. Nevertheless, reuse here is justified because the full gradient does not change very fast (the gradient is Lipschitz by assumption) and we only perform a tiny update at each step, so can be assumed to be stationary within a small number of steps. Intuitively, by linearity of the subspace operation, the sequence of subspace steps with the reuse then is converging to the eigenvector of the averaged stochastic gradients over these steps, thus having a lower variance than the analogue without re-use, which has no such averaging effect.

For simplicity, we assume all matrices to be square and symmetric in this sub-section. These insights can be generalized to arbitrary matrices but with a substantial increase in complexity of exposition. Here, we simply note that for any non-square matrix A , we can instead consider

$$\tilde{A} = \begin{bmatrix} 0 & A \\ A^\top & 0 \end{bmatrix}$$

which is symmetric and has the same eigenvectors and eigenvalues as the original matrix A —see (Stewart, 1976) for more details on handling such cases.

We can now state an informal theorem about the convergence of subspace iteration.

Theorem XXV. *Suppose that we run subspace iteration as in (10.1) on a fixed matrix $A_t = M$. Also let $M = \sum_{i=1}^n \sigma_i \mathbf{u}_i \mathbf{u}_i^\top$ be the eigen decomposition of M with $\sigma_1 \geq \dots \sigma_r > \sigma_{r+1} \geq \dots \geq \sigma_n$. Then there exists an orthogonal matrix $Q \in \mathbb{R}^{r \times r}$ such that*

$$\lim_{t \rightarrow \infty} X_t = [\mathbf{u}_1, \dots, \mathbf{u}_r] Q.$$

In other words, (10.1) recovers the best rank- r approximation of M as long as there is a gap between the σ_r and σ_{r+1} eigenvalues.

Suppose that at each iteration we receive a matrix $A_t \in \mathbb{R}^{n \times n}$ whose expectation is the same fixed matrix $M \in \mathbb{R}^{n \times n}$. Starting from an orthonormalized $X_0 \in \mathbb{R}^{n \times r}$ (i.e. $X_0^\top X_0 = I_r$), the rank- r subspace iteration algorithm performs the following update:

$$X_{t+1} = \text{ORTHOGONALIZE}(A_t X_t). \quad (10.1)$$

The final output of the algorithm (i.e.) the matrix approximation is $(A_{T+1} X_T) X_T^\top$. This closely resembles the method of POWERSGD as outlines in Algorithm 3. We recommend (Arbenz, 2016) for an in-depth analysis of the (non-varying) subspace iteration algorithm.

Remark 27 (Orthogonalization is a linear operation). *We recall some more facts from linear algebra. For any square matrix B , there exists an orthogonal matrix Q and a triangular matrix R such that $QQ^\top = I$ and $B = QR$. This is true e.g. if we use Gram–Schmidt procedure to orthonormalize B : Suppose $\text{ORTHOGONALIZE}(B)$ uses the Gram–Schmidt procedure to orthogonalize B . Then there exists a triangular matrix R such that*

$$\text{ORTHOGONALIZE}(B) = BR^{-1}.$$

10.1.3 Proof that Orthogonalization is a linear operation (Remark 27)

It is easy to see that for any orthogonal matrix Q , the matrix $[\mathbf{u}_1, \dots, \mathbf{u}_r]Q$ is also orthogonal, and further is the fixed point of (10.1). In fact all rank- r matrices which are fixed points of (10.1) are of this form.

We will use the observation in Remark 27 to rewrite the update (10.1) in a more convenient fashion. There exist triangular matrices R_0, \dots, R_t such that

$$X_{t+1} = \text{ORTHOGONALIZE}(A_t X_t) = A_t X_t R_t^{-1} = (A_t A_{t-1} \cdots A_0) X_0 (R_0^{-1} R_1^{-1} \cdots R_t^{-1}).$$

Thus X_{t+1} can alternatively be written as

$$X_{t+1} = \text{ORTHOGONALIZE}((A_t A_{t-1} \cdots A_0) X_0) = \text{ORTHOGONALIZE}(M^{t+1} X_0).$$

Here we assumed that the matrix was fixed i.e. $A_t = M$. Let us further assume that X_0 has a non-zero support on the first r eigenvectors of M . Then, a gap in the eigenvalues $\sigma_r > \sigma_{r+1}$ implies that $\text{ORTHOGONALIZE}(M^{t+1} X_0)$ converges to $[\mathbf{u}_1, \dots, \mathbf{u}_r]Q$. We refer to Chapter 7.2 of (Arbenz, 2016) for the actual proof of this fact. \square

10.1.4 Proof of Single/multi worker equivalence (Lemma 3.5.1)

Consider the update performed by POWERSGD for arbitrary vectors $\{\mathbf{v}_w\}$. Let $\mathcal{C}(\mathbf{v}_w)$ be the compressed version of \mathbf{v}_w for $w \in \{1, \dots, W\}$. Then by design of POWERSGD, the following

holds:

$$\text{DECOMPRESS}(\text{AGGREGATE}(\mathcal{C}(\mathbf{v}_1), \dots, \mathcal{C}(\mathbf{v}_W))) = \text{DECOMPRESS}(\mathcal{C}(\frac{1}{W} \sum_w \mathbf{v}_w)).$$

This implies that running the algorithm on multiple workers, or running it on a single worker with a larger batch-size is identical. In particular,

$$\begin{aligned} & \text{DECOMPRESS}(\text{AGGREGATE}(\mathcal{C}(\mathbf{g}_{t,1} + \mathbf{e}_{t,1}), \dots, \mathcal{C}(\mathbf{g}_{t,W} + \mathbf{e}_{t,W}))) \\ &= \text{DECOMPRESS}(\mathcal{C}(\frac{1}{W} \sum_w \mathbf{g}_{t,w} + \mathbf{e}_{t,w})) \\ &= \text{DECOMPRESS}(\frac{1}{W} \mathcal{C}(\mathbf{g}_t + \mathbf{e}_t)). \end{aligned}$$

□

10.1.5 Proof of convergence (Theorem V)

Let us first state some standard assumptions we need for our proof.

Assumption I (*L-smooth function*). *The function f is assumed to be L -smooth and for any \mathbf{x}, \mathbf{y} the following holds:*

$$\begin{aligned} f(\mathbf{y}) - f(\mathbf{x}) &\leq \langle \nabla f(\mathbf{x}), \mathbf{y} - \mathbf{x} \rangle + \frac{L}{2} \|\mathbf{y} - \mathbf{x}\|^2, \text{ and} \\ \|\nabla f(\mathbf{x}) - \nabla f(\mathbf{y})\| &\leq L \|\mathbf{x} - \mathbf{y}\|. \end{aligned}$$

The second definition of smoothness is stronger and implies the first.

Assumption J (δ -approximate compressor (Karimireddy et al., 2019)). *Consider any $\mathbf{g}_t + \mathbf{e}_t$ encountered during the run of Algorithm 4. We assume there exists a $\delta > 0$ such that*

$$\|\Delta'_t - (\mathbf{g}_t + \mathbf{e}_t)\| \leq (1 - \delta) \|\mathbf{g}_t + \mathbf{e}_t\|^2.$$

Assumption K (*Moment bounds*). *For all iterations t while running Algorithm 4 with W workers, there exist constants σ and B such that*

$$\mathbb{E}_t[\|\mathbf{g}_t - \nabla f(\mathbf{x}_t)\|^2] \leq \frac{\sigma^2}{W}, \text{ and } \max(\mathbb{E}_t[\|\mathbf{g}_t\|^2], \mathbb{E}_t[\|\Delta'_t\|^2]) \leq B^2.$$

The key idea behind the convergence proof of (Karimireddy et al., 2019; Stich et al., 2018) is the use of an auxillary sequence $\{\tilde{\mathbf{x}}_t\}$. We will use a different sequence than the one in those papers. This is because of the presence of Nesterov *momentum* in (4) which makes the effective step-size larger.

We define $\tilde{\mathbf{x}}_0 := \mathbf{x}_0$ and then inductively for any $t \geq 0$ as

$$\tilde{\mathbf{x}}_{t+1} = \tilde{\mathbf{x}}_t - \gamma \left(1 + \frac{1}{1-\lambda} \right) \mathbf{g}_t. \quad (10.2)$$

Note that here we use the stochastic gradient computed at \mathbf{x}_t (instead of at $\tilde{\mathbf{x}}_t$ as in (3.1)).

Lemma 28 (Real and virtual sequence relation). *For the real sequence $\{\mathbf{x}_t\}$ as defined in (3.2) and virtual sequence $\{\tilde{\mathbf{x}}_t\}$ as in (10.2), the following holds almost surely:*

$$\tilde{\mathbf{x}}_t = \mathbf{x}_t + \frac{\gamma}{1-\lambda} ((2-\lambda)\mathbf{e}_t + \lambda\mathbf{m}_t).$$

Proof. This follows from a careful manipulation of the respective definitions in (3.2). First from the definition of the error \mathbf{e}_{t+1} we have:

$$\mathbf{e}_{t+1} = (\mathbf{g}_t + \mathbf{e}_t) - \Delta'_t \Rightarrow \Delta_t = \mathbf{g}_t + (\mathbf{e}_t - \mathbf{e}_{t+1}).$$

Then from the definition of the momentum sequence, \mathbf{m}_{t+1} we have:

$$(1-\lambda)\mathbf{m}_{t+1} = \mathbf{m}_{t+1} - \lambda\mathbf{m}_{t+1} = \lambda(\mathbf{m}_t - \mathbf{m}_{t+1}) + \gamma\Delta'_t.$$

Now, using these relations in the update relation of \mathbf{x}_t gives:

$$\begin{aligned} \mathbf{x}_{t+1} &= \mathbf{x}_t - \gamma(\Delta'_t + \mathbf{m}_{t+1}) \\ &= \mathbf{x}_t - \gamma \left(\Delta'_t + \frac{1}{1-\lambda} \Delta'_t + \frac{\lambda}{1-\lambda} (\mathbf{m}_t - \mathbf{m}_{t+1}) \right) \\ &= \mathbf{x}_t - \gamma \left(\frac{2-\lambda}{1-\lambda} (\mathbf{g}_t + \mathbf{e}_t - \mathbf{e}_{t+1}) + \frac{\lambda}{1-\lambda} (\mathbf{m}_t - \mathbf{m}_{t+1}) \right) \\ &= \mathbf{x}_t + \frac{\gamma}{1-\lambda} ((2-\lambda)\mathbf{e}_t + \lambda\mathbf{m}_t) - \gamma \frac{2-\lambda}{1-\lambda} \mathbf{g}_t - \frac{\gamma}{1-\lambda} ((2-\lambda)\mathbf{e}_{t+1} + \lambda\mathbf{m}_{t+1}). \end{aligned}$$

This shows that the following relation closely resembling the update of $\{\tilde{\mathbf{x}}_t\}$ holds:

$$\mathbf{x}_{t+1} + \frac{\gamma}{1-\lambda} ((2-\lambda)\mathbf{e}_{t+1} + \lambda\mathbf{m}_{t+1}) = \mathbf{x}_t + \frac{\gamma}{1-\lambda} ((2-\lambda)\mathbf{e}_t + \lambda\mathbf{m}_t) - \gamma \frac{2-\lambda}{1-\lambda} \mathbf{g}_t.$$

We will finish our proof by induction. For $t = 0$, $\mathbf{m}_0 = 0$, $\mathbf{e}_0 = 0$ and $\tilde{\mathbf{x}}_0 = \mathbf{x}_0$. Hence the claim holds for $t = 0$. Assuming it holds for some $t \geq 0$,

$$\begin{aligned} \mathbf{x}_{t+1} + \frac{\gamma}{1-\lambda} ((2-\lambda)\mathbf{e}_{t+1} + \lambda\mathbf{m}_{t+1}) &= \mathbf{x}_t + \frac{\gamma}{1-\lambda} ((2-\lambda)\mathbf{e}_t + \lambda\mathbf{m}_t) - \gamma \frac{2-\lambda}{1-\lambda} \mathbf{g}_t \\ &= \tilde{\mathbf{x}}_t - \gamma \frac{2-\lambda}{1-\lambda} \mathbf{g}_t \\ &= \tilde{\mathbf{x}}_{t+1}. \end{aligned}$$

Thus we have an inductive proof of our claim. \square

Lemma 29 (Distance between virtual and real sequences). *For the updates of Algorithm 4, given assumptions J and K we have:*

$$\mathbb{E}\|\tilde{\mathbf{x}}_t - \mathbf{x}_t\|^2 \leq \frac{2\gamma^2 B^2}{(1-\lambda)^2} \left(\frac{4(1-\delta)(2-\lambda)^2}{\delta^2} + \frac{\lambda^2}{(1-\lambda)^2} \right).$$

Proof. First let us obtain a bound on the norm of the momentum:

$$\begin{aligned} \|\mathbf{m}_{t+1}\|^2 &= \|\lambda \mathbf{m}_t + \Delta'_t\|^2 \\ &\leq \frac{\lambda^2}{\eta} \|\mathbf{m}_t\|^2 + \frac{1}{1-\eta} \|\Delta'_t\|^2 \\ &\leq \sum_{i=0}^t [\lambda^2/\eta]^{t-i} \frac{1}{1-\eta} \|\Delta'_i\|^2. \end{aligned}$$

The first inequality above used Young's inequality and holds for any $\eta \in (0, 1)$. Next we use assumption K about bounded moments as:

$$\begin{aligned} \mathbb{E}[\|\mathbf{m}_{t+1}\|^2] &\leq \sum_{i=0}^t [\lambda^2/\eta]^{t-i} \frac{1}{1-\eta} \mathbb{E}[\|\Delta'_i\|^2] \\ &\leq \sum_{i=0}^t [\lambda^2/\eta]^{t-i} \frac{1}{1-\eta} B^2 \\ &\leq \sum_{i=0}^{\infty} [\lambda^2/\eta]^i \frac{1}{1-\eta} B^2 \\ &= \frac{B^2}{(1-\lambda^2/\eta)(1-\eta)} \\ &= \frac{B^2}{(1-\lambda)^2}. \end{aligned}$$

We used $\eta = \lambda$ in the final step. Next we recall Lemma 3 from (Karimireddy et al., 2019) to claim that Assumption J and K imply:

$$\mathbb{E}[\mathbf{e}_t]^2 \leq \frac{4(1-\delta)B^2}{\delta^2}.$$

Using the derived inequalities in Lemma 28, we get

$$\begin{aligned} \mathbb{E}\|\tilde{\mathbf{x}}_t - \mathbf{x}_t\|^2 &= \mathbb{E}\left\| \frac{\gamma}{1-\lambda} ((2-\lambda)\mathbf{e}_t + \lambda \mathbf{m}_t) \right\|^2 \\ &\leq \frac{2\gamma^2(2-\lambda)^2}{(1-\lambda)^2} \mathbb{E}\|\mathbf{e}_t\|^2 + \frac{2\gamma^2\lambda^2}{(1-\lambda)^2} \mathbb{E}\|\mathbf{m}_t\|^2 \\ &\leq \frac{2\gamma^2(2-\lambda)^2}{(1-\lambda)^2} \frac{4(1-\delta)B^2}{\delta^2} + \frac{2\gamma^2\lambda^2}{(1-\lambda)^2} \frac{B^2}{(1-\lambda)^2} \\ &= \frac{2\gamma^2 B^2}{(1-\lambda)^2} \left(\frac{4(1-\delta)(2-\lambda)^2}{\delta^2} + \frac{\lambda^2}{(1-\lambda)^2} \right). \end{aligned}$$

□

We are finally ready to prove the main theorem on convergence of Error-Feedback SGD when including momentum.

Proof of main theorem. Consider the virtual sequence $\tilde{\mathbf{x}}_t$ as defined as in (10.2):

$$\tilde{\mathbf{x}}_{t+1} = \tilde{\mathbf{x}}_t - \tilde{\gamma} \mathbf{g}_t.$$

We will view the actual sequence $\{\mathbf{x}_t\}$ as being an approximation of the actual sequence $\tilde{\mathbf{x}}_t$.

Using Assumption I, the following statements hold almost surely:

$$\begin{aligned} \mathbb{E}_t[f(\tilde{\mathbf{x}}_{t+1}) - f(\tilde{\mathbf{x}}_t)] &\leq \mathbb{E}[\langle \nabla f(\tilde{\mathbf{x}}_t), \tilde{\mathbf{x}}_{t+1} - \tilde{\mathbf{x}}_t \rangle] + \frac{L}{2} \mathbb{E}[\|\tilde{\mathbf{x}}_{t+1} - \tilde{\mathbf{x}}_t\|^2] \\ &= -\tilde{\gamma} \langle \nabla f(\tilde{\mathbf{x}}_t), \mathbb{E}[\mathbf{g}_t] \rangle + \frac{L\tilde{\gamma}^2}{2} \mathbb{E}[\|\mathbf{g}_t\|^2] \\ &= -\tilde{\gamma} \langle \nabla f(\tilde{\mathbf{x}}_t), \nabla f(\mathbf{x}_t) \rangle + \frac{L\tilde{\gamma}^2}{2} \|\nabla f(\mathbf{x}_t)\|^2 + \frac{L\tilde{\gamma}^2\sigma^2}{2W} \\ &= \left(\frac{L\tilde{\gamma}^2}{2} - \tilde{\gamma} \right) \|\nabla f(\mathbf{x}_t)\|^2 + \tilde{\gamma} \langle \nabla f(\mathbf{x}_t), \nabla f(\mathbf{x}_t) - \nabla f(\tilde{\mathbf{x}}_t) \rangle + \frac{L\tilde{\gamma}^2\sigma^2}{2W} \\ &\leq \left(\frac{L\tilde{\gamma}^2}{2} - \tilde{\gamma} \right) \|\nabla f(\mathbf{x}_t)\|^2 + \frac{\tilde{\gamma}}{4} \|\nabla f(\mathbf{x}_t)\|^2 + \tilde{\gamma} \|\nabla f(\mathbf{x}_t) - \nabla f(\tilde{\mathbf{x}}_t)\|^2 + \frac{L\tilde{\gamma}^2\sigma^2}{2W}. \end{aligned}$$

The last step follows from the mean-value inequality. We never really encounter the gradient $\nabla f(\tilde{\mathbf{x}}_t)$ during the course of the algorithm and so cannot directly control it. We can instead replace this term using the second notion of smoothness from assumption I and Lemma 29:

$$\|\nabla f(\tilde{\mathbf{x}}_t) - \nabla f(\mathbf{x}_t)\|^2 \leq L^2 \|\tilde{\mathbf{x}}_t - \mathbf{x}_t\|^2 \leq \frac{2L^2\gamma^2 B^2}{(1-\lambda)^2} \left(\frac{4(1-\delta)(2-\lambda)^2}{\delta^2} + \frac{\lambda^2}{(1-\lambda)^2} \right).$$

Recall that $2L\tilde{\gamma} \leq 1$. We thus have

$$\begin{aligned} \mathbb{E}_t[f(\tilde{\mathbf{x}}_{t+1}) - f(\tilde{\mathbf{x}}_t)] &\leq \left(\frac{L\tilde{\gamma}^2}{2} - \tilde{\gamma} + \frac{\tilde{\gamma}}{4} \right) \|\nabla f(\mathbf{x}_t)\|^2 + \tilde{\gamma} \frac{2L^2\gamma^2 B^2}{(1-\lambda)^2} \left(\frac{4(1-\delta)(2-\lambda)^2}{\delta^2} + \frac{\lambda^2}{(1-\lambda)^2} \right) + \frac{L\tilde{\gamma}^2\sigma^2}{2W} \\ &\leq -\frac{\tilde{\gamma}}{4} \|\nabla f(\mathbf{x}_t)\|^2 + \tilde{\gamma}^3 2L^2 B^2 \left(\frac{4(1-\delta)}{\delta^2} + \frac{\lambda^2}{(1-\lambda)^2(2-\lambda)^2} \right) + \frac{L\tilde{\gamma}^2\sigma^2}{2W} \\ &= -\frac{\tilde{\gamma}}{4} \|\nabla f(\mathbf{x}_t)\|^2 + \tilde{\gamma}^3 + \tilde{\gamma}^2 \frac{L\sigma^2}{2W}. \end{aligned}$$

Shifting the gradient term to the left and averaging over T gives:

$$\begin{aligned} \frac{1}{T+1} \sum_{t=0}^T \mathbb{E}[\|\nabla f(\mathbf{x}_t)\|^2] &\leq \frac{4}{\tilde{\gamma}(T+1)} \left(\sum_{t=0}^T f(\tilde{\mathbf{x}}_t) - f(\tilde{\mathbf{x}}_{t+1}) \right) + \tilde{\gamma} \frac{4\sigma^2}{W} + \tilde{\gamma}^2 C \\ &= \frac{4(f(\mathbf{x}_0) - f^*)}{\tilde{\gamma}(T+1)} + \frac{4\tilde{\gamma}\sigma^2}{W} + \tilde{\gamma}^2 C. \end{aligned}$$

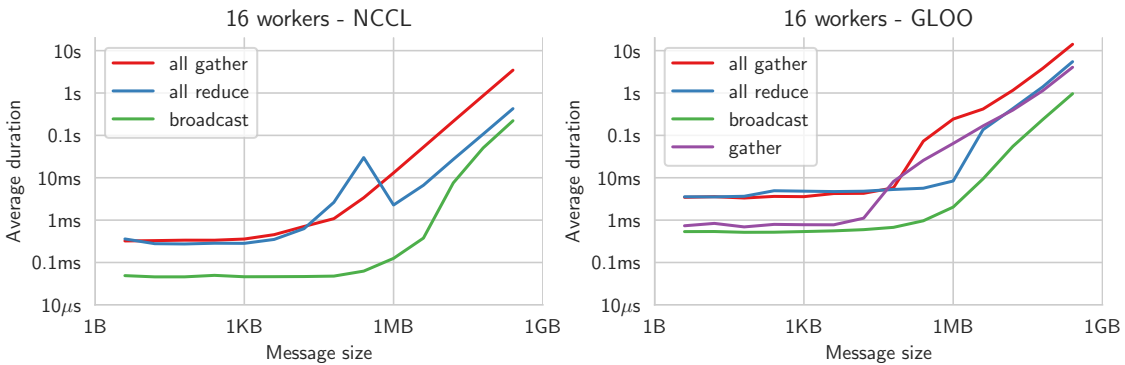
This finishes the proof of the theorem.

10.2 Cluster specifications

- 8 nodes
- GPUs: 2× Nvidia GeForce GTX Titan X with 12 GB memory per node
- GPU connection: traversing PCIe and the SMP interconnect between NUMA nodes
- CPU: Intel Xeon E5-2680 v3 @ 2.50Ghz, 48 cores
- System memory: 251GiB
- Ethernet: 10Gbit/s SFI/SFP+
- *Fat tree* network topology
- Runing PYTORCH 1.1 on Anaconda Python 3.7

Timings of collective communication operations

The figure below shows timings for the NCCL backend, which is the default in our experiments, and the GLOO backend. Note that NCCL does not support the ‘gather’ operation in PYTORCH at the time of writing.



10.3 Convergence curves

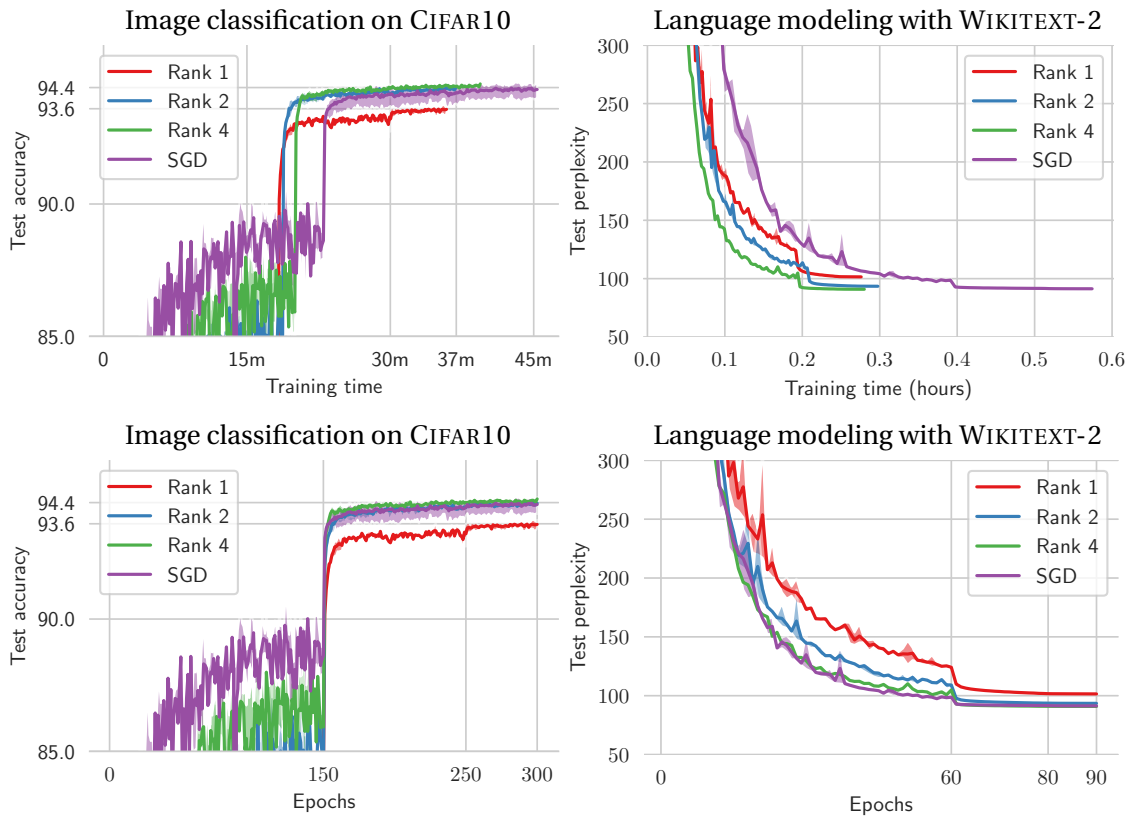


Figure 10.1 – Convergence curves of POWERSGD with varying rank. This figure is meant to give context to the final results and timings presented in Table 3.3. In two different tasks, POWERSGD with high enough rank can achieve the test quality of full-precision SGD with lower wall-clock duration. Contrary to Table 3.3, these timings include testing overhead at the end of each epoch, checkpointing, and other bookkeeping. Shaded areas show the min—max values over 3 replications of the experiments.

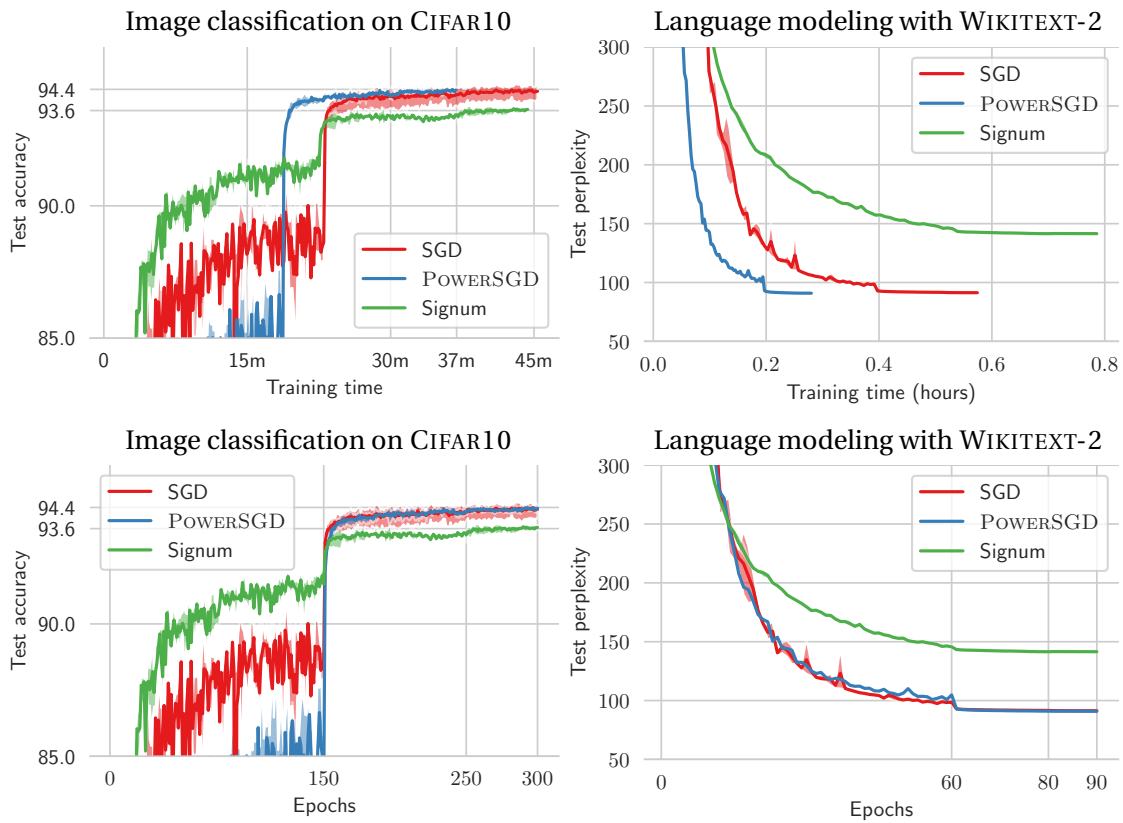


Figure 10.2 – Convergence curves comparing POWERSGD to the Signum optimizer (Bernstein et al., 2019) (with tuned learning rate). Out of the compared methods, Signum came out as the most competitive. This figure is meant to give context to the final results and timings presented in Table 3.6. Contrary to Table 3.3, these timings include testing overhead at the end of each epoch, checkpointing, and other bookkeeping. Shaded areas show the min—max values over 3 replications of the experiments.

10.4 Language Modeling with Transformers

In this case study, we assess PowerSGD’s universality and ease of tuning. We implemented PowerSGD communication in Facebook AI Research’s fairseq library (Ott et al., 2019). We trained fairseq’s language modeling example¹ with transformers (Baeovski and Auli, 2019) on Google’s public cloud. The communication infrastructure, hardware, number of workers (32), and model architecture are all different from any experiments we have conducted before. See Table 10.1 for details.

The results of our experiments for various ranks are shown in Figure 10.3 and Table 10.2. For this task, we need a higher rank than previously (32 vs 4) to achieve a validation loss competitive to uncompressed SGD. We hypothesize this may be due differences in architecture to the cosine learning rate schedule. Nevertheless, even at this higher rank, we achieve a time-to-accuracy (to loss = 5) of around 1.5× and a compression ratio of 14×. These numbers could probably be further improved by re-tuning learning-rate-related hyperparameters.

Table 10.1 – Experimental setting for the experiments in Appendix 10.4

Dataset	WikiText-103
Architecture	Transformer-based (Baeovski and Auli, 2019)
Framework & defaults	https://github.com/pytorch/fairseq/tree/920b85d4bd39e181229db5639c701c854c83ec5c/examples/language_model
Number of workers	32
Backend	NCCL (fastest in PyTorch)
Hardware	n1-standard-8 nodes on Google Cloud with 1 Nvidia Tesla K80 GPU
Hyperparameters	Taken from the example, not re-tuned, with minor changes for the higher number of workers and different GPU memory:
lr period updates	16875
max update	17875
max tokens (valid)	1536 (to fit on a K80 gpu)
tokens per sample	1536 (to fit on a K80 gpu)
warmup updates	1000
update freq	[1] — don’t aggregate multiple mini-batches locally
Optimizer	original: Nesterov accelerated gradient, we just added PowerSGD for communication
Learning rate	original cosine schedule from the example
Float precision	32-bit (16-bit is unavailable on the K80)
Repetitions	1

¹https://github.com/pytorch/fairseq/tree/920b85d4bd39e181229db5639c701c854c83ec5c/examples/language_model

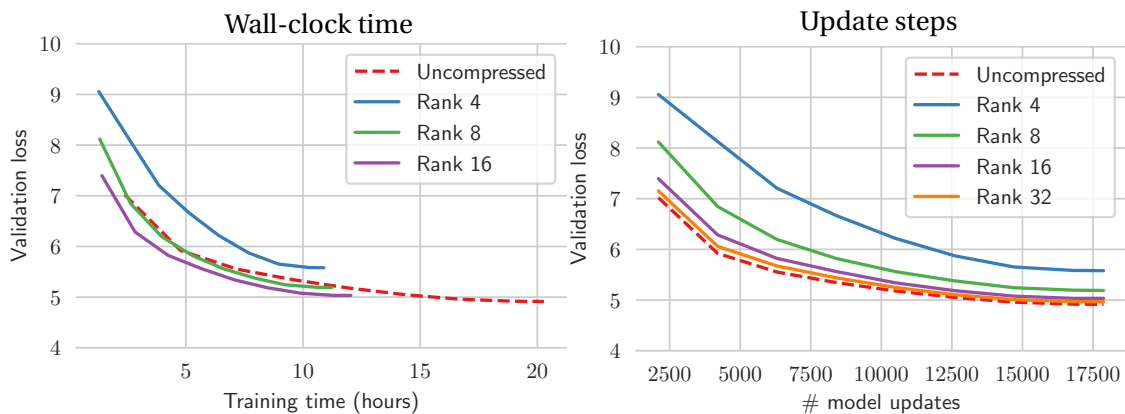


Figure 10.3 – Language Modeling on WIKITEXT-2 with Transformers. With a large enough rank, POWERSGD can roughly match the validation loss of full-precision SGD in the same number of iterations. A speedup of $1.5\times$ in time-to-accuracy (loss=5) is achieved with a rank of 16.

Table 10.2 – POWERSGD for Language Modeling with Transformers. With rank 32, POWERSGD achieves similar validation loss to uncompressed SGD in the same number of update steps. At this rank, the compression ratio is $14\times$ and we can train the model in 12h compared to 20h for the baseline.

Compression	Total training time for 17875 updates	Compression ratio	Validation loss at 17875 updates
Uncompressed	<div><div></div><div></div><div></div></div> 20h	1×	4.92
Rank 4	<div><div></div><div></div><div></div></div> 11h	105×	5.58
Rank 8	<div><div></div><div></div><div></div></div> 11h	55×	5.19
Rank 16	<div><div></div><div></div><div></div></div> 12h	28×	5.03
Rank 32	<div><div></div><div></div><div></div></div> 13h	14×	4.97

■ Forward pass ■ Backward pass ■ Gradient exchange including computation

10.5 The need for error feedback

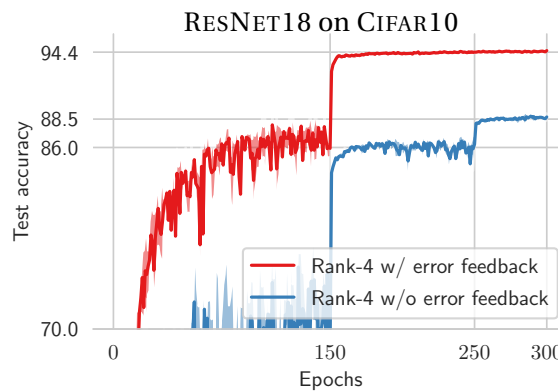


Figure 10.4 – PowerSGD with and without error feedback compared. While rank-4 PowerSGD achieves the same test accuracy as full-precision SGD, the same method without error feedback does not converge to a good accuracy at all. Both experiments use the same learning rate that was tuned for full-precision SGD.

10.6 Network parameters

Table 10.3 – Parameters in the ResNet18 architecture and their shapes. The table shows the per-tensor compression ratio achieved by rank- r POWERSGD.

Parameter	Gradient tensor shape	Matrix shape	Uncompressed	Compression
layer4.1.conv2	$512 \times 512 \times 3 \times 3$	512×4608	9216 KB	$461/r \times$
layer4.0.conv2	$512 \times 512 \times 3 \times 3$	512×4608	9216 KB	$461/r \times$
layer4.1.conv1	$512 \times 512 \times 3 \times 3$	512×4608	9216 KB	$461/r \times$
layer4.0.conv1	$512 \times 256 \times 3 \times 3$	512×2304	4608 KB	$419/r \times$
layer3.1.conv2	$256 \times 256 \times 3 \times 3$	256×2304	2304 KB	$230/r \times$
layer3.1.conv1	$256 \times 256 \times 3 \times 3$	256×2304	2304 KB	$230/r \times$
layer3.0.conv2	$256 \times 256 \times 3 \times 3$	256×2304	2304 KB	$230/r \times$
layer3.0.conv1	$256 \times 128 \times 3 \times 3$	256×1152	1152 KB	$209/r \times$
layer2.1.conv2	$128 \times 128 \times 3 \times 3$	128×1152	576 KB	$115/r \times$
layer2.1.conv1	$128 \times 128 \times 3 \times 3$	128×1152	576 KB	$115/r \times$
layer2.0.conv2	$128 \times 128 \times 3 \times 3$	128×1152	576 KB	$115/r \times$
layer4.0.shortcut.0	$512 \times 256 \times 1 \times 1$	512×256	512 KB	$171/r \times$
layer2.0.conv1	$128 \times 64 \times 3 \times 3$	128×576	288 KB	$105/r \times$
layer1.1.conv1	$64 \times 64 \times 3 \times 3$	64×576	144 KB	$58/r \times$
layer1.1.conv2	$64 \times 64 \times 3 \times 3$	64×576	144 KB	$58/r \times$
layer1.0.conv2	$64 \times 64 \times 3 \times 3$	64×576	144 KB	$58/r \times$
layer1.0.conv1	$64 \times 64 \times 3 \times 3$	64×576	144 KB	$58/r \times$
layer3.0.shortcut.0	$256 \times 128 \times 1 \times 1$	256×128	128 KB	$85/r \times$
layer2.0.shortcut.0	$128 \times 64 \times 1 \times 1$	128×64	32 KB	$43/r \times$
linear	10×512	10×512	20 KB	$10/r \times$
conv1	$64 \times 3 \times 3 \times 3$	64×27	7 KB	$19/r \times$
Bias vectors (total)			38 KB	None
Total			43 MB	$243/r \times$

10.7. Compressor implementation details

Table 10.4 – Parameters in the LSTM architecture and their shapes. The table shows the per-tensor compression ratio achieved by rank- r POWERSGD.

Parameter	Gradient tensor shape	Matrix shape	Uncompressed	Compression
encoder	28869×650	28869×650	73300 KB	$636/r \times$
rnn-ih-l0	2600×650	2600×650	6602 KB	$520/r \times$
rnn-hh-l0	2600×650	2600×650	6602 KB	$520/r \times$
rnn-ih-l1	2600×650	2600×650	6602 KB	$520/r \times$
rnn-hh-l1	2600×650	2600×650	6602 KB	$520/r \times$
rnn-ih-l2	2600×650	2600×650	6602 KB	$520/r \times$
rnn-hh-l2	2600×650	2600×650	6602 KB	$520/r \times$
Bias vectors (total)			174 KB	None
Total			110 MB	$310/r \times$

10.7 Compressor implementation details

10.7.1 Random Block

This implements compression for error-feedback with momentum (Algorithm 4).

Algorithm 11 Random Block compression

```

1: function COMPRESS(update matrix  $M \in \mathbb{R}^{n \times m}$ )
2:   Treat  $M$  as a vector of length  $nm$ .
3:   Sample an index  $s$  uniformly between 0 and  $nm-1$ , using the same seed on all workers.
4:   The block length  $b$  is set to  $(m+n)r$  to match rank- $r$  POWERSGD.
5:   return A consecutive memory slice  $S = M(s : s + b)$ .
6: end function
7: function AGGREGATE+DECOMPRESS(worker's slices  $S_1 \dots S_W$ )
8:    $\hat{M} \leftarrow \mathbf{0} \in \mathbb{R}^{n \times m}$ 
9:    $\hat{M}(s : s + b) \leftarrow \frac{1}{W} \sum_{i=1}^W S_i$  ▷ using all-reduce
10:  return  $\hat{M}$ 
11: end function

```

10.7.2 Random K

This implements compression for error-feedback with momentum (Algorithm 4).

Algorithm 12 Random K compression

```

1: function COMPRESS(update matrix  $M \in \mathbb{R}^{n \times m}$ )
2:   Treat  $M$  as a vector of length  $nm$ .
3:   The number of samples  $b$  is set to  $(m + n)r$  to match rank- $r$  POWERSGD.
4:   Sample a set of  $b$  indices  $I$  without replacement, using the same seed on all workers.
5:   return Looked up values  $S = M(I)$ .
6: end function
7: function AGGREGATE+DECOMPRESS(worker's values  $S_1 \dots S_W$ )
8:    $\hat{M} \leftarrow \mathbf{0} \in \mathbb{R}^{n \times m}$ 
9:    $\hat{M}(I) \leftarrow \frac{1}{W} \sum_{i=1}^W S_i$  ▷ using all-reduce
10:  return  $\hat{M}$ 
11: end function

```

Sampling of indices We sample random indices on the CPU using Numpy. This operation is relatively expensive. Together with the many random lookups, this explains why Random K compression is significantly slower than Random Block compression.

10.7.3 Sign+Norm

This implements compression for error-feedback with momentum (Algorithm 4).

Algorithm 13 Sign+Norm compression

```

1: function COMPRESS(update matrix  $M \in \mathbb{R}^{n \times m}$ )
2:   Compute the signs  $S \in \{-1, 1\}^{n \times m}$  of  $M$ 
3:   Compute the  $L_1$  norm  $\ell$  of  $M$ .
4:   return  $(\ell, S)$ 
5: end function
6: function AGGREGATE+DECOMPRESS(worker's norms  $\ell_1 \dots \ell_W$  and signs  $S_1 \dots S_W$ )
7:   return  $\frac{1}{W} \sum_{i=1}^W \frac{\ell_i}{nm} S_i$  ▷ Executed on all workers using NCCL's all-gather
8: end function

```

Because PYTORCH does not natively support data types smaller than 8 bits per scalar, we use a C++ extension (Bernstein et al., 2019) to actually send single bits to other workers. The employed all-gather operation from NCCL is faster than aggregation using a parameter server using GLOO. We cannot implement a parameter server in NCCL due to lack of a ‘gather’ operation.

10.7.4 Top K

This implements compression for error-feedback with momentum (Algorithm 4).

Algorithm 14 Top K compression

```

1: function COMPRESS(update matrix  $M \in \mathbb{R}^{n \times m}$ )
2:   Treat  $M$  as a vector of length  $nm$ .
3:   The number of samples  $b$  is set to  $(m + n)r$  to match rank- $r$  POWERSGD.
4:   Construct a list of  $b$  indices  $I$  corresponding to the top absolute values in  $M$ .
5:   return Looked up values  $S = M(I)$  and indices  $I$ .
6: end function
7: function AGGREGATE+DECOMPRESS(worker's values  $S_1 \dots S_W$  and indices  $I_1 \dots I_W$ )
8:    $\hat{M} \leftarrow \mathbf{0} \in \mathbb{R}^{n \times m}$ 
9:   for worker index  $i$  in  $1, \dots, W$  do
10:     $\hat{M}(I_i) \leftarrow \frac{1}{W} S_i$  ▷ using all-gather in NCCL
11:   end for
12:   return  $\hat{M}$ 
13: end function

```

The employed all-gather operation from NCCL is faster than aggregation using a parameter server using GLOO. We cannot implement a parameter server in NCCL due to lack of a ‘gather’ operation.

10.7.5 Signum

This is our implementation of the Signum compression algorithm by (Bernstein et al., 2019). We run it in its original form, without error feedback, with momentum of 0.9, and a learning rate tuned based on 5 experiments in the 16-worker setting.

Algorithm 15 Signum compression

```

1: function COMPRESS(update matrix  $M \in \mathbb{R}^{n \times m}$ )
2:   Compute the signs  $S \in \{-1, 1\}^{n \times m}$  of  $M$ 
3:   return  $S$ 
4: end function
5: function AGGREGATE+DECOMPRESS(worker's signs  $S_1 \dots S_W$ )
6:   return  $\text{SIGN}(\sum_{i=1}^W S_i)$  ▷ Majority vote, on all workers using NCCL's all-gather
7: end function

```

Because PyTorch does not natively support data types smaller than 8 bits per number, we use a C++ extension (Zhao, 2019) to actually send single bits to other workers. The employed all-gather operation from NCCL is faster than aggregation using a parameter server using GLOO. We cannot implement a parameter server in NCCL due to lack of a ‘gather’ operation.

10.7.6 Atomo

This is our implementation of the Spectral Atomo algorithm presented by (Wang et al., 2018). We run it in its original form, without error feedback, with momentum of 0.9, and a learning rate tuned based on 4 experiments in the 16-worker setting.

Matix shape Atomo differs from POWERSGD in how it treats tensors as matrices. This results in lower compression at the same rank.

Number of sampled components Atomo decomposes gradient matrices M using a Singular Value Decomposition into $M \sim \sum_i U_i S_{ii} V_i^\top$ and importance-samples components from this summation based on probabilities derived from the absolute singular values S_{ii} . The probabilities are such, that the expected number of samples components is equal to the target rank r , but there is no guarantee. We modify the algorithm to always use exactly r components, to allow for faster communication. We achieve this by repeating the sampling procedure until the number of selected components is r . This has no significant impact on the runtime performance.

Algorithm 16 Rank- r Spectral-Atomo compression

```

1: function COMPRESS(update matrix  $M \in \mathbb{R}^{n \times m}$ )
2:    $U, S, V \leftarrow \text{SVD}(M)$ . ▷ on CPU using Numpy, faster than PYTORCH
3:   Compute Atomo probabilities  $p_1 \dots p_k$  from  $S_{11}, \dots S_{kk}$ . ▷ see (Wang et al., 2018).
4:   Sampling: include index  $i$  independently with probability  $p_i$ .
5:   Repeat sampling until a set of  $r$  indices  $C$  is selected. ▷ our modification (see above)
6:   return  $\{(U_i \cdot S_{ii} / p_i, V_i) \mid i \in C\}$  as two matrices  $U' \in \mathbb{R}^{n \times r}$  and  $V' \in \mathbb{R}^{m \times r}$ .
7: end function
8: function AGGREGATE+DECOMPRESS(rank- $r$  approximations  $(U'_1, V'_1) \dots (U'_W, V'_W)$  for each
   worker)
9:   return  $\sum_{i=1}^W U'_i V_i'^\top$  ▷ using all-gather in NCCL
10: end function

```

The employed all-gather operation from NCCL is faster than aggregation using a parameter server using GLOO. We cannot implement a parameter server in NCCL due to lack of a ‘gather’ operation.

10.7.7 Best-approximation POWERSGD

This variant is the same as POWERSGD (Algorithm 3), but with more steps of subspace iteration, and without reuse of previous steps. We find that 4 steps of subspace iterations (8 matrix multiplications) is enough to converge to the best low-rank approximation of gradient matrices, when measuring final test accuracy achieved by POWERSGD.

10.8 Performance optimizations

Because we compare timings, we have aimed to optimize all compared optimizers to a similar level. For sign-based methods, we used a publicly available C++ library by (Bernstein et al., 2019) to efficiently pack the signs into bitmaps, an operation which is not supported by PyTorch natively. For Atomo, we have benchmarked the SVD operation on the GPU and CPU, and chose the faster CPU implementation. For all methods, we pack all gradient tensors into one flat buffer to reduce the number of communications. Where possible, we overlay communication with computation. Algorithms that do not support all-reduce are implemented using NCCL’s all-gather, which is faster than a parameter server with GLOO.²

10.9 Learning rate tuning

For each task and each optimization algorithm without error feedback, learning rates were tuned separately. For algorithms based on error feedback with momentum, we use the learning rate tuned for SGD.

Learning rates are defined as rates for 1 worker, and scaled linearly with 5-epoch warmup to the number of workers (16 by default). We tune them in the 16-worker setting.

We determine the best learning rate by comparing test accuracy of one replication after running the full number of epochs. We start training with 3 different learning rates, a factor 2 apart, based on commonly used rates for the optimizer, and if the best learning rate is either the lower or higher end, we extended the range.

For CIFAR10, the rates considered for SGD were [0.05, 0.1, 0.2], we chose 0.1. For rank-2 Spectral Atomo, we considered [0.025, 0.05, 0.1, 0.2] and chose 0.1. For Signum, we considered [2e-5, 5e-5, 1e-4, 2e-4] and chose 5e-5.

For WIKITEXT-2, the rates considered for SGD were [0.6, 1.25, 2.5, 5, 10], we chose 1.25. For Signum, we considered [2e-4, 1e-1, 5e-5, 1e-5, 1e-6], and chose 1e-5.

We have not tuned the momentum parameter or L_2 , weight decay parameters or learning rate schedule for any experiment.

²‘reduce’+‘gather’ (parameter server communication) with GLOO takes longer than all-gather with NCCL, as shown in Appendix 10.2. NCCL in PyTorch currently lacks support for a ‘gather’ operator.

11 Appendix for SCAFFOLD

11.1 Related work and significance

Federated learning. As stated earlier, federated learning involves learning a centralized model from distributed client data. This centralized model benefits from all client data and can often result in a beneficial performance e.g. in including next word prediction (Hard et al., 2018; Yang et al., 2018), emoji prediction (Ramaswamy et al., 2019), decoder models (Chen et al., 2019b), vocabulary estimation (Chen et al., 2019a), low latency vehicle-to-vehicle communication (Samarakoon et al., 2018), and predictive models in health (Brisimi et al., 2018). Nevertheless, federated learning raises several types of issues and has been the topic of multiple research efforts studying the issues of generalization and fairness (Mohri et al., 2019; Li et al., 2019b), the design of more efficient communication strategies (Konečný et al., 2016; Suresh et al., 2017; Stich et al., 2018; Karimireddy et al., 2019; Basu et al., 2019), the study of lower bounds (Woodworth et al., 2018), differential privacy guarantees (Agarwal et al., 2018), security (Bonawitz et al., 2017), etc. We refer to Kairouz et al. (2019) for an in-depth survey of this area.

Convergence of FEDAVG For identical clients, FEDAVG coincides with parallel SGD analyzed by (Zinkevich et al., 2010) who proved asymptotic convergence. Stich (2019a) and, more recently Stich and Karimireddy (2019); Patel and Dieuleveut (2019); Khaled et al. (2020), gave a sharper analysis of the same method, under the name of local SGD, also for identical functions. However, there still remains a gap between their upper bounds and the lower bound of Woodworth et al. (2018). The analysis of FEDAVG for heterogeneous clients is more delicate due to the afore-mentioned client-drift, first empirically observed by Zhao et al. (2018). Several analyses bound this drift by assuming bounded gradients (Wang et al., 2019; Yu et al., 2019b), or view it as additional noise (Khaled et al., 2020), or assume that the client optima are ϵ -close (Li et al., 2018b; Haddadpour and Mahdavi, 2019). In a concurrent work, (Liang et al., 2019) propose to use variance reduction to deal with client heterogeneity but still show rates slower than SGD. We summarize the communication complexities of different methods for heterogeneous clients in Table 4.2.

Variance reduction. The use of *control variates* is a classical technique to reduce variance in Monte Carlo sampling methods (cf. (Glasserman, 2013)). In optimization, they were used for finite-sum minimization by SVRG (Johnson and Zhang, 2013; Zhang et al., 2013a) and then in SAGA (Defazio et al., 2014) to simplify the linearly convergent method SAG (Schmidt et al., 2017). Numerous variations and extensions of the technique are studied in (Hanzely and Richtárik, 2019). Starting from (Reddi et al., 2016a), control variates have also frequently been used to reduce variance in finite-sum non-convex settings (Reddi et al., 2016c; Nguyen et al., 2018; Fang et al., 2018; Tran-Dinh et al., 2019). Further, they are used to obtain linearly converging decentralized algorithms under the guise of ‘gradient-tracking’ in (Shi et al., 2015; Nedich et al., 2016) and for gradient compression as ‘compressed-differences’ in (Mishchenko et al., 2019). Our method can be viewed as seeking to remove the ‘client-variance’ in the gradients across the clients, though there still remains additional stochasticity as in (Kulunchakov and Mairal, 2019), which is important in deep learning (Defazio and Bottou, 2019).

Distributed optimization. The problem of client-drift we described is a common phenomenon in distributed optimization. In fact, classic techniques such as ADMM mitigate this drift, though they are not applicable in federated learning. For well structured convex problems, CoCoA (Smith et al., 2018) and its extensions (Karimireddy et al., 2018c) use the dual variable as the control variates, enabling flexible distributed methods. This can also be extended to include second order information (Dünner et al., 2018; Karimireddy et al., 2018b). DANE by (Shamir et al., 2014) obtain a closely related primal only algorithm, which was later accelerated by Reddi et al. (2016b) and recently extended to federated learning (Li et al., 2020). SCAFFOLD CAN BE VIEWED AS AN IMPROVED VERSION OF DANE WHERE A FIXED NUMBER OF (STOCHASTIC) GRADIENT STEPS ARE USED IN PLACE OF A PROXIMAL POINT UPDATE. IN A SIMILAR SPIRIT, DISTRIBUTED VARIANCE REDUCTION TECHNIQUES HAVE BEEN PROPOSED FOR THE FINITE-SUM CASE (LEE ET AL., 2015; KONEČNÝ ET AL., 2016; CEN ET AL., 2019). HOWEVER, THESE METHODS ARE RESTRICTED TO FINITE-SUMS AND ARE NOT APPLICABLE TO THE STOCHASTIC SETTING STUDIED HERE.

11.2 Technicalities

WE EXAMINE SOME ADDITIONAL DEFINITIONS AND INTRODUCE SOME TECHNICAL LEMMAS.

11.2.1 Additional definitions

WE MAKE PRECISE A FEW DEFINITIONS AND EXPLAIN SOME OF THEIR IMPLICATIONS.

(A3) f_i IS μ -CONVEX FOR $\mu \geq 0$ AND SATISFIES:

$$\langle \nabla f_i(\mathbf{x}), \mathbf{y} - \mathbf{x} \rangle \leq -\left(f_i(\mathbf{x}) - f_i(\mathbf{y}) + \frac{\mu}{2} \|\mathbf{x} - \mathbf{y}\|^2\right), \text{ FOR ANY } i, \mathbf{x}, \mathbf{y}.$$

HERE, WE ALLOW THAT $\mu = 0$ (WE REFER TO THIS CASE AS THE GENERAL CONVEX CASE AS

OPPOSED TO STRONGLY CONVEX). IT IS ALSO POSSIBLE TO GENERALIZE ALL PROOFS HERE TO THE WEAKER NOTION OF PL-STRONG CONVEXITY (KARIMI ET AL., 2016).

(A4) $g_i(\mathbf{x}) := \nabla f_i(\mathbf{x}; \zeta_i)$ IS UNBIASED STOCHASTIC GRADIENT OF f_i WITH **BOUNDED VARIANCE**

$$\mathbb{E}_{\zeta_i} [\|g_i(\mathbf{x}) - \nabla f_i(\mathbf{x})\|^2] \leq \sigma^2, \text{ FOR ANY } i, \mathbf{x}.$$

NOTE THAT (A4) ONLY BOUNDS THE VARIANCE WITHIN THE SAME CLIENT, BUT NOT THE VARIANCE ACROSS THE CLIENTS.

(A5) $\{f_i\}$ ARE β -SMOOTH AND SATISFY:

$$\|\nabla f_i(\mathbf{x}) - \nabla f_i(\mathbf{y})\| \leq \beta \|\mathbf{x} - \mathbf{y}\|, \text{ FOR ANY } i, \mathbf{x}, \mathbf{y}. \quad (11.1)$$

THE ASSUMPTION (A5) ALSO IMPLIES THE FOLLOWING QUADRATIC UPPER BOUND ON f_i

$$f_i(\mathbf{y}) \leq f_i(\mathbf{x}) + \langle \nabla f_i(\mathbf{x}), \mathbf{y} - \mathbf{x} \rangle + \frac{\beta}{2} \|\mathbf{y} - \mathbf{x}\|^2. \quad (11.2)$$

IF ADDITIONALLY THE FUNCTION $\{f_i\}$ ARE CONVEX AND \mathbf{x}^* IS AN OPTIMUM OF f , (A5) IMPLIES (VIA NESTEROV (2018), THEOREM 2.1.5)

$$\frac{1}{2\beta N} \sum_{i=1}^N \|\nabla f_i(\mathbf{x}) - \nabla f_i(\mathbf{x}^*)\|^2 \leq f(\mathbf{x}) - f^*. \quad (11.3)$$

FURTHER, IF f_i IS TWICE-DIFFERENTIABLE, (A5) IMPLIES THAT $\|\nabla^2 f_i(\mathbf{x})\| \leq \beta$ FOR ANY \mathbf{x} .

11.2.2 Some technical lemmas

NOW WE COVER SOME TECHNICAL LEMMAS WHICH ARE USEFUL FOR COMPUTATIONS LATER ON. THE TWO LEMMAS BELOW ARE USEFUL TO UNROLL RECURSIONS AND DERIVE CONVERGENCE RATES. THE FIRST ONE IS A SLIGHTLY IMPROVED (AND SIMPLIFIED) VERSION OF (STICH, 2019B, THEOREM 2). IT IS STRAIGHTFORWARD TO REMOVE THE ADDITIONAL LOGARITHMIC TERMS IF WE USE A VARYING STEP-SIZE (KULUNCHAKOV AND MAIRAL, 2019, LEMMA 13).

Lemma 30 (linear convergence rate). *For every non-negative sequence $\{d_{r-1}\}_{r \geq 1}$ and any parameters $\mu > 0$, $\eta_{\max} \in (0, 1/\mu]$, $c \geq 0$, $R \geq \frac{1}{2\eta_{\max}\mu}$, there exists a constant step-size $\eta \leq \eta_{\max}$ and weights $w_r := (1 - \mu\eta)^{1-r}$ such that for $W_R := \sum_{r=1}^{R+1} w_r$,*

$$\Psi_R := \frac{1}{W_R} \sum_{r=1}^{R+1} \left(\frac{w_r}{\eta} (1 - \mu\eta) d_{r-1} - \frac{w_r}{\eta} d_r + c\eta w_r \right) = \tilde{\mathcal{O}} \left(\mu d_0 \exp(-\mu\eta_{\max} R) + \frac{c}{\mu R} \right).$$

Proof. By substituting the value of w_r , we observe that we end up with a telescoping sum and

estimate

$$\Psi_R = \frac{1}{\eta W_R} \sum_{r=1}^{R+1} (w_{r-1} d_{r-1} - w_r d_r) + \frac{c\eta}{W_R} \sum_{r=1}^{R+1} w_r \leq \frac{d_0}{\eta W_R} + c\eta.$$

When $R \geq \frac{1}{2\mu\eta}$, $(1 - \mu\eta)^R \leq \exp(-\mu\eta R) \leq \frac{2}{3}$. For such an R , we can lower bound ηW_R using

$$\eta W_R = \eta(1 - \mu\eta)^{-R} \sum_{r=0}^R (1 - \mu\eta)^r = \eta(1 - \mu\eta)^{-R} \frac{1 - (1 - \mu\eta)^{R+1}}{\mu\eta} \geq (1 - \mu\eta)^{-R} \frac{1}{3\mu}.$$

This proves that for all $R \geq \frac{1}{2\mu\eta}$,

$$\Psi_R \leq 3\mu d_0 (1 - \mu\eta)^R + c\eta \leq 3\mu d_0 \exp(-\mu\eta R) + c\eta.$$

The lemma now follows by carefully tuning η . Consider the following two cases depending on the magnitude of R and η_{\max} :

- Suppose $\frac{1}{2\mu R} \leq \eta_{\max} \leq \frac{\log(\max(1, \mu^2 R d_0 / c))}{\mu R}$. Then we can choose $\eta = \eta_{\max}$,

$$\Psi_R \leq 3\mu d_0 \exp[-\mu\eta_{\max} R] + c\eta_{\max} \leq 3\mu d_0 \exp[-\mu\eta_{\max} R] + \tilde{\mathcal{O}}\left(\frac{c}{\mu R}\right).$$

- Instead if $\eta_{\max} > \frac{\log(\max(1, \mu^2 R d_0 / c))}{\mu R}$, we pick $\eta = \frac{\log(\max(1, \mu^2 R d_0 / c))}{\mu R}$ to claim that

$$\Psi_R \leq 3\mu d_0 \exp[-\log(\max(1, \mu^2 R d_0 / c))] + \tilde{\mathcal{O}}\left(\frac{c}{\mu R}\right) \leq \tilde{\mathcal{O}}\left(\frac{c}{\mu R}\right).$$

□

THE NEXT LEMMA IS AN EXTENSION OF (STICH AND KARIMIREDDY, 2019, LEMMA 13), (KULUNCHAKOV AND MAIRAL, 2019, LEMMA 13) AND IS USEFUL TO DERIVE CONVERGENCE RATES FOR GENERAL CONVEX FUNCTIONS ($\mu = 0$) AND NON-CONVEX FUNCTIONS.

Lemma 31 (sub-linear convergence rate). *For every non-negative sequence $\{d_{r-1}\}_{r \geq 1}$ and any parameters $\eta_{\max} \geq 0$, $c \geq 0$, $R \geq 0$, there exists a constant step-size $\eta \leq \eta_{\max}$ and weights $w_r = 1$ such that,*

$$\Psi_R := \frac{1}{R+1} \sum_{r=1}^{R+1} \left(\frac{d_{r-1}}{\eta} - \frac{d_r}{\eta} + c_1 \eta + c_2 \eta^2 \right) \leq \frac{d_0}{\eta_{\max}(R+1)} + \frac{2\sqrt{c_1 d_0}}{\sqrt{R+1}} + 2 \left(\frac{d_0}{R+1} \right)^{\frac{2}{3}} c_2^{\frac{1}{3}}.$$

Proof. Unrolling the sum, we can simplify

$$\Psi_R \leq \frac{d_0}{\eta(R+1)} + c_1 \eta + c_2 \eta^2.$$

Similar to the strongly convex case (Lemma 55), we distinguish the following cases:

- When $R+1 \leq \frac{d_0}{c_1 \eta_{\max}^2}$, and $R+1 \leq \frac{d_0}{c_2 \eta_{\max}^3}$ we pick $\eta = \eta_{\max}$ to claim

$$\Psi_R \leq \frac{d_0}{\eta_{\max}(R+1)} + c_1 \eta_{\max} + c_2 \eta_{\max}^2 \leq \frac{d_0}{\eta_{\max}(R+1)} + \frac{\sqrt{c_1 d_0}}{\sqrt{R+1}} + \left(\frac{d_0}{R+1} \right)^{\frac{2}{3}} c_2^{\frac{1}{3}}.$$

- In the other case, we have $\eta_{\max}^2 \geq \frac{d_0}{c_1(R+1)}$ or $\eta_{\max}^3 \geq \frac{d_0}{c_2(R+1)}$. We choose $\eta = \min \left\{ \sqrt{\frac{d_0}{c_1(R+1)}}, \sqrt[3]{\frac{d_0}{c_2(R+1)}} \right\}$ to prove

$$\Psi_R \leq \frac{d_0}{\eta(R+1)} + c\eta = \frac{2\sqrt{c_1 d_0}}{\sqrt{R+1}} + 2\sqrt[3]{\frac{d_0^2 c_2}{(R+1)^2}}.$$

□

NEXT, WE STATE A RELAXED TRIANGLE INEQUALITY TRUE FOR THE SQUARED ℓ_2 NORM.

Lemma 32 (relaxed triangle inequality). *Let $\{\mathbf{v}_1, \dots, \mathbf{v}_\tau\}$ be τ vectors in \mathbb{R}^d . Then the following are true:*

1. $\|\mathbf{v}_i + \mathbf{v}_j\|^2 \leq (1+a)\|\mathbf{v}_i\|^2 + (1+\frac{1}{a})\|\mathbf{v}_j\|^2$ for any $a > 0$, and
2. $\|\sum_{i=1}^\tau \mathbf{v}_i\|^2 \leq \tau \sum_{i=1}^\tau \|\mathbf{v}_i\|^2$.

Proof. The proof of the first statement for any $a > 0$ follows from the identity:

$$\|\mathbf{v}_i + \mathbf{v}_j\|^2 = (1+a)\|\mathbf{v}_i\|^2 + (1+\frac{1}{a})\|\mathbf{v}_j\|^2 - \|\sqrt{a}\mathbf{v}_i + \frac{1}{\sqrt{a}}\mathbf{v}_j\|^2.$$

For the second inequality, we use the convexity of $\mathbf{x} \rightarrow \|\mathbf{x}\|^2$ and Jensen's inequality

$$\left\| \frac{1}{\tau} \sum_{i=1}^\tau \mathbf{v}_i \right\|^2 \leq \frac{1}{\tau} \sum_{i=1}^\tau \|\mathbf{v}_i\|^2.$$

□

NEXT WE STATE AN ELEMENTARY LEMMA ABOUT EXPECTATIONS OF NORMS OF RANDOM VECTORS.

Lemma 33 (separating mean and variance). *Let $\{\Xi_1, \dots, \Xi_\tau\}$ be τ random variables in \mathbb{R}^d which are not necessarily independent. First suppose that their mean is $\mathbb{E}[\Xi_i] = \xi_i$ and variance is bounded as $\mathbb{E}[\|\Xi_i - \xi_i\|^2] \leq \sigma^2$. Then, the following holds*

$$\mathbb{E}[\|\sum_{i=1}^\tau \Xi_i\|^2] \leq \|\sum_{i=1}^\tau \xi_i\|^2 + \tau^2 \sigma^2.$$

Now instead suppose that their conditional mean is $\mathbb{E}[\Xi_i | \Xi_{i-1}, \dots, \Xi_1] = \xi_i$ i.e. the variables $\{\Xi_i - \xi_i\}$ form a martingale difference sequence, and the variance is bounded by $\mathbb{E}[\|\Xi_i - \xi_i\|^2] \leq$

σ^2 as before. Then we can show the tighter bound

$$\mathbb{E}[\|\sum_{i=1}^{\tau} \Xi_i\|^2] \leq 2\|\sum_{i=1}^{\tau} \xi_i\|^2 + 2\tau\sigma^2.$$

Proof. For any random variable X , $\mathbb{E}[X^2] = (\mathbb{E}[X - \mathbb{E}[X]])^2 + (\mathbb{E}[X])^2$ implying

$$\mathbb{E}[\|\sum_{i=1}^{\tau} \Xi_i\|^2] = \|\sum_{i=1}^{\tau} \xi_i\|^2 + \mathbb{E}[\|\sum_{i=1}^{\tau} \Xi_i - \xi_i\|^2].$$

Expanding the above expression using relaxed triangle inequality (Lemma 57) proves the first claim:

$$\mathbb{E}[\|\sum_{i=1}^{\tau} \Xi_i - \xi_i\|^2] \leq \tau \sum_{i=1}^{\tau} \mathbb{E}[\|\Xi_i - \xi_i\|^2] \leq \tau^2 \sigma^2.$$

For the second statement, ξ_i is not deterministic and depends on Ξ_{i-1}, \dots, Ξ_1 . Hence we have to resort to the cruder relaxed triangle inequality to claim

$$\mathbb{E}[\|\sum_{i=1}^{\tau} \Xi_i\|^2] \leq 2\|\sum_{i=1}^{\tau} \xi_i\|^2 + 2\mathbb{E}[\|\sum_{i=1}^{\tau} \Xi_i - \xi_i\|^2]$$

and then use the tighter expansion of the second term:

$$\mathbb{E}[\|\sum_{i=1}^{\tau} \Xi_i - \xi_i\|^2] = \sum_{i,j} \mathbb{E}[(\Xi_i - \xi_i)^\top (\Xi_j - \xi_j)] = \sum_i \mathbb{E}[\|\Xi_i - \xi_i\|^2] \leq \tau\sigma^2.$$

The cross terms in the above expression have zero mean since $\{\Xi_i - \xi_i\}$ form a martingale difference sequence. \square

11.3 Properties of convex functions

WE NOW STUDY TWO LEMMAS WHICH HOLD FOR ANY SMOOTH AND STRONGLY-CONVEX FUNCTIONS. THE FIRST IS A GENERALIZATION OF THE STANDARD STRONG CONVEXITY INEQUALITY (A3), BUT CAN HANDLE GRADIENTS COMPUTED AT SLIGHTLY PERTURBED POINTS.

Lemma 34 (perturbed strong convexity). *The following holds for any β -smooth and μ -strongly convex function h , and any $\mathbf{x}, \mathbf{y}, \mathbf{z}$ in the domain of h :*

$$\langle \nabla h(\mathbf{x}), \mathbf{z} - \mathbf{y} \rangle \geq h(\mathbf{z}) - h(\mathbf{y}) + \frac{\mu}{4} \|\mathbf{y} - \mathbf{z}\|^2 - \beta \|\mathbf{z} - \mathbf{x}\|^2.$$

Proof. Given any \mathbf{x}, \mathbf{y} , and \mathbf{z} , we get the following two inequalities using smoothness and

strong convexity of h :

$$\begin{aligned}\langle \nabla h(\mathbf{x}), \mathbf{z} - \mathbf{x} \rangle &\geq h(\mathbf{z}) - h(\mathbf{x}) - \frac{\beta}{2} \|\mathbf{z} - \mathbf{x}\|^2 \\ \langle \nabla h(\mathbf{x}), \mathbf{x} - \mathbf{y} \rangle &\geq h(\mathbf{x}) - h(\mathbf{y}) + \frac{\mu}{2} \|\mathbf{y} - \mathbf{x}\|^2.\end{aligned}$$

Further, applying the relaxed triangle inequality gives

$$\frac{\mu}{2} \|\mathbf{y} - \mathbf{x}\|^2 \geq \frac{\mu}{4} \|\mathbf{y} - \mathbf{z}\|^2 - \frac{\mu}{2} \|\mathbf{x} - \mathbf{z}\|^2.$$

Combining all the inequalities together we have

$$\langle \nabla h(\mathbf{x}), \mathbf{z} - \mathbf{y} \rangle \geq h(\mathbf{z}) - h(\mathbf{y}) + \frac{\mu}{4} \|\mathbf{y} - \mathbf{z}\|^2 - \frac{\beta + \mu}{2} \|\mathbf{z} - \mathbf{x}\|^2.$$

The lemma follows since $\beta \geq \mu$. □

HERE, WE SEE THAT A GRADIENT STEP IS A CONTRACTIVE OPERATOR.

Lemma 35 (contractive mapping). *For any β -smooth and μ -strongly convex function h , points \mathbf{x}, \mathbf{y} in the domain of h , and step-size $\eta \leq \frac{1}{\beta}$, the following is true*

$$\|\mathbf{x} - \eta \nabla h(\mathbf{x}) - \mathbf{y} + \eta \nabla h(\mathbf{y})\|^2 \leq (1 - \mu\eta) \|\mathbf{x} - \mathbf{y}\|^2.$$

Proof.

$$\begin{aligned}\|\mathbf{x} - \eta \nabla h(\mathbf{x}) - \mathbf{y} + \eta \nabla h(\mathbf{y})\|^2 &= \|\mathbf{x} - \mathbf{y}\|^2 + \eta^2 \|\nabla h(\mathbf{x}) - \nabla h(\mathbf{y})\|^2 - 2\eta \langle \nabla h(\mathbf{x}) - \nabla h(\mathbf{y}), \mathbf{x} - \mathbf{y} \rangle \\ &\stackrel{(A5)}{\leq} \|\mathbf{x} - \mathbf{y}\|^2 + (\eta^2 \beta - 2\eta) \langle \nabla h(\mathbf{x}) - \nabla h(\mathbf{y}), \mathbf{x} - \mathbf{y} \rangle.\end{aligned}$$

Recall our bound on the step-size $\eta \leq \frac{1}{\beta}$ which implies that $(\eta^2 \beta - 2\eta) \leq -\eta$. Finally, apply the μ -strong convexity of h to get

$$-\eta \langle \nabla h(\mathbf{x}) - \nabla h(\mathbf{y}), \mathbf{x} - \mathbf{y} \rangle \leq -\eta\mu \|\mathbf{x} - \mathbf{y}\|^2.$$

□

11.4 Convergence of FEDAVG

WE OUTLINE THE FEDAVG METHOD IN ALGORITHM 18. IN ROUND r WE SAMPLE $\mathcal{S}^r \subseteq [N]$ CLIENTS WITH $|\mathcal{S}^r| = S$ AND THEN PERFORM THE FOLLOWING UPDATES:

- STARTING FROM THE SHARED GLOBAL PARAMETERS $\mathbf{y}_{i,0}^r = \mathbf{x}^{r-1}$, WE UPDATE THE LOCAL PARAMETERS FOR $k \in [K]$

$$\mathbf{y}_{i,k}^r = \mathbf{y}_{i,k-1}^r - \eta_l g_i(\mathbf{y}_{i,k-1}^r). \quad (11.4)$$

Algorithm 17 FEDAVG: Federated Averaging

```

1: server input: initial  $\mathbf{x}$ , and global step-size  $\eta_g$ 
2: client  $i$ 's input: local step-size  $\eta_l$ 
3: for each round  $r = 1, \dots, R$  do
4:   sample clients  $\mathcal{S} \subseteq \{1, \dots, N\}$ 
5:   communicate  $\mathbf{x}$  to all clients  $i \in \mathcal{S}$ 
6:   for each client  $i \in \mathcal{S}$  in parallel do
7:     initialize local model  $\mathbf{y}_i \leftarrow \mathbf{x}$ 
8:     for  $k = 1, \dots, K$  do
9:       compute mini-batch gradient  $\mathbf{g}_i(\mathbf{y}_i)$ 
10:       $\mathbf{y}_i \leftarrow \mathbf{y}_i - \eta_l \mathbf{g}_i(\mathbf{y}_i)$ 
11:    end for
12:    communicate  $\Delta \mathbf{y}_i \leftarrow \mathbf{y}_i - \mathbf{x}$ 
13:  end for
14:   $\Delta \mathbf{x} \leftarrow \frac{1}{|\mathcal{S}|} \sum_{i \in \mathcal{S}} \Delta \mathbf{y}_i$ 
15:   $\mathbf{x} \leftarrow \mathbf{x} + \eta_g \Delta \mathbf{x}$ 
16: end for

```

- COMPUTE THE NEW GLOBAL PARAMETERS USING ONLY UPDATES FROM THE CLIENTS $i \in \mathcal{S}^r$ AND A GLOBAL STEP-SIZE η_g :

$$\mathbf{x}^r = \mathbf{x}^{r-1} + \frac{\eta_g}{S} \sum_{i \in \mathcal{S}^r} (\mathbf{y}_{i,K}^r - \mathbf{x}^{r-1}). \quad (11.5)$$

FINALLY, FOR SOME WEIGHTS $\{w_r\}$, WE OUTPUT

$$\bar{\mathbf{x}}^R = \mathbf{x}^{r-1} \text{ WITH PROBABILITY } \frac{w_r}{\sum_{\tau} w_{\tau}} \text{ FOR } r \in \{1, \dots, R+1\}. \quad (11.6)$$

11.4.1 Bounding heterogeneity

RECALL OUR BOUND ON THE GRADIENT DISSIMILARITY:

$$\frac{1}{N} \sum_{i=1}^N \|\nabla f_i(\mathbf{x})\|^2 \leq G^2 + B^2 \|\nabla f(\mathbf{x})\|^2. \quad (11.7)$$

IF $\{f_i\}$ ARE CONVEX, WE CAN RELAX THE ASSUMPTION TO

$$\frac{1}{N} \sum_{i=1}^N \|\nabla f_i(\mathbf{x})\|^2 \leq G^2 + 2\beta B^2 (f(\mathbf{x}) - f^*). \quad (11.8)$$

WE DEFINED TWO VARIANTS OF THE BOUNDS ON THE HETEROGENEITY DEPENDING OF WHETHER THE FUNCTIONS ARE CONVEX OR NOT. SUPPOSE THAT THE FUNCTIONS f IS INDEED CONVEX AS IN (A3) AND β -SMOOTH AS IN (A5), THEN IT IS STRAIGHTFORWARD TO SEE THAT (12.7) IMPLIES (12.8). THUS FOR CONVEX FUNCTIONS, (A1) IS MILDLY WEAKER. SUPPOSE THAT THE FUNC-

CTIONS $\{f_1, \dots, f_N\}$ ARE CONVEX AND β -SMOOTH. THEN (12.8) IS SATISFIED WITH $B^2 = 2$ SINCE

$$\begin{aligned} \frac{1}{N} \sum_{i=1}^N \|\nabla f_i(\mathbf{x})\|^2 &\leq \frac{2}{N} \sum_{i=1}^N \|\nabla f_i(\mathbf{x}^\star)\|^2 + \frac{2}{N} \sum_{i=1}^N \|\nabla f_i(\mathbf{x}) - \nabla f_i(\mathbf{x}^\star)\|^2 \\ &\stackrel{(12.3)}{\leq} \underbrace{\frac{2}{N} \sum_{i=1}^N \|\nabla f_i(\mathbf{x}^\star)\|^2}_{=: \sigma_f^2} + 4\beta(f(\mathbf{x}) - f^\star). \end{aligned}$$

THUS, (G, B) -BGD (12.8) IS EQUIVALENT TO THE HETEROGENEITY ASSUMPTION OF (MISHCHENKO ET AL., 2019) WITH $G^2 = \sigma_f^2$. INSTEAD, IF WE HAVE THE STRONGER ASSUMPTION (A1) BUT THE FUNCTIONS ARE POSSIBLY NON-CONVEX, THEN $G = \epsilon$ CORRESPONDS TO THE **LOCAL DISSIMILARITY** DEFINED IN (LI ET AL., 2018B). NOTE THAT ASSUMING G IS NEGLIGIBLE IS QUITE STRONG AND CORRESPONDS TO THE STRONG-GROWTH CONDITION (VASWANI ET AL., 2019).

11.4.2 Rates of convergence (Theorem VI)

WE FIRST RESTATE THEOREM VI WITH SOME ADDITIONAL DETAILS AND THEN SEE ITS PROOF.

Theorem XXVI. *Suppose that the functions $\{f_i\}$ satisfies assumptions A4, A5, and A1. Then, in each of the following cases, there exist weights $\{w_r\}$ and local step-sizes η_l such that for any $\eta_g \geq 1$ the output of FEDAVG (12.6) $\bar{\mathbf{x}}^R$ satisfies*

- **Strongly convex:** f_i satisfies (A3) for $\mu > 0$, $\eta_l \leq \frac{1}{8(1+B^2)\beta K \eta_g}$, $R \geq \frac{8(1+B^2)\beta}{\mu}$ then

$$\mathbb{E}[f(\bar{\mathbf{x}}^R)] - f(\mathbf{x}^\star) \leq \tilde{\mathcal{O}}\left(\frac{M^2}{\mu R K S} + \frac{\beta G^2}{\mu^2 R^2} + \mu D^2 \exp\left(-\frac{\mu}{16(1+B^2)\beta} R\right)\right),$$

- **General convex:** f_i satisfies (A3) for $\mu = 0$, $\eta_l \leq \frac{1}{(1+B^2)8\beta K \eta_g}$, $R \geq 1$ then

$$\mathbb{E}[f(\bar{\mathbf{x}}^R)] - f(\mathbf{x}^\star) \leq \mathcal{O}\left(\frac{MD}{\sqrt{R K S}} + \frac{D^{4/3}(\beta G^2)^{1/3}}{(R+1)^{2/3}} + \frac{B^2 \beta D^2}{R}\right),$$

- **Non-convex:** f_i satisfies (A1) and $\eta_l \leq \frac{1}{(1+B^2)8\beta K \eta_g}$, then

$$\mathbb{E}[\|\nabla f(\bar{\mathbf{x}}^R)\|^2] \leq \mathcal{O}\left(\frac{\beta M \sqrt{F}}{\sqrt{R K S}} + \frac{F^{2/3}(\beta G^2)^{1/3}}{(R+1)^{2/3}} + \frac{B^2 \beta F}{R}\right),$$

where $M^2 := \sigma^2(1 + \frac{S}{\eta_g^2}) + K(1 - \frac{S}{N})G^2$, $D := \|\mathbf{x}^0 - \mathbf{x}^\star\|$, and $F := f(\mathbf{x}^0) - f^\star$.

11.4.3 Proof of convergence

WE WILL ONLY PROVE THE RATE OF CONVERGENCE FOR CONVEX FUNCTIONS HERE. THE CORRESPONDING RATES FOR NON-CONVEX FUNCTIONS ARE EASY TO DERIVE FOLLOWING THE TECHNIQUES IN THE REST OF THE PAPER.

Lemma 36. (one round progress) Suppose our functions satisfies assumptions (A1) and (A3)–(A5). For any step-size satisfying $\eta_l \leq \frac{1}{(1+B^2)8\beta K\eta_g}$ and effective step-size $\tilde{\eta} := K\eta_g\eta_l$, the updates of FEDAVG satisfy

$$\mathbb{E}\|\mathbf{x}^r - \mathbf{x}^\star\|^2 \leq (1 - \frac{\mu\tilde{\eta}}{2})\mathbb{E}\|\mathbf{x}^{r-1} - \mathbf{x}^\star\|^2 + (\frac{1}{KS})\tilde{\eta}^2\sigma^2 + (1 - \frac{S}{N})\frac{4\tilde{\eta}^2}{S}G^2 - \tilde{\eta}(\mathbb{E}[f(\mathbf{x}^{r-1})] - f(\mathbf{x}^\star)) + 3\beta\tilde{\eta}\mathcal{E}_r,$$

where \mathcal{E}_r is the drift caused by the local updates on the clients defined to be

$$\mathcal{E}_r := \frac{1}{KN} \sum_{k=1}^K \sum_{i=1}^N \mathbb{E}_r[\|\mathbf{y}_{i,k}^r - \mathbf{x}^{r-1}\|^2].$$

Proof. We start with the observation that the updates (12.4) and (12.5) imply that the server update in round r can be written as below (dropping the superscripts everywhere)

$$\Delta\mathbf{x} = -\frac{\tilde{\eta}}{KS} \sum_{k,i \in \mathcal{S}} g_i(\mathbf{y}_{i,k-1}) \text{ and } \mathbb{E}[\Delta\mathbf{x}] = -\frac{\tilde{\eta}}{KN} \sum_{k,i} \mathbb{E}[\nabla f_i(\mathbf{y}_{i,k-1})].$$

We adopt the convention that summations are always over $k \in [K]$ or $i \in [N]$ unless otherwise stated. Expanding using above observing, we proceed as¹

$$\begin{aligned} \mathbb{E}_{r-1}\|\mathbf{x} + \Delta\mathbf{x} - \mathbf{x}^\star\|^2 &= \|\mathbf{x} - \mathbf{x}^\star\|^2 - \frac{2\tilde{\eta}}{KN} \sum_{k,i} \langle \nabla f_i(\mathbf{y}_{i,k-1}), \mathbf{x} - \mathbf{x}^\star \rangle + \tilde{\eta}^2 \mathbb{E}_{r-1} \left\| \frac{1}{KS} \sum_{k,i \in \mathcal{S}} g_i(\mathbf{y}_{i,k-1}) \right\|^2 \\ &\stackrel{\text{Lem. 58}}{\leq} \underbrace{\|\mathbf{x} - \mathbf{x}^\star\|^2 - \frac{2\tilde{\eta}}{KN} \sum_{k,i} \langle \nabla f_i(\mathbf{y}_{i,k-1}), \mathbf{x} - \mathbf{x}^\star \rangle}_{\mathcal{A}_1} \\ &\quad + \underbrace{\tilde{\eta}^2 \mathbb{E}_{r-1} \left\| \frac{1}{KS} \sum_{k,i \in \mathcal{S}} \nabla f_i(\mathbf{y}_{i,k-1}) \right\|^2}_{\mathcal{A}_2} + \frac{\tilde{\eta}^2\sigma^2}{KS}. \end{aligned}$$

We can directly apply Lemma 59 with $h = f_i$, $\mathbf{x} = \mathbf{y}_{i,k-1}$, $\mathbf{y} = \mathbf{x}^\star$, and $\mathbf{z} = \mathbf{x}$ to the first term \mathcal{A}_1

$$\begin{aligned} \mathcal{A}_1 &= \frac{2\tilde{\eta}}{KN} \sum_{k,i} \langle \nabla f_i(\mathbf{y}_{i,k-1}), \mathbf{x}^\star - \mathbf{x} \rangle \\ &\leq \frac{2\tilde{\eta}}{KN} \sum_{k,i} \left(f_i(\mathbf{x}^\star) - f_i(\mathbf{x}) + \beta\|\mathbf{y}_{i,k-1} - \mathbf{x}\|^2 - \frac{\mu}{4}\|\mathbf{x} - \mathbf{x}^\star\|^2 \right) \\ &= -2\tilde{\eta} \left(f(\mathbf{x}) - f(\mathbf{x}^\star) + \frac{\mu}{4}\|\mathbf{x} - \mathbf{x}^\star\|^2 \right) + 2\beta\tilde{\eta}\mathcal{E}. \end{aligned}$$

¹We use the notation $\mathbb{E}_{r-1}[\cdot]$ to mean conditioned on filtration r i.e. on all the randomness generated prior to round r .

For the second term \mathcal{A}_2 , we repeatedly apply the relaxed triangle inequality (Lemma 58)

$$\begin{aligned}
 \mathcal{A}_2 &= \tilde{\eta}^2 \mathbb{E}_{r-1} \left\| \frac{1}{KS} \sum_{k,i \in \mathcal{S}} \nabla f_i(\mathbf{y}_{i,k-1}) - \nabla f_i(\mathbf{x}) + \nabla f_i(\mathbf{x}) \right\|^2 \\
 &\leq 2\tilde{\eta}^2 \mathbb{E}_{r-1} \left\| \frac{1}{KS} \sum_{k,i \in \mathcal{S}} \nabla f_i(\mathbf{y}_{i,k-1}) - \nabla f_i(\mathbf{x}) \right\|^2 + 2\tilde{\eta}^2 \mathbb{E}_{r-1} \left\| \frac{1}{S} \sum_{i \in \mathcal{S}} \nabla f_i(\mathbf{x}) \right\|^2 \\
 &\leq \frac{2\tilde{\eta}^2}{KN} \sum_{i,k} \mathbb{E}_{r-1} \|\nabla f_i(\mathbf{y}_{i,k-1}) - \nabla f_i(\mathbf{x})\|^2 + 2\tilde{\eta}^2 \mathbb{E}_{r-1} \left\| \frac{1}{S} \sum_{i \in \mathcal{S}} \nabla f_i(\mathbf{x}) - \nabla f(\mathbf{x}) + \nabla f(\mathbf{x}) \right\|^2 \\
 &\leq \frac{2\tilde{\eta}^2 \beta^2}{KN} \sum_{i,k} \mathbb{E}_{r-1} \|\mathbf{y}_{i,k-1} - \mathbf{x}\|^2 + 2\tilde{\eta}^2 \|\nabla f(\mathbf{x})\|^2 + (1 - \frac{S}{N}) 4\tilde{\eta}^2 \frac{1}{SN} \sum_i \|\nabla f_i(\mathbf{x})\|^2 \\
 &\leq 2\tilde{\eta}^2 \beta^2 \mathcal{E} + 8\tilde{\eta}^2 \beta(B^2 + 1)(f(\mathbf{x}) - f(\mathbf{x}^*)) + (1 - \frac{S}{N}) \frac{4\tilde{\eta}^2}{S} G^2
 \end{aligned}$$

The last step used Assumption (G, B) -BGD assumption (12.8) that $\frac{1}{N} \sum_{i=1}^N \|\nabla f_i(\mathbf{x})\|^2 \leq G^2 + 2\beta B^2(f(\mathbf{x}) - f(\mathbf{x}^*))$. The extra $(1 - \frac{S}{N})$ improvement we get is due to sampling the functions $\{f_i\}$ *without* replacement. Plugging back the bounds on \mathcal{A}_1 and \mathcal{A}_2 ,

$$\begin{aligned}
 \mathbb{E}_{r-1} \|\mathbf{x} + \Delta \mathbf{x} - \mathbf{x}^*\|^2 &\leq (1 - \frac{\mu\tilde{\eta}}{2}) \|\mathbf{x} - \mathbf{x}^*\|^2 - (2\tilde{\eta} - 8\beta\tilde{\eta}^2(B^2 + 1))(f(\mathbf{x}) - f(\mathbf{x}^*)) \\
 &\quad + (1 + \tilde{\eta}\beta) 2\beta\tilde{\eta}\mathcal{E} + \frac{1}{KS} \tilde{\eta}^2 \sigma^2 + (1 - \frac{S}{N}) \frac{4\tilde{\eta}^2}{S} G^2.
 \end{aligned}$$

The lemma now follows by observing that $8\beta\tilde{\eta}(B^2 + 1) \leq 1$ and that $B \geq 0$. \square

Lemma 37. (bounded drift) Suppose our functions satisfies assumptions (A1) and (A3)–(A5). Then the updates of FEDAVG for any step-size satisfying $\eta_l \leq \frac{1}{(1+B^2)8\beta K\eta_g}$ have bounded drift:

$$3\beta\tilde{\eta}\mathcal{E}_r \leq \frac{2\tilde{\eta}}{3} (\mathbb{E}[f(\mathbf{x}^{r-1})]) - f(\mathbf{x}^*) + \frac{\tilde{\eta}^2 \sigma^2}{2K\eta_g^2} + 18\beta\tilde{\eta}^3 G^2.$$

Proof. If $K = 1$, the lemma trivially holds since $\mathbf{y}_{i,0} = \mathbf{x}$ for all $i \in [N]$ and $\mathcal{E}_r = 0$. Assume $K \geq 2$ here on. Recall that the local update made on client i is $\mathbf{y}_{i,k} = \mathbf{y}_{i,k-1} - \eta_l g_i(\mathbf{y}_{i,k-1})$.

Then,

$$\begin{aligned}
 \mathbb{E}\|\mathbf{y}_{i,k} - \mathbf{x}\|^2 &= \mathbb{E}\|\mathbf{y}_{i,k-1} - \mathbf{x} - \eta_l g_i(\mathbf{y}_{i,k-1})\|^2 \\
 &\leq \mathbb{E}\|\mathbf{y}_{i,k-1} - \mathbf{x} - \eta_l \nabla f_i(\mathbf{y}_{i,k-1})\|^2 + \eta_l^2 \sigma^2 \\
 &\leq (1 + \frac{1}{K-1}) \mathbb{E}\|\mathbf{y}_{i,k-1} - \mathbf{x}\|^2 + K\eta_l^2 \|\nabla f_i(\mathbf{y}_{i,k-1})\|^2 + \eta_l^2 \sigma^2 \\
 &= (1 + \frac{1}{K-1}) \mathbb{E}\|\mathbf{y}_{i,k-1} - \mathbf{x}\|^2 + \frac{\tilde{\eta}^2}{\eta_g K} \|\nabla f_i(\mathbf{y}_{i,k-1})\|^2 + \frac{\tilde{\eta}^2 \sigma^2}{K^2 \eta_g^2} \\
 &\leq (1 + \frac{1}{K-1}) \mathbb{E}\|\mathbf{y}_{i,k-1} - \mathbf{x}\|^2 + \frac{2\tilde{\eta}^2}{\eta_g K} \|\nabla f_i(\mathbf{y}_{i,k-1}) - \nabla f_i(\mathbf{x})\|^2 + \frac{2\tilde{\eta}^2}{\eta_g K} \|\nabla f_i(\mathbf{x})\|^2 + \frac{\tilde{\eta}^2 \sigma^2}{K^2 \eta_g^2} \\
 &\leq (1 + \frac{1}{K-1} + \frac{2\tilde{\eta}^2 \beta^2}{\eta_g K}) \mathbb{E}\|\mathbf{y}_{i,k-1} - \mathbf{x}\|^2 + \frac{2\tilde{\eta}^2}{\eta_g K} \|\nabla f_i(\mathbf{x})\|^2 + \frac{\tilde{\eta}^2 \sigma^2}{K^2 \eta_g^2} \\
 &\leq (1 + \frac{2}{(K-1)}) \mathbb{E}\|\mathbf{y}_{i,k-1} - \mathbf{x}\|^2 + \frac{2\tilde{\eta}^2}{\eta_g K} \|\nabla f_i(\mathbf{x})\|^2 + \frac{\tilde{\eta}^2 \sigma^2}{K^2 \eta_g^2}.
 \end{aligned}$$

In the above proof we separated the mean and the variance in the first inequality, then used the relaxed triangle inequality with $a = \frac{1}{K-1}$ in the next inequality. Next equality uses the definition of $\tilde{\eta}$, and the rest follow from the Lipschitzness of the gradient. Unrolling the recursion above,

$$\begin{aligned}
 \mathbb{E}\|\mathbf{y}_{i,k} - \mathbf{x}\|^2 &\leq \sum_{\tau=1}^{k-1} (\frac{2\tilde{\eta}^2}{\eta_g K} \|\nabla f_i(\mathbf{x})\|^2 + \frac{\tilde{\eta}^2 \sigma^2}{K^2 \eta_g^2}) (1 + \frac{2}{(K-1)})^\tau \\
 &\leq (\frac{2\tilde{\eta}^2}{\eta_g K} \|\nabla f_i(\mathbf{x})\|^2 + \frac{\tilde{\eta}^2 \sigma^2}{K^2 \eta_g^2}) 3K.
 \end{aligned}$$

Averaging over i and k , multiplying by $3\beta\tilde{\eta}$ and then using Assumption A1,

$$\begin{aligned}
 3\beta\tilde{\eta}\mathcal{E}_r &\leq \frac{1}{N} \sum_i 18\beta\tilde{\eta}^3 \|\nabla f_i(\mathbf{x})\|^2 + \frac{3\beta\tilde{\eta}^3 \sigma^2}{K\eta_g^2} \\
 &\leq 18\beta\tilde{\eta}^3 G^2 + \frac{3\beta\tilde{\eta}^3 \sigma^2}{K\eta_g^2} + 36\beta^2 \tilde{\eta}^3 B^2 (f(\mathbf{x}) - f(\mathbf{x}^*))
 \end{aligned}$$

The lemma now follows from our assumption that $8(B^2 + 1)\beta\tilde{\eta} \leq 1$. \square

PROOF OF THEOREMS VI, XXX ADDING THE STATEMENTS OF LEMMAS 61 AND 62, WE GET

$$\begin{aligned}
 \mathbb{E}\|\mathbf{x} + \Delta\mathbf{x} - \mathbf{x}^*\|^2 &\leq (1 - \frac{\mu\tilde{\eta}}{2}) \mathbb{E}\|\mathbf{x} - \mathbf{x}^*\|^2 + (\frac{1}{KS}) \tilde{\eta}^2 \sigma^2 + (1 - \frac{S}{N}) \frac{4\tilde{\eta}^2}{S} G^2 - \tilde{\eta}(\mathbb{E}[f(\mathbf{x})] - f(\mathbf{x}^*)) \\
 &\quad + \frac{2\tilde{\eta}}{3} (\mathbb{E}[f(\mathbf{x})] - f(\mathbf{x}^*)) + \frac{\tilde{\eta}^2 \sigma^2}{2K\eta_g^2} + 18\beta\tilde{\eta}^3 G^2 \\
 &= (1 - \frac{\mu\tilde{\eta}}{2}) \mathbb{E}\|\mathbf{x} - \mathbf{x}^*\|^2 - \frac{\tilde{\eta}}{3} (\mathbb{E}[f(\mathbf{x})] - f(\mathbf{x}^*)) + \tilde{\eta}^2 \left(\frac{\sigma^2}{KS} (1 + \frac{S}{\eta_g^2}) + \frac{4G^2}{S} (1 - \frac{S}{N}) + 18\beta\tilde{\eta}G^2 \right).
 \end{aligned}$$

MOVING THE $(f(\mathbf{x}) - f(\mathbf{x}^*))$ TERM AND DIVIDING THROUGHOUT BY $\frac{\tilde{\eta}}{3}$, WE GET THE FOLLOWING BOUND FOR ANY $\tilde{\eta} \leq \frac{1}{8(1+B^2)\beta}$

$$\mathbb{E}[f(\mathbf{x}^{r-1})] - f(\mathbf{x}^*) \leq \frac{3}{\tilde{\eta}}(1 - \frac{\mu\tilde{\eta}}{2})\|\mathbf{x}^{r-1} - \mathbf{x}^*\|^2 - \frac{3}{\tilde{\eta}}\|\mathbf{x}^r - \mathbf{x}^*\|^2 + 3\tilde{\eta}\left(\frac{\sigma^2}{KS}(1 + \frac{S}{\eta_g^2}) + \frac{4G^2}{S}(1 - \frac{S}{N}) + 18\beta\tilde{\eta}G^2\right).$$

IF $\mu = 0$ (GENERAL CONVEX), WE CAN DIRECTLY APPLY LEMMA 56. OTHERWISE, BY AVERAGING USING WEIGHTS $w_r = (1 - \frac{\mu\tilde{\eta}}{2})^{1-r}$ AND USING THE SAME WEIGHTS TO PICK OUTPUT $\bar{\mathbf{x}}^R$, WE CAN SIMPLIFY THE ABOVE RECURSIVE BOUND (SEE PROOF OF LEM. 55) TO PROVE THAT FOR ANY $\tilde{\eta}$ SATISFYING $\frac{1}{\mu R} \leq \tilde{\eta} \leq \frac{1}{8(1+B^2)\beta}$

$$\mathbb{E}[f(\bar{\mathbf{x}}^R)] - f(\mathbf{x}^*) \leq \underbrace{3\|\mathbf{x}^0 - \mathbf{x}^*\|^2}_{=:d} \mu \exp(-\frac{\tilde{\eta}}{2}\mu R) + \tilde{\eta} \underbrace{\left(\frac{2\sigma^2}{KS}(1 + \frac{S}{\eta_g^2}) + \frac{8G^2}{S}(1 - \frac{S}{N})\right)}_{=:c_1} + \tilde{\eta}^2 \underbrace{(36\beta G^2)}_{=:c_2}$$

NOW, THE CHOICE OF $\tilde{\eta} = \min\left\{\frac{\log(\max(1, \mu^2 R d / c_1))}{\mu R}, \frac{1}{(1+B^2)8\beta}\right\}$ YIELDS THE DESIRED RATE. THE PROOF OF THE NON-CONVEX CASE IS VERY SIMILAR AND ALSO RELIES ON LEMMA 56.

11.4.4 Lower bound for FEDAVG (Theorem VII)

WE FIRST FORMALIZE THE CLASS OF ALGORITHMS WE LOOK AT BEFORE PROVING OUT LOWER BOUND.

(A6) WE ASSUME THAT FEDAVG IS RUN WITH $\eta_g = 1$, $K > 1$, AND ARBITRARY POSSIBLY ADAPTIVE POSITIVE STEP-SIZES $\{\eta_1, \dots, \eta_R\}$ ARE USED WITH $\eta_r \leq \frac{1}{\mu}$ AND FIXED WITHIN A ROUND FOR ALL CLIENTS. FURTHER, THE SERVER UPDATE IS A CONVEX COMBINATION OF THE CLIENT UPDATES WITH NON-ADAPTIVE WEIGHTS.

NOTE THAT WE ONLY PROVE THE LOWER BOUND HERE FOR $\eta_g = 1$. IN FACT, BY TAKING η_g INFINITELY LARGE AND SCALING $\eta_l \propto \frac{1}{K\eta_g}$ SUCH THAT THE EFFECTIVE STEP SIZE $\tilde{\eta} = \eta_l \eta_g K$ REMAINS CONSTANT, FEDAVG REDUCES TO THE SIMPLE LARGE BATCH SGD METHOD. HENCE, PROVING A LOWER BOUND FOR ARBITRARY η_g IS NOT POSSIBLE, BUT ALSO IS OF QUESTIONABLE RELEVANCE. FURTHER, NOTE THAT WHEN $\sigma^2 = 0$, THE UPPER BOUND IN THEOREM XXX USES $\eta_g = 1$ AND HENCE THE LOWER BOUND SERVES TO SHOW THAT OUR ANALYSIS IS TIGHT.

BELOW WE STATE A MORE FORMAL VERSION OF THEOREM VII.

Theorem XXVII. *For any positive constants G, μ , there exist μ -strongly convex functions satisfying A1 for which that the output of FEDAVG satisfying A6 has the error for any $r \geq 1$:*

$$f(\mathbf{x}^r) - f(\mathbf{x}^*) \geq \Omega\left(\min(f(\mathbf{x}^0) - f(\mathbf{x}^*), \frac{G^2}{\mu R^2})\right).$$

Proof. Consider the following simple one-dimensional functions for any given μ and G :

$$f_1(x) := \mu x^2 + Gx, \text{ and } f_2(x) := -Gx,$$

with $f(x) = \frac{1}{2}(f_1(x) + f_2(x)) = \frac{\mu}{2}x^2$ and optimum at $x = 0$. Clearly f is μ -strongly convex and further f_1 and f_2 satisfy A1 with $B = 3$. Note that we chose f_2 to be a linear function (not strongly convex) to simplify computations. The calculations made here can be extended with slightly more work for $(\tilde{f}_2 = \frac{\mu}{2}x^2 - Gx)$ (e.g. see Theorem 1 of (Safran and Shamir, 2019)).

Let us start FEDAVG from $x^0 > 0$. A single local update for f_1 and f_2 in round $r \geq 1$ is respectively

$$y_1 = y_1 - \eta_r(2\mu x + G) \text{ and } y_2 = y_2 + \eta_r G.$$

Then, straightforward computations show that the update at the end of round r is of the following form for some averaging weight $\alpha \in [0, 1]$

$$x^r = x^{r-1}((1-\alpha)(1-2\mu\eta_r)^K + \alpha) + \eta_r G \sum_{\tau=0}^{K-1} (\alpha - (1-\alpha)(1-2\mu\eta_r)^\tau).$$

Since α was picked obliviously, we can assume that $\alpha \leq 0.5$. If indeed $\alpha > 0.5$, we can swap the definitions of f_1 and f_2 and the sign of x^0 . With this, we can simplify as

$$\begin{aligned} x^r &\geq x^{r-1} \frac{(1-2\mu\eta_r)^K + 1}{2} + \frac{\eta_r G}{2} \sum_{\tau=0}^{K-1} (1 - (1-2\mu\eta_r)^\tau) \\ &\geq x^{r-1} (1-2\mu\eta_r)^K + \frac{\eta_r G}{2} \sum_{\tau=0}^{K-1} (1 - (1-2\mu\eta_r)^\tau). \end{aligned}$$

Observe that in the above expression, the right hand side is increasing with η_r —this represents the effect of the client drift and increases the error as the step-size increases. The left hand side decreases with η_r —this is the usual convergence observed due to taking gradient steps. The rest of the proof is to show that even with a careful balancing of the two terms, the effect of G cannot be removed. Lemma 63 performs exactly such a computation to prove that for any $r \geq 1$,

$$x^r \geq c \min(x_0, \frac{G}{\mu R}).$$

We finish the proof by noting that $f(x^r) = \frac{\mu}{2}(x^r)^2$. □

Lemma 38. Suppose that for all $r \geq 1$, $\eta_r \leq \frac{1}{\mu}$ and the following is true:

$$x^r \geq x^{r-1} (1-2\mu\eta_r)^K + \frac{\eta_r G}{2} \sum_{\tau=0}^{K-1} (1 - (1-2\mu\eta_r)^\tau).$$

Then, there exists a constant $c > 0$ such that for any sequence of step-sizes $\{\eta^r\}$:

$$x^r \geq c \min(x_0, \frac{G}{\mu R})$$

Proof. Define $\gamma_r = \mu\eta_r R(K-1)$. Such a γ_r exists and is positive since $K \geq 2$. Then, γ_r satisfies

$$(1 - 2\mu\eta_r)^{\frac{K-1}{2}} = \left(1 - \frac{2\gamma_r}{R(K-1)}\right)^{\frac{K-1}{2}} \leq \exp(-\gamma_r/R).$$

We then have

$$\begin{aligned} x^r &\geq x^{r-1}(1 - 2\mu\eta_r)^K + \frac{\eta_r G}{2} \sum_{\tau=0}^{K-1} (1 - (1 - 2\mu\eta_r)^\tau) \\ &\geq x^{r-1}(1 - 2\mu\eta_r)^K + \frac{\eta_r G}{2} \sum_{\tau=(K-1)/2}^{K-1} (1 - (1 - 2\mu\eta_r)^\tau) \\ &\geq x^{r-1}(1 - 2\mu\eta_r)^K + \frac{\gamma_r G}{4\mu} (1 - \exp(-\gamma_r/R)). \end{aligned}$$

The second inequality follows because $\eta_r \leq \frac{1}{\mu}$ implies that $(1 - (1 - 2\mu\eta_r)^\tau)$ is always positive. If $\gamma_r \geq R/8$, then we have a constant $c_1 \in (0, 1/32)$ which satisfies

$$x^r \geq \frac{c_1 G}{\mu}. \quad (11.9)$$

On the other hand, if $\gamma_r < R/8$, we have a tighter inequality

$$(1 - 2\mu\eta_r)^{\frac{K-1}{2}} = \left(1 - \frac{2\gamma_r}{R(K-1)}\right)^{\frac{K-1}{2}} \leq 1 - \frac{\gamma_r}{R},$$

implying that

$$\begin{aligned} x^r &\geq x^{r-1} \left(1 - \frac{2\gamma_r}{R(K-1)}\right)^K + \frac{\gamma_r^2 G}{4R\mu} \\ &\geq x^{r-1} \left(1 - \frac{4\gamma_r}{R}\right) + \frac{\gamma_r^2 G}{4\mu R}. \end{aligned} \quad (11.10)$$

The last step used Bernoulli's inequality and the fact that $K-1 \leq K/2$ for $K \geq 2$. Observe that in the above expression, the right hand side is increasing with γ_r —this represents the effect of the client drift and increases the error as the step-size increases. The left hand side decreases with γ_r —this is the usual convergence observed due to taking gradient steps. The rest of the proof is to show that even with a careful balancing of the two terms, the effect of G cannot be removed.

Suppose that all rounds after $r_0 \geq 0$ have a small step-size i.e. $\gamma_r \leq R/8$ for all $r > r_0$ and hence satisfies (11.10). Then we will prove via induction that

$$x^r \geq \min(c_r x^{r_0}, \frac{G}{256\mu R}), \text{ for constants } c_r := (1 - \frac{1}{2R})^{r-r_0}. \quad (11.11)$$

For $r = r_0$, (12.11) is trivially satisfied. Now for $r > r_0$,

$$\begin{aligned} x^r &\geq x^{r-1} \left(1 - \frac{4\gamma_r}{R}\right) + \frac{\gamma_r^2 G}{4\mu R} \\ &\geq \min\left(x^{r-1} \left(1 - \frac{1}{2R}\right), \frac{G}{256\mu R}\right) \\ &= \min\left(c_r x^{r_0}, \frac{G}{256\mu R}\right). \end{aligned}$$

The first step is because of (12.10) and the last step uses the induction hypothesis. The second step considers two cases for γ_r : either $\gamma_r \leq \frac{1}{8}$ and $(1 - \frac{1}{2R}) \geq (1 - \frac{1}{2R})$, or $\gamma_r^2 \geq \frac{1}{64}$. Finally note that $c^r \geq \frac{1}{2}$ using Bernoulli's inequality. We have hence proved

$$x^R \geq \min\left(\frac{1}{2} x^{r_0}, \frac{G}{256\mu R}\right)$$

Now suppose $\gamma_{r_0} > R/8$. Then (12.9) implies that $x^R \geq \frac{cG}{\mu R}$ for some constant $c > 0$. If instead no such $r_0 \geq 1$ exists, then we can set $r_0 = 0$. Now finally observe that the previous proof did not make any assumption on R , and in fact the inequality stated above holds for all $r \geq 1$. \square

11.5 Convergence of SCAFFOLD

WE FIRST RESTATE THE CONVERGENCE THEOREM MORE FORMALLY, THEN PROVE THE RESULT FOR THE CONVEX CASE, AND THEN FOR NON-CONVEX CASE. THROUGHOUT THE PROOF, WE WILL FOCUS ON THE HARDER OPTION II. THE PROOFS FOR SCAFFOLD WITH OPTION I ARE NEARLY IDENTICAL AND SO WE SKIP THEM.

Theorem XXVIII. *Suppose that the functions $\{f_i\}$ satisfies assumptions A4 and A5. Then, in each of the following cases, there exist weights $\{w_r\}$ and local step-sizes η_l such that for any $\eta_g \geq 1$ the output (12.16) of SCAFFOLD SATISFIES:*

- **STRONGLY CONVEX:** f_i SATISFIES (A3) FOR $\mu > 0$, $\eta_l \leq \min\left(\frac{1}{81\beta K\eta_g}, \frac{S}{15\mu N K\eta_g}\right)$, $R \geq \max\left(\frac{162\beta}{\mu}, \frac{30N}{S}\right)$ THEN

$$\mathbb{E}[f(\bar{\mathbf{x}}^R)] - f(\mathbf{x}^*) \leq \tilde{\mathcal{O}}\left(\frac{\sigma^2}{\mu R K S} \left(1 + \frac{S}{\eta_g^2}\right) + \frac{N\mu}{S} \tilde{D}^2 \exp\left(-\min\left\{\frac{S}{30N}, \frac{\mu}{162\beta}\right\} R\right)\right).$$

- **GENERAL CONVEX:** f_i SATISFIES (A3) FOR $\mu = 0$, $\eta_l \leq \frac{1}{81\beta K\eta_g}$, $R \geq 1$ THEN

$$\mathbb{E}[f(\bar{\mathbf{x}}^R)] - f(\mathbf{x}^*) \leq \mathcal{O}\left(\frac{\sigma \tilde{D}}{\sqrt{R K S}} \left(\sqrt{1 + \frac{S}{\eta_g^2}}\right) + \sqrt{\frac{N}{S}} \frac{\beta \tilde{D}^2}{R}\right),$$

- **NON-CONVEX:** $\eta_l \leq \frac{1}{24K\eta_g\beta} \left(\frac{S}{N}\right)^{\frac{2}{3}}$, AND $R \geq 1$, THEN

$$\mathbb{E}[\|\nabla f(\bar{\mathbf{x}}^R)\|^2] \leq \mathcal{O}\left(\frac{\sigma\sqrt{F}}{\sqrt{RKS}}\left(\sqrt{1+\frac{N}{\eta_g^2}}\right) + \frac{\beta F}{R}\left(\frac{N}{S}\right)^{\frac{2}{3}}\right).$$

HERE $\tilde{D}^2 := (\|\mathbf{x}^0 - \mathbf{x}^\star\|^2 + \frac{1}{2N\beta^2} \sum_{i=1}^N \|\mathbf{c}_i^0 - \nabla f_i(\mathbf{x}^\star)\|^2)$ AND $F := (f(\mathbf{x}_0) - f(\mathbf{x}^\star))$.

Remark 39. Note that the \tilde{D}^2 defined above involves an additional term $\frac{1}{2N\beta^2} \sum_{i=1}^N \|\mathbf{c}_i^0 - \nabla f_i(\mathbf{x}^\star)\|^2$. This is standard in variance reduction methods (Johnson and Zhang, 2013; Defazio et al., 2014; Hanzely and Richtárik, 2019). Theoretically, we will use a warm-start strategy to set \mathbf{c}_i^0 and in the first N/S rounds, we compute $\mathbf{c}_i^0 = \mathbf{g}_i(\mathbf{x}^0)$ over a batch size of size K . Then, using smoothness of f_i , we can bound this additional term as

$$\frac{1}{2N\beta^2} \sum_{i=1}^N \|\mathbf{c}_i^0 - \nabla f_i(\mathbf{x}^\star)\|^2 \leq \frac{1}{\beta} (f(\mathbf{x}^0) - f^\star) + \frac{\sigma^2}{K\beta^2} \leq D^2 + \frac{\sigma^2}{K\beta^2}.$$

Thus, the asymptotic rates of SCAFFOLD FOR GENERAL CONVEX FUNCTIONS ONLY INCURS AN ADDITIVE TERM OF THE ORDER OF $O(\sqrt{\frac{N}{S}} \frac{1}{R})$. FOR STRONGLY CONVEX FUNCTIONS, WE ONLY SEE THE AFFECTS IN THE LOGARITHMIC TERMS.

Remark 40. When $\sigma = 0$ i.e. when clients compute full gradients, the communication complexity of SCAFFOLD is: i) for strongly convex case it is $\tilde{O}\left(\frac{N}{S} + \frac{\beta}{\mu}\right)$, ii) for general convex functions it is $O\left(\sqrt{\frac{N}{S}} \frac{\beta}{R}\right)$,² and iii) for non-convex functions it is $O\left(\frac{N^{2/3}}{S} \frac{\beta}{R}\right)$. In comparison, the follow up work of FedDyn (Acar et al., 2021) proves communication complexity matching ours in the convex and strongly convex settings, but a worse $O\left(\frac{N}{S} \frac{\beta}{R}\right)$ complexity in the non-convex settings (all when $\sigma = 0$).

WE WILL REWRITE SCAFFOLD USING NOTATION WHICH IS CONVENIENT FOR THE PROOFS: $\{\mathbf{y}_i\}$ REPRESENT THE CLIENT MODELS, \mathbf{x} IS THE AGGREGATE SERVER MODEL, AND \mathbf{c}_i AND \mathbf{c} ARE THE CLIENT AND SERVER CONTROL VARIATES. FOR AN EQUIVALENT DESCRIPTION WHICH IS EASIER TO IMPLEMENT, WE REFER TO ALGORITHM 5. THE SERVER MAINTAINS A GLOBAL CONTROL VARIATE \mathbf{c} AS BEFORE AND EACH CLIENT MAINTAINS ITS OWN CONTROL VARIATE \mathbf{c}_i . IN ROUND r , A SUBSET OF CLIENTS \mathcal{S}^r OF SIZE S ARE SAMPLED UNIFORMLY FROM $\{1, \dots, N\}$. SUPPOSE THAT *every* CLIENT PERFORMS THE FOLLOWING UPDATES

- STARTING FROM THE SHARED GLOBAL PARAMETERS $\mathbf{y}_{i,r}^0 = \mathbf{x}^{r-1}$, WE UPDATE THE LOCAL PARAMETERS FOR $k \in [K]$

$$\mathbf{y}_{i,k}^r = \mathbf{y}_{i,k-1}^r - \eta_l \mathbf{v}_{i,k}^r, \quad \text{WHERE} \quad \mathbf{v}_{i,k}^r := \mathbf{g}_i(\mathbf{y}_{i,k-1}^r) - \mathbf{c}_i^{r-1} + \mathbf{c}^{r-1} \quad (11.12)$$

² A previous version of the paper showed a worse dependence of $O\left(\frac{N}{S} \frac{\beta}{R}\right)$ due to sub-optimal choice of step-size η .

- UPDATE THE CONTROL ITERATES USING (OPTION II):

$$\tilde{\mathbf{c}}_i^r = \mathbf{c}^{r-1} - \mathbf{c}_i^{r-1} + \frac{1}{K\eta_l}(\mathbf{x}^{r-1} - \mathbf{x}_{i,K}^r) = \frac{1}{K} \sum_{k=1}^K g_i(\mathbf{y}_{i,k-1}^r). \quad (11.13)$$

WE UPDATE THE LOCAL CONTROL VARIATES ONLY FOR CLIENTS $i \in \mathcal{S}^r$

$$\mathbf{c}_i^r = \begin{cases} \tilde{\mathbf{c}}_i^r & \text{IF } i \in \mathcal{S}^r \\ \mathbf{c}_i^{r-1} & \text{OTHERWISE.} \end{cases} \quad (11.14)$$

- COMPUTE THE NEW GLOBAL PARAMETERS AND GLOBAL CONTROL VARIATE USING ONLY UPDATES FROM THE CLIENTS $i \in \mathcal{S}^r$:

$$\mathbf{x}^r = \mathbf{x}^{r-1} + \frac{\eta_g}{S} \sum_{i \in \mathcal{S}^r} (\mathbf{y}_{i,K}^r - \mathbf{x}^{r-1}) \text{ AND } \mathbf{c}^r = \frac{1}{N} \sum_{i=1}^N \mathbf{c}_i^r = \frac{1}{N} \left(\sum_{i \in \mathcal{S}^r} \mathbf{c}_i^r + \sum_{j \notin \mathcal{S}^r} \mathbf{c}_j^{r-1} \right). \quad (11.15)$$

FINALLY, FOR SOME WEIGHTS $\{w_r\}$, WE OUTPUT

$$\bar{\mathbf{x}}^R = \mathbf{x}^{r-1} \text{ WITH PROBABILITY } \frac{w_r}{\sum_{\tau} w_{\tau}} \text{ FOR } r \in \{1, \dots, R+1\}. \quad (11.16)$$

NOTE THAT THE CLIENTS ARE AGNOSTIC TO THE SAMPLING AND THEIR UPDATES ARE IDENTICAL TO WHEN ALL CLIENTS ARE PARTICIPATING. ALSO NOTE THAT THE CONTROL VARIATE CHOICE (12.13) CORRESPONDS TO (OPTION II) OF ALGORITHM 5. FURTHER, THE UPDATES OF THE CLIENTS $i \notin \mathcal{S}^r$ IS FORGOTTEN AND IS DEFINED ONLY TO MAKE THE PROOFS EASIER. WHILE ACTUALLY IMPLEMENTING THE METHOD, ONLY CLIENTS $i \in \mathcal{S}^r$ PARTICIPATE AND THE REST REMAIN INACTIVE (SEE ALGORITHM 5).

11.5.1 Convergence of SCAFFOLD FOR CONVEX FUNCTIONS (THEOREM VIII)

WE WILL FIRST BOUND THE VARIANCE OF SCAFFOLD UPDATE IN LEMMA 66, THEN SEE HOW SAMPLING OF CLIENTS EFFECTS OUR CONTROL VARIATES IN LEMMA 67, AND FINALLY BOUND THE AMOUNT OF CLIENT-DRIFT IN LEMMA 68. WE WILL THEN USE THESE THREE LEMMAS TO PROVE THE PROGRESS IN A SINGLE ROUND IN LEMMA 69. COMBINING THIS PROGRESS WITH LEMMAS 55 AND 56 GIVES US THE DESIRED RATES.

ADDITIONAL DEFINITIONS. BEFORE PROCEEDING WITH THE PROOF OF OUR LEMMAS, WE NEED SOME ADDITIONAL DEFINITIONS OF THE VARIOUS ERRORS WE TRACK. AS BEFORE, WE DEFINE THE EFFECTIVE STEP-SIZE TO BE

$$\tilde{\eta} := K\eta_l\eta_g.$$

WE DEFINE CLIENT-DRIFT TO BE HOW MUCH THE CLIENTS MOVE FROM THEIR STARTING POINT:

$$\mathcal{E}_r := \frac{1}{KN} \sum_{k=1}^K \sum_{i=1}^N \mathbb{E}[\|\mathbf{y}_{i,k}^r - \mathbf{x}^{r-1}\|^2]. \quad (11.17)$$

BECAUSE WE ARE SAMPLING THE CLIENTS, NOT ALL THE CLIENT CONTROL-VARIATES GET UPDATED EVERY ROUND. THIS LEADS TO SOME ‘LAG’ WHICH WE CALL CONTROL-LAG:

$$\mathcal{C}_r := \frac{1}{N} \sum_{j=1}^N \mathbb{E} \|\mathbb{E}[\mathbf{c}_i^r] - \nabla f_i(\mathbf{x}^*)\|^2. \quad (11.18)$$

VARIANCE OF SERVER UPDATE. WE STUDY HOW THE VARIANCE OF THE SERVER UPDATE CAN BE BOUNDED.

Lemma 41. *For updates (12.12)—(12.15), we can bound the variance of the server update in any round r and any $\tilde{\eta} := \eta_l \eta_g K \geq 0$ as follows*

$$\mathbb{E}[\|\mathbf{x}^r - \mathbf{x}^{r-1}\|^2] \leq 8\beta\tilde{\eta}^2(\mathbb{E}[f(\mathbf{x}^{r-1})] - f(\mathbf{x}^*)) + 8\tilde{\eta}^2\mathcal{C}_{r-1} + 4\tilde{\eta}^2\beta^2\mathcal{E}_r + \frac{12\tilde{\eta}^2\sigma^2}{KS}.$$

Proof. The server update in round r can be written as follows (dropping the superscript r everywhere)

$$\mathbb{E}\|\Delta\mathbf{x}\|^2 = \mathbb{E}\left\|\frac{\tilde{\eta}}{KS} \sum_{k,i \in \mathcal{S}} \mathbf{v}_{i,k}\right\|^2 = \mathbb{E}\left\|\frac{\tilde{\eta}}{KS} \sum_{k,i \in \mathcal{S}} (g_i(\mathbf{y}_{i,k-1}) + \mathbf{c} - \mathbf{c}_i)\right\|^2,$$

which can then be expanded as

$$\begin{aligned} \mathbb{E}\|\Delta\mathbf{x}\|^2 &\leq \mathbb{E}\left\|\frac{\tilde{\eta}}{KS} \sum_{k,i \in \mathcal{S}} (g_i(\mathbf{y}_{i,k-1}) + \mathbf{c} - \mathbf{c}_i)\right\|^2 \\ &\leq 4\mathbb{E}\left\|\frac{\tilde{\eta}}{KS} \sum_{k,i \in \mathcal{S}} g_i(\mathbf{y}_{i,k-1}) - \nabla f_i(\mathbf{x})\right\|^2 + 4\tilde{\eta}^2\mathbb{E}\|\mathbf{c}\|^2 + 4\mathbb{E}\left\|\frac{\tilde{\eta}}{KS} \sum_{k,i \in \mathcal{S}} \nabla f_i(\mathbf{x}^*) - \mathbf{c}_i\right\|^2 \\ &\quad + 4\mathbb{E}\left\|\frac{\tilde{\eta}}{KS} \sum_{k,i \in \mathcal{S}} \nabla f_i(\mathbf{x}) - \nabla f_i(\mathbf{x}^*)\right\|^2 \\ &\stackrel{(12.3)}{\leq} 4\mathbb{E}\left\|\frac{\tilde{\eta}}{KS} \sum_{k,i \in \mathcal{S}} g_i(\mathbf{y}_{i,k-1}) - \nabla f_i(\mathbf{x})\right\|^2 + 4\tilde{\eta}^2\mathbb{E}\|\mathbf{c}\|^2 + 4\mathbb{E}\left\|\frac{\tilde{\eta}}{S} \sum_{i \in \mathcal{S}} \nabla f_i(\mathbf{x}^*) - \mathbf{c}_i\right\|^2 \\ &\quad + 8\beta\tilde{\eta}^2(\mathbb{E}[f(\mathbf{x})] - f(\mathbf{x}^*)) \\ &\leq 4\mathbb{E}\left\|\frac{\tilde{\eta}}{KS} \sum_{k,i \in \mathcal{S}} \nabla f_i(\mathbf{y}_{i,k-1}) - \nabla f_i(\mathbf{x})\right\|^2 + 4\tilde{\eta}^2\mathbb{E}\|\mathbf{c}\|^2 + 4\left\|\frac{\tilde{\eta}}{S} \sum_{i \in \mathcal{S}} \nabla f_i(\mathbf{x}^*) - \mathbb{E}[\mathbf{c}_i]\right\|^2 \\ &\quad + 8\beta\tilde{\eta}^2(\mathbb{E}[f(\mathbf{x})] - f(\mathbf{x}^*)) + \frac{12\tilde{\eta}^2\sigma^2}{KS}. \end{aligned}$$

The inequality before the last used the smoothness of $\{f_i\}$. The last inequality which separates the mean and the variance is an application of Lemma 58: the variance of $(\frac{1}{KS} \sum_{k,i \in \mathcal{S}} g_i(\mathbf{y}_{i,k-1}))$ is bounded by σ^2/KS . Similarly, \mathbf{c}_j as defined in (12.13) for any $j \in [N]$ has variance smaller than σ^2/K and hence the variance of $(\frac{1}{S} \sum_{i \in \mathcal{S}} \mathbf{c}_i)$ is smaller than σ^2/KS .

Using Lemma 57.2 twice to simplify:

$$\begin{aligned}
 \mathbb{E}\|\Delta \mathbf{x}\|^2 &\leq \frac{4\tilde{\eta}^2}{KN} \sum_{k,i} \mathbb{E}\|\nabla f_i(\mathbf{y}_{i,k-1}) - \nabla f_i(\mathbf{x})\|^2 + 4\tilde{\eta}^2 \|\mathbb{E}[\mathbf{c}]\|^2 + \frac{4\tilde{\eta}^2}{N} \sum_i \|\nabla f_i(\mathbf{x}^\star) - \mathbb{E}[\mathbf{c}_i]\|^2 \\
 &\quad + 8\beta\tilde{\eta}^2(\mathbb{E}[f(\mathbf{x})] - f(\mathbf{x}^\star)) + \frac{12\tilde{\eta}^2\sigma^2}{KS} \\
 &\leq \underbrace{\frac{4\tilde{\eta}^2}{KN} \sum_{k,i} \mathbb{E}\|\nabla f_i(\mathbf{y}_{i,k-1}) - \nabla f_i(\mathbf{x})\|^2}_{\mathcal{T}_1} + \frac{8\tilde{\eta}^2}{N} \sum_i \|\nabla f_i(\mathbf{x}^\star) - \mathbb{E}[\mathbf{c}_i]\|^2 \\
 &\quad + 8\beta\tilde{\eta}^2(\mathbb{E}[f(\mathbf{x})] - f(\mathbf{x}^\star)) + \frac{12\tilde{\eta}^2\sigma^2}{KS}.
 \end{aligned}$$

The second step follows because $\mathbf{c} = \frac{1}{N} \sum_i \mathbf{c}_i$. Since the gradient of f_i is β -Lipschitz, $\mathcal{T}_1 \leq \frac{\beta^2 4\tilde{\eta}^2}{KN} \sum_{k,i} \mathbb{E}\|\mathbf{y}_{i,k-1} - \mathbf{x}\|^2 = 4\tilde{\eta}^2 \beta^2 \mathcal{E}$. The definition of the error in the control variate $\mathcal{C}_{r-1} := \frac{1}{N} \sum_{j=1}^N \mathbb{E}\|\mathbf{c}_j - \nabla f_j(\mathbf{x}^\star)\|^2$ completes the proof. \square

CHANGE IN CONTROL LAG. WE HAVE PREVIOUSLY RELATED THE VARIANCE OF THE SERVER UPDATE TO THE CONTROL LAG. WE NOW EXAMINE HOW THE CONTROL-LAG GROWS EACH ROUND.

Lemma 42. For updates (12.12)—(12.15) with the control update (12.13) and assumptions A3–A5, the following holds true for any $\tilde{\eta} := \eta_l \eta_g K \in [0, 1/\beta]$:

$$\mathcal{C}_r \leq (1 - \frac{S}{N})\mathcal{C}_{r-1} + \frac{S}{N}(4\beta(\mathbb{E}[f(\mathbf{x}^{r-1})] - f(\mathbf{x}^\star)) + 2\beta^2 \mathcal{E}_r).$$

Proof. Recall that after round r , the control update rule (12.13) implies that \mathbf{c}_i^r is set as per

$$\mathbf{c}_i^r = \begin{cases} \mathbf{c}_i^{r-1} & \text{if } i \notin \mathcal{S}^r \text{ i.e. with probability } (1 - \frac{S}{N}), \\ \frac{1}{K} \sum_{k=1}^K \mathbf{g}_i(\mathbf{y}_{i,k-1}^r) & \text{with probability } \frac{S}{N}. \end{cases}$$

Taking expectations on both sides yields

$$\mathbb{E}[\mathbf{c}_i^r] = (1 - \frac{S}{N})\mathbb{E}[\mathbf{c}_i^{r-1}] + \frac{S}{KN} \sum_{k=1}^K \mathbb{E}[\nabla f_i(\mathbf{y}_{i,k-1}^r)], \quad \forall i \in [N].$$

Plugging the above expression in the definition of \mathcal{C}_r we get

$$\begin{aligned}
 \mathcal{C}_r &= \frac{1}{N} \sum_{i=1}^N \|\mathbb{E}[\mathbf{c}_i^r] - \nabla f_i(\mathbf{x}^\star)\|^2 \\
 &= \frac{1}{N} \sum_{i=1}^N \|(1 - \frac{S}{N})(\mathbb{E}[\mathbf{c}_i^{r-1}] - \nabla f_i(\mathbf{x}^\star)) + \frac{S}{N}(\frac{1}{K} \sum_{k=1}^K \mathbb{E}[\nabla f_i(\mathbf{y}_{i,k-1}^r)] - \nabla f_i(\mathbf{x}^\star))\|^2 \\
 &\leq (1 - \frac{S}{N})\mathcal{C}_{r-1} + \frac{S}{N^2 K} \sum_{k=1}^K \mathbb{E}\|\nabla f_i(\mathbf{y}_{i,k-1}^r) - \nabla f_i(\mathbf{x}^\star)\|^2.
 \end{aligned}$$

The final step applied Jensen's inequality twice. We can then further simplify using the re-

laxed triangle inequality as

$$\begin{aligned}
 \mathbb{E}_{r-1}[\mathcal{E}_r] &\leq \left(1 - \frac{S}{N}\right) \mathcal{E}_{r-1} + \frac{S}{N^2 K} \sum_{i,k} \mathbb{E} \|\nabla f_i(\mathbf{y}_{i,k-1}^r) - \nabla f_i(\mathbf{x}^*)\|^2 \\
 &\leq \left(1 - \frac{S}{N}\right) \mathcal{E}_{r-1} + \frac{2S}{N^2} \sum_i \mathbb{E} \|\nabla f_i(\mathbf{x}^{r-1}) - \nabla f_i(\mathbf{x}^*)\|^2 + \frac{2S}{N^2 K} \sum_{i,k} \mathbb{E} \|\nabla f_i(\mathbf{y}_{i,k-1}^r) - \nabla f_i(\mathbf{x}^{r-1})\|^2 \\
 &\stackrel{(12.1)}{\leq} \left(1 - \frac{S}{N}\right) \mathcal{E}_{r-1} + \frac{2S}{N^2} \sum_i \mathbb{E} \|\nabla f_i(\mathbf{x}^{r-1}) - \nabla f_i(\mathbf{x}^*)\|^2 + \frac{2S}{N^2 K} \beta^2 \sum_{i,k} \mathbb{E} \|\mathbf{y}_{i,k-1}^r - \mathbf{x}^{r-1}\|^2 \\
 &\stackrel{(12.3)}{\leq} \left(1 - \frac{S}{N}\right) \mathcal{E}_{r-1} + \frac{S}{N} (4\beta(\mathbb{E}[f(\mathbf{x}^{r-1})] - f(\mathbf{x}^*)) + \beta^2 \mathcal{E}_r).
 \end{aligned}$$

The last two inequalities follow from smoothness of $\{f_i\}$ and the definition $\mathcal{E}_r = \frac{1}{NK} \beta^2 \sum_{i,k} \mathbb{E} \|\mathbf{y}_{i,k-1}^r - \mathbf{x}^{r-1}\|^2$. \square

BOUNDING CLIENT-DRIFT. WE WILL NOW BOUND THE FINAL SOURCE OF ERROR WHICH IS THE CLIENT-DRIFT.

Lemma 43. Suppose our step-sizes satisfy $\eta_l \leq \frac{1}{81\beta K \eta_g}$ and f_i satisfies assumptions A3–A5. Then, for any global $\eta_g \geq 1$ we can bound the drift as

$$3\beta\tilde{\eta}\mathcal{E}_r \leq \frac{2\tilde{\eta}^2}{3} \mathcal{E}_{r-1} + \frac{\tilde{\eta}}{25\eta_g^2} (\mathbb{E}[f(\mathbf{x}^{r-1})] - f(\mathbf{x}^*)) + \frac{\tilde{\eta}^2}{K\eta_g^2} \sigma^2.$$

Proof. First, observe that if $K = 1$, $\mathcal{E}_r = 0$ since $\mathbf{y}_{i,0} = \mathbf{x}$ for all $i \in [N]$ and that \mathcal{E}_{r-1} and the right hand side are both positive. Thus the lemma is trivially true if $K = 1$. For $K > 1$, we build a recursive bound of the drift. Starting from the definition of the update (12.12) and then applying the relaxed triangle inequality, we can expand

$$\begin{aligned}
 \frac{1}{S} \mathbb{E}_{r-1} \left[\sum_{i \in \mathcal{S}} \|\mathbf{y}_i - \eta_l \mathbf{v}_i - \mathbf{x}\|^2 \right] &= \frac{1}{S} \mathbb{E}_{r-1} \left[\sum_{i \in \mathcal{S}} \|\mathbf{y}_i - \eta_l \mathbf{g}_i(\mathbf{y}_i) + \eta_l \mathbf{c} - \eta_l \mathbf{c}_i - \mathbf{x}\|^2 \right] \\
 &\leq \frac{1}{S} \mathbb{E}_{r-1} \left[\sum_{i \in \mathcal{S}} \|\mathbf{y}_i - \eta_l \nabla f_i(\mathbf{y}_i) + \eta_l \mathbf{c} - \eta_l \mathbf{c}_i - \mathbf{x}\|^2 \right] + \eta_l^2 \sigma^2 \\
 &\leq \frac{(1+a)}{S} \mathbb{E}_{r-1} \left[\sum_{i \in \mathcal{S}} \underbrace{\|\mathbf{y}_i - \eta_l \nabla f_i(\mathbf{y}_i) + \eta_l \nabla f_i(\mathbf{x}) - \mathbf{x}\|^2}_{\mathcal{T}_2} \right] \\
 &\quad + \underbrace{(1 + \frac{1}{a}) \eta_l^2 \mathbb{E}_{r-1} \left[\frac{1}{S} \sum_{i \in \mathcal{S}} \|\mathbf{c} - \mathbf{c}_i + \nabla f_i(\mathbf{x})\|^2 \right]}_{\mathcal{T}_3} + \eta_l^2 \sigma^2.
 \end{aligned}$$

The final step follows from the relaxed triangle inequality (Lemma 57). Applying the contractive mapping Lemma 60 for $\eta_l \leq 1/\beta$ shows

$$\mathcal{T}_2 = \frac{1}{S} \sum_{i \in \mathcal{S}} \|\mathbf{y}_i - \eta_l \nabla f_i(\mathbf{y}_i) + \eta_l \nabla f_i(\mathbf{x}) - \mathbf{x}\|^2 \leq \|\mathbf{y}_i - \mathbf{x}\|^2.$$

Once again using our relaxed triangle inequality to expand the other term \mathcal{T}_3 , we get

$$\begin{aligned}
 \mathcal{T}_3 &= \mathbb{E}_{r-1} \left[\frac{1}{S} \sum_{i \in \mathcal{S}} \|\mathbf{c} - \mathbf{c}_i + \nabla f_i(\mathbf{x})\|^2 \right] \\
 &= \frac{1}{N} \sum_{j=1}^N \|\mathbf{c} - \mathbf{c}_i + \nabla f_i(\mathbf{x})\|^2 \\
 &= \frac{1}{N} \sum_{j=1}^N \|\mathbf{c} - \mathbf{c}_i + \nabla f_i(\mathbf{x}^*) + \nabla f_i(\mathbf{x}) - \nabla f_i(\mathbf{x}^*)\|^2 \\
 &\leq 3\|\mathbf{c}\|^2 + \frac{3}{N} \sum_{j=1}^N \|\mathbf{c}_i - \nabla f_i(\mathbf{x}^*)\|^2 + \frac{3}{N} \sum_{j=1}^N \|\nabla f_i(\mathbf{x}) - \nabla f_i(\mathbf{x}^*)\|^2 \\
 &\leq \frac{6}{N} \sum_{j=1}^N \|\mathbf{c}_i - \nabla f_i(\mathbf{x}^*)\|^2 + \frac{3}{N} \sum_{j=1}^N \|\nabla f_i(\mathbf{x}) - \nabla f_i(\mathbf{x}^*)\|^2 \\
 &\leq \frac{6}{N} \sum_{j=1}^N \|\mathbf{c}_i - \nabla f_i(\mathbf{x}^*)\|^2 + 6\beta(f(\mathbf{x}) - f(\mathbf{x}^*)).
 \end{aligned}$$

The last step used the smoothness of f_i . Combining the bounds on \mathcal{T}_2 and \mathcal{T}_3 in the original inequality and using $a = \frac{1}{K-1}$ gives

$$\begin{aligned}
 \frac{1}{N} \sum_i \mathbb{E} \|\mathbf{y}_{i,k} - \mathbf{x}\|^2 &\leq \frac{(1 + \frac{1}{K-1})}{N} \sum_i \mathbb{E} \|\mathbf{y}_{i,k-1} - \mathbf{x}\|^2 + \eta_l^2 \sigma^2 \\
 &\quad + 6\eta_l^2 K \beta(f(\mathbf{x}) - f(\mathbf{x}^*)) + \frac{6K\eta_l^2}{N} \sum_i \mathbb{E} \|\mathbf{c}_i - \nabla f_i(\mathbf{x}^*)\|^2.
 \end{aligned}$$

Recall that with the choice of \mathbf{c}_i in (12.13), the variance of c_i is less than $\frac{\sigma^2}{K}$. Separating its mean and variance gives

$$\begin{aligned}
 \frac{1}{N} \sum_i \mathbb{E} \|\mathbf{y}_{i,k} - \mathbf{x}\|^2 &\leq \left(1 + \frac{1}{K-1}\right) \frac{1}{N} \sum_i \mathbb{E} \|\mathbf{y}_{i,k-1} - \mathbf{x}\|^2 + 7\eta_l^2 \sigma^2 + \\
 &\quad 6\eta_l^2 K \beta(f(\mathbf{x}) - f(\mathbf{x}^*)) + \frac{6K\eta_l^2}{N} \sum_i \mathbb{E} \|\mathbf{c}_i - \nabla f_i(\mathbf{x}^*)\|^2 \quad (11.19)
 \end{aligned}$$

Unrolling the recursion (12.19), we get the following for any $k \in \{1, \dots, K\}$

$$\begin{aligned}
 \frac{1}{N} \sum_i \mathbb{E} \|\mathbf{y}_{i,k} - \mathbf{x}\|^2 &\leq (6K\beta\eta_l^2(f(\mathbf{x}) - f(\mathbf{x}^*)) + 6K\eta_l^2\mathcal{C}_{r-1} + 7\beta\eta_l^2\sigma^2) \left(\sum_{\tau=0}^{k-1} \left(1 + \frac{1}{K-1}\right)^\tau \right) \\
 &\leq (6K\beta\eta_l^2(f(\mathbf{x}) - f(\mathbf{x}^*)) + 6K\eta_l^2\mathcal{C}_{r-1} + 7\beta\eta_l^2\sigma^2) (K-1) \left(\left(1 + \frac{1}{K-1}\right)^K - 1 \right) \\
 &\leq (6K\beta\eta_l^2(f(\mathbf{x}) - f(\mathbf{x}^*)) + 6K\eta_l^2\mathcal{C}_{r-1} + 7\beta\eta_l^2\sigma^2) 3K \\
 &\leq 18K^2\beta\eta_l^2(f(\mathbf{x}) - f(\mathbf{x}^*)) + 18K^2\eta_l^2\mathcal{C}_{r-1} + 21K\beta\eta_l^2\sigma^2.
 \end{aligned}$$

The inequality $(K-1)((1 + \frac{1}{K-1})^K - 1) \leq 3K$ can be verified for $K = 2, 3$ manually. For $K \geq 4$,

$$(K-1)((1 + \frac{1}{K-1})^K - 1) < K(\exp(\frac{K}{K-1}) - 1) \leq K(\exp(\frac{4}{3}) - 1) < 3K.$$

Again averaging over k and multiplying by 3β yields

$$\begin{aligned} 3\beta\mathcal{E}_r &\leq 54K^2\beta^2\eta_l^2(f(\mathbf{x}) - f(\mathbf{x}^*)) + 54K^2\beta\eta_l^2\mathcal{C}_{r-1} + 63\beta K\eta_l^2\sigma^2 \\ &= \frac{1}{\eta_g^2} \left(54\beta^2\tilde{\eta}^2(f(\mathbf{x}) - f(\mathbf{x}^*)) + 54\beta\tilde{\eta}^2\mathcal{C}_{r-1} + 63\beta\tilde{\eta}^2\frac{\sigma^2}{K} \right) \\ &\leq \frac{1}{\eta_g^2} \left(\frac{1}{25}(f(\mathbf{x}) - f(\mathbf{x}^*)) + \frac{2}{3}\tilde{\eta}\mathcal{C}_{r-1} + \tilde{\eta}\frac{\sigma^2}{K} \right). \end{aligned}$$

The equality follows from the definition $\tilde{\eta} = K\eta_l\eta_g$, and the final inequality uses the bound that $\tilde{\eta} \leq \frac{1}{81\beta}$. \square

PROGRESS IN ONE ROUND. NOW THAT WE HAVE A BOUND ON ALL ERRORS, WE CAN DESCRIBE OUR PROGRESS.

Lemma 44. Suppose assumptions A3–A5 are true. Then the following holds for any step-sizes satisfying $\eta_g \geq 1$, $\eta_l \leq \min\left(\frac{1}{81\beta K\eta_g}, \frac{S}{15\mu N K\eta_g}\right)$, and effective step-size $\tilde{\eta} := K\eta_g\eta_l$

$$\mathbb{E} \left[\|\mathbf{x}^r - \mathbf{x}^*\|^2 + \frac{9N\tilde{\eta}^2}{S}\mathcal{C}_r \right] \leq (1 - \frac{\mu\tilde{\eta}}{2}) \left(\mathbb{E} \|\mathbf{x}^{r-1} - \mathbf{x}^*\|^2 + \frac{9N\tilde{\eta}^2}{S}\mathcal{C}_{r-1} \right) - \tilde{\eta}(\mathbb{E}[f(\mathbf{x}^{r-1})] - f(\mathbf{x}^*)) + \frac{12\tilde{\eta}^2}{KS} \left(1 + \frac{S}{\eta_g^2} \right) \sigma^2.$$

Proof. Starting from our server update equation,

$$\Delta\mathbf{x} = -\frac{\tilde{\eta}}{KS} \sum_{k,i \in \mathcal{S}} (g_i(\mathbf{y}_{i,k-1}) + \mathbf{c} - \mathbf{c}_i), \text{ and } \mathbb{E}[\Delta\mathbf{x}] = -\frac{\tilde{\eta}}{KN} \sum_{k,i} g_i(\mathbf{y}_{i,k-1}).$$

We can then apply Lemma 66 to bound the second moment of the server update as

$$\begin{aligned} \mathbb{E}_{r-1} \|\mathbf{x} + \Delta\mathbf{x} - \mathbf{x}^*\|^2 &= \mathbb{E}_{r-1} \|\mathbf{x} - \mathbf{x}^*\|^2 - \frac{2\tilde{\eta}}{KS} \mathbb{E}_{r-1} \sum_{k,i \in \mathcal{S}} \langle \nabla f_i(\mathbf{y}_{i,k-1}), \mathbf{x} - \mathbf{x}^* \rangle + \mathbb{E}_{r-1} \|\Delta\mathbf{x}\|^2 \\ &\leq \underbrace{\frac{2\tilde{\eta}}{KS} \mathbb{E}_{r-1} \sum_{k,i \in \mathcal{S}} \langle \nabla f_i(\mathbf{y}_{i,k-1}), \mathbf{x}^* - \mathbf{x} \rangle + \mathbb{E}_{r-1} \|\mathbf{x} - \mathbf{x}^*\|^2}_{\mathcal{T}_4} \\ &\quad + 8\beta\tilde{\eta}^2(\mathbb{E}[f(\mathbf{x}^{r-1})] - f(\mathbf{x}^*)) + 8\tilde{\eta}^2\mathcal{C}_{r-1} + 4\tilde{\eta}^2\beta^2\mathcal{E} + \frac{12\tilde{\eta}^2\sigma^2}{KS}. \end{aligned}$$

The term \mathcal{T}_4 can be bounded by using perturbed strong-convexity (Lemma 59) with $h = f_i$,

$\mathbf{x} = \mathbf{y}_{i,k-1}$, $\mathbf{y} = \mathbf{x}^\star$, and $\mathbf{z} = \mathbf{x}$ to get

$$\begin{aligned}\mathbb{E}[\mathcal{T}_4] &= \frac{2\tilde{\eta}}{KS} \mathbb{E} \sum_{k,i \in \mathcal{S}} \langle \nabla f_i(\mathbf{y}_{i,k-1}), \mathbf{x}^\star - \mathbf{x} \rangle \\ &\leq \frac{2\tilde{\eta}}{KS} \mathbb{E} \sum_{k,i \in \mathcal{S}} \left(f_i(\mathbf{x}^\star) - f_i(\mathbf{x}) + \beta \|\mathbf{y}_{i,k-1} - \mathbf{x}\|^2 - \frac{\mu}{4} \|\mathbf{x} - \mathbf{x}^\star\|^2 \right) \\ &= -2\tilde{\eta} \mathbb{E} \left(f(\mathbf{x}) - f(\mathbf{x}^\star) + \frac{\mu}{4} \|\mathbf{x} - \mathbf{x}^\star\|^2 \right) + 2\beta\tilde{\eta}\mathcal{E}.\end{aligned}$$

Plugging \mathcal{T}_4 back, we can further simplify the expression to get

$$\begin{aligned}\mathbb{E}\|\mathbf{x} + \Delta\mathbf{x} - \mathbf{x}^\star\|^2 &\leq \mathbb{E}\|\mathbf{x} - \mathbf{x}^\star\|^2 - 2\tilde{\eta} \left(f(\mathbf{x}) - f(\mathbf{x}^\star) + \frac{\mu}{4} \|\mathbf{x} - \mathbf{x}^\star\|^2 \right) + 2\beta\tilde{\eta}\mathcal{E} \\ &\quad + \frac{12\tilde{\eta}^2\sigma^2}{KS} + 8\beta\tilde{\eta}^2(\mathbb{E}[f(\mathbf{x}^{r-1})] - f(\mathbf{x}^\star)) + 8\tilde{\eta}^2\mathcal{C}_{r-1} + 4\tilde{\eta}^2\beta^2\mathcal{E} \\ &= (1 - \frac{\mu\tilde{\eta}}{2})\|\mathbf{x} - \mathbf{x}^\star\|^2 + (8\beta\tilde{\eta}^2 - 2\tilde{\eta})(f(\mathbf{x}) - f(\mathbf{x}^\star)) \\ &\quad + \frac{12\tilde{\eta}^2\sigma^2}{KS} + (2\beta\tilde{\eta} + 4\beta^2\tilde{\eta}^2)\mathcal{E} + 8\tilde{\eta}^2\mathcal{C}_{r-1}.\end{aligned}$$

We can use Lemma 67 (scaled by $9\tilde{\eta}^2 \frac{N}{S}$) to bound the control-lag

$$9\tilde{\eta}^2 \frac{N}{S} \mathcal{C}_r \leq (1 - \frac{\mu\tilde{\eta}}{2})9\tilde{\eta}^2 \frac{N}{S} \mathcal{C}_{r-1} + 9(\frac{\mu\tilde{\eta}N}{2S} - 1)\tilde{\eta}^2\mathcal{C}_{r-1} + 9\tilde{\eta}^2(4\beta(\mathbb{E}[f(\mathbf{x}^{r-1})] - f(\mathbf{x}^\star)) + 2\beta^2\mathcal{E})$$

Now recall that Lemma 68 bounds the client-drift:

$$3\beta\tilde{\eta}\mathcal{E}_r \leq \frac{2\tilde{\eta}^2}{3}\mathcal{C}_{r-1} + \frac{\tilde{\eta}}{25\tilde{\eta}_g^2}(\mathbb{E}[f(\mathbf{x}^{r-1})] - f(\mathbf{x}^\star)) + \frac{\tilde{\eta}^2}{K\tilde{\eta}_g^2}\sigma^2.$$

Adding all three inequalities together,

$$\begin{aligned}\mathbb{E}\|\mathbf{x} + \Delta\mathbf{x} - \mathbf{x}^\star\|^2 + \frac{9\tilde{\eta}^2 N \mathcal{C}_r}{S} &\leq (1 - \frac{\mu\tilde{\eta}}{2}) \left(\mathbb{E}\|\mathbf{x} - \mathbf{x}^\star\|^2 + \frac{9\tilde{\eta}^2 N \mathcal{C}_{r-1}}{S} \right) + (44\beta\tilde{\eta}^2 - \frac{49}{25}\tilde{\eta})(f(\mathbf{x}) - f(\mathbf{x}^\star)) \\ &\quad + \frac{12\tilde{\eta}^2\sigma^2}{KS} (1 + \frac{S}{\tilde{\eta}_g^2}) + (22\beta^2\tilde{\eta}^2 - \beta\tilde{\eta})\mathcal{E} + (\frac{9\mu\tilde{\eta}N}{2S} - \frac{1}{3})\tilde{\eta}^2\mathcal{C}_{r-1}\end{aligned}$$

Finally, the lemma follows from noting that $\tilde{\eta} \leq \frac{1}{81\tilde{\beta}}$ implies $44\beta^2\tilde{\eta}^2 \leq \frac{24}{25}\tilde{\beta}$ and $\tilde{\eta} \leq \frac{S}{15\mu N}$ implies $\frac{9\mu\tilde{\eta}N}{2S} \leq \frac{1}{3}$. \square

THE FINAL RATE FOR STRONGLY CONVEX FOLLOWS SIMPLY BY UNROLLING THE RECURSIVE BOUND IN LEMMA 69 USING LEMMA 55. ALSO NOTE THAT IF $c_i^0 = \mathbf{g}_i(\mathbf{x}^0)$, THEN $\frac{\tilde{\eta}N}{S}\mathcal{C}_0$ CAN BE BOUNDED IN TERMS OF FUNCTION SUB-OPTIMALITY F . FOR THE **GENERAL CONVEX** SETTING, AVERAGING

OVER r IN LEMMA 69 WITH $\mu = 0$ GIVES

$$\begin{aligned} \frac{1}{R} \sum_{r=1}^R \mathbb{E}[f(\mathbf{x}^{r-1})] - f(\mathbf{x}^\star) &\leq \frac{1}{\tilde{\eta}R} \|\mathbf{x}^0 - \mathbf{x}^\star\|^2 + \frac{9N\tilde{\eta}}{SR} \mathcal{C}_0 + \frac{12\tilde{\eta}}{KS} (1 + \frac{S}{\eta_g^2}) \sigma^2 \\ &\leq 4 \|\mathbf{x}^0 - \mathbf{x}^\star\| \sigma \sqrt{\frac{3(1 + S/\eta_g^2)}{RKS}} \\ &\quad + \sqrt{\frac{N}{S} \frac{\|\mathbf{x}^0 - \mathbf{x}^\star\|^2 + 9\mathcal{C}_0}{R} + \frac{81\beta \|\mathbf{x}^0 - \mathbf{x}^\star\|^2}{R}} \dots \end{aligned}$$

THE LAST STEP FOLLOWS FROM USING A STEP SIZE OF $\tilde{\eta} = \min \left(\frac{1}{81\beta}, \sqrt{\frac{S}{N}}, \frac{\|\mathbf{x}^0 - \mathbf{x}^\star\|}{\sigma} \sqrt{\frac{KS}{12R(1 + \frac{S}{\eta_g^2})}} \right)$.

11.5.2 Convergence of SCAFFOLD FOR NON-CONVEX FUNCTIONS (THEOREM VIII)

WE NOW ANALYZE THE MOST GENERAL CASE OF SCAFFOLD WITH OPTION II ON FUNCTIONS WHICH ARE POTENTIALLY NON-CONVEX. JUST AS IN THE NON-CONVEX PROOF, WE WILL FIRST BOUND THE VARIANCE OF THE SERVER UPDATE IN LEMMA 70, THE CHANGE IN CONTROL LAG IN LEMMA 71 AND FINALLY WE BOUND THE CLIENT-DRIFT IN LEMMA 72. COMBINING THESE THREE TOGETHER GIVES US THE PROGRESS MADE IN ONE ROUND IN LEMMA 73. THE FINAL RATE IS DERIVED FROM THE PROGRESS MADE USING LEMMA 56.

ADDITIONAL NOTATION. RECALL THAT IN ROUND r , WE UPDATE THE CONTROL VARIATE AS (12.13)

$$\mathbf{c}_i^r = \begin{cases} \frac{1}{K} \sum_{k=1}^K \mathbf{g}_i(\mathbf{y}_{i,k-1}^r) & \text{IF } i \in \mathcal{S}^r, \\ \mathbf{c}_i^{r-1} & \text{OTHERWISE.} \end{cases}$$

WE INTRODUCE THE FOLLOWING NOTATION TO KEEP TRACK OF THE ‘LAG’ IN THE UPDATE OF THE CONTROL VARIATE: DEFINE A SEQUENCE OF PARAMETERS $\{\boldsymbol{\alpha}_{i,k-1}^{r-1}\}$ SUCH THAT FOR ANY $i \in [N]$ AND $k \in [K]$ WE HAVE $\boldsymbol{\alpha}_{i,k-1}^0 := \mathbf{x}^0$ AND FOR $r \geq 1$,

$$\boldsymbol{\alpha}_{i,k-1}^r := \begin{cases} \mathbf{y}_{i,k-1}^r & \text{IF } i \in \mathcal{S}^r, \\ \boldsymbol{\alpha}_{i,k-1}^{r-1} & \text{OTHERWISE.} \end{cases} \quad (11.20)$$

BY THE UPDATE RULE FOR CONTROL VARIATES (12.13) AND THE DEFINITION OF $\{\boldsymbol{\alpha}_{i,k-1}^{r-1}\}$ ABOVE, THE FOLLOWING PROPERTY ALWAYS HOLDS:

$$\mathbf{c}_i^r = \frac{1}{K} \sum_{k=1}^K \mathbf{g}_i(\boldsymbol{\alpha}_{i,k-1}^r).$$

WE CAN THEN DEFINE THE FOLLOWING Ξ_r TO BE THE ERROR IN CONTROL VARIATE FOR ROUND r :

$$\Xi_r := \frac{1}{KN} \sum_{k=1}^K \sum_{i=1}^N \mathbb{E} \|\boldsymbol{\alpha}_{i,k-1}^r - \mathbf{x}^r\|^2. \quad (11.21)$$

ALSO RECALL THE CLOSELY RELATED DEFINITION OF CLIENT DRIFT CAUSED BY LOCAL UPDATES:

$$\mathcal{E}_r := \frac{1}{KN} \sum_{k=1}^K \sum_{i=1}^N \mathbb{E}[\|\mathbf{y}_{i,k}^r - \mathbf{x}^{r-1}\|^2].$$

VARIANCE OF SERVER UPDATE. LET US ANALYZE HOW THE CONTROL VARIATES EFFECT THE VARIANCE OF THE AGGREGATE SERVER UPDATE.

Lemma 45. *For updates (12.12)—(12.15) and assumptions A4 and A5, the following holds true for any $\tilde{\eta} := \eta_l \eta_g K \in [0, 1/\beta]$:*

$$\begin{aligned} \mathbb{E}\|\mathbb{E}_{r-1}[\mathbf{x}^r] - \mathbf{x}^{r-1}\|^2 &\leq 2\tilde{\eta}^2 \beta^2 \mathcal{E}_r + 2\tilde{\eta}^2 \mathbb{E}\|\nabla f(\mathbf{x}^{r-1})\|^2, \text{ and} \\ \mathbb{E}\|\mathbf{x}^r - \mathbf{x}^{r-1}\|^2 &\leq 4\tilde{\eta}^2 \beta^2 \mathcal{E}_r + 8\tilde{\eta}^2 \beta^2 \Xi_{r-1} + 4\tilde{\eta}^2 \mathbb{E}\|\nabla f(\mathbf{x}^{r-1})\|^2 + \frac{9\tilde{\eta}^2 \sigma^2}{KS}. \end{aligned}$$

Proof. Recall that the server update satisfies

$$\mathbb{E}[\Delta \mathbf{x}] = -\frac{\tilde{\eta}}{KN} \sum_{k,i} \mathbb{E}[g_i(\mathbf{y}_{i,k-1})].$$

From the definition of $\alpha_{i,k-1}^{r-1}$ and dropping the superscript everywhere we have

$$\Delta \mathbf{x} = -\frac{\tilde{\eta}}{KS} \sum_{k,i \in \mathcal{S}} (g_i(\mathbf{y}_{i,k-1}) + \mathbf{c} - \mathbf{c}_i) \text{ where } \mathbf{c}_i = \frac{1}{K} \sum_k g_i(\alpha_{i,k-1}).$$

Taking norm on both sides and separating mean and variance, we proceed as

$$\begin{aligned} \mathbb{E}\|\Delta \mathbf{x}\|^2 &= \mathbb{E}\left\| -\frac{\tilde{\eta}}{KS} \sum_{k,i \in \mathcal{S}} (g_i(\mathbf{y}_{i,k-1}) - g_i(\alpha_{i,k-1}) + \mathbf{c} - \mathbf{c}_i) \right\|^2 \\ &\leq \mathbb{E}\left\| -\frac{\tilde{\eta}}{KS} \sum_{k,i \in \mathcal{S}} (\nabla f_i(\mathbf{y}_{i,k-1}) + \mathbb{E}[\mathbf{c}] - \mathbb{E}[\mathbf{c}_i]) \right\|^2 + \frac{9\tilde{\eta}^2 \sigma^2}{KS} \\ &\leq \mathbb{E}\left[\frac{\tilde{\eta}^2}{KS} \sum_{k,i \in \mathcal{S}} \left\| \nabla f_i(\mathbf{y}_{i,k-1}) + \mathbb{E}[\mathbf{c}] - \mathbb{E}[\mathbf{c}_i] \right\|^2 \right] + \frac{9\tilde{\eta}^2 \sigma^2}{KS} \\ &= \frac{\tilde{\eta}^2}{KN} \sum_{k,i} \mathbb{E}\left\| (\nabla f_i(\mathbf{y}_{i,k-1}) - \nabla f_i(\mathbf{x})) + (\mathbb{E}[\mathbf{c}] - \nabla f(\mathbf{x})) + \nabla f(\mathbf{x}) - (\mathbb{E}[\mathbf{c}_i] - \nabla f_i(\mathbf{x})) \right\|^2 + \frac{9\tilde{\eta}^2 \sigma^2}{KS} \\ &\leq \frac{4\tilde{\eta}^2}{KN} \sum_{k,i} \mathbb{E}\|\nabla f_i(\mathbf{y}_{i,k-1}) - \nabla f_i(\mathbf{x})\|^2 + \frac{8\tilde{\eta}^2}{KN} \sum_{k,i} \mathbb{E}\|\nabla f_i(\alpha_{i,k-1}) - \nabla f_i(\mathbf{x})\|^2 \\ &\quad + 4\tilde{\eta}^2 \mathbb{E}\|\nabla f(\mathbf{x})\|^2 + \frac{9\tilde{\eta}^2 \sigma^2}{KS} \\ &\leq 4\tilde{\eta}^2 \beta^2 \mathcal{E}_r + 8\beta^2 \tilde{\eta}^2 \Xi_{r-1} + 4\tilde{\eta}^2 \mathbb{E}\|\nabla f(\mathbf{x})\|^2 + \frac{9\tilde{\eta}^2 \sigma^2}{KS}. \end{aligned}$$

In the first inequality, note that the three random variables— $\frac{1}{KS} \sum_{k,i \in \mathcal{S}} g_i(\mathbf{y}_{i,k})$, $\frac{1}{S} \sum_{i \in \mathcal{S}} \mathbf{c}_i$, and \mathbf{c} —may not be independent but each have variance smaller than $\frac{\sigma^2}{KS}$ and so we can ap-

ply Lemma 58. The rest of the inequalities follow from repeated applications of the relaxed triangle inequality, β -Lipschitzness of f_i , and the definition of Ξ_{r-1} (12.21). This proves the second statement. The first statement follows from our expression of $\mathbb{E}_{r-1}[\Delta \mathbf{x}]$ and similar computations. \square

LAG IN THE CONTROL VARIATES. WE NOW ANALYZE THE ‘LAG’ IN THE CONTROL VARIATES DUE TO US SAMPLING ONLY A SMALL SUBSET OF CLIENTS EACH ROUND. BECAUSE WE CANNOT RELY ON CONVEXITY ANYMORE BUT ONLY ON THE LIPSCHITZNESS OF THE GRADIENTS, THE CONTROL-LAG INCREASES FASTER IN THE NON-CONVEX CASE.

Lemma 46. *For updates (12.12)—(12.15) and assumptions A4, A5, the following holds true for any $\tilde{\eta} \leq \frac{1}{24\beta}(\frac{S}{N})^\alpha$ for $\alpha \in [\frac{1}{2}, 1]$ where $\tilde{\eta} := \eta_l \eta_g K$:*

$$\Xi_r \leq (1 - \frac{17S}{36N})\Xi_{r-1} + \frac{1}{48\beta^2}(\frac{S}{N})^{2\alpha-1}\|\nabla f(\mathbf{x}^{r-1})\|^2 + \frac{97}{48}(\frac{S}{N})^{2\alpha-1}\mathcal{E}_r + (\frac{S}{N\beta^2})\frac{\sigma^2}{32KS}.$$

Proof. The proof proceeds similar to that of Lemma 67 except that we cannot rely on convexity. Recall that after round r , the definition of $\boldsymbol{\alpha}_{i,k-1}^r$ (12.20) implies that

$$\mathbb{E}_{\mathcal{S}^r}[\boldsymbol{\alpha}_{i,k-1}^r] = (1 - \frac{S}{N})\boldsymbol{\alpha}_{i,k-1}^{r-1} + \frac{S}{N}\mathbf{y}_{i,k-1}^r.$$

Plugging the above expression in the definition of Ξ_r we get

$$\begin{aligned} \Xi_r &= \frac{1}{KN} \sum_{i,k} \mathbb{E} \|\boldsymbol{\alpha}_{i,k-1}^r - \mathbf{x}^r\|^2 \\ &= \left(1 - \frac{S}{N}\right) \cdot \underbrace{\frac{1}{KN} \sum_i \mathbb{E} \|\boldsymbol{\alpha}_{i,k-1}^{r-1} - \mathbf{x}^r\|^2}_{\mathcal{T}_5} + \underbrace{\frac{S}{N} \cdot \frac{1}{KN} \sum_{k,i} \mathbb{E} \|\mathbf{y}_{i,k-1}^r - \mathbf{x}^r\|^2}_{\mathcal{T}_6}. \end{aligned}$$

We can expand the second term \mathcal{T}_6 with the relaxed triangle inequality to claim

$$\mathcal{T}_6 \leq 2(\mathcal{E}_r + \mathbb{E} \|\Delta \mathbf{x}^r\|^2).$$

We will expand the first term \mathcal{T}_5 to claim for a constant $b \geq 0$ to be chosen later

$$\begin{aligned} \mathcal{T}_5 &= \frac{1}{KN} \sum_i \mathbb{E} (\|\boldsymbol{\alpha}_{i,k-1}^{r-1} - \mathbf{x}^{r-1}\|^2 + \|\Delta \mathbf{x}^r\|^2 + \mathbb{E}_{r-1} \langle \Delta \mathbf{x}^r, \boldsymbol{\alpha}_{i,k-1}^{r-1} - \mathbf{x}^{r-1} \rangle) \\ &\leq \frac{1}{KN} \sum_i \mathbb{E} (\|\boldsymbol{\alpha}_{i,k-1}^{r-1} - \mathbf{x}^{r-1}\|^2 + \|\Delta \mathbf{x}^r\|^2 + \frac{1}{b} \|\mathbb{E}_{r-1}[\Delta \mathbf{x}^r]\|^2 + b \|\boldsymbol{\alpha}_{i,k-1}^{r-1} - \mathbf{x}^{r-1}\|^2) \end{aligned}$$

where we used Young’s inequality which holds for any $b \geq 0$. Combining the bounds for \mathcal{T}_5

and \mathcal{T}_6 ,

$$\begin{aligned}\Xi_r &\leq \left(1 - \frac{S}{N}\right)(1+b)\Xi_{r-1} + 2\frac{S}{N}\mathcal{E}_r + 2\mathbb{E}\|\Delta\mathbf{x}^r\|^2 + \frac{1}{b}\mathbb{E}\|\mathbb{E}_{r-1}[\Delta\mathbf{x}^r]\|^2 \\ &\leq \left(\left(1 - \frac{S}{N}\right)(1+b) + 16\tilde{\eta}^2\beta^2\right)\Xi_{r-1} + \left(\frac{2S}{N} + 8\tilde{\eta}^2\beta^2 + 2\frac{1}{b}\tilde{\eta}^2\beta^2\right)\mathcal{E}_r + \left(8 + 2\frac{1}{b}\right)\tilde{\eta}^2\mathbb{E}\|\nabla f(\mathbf{x})\|^2 + \frac{18\tilde{\eta}^2\sigma^2}{KS}\end{aligned}$$

The last inequality applied Lemma 70. Verify that with choice of $b = \frac{S}{2(N-S)}$, we have $\left(1 - \frac{S}{N}\right)(1+b) \leq \left(1 - \frac{S}{2N}\right)$ and $\frac{1}{b} \leq \frac{2N}{S}$. Plugging these values along with the bound on the step-size $16\beta^2\tilde{\eta}^2 \leq \frac{1}{36}\left(\frac{S}{N}\right)^{2\alpha} \leq \frac{S}{36N}$ completes the lemma. \square

BOUNDING THE DRIFT. WE WILL NEXT BOUND THE CLIENT DRIFT \mathcal{E}_r . FOR THIS, CONVEXITY IS NOT CRUCIAL AND WE WILL RECOVER A VERY SIMILAR RESULT TO LEMMA 68 ONLY USE THE LIPSCHITZNESS OF THE GRADIENT.

Lemma 47. Suppose our step-sizes satisfy $\eta_l \leq \frac{1}{24\beta K\eta_g}$ and f_i satisfies assumptions A4–A5. Then, for any global $\eta_g \geq 1$ we can bound the drift as

$$\frac{5}{3}\beta^2\tilde{\eta}\mathcal{E}_r \leq \frac{5}{3}\beta^3\tilde{\eta}^2\Xi_{r-1} + \frac{\tilde{\eta}}{24\beta\eta_g^2}\mathbb{E}\|\nabla f(\mathbf{x}^{r-1})\|^2 + \frac{\tilde{\eta}^2\beta}{4K\eta_g^2}\sigma^2.$$

Proof. First, observe that if $K = 1$, $\mathcal{E}_r = 0$ since $\mathbf{y}_{i,0} = \mathbf{x}$ for all $i \in [N]$ and that Ξ_{r-1} and the right hand side are both positive. Thus the Lemma is trivially true if $K = 1$ and we will henceforth assume $K \geq 2$. Starting from the update rule (12.12) for $i \in [N]$ and $k \in [K]$

$$\begin{aligned}\mathbb{E}\|\mathbf{y}_{i,k} - \mathbf{x}\|^2 &= \mathbb{E}\|\mathbf{y}_{i,k-1} - \eta_l(g_i(\mathbf{y}_{i,k-1}) + \mathbf{c} - \mathbf{c}_i) - \mathbf{x}\|^2 \\ &\leq \mathbb{E}\|\mathbf{y}_{i,k-1} - \eta_l(\nabla f_i(\mathbf{y}_{i,k-1}) + \mathbf{c} - \mathbf{c}_i) - \mathbf{x}\|^2 + \eta_l^2\sigma^2 \\ &\leq \left(1 + \frac{1}{K-1}\right)\mathbb{E}\|\mathbf{y}_{i,k-1} - \mathbf{x}\|^2 + K\eta_l^2\mathbb{E}\|\nabla f_i(\mathbf{y}_{i,k-1}) + \mathbf{c} - \mathbf{c}_i\|^2 + \eta_l^2\sigma^2 \\ &= \left(1 + \frac{1}{K-1}\right)\mathbb{E}\|\mathbf{y}_{i,k-1} - \mathbf{x}\|^2 + \eta_l^2\sigma^2 \\ &\quad + K\eta_l^2\mathbb{E}\|\nabla f_i(\mathbf{y}_{i,k-1}) - \nabla f_i(\mathbf{x}) + (\mathbf{c} - \nabla f(\mathbf{x})) + \nabla f(\mathbf{x}) - (\mathbf{c}_i - \nabla f_i(\mathbf{x}))\|^2 \\ &\leq \left(1 + \frac{1}{K-1}\right)\mathbb{E}\|\mathbf{y}_{i,k-1} - \mathbf{x}\|^2 + 4K\eta_l^2\mathbb{E}\|\nabla f_i(\mathbf{y}_{i,k-1}) - \nabla f_i(\mathbf{x})\|^2 + \eta_l^2\sigma^2 \\ &\quad + 4K\eta_l^2\mathbb{E}\|\mathbf{c} - \nabla f(\mathbf{x})\|^2 + 4K\eta_l^2\mathbb{E}\|\nabla f(\mathbf{x})\|^2 + 4K\eta_l^2\mathbb{E}\|\mathbf{c}_i - \nabla f_i(\mathbf{x})\|^2 \\ &\leq \left(1 + \frac{1}{K-1} + 4K\beta^2\eta_l^2\right)\mathbb{E}\|\mathbf{y}_{i,k-1} - \mathbf{x}\|^2 + \eta_l^2\sigma^2 + 4K\eta_l^2\mathbb{E}\|\nabla f(\mathbf{x})\|^2 \\ &\quad + 4K\eta_l^2\mathbb{E}\|\mathbf{c} - \nabla f(\mathbf{x})\|^2 + 4K\eta_l^2\mathbb{E}\|\mathbf{c}_i - \nabla f_i(\mathbf{x})\|^2\end{aligned}$$

The inequalities above follow from repeated application of the relaxed triangle inequalities and the β -Lipschitzness of f_i . Averaging the above over i , the definition of $\mathbf{c} = \frac{1}{N}\sum_i \mathbf{c}_i$ and

Ξ_{r-1} (12.21) gives

$$\begin{aligned}
 \frac{1}{N} \sum_i \mathbb{E} \|\mathbf{y}_{i,k} - \mathbf{x}\|^2 &\leq \left(1 + \frac{1}{K-1} + 4K\beta^2\eta_l^2\right) \frac{1}{N} \sum_i \mathbb{E} \|\mathbf{y}_{i,k-1} - \mathbf{x}\|^2 \\
 &\quad + \eta_l^2 \sigma^2 + 4K\eta_l^2 \mathbb{E} \|\nabla f(\mathbf{x})\|^2 + 8K\eta_l^2 \beta^2 \Xi_{r-1} \\
 &\leq \left(\eta_l^2 \sigma^2 + 4K\eta_l^2 \mathbb{E} \|\nabla f(\mathbf{x})\|^2 + 8K\eta_l^2 \beta^2 \Xi_{r-1}\right) \left(\sum_{\tau=0}^{k-1} \left(1 + \frac{1}{K-1} + 4K\beta^2\eta_l^2\right)^\tau\right) \\
 &= \left(\frac{\tilde{\eta}^2 \sigma^2}{K^2 \eta_g^2} + \frac{4\tilde{\eta}^2}{K\eta_g^2} \mathbb{E} \|\nabla f(\mathbf{x})\|^2 + \frac{8\tilde{\eta}^2 \beta^2}{K\eta_g^2} \Xi_{r-1}\right) \left(\sum_{\tau=0}^{k-1} \left(1 + \frac{1}{K-1} + \frac{4\beta^2 \tilde{\eta}^2}{K\eta_g^2}\right)^\tau\right) \\
 &\leq \left(\frac{\tilde{\eta} \sigma^2}{24\beta K^2 \eta_g^2} + \frac{1}{144\beta^2 K \eta_g^2} \mathbb{E} \|\nabla f(\mathbf{x})\|^2 + \frac{\tilde{\eta} \beta}{3K\eta_g^2} \Xi_{r-1}\right) 3K.
 \end{aligned}$$

The last inequality used the bound on the step-size $\beta\tilde{\eta} \leq \frac{1}{24}$. Averaging over k and multiplying both sides by $\frac{5}{3}\beta^2\tilde{\eta}$ yields the lemma statement. \square

PROGRESS MADE IN EACH ROUND. GIVEN THAT WE CAN BOUND ALL SOURCES OF ERROR, WE CAN FINALLY PROVE THE PROGRESS MADE IN EACH ROUND.

Lemma 48. *Suppose the updates (12.12)–(12.15) satisfy assumptions A4–A5. For any effective step-size $\tilde{\eta} := K\eta_g\eta_l$ satisfying $\tilde{\eta} \leq \frac{1}{24\beta} \left(\frac{S}{N}\right)^{\frac{2}{3}}$,*

$$\left(\mathbb{E}[f(\mathbf{x}^r)] + 12\beta^3\tilde{\eta}^2 \frac{N}{S} \Xi_r\right) \leq \left(\mathbb{E}[f(\mathbf{x}^{r-1})] + 12\beta^3\tilde{\eta}^2 \frac{N}{S} \Xi_{r-1}\right) + \frac{5\beta\tilde{\eta}^2\sigma^2}{KS} \left(1 + \frac{S}{\eta_g^2}\right) - \frac{\tilde{\eta}}{14} \mathbb{E} \|\nabla f(\mathbf{x}^{r-1})\|^2.$$

Proof. Starting from the smoothness of f and taking conditional expectation gives

$$\mathbb{E}_{r-1}[f(\mathbf{x} + \Delta\mathbf{x})] \leq f(\mathbf{x}) + \langle \nabla f(\mathbf{x}), \mathbb{E}_{r-1}[\Delta\mathbf{x}] \rangle + \frac{\beta}{2} \mathbb{E}_{r-1} \|\Delta\mathbf{x}\|^2.$$

We as usual dropped the superscript everywhere. Recall that the server update can be written as

$$\Delta\mathbf{x} = -\frac{\tilde{\eta}}{KS} \sum_{k,i \in \mathcal{S}} (g_i(\mathbf{y}_{i,k-1}) + \mathbf{c} - \mathbf{c}_i), \text{ and } \mathbb{E}_{\mathcal{S}}[\Delta\mathbf{x}] = -\frac{\tilde{\eta}}{KN} \sum_{k,i} g_i(\mathbf{y}_{i,k-1}).$$

Substituting this in the previous inequality and applying Lemma 70 to bound $\mathbb{E}[\|\Delta \mathbf{x}\|^2]$ gives

$$\begin{aligned}
 \mathbb{E}[f(\mathbf{x} + \Delta \mathbf{x})] - f(\mathbf{x}) &\leq -\frac{\tilde{\eta}}{KN} \sum_{k,i} \langle \nabla f(\mathbf{x}), \mathbb{E}[\nabla f_i(\mathbf{y}_{i,k-1})] \rangle + \frac{\beta}{2} \mathbb{E}[\|\Delta \mathbf{x}\|^2] \\
 &\leq -\frac{\tilde{\eta}}{KN} \sum_{k,i} \langle \nabla f(\mathbf{x}), \mathbb{E}[\nabla f_i(\mathbf{y}_{i,k-1})] \rangle + \\
 &\quad 2\tilde{\eta}^2 \beta^3 \mathcal{E}_r + 4\tilde{\eta}^2 \beta^3 \Xi_{r-1} + 2\beta \tilde{\eta}^2 \mathbb{E}[\|\nabla f(\mathbf{x})\|^2] + \frac{9\beta \tilde{\eta}^2 \sigma^2}{2KS} \\
 &\leq -\frac{\tilde{\eta}}{2} \|\nabla f(\mathbf{x})\|^2 + \frac{\tilde{\eta}}{2} \sum_{i,k} \mathbb{E} \left\| \frac{1}{KN} \sum \nabla f_i(\mathbf{y}_{i,k-1}) - \nabla f(\mathbf{x}) \right\|^2 + \\
 &\quad 2\tilde{\eta}^2 \beta^3 \mathcal{E}_r + 4\tilde{\eta}^2 \beta^3 \Xi_{r-1} + 2\beta \tilde{\eta}^2 \mathbb{E}[\|\nabla f(\mathbf{x})\|^2] + \frac{9\beta \tilde{\eta}^2 \sigma^2}{2KS} \\
 &\leq -\frac{\tilde{\eta}}{2} \|\nabla f(\mathbf{x})\|^2 + \frac{\tilde{\eta}}{2KN} \sum_{i,k} \mathbb{E} \left\| \nabla f_i(\mathbf{y}_{i,k-1}) - \nabla f_i(\mathbf{x}) \right\|^2 + \\
 &\quad 2\tilde{\eta}^2 \beta^3 \mathcal{E}_r + 4\tilde{\eta}^2 \beta^3 \Xi_{r-1} + 2\beta \tilde{\eta}^2 \mathbb{E}[\|\nabla f(\mathbf{x})\|^2] + \frac{9\beta \tilde{\eta}^2 \sigma^2}{2KS} \\
 &\leq -(\frac{\tilde{\eta}}{2} - 2\beta \tilde{\eta}^2) \|\nabla f(\mathbf{x})\|^2 + (\frac{\tilde{\eta}}{2} + 2\beta \tilde{\eta}^2) \beta^2 \mathcal{E}_r + 4\beta^3 \tilde{\eta}^2 \Xi_{r-1} + \frac{9\beta \tilde{\eta}^2 \sigma^2}{2KS}.
 \end{aligned}$$

The third inequality follows from the observation that $-ab = \frac{1}{2}((b-a)^2 - a^2) - \frac{1}{2}b^2 \leq \frac{1}{2}((b-a)^2 - a^2)$ for any $a, b \in \mathbb{R}$, and the last from the β -Lipschitzness of f_i . Now we use Lemma 71 to bound Ξ_r as

$$\begin{aligned}
 12\beta^3 \tilde{\eta}^2 \frac{N}{S} \Xi_r &\leq 12\beta^3 \tilde{\eta}^2 \frac{N}{S} \left((1 - \frac{17S}{36N}) \Xi_{r-1} + \frac{1}{48\beta^2} (\frac{S}{N})^{2\alpha-1} \|\nabla f(\mathbf{x}^{r-1})\|^2 + \frac{97}{48} (\frac{S}{N})^{2\alpha-1} \mathcal{E}_r + (\frac{S}{N\beta^2}) \frac{\sigma^2}{32KS} \right) \\
 &= 12\beta^3 \tilde{\eta}^2 \frac{N}{S} \Xi_{r-1} - \frac{17}{3} \beta^3 \tilde{\eta}^2 \Xi_{r-1} + \frac{1}{4} \beta \tilde{\eta}^2 (\frac{N}{S})^{2-2\alpha} \|\nabla f(\mathbf{x})\|^2 + \frac{97}{4} \beta^3 \tilde{\eta}^2 (\frac{N}{S})^{2-2\alpha} \mathcal{E}_r + \frac{3\beta \tilde{\eta}^2 \sigma^2}{8KS}.
 \end{aligned}$$

Also recall that Lemma 72 states that

$$\frac{5}{3} \beta^2 \tilde{\eta} \mathcal{E}_r \leq \frac{5}{3} \beta^3 \tilde{\eta}^2 \Xi_{r-1} + \frac{\tilde{\eta}}{24\eta_g^2} \mathbb{E}[\|\nabla f(\mathbf{x}^{r-1})\|^2] + \frac{\tilde{\eta}^2 \beta}{4K\eta_g^2} \sigma^2.$$

Adding these bounds on Ξ_r and \mathcal{E}_r to that of $\mathbb{E}[f(\mathbf{x} + \Delta \mathbf{x})]$ gives

$$\begin{aligned}
 (\mathbb{E}[f(\mathbf{x} + \Delta \mathbf{x})] + 12\beta^3 \tilde{\eta}^2 \frac{N}{S} \Xi_r) &\leq (\mathbb{E}[f(\mathbf{x})] + 12\beta^3 \tilde{\eta}^2 \frac{N}{S} \Xi_{r-1}) + (\frac{5}{3} - \frac{17}{3}) \beta^3 \tilde{\eta}^2 \Xi_{r-1} \\
 &\quad - (\frac{\tilde{\eta}}{2} - 2\beta \tilde{\eta}^2 - \frac{1}{4} \beta \tilde{\eta}^2 (\frac{N}{S})^{2-2\alpha}) \|\nabla f(\mathbf{x})\|^2 + (\frac{\tilde{\eta}}{2} - \frac{5\tilde{\eta}}{3} + 2\beta \tilde{\eta}^2 + \frac{97}{4} \beta \tilde{\eta}^2 (\frac{N}{S})^{2-2\alpha}) \beta^2 \mathcal{E}_r + \frac{39\beta \tilde{\eta}^2 \sigma^2}{8KS} (1 + \frac{S}{\eta_g^2}).
 \end{aligned}$$

By our choice of $\alpha = \frac{2}{3}$ and plugging in the bound on step-size $\beta \tilde{\eta} (\frac{N}{S})^{2-2\alpha} \leq \frac{1}{24}$ proves the lemma. \square

THE **NON-CONVEX RATE** OF CONVERGENCE NOW FOLLOWS BY UNROLLING THE RECURSION IN LEMMA 73 AND SELECTING AN APPROPRIATE STEP-SIZE $\tilde{\eta}$ AS IN LEMMA 56. FINALLY NOTE THAT IF WE INITIALIZE $\mathbf{c}_i^0 = \mathbf{g}_i(\mathbf{x}^0)$ THEN WE HAVE $\Xi_0 = 0$.

11.6 Usefulness of local steps (Theorem IX)

LET US STATE OUR RATES OF CONVERGENCE FOR SCAFFOLD WHICH INTERPOLATES BETWEEN IDENTICAL AND COMPLETELY HETEROGENEOUS CLIENTS. IN THIS SECTION, WE ALWAYS SET $\eta_g = 1$ AND ASSUME ALL CLIENTS PARTICIPATE ($S = N$).

Theorem XXIX. *Suppose that the functions $\{f_i\}$ are quadratic and satisfy assumptions A4, A5 and additionally A2. Then, for global step-size $\eta_g = 1$ in each of the following cases, there exist probabilities $\{p_k^r\}$ and local step-size η_l such that the output (12.23) of SCAFFOLD WHEN RUN WITH NO CLIENT SAMPLING ($S = N$) USING UPDATE (12.22) SATISFIES:*

- **STRONGLY CONVEX:** f_i SATISFIES (A3) FOR $\mu > 0$, $\eta_l \leq \min(\frac{1}{10\beta}, \frac{1}{22\delta K}, \frac{1}{10\mu K})$, $R \geq \max(\frac{20\beta}{\mu}, \frac{44\delta K + 20\mu K}{\mu}, 20K)$ THEN

$$\mathbb{E}[\|\nabla f(\bar{\mathbf{x}}^R)\|^2] \leq \tilde{\mathcal{O}}\left(\frac{\beta\sigma^2}{\mu RKN} + \mu D^2 \exp\left(-\frac{\mu}{20\beta + 44\delta K + 20\mu K} RK\right)\right).$$

- **GENERAL CONVEX:** f SATISFIES $\nabla^2 f \succeq -\delta I$, $\eta_l \leq \min(\frac{1}{10\beta}, \frac{1}{22\delta K})$, AND $R \geq 1$, THEN

$$\mathbb{E}[\|\nabla f(\bar{\mathbf{x}}^R)\|^2] \leq \mathcal{O}\left(\frac{\sigma\sqrt{\beta(f(\mathbf{x}_0) - f^*)}}{\sqrt{RKN}} + \frac{(\beta + \delta K)(f(\mathbf{x}_0) - f^*)}{RK}\right).$$

NOTE THAT IF $\delta = 0$, WE MATCH (UP TO ACCELERATION) THE LOWER BOUND IN (WOODWORTH ET AL., 2018). WHILE CERTAINLY $\delta = 0$ WHEN THE FUNCTIONS ARE IDENTICAL AS STUDIED IN (WOODWORTH ET AL., 2018), OUR UPPER-BOUND IS SIGNIFICANTLY STRONGER SINCE IT IS POSSIBLE THAT $\delta = 0$ EVEN FOR HIGHLY HETEROGENEOUS FUNCTIONS. FOR EXAMPLE, OBJECTIVE PERTURBATION (CHAUDHURI ET AL., 2011; KIFER ET AL., 2012) IS AN OPTIMAL MECHANISM TO ACHIEVE DIFFERENTIAL PRIVACY FOR SMOOTH CONVEX OBJECTIVES (BASSILY ET AL., 2014). INTUITIVELY, OBJECTIVE PERTURBATION RELIES ON MASKING EACH CLIENT'S GRADIENTS BY ADDING A LARGE RANDOM LINEAR TERM TO THE OBJECTIVE FUNCTION. IN SUCH A CASE, WE WOULD HAVE HIGH GRADIENT DISSIMILARITY BUT NO HESSIAN DISSIMILARITY.

OUR NON-CONVEX CONVERGENCE RATES ARE THE FIRST OF THEIR KIND AS FAR AS WE ARE AWARE—NO PREVIOUS WORK SHOWS HOW ONE CAN TAKE ADVANTAGE OF SIMILARITY FOR NON-CONVEX FUNCTIONS. HOWEVER, WE SHOULD NOTE THAT NON-CONVEX QUADRATICS DO NOT HAVE A GLOBAL LOWER-BOUND ON THE FUNCTION VALUE f^* . WE WILL INSTEAD ASSUME THAT f^* ALMOST SURELY LOWER-BOUNDS THE VALUE OF $f(\mathbf{x}^R)$, IMPLICITLY ASSUMING THAT THE ITERATES REMAIN BOUNDED.

OUTLINE. IN THE REST OF THIS SECTION, WE WILL FOCUS ON PROVING THEOREM XXXIII. WE WILL SHOW HOW TO BOUND VARIANCE IN LEMMA 77, BOUND THE AMOUNT OF DRIFT IN LEMMA 76, AND SHOW PROGRESS MADE IN ONE STEP IN LEMMA 78. IN ALL OF THESE WE DO NOT USE CONVEXITY, BUT STRONGLY RELY ON THE FUNCTIONS BEING QUADRATICS. THEN WE COMBINE THESE TO DERIVE THE PROGRESS MADE BY THE SERVER IN ONE ROUND—FOR THIS WE NEED *weak*-CONVEXITY TO ARGUE THAT AVERAGING THE PARAMETERS DOES NOT HURT CON-

VERGENCE TOO MUCH. AS BEFORE, IT IS STRAIGHT-FORWARD TO DERIVE RATES OF CONVERGENCE FROM THE ONE-ROUND PROGRESS USING LEMMAS 55 AND 56.

11.6.1 Additional notation and assumptions

FOR ANY MATRIX M AND VECTOR \mathbf{v} , LET $\|\mathbf{v}\|_M^2 := \mathbf{v}^\top M \mathbf{v}$. SINCE ALL FUNCTIONS IN THIS SECTION ARE QUADRATICS, WE CAN ASSUME W.L.O.G THEY ARE OF THE FOLLOWING FORM:

$$f_i(\mathbf{x}) - f_i(\mathbf{x}_i^*) = \frac{1}{2} \|\mathbf{x} - \mathbf{x}_i^*\|_{A_i}^2 \text{ FOR } i \in [N], \text{ AND } f(\mathbf{x}) = \frac{1}{2} \|\mathbf{x} - \mathbf{x}^*\|_A^2, \text{ FOR ALL } \mathbf{x},$$

FOR SOME $\{\mathbf{x}_i^*\}$ AND \mathbf{x}^* , $A := \frac{1}{N} \sum_{i=1}^N A_i$. WE ALSO ASSUME THAT A IS A SYMMETRIC MATRIX THOUGH THIS REQUIREMENT IS EASILY RELAXED. NOTE THAT THIS IMPLIES $f(\mathbf{x}^*) = 0$ AND THAT $\nabla f_i(\mathbf{x}) = A(\mathbf{x} - \mathbf{x}_i^*)$. IF $\{f_i\}$ ARE ADDITIONALLY CONVEX, WE HAVE THAT \mathbf{x}_i^* IS THE OPTIMUM OF f_i AND \mathbf{x}^* THE OPTIMUM OF f . HOWEVER, THIS IS NOT NECESSARILY TRUE IN GENERAL.

WE WILL ALSO FOCUS ON A SIMPLIFIED VERSION OF SCAFFOLD WHERE IN EACH ROUND r , CLIENT i PERFORMS THE FOLLOWING UPDATE STARTING FROM $\mathbf{y}_{i,0}^r \leftarrow \mathbf{x}^{r-1}$:

$$\begin{aligned} \mathbf{y}_{i,k}^r &= \mathbf{y}_{i,k-1}^r - \eta(g_i(\mathbf{y}_{i,k-1}^r) + \nabla f(\mathbf{x}^{r-1}) - \nabla f_i(\mathbf{x}^{r-1})), \text{ I.E.} \\ \mathbb{E}_{r-1,k-1}[\mathbf{y}_{i,k}^r] &= \mathbf{y}_{i,k-1}^r - \eta A(\mathbf{y}_{i,k-1}^r - \mathbf{x}^*) - \eta(A_i - A)(\mathbf{y}_{i,k-1}^r - \mathbf{x}^{r-1}), \end{aligned} \quad (11.22)$$

WHERE THE SECOND PART IS SPECIALIZED TO QUADRATICS AND THE EXPECTATION IS CONDITIONED OVER EVERYTHING BEFORE CURRENT STEP k OF ROUND r . AT THE END OF EACH ROUND, AS BEFORE, $\mathbf{x}^r = \frac{1}{N} \sum_{i=1}^N \mathbf{y}_{i,K}^r$. THE FINAL OUTPUT OF THE ALGORITHM IS CHOSEN USING PROBABILITIES $\{p_k^r\}$ AS

$$\bar{\mathbf{x}}^R = \mathbf{x}_k^r \text{ WITH PROBABILITY } p_k^r, \text{ WHERE } \mathbf{x}_k^r := \frac{1}{N} \sum_{i=1}^N \mathbf{y}_{i,k}^r. \quad (11.23)$$

NOTE THAT WE ARE NOW POSSIBLY OUTPUTTING ITERATES COMPUTED WITHIN A SINGLE ROUND AND THAT $\mathbf{x}^r = \mathbf{x}_K^r$. BEYOND THIS, THE UPDATE ABOVE DIFFERS FROM OUR USUAL SCAFFOLD IN TWO KEY ASPECTS: A) IT USES GRADIENTS COMPUTED AT \mathbf{x}^{r-1} AS CONTROL VARIATES INSTEAD OF THOSE AT EITHER \mathbf{x}^{r-2} (AS IN OPTION I) OR $\mathbf{y}_{i,k}^{r-1}$ (AS IN OPTION II), AND B) IT USES FULL BATCH GRADIENTS TO COMPUTE ITS CONTROL VARIATES INSTEAD OF STOCHASTIC GRADIENTS. THE FIRST ISSUE IS EASY TO FIX AND OUR PROOF EXTENDS TO USING BOTH OPTION I OR OPTION II USING TECHNIQUES IN SECTION 12.5. THE SECOND ISSUE IS MORE TECHNICAL—USING STOCHASTIC GRADIENTS FOR CONTROL VARIATES COUPLES THE RANDOMNESS ACROSS THE CLIENTS IN MAKING THE LOCAL-UPDATES *biased*. WHILE IT MAY BE POSSIBLE TO GET AROUND THIS (CF. (LEI AND JORDAN, 2017; NGUYEN ET AL., 2017; TRAN-DINH ET AL., 2019)), WE WILL NOT ATTEMPT TO DO SO IN THIS WORK. NOTE THAT IF K LOCAL UPDATE STEPS TYPICALLY REPRESENTS RUNNING MULTIPLE EPOCHS ON EACH CLIENT. HENCE ONE ADDITIONAL EPOCH TO COMPUTE THE CONTROL VARIATE $\nabla f_i(\mathbf{x})$ DOES NOT SIGNIFICANTLY ADD TO THE COST.

FINALLY, WE DEFINE THE FOLLOWING SEQUENCE OF POSITIVE NUMBERS FOR NOTATION CONVENIENCE:

$$\begin{aligned}\xi_{i,k}^r &:= \left(\mathbb{E}_{r-1}[f(\mathbf{y}_{i,k}^r)] - f(\mathbf{x}^*) + \delta(1 + \frac{1}{K})^{K-k} \mathbb{E}_{r-1} \|\mathbf{y}_{i,k}^r - \mathbf{x}^{r-1}\|^2 \right), \text{ AND} \\ \tilde{\xi}_{i,k}^r &:= \left([f(\mathbb{E}_{r-1}[\mathbf{y}_{i,k}^r])] - f(\mathbf{x}^*) + \delta(1 + \frac{1}{K})^{K-k} \mathbb{E}_{r-1,k-1} \|\mathbb{E}_{r-1}[\mathbf{y}_{i,k}^r] - \mathbf{x}^{r-1}\|^2 \right).\end{aligned}$$

OBSERVE THAT FOR $k = 0$, $\xi_{i,0}^r = \tilde{\xi}_{i,0}^r = f(\mathbf{x}^{r-1}) - f(\mathbf{x}^*)$.

11.6.2 Lemmas tracking errors

EFFECT OF AVERAGING. WE SEE HOW AVERAGING CAN REDUCE VARIANCE. A SIMILAR ARGUMENT WAS USED IN THE SPECIAL CASE OF ONE-SHOT AVERAGING IN (ZHANG ET AL., 2013B).

Lemma 49. Suppose $\{f_i\}$ are quadratic functions and assumption A4 is satisfied. Then let \mathbf{x}_k^r and $\mathbf{y}_{i,k}^r$ be vectors in step k and round r generated using (12.22)–(12.23). Then,

$$\mathbb{E}_{r-1} \|\nabla f(\mathbf{x}_k^r)\|^2 \leq \frac{1}{N} \sum_{i=1}^N \|\nabla f(\mathbb{E}_{r-1}[\mathbf{y}_{i,k}^r])\|^2 + \frac{1}{N^2} \sum_{i=1}^N \mathbb{E}_{r-1} [\|\nabla f(\mathbf{y}_{i,k}^r)\|^2].$$

Proof. Observe that the variables $\{\mathbf{y}_{i,k} - \mathbf{x}\}$ are independent of each other (the only source of randomness is the local gradient computations). The rest of the proof is exactly that of Lemma 58. Dropping superscripts everywhere,

$$\begin{aligned}\mathbb{E}_{r-1} \|A(\mathbf{x}_k^r - \mathbf{x}^*)\|^2 &= \mathbb{E}_{r-1} \left\| \frac{1}{N} \sum_i A(\mathbf{y}_{i,k} - \mathbf{x}^*) \right\|^2 \\ &= \mathbb{E}_{r-1} \left\| \frac{1}{N} \sum_i A(\mathbb{E}_{r-1}[\mathbf{y}_{i,k}] - \mathbf{x}^*) \right\|^2 + \mathbb{E}_{r-1} \left\| \frac{1}{N} \sum_i A(\mathbb{E}_{r-1}[\mathbf{y}_{i,k}] - \mathbf{y}_{i,k}) \right\|^2 \\ &= \mathbb{E}_{r-1} \left\| \frac{1}{N} \sum_i A(\mathbb{E}_{r-1}[\mathbf{y}_{i,k}] - \mathbf{x}^*) \right\|^2 + \frac{1}{N^2} \sum_i \mathbb{E}_{r-1} \|A(\mathbb{E}_{r-1}[\mathbf{y}_{i,k}] - \mathbf{y}_{i,k})\|^2 \\ &= \mathbb{E}_{r-1} \left\| \frac{1}{N} \sum_i A(\mathbb{E}_{r-1}[\mathbf{y}_{i,k}] - \mathbf{x}^*) \right\|^2 + \frac{1}{N^2} \sum_i \mathbb{E}_{r-1} \|A(\mathbf{y}_{i,k} - \mathbf{x}^* - \mathbb{E}_{r-1}[\mathbf{y}_{i,k} - \mathbf{x}^*])\|^2 \\ &\leq \mathbb{E}_{r-1} \left\| \frac{1}{N} \sum_i A(\mathbb{E}_{r-1}[\mathbf{y}_{i,k}] - \mathbf{x}^*) \right\|^2 + \frac{1}{N^2} \sum_i \mathbb{E}_{r-1} \|A(\mathbf{y}_{i,k} - \mathbf{x}^*)\|^2.\end{aligned}$$

The third equality was because $\{\mathbf{y}_{i,k}\}$ are independent of each other conditioned on everything before round r . \square

WE NEXT SEE THE EFFECT OF AVERAGING ON FUNCTION VALUES.

Lemma 50. Suppose that f is δ general-convex, then we have:

$$\frac{1}{N} \sum_{i=1}^n \xi_{i,k}^r \geq \mathbb{E}_{r-1}[f(\mathbf{x}_k^r)] - f(\mathbf{x}^*), \text{ and } \frac{1}{N} \sum_{i=1}^n \tilde{\xi}_{i,k}^r \geq f(\mathbb{E}_{r-1}[\mathbf{x}_k^r]) - f(\mathbf{x}^*).$$

Proof. Since f is δ -general convex, it follows that the function $f(\mathbf{z}) + \delta(1 + \frac{1}{K})^{K-k} \|\mathbf{z} - \mathbf{x}\|_2^2$ is convex in \mathbf{z} for any $k \in [K]$. The lemma now follows directly from using convexity and the definition of $\mathbf{x}_k^r = \frac{1}{N} \mathbf{y}_{i,k}^r$. \square

BOUNDING DRIFT OF ONE CLIENT. WE SEE HOW THE CLIENT DRIFT OF SCAFFOLD DEPENDS ON δ .

Lemma 51. *For the update (12.22), assuming (A2) and that $\{f_i\}$ are quadratics, the following holds for any $\eta \leq \frac{1}{21\delta K}$*

$$\mathbb{E}_{r-1, k-1} \|\mathbf{y}_{i,k}^r - \mathbf{x}^{r-1}\|^2 \leq (1 + \frac{1}{2K}) \|\mathbf{y}_{i,k-1}^r - \mathbf{x}^{r-1}\|^2 + 7K\eta^2 \|\nabla f(\mathbf{y}_{i,k-1}^r)\|^2 + \eta^2 \sigma^2.$$

Proof. Starting from the update step (12.22)

$$\begin{aligned} \mathbb{E}_{r-1, k-1} \|\mathbf{y}_i^+ - \mathbf{x}\|^2 &\leq \|\mathbf{y}_i - \mathbf{x} - \eta A(\mathbf{y}_i - \mathbf{x}^*) - \eta(A_i - A)(\mathbf{y}_i - \mathbf{x})\|^2 + \eta^2 \sigma^2 \\ &\leq (1 + \frac{1}{7(K-1)}) \|(I - \eta(A_i - A))(\mathbf{y}_i - \mathbf{x})\|^2 + 7K\eta^2 \|A(\mathbf{y}_i - \mathbf{x}^*)\|^2 + \eta^2 \sigma^2. \end{aligned}$$

Note that if $K = 1$, then the first inequality directly proves the lemma. For the second inequality, we assumed $K \geq 2$ and then applied our relaxed triangle inequality. By assumption A2, we have the following for $\eta\delta \leq 1$

$$\|(I - \eta(A_i - A))^2\| = \|I - \eta(A_i - A)\|^2 \leq (1 + \eta\delta)^2 \leq 1 + 3\eta\delta.$$

Using the bound on the step-size $\eta \leq \frac{1}{21\delta K}$ gives

$$\mathbb{E}_{r-1, k-1} \|\mathbf{y}_i^+ - \mathbf{x}\|^2 \leq (1 + \frac{1}{7K})(1 + \frac{1}{7(K-1)}) \|\mathbf{y}_i - \mathbf{x}\|^2 + 7K\eta^2 \|A(\mathbf{y}_i - \mathbf{x}^*)\|^2 + \eta^2 \sigma^2$$

Simple computations now give the Lemma statement for all $K \geq 1$. \square

TRACKING THE VARIANCE. WE WILL SEE HOW TO BOUND THE VARIANCE OF THE OUTPUT.

Lemma 52. *Consider the update (12.22) for quadratic $\{f_i\}$ with $\eta \leq \max(\frac{1}{2\delta K}, \frac{1}{\beta})$. Then, if further (A2), (A5) and (A4) are satisfied, we have*

$$\mathbb{E}_{r-1} f(\mathbf{x}^r) \leq f(\mathbb{E}_{r-1}[\mathbf{x}^r]) + 3K\beta \frac{\sigma^2}{N}.$$

Further if $\{f_i\}$ are strongly convex satisfying (A3), we have

$$\mathbb{E}_{r-1} f(\mathbf{x}^r) \leq f(\mathbb{E}_{r-1}[\mathbf{x}^r]) + \beta \frac{\sigma^2}{N} \sum_{k=1}^K (1 - \mu\eta)^{k-1}.$$

Proof. We can rewrite the update step (12.22) as below:

$$\mathbf{y}_{i,k} = \mathbf{y}_{i,k-1} - \eta(A_i(\mathbf{y}_{i,k-1} - \mathbf{x}^*) + (A - A_i)(\mathbf{x} - \mathbf{x}^*)) - \eta\zeta_{i,k},$$

where by the bounded variance assumption A4, $\zeta_{i,k}$ is a random variable satisfying $\mathbb{E}_{k-1,r-1}[\zeta_{i,k}] = 0$ and $\mathbb{E}_{k-1,r-1}\|\zeta_{i,k}\|^2 \leq \sigma^2$. Subtracting \mathbf{x}^\star from both sides and unrolling the recursion gives

$$\begin{aligned} \mathbf{y}_{i,K} - \mathbf{x}^\star &= (I - \eta A_i)(\mathbf{y}_{i,K-1} - \mathbf{x}^\star) - \eta((A - A_i)(\mathbf{x} - \mathbf{x}^\star) + \zeta_{i,K}) \\ &= (I - \eta A_i)^K(\mathbf{x} - \mathbf{x}^\star) - \sum_{k=1}^K \eta(I - \eta A_i)^{k-1}(\zeta_{i,k} + (A - A_i)(\mathbf{x} - \mathbf{x}^\star)). \end{aligned}$$

Similarly, the expected iterate satisfies the same equation without the $\zeta_{i,k}$

$$\mathbb{E}_{r-1}[\mathbf{y}_{i,K}] - \mathbf{x}^\star = (I - \eta A_i)^K(\mathbf{x} - \mathbf{x}^\star) - \sum_{k=1}^K \eta(I - \eta A_i)^{k-1}(A - A_i)(\mathbf{x} - \mathbf{x}^\star).$$

This implies that the difference satisfies

$$\mathbb{E}_{r-1}[\mathbf{y}_{i,K}] - \mathbf{y}_{i,K} = \eta \sum_{k=1}^K (I - \eta A_i)^{k-1} \zeta_{i,k}.$$

We can relate this to the function value as follows:

$$\begin{aligned} \mathbb{E}_{r-1}\|\mathbf{x}_K^r - \mathbf{x}^\star\|_A^2 &= \|\mathbb{E}_{r-1}[\mathbf{x}_K^r] - \mathbf{x}^\star\|_A^2 + \mathbb{E}_{r-1}\|\mathbb{E}_{r-1}[\mathbf{x}_K^r] - \mathbf{x}_K^r\|_A^2 \\ &= \|\mathbb{E}_{r-1}[\mathbf{x}_K^r] - \mathbf{x}^\star\|_A^2 + \mathbb{E}_{r-1}\|\frac{1}{N} \sum_i (\mathbb{E}_{r-1}[\mathbf{y}_{i,K}] - \mathbf{y}_{i,K})\|_A^2 \\ &= \|\mathbb{E}_{r-1}[\mathbf{x}_K^r] - \mathbf{x}^\star\|_A^2 + \eta^2 \mathbb{E}_{r-1}\|\frac{1}{N} \sum_{i,k} (I - \eta A_i)^{k-1} \zeta_{i,k}\|_A^2 \\ &= \|\mathbb{E}_{r-1}[\mathbf{x}_K^r] - \mathbf{x}^\star\|_A^2 + \frac{\eta^2}{N^2} \mathbb{E}_{r-1} \sum_{i,k} \|(I - \eta A_i)^{k-1} \zeta_{i,k}\|_A^2 \\ &\leq \|\mathbb{E}_{r-1}[\mathbf{x}_K^r] - \mathbf{x}^\star\|_A^2 + \frac{\beta \eta^2}{N^2} \mathbb{E}_{r-1} \sum_{i,k} \|(I - \eta A_i)^{k-1} \zeta_{i,k}\|_2^2. \end{aligned}$$

The last inequality used smoothness of f and the one before that relied on the independence of $\zeta_{i,k}$. Now, if f_i is general convex we have for $\eta \leq \frac{1}{2\delta K}$ that $I - \eta A_i \leq (1 + \frac{1}{2K})I$ and hence

$$\|(I - \eta A_i)^{k-1} \zeta_{i,k}\|_2^2 \leq \sigma^2 (1 + \frac{1}{2K})^{2(k-1)} \leq 3\sigma^2.$$

This proves our second statement of the lemma. For strongly convex functions, we have for $\eta \leq \frac{1}{\beta}$,

$$\|(I - \eta A_i)^{k-1} \zeta_{i,k}\|_2^2 \leq \sigma^2 (1 - \eta \mu)^{2(k-1)} \leq \sigma^2 (1 - \eta \mu)^{k-1}.$$

□

11.6.3 Lemmas showing progress

PROGRESS OF ONE CLIENT IN ONE STEP. NOW WE FOCUS ONLY ON A SINGLE CLIENT AND MONITOR THEIR PROGRESS.

Lemma 53. Suppose (A2), (A5) and (A4) hold, and $\{f_i\}$ are quadratics. Then, the following holds for the update (12.22) with $\eta \leq \min(\frac{1}{10\beta}, \frac{1}{22\delta K}, \frac{1}{\mu K})$ with $\mu = 0$ if f is non-convex or

general-convex

$$\begin{aligned}\xi_{i,k}^r &\leq (1 - \frac{\mu\eta}{6})\xi_{i,k-1}^r - \frac{\eta}{6}\mathbb{E}_{r-1}\|\nabla f(\mathbf{y}_{i,k-1}^r)\|^2 + 7\beta\eta^2\sigma^2, \text{ and} \\ \tilde{\xi}_{i,k}^r &\leq (1 - \frac{\mu\eta}{6})\tilde{\xi}_{i,k-1}^r - \frac{\eta}{6}\|\nabla f(\mathbb{E}_{r-1}[\mathbf{y}_{i,k-1}^r])\|^2.\end{aligned}$$

Proof. Recall that $\xi_{i,k}^r \geq 0$ is defined to be

$$\xi_{i,k}^r := \left(\mathbb{E}_{r-1}[f(\mathbf{y}_{i,k}^r)] - f(\mathbf{x}^\star) + \delta(1 + \frac{1}{K})^{K-k}\mathbb{E}_{r-1}\|\mathbf{y}_{i,k}^r - \mathbf{x}^{r-1}\|^2 \right).$$

Let us start from the local update step (12.22) (dropping unnecessary subscripts and superscripts)

$$\begin{aligned}\mathbb{E}_{r-1,k-1}\|\mathbf{y}_i^+ - \mathbf{x}^\star\|_A^2 &\leq \|\mathbf{y}_i - \mathbf{x}^\star\|_A^2 - 2\eta\langle A(\mathbf{y}_i - \mathbf{x}^\star), A(\mathbf{y}_i - \mathbf{x}^\star) \rangle + 2\eta\langle (A - A_i)(\mathbf{y}_i - \mathbf{x}), A(\mathbf{y}_i - \mathbf{x}^\star) \rangle \\ &\quad + \eta^2\|A(\mathbf{y}_i - \mathbf{x}^\star) + (A_i - A)(\mathbf{y}_i - \mathbf{x})\|_A^2 + \beta\eta^2\sigma^2 \\ &\leq \|\mathbf{y}_i - \mathbf{x}^\star\|_A^2 - \frac{3\eta}{2}\|A(\mathbf{y}_i - \mathbf{x}^\star)\|_2^2 + 2\eta\|(A - A_i)(\mathbf{y}_i - \mathbf{x})\|_2^2 \\ &\quad + 2\eta^2\|A(\mathbf{y}_i - \mathbf{x}^\star)\|_A^2 + 2\eta^2\|(A_i - A)(\mathbf{y}_i - \mathbf{x})\|_A^2 + \beta\eta^2\sigma^2 \\ &\leq \|\mathbf{y}_i - \mathbf{x}^\star\|_A^2 - (\frac{3\eta}{2} - 2\eta^2\beta)\|A(\mathbf{y}_i - \mathbf{x}^\star)\|_2^2 + \beta\eta^2\sigma^2 + \delta^2(2\eta^2\beta + 2\eta)\|\mathbf{y}_i - \mathbf{x}\|_2^2 \\ &\leq \|\mathbf{y}_i - \mathbf{x}^\star\|_A^2 - (\frac{3\eta}{2} - 2\eta^2\beta)\|A(\mathbf{y}_i - \mathbf{x}^\star)\|_2^2 + \beta\eta^2\sigma^2 + \frac{\delta}{10K}\|\mathbf{y}_i - \mathbf{x}\|_2^2.\end{aligned}$$

The second to last inequality used that $\|\cdot\|_A^2 \leq \beta\|\cdot\|_2^2$ by (A5) and that $\|(A - A_i)(\cdot)\|_2^2 \leq \delta^2\|\cdot\|_2^2$ by (A2). The final inequality used that $\eta \leq \max(\frac{1}{10\beta}, \frac{1}{22\delta K})$. Now, multiplying Lemma 76 by $\delta(1 + \frac{1}{K})^{K-k} \leq \frac{20\delta}{7}$ we have

$$\begin{aligned}\delta(1 + \frac{1}{K})^{K-k}\mathbb{E}_{r-1,k-1}\|\mathbf{y}_i^+ - \mathbf{x}\|^2 &\leq \delta(1 + \frac{1}{K})^{K-k}(1 + \frac{1}{2K})\|\mathbf{y}_i - \mathbf{x}\|^2 + 20\delta K\eta^2\|A(\mathbf{y}_i - \mathbf{x}^\star)\|^2 + 3\delta\eta^2\sigma^2 \\ &\leq \delta(1 + \frac{1}{K})^{K-k}(1 + \frac{1}{2K} + \frac{1}{10K})\|\mathbf{y}_i - \mathbf{x}\|^2 - \frac{\delta}{10K}\|\mathbf{y}_i - \mathbf{x}\|^2 \\ &\quad + 20\delta K\eta^2\|A(\mathbf{y}_i - \mathbf{x}^\star)\|^2 + 3\delta\eta^2\sigma^2 \\ &\leq (1 - \frac{1}{5K})\delta(1 + \frac{1}{K})^{K-k+1}(1 + \frac{1}{K})\|\mathbf{y}_i - \mathbf{x}\|^2 - \frac{\delta}{10K}\|\mathbf{y}_i - \mathbf{x}\|^2 \\ &\quad + 20\delta K\eta^2\|A(\mathbf{y}_i - \mathbf{x}^\star)\|^2 + 3\delta\eta^2\sigma^2.\end{aligned}$$

Adding this to our previous equation gives the following recursive bound:

$$\begin{aligned}&\left(\mathbb{E}_{r-1,k-1}\|\mathbf{y}_i^+ - \mathbf{x}^\star\|_A^2 + \delta(1 + \frac{1}{K})^{K-k}\mathbb{E}_{r-1,k-1}\|\mathbf{y}_i^+ - \mathbf{x}\|^2 \right) \leq \\ &\left(\|\mathbf{y}_i - \mathbf{x}^\star\|_A^2 + (1 - \frac{1}{5K})\delta(1 + \frac{1}{K})^{K-k+1}\|\mathbf{y}_i - \mathbf{x}\|^2 \right) - (\frac{3\eta}{2} - 2\eta^2\beta - 20\delta K\eta^2)\|A(\mathbf{y}_i - \mathbf{x}^\star)\|_2^2 + (3\delta + \beta)\eta^2\sigma^2\end{aligned}$$

The bound on our step-size $\eta \leq \min(\frac{1}{10\beta}, \frac{1}{22\delta K})$ implies that $\frac{3\eta}{2} - 2\eta^2\beta - 20\delta K\eta^2 \geq \frac{\eta}{3}$ and recall that $\delta \leq 2\beta$. This proves first statement of the lemma for non-strongly convex functions ($\mu = 0$). If additionally f is strongly-convex with $\mu > 0$, we have

$$\eta\|A(\mathbf{y}_i - \mathbf{x}^\star)\|_2^2 \geq \frac{\mu\eta}{2}\|\mathbf{y}_i - \mathbf{x}^\star\|_A^2 + \frac{\eta}{2}\|A(\mathbf{y}_i - \mathbf{x}^\star)\|_2^2.$$

This can be used to tighten the inequality as follows

$$\begin{aligned} & \left(\mathbb{E}_{r-1, k-1} \|\mathbf{y}_i^+ - \mathbf{x}^\star\|_A^2 + \delta(1 + \frac{1}{K})^{K-(k-1)} \mathbb{E}_{r-1, k-1} \|\mathbf{y}_i^+ - \mathbf{x}\|^2 \right) \leq \\ & \left((1 - \frac{\mu\eta}{6}) \|\mathbf{y}_i - \mathbf{x}^\star\|_A^2 + (1 - \frac{1}{5K}) \delta(1 + \frac{1}{K})^{K-k+1} \|\mathbf{y}_i - \mathbf{x}\|^2 \right) - \frac{\eta}{2} \|A(\mathbf{y}_i - \mathbf{x}^\star)\|_2^2 + 7\beta\eta^2\sigma^2 \end{aligned}$$

If $\eta \leq \frac{1}{\mu K}$, then $(1 - \frac{1}{5K}) \leq (1 - \frac{\mu\eta}{6})$ and we have the strongly-convex version of the first statement.

Now for the second statement, recall that $\tilde{\xi}_{i,k}^r \geq 0$ was defined to be

$$\tilde{\xi}_{i,k}^r := \left([f(\mathbb{E}_{r-1}[\mathbf{y}_{i,k}^r])] - f(\mathbf{x}^\star) + \delta(1 + \frac{1}{K})^{K-k} \mathbb{E}_{r-1} \|\mathbb{E}_{r-1}[\mathbf{y}_{i,k}^r] - \mathbf{x}^{r-1}\|^2 \right).$$

Observe that for quadratics, $\mathbb{E}_{r-1}[\nabla f(\mathbf{x})] = \nabla f(\mathbb{E}_{r-1}[\mathbf{x}])$. This implies that the analysis of $\tilde{\xi}_{i,k}^r$ is essentially of a deterministic process with $\sigma = 0$, proving the second statement. It is also straightforward to repeat exactly the above argument to formally verify the second statement. \square

SERVER PROGRESS IN ONE ROUND. NOW WE COMBINE THE PROGRESS MADE BY EACH CLIENT IN ONE STEP TO CALCULATE THE SERVER PROGRESS.

Lemma 54. Suppose (A2), (A5) and (A4) hold, and $\{f_i\}$ are quadratics. Then, the following holds for the update (12.22) with $\eta \leq \min(\frac{1}{10\beta}, \frac{1}{21\delta K}, \frac{1}{10\mu K})$ and weights $w_k := (1 - \frac{\mu\eta}{6})^{1-k}$:

$$\frac{\eta}{6} \sum_{k=1}^K w_k \mathbb{E}_{r-1} \|\nabla f(\mathbf{x}_k^r)\|^2 \leq (f(\mathbb{E}_{r-2}[\mathbf{x}^{r-1}]) - f^\star) - w_K (f(\mathbb{E}_{r-1}[\mathbf{x}^r]) - f^\star) + \sum_{k=1}^K w_k 8\eta \frac{\sigma^2}{N}.$$

Set $\mu = 0$ if $\{f_i\}$ s are not strongly-convex (is only general-convex).

Proof. Let us do the non-convex (and general convex) case first. By summing over Lemma 78 we have

$$\frac{\eta}{6} \sum_{k=1}^K \mathbb{E}_{r-1} \|\nabla f(\mathbf{y}_{i,k})\|^2 \leq \xi_{i,0}^r - \xi_{i,K}^r + 7K\beta\eta^2\sigma^2.$$

A similar result holds with $\sigma = 0$ for $\mathbb{E}_{r-1}[\mathbf{y}_{i,k}]$. Now, using Lemma 74 we have that

$$\frac{\eta}{6} \sum_{k=1}^K \mathbb{E}_{r-1} \|\nabla f(\mathbf{x}_k^r)\|^2 \leq \underbrace{\frac{1}{N} \sum_{i=1}^N (\tilde{\xi}_{i,0}^r + \frac{1}{N} \xi_{i,0})}_{=: \theta_+^r} - \underbrace{\frac{1}{N} \sum_{i=1}^N (\tilde{\xi}_{i,K}^r + \frac{1}{N} \xi_{i,K})}_{=: \theta_-^r} + 7K\beta\eta^2\frac{\sigma^2}{N}.$$

Using Lemma 77, we have that

$$\theta_+^r = (1 + \frac{1}{N})(f(\mathbf{x}^{r-1}) - f(\mathbf{x}^\star)) \leq f(\mathbb{E}_{r-1}[\mathbf{x}^r]) + \frac{1}{N} \mathbb{E} f(\mathbf{x}^r) - (1 + \frac{1}{N})f(\mathbf{x}^\star) + 3K\beta\frac{\sigma^2}{N}.$$

Further, by Lemma 75, we have that

$$\theta_-^r \geq f(\mathbb{E}_{r-1}[\mathbf{x}^r]) + \frac{1}{N}f(\mathbf{x}^r) - (1 + \frac{1}{N})f(\mathbf{x}^\star).$$

Combining the above gives:

$$\frac{\eta}{6} \sum_{k=1}^K \mathbb{E}_{r-1} \|\nabla f(\mathbf{x}_k^r)\|^2 \leq f(\mathbb{E}_{r-2}[\mathbf{x}^{r-1}]) - f(\mathbb{E}_{r-1}[\mathbf{x}^r]) + 10\beta K \frac{\sigma^2}{N}.$$

proving the second part of the Lemma for weights $w_k = 1$. The proof of strongly convex follows a very similar argument. Unrolling Lemma 78 using weights $w_k := (1 - \frac{\mu\eta}{6})^{1-k}$ gives

$$\frac{\eta}{6} \sum_{k=1}^K w_k \mathbb{E}_{r-1} \|\nabla f(\mathbf{x}_k^r)\|^2 \leq \theta_+^r - w_K \theta_-^r + \sum_{k=1}^K w_k 7\eta \frac{\sigma^2}{N}.$$

As in the general-convex case, we can use Lemmas 75, 74 and 77 to prove that

$$\frac{\eta}{6} \sum_{k=1}^K w_k \mathbb{E}_{r-1} \|\nabla f(\mathbf{x}_k^r)\|^2 \leq (f(\mathbb{E}_{r-2}[\mathbf{x}^{r-1}]) - f^\star) - w_K (f(\mathbb{E}_{r-1}[\mathbf{x}^r]) - f^\star) + \sum_{k=1}^K w_k 8\eta \frac{\sigma^2}{N}.$$

□

DERIVING FINAL RATES. THE PROOF OF THEOREM XXXIII FOLLOWS BY APPROPRIATELY UNROLLING LEMMA 79. FOR GENERAL-CONVEX FUNCTIONS, WE CAN SIMPLY USE LEMMA 56 WITH THE PROBABILITIES SET AS $p_k^r = \frac{1}{KR}$. FOR STRONGLY-CONVEX FUNCTIONS, WE USE $p_k^r \propto (1 - \frac{\mu\eta}{6})^{1-rk}$ AND FOLLOW THE COMPUTATIONS IN LEMMA 55.

12 Appendix for Mime

12.1 Related work and significance

FEDERATED LEARNING. AS STATED EARLIER, FEDERATED LEARNING INVOLVES LEARNING A CENTRALIZED MODEL FROM DISTRIBUTED CLIENT DATA. THIS CENTRALIZED MODEL BENEFITS FROM ALL CLIENT DATA AND CAN OFTEN RESULT IN A BENEFICIAL PERFORMANCE E.G. IN INCLUDING NEXT WORD PREDICTION (HARD ET AL., 2018; YANG ET AL., 2018), EMOJI PREDICTION (RAMASWAMY ET AL., 2019), DECODER MODELS (CHEN ET AL., 2019B), VOCABULARY ESTIMATION (CHEN ET AL., 2019A), LOW LATENCY VEHICLE-TO-VEHICLE COMMUNICATION (SAMARAKOON ET AL., 2018), AND PREDICTIVE MODELS IN HEALTH (BRISIMI ET AL., 2018). NEVERTHELESS, FEDERATED LEARNING RAISES SEVERAL TYPES OF ISSUES AND HAS BEEN THE TOPIC OF MULTIPLE RESEARCH EFFORTS STUDYING THE ISSUES OF GENERALIZATION AND FAIRNESS (MOHRI ET AL., 2019; LI ET AL., 2019B), THE DESIGN OF MORE EFFICIENT COMMUNICATION STRATEGIES (KONEČNÝ ET AL., 2016; SURESH ET AL., 2017; STICH ET AL., 2018; KARIMIREDDY ET AL., 2019; BASU ET AL., 2019), THE STUDY OF LOWER BOUNDS (WOODWORTH ET AL., 2018), DIFFERENTIAL PRIVACY GUARANTEES (AGARWAL ET AL., 2018), SECURITY (BONAWITZ ET AL., 2017), ETC. WE REFER TO KAIROUZ ET AL. (2019) FOR AN IN-DEPTH SURVEY OF THIS AREA.

CONVERGENCE OF FEDAVG FOR IDENTICAL CLIENTS, FEDAVG COINCIDES WITH PARALLEL SGD ANALYZED BY (ZINKEVICH ET AL., 2010) WHO PROVED ASYMPTOTIC CONVERGENCE. STICH (2019A) AND, MORE RECENTLY STICH AND KARIMIREDDY (2019); PATEL AND DIEULEVEUT (2019); KHALED ET AL. (2020), GAVE A SHARPER ANALYSIS OF THE SAME METHOD, UNDER THE NAME OF LOCAL SGD, ALSO FOR IDENTICAL FUNCTIONS. HOWEVER, THERE STILL REMAINS A GAP BETWEEN THEIR UPPER BOUNDS AND THE LOWER BOUND OF WOODWORTH ET AL. (2018). THE ANALYSIS OF FEDAVG FOR HETEROGENEOUS CLIENTS IS MORE DELICATE DUE TO THE AFORE-MENTIONED CLIENT-DRIFT, FIRST EMPIRICALLY OBSERVED BY ZHAO ET AL. (2018). SEVERAL ANALYSES BOUND THIS DRIFT BY ASSUMING BOUNDED GRADIENTS (WANG ET AL., 2019; YU ET AL., 2019B), OR VIEW IT AS ADDITIONAL NOISE (KHALED ET AL., 2020), OR ASSUME THAT THE CLIENT OPTIMA ARE ϵ -CLOSE (LI ET AL., 2018B; HADDADPOUR AND MAHDAVI,

2019). In a concurrent work, (LIANG ET AL., 2019) PROPOSE TO USE VARIANCE REDUCTION TO DEAL WITH CLIENT HETEROGENEITY BUT STILL SHOW RATES SLOWER THAN SGD. WE SUMMARIZE THE COMMUNICATION COMPLEXITIES OF DIFFERENT METHODS FOR HETEROGENEOUS CLIENTS IN TABLE 4.2.

VARIANCE REDUCTION. THE USE OF *control variates* IS A CLASSICAL TECHNIQUE TO REDUCE VARIANCE IN MONTE CARLO SAMPLING METHODS (CF. (GLASSERMAN, 2013)). IN OPTIMIZATION, THEY WERE USED FOR FINITE-SUM MINIMIZATION BY SVRG (JOHNSON AND ZHANG, 2013; ZHANG ET AL., 2013A) AND THEN IN SAGA (DEFAZIO ET AL., 2014) TO SIMPLIFY THE LINEARLY CONVERGENT METHOD SAG (SCHMIDT ET AL., 2017). NUMEROUS VARIATIONS AND EXTENSIONS OF THE TECHNIQUE ARE STUDIED IN (HANZELY AND RICHTÁRIK, 2019). STARTING FROM (REDDI ET AL., 2016A), CONTROL VARIATES HAVE ALSO FREQUENTLY BEEN USED TO REDUCE VARIANCE IN FINITE-SUM NON-CONVEX SETTINGS (REDDI ET AL., 2016C; NGUYEN ET AL., 2018; FANG ET AL., 2018; TRAN-DINH ET AL., 2019). FURTHER, THEY ARE USED TO OBTAIN LINEARLY CONVERGING DECENTRALIZED ALGORITHMS UNDER THE GUISE OF ‘GRADIENT-TRACKING’ IN (SHI ET AL., 2015; NEDICH ET AL., 2016) AND FOR GRADIENT COMPRESSION AS ‘COMPRESSED-DIFFERENCES’ IN (MISHCHENKO ET AL., 2019). OUR METHOD CAN BE VIEWED AS SEEKING TO REMOVE THE ‘CLIENT-VARIANCE’ IN THE GRADIENTS ACROSS THE CLIENTS, THOUGH THERE STILL REMAINS ADDITIONAL STOCHASTICITY AS IN (KULUNCHAKOV AND MAIRAL, 2019), WHICH IS IMPORTANT IN DEEP LEARNING (DEFAZIO AND BOTTOU, 2019).

DISTRIBUTED OPTIMIZATION. THE PROBLEM OF CLIENT-DRIFT WE DESCRIBED IS A COMMON PHENOMENON IN DISTRIBUTED OPTIMIZATION. IN FACT, CLASSIC TECHNIQUES SUCH AS ADMM MITIGATE THIS DRIFT, THOUGH THEY ARE NOT APPLICABLE IN FEDERATED LEARNING. FOR WELL STRUCTURED CONVEX PROBLEMS, CoCoA (SMITH ET AL., 2018) AND ITS EXTENSIONS (KARIMIREDDY ET AL., 2018C) USE THE DUAL VARIABLE AS THE CONTROL VARIATES, ENABLING FLEXIBLE DISTRIBUTED METHODS. THIS CAN ALSO BE EXTENDED TO INCLUDE SECOND ORDER INFORMATION (DÜNNER ET AL., 2018; KARIMIREDDY ET AL., 2018B). DANE BY (SHAMIR ET AL., 2014) OBTAIN A CLOSELY RELATED PRIMAL ONLY ALGORITHM, WHICH WAS LATER ACCELERATED BY REDDI ET AL. (2016B) AND RECENTLY EXTENDED TO FEDERATED LEARNING (LI ET AL., 2020). SCAFFOLD CAN BE VIEWED AS AN IMPROVED VERSION OF DANE WHERE A FIXED NUMBER OF (STOCHASTIC) GRADIENT STEPS ARE USED IN PLACE OF A PROXIMAL POINT UPDATE. IN A SIMILAR SPIRIT, DISTRIBUTED VARIANCE REDUCTION TECHNIQUES HAVE BEEN PROPOSED FOR THE FINITE-SUM CASE (LEE ET AL., 2015; KONEČNÝ ET AL., 2016; CEN ET AL., 2019). HOWEVER, THESE METHODS ARE RESTRICTED TO FINITE-SUMS AND ARE NOT APPLICABLE TO THE STOCHASTIC SETTING STUDIED HERE.

12.2 Technicalities

WE EXAMINE SOME ADDITIONAL DEFINITIONS AND INTRODUCE SOME TECHNICAL LEMMAS.

12.2.1 Additional definitions

WE MAKE PRECISE A FEW DEFINITIONS AND EXPLAIN SOME OF THEIR IMPLICATIONS.

(A3) f_i IS μ -CONVEX FOR $\mu \geq 0$ AND SATISFIES:

$$\langle \nabla f_i(\mathbf{x}), \mathbf{y} - \mathbf{x} \rangle \leq -\left(f_i(\mathbf{x}) - f_i(\mathbf{y}) + \frac{\mu}{2} \|\mathbf{x} - \mathbf{y}\|^2\right), \text{ FOR ANY } i, \mathbf{x}, \mathbf{y}.$$

HERE, WE ALLOW THAT $\mu = 0$ (WE REFER TO THIS CASE AS THE GENERAL CONVEX CASE AS OPPOSED TO STRONGLY CONVEX). IT IS ALSO POSSIBLE TO GENERALIZE ALL PROOFS HERE TO THE WEAKER NOTION OF PL-STRONG CONVEXITY (KARIMI ET AL., 2016).

(A4) $g_i(\mathbf{x}) := \nabla f_i(\mathbf{x}; \zeta_i)$ IS UNBIASED STOCHASTIC GRADIENT OF f_i WITH **BOUNDED VARIANCE**

$$\mathbb{E}_{\zeta_i} [\|g_i(\mathbf{x}) - \nabla f_i(\mathbf{x})\|^2] \leq \sigma^2, \text{ FOR ANY } i, \mathbf{x}.$$

NOTE THAT (A4) ONLY BOUNDS THE VARIANCE WITHIN THE SAME CLIENT, BUT NOT THE VARIANCE ACROSS THE CLIENTS.

(A5) $\{f_i\}$ ARE β -SMOOTH AND SATISFY:

$$\|\nabla f_i(\mathbf{x}) - \nabla f_i(\mathbf{y})\| \leq \beta \|\mathbf{x} - \mathbf{y}\|, \text{ FOR ANY } i, \mathbf{x}, \mathbf{y}. \quad (12.1)$$

THE ASSUMPTION (A5) ALSO IMPLIES THE FOLLOWING QUADRATIC UPPER BOUND ON f_i

$$f_i(\mathbf{y}) \leq f_i(\mathbf{x}) + \langle \nabla f_i(\mathbf{x}), \mathbf{y} - \mathbf{x} \rangle + \frac{\beta}{2} \|\mathbf{y} - \mathbf{x}\|^2. \quad (12.2)$$

IF ADDITIONALLY THE FUNCTION $\{f_i\}$ ARE CONVEX AND \mathbf{x}^* IS AN OPTIMUM OF f , (A5) IMPLIES (VIA NESTEROV (2018), THEOREM 2.1.5)

$$\frac{1}{2\beta N} \sum_{i=1}^N \|\nabla f_i(\mathbf{x}) - \nabla f_i(\mathbf{x}^*)\|^2 \leq f(\mathbf{x}) - f^*. \quad (12.3)$$

FURTHER, IF f_i IS TWICE-DIFFERENTIABLE, (A5) IMPLIES THAT $\|\nabla^2 f_i(\mathbf{x})\| \leq \beta$ FOR ANY \mathbf{x} .

12.2.2 Some technical lemmas

NOW WE COVER SOME TECHNICAL LEMMAS WHICH ARE USEFUL FOR COMPUTATIONS LATER ON. THE TWO LEMMAS BELOW ARE USEFUL TO UNROLL RECURSIONS AND DERIVE CONVERGENCE RATES. THE FIRST ONE IS A SLIGHTLY IMPROVED (AND SIMPLIFIED) VERSION OF (STICH, 2019B, THEOREM 2). IT IS STRAIGHTFORWARD TO REMOVE THE ADDITIONAL LOGARITHMIC TERMS IF WE USE A VARYING STEP-SIZE (KULUNCHAKOV AND MAIRAL, 2019, LEMMA 13).

Lemma 55 (linear convergence rate). *For every non-negative sequence $\{d_{r-1}\}_{r \geq 1}$ and any parameters $\mu > 0$, $\eta_{\max} \in (0, 1/\mu]$, $c \geq 0$, $R \geq \frac{1}{2\eta_{\max}\mu}$, there exists a constant step-size $\eta \leq \eta_{\max}$ and*

weights $w_r := (1 - \mu\eta)^{1-r}$ such that for $W_R := \sum_{r=1}^{R+1} w_r$,

$$\Psi_R := \frac{1}{W_R} \sum_{r=1}^{R+1} \left(\frac{w_r}{\eta} (1 - \mu\eta) d_{r-1} - \frac{w_r}{\eta} d_r + c\eta w_r \right) = \tilde{\mathcal{O}} \left(\mu d_0 \exp(-\mu\eta_{\max} R) + \frac{c}{\mu R} \right).$$

Proof. By substituting the value of w_r , we observe that we end up with a telescoping sum and estimate

$$\Psi_R = \frac{1}{\eta W_R} \sum_{r=1}^{R+1} (w_{r-1} d_{r-1} - w_r d_r) + \frac{c\eta}{W_R} \sum_{r=1}^{R+1} w_r \leq \frac{d_0}{\eta W_R} + c\eta.$$

When $R \geq \frac{1}{2\mu\eta}$, $(1 - \mu\eta)^R \leq \exp(-\mu\eta R) \leq \frac{2}{3}$. For such an R , we can lower bound ηW_R using

$$\eta W_R = \eta(1 - \mu\eta)^{-R} \sum_{r=0}^R (1 - \mu\eta)^r = \eta(1 - \mu\eta)^{-R} \frac{1 - (1 - \mu\eta)^{R+1}}{\mu\eta} \geq (1 - \mu\eta)^{-R} \frac{1}{3\mu}.$$

This proves that for all $R \geq \frac{1}{2\mu\eta}$,

$$\Psi_R \leq 3\mu d_0 (1 - \mu\eta)^R + c\eta \leq 3\mu d_0 \exp(-\mu\eta R) + c\eta.$$

The lemma now follows by carefully tuning η . Consider the following two cases depending on the magnitude of R and η_{\max} :

- Suppose $\frac{1}{2\mu R} \leq \eta_{\max} \leq \frac{\log(\max(1, \mu^2 R d_0 / c))}{\mu R}$. Then we can choose $\eta = \eta_{\max}$,

$$\Psi_R \leq 3\mu d_0 \exp[-\mu\eta_{\max} R] + c\eta_{\max} \leq 3\mu d_0 \exp[-\mu\eta_{\max} R] + \tilde{\mathcal{O}}\left(\frac{c}{\mu R}\right).$$

- Instead if $\eta_{\max} > \frac{\log(\max(1, \mu^2 R d_0 / c))}{\mu R}$, we pick $\eta = \frac{\log(\max(1, \mu^2 R d_0 / c))}{\mu R}$ to claim that

$$\Psi_R \leq 3\mu d_0 \exp[-\log(\max(1, \mu^2 R d_0 / c))] + \tilde{\mathcal{O}}\left(\frac{c}{\mu R}\right) \leq \tilde{\mathcal{O}}\left(\frac{c}{\mu R}\right).$$

□

THE NEXT LEMMA IS AN EXTENSION OF (STICH AND KARIMIREDDY, 2019, LEMMA 13), (KULUNCHAKOV AND MAIRAL, 2019, LEMMA 13) AND IS USEFUL TO DERIVE CONVERGENCE RATES FOR GENERAL CONVEX FUNCTIONS ($\mu = 0$) AND NON-CONVEX FUNCTIONS.

Lemma 56 (sub-linear convergence rate). *For every non-negative sequence $\{d_{r-1}\}_{r \geq 1}$ and any parameters $\eta_{\max} \geq 0$, $c \geq 0$, $R \geq 0$, there exists a constant step-size $\eta \leq \eta_{\max}$ and weights $w_r = 1$ such that,*

$$\Psi_R := \frac{1}{R+1} \sum_{r=1}^{R+1} \left(\frac{d_{r-1}}{\eta} - \frac{d_r}{\eta} + c_1 \eta + c_2 \eta^2 \right) \leq \frac{d_0}{\eta_{\max}(R+1)} + \frac{2\sqrt{c_1 d_0}}{\sqrt{R+1}} + 2 \left(\frac{d_0}{R+1} \right)^{\frac{2}{3}} c_2^{\frac{1}{3}}.$$

Proof. Unrolling the sum, we can simplify

$$\Psi_R \leq \frac{d_0}{\eta(R+1)} + c_1\eta + c_2\eta^2.$$

Similar to the strongly convex case (Lemma 55), we distinguish the following cases:

- When $R+1 \leq \frac{d_0}{c_1\eta_{\max}^2}$, and $R+1 \leq \frac{d_0}{c_2\eta_{\max}^3}$ we pick $\eta = \eta_{\max}$ to claim

$$\Psi_R \leq \frac{d_0}{\eta_{\max}(R+1)} + c_1\eta_{\max} + c_2\eta_{\max}^2 \leq \frac{d_0}{\eta_{\max}(R+1)} + \frac{\sqrt{c_1 d_0}}{\sqrt{R+1}} + \left(\frac{d_0}{R+1}\right)^{\frac{2}{3}} c_2^{\frac{1}{3}}.$$

- In the other case, we have $\eta_{\max}^2 \geq \frac{d_0}{c_1(R+1)}$ or $\eta_{\max}^3 \geq \frac{d_0}{c_2(R+1)}$. We choose $\eta = \min\left\{\sqrt{\frac{d_0}{c_1(R+1)}}, \sqrt[3]{\frac{d_0}{c_2(R+1)}}\right\}$ to prove

$$\Psi_R \leq \frac{d_0}{\eta(R+1)} + c\eta = \frac{2\sqrt{c_1 d_0}}{\sqrt{R+1}} + 2\sqrt[3]{\frac{d_0^2 c_2}{(R+1)^2}}.$$

□

NEXT, WE STATE A RELAXED TRIANGLE INEQUALITY TRUE FOR THE SQUARED ℓ_2 NORM.

Lemma 57 (relaxed triangle inequality). *Let $\{\mathbf{v}_1, \dots, \mathbf{v}_\tau\}$ be τ vectors in \mathbb{R}^d . Then the following are true:*

1. $\|\mathbf{v}_i + \mathbf{v}_j\|^2 \leq (1+a)\|\mathbf{v}_i\|^2 + (1+\frac{1}{a})\|\mathbf{v}_j\|^2$ for any $a > 0$, and
2. $\|\sum_{i=1}^\tau \mathbf{v}_i\|^2 \leq \tau \sum_{i=1}^\tau \|\mathbf{v}_i\|^2$.

Proof. The proof of the first statement for any $a > 0$ follows from the identity:

$$\|\mathbf{v}_i + \mathbf{v}_j\|^2 = (1+a)\|\mathbf{v}_i\|^2 + (1+\frac{1}{a})\|\mathbf{v}_j\|^2 - \|\sqrt{a}\mathbf{v}_i + \frac{1}{\sqrt{a}}\mathbf{v}_j\|^2.$$

For the second inequality, we use the convexity of $\mathbf{x} \rightarrow \|\mathbf{x}\|^2$ and Jensen's inequality

$$\left\|\frac{1}{\tau} \sum_{i=1}^\tau \mathbf{v}_i\right\|^2 \leq \frac{1}{\tau} \sum_{i=1}^\tau \|\mathbf{v}_i\|^2.$$

□

NEXT WE STATE AN ELEMENTARY LEMMA ABOUT EXPECTATIONS OF NORMS OF RANDOM VECTORS.

Lemma 58 (separating mean and variance). *Let $\{\Xi_1, \dots, \Xi_\tau\}$ be τ random variables in \mathbb{R}^d which are not necessarily independent. First suppose that their mean is $\mathbb{E}[\Xi_i] = \xi_i$ and variance*

is bounded as $\mathbb{E}[\|\Xi_i - \xi_i\|^2] \leq \sigma^2$. Then, the following holds

$$\mathbb{E}[\|\sum_{i=1}^{\tau} \Xi_i\|^2] \leq \|\sum_{i=1}^{\tau} \xi_i\|^2 + \tau^2 \sigma^2.$$

Now instead suppose that their conditional mean is $\mathbb{E}[\Xi_i | \Xi_{i-1}, \dots, \Xi_1] = \xi_i$ i.e. the variables $\{\Xi_i - \xi_i\}$ form a martingale difference sequence, and the variance is bounded by $\mathbb{E}[\|\Xi_i - \xi_i\|^2] \leq \sigma^2$ as before. Then we can show the tighter bound

$$\mathbb{E}[\|\sum_{i=1}^{\tau} \Xi_i\|^2] \leq 2\|\sum_{i=1}^{\tau} \xi_i\|^2 + 2\tau\sigma^2.$$

Proof. For any random variable X , $\mathbb{E}[X^2] = (\mathbb{E}[X - \mathbb{E}[X]])^2 + (\mathbb{E}[X])^2$ implying

$$\mathbb{E}[\|\sum_{i=1}^{\tau} \Xi_i\|^2] = \|\sum_{i=1}^{\tau} \xi_i\|^2 + \mathbb{E}[\|\sum_{i=1}^{\tau} \Xi_i - \xi_i\|^2].$$

Expanding the above expression using relaxed triangle inequality (Lemma 57) proves the first claim:

$$\mathbb{E}[\|\sum_{i=1}^{\tau} \Xi_i - \xi_i\|^2] \leq \tau \sum_{i=1}^{\tau} \mathbb{E}[\|\Xi_i - \xi_i\|^2] \leq \tau^2 \sigma^2.$$

For the second statement, ξ_i is not deterministic and depends on Ξ_{i-1}, \dots, Ξ_1 . Hence we have to resort to the cruder relaxed triangle inequality to claim

$$\mathbb{E}[\|\sum_{i=1}^{\tau} \Xi_i\|^2] \leq 2\|\sum_{i=1}^{\tau} \xi_i\|^2 + 2\mathbb{E}[\|\sum_{i=1}^{\tau} \Xi_i - \xi_i\|^2]$$

and then use the tighter expansion of the second term:

$$\mathbb{E}[\|\sum_{i=1}^{\tau} \Xi_i - \xi_i\|^2] = \sum_{i,j} \mathbb{E}[(\Xi_i - \xi_i)^\top (\Xi_j - \xi_j)] = \sum_i \mathbb{E}[\|\Xi_i - \xi_i\|^2] \leq \tau\sigma^2.$$

The cross terms in the above expression have zero mean since $\{\Xi_i - \xi_i\}$ form a martingale difference sequence. \square

12.3 Properties of convex functions

WE NOW STUDY TWO LEMMAS WHICH HOLD FOR ANY SMOOTH AND STRONGLY-CONVEX FUNCTIONS. THE FIRST IS A GENERALIZATION OF THE STANDARD STRONG CONVEXITY INEQUALITY (A3), BUT CAN HANDLE GRADIENTS COMPUTED AT SLIGHTLY PERTURBED POINTS.

Lemma 59 (perturbed strong convexity). *The following holds for any β -smooth and μ -strongly convex function h , and any $\mathbf{x}, \mathbf{y}, \mathbf{z}$ in the domain of h :*

$$\langle \nabla h(\mathbf{x}), \mathbf{z} - \mathbf{y} \rangle \geq h(\mathbf{z}) - h(\mathbf{y}) + \frac{\mu}{4} \|\mathbf{y} - \mathbf{z}\|^2 - \beta \|\mathbf{z} - \mathbf{x}\|^2.$$

Proof. Given any \mathbf{x} , \mathbf{y} , and \mathbf{z} , we get the following two inequalities using smoothness and strong convexity of h :

$$\begin{aligned}\langle \nabla h(\mathbf{x}), \mathbf{z} - \mathbf{x} \rangle &\geq h(\mathbf{z}) - h(\mathbf{x}) - \frac{\beta}{2} \|\mathbf{z} - \mathbf{x}\|^2 \\ \langle \nabla h(\mathbf{x}), \mathbf{x} - \mathbf{y} \rangle &\geq h(\mathbf{x}) - h(\mathbf{y}) + \frac{\mu}{2} \|\mathbf{y} - \mathbf{x}\|^2.\end{aligned}$$

Further, applying the relaxed triangle inequality gives

$$\frac{\mu}{2} \|\mathbf{y} - \mathbf{x}\|^2 \geq \frac{\mu}{4} \|\mathbf{y} - \mathbf{z}\|^2 - \frac{\mu}{2} \|\mathbf{x} - \mathbf{z}\|^2.$$

Combining all the inequalities together we have

$$\langle \nabla h(\mathbf{x}), \mathbf{z} - \mathbf{y} \rangle \geq h(\mathbf{z}) - h(\mathbf{y}) + \frac{\mu}{4} \|\mathbf{y} - \mathbf{z}\|^2 - \frac{\beta + \mu}{2} \|\mathbf{z} - \mathbf{x}\|^2.$$

The lemma follows since $\beta \geq \mu$. □

HERE, WE SEE THAT A GRADIENT STEP IS A CONTRACTIVE OPERATOR.

Lemma 60 (contractive mapping). *For any β -smooth and μ -strongly convex function h , points \mathbf{x}, \mathbf{y} in the domain of h , and step-size $\eta \leq \frac{1}{\beta}$, the following is true*

$$\|\mathbf{x} - \eta \nabla h(\mathbf{x}) - \mathbf{y} + \eta \nabla h(\mathbf{y})\|^2 \leq (1 - \mu\eta) \|\mathbf{x} - \mathbf{y}\|^2.$$

Proof.

$$\begin{aligned}\|\mathbf{x} - \eta \nabla h(\mathbf{x}) - \mathbf{y} + \eta \nabla h(\mathbf{y})\|^2 &= \|\mathbf{x} - \mathbf{y}\|^2 + \eta^2 \|\nabla h(\mathbf{x}) - \nabla h(\mathbf{y})\|^2 - 2\eta \langle \nabla h(\mathbf{x}) - \nabla h(\mathbf{y}), \mathbf{x} - \mathbf{y} \rangle \\ &\stackrel{(A5)}{\leq} \|\mathbf{x} - \mathbf{y}\|^2 + (\eta^2 \beta - 2\eta) \langle \nabla h(\mathbf{x}) - \nabla h(\mathbf{y}), \mathbf{x} - \mathbf{y} \rangle.\end{aligned}$$

Recall our bound on the step-size $\eta \leq \frac{1}{\beta}$ which implies that $(\eta^2 \beta - 2\eta) \leq -\eta$. Finally, apply the μ -strong convexity of h to get

$$-\eta \langle \nabla h(\mathbf{x}) - \nabla h(\mathbf{y}), \mathbf{x} - \mathbf{y} \rangle \leq -\eta\mu \|\mathbf{x} - \mathbf{y}\|^2.$$

□

12.4 Convergence of FEDAVG

WE OUTLINE THE FEDAVG METHOD IN ALGORITHM 18. IN ROUND r WE SAMPLE $\mathcal{S}^r \subseteq [N]$ CLIENTS WITH $|\mathcal{S}^r| = S$ AND THEN PERFORM THE FOLLOWING UPDATES:

- STARTING FROM THE SHARED GLOBAL PARAMETERS $\mathbf{y}_{i,0}^r = \mathbf{x}^{r-1}$, WE UPDATE THE LOCAL PA-

Algorithm 18 FEDAVG: Federated Averaging

```

1: server input: initial  $\mathbf{x}$ , and global step-size  $\eta_g$ 
2: client  $i$ 's input: local step-size  $\eta_l$ 
3: for each round  $r = 1, \dots, R$  do
4:   sample clients  $\mathcal{S} \subseteq \{1, \dots, N\}$ 
5:   communicate  $\mathbf{x}$  to all clients  $i \in \mathcal{S}$ 
6:   for each client  $i \in \mathcal{S}$  in parallel do
7:     initialize local model  $\mathbf{y}_i \leftarrow \mathbf{x}$ 
8:     for  $k = 1, \dots, K$  do
9:       compute mini-batch gradient  $g_i(\mathbf{y}_i)$ 
10:       $\mathbf{y}_i \leftarrow \mathbf{y}_i - \eta_l g_i(\mathbf{y}_i)$ 
11:    end for
12:    communicate  $\Delta \mathbf{y}_i \leftarrow \mathbf{y}_i - \mathbf{x}$ 
13:  end for
14:   $\Delta \mathbf{x} \leftarrow \frac{1}{|\mathcal{S}|} \sum_{i \in \mathcal{S}} \Delta \mathbf{y}_i$ 
15:   $\mathbf{x} \leftarrow \mathbf{x} + \eta_g \Delta \mathbf{x}$ 
16: end for

```

PARAMETERS FOR $k \in [K]$

$$\mathbf{y}_{i,k}^r = \mathbf{y}_{i,k-1}^r - \eta_l g_i(\mathbf{y}_{i,k-1}^r). \quad (12.4)$$

- COMPUTE THE NEW GLOBAL PARAMETERS USING ONLY UPDATES FROM THE CLIENTS $i \in \mathcal{S}^r$ AND A GLOBAL STEP-SIZE η_g :

$$\mathbf{x}^r = \mathbf{x}^{r-1} + \frac{\eta_g}{S} \sum_{i \in \mathcal{S}^r} (\mathbf{y}_{i,K}^r - \mathbf{x}^{r-1}). \quad (12.5)$$

FINALLY, FOR SOME WEIGHTS $\{w_r\}$, WE OUTPUT

$$\bar{\mathbf{x}}^R = \mathbf{x}^{r-1} \text{ WITH PROBABILITY } \frac{w_r}{\sum_{\tau} w_{\tau}} \text{ FOR } r \in \{1, \dots, R+1\}. \quad (12.6)$$

12.4.1 Bounding heterogeneity

RECALL OUR BOUND ON THE GRADIENT DISSIMILARITY:

$$\frac{1}{N} \sum_{i=1}^N \|\nabla f_i(\mathbf{x})\|^2 \leq G^2 + B^2 \|\nabla f(\mathbf{x})\|^2. \quad (12.7)$$

IF $\{f_i\}$ ARE CONVEX, WE CAN RELAX THE ASSUMPTION TO

$$\frac{1}{N} \sum_{i=1}^N \|\nabla f_i(\mathbf{x})\|^2 \leq G^2 + 2\beta B^2 (f(\mathbf{x}) - f^*). \quad (12.8)$$

WE DEFINED TWO VARIANTS OF THE BOUNDS ON THE HETEROGENEITY DEPENDING OF WHETHER THE FUNCTIONS ARE CONVEX OR NOT. SUPPOSE THAT THE FUNCTIONS f IS INDEED CONVEX AS

IN (A3) AND β -SMOOTH AS IN (A5), THEN IT IS STRAIGHTFORWARD TO SEE THAT (12.7) IMPLIES (12.8). THUS FOR CONVEX FUNCTIONS, (A1) IS MILDLY WEAKER. SUPPOSE THAT THE FUNCTIONS $\{f_1, \dots, f_N\}$ ARE CONVEX AND β -SMOOTH. THEN (12.8) IS SATISFIED WITH $B^2 = 2$ SINCE

$$\begin{aligned} \frac{1}{N} \sum_{i=1}^N \|\nabla f_i(\mathbf{x})\|^2 &\leq \frac{2}{N} \sum_{i=1}^N \|\nabla f_i(\mathbf{x}^*)\|^2 + \frac{2}{N} \sum_{i=1}^N \|\nabla f_i(\mathbf{x}) - \nabla f_i(\mathbf{x}^*)\|^2 \\ &\stackrel{(12.3)}{\leq} \underbrace{\frac{2}{N} \sum_{i=1}^N \|\nabla f_i(\mathbf{x}^*)\|^2}_{=: \sigma_f^2} + 4\beta(f(\mathbf{x}) - f^*). \end{aligned}$$

THUS, (G, B) -BGD (12.8) IS EQUIVALENT TO THE HETEROGENEITY ASSUMPTION OF (MISHCHENKO ET AL., 2019) WITH $G^2 = \sigma_f^2$. INSTEAD, IF WE HAVE THE STRONGER ASSUMPTION (A1) BUT THE FUNCTIONS ARE POSSIBLY NON-CONVEX, THEN $G = \epsilon$ CORRESPONDS TO THE **LOCAL DISSIMILARITY** DEFINED IN (LI ET AL., 2018B). NOTE THAT ASSUMING G IS NEGLIGIBLE IS QUITE STRONG AND CORRESPONDS TO THE STRONG-GROWTH CONDITION (VASWANI ET AL., 2019).

12.4.2 Rates of convergence (Theorem VI)

WE FIRST RESTATE THEOREM VI WITH SOME ADDITIONAL DETAILS AND THEN SEE ITS PROOF.

Theorem XXX. *Suppose that the functions $\{f_i\}$ satisfies assumptions A4, A5, and A1. Then, in each of the following cases, there exist weights $\{w_r\}$ and local step-sizes η_l such that for any $\eta_g \geq 1$ the output of FEDAVG (12.6) $\bar{\mathbf{x}}^R$ satisfies*

- **Strongly convex:** f_i satisfies (A3) for $\mu > 0$, $\eta_l \leq \frac{1}{8(1+B^2)\beta K \eta_g}$, $R \geq \frac{8(1+B^2)\beta}{\mu}$ then

$$\mathbb{E}[f(\bar{\mathbf{x}}^R)] - f(\mathbf{x}^*) \leq \tilde{\mathcal{O}}\left(\frac{M^2}{\mu R K S} + \frac{\beta G^2}{\mu^2 R^2} + \mu D^2 \exp\left(-\frac{\mu}{16(1+B^2)\beta} R\right)\right),$$

- **General convex:** f_i satisfies (A3) for $\mu = 0$, $\eta_l \leq \frac{1}{(1+B^2)8\beta K \eta_g}$, $R \geq 1$ then

$$\mathbb{E}[f(\bar{\mathbf{x}}^R)] - f(\mathbf{x}^*) \leq \mathcal{O}\left(\frac{MD}{\sqrt{R K S}} + \frac{D^{4/3}(\beta G^2)^{1/3}}{(R+1)^{2/3}} + \frac{B^2 \beta D^2}{R}\right),$$

- **Non-convex:** f_i satisfies (A1) and $\eta_l \leq \frac{1}{(1+B^2)8\beta K \eta_g}$, then

$$\mathbb{E}[\|\nabla f(\bar{\mathbf{x}}^R)\|^2] \leq \mathcal{O}\left(\frac{\beta M \sqrt{F}}{\sqrt{R K S}} + \frac{F^{2/3}(\beta G^2)^{1/3}}{(R+1)^{2/3}} + \frac{B^2 \beta F}{R}\right),$$

where $M^2 := \sigma^2(1 + \frac{S}{\eta_g^2}) + K(1 - \frac{S}{N})G^2$, $D := \|\mathbf{x}^0 - \mathbf{x}^*\|$, and $F := f(\mathbf{x}^0) - f^*$.

12.4.3 Proof of convergence

WE WILL ONLY PROVE THE RATE OF CONVERGENCE FOR CONVEX FUNCTIONS HERE. THE CORRESPONDING RATES FOR NON-CONVEX FUNCTIONS ARE EASY TO DERIVE FOLLOWING THE TECHNIQUES IN THE REST OF THE PAPER.

Lemma 61. (one round progress) Suppose our functions satisfies assumptions (A1) and (A3)–(A5). For any step-size satisfying $\eta_l \leq \frac{1}{(1+B^2)8\beta K\eta_g}$ and effective step-size $\tilde{\eta} := K\eta_g\eta_l$, the updates of FEDAVG satisfy

$$\mathbb{E}\|\mathbf{x}^r - \mathbf{x}^\star\|^2 \leq (1 - \frac{\mu\tilde{\eta}}{2})\mathbb{E}\|\mathbf{x}^{r-1} - \mathbf{x}^\star\|^2 + (\frac{1}{KS})\tilde{\eta}^2\sigma^2 + (1 - \frac{S}{N})\frac{4\tilde{\eta}^2}{S}G^2 - \tilde{\eta}(\mathbb{E}[f(\mathbf{x}^{r-1})] - f(\mathbf{x}^\star)) + 3\beta\tilde{\eta}\mathcal{E}_r,$$

where \mathcal{E}_r is the drift caused by the local updates on the clients defined to be

$$\mathcal{E}_r := \frac{1}{KN} \sum_{k=1}^K \sum_{i=1}^N \mathbb{E}_r[\|\mathbf{y}_{i,k}^r - \mathbf{x}^{r-1}\|^2].$$

Proof. We start with the observation that the updates (12.4) and (12.5) imply that the server update in round r can be written as below (dropping the superscripts everywhere)

$$\Delta\mathbf{x} = -\frac{\tilde{\eta}}{KS} \sum_{k,i \in \mathcal{S}} g_i(\mathbf{y}_{i,k-1}) \text{ and } \mathbb{E}[\Delta\mathbf{x}] = -\frac{\tilde{\eta}}{KN} \sum_{k,i} \mathbb{E}[\nabla f_i(\mathbf{y}_{i,k-1})].$$

We adopt the convention that summations are always over $k \in [K]$ or $i \in [N]$ unless otherwise stated. Expanding using above observing, we proceed as¹

$$\begin{aligned} \mathbb{E}_{r-1}\|\mathbf{x} + \Delta\mathbf{x} - \mathbf{x}^\star\|^2 &= \|\mathbf{x} - \mathbf{x}^\star\|^2 - \frac{2\tilde{\eta}}{KN} \sum_{k,i} \langle \nabla f_i(\mathbf{y}_{i,k-1}), \mathbf{x} - \mathbf{x}^\star \rangle + \tilde{\eta}^2 \mathbb{E}_{r-1} \left\| \frac{1}{KS} \sum_{k,i \in \mathcal{S}} g_i(\mathbf{y}_{i,k-1}) \right\|^2 \\ &\stackrel{\text{Lem. 58}}{\leq} \underbrace{\|\mathbf{x} - \mathbf{x}^\star\|^2 - \frac{2\tilde{\eta}}{KN} \sum_{k,i} \langle \nabla f_i(\mathbf{y}_{i,k-1}), \mathbf{x} - \mathbf{x}^\star \rangle}_{\mathcal{A}_1} \\ &\quad + \underbrace{\tilde{\eta}^2 \mathbb{E}_{r-1} \left\| \frac{1}{KS} \sum_{k,i \in \mathcal{S}} \nabla f_i(\mathbf{y}_{i,k-1}) \right\|^2}_{\mathcal{A}_2} + \frac{\tilde{\eta}^2\sigma^2}{KS}. \end{aligned}$$

We can directly apply Lemma 59 with $h = f_i$, $\mathbf{x} = \mathbf{y}_{i,k-1}$, $\mathbf{y} = \mathbf{x}^\star$, and $\mathbf{z} = \mathbf{x}$ to the first term \mathcal{A}_1

$$\begin{aligned} \mathcal{A}_1 &= \frac{2\tilde{\eta}}{KN} \sum_{k,i} \langle \nabla f_i(\mathbf{y}_{i,k-1}), \mathbf{x}^\star - \mathbf{x} \rangle \\ &\leq \frac{2\tilde{\eta}}{KN} \sum_{k,i} \left(f_i(\mathbf{x}^\star) - f_i(\mathbf{x}) + \beta\|\mathbf{y}_{i,k-1} - \mathbf{x}\|^2 - \frac{\mu}{4}\|\mathbf{x} - \mathbf{x}^\star\|^2 \right) \\ &= -2\tilde{\eta} \left(f(\mathbf{x}) - f(\mathbf{x}^\star) + \frac{\mu}{4}\|\mathbf{x} - \mathbf{x}^\star\|^2 \right) + 2\beta\tilde{\eta}\mathcal{E}. \end{aligned}$$

¹We use the notation $\mathbb{E}_{r-1}[\cdot]$ to mean conditioned on filtration r i.e. on all the randomness generated prior to round r .

For the second term \mathcal{A}_2 , we repeatedly apply the relaxed triangle inequality (Lemma 58)

$$\begin{aligned}
 \mathcal{A}_2 &= \tilde{\eta}^2 \mathbb{E}_{r-1} \left\| \frac{1}{KS} \sum_{k,i \in \mathcal{S}} \nabla f_i(\mathbf{y}_{i,k-1}) - \nabla f_i(\mathbf{x}) + \nabla f_i(\mathbf{x}) \right\|^2 \\
 &\leq 2\tilde{\eta}^2 \mathbb{E}_{r-1} \left\| \frac{1}{KS} \sum_{k,i \in \mathcal{S}} \nabla f_i(\mathbf{y}_{i,k-1}) - \nabla f_i(\mathbf{x}) \right\|^2 + 2\tilde{\eta}^2 \mathbb{E}_{r-1} \left\| \frac{1}{S} \sum_{i \in \mathcal{S}} \nabla f_i(\mathbf{x}) \right\|^2 \\
 &\leq \frac{2\tilde{\eta}^2}{KN} \sum_{i,k} \mathbb{E}_{r-1} \|\nabla f_i(\mathbf{y}_{i,k-1}) - \nabla f_i(\mathbf{x})\|^2 + 2\tilde{\eta}^2 \mathbb{E}_{r-1} \left\| \frac{1}{S} \sum_{i \in \mathcal{S}} \nabla f_i(\mathbf{x}) - \nabla f(\mathbf{x}) + \nabla f(\mathbf{x}) \right\|^2 \\
 &\leq \frac{2\tilde{\eta}^2 \beta^2}{KN} \sum_{i,k} \mathbb{E}_{r-1} \|\mathbf{y}_{i,k-1} - \mathbf{x}\|^2 + 2\tilde{\eta}^2 \|\nabla f(\mathbf{x})\|^2 + (1 - \frac{S}{N}) 4\tilde{\eta}^2 \frac{1}{SN} \sum_i \|\nabla f_i(\mathbf{x})\|^2 \\
 &\leq 2\tilde{\eta}^2 \beta^2 \mathcal{E} + 8\tilde{\eta}^2 \beta(B^2 + 1)(f(\mathbf{x}) - f(\mathbf{x}^*)) + (1 - \frac{S}{N}) \frac{4\tilde{\eta}^2}{S} G^2
 \end{aligned}$$

The last step used Assumption (G, B) -BGD assumption (12.8) that $\frac{1}{N} \sum_{i=1}^N \|\nabla f_i(\mathbf{x})\|^2 \leq G^2 + 2\beta B^2(f(\mathbf{x}) - f(\mathbf{x}^*))$. The extra $(1 - \frac{S}{N})$ improvement we get is due to sampling the functions $\{f_i\}$ *without* replacement. Plugging back the bounds on \mathcal{A}_1 and \mathcal{A}_2 ,

$$\begin{aligned}
 \mathbb{E}_{r-1} \|\mathbf{x} + \Delta \mathbf{x} - \mathbf{x}^*\|^2 &\leq (1 - \frac{\mu\tilde{\eta}}{2}) \|\mathbf{x} - \mathbf{x}^*\|^2 - (2\tilde{\eta} - 8\beta\tilde{\eta}^2(B^2 + 1))(f(\mathbf{x}) - f(\mathbf{x}^*)) \\
 &\quad + (1 + \tilde{\eta}\beta) 2\beta\tilde{\eta}\mathcal{E} + \frac{1}{KS} \tilde{\eta}^2 \sigma^2 + (1 - \frac{S}{N}) \frac{4\tilde{\eta}^2}{S} G^2.
 \end{aligned}$$

The lemma now follows by observing that $8\beta\tilde{\eta}(B^2 + 1) \leq 1$ and that $B \geq 0$. \square

Lemma 62. (bounded drift) Suppose our functions satisfies assumptions (A1) and (A3)–(A5). Then the updates of FEDAVG for any step-size satisfying $\eta_l \leq \frac{1}{(1+B^2)8\beta K\eta_g}$ have bounded drift:

$$3\beta\tilde{\eta}\mathcal{E}_r \leq \frac{2\tilde{\eta}}{3} (\mathbb{E}[f(\mathbf{x}^{r-1})]) - f(\mathbf{x}^*) + \frac{\tilde{\eta}^2 \sigma^2}{2K\eta_g^2} + 18\beta\tilde{\eta}^3 G^2.$$

Proof. If $K = 1$, the lemma trivially holds since $\mathbf{y}_{i,0} = \mathbf{x}$ for all $i \in [N]$ and $\mathcal{E}_r = 0$. Assume $K \geq 2$ here on. Recall that the local update made on client i is $\mathbf{y}_{i,k} = \mathbf{y}_{i,k-1} - \eta_l g_i(\mathbf{y}_{i,k-1})$.

Then,

$$\begin{aligned}
 \mathbb{E}\|\mathbf{y}_{i,k} - \mathbf{x}\|^2 &= \mathbb{E}\|\mathbf{y}_{i,k-1} - \mathbf{x} - \eta_l g_i(\mathbf{y}_{i,k-1})\|^2 \\
 &\leq \mathbb{E}\|\mathbf{y}_{i,k-1} - \mathbf{x} - \eta_l \nabla f_i(\mathbf{y}_{i,k-1})\|^2 + \eta_l^2 \sigma^2 \\
 &\leq (1 + \frac{1}{K-1}) \mathbb{E}\|\mathbf{y}_{i,k-1} - \mathbf{x}\|^2 + K\eta_l^2 \|\nabla f_i(\mathbf{y}_{i,k-1})\|^2 + \eta_l^2 \sigma^2 \\
 &= (1 + \frac{1}{K-1}) \mathbb{E}\|\mathbf{y}_{i,k-1} - \mathbf{x}\|^2 + \frac{\tilde{\eta}^2}{\eta_g K} \|\nabla f_i(\mathbf{y}_{i,k-1})\|^2 + \frac{\tilde{\eta}^2 \sigma^2}{K^2 \eta_g^2} \\
 &\leq (1 + \frac{1}{K-1}) \mathbb{E}\|\mathbf{y}_{i,k-1} - \mathbf{x}\|^2 + \frac{2\tilde{\eta}^2}{\eta_g K} \|\nabla f_i(\mathbf{y}_{i,k-1}) - \nabla f_i(\mathbf{x})\|^2 + \frac{2\tilde{\eta}^2}{\eta_g K} \|\nabla f_i(\mathbf{x})\|^2 + \frac{\tilde{\eta}^2 \sigma^2}{K^2 \eta_g^2} \\
 &\leq (1 + \frac{1}{K-1} + \frac{2\tilde{\eta}^2 \beta^2}{\eta_g K}) \mathbb{E}\|\mathbf{y}_{i,k-1} - \mathbf{x}\|^2 + \frac{2\tilde{\eta}^2}{\eta_g K} \|\nabla f_i(\mathbf{x})\|^2 + \frac{\tilde{\eta}^2 \sigma^2}{K^2 \eta_g^2} \\
 &\leq (1 + \frac{2}{(K-1)}) \mathbb{E}\|\mathbf{y}_{i,k-1} - \mathbf{x}\|^2 + \frac{2\tilde{\eta}^2}{\eta_g K} \|\nabla f_i(\mathbf{x})\|^2 + \frac{\tilde{\eta}^2 \sigma^2}{K^2 \eta_g^2}.
 \end{aligned}$$

In the above proof we separated the mean and the variance in the first inequality, then used the relaxed triangle inequality with $a = \frac{1}{K-1}$ in the next inequality. Next equality uses the definition of $\tilde{\eta}$, and the rest follow from the Lipschitzness of the gradient. Unrolling the recursion above,

$$\begin{aligned}
 \mathbb{E}\|\mathbf{y}_{i,k} - \mathbf{x}\|^2 &\leq \sum_{\tau=1}^{k-1} (\frac{2\tilde{\eta}^2}{\eta_g K} \|\nabla f_i(\mathbf{x})\|^2 + \frac{\tilde{\eta}^2 \sigma^2}{K^2 \eta_g^2}) (1 + \frac{2}{(K-1)})^\tau \\
 &\leq (\frac{2\tilde{\eta}^2}{\eta_g K} \|\nabla f_i(\mathbf{x})\|^2 + \frac{\tilde{\eta}^2 \sigma^2}{K^2 \eta_g^2}) 3K.
 \end{aligned}$$

Averaging over i and k , multiplying by $3\beta\tilde{\eta}$ and then using Assumption A1,

$$\begin{aligned}
 3\beta\tilde{\eta}\mathcal{E}_r &\leq \frac{1}{N} \sum_i 18\beta\tilde{\eta}^3 \|\nabla f_i(\mathbf{x})\|^2 + \frac{3\beta\tilde{\eta}^3 \sigma^2}{K\eta_g^2} \\
 &\leq 18\beta\tilde{\eta}^3 G^2 + \frac{3\beta\tilde{\eta}^3 \sigma^2}{K\eta_g^2} + 36\beta^2 \tilde{\eta}^3 B^2 (f(\mathbf{x}) - f(\mathbf{x}^*))
 \end{aligned}$$

The lemma now follows from our assumption that $8(B^2 + 1)\beta\tilde{\eta} \leq 1$. \square

PROOF OF THEOREMS VI, XXX ADDING THE STATEMENTS OF LEMMAS 61 AND 62, WE GET

$$\begin{aligned}
 \mathbb{E}\|\mathbf{x} + \Delta\mathbf{x} - \mathbf{x}^*\|^2 &\leq (1 - \frac{\mu\tilde{\eta}}{2}) \mathbb{E}\|\mathbf{x} - \mathbf{x}^*\|^2 + (\frac{1}{KS}) \tilde{\eta}^2 \sigma^2 + (1 - \frac{S}{N}) \frac{4\tilde{\eta}^2}{S} G^2 - \tilde{\eta}(\mathbb{E}[f(\mathbf{x})] - f(\mathbf{x}^*)) \\
 &\quad + \frac{2\tilde{\eta}}{3} (\mathbb{E}[f(\mathbf{x})] - f(\mathbf{x}^*)) + \frac{\tilde{\eta}^2 \sigma^2}{2K\eta_g^2} + 18\beta\tilde{\eta}^3 G^2 \\
 &= (1 - \frac{\mu\tilde{\eta}}{2}) \mathbb{E}\|\mathbf{x} - \mathbf{x}^*\|^2 - \frac{\tilde{\eta}}{3} (\mathbb{E}[f(\mathbf{x})] - f(\mathbf{x}^*)) + \tilde{\eta}^2 \left(\frac{\sigma^2}{KS} (1 + \frac{S}{\eta_g^2}) + \frac{4G^2}{S} (1 - \frac{S}{N}) + 18\beta\tilde{\eta}G^2 \right).
 \end{aligned}$$

MOVING THE $(f(\mathbf{x}) - f(\mathbf{x}^*))$ TERM AND DIVIDING THROUGHOUT BY $\frac{\tilde{\eta}}{3}$, WE GET THE FOLLOWING BOUND FOR ANY $\tilde{\eta} \leq \frac{1}{8(1+B^2)\beta}$

$$\mathbb{E}[f(\mathbf{x}^{r-1})] - f(\mathbf{x}^*) \leq \frac{3}{\tilde{\eta}}(1 - \frac{\mu\tilde{\eta}}{2})\|\mathbf{x}^{r-1} - \mathbf{x}^*\|^2 - \frac{3}{\tilde{\eta}}\|\mathbf{x}^r - \mathbf{x}^*\|^2 + 3\tilde{\eta}\left(\frac{\sigma^2}{KS}(1 + \frac{S}{\eta_g^2}) + \frac{4G^2}{S}(1 - \frac{S}{N}) + 18\beta\tilde{\eta}G^2\right).$$

IF $\mu = 0$ (GENERAL CONVEX), WE CAN DIRECTLY APPLY LEMMA 56. OTHERWISE, BY AVERAGING USING WEIGHTS $w_r = (1 - \frac{\mu\tilde{\eta}}{2})^{1-r}$ AND USING THE SAME WEIGHTS TO PICK OUTPUT $\bar{\mathbf{x}}^R$, WE CAN SIMPLIFY THE ABOVE RECURSIVE BOUND (SEE PROOF OF LEM. 55) TO PROVE THAT FOR ANY $\tilde{\eta}$ SATISFYING $\frac{1}{\mu R} \leq \tilde{\eta} \leq \frac{1}{8(1+B^2)\beta}$

$$\mathbb{E}[f(\bar{\mathbf{x}}^R)] - f(\mathbf{x}^*) \leq \underbrace{3\|\mathbf{x}^0 - \mathbf{x}^*\|^2}_{=:d} \mu \exp(-\frac{\tilde{\eta}}{2}\mu R) + \underbrace{\tilde{\eta}\left(\frac{2\sigma^2}{KS}(1 + \frac{S}{\eta_g^2}) + \frac{8G^2}{S}(1 - \frac{S}{N})\right)}_{=:c_1} + \tilde{\eta}^2 \underbrace{(36\beta G^2)}_{=:c_2}$$

NOW, THE CHOICE OF $\tilde{\eta} = \min\left\{\frac{\log(\max(1, \mu^2 R d / c_1))}{\mu R}, \frac{1}{(1+B^2)8\beta}\right\}$ YIELDS THE DESIRED RATE. THE PROOF OF THE NON-CONVEX CASE IS VERY SIMILAR AND ALSO RELIES ON LEMMA 56.

12.4.4 Lower bound for FEDAVG (Theorem VII)

WE FIRST FORMALIZE THE CLASS OF ALGORITHMS WE LOOK AT BEFORE PROVING OUT LOWER BOUND.

(A6) WE ASSUME THAT FEDAVG IS RUN WITH $\eta_g = 1$, $K > 1$, AND ARBITRARY POSSIBLY ADAPTIVE POSITIVE STEP-SIZES $\{\eta_1, \dots, \eta_R\}$ ARE USED WITH $\eta_r \leq \frac{1}{\mu}$ AND FIXED WITHIN A ROUND FOR ALL CLIENTS. FURTHER, THE SERVER UPDATE IS A CONVEX COMBINATION OF THE CLIENT UPDATES WITH NON-ADAPTIVE WEIGHTS.

NOTE THAT WE ONLY PROVE THE LOWER BOUND HERE FOR $\eta_g = 1$. IN FACT, BY TAKING η_g INFINITELY LARGE AND SCALING $\eta_l \propto \frac{1}{K\eta_g}$ SUCH THAT THE EFFECTIVE STEP SIZE $\tilde{\eta} = \eta_l \eta_g K$ REMAINS CONSTANT, FEDAVG REDUCES TO THE SIMPLE LARGE BATCH SGD METHOD. HENCE, PROVING A LOWER BOUND FOR ARBITRARY η_g IS NOT POSSIBLE, BUT ALSO IS OF QUESTIONABLE RELEVANCE. FURTHER, NOTE THAT WHEN $\sigma^2 = 0$, THE UPPER BOUND IN THEOREM XXX USES $\eta_g = 1$ AND HENCE THE LOWER BOUND SERVES TO SHOW THAT OUR ANALYSIS IS TIGHT.

BELOW WE STATE A MORE FORMAL VERSION OF THEOREM VII.

Theorem XXXI. *For any positive constants G, μ , there exist μ -strongly convex functions satisfying A1 for which that the output of FEDAVG satisfying A6 has the error for any $r \geq 1$:*

$$f(\mathbf{x}^r) - f(\mathbf{x}^*) \geq \Omega\left(\min(f(\mathbf{x}^0) - f(\mathbf{x}^*), \frac{G^2}{\mu R^2})\right).$$

Proof. Consider the following simple one-dimensional functions for any given μ and G :

$$f_1(x) := \mu x^2 + Gx, \text{ and } f_2(x) := -Gx,$$

with $f(x) = \frac{1}{2}(f_1(x) + f_2(x)) = \frac{\mu}{2}x^2$ and optimum at $x = 0$. Clearly f is μ -strongly convex and further f_1 and f_2 satisfy A1 with $B = 3$. Note that we chose f_2 to be a linear function (not strongly convex) to simplify computations. The calculations made here can be extended with slightly more work for $(\tilde{f}_2 = \frac{\mu}{2}x^2 - Gx)$ (e.g. see Theorem 1 of (Safran and Shamir, 2019)).

Let us start FEDAVG from $x^0 > 0$. A single local update for f_1 and f_2 in round $r \geq 1$ is respectively

$$y_1 = y_1 - \eta_r(2\mu x + G) \text{ and } y_2 = y_2 + \eta_r G.$$

Then, straightforward computations show that the update at the end of round r is of the following form for some averaging weight $\alpha \in [0, 1]$

$$x^r = x^{r-1}((1-\alpha)(1-2\mu\eta_r)^K + \alpha) + \eta_r G \sum_{\tau=0}^{K-1} (\alpha - (1-\alpha)(1-2\mu\eta_r)^\tau).$$

Since α was picked obliviously, we can assume that $\alpha \leq 0.5$. If indeed $\alpha > 0.5$, we can swap the definitions of f_1 and f_2 and the sign of x^0 . With this, we can simplify as

$$\begin{aligned} x^r &\geq x^{r-1} \frac{(1-2\mu\eta_r)^K + 1}{2} + \frac{\eta_r G}{2} \sum_{\tau=0}^{K-1} (1 - (1-2\mu\eta_r)^\tau) \\ &\geq x^{r-1} (1-2\mu\eta_r)^K + \frac{\eta_r G}{2} \sum_{\tau=0}^{K-1} (1 - (1-2\mu\eta_r)^\tau). \end{aligned}$$

Observe that in the above expression, the right hand side is increasing with η_r —this represents the effect of the client drift and increases the error as the step-size increases. The left hand side decreases with η_r —this is the usual convergence observed due to taking gradient steps. The rest of the proof is to show that even with a careful balancing of the two terms, the effect of G cannot be removed. Lemma 63 performs exactly such a computation to prove that for any $r \geq 1$,

$$x^r \geq c \min(x_0, \frac{G}{\mu R}).$$

We finish the proof by noting that $f(x^r) = \frac{\mu}{2}(x^r)^2$. □

Lemma 63. Suppose that for all $r \geq 1$, $\eta_r \leq \frac{1}{\mu}$ and the following is true:

$$x^r \geq x^{r-1} (1-2\mu\eta_r)^K + \frac{\eta_r G}{2} \sum_{\tau=0}^{K-1} (1 - (1-2\mu\eta_r)^\tau).$$

Then, there exists a constant $c > 0$ such that for any sequence of step-sizes $\{\eta^r\}$:

$$x^r \geq c \min(x_0, \frac{G}{\mu R})$$

Proof. Define $\gamma_r = \mu\eta_r R(K-1)$. Such a γ_r exists and is positive since $K \geq 2$. Then, γ_r satisfies

$$(1 - 2\mu\eta_r)^{\frac{K-1}{2}} = \left(1 - \frac{2\gamma_r}{R(K-1)}\right)^{\frac{K-1}{2}} \leq \exp(-\gamma_r/R).$$

We then have

$$\begin{aligned} x^r &\geq x^{r-1}(1 - 2\mu\eta_r)^K + \frac{\eta_r G}{2} \sum_{\tau=0}^{K-1} (1 - (1 - 2\mu\eta_r)^\tau) \\ &\geq x^{r-1}(1 - 2\mu\eta_r)^K + \frac{\eta_r G}{2} \sum_{\tau=(K-1)/2}^{K-1} (1 - (1 - 2\mu\eta_r)^\tau) \\ &\geq x^{r-1}(1 - 2\mu\eta_r)^K + \frac{\gamma_r G}{4\mu} (1 - \exp(-\gamma_r/R)). \end{aligned}$$

The second inequality follows because $\eta_r \leq \frac{1}{\mu}$ implies that $(1 - (1 - 2\mu\eta_r)^\tau)$ is always positive. If $\gamma_r \geq R/8$, then we have a constant $c_1 \in (0, 1/32)$ which satisfies

$$x^r \geq \frac{c_1 G}{\mu}. \quad (12.9)$$

On the other hand, if $\gamma_r < R/8$, we have a tighter inequality

$$(1 - 2\mu\eta_r)^{\frac{K-1}{2}} = \left(1 - \frac{2\gamma_r}{R(K-1)}\right)^{\frac{K-1}{2}} \leq 1 - \frac{\gamma_r}{R},$$

implying that

$$\begin{aligned} x^r &\geq x^{r-1} \left(1 - \frac{2\gamma_r}{R(K-1)}\right)^K + \frac{\gamma_r^2 G}{4R\mu} \\ &\geq x^{r-1} \left(1 - \frac{4\gamma_r}{R}\right) + \frac{\gamma_r^2 G}{4\mu R}. \end{aligned} \quad (12.10)$$

The last step used Bernoulli's inequality and the fact that $K-1 \leq K/2$ for $K \geq 2$. Observe that in the above expression, the right hand side is increasing with γ_r —this represents the effect of the client drift and increases the error as the step-size increases. The left hand side decreases with γ_r —this is the usual convergence observed due to taking gradient steps. The rest of the proof is to show that even with a careful balancing of the two terms, the effect of G cannot be removed.

Suppose that all rounds after $r_0 \geq 0$ have a small step-size i.e. $\gamma_r \leq R/8$ for all $r > r_0$ and hence satisfies (12.10). Then we will prove via induction that

$$x^r \geq \min(c_r x^{r_0}, \frac{G}{256\mu R}), \text{ for constants } c_r := (1 - \frac{1}{2R})^{r-r_0}. \quad (12.11)$$

For $r = r_0$, (12.11) is trivially satisfied. Now for $r > r_0$,

$$\begin{aligned} x^r &\geq x^{r-1} \left(1 - \frac{4\gamma_r}{R}\right) + \frac{\gamma_r^2 G}{4\mu R} \\ &\geq \min\left(x^{r-1} \left(1 - \frac{1}{2R}\right), \frac{G}{256\mu R}\right) \\ &= \min\left(c_r x^{r_0}, \frac{G}{256\mu R}\right). \end{aligned}$$

The first step is because of (12.10) and the last step uses the induction hypothesis. The second step considers two cases for γ_r : either $\gamma_r \leq \frac{1}{8}$ and $(1 - \frac{1}{2R}) \geq (1 - \frac{1}{2R})$, or $\gamma_r^2 \geq \frac{1}{64}$. Finally note that $c^r \geq \frac{1}{2}$ using Bernoulli's inequality. We have hence proved

$$x^R \geq \min\left(\frac{1}{2} x^{r_0}, \frac{G}{256\mu R}\right)$$

Now suppose $\gamma_{r_0} > R/8$. Then (12.9) implies that $x^R \geq \frac{cG}{\mu R}$ for some constant $c > 0$. If instead no such $r_0 \geq 1$ exists, then we can set $r_0 = 0$. Now finally observe that the previous proof did not make any assumption on R , and in fact the inequality stated above holds for all $r \geq 1$. \square

12.5 Convergence of SCAFFOLD

WE FIRST RESTATE THE CONVERGENCE THEOREM MORE FORMALLY, THEN PROVE THE RESULT FOR THE CONVEX CASE, AND THEN FOR NON-CONVEX CASE. THROUGHOUT THE PROOF, WE WILL FOCUS ON THE HARDER OPTION II. THE PROOFS FOR SCAFFOLD WITH OPTION I ARE NEARLY IDENTICAL AND SO WE SKIP THEM.

Theorem XXXII. *Suppose that the functions $\{f_i\}$ satisfies assumptions A4 and A5. Then, in each of the following cases, there exist weights $\{w_r\}$ and local step-sizes η_l such that for any $\eta_g \geq 1$ the output (12.16) of SCAFFOLD SATISFIES:*

- **STRONGLY CONVEX:** f_i SATISFIES (A3) FOR $\mu > 0$, $\eta_l \leq \min\left(\frac{1}{81\beta K\eta_g}, \frac{S}{15\mu N K\eta_g}\right)$, $R \geq \max\left(\frac{162\beta}{\mu}, \frac{30N}{S}\right)$ THEN

$$\mathbb{E}[f(\bar{\mathbf{x}}^R)] - f(\mathbf{x}^*) \leq \tilde{\mathcal{O}}\left(\frac{\sigma^2}{\mu R K S} \left(1 + \frac{S}{\eta_g^2}\right) + \frac{N\mu}{S} \tilde{D}^2 \exp\left(-\min\left\{\frac{S}{30N}, \frac{\mu}{162\beta}\right\} R\right)\right).$$

- **GENERAL CONVEX:** f_i SATISFIES (A3) FOR $\mu = 0$, $\eta_l \leq \frac{1}{81\beta K\eta_g}$, $R \geq 1$ THEN

$$\mathbb{E}[f(\bar{\mathbf{x}}^R)] - f(\mathbf{x}^*) \leq \mathcal{O}\left(\frac{\sigma \tilde{D}}{\sqrt{R K S}} \left(\sqrt{1 + \frac{S}{\eta_g^2}}\right) + \sqrt{\frac{N}{S}} \frac{\beta \tilde{D}^2}{R}\right),$$

- **NON-CONVEX:** $\eta_l \leq \frac{1}{24K\eta_g\beta} \left(\frac{S}{N}\right)^{\frac{2}{3}}$, AND $R \geq 1$, THEN

$$\mathbb{E}[\|\nabla f(\bar{\mathbf{x}}^R)\|^2] \leq \mathcal{O}\left(\frac{\sigma\sqrt{F}}{\sqrt{RKS}}\left(\sqrt{1+\frac{N}{\eta_g^2}}\right) + \frac{\beta F}{R}\left(\frac{N}{S}\right)^{\frac{2}{3}}\right).$$

HERE $\tilde{D}^2 := (\|\mathbf{x}^0 - \mathbf{x}^\star\|^2 + \frac{1}{2N\beta^2} \sum_{i=1}^N \|\mathbf{c}_i^0 - \nabla f_i(\mathbf{x}^\star)\|^2)$ AND $F := (f(\mathbf{x}_0) - f(\mathbf{x}^\star))$.

Remark 64. Note that the \tilde{D}^2 defined above involves an additional term $\frac{1}{2N\beta^2} \sum_{i=1}^N \|\mathbf{c}_i^0 - \nabla f_i(\mathbf{x}^\star)\|^2$. This is standard in variance reduction methods (Johnson and Zhang, 2013; Defazio et al., 2014; Hanzely and Richtárik, 2019). Theoretically, we will use a warm-start strategy to set \mathbf{c}_i^0 and in the first N/S rounds, we compute $\mathbf{c}_i^0 = \mathbf{g}_i(\mathbf{x}^0)$ over a batch size of size K . Then, using smoothness of f_i , we can bound this additional term as

$$\frac{1}{2N\beta^2} \sum_{i=1}^N \|\mathbf{c}_i^0 - \nabla f_i(\mathbf{x}^\star)\|^2 \leq \frac{1}{\beta} (f(\mathbf{x}^0) - f^\star) + \frac{\sigma^2}{K\beta^2} \leq D^2 + \frac{\sigma^2}{K\beta^2}.$$

Thus, the asymptotic rates of SCAFFOLD FOR GENERAL CONVEX FUNCTIONS ONLY INCURS AN ADDITIVE TERM OF THE ORDER OF $O(\sqrt{\frac{N}{S}} \frac{1}{R})$. FOR STRONGLY CONVEX FUNCTIONS, WE ONLY SEE THE AFFECTS IN THE LOGARITHMIC TERMS.

Remark 65. When $\sigma = 0$ i.e. when clients compute full gradients, the communication complexity of SCAFFOLD is: i) for strongly convex case it is $\tilde{O}\left(\frac{N}{S} + \frac{\beta}{\mu}\right)$, ii) for general convex functions it is $O\left(\sqrt{\frac{N}{S}} \frac{\beta}{R}\right)$,² and iii) for non-convex functions it is $O\left(\frac{N^{2/3}}{S} \frac{\beta}{R}\right)$. In comparison, the follow up work of FedDyn (Acar et al., 2021) proves communication complexity matching ours in the convex and strongly convex settings, but a worse $O\left(\frac{N}{S} \frac{\beta}{R}\right)$ complexity in the non-convex settings (all when $\sigma = 0$).

WE WILL REWRITE SCAFFOLD USING NOTATION WHICH IS CONVENIENT FOR THE PROOFS: $\{\mathbf{y}_i\}$ REPRESENT THE CLIENT MODELS, \mathbf{x} IS THE AGGREGATE SERVER MODEL, AND \mathbf{c}_i AND \mathbf{c} ARE THE CLIENT AND SERVER CONTROL VARIATES. FOR AN EQUIVALENT DESCRIPTION WHICH IS EASIER TO IMPLEMENT, WE REFER TO ALGORITHM 5. THE SERVER MAINTAINS A GLOBAL CONTROL VARIATE \mathbf{c} AS BEFORE AND EACH CLIENT MAINTAINS ITS OWN CONTROL VARIATE \mathbf{c}_i . IN ROUND r , A SUBSET OF CLIENTS \mathcal{S}^r OF SIZE S ARE SAMPLED UNIFORMLY FROM $\{1, \dots, N\}$. SUPPOSE THAT *every* CLIENT PERFORMS THE FOLLOWING UPDATES

- STARTING FROM THE SHARED GLOBAL PARAMETERS $\mathbf{y}_{i,r}^0 = \mathbf{x}^{r-1}$, WE UPDATE THE LOCAL PARAMETERS FOR $k \in [K]$

$$\mathbf{y}_{i,k}^r = \mathbf{y}_{i,k-1}^r - \eta_l \mathbf{v}_{i,k}^r, \quad \text{WHERE} \quad \mathbf{v}_{i,k}^r := \mathbf{g}_i(\mathbf{y}_{i,k-1}^r) - \mathbf{c}_i^{r-1} + \mathbf{c}^{r-1} \quad (12.12)$$

² A previous version of the paper showed a worse dependence of $O\left(\frac{N}{S} \frac{\beta}{R}\right)$ due to sub-optimal choice of step-size η .

- UPDATE THE CONTROL ITERATES USING (OPTION II):

$$\tilde{\mathbf{c}}_i^r = \mathbf{c}^{r-1} - \mathbf{c}_i^{r-1} + \frac{1}{K\eta_l}(\mathbf{x}^{r-1} - \mathbf{x}_{i,K}^r) = \frac{1}{K} \sum_{k=1}^K g_i(\mathbf{y}_{i,k-1}^r). \quad (12.13)$$

WE UPDATE THE LOCAL CONTROL VARIATES ONLY FOR CLIENTS $i \in \mathcal{S}^r$

$$\mathbf{c}_i^r = \begin{cases} \tilde{\mathbf{c}}_i^r & \text{IF } i \in \mathcal{S}^r \\ \mathbf{c}_i^{r-1} & \text{OTHERWISE.} \end{cases} \quad (12.14)$$

- COMPUTE THE NEW GLOBAL PARAMETERS AND GLOBAL CONTROL VARIATE USING ONLY UPDATES FROM THE CLIENTS $i \in \mathcal{S}^r$:

$$\mathbf{x}^r = \mathbf{x}^{r-1} + \frac{\eta_g}{S} \sum_{i \in \mathcal{S}^r} (\mathbf{y}_{i,K}^r - \mathbf{x}^{r-1}) \text{ AND } \mathbf{c}^r = \frac{1}{N} \sum_{i=1}^N \mathbf{c}_i^r = \frac{1}{N} \left(\sum_{i \in \mathcal{S}^r} \mathbf{c}_i^r + \sum_{j \notin \mathcal{S}^r} \mathbf{c}_j^{r-1} \right). \quad (12.15)$$

FINALLY, FOR SOME WEIGHTS $\{w_r\}$, WE OUTPUT

$$\bar{\mathbf{x}}^R = \mathbf{x}^{r-1} \text{ WITH PROBABILITY } \frac{w_r}{\sum_{\tau} w_{\tau}} \text{ FOR } r \in \{1, \dots, R+1\}. \quad (12.16)$$

NOTE THAT THE CLIENTS ARE AGNOSTIC TO THE SAMPLING AND THEIR UPDATES ARE IDENTICAL TO WHEN ALL CLIENTS ARE PARTICIPATING. ALSO NOTE THAT THE CONTROL VARIATE CHOICE (12.13) CORRESPONDS TO (OPTION II) OF ALGORITHM 5. FURTHER, THE UPDATES OF THE CLIENTS $i \notin \mathcal{S}^r$ IS FORGOTTEN AND IS DEFINED ONLY TO MAKE THE PROOFS EASIER. WHILE ACTUALLY IMPLEMENTING THE METHOD, ONLY CLIENTS $i \in \mathcal{S}^r$ PARTICIPATE AND THE REST REMAIN INACTIVE (SEE ALGORITHM 5).

12.5.1 Convergence of SCAFFOLD FOR CONVEX FUNCTIONS (THEOREM VIII)

WE WILL FIRST BOUND THE VARIANCE OF SCAFFOLD UPDATE IN LEMMA 66, THEN SEE HOW SAMPLING OF CLIENTS EFFECTS OUR CONTROL VARIATES IN LEMMA 67, AND FINALLY BOUND THE AMOUNT OF CLIENT-DRIFT IN LEMMA 68. WE WILL THEN USE THESE THREE LEMMAS TO PROVE THE PROGRESS IN A SINGLE ROUND IN LEMMA 69. COMBINING THIS PROGRESS WITH LEMMAS 55 AND 56 GIVES US THE DESIRED RATES.

ADDITIONAL DEFINITIONS. BEFORE PROCEEDING WITH THE PROOF OF OUR LEMMAS, WE NEED SOME ADDITIONAL DEFINITIONS OF THE VARIOUS ERRORS WE TRACK. AS BEFORE, WE DEFINE THE EFFECTIVE STEP-SIZE TO BE

$$\tilde{\eta} := K\eta_l\eta_g.$$

WE DEFINE CLIENT-DRIFT TO BE HOW MUCH THE CLIENTS MOVE FROM THEIR STARTING POINT:

$$\mathcal{E}_r := \frac{1}{KN} \sum_{k=1}^K \sum_{i=1}^N \mathbb{E}[\|\mathbf{y}_{i,k}^r - \mathbf{x}^{r-1}\|^2]. \quad (12.17)$$

BECAUSE WE ARE SAMPLING THE CLIENTS, NOT ALL THE CLIENT CONTROL-VARIATES GET UPDATED EVERY ROUND. THIS LEADS TO SOME ‘LAG’ WHICH WE CALL CONTROL-LAG:

$$\mathcal{C}_r := \frac{1}{N} \sum_{j=1}^N \mathbb{E} \|\mathbb{E}[\mathbf{c}_i^r] - \nabla f_i(\mathbf{x}^*)\|^2. \quad (12.18)$$

VARIANCE OF SERVER UPDATE. WE STUDY HOW THE VARIANCE OF THE SERVER UPDATE CAN BE BOUNDED.

Lemma 66. *For updates (12.12)—(12.15), we can bound the variance of the server update in any round r and any $\tilde{\eta} := \eta_l \eta_g K \geq 0$ as follows*

$$\mathbb{E}[\|\mathbf{x}^r - \mathbf{x}^{r-1}\|^2] \leq 8\beta\tilde{\eta}^2(\mathbb{E}[f(\mathbf{x}^{r-1})] - f(\mathbf{x}^*)) + 8\tilde{\eta}^2\mathcal{C}_{r-1} + 4\tilde{\eta}^2\beta^2\mathcal{E}_r + \frac{12\tilde{\eta}^2\sigma^2}{KS}.$$

Proof. The server update in round r can be written as follows (dropping the superscript r everywhere)

$$\mathbb{E}\|\Delta\mathbf{x}\|^2 = \mathbb{E}\left\|\frac{\tilde{\eta}}{KS} \sum_{k,i \in \mathcal{S}} \mathbf{v}_{i,k}\right\|^2 = \mathbb{E}\left\|\frac{\tilde{\eta}}{KS} \sum_{k,i \in \mathcal{S}} (g_i(\mathbf{y}_{i,k-1}) + \mathbf{c} - \mathbf{c}_i)\right\|^2,$$

which can then be expanded as

$$\begin{aligned} \mathbb{E}\|\Delta\mathbf{x}\|^2 &\leq \mathbb{E}\left\|\frac{\tilde{\eta}}{KS} \sum_{k,i \in \mathcal{S}} (g_i(\mathbf{y}_{i,k-1}) + \mathbf{c} - \mathbf{c}_i)\right\|^2 \\ &\leq 4\mathbb{E}\left\|\frac{\tilde{\eta}}{KS} \sum_{k,i \in \mathcal{S}} g_i(\mathbf{y}_{i,k-1}) - \nabla f_i(\mathbf{x})\right\|^2 + 4\tilde{\eta}^2\mathbb{E}\|\mathbf{c}\|^2 + 4\mathbb{E}\left\|\frac{\tilde{\eta}}{KS} \sum_{k,i \in \mathcal{S}} \nabla f_i(\mathbf{x}^*) - \mathbf{c}_i\right\|^2 \\ &\quad + 4\mathbb{E}\left\|\frac{\tilde{\eta}}{KS} \sum_{k,i \in \mathcal{S}} \nabla f_i(\mathbf{x}) - \nabla f_i(\mathbf{x}^*)\right\|^2 \\ &\stackrel{(12.3)}{\leq} 4\mathbb{E}\left\|\frac{\tilde{\eta}}{KS} \sum_{k,i \in \mathcal{S}} g_i(\mathbf{y}_{i,k-1}) - \nabla f_i(\mathbf{x})\right\|^2 + 4\tilde{\eta}^2\mathbb{E}\|\mathbf{c}\|^2 + 4\mathbb{E}\left\|\frac{\tilde{\eta}}{S} \sum_{i \in \mathcal{S}} \nabla f_i(\mathbf{x}^*) - \mathbf{c}_i\right\|^2 \\ &\quad + 8\beta\tilde{\eta}^2(\mathbb{E}[f(\mathbf{x})] - f(\mathbf{x}^*)) \\ &\leq 4\mathbb{E}\left\|\frac{\tilde{\eta}}{KS} \sum_{k,i \in \mathcal{S}} \nabla f_i(\mathbf{y}_{i,k-1}) - \nabla f_i(\mathbf{x})\right\|^2 + 4\tilde{\eta}^2\mathbb{E}\|\mathbf{c}\|^2 + 4\mathbb{E}\left\|\frac{\tilde{\eta}}{S} \sum_{i \in \mathcal{S}} \nabla f_i(\mathbf{x}^*) - \mathbb{E}[\mathbf{c}_i]\right\|^2 \\ &\quad + 8\beta\tilde{\eta}^2(\mathbb{E}[f(\mathbf{x})] - f(\mathbf{x}^*)) + \frac{12\tilde{\eta}^2\sigma^2}{KS}. \end{aligned}$$

The inequality before the last used the smoothness of $\{f_i\}$. The last inequality which separates the mean and the variance is an application of Lemma 58: the variance of $(\frac{1}{KS} \sum_{k,i \in \mathcal{S}} g_i(\mathbf{y}_{i,k-1}))$ is bounded by σ^2/KS . Similarly, \mathbf{c}_j as defined in (12.13) for any $j \in [N]$ has variance smaller than σ^2/K and hence the variance of $(\frac{1}{S} \sum_{i \in \mathcal{S}} \mathbf{c}_i)$ is smaller than σ^2/KS .

Using Lemma 57.2 twice to simplify:

$$\begin{aligned}
 \mathbb{E}\|\Delta \mathbf{x}\|^2 &\leq \frac{4\tilde{\eta}^2}{KN} \sum_{k,i} \mathbb{E}\|\nabla f_i(\mathbf{y}_{i,k-1}) - \nabla f_i(\mathbf{x})\|^2 + 4\tilde{\eta}^2 \|\mathbb{E}[\mathbf{c}]\|^2 + \frac{4\tilde{\eta}^2}{N} \sum_i \|\nabla f_i(\mathbf{x}^\star) - \mathbb{E}[\mathbf{c}_i]\|^2 \\
 &\quad + 8\beta\tilde{\eta}^2 (\mathbb{E}[f(\mathbf{x})] - f(\mathbf{x}^\star)) + \frac{12\tilde{\eta}^2\sigma^2}{KS} \\
 &\leq \underbrace{\frac{4\tilde{\eta}^2}{KN} \sum_{k,i} \mathbb{E}\|\nabla f_i(\mathbf{y}_{i,k-1}) - \nabla f_i(\mathbf{x})\|^2}_{\mathcal{T}_1} + \frac{8\tilde{\eta}^2}{N} \sum_i \|\nabla f_i(\mathbf{x}^\star) - \mathbb{E}[\mathbf{c}_i]\|^2 \\
 &\quad + 8\beta\tilde{\eta}^2 (\mathbb{E}[f(\mathbf{x})] - f(\mathbf{x}^\star)) + \frac{12\tilde{\eta}^2\sigma^2}{KS}.
 \end{aligned}$$

The second step follows because $\mathbf{c} = \frac{1}{N} \sum_i \mathbf{c}_i$. Since the gradient of f_i is β -Lipschitz, $\mathcal{T}_1 \leq \frac{\beta^2 4\tilde{\eta}^2}{KN} \sum_{k,i} \mathbb{E}\|\mathbf{y}_{i,k-1} - \mathbf{x}\|^2 = 4\tilde{\eta}^2 \beta^2 \mathcal{E}$. The definition of the error in the control variate $\mathcal{C}_{r-1} := \frac{1}{N} \sum_{j=1}^N \mathbb{E}\|\mathbf{c}_j - \nabla f_j(\mathbf{x}^\star)\|^2$ completes the proof. \square

CHANGE IN CONTROL LAG. WE HAVE PREVIOUSLY RELATED THE VARIANCE OF THE SERVER UPDATE TO THE CONTROL LAG. WE NOW EXAMINE HOW THE CONTROL-LAG GROWS EACH ROUND.

Lemma 67. For updates (12.12)—(12.15) with the control update (12.13) and assumptions A3–A5, the following holds true for any $\tilde{\eta} := \eta_l \eta_g K \in [0, 1/\beta]$:

$$\mathcal{C}_r \leq (1 - \frac{S}{N}) \mathcal{C}_{r-1} + \frac{S}{N} (4\beta (\mathbb{E}[f(\mathbf{x}^{r-1})] - f(\mathbf{x}^\star)) + 2\beta^2 \mathcal{E}_r).$$

Proof. Recall that after round r , the control update rule (12.13) implies that \mathbf{c}_i^r is set as per

$$\mathbf{c}_i^r = \begin{cases} \mathbf{c}_i^{r-1} & \text{if } i \notin \mathcal{S}^r \text{ i.e. with probability } (1 - \frac{S}{N}), \\ \frac{1}{K} \sum_{k=1}^K \mathbf{g}_i(\mathbf{y}_{i,k-1}^r) & \text{with probability } \frac{S}{N}. \end{cases}$$

Taking expectations on both sides yields

$$\mathbb{E}[\mathbf{c}_i^r] = (1 - \frac{S}{N}) \mathbb{E}[\mathbf{c}_i^{r-1}] + \frac{S}{KN} \sum_{k=1}^K \mathbb{E}[\nabla f_i(\mathbf{y}_{i,k-1}^r)], \quad \forall i \in [N].$$

Plugging the above expression in the definition of \mathcal{C}_r we get

$$\begin{aligned}
 \mathcal{C}_r &= \frac{1}{N} \sum_{i=1}^N \|\mathbb{E}[\mathbf{c}_i^r] - \nabla f_i(\mathbf{x}^\star)\|^2 \\
 &= \frac{1}{N} \sum_{i=1}^N \|(1 - \frac{S}{N})(\mathbb{E}[\mathbf{c}_i^{r-1}] - \nabla f_i(\mathbf{x}^\star)) + \frac{S}{N} (\frac{1}{K} \sum_{k=1}^K \mathbb{E}[\nabla f_i(\mathbf{y}_{i,k-1}^r)] - \nabla f_i(\mathbf{x}^\star))\|^2 \\
 &\leq (1 - \frac{S}{N}) \mathcal{C}_{r-1} + \frac{S}{N^2 K} \sum_{k=1}^K \mathbb{E}\|\nabla f_i(\mathbf{y}_{i,k-1}^r) - \nabla f_i(\mathbf{x}^\star)\|^2.
 \end{aligned}$$

The final step applied Jensen's inequality twice. We can then further simplify using the re-

laxed triangle inequality as

$$\begin{aligned}
 \mathbb{E}_{r-1}[\mathcal{E}_r] &\leq \left(1 - \frac{S}{N}\right) \mathcal{E}_{r-1} + \frac{S}{N^2 K} \sum_{i,k} \mathbb{E} \|\nabla f_i(\mathbf{y}_{i,k-1}^r) - \nabla f_i(\mathbf{x}^*)\|^2 \\
 &\leq \left(1 - \frac{S}{N}\right) \mathcal{E}_{r-1} + \frac{2S}{N^2} \sum_i \mathbb{E} \|\nabla f_i(\mathbf{x}^{r-1}) - \nabla f_i(\mathbf{x}^*)\|^2 + \frac{2S}{N^2 K} \sum_{i,k} \mathbb{E} \|\nabla f_i(\mathbf{y}_{i,k-1}^r) - \nabla f_i(\mathbf{x}^{r-1})\|^2 \\
 &\stackrel{(12.1)}{\leq} \left(1 - \frac{S}{N}\right) \mathcal{E}_{r-1} + \frac{2S}{N^2} \sum_i \mathbb{E} \|\nabla f_i(\mathbf{x}^{r-1}) - \nabla f_i(\mathbf{x}^*)\|^2 + \frac{2S}{N^2 K} \beta^2 \sum_{i,k} \mathbb{E} \|\mathbf{y}_{i,k-1}^r - \mathbf{x}^{r-1}\|^2 \\
 &\stackrel{(12.3)}{\leq} \left(1 - \frac{S}{N}\right) \mathcal{E}_{r-1} + \frac{S}{N} (4\beta(\mathbb{E}[f(\mathbf{x}^{r-1})] - f(\mathbf{x}^*)) + \beta^2 \mathcal{E}_r).
 \end{aligned}$$

The last two inequalities follow from smoothness of $\{f_i\}$ and the definition $\mathcal{E}_r = \frac{1}{NK} \beta^2 \sum_{i,k} \mathbb{E} \|\mathbf{y}_{i,k-1}^r - \mathbf{x}^{r-1}\|^2$. \square

BOUNDING CLIENT-DRIFT. WE WILL NOW BOUND THE FINAL SOURCE OF ERROR WHICH IS THE CLIENT-DRIFT.

Lemma 68. Suppose our step-sizes satisfy $\eta_l \leq \frac{1}{81\beta K \eta_g}$ and f_i satisfies assumptions A3–A5. Then, for any global $\eta_g \geq 1$ we can bound the drift as

$$3\beta\tilde{\eta}\mathcal{E}_r \leq \frac{2\tilde{\eta}^2}{3} \mathcal{E}_{r-1} + \frac{\tilde{\eta}}{25\eta_g^2} (\mathbb{E}[f(\mathbf{x}^{r-1})] - f(\mathbf{x}^*)) + \frac{\tilde{\eta}^2}{K\eta_g^2} \sigma^2.$$

Proof. First, observe that if $K = 1$, $\mathcal{E}_r = 0$ since $\mathbf{y}_{i,0} = \mathbf{x}$ for all $i \in [N]$ and that \mathcal{E}_{r-1} and the right hand side are both positive. Thus the lemma is trivially true if $K = 1$. For $K > 1$, we build a recursive bound of the drift. Starting from the definition of the update (12.12) and then applying the relaxed triangle inequality, we can expand

$$\begin{aligned}
 \frac{1}{S} \mathbb{E}_{r-1} \left[\sum_{i \in \mathcal{S}} \|\mathbf{y}_i - \eta_l \mathbf{v}_i - \mathbf{x}\|^2 \right] &= \frac{1}{S} \mathbb{E}_{r-1} \left[\sum_{i \in \mathcal{S}} \|\mathbf{y}_i - \eta_l \mathbf{g}_i(\mathbf{y}_i) + \eta_l \mathbf{c} - \eta_l \mathbf{c}_i - \mathbf{x}\|^2 \right] \\
 &\leq \frac{1}{S} \mathbb{E}_{r-1} \left[\sum_{i \in \mathcal{S}} \|\mathbf{y}_i - \eta_l \nabla f_i(\mathbf{y}_i) + \eta_l \mathbf{c} - \eta_l \mathbf{c}_i - \mathbf{x}\|^2 \right] + \eta_l^2 \sigma^2 \\
 &\leq \frac{(1+a)}{S} \mathbb{E}_{r-1} \left[\sum_{i \in \mathcal{S}} \underbrace{\|\mathbf{y}_i - \eta_l \nabla f_i(\mathbf{y}_i) + \eta_l \nabla f_i(\mathbf{x}) - \mathbf{x}\|^2}_{\mathcal{T}_2} \right] \\
 &\quad + \underbrace{(1 + \frac{1}{a}) \eta_l^2 \mathbb{E}_{r-1} \left[\frac{1}{S} \sum_{i \in \mathcal{S}} \|\mathbf{c} - \mathbf{c}_i + \nabla f_i(\mathbf{x})\|^2 \right]}_{\mathcal{T}_3} + \eta_l^2 \sigma^2.
 \end{aligned}$$

The final step follows from the relaxed triangle inequality (Lemma 57). Applying the contractive mapping Lemma 60 for $\eta_l \leq 1/\beta$ shows

$$\mathcal{T}_2 = \frac{1}{S} \sum_{i \in \mathcal{S}} \|\mathbf{y}_i - \eta_l \nabla f_i(\mathbf{y}_i) + \eta_l \nabla f_i(\mathbf{x}) - \mathbf{x}\|^2 \leq \|\mathbf{y}_i - \mathbf{x}\|^2.$$

Once again using our relaxed triangle inequality to expand the other term \mathcal{T}_3 , we get

$$\begin{aligned}
 \mathcal{T}_3 &= \mathbb{E}_{r-1} \left[\frac{1}{S} \sum_{i \in \mathcal{S}} \|\mathbf{c} - \mathbf{c}_i + \nabla f_i(\mathbf{x})\|^2 \right] \\
 &= \frac{1}{N} \sum_{j=1}^N \|\mathbf{c} - \mathbf{c}_i + \nabla f_i(\mathbf{x})\|^2 \\
 &= \frac{1}{N} \sum_{j=1}^N \|\mathbf{c} - \mathbf{c}_i + \nabla f_i(\mathbf{x}^*) + \nabla f_i(\mathbf{x}) - \nabla f_i(\mathbf{x}^*)\|^2 \\
 &\leq 3\|\mathbf{c}\|^2 + \frac{3}{N} \sum_{j=1}^N \|\mathbf{c}_i - \nabla f_i(\mathbf{x}^*)\|^2 + \frac{3}{N} \sum_{j=1}^N \|\nabla f_i(\mathbf{x}) - \nabla f_i(\mathbf{x}^*)\|^2 \\
 &\leq \frac{6}{N} \sum_{j=1}^N \|\mathbf{c}_i - \nabla f_i(\mathbf{x}^*)\|^2 + \frac{3}{N} \sum_{j=1}^N \|\nabla f_i(\mathbf{x}) - \nabla f_i(\mathbf{x}^*)\|^2 \\
 &\leq \frac{6}{N} \sum_{j=1}^N \|\mathbf{c}_i - \nabla f_i(\mathbf{x}^*)\|^2 + 6\beta(f(\mathbf{x}) - f(\mathbf{x}^*)).
 \end{aligned}$$

The last step used the smoothness of f_i . Combining the bounds on \mathcal{T}_2 and \mathcal{T}_3 in the original inequality and using $a = \frac{1}{K-1}$ gives

$$\begin{aligned}
 \frac{1}{N} \sum_i \mathbb{E} \|\mathbf{y}_{i,k} - \mathbf{x}\|^2 &\leq \frac{(1 + \frac{1}{K-1})}{N} \sum_i \mathbb{E} \|\mathbf{y}_{i,k-1} - \mathbf{x}\|^2 + \eta_l^2 \sigma^2 \\
 &\quad + 6\eta_l^2 K \beta(f(\mathbf{x}) - f(\mathbf{x}^*)) + \frac{6K\eta_l^2}{N} \sum_i \mathbb{E} \|\mathbf{c}_i - \nabla f_i(\mathbf{x}^*)\|^2.
 \end{aligned}$$

Recall that with the choice of \mathbf{c}_i in (12.13), the variance of c_i is less than $\frac{\sigma^2}{K}$. Separating its mean and variance gives

$$\begin{aligned}
 \frac{1}{N} \sum_i \mathbb{E} \|\mathbf{y}_{i,k} - \mathbf{x}\|^2 &\leq \left(1 + \frac{1}{K-1}\right) \frac{1}{N} \sum_i \mathbb{E} \|\mathbf{y}_{i,k-1} - \mathbf{x}\|^2 + 7\eta_l^2 \sigma^2 + \\
 &\quad 6\eta_l^2 K \beta(f(\mathbf{x}) - f(\mathbf{x}^*)) + \frac{6K\eta_l^2}{N} \sum_i \mathbb{E} \|\mathbf{c}_i - \nabla f_i(\mathbf{x}^*)\|^2 \quad (12.19)
 \end{aligned}$$

Unrolling the recursion (12.19), we get the following for any $k \in \{1, \dots, K\}$

$$\begin{aligned}
 \frac{1}{N} \sum_i \mathbb{E} \|\mathbf{y}_{i,k} - \mathbf{x}\|^2 &\leq (6K\beta\eta_l^2(f(\mathbf{x}) - f(\mathbf{x}^*)) + 6K\eta_l^2\mathcal{C}_{r-1} + 7\beta\eta_l^2\sigma^2) \left(\sum_{\tau=0}^{k-1} (1 + \frac{1}{K-1})^\tau \right) \\
 &\leq (6K\beta\eta_l^2(f(\mathbf{x}) - f(\mathbf{x}^*)) + 6K\eta_l^2\mathcal{C}_{r-1} + 7\beta\eta_l^2\sigma^2) (K-1) ((1 + \frac{1}{K-1})^K - 1) \\
 &\leq (6K\beta\eta_l^2(f(\mathbf{x}) - f(\mathbf{x}^*)) + 6K\eta_l^2\mathcal{C}_{r-1} + 7\beta\eta_l^2\sigma^2) 3K \\
 &\leq 18K^2\beta\eta_l^2(f(\mathbf{x}) - f(\mathbf{x}^*)) + 18K^2\eta_l^2\mathcal{C}_{r-1} + 21K\beta\eta_l^2\sigma^2.
 \end{aligned}$$

The inequality $(K-1)((1 + \frac{1}{K-1})^K - 1) \leq 3K$ can be verified for $K = 2, 3$ manually. For $K \geq 4$,

$$(K-1)((1 + \frac{1}{K-1})^K - 1) < K(\exp(\frac{K}{K-1}) - 1) \leq K(\exp(\frac{4}{3}) - 1) < 3K.$$

Again averaging over k and multiplying by 3β yields

$$\begin{aligned} 3\beta\mathcal{E}_r &\leq 54K^2\beta^2\eta_l^2(f(\mathbf{x}) - f(\mathbf{x}^*)) + 54K^2\beta\eta_l^2\mathcal{C}_{r-1} + 63\beta K\eta_l^2\sigma^2 \\ &= \frac{1}{\eta_g^2} \left(54\beta^2\tilde{\eta}^2(f(\mathbf{x}) - f(\mathbf{x}^*)) + 54\beta\tilde{\eta}^2\mathcal{C}_{r-1} + 63\beta\tilde{\eta}^2\frac{\sigma^2}{K} \right) \\ &\leq \frac{1}{\eta_g^2} \left(\frac{1}{25}(f(\mathbf{x}) - f(\mathbf{x}^*)) + \frac{2}{3}\tilde{\eta}\mathcal{C}_{r-1} + \tilde{\eta}\frac{\sigma^2}{K} \right). \end{aligned}$$

The equality follows from the definition $\tilde{\eta} = K\eta_l\eta_g$, and the final inequality uses the bound that $\tilde{\eta} \leq \frac{1}{81\beta}$. \square

PROGRESS IN ONE ROUND. NOW THAT WE HAVE A BOUND ON ALL ERRORS, WE CAN DESCRIBE OUR PROGRESS.

Lemma 69. Suppose assumptions A3–A5 are true. Then the following holds for any step-sizes satisfying $\eta_g \geq 1$, $\eta_l \leq \min\left(\frac{1}{81\beta K\eta_g}, \frac{S}{15\mu N K\eta_g}\right)$, and effective step-size $\tilde{\eta} := K\eta_g\eta_l$

$$\mathbb{E} \left[\|\mathbf{x}^r - \mathbf{x}^*\|^2 + \frac{9N\tilde{\eta}^2}{S}\mathcal{C}_r \right] \leq (1 - \frac{\mu\tilde{\eta}}{2}) \left(\mathbb{E} \|\mathbf{x}^{r-1} - \mathbf{x}^*\|^2 + \frac{9N\tilde{\eta}^2}{S}\mathcal{C}_{r-1} \right) - \tilde{\eta}(\mathbb{E}[f(\mathbf{x}^{r-1})] - f(\mathbf{x}^*)) + \frac{12\tilde{\eta}^2}{KS} \left(1 + \frac{S}{\eta_g^2} \right) \sigma^2.$$

Proof. Starting from our server update equation,

$$\Delta\mathbf{x} = -\frac{\tilde{\eta}}{KS} \sum_{k,i \in \mathcal{S}} (g_i(\mathbf{y}_{i,k-1}) + \mathbf{c} - \mathbf{c}_i), \text{ and } \mathbb{E}[\Delta\mathbf{x}] = -\frac{\tilde{\eta}}{KN} \sum_{k,i} g_i(\mathbf{y}_{i,k-1}).$$

We can then apply Lemma 66 to bound the second moment of the server update as

$$\begin{aligned} \mathbb{E}_{r-1} \|\mathbf{x} + \Delta\mathbf{x} - \mathbf{x}^*\|^2 &= \mathbb{E}_{r-1} \|\mathbf{x} - \mathbf{x}^*\|^2 - \frac{2\tilde{\eta}}{KS} \mathbb{E}_{r-1} \sum_{k,i \in \mathcal{S}} \langle \nabla f_i(\mathbf{y}_{i,k-1}), \mathbf{x} - \mathbf{x}^* \rangle + \mathbb{E}_{r-1} \|\Delta\mathbf{x}\|^2 \\ &\leq \underbrace{\frac{2\tilde{\eta}}{KS} \mathbb{E}_{r-1} \sum_{k,i \in \mathcal{S}} \langle \nabla f_i(\mathbf{y}_{i,k-1}), \mathbf{x}^* - \mathbf{x} \rangle + \mathbb{E}_{r-1} \|\mathbf{x} - \mathbf{x}^*\|^2}_{\mathcal{T}_4} \\ &\quad + 8\beta\tilde{\eta}^2(\mathbb{E}[f(\mathbf{x}^{r-1})] - f(\mathbf{x}^*)) + 8\tilde{\eta}^2\mathcal{C}_{r-1} + 4\tilde{\eta}^2\beta^2\mathcal{E} + \frac{12\tilde{\eta}^2\sigma^2}{KS}. \end{aligned}$$

The term \mathcal{T}_4 can be bounded by using perturbed strong-convexity (Lemma 59) with $h = f_i$,

$\mathbf{x} = \mathbf{y}_{i,k-1}$, $\mathbf{y} = \mathbf{x}^\star$, and $\mathbf{z} = \mathbf{x}$ to get

$$\begin{aligned}\mathbb{E}[\mathcal{T}_4] &= \frac{2\tilde{\eta}}{KS} \mathbb{E} \sum_{k,i \in \mathcal{S}} \langle \nabla f_i(\mathbf{y}_{i,k-1}), \mathbf{x}^\star - \mathbf{x} \rangle \\ &\leq \frac{2\tilde{\eta}}{KS} \mathbb{E} \sum_{k,i \in \mathcal{S}} \left(f_i(\mathbf{x}^\star) - f_i(\mathbf{x}) + \beta \|\mathbf{y}_{i,k-1} - \mathbf{x}\|^2 - \frac{\mu}{4} \|\mathbf{x} - \mathbf{x}^\star\|^2 \right) \\ &= -2\tilde{\eta} \mathbb{E} \left(f(\mathbf{x}) - f(\mathbf{x}^\star) + \frac{\mu}{4} \|\mathbf{x} - \mathbf{x}^\star\|^2 \right) + 2\beta\tilde{\eta}\mathcal{E}.\end{aligned}$$

Plugging \mathcal{T}_4 back, we can further simplify the expression to get

$$\begin{aligned}\mathbb{E} \|\mathbf{x} + \Delta\mathbf{x} - \mathbf{x}^\star\|^2 &\leq \mathbb{E} \|\mathbf{x} - \mathbf{x}^\star\|^2 - 2\tilde{\eta} \left(f(\mathbf{x}) - f(\mathbf{x}^\star) + \frac{\mu}{4} \|\mathbf{x} - \mathbf{x}^\star\|^2 \right) + 2\beta\tilde{\eta}\mathcal{E} \\ &\quad + \frac{12\tilde{\eta}^2\sigma^2}{KS} + 8\beta\tilde{\eta}^2(\mathbb{E}[f(\mathbf{x}^{r-1})] - f(\mathbf{x}^\star)) + 8\tilde{\eta}^2\mathcal{C}_{r-1} + 4\tilde{\eta}^2\beta^2\mathcal{E} \\ &= (1 - \frac{\mu\tilde{\eta}}{2}) \|\mathbf{x} - \mathbf{x}^\star\|^2 + (8\beta\tilde{\eta}^2 - 2\tilde{\eta})(f(\mathbf{x}) - f(\mathbf{x}^\star)) \\ &\quad + \frac{12\tilde{\eta}^2\sigma^2}{KS} + (2\beta\tilde{\eta} + 4\beta^2\tilde{\eta}^2)\mathcal{E} + 8\tilde{\eta}^2\mathcal{C}_{r-1}.\end{aligned}$$

We can use Lemma 67 (scaled by $9\tilde{\eta}^2 \frac{N}{S}$) to bound the control-lag

$$9\tilde{\eta}^2 \frac{N}{S} \mathcal{C}_r \leq (1 - \frac{\mu\tilde{\eta}}{2}) 9\tilde{\eta}^2 \frac{N}{S} \mathcal{C}_{r-1} + 9(\frac{\mu\tilde{\eta}N}{2S} - 1)\tilde{\eta}^2\mathcal{C}_{r-1} + 9\tilde{\eta}^2(4\beta(\mathbb{E}[f(\mathbf{x}^{r-1})] - f(\mathbf{x}^\star)) + 2\beta^2\mathcal{E})$$

Now recall that Lemma 68 bounds the client-drift:

$$3\beta\tilde{\eta}\mathcal{E}_r \leq \frac{2\tilde{\eta}^2}{3}\mathcal{C}_{r-1} + \frac{\tilde{\eta}}{25\tilde{\eta}_g^2}(\mathbb{E}[f(\mathbf{x}^{r-1})] - f(\mathbf{x}^\star)) + \frac{\tilde{\eta}^2}{K\tilde{\eta}_g^2}\sigma^2.$$

Adding all three inequalities together,

$$\begin{aligned}\mathbb{E} \|\mathbf{x} + \Delta\mathbf{x} - \mathbf{x}^\star\|^2 + \frac{9\tilde{\eta}^2 N \mathcal{C}_r}{S} &\leq (1 - \frac{\mu\tilde{\eta}}{2}) \left(\mathbb{E} \|\mathbf{x} - \mathbf{x}^\star\|^2 + \frac{9\tilde{\eta}^2 N \mathcal{C}_{r-1}}{S} \right) + (44\beta\tilde{\eta}^2 - \frac{49}{25}\tilde{\eta})(f(\mathbf{x}) - f(\mathbf{x}^\star)) \\ &\quad + \frac{12\tilde{\eta}^2\sigma^2}{KS} (1 + \frac{S}{\tilde{\eta}_g^2}) + (22\beta^2\tilde{\eta}^2 - \beta\tilde{\eta})\mathcal{E} + (\frac{9\mu\tilde{\eta}N}{2S} - \frac{1}{3})\tilde{\eta}^2\mathcal{C}_{r-1}\end{aligned}$$

Finally, the lemma follows from noting that $\tilde{\eta} \leq \frac{1}{81\tilde{\beta}}$ implies $44\beta^2\tilde{\eta}^2 \leq \frac{24}{25}\tilde{\beta}$ and $\tilde{\eta} \leq \frac{S}{15\mu N}$ implies $\frac{9\mu\tilde{\eta}N}{2S} \leq \frac{1}{3}$. \square

THE FINAL RATE FOR STRONGLY CONVEX FOLLOWS SIMPLY BY UNROLLING THE RECURSIVE BOUND IN LEMMA 69 USING LEMMA 55. ALSO NOTE THAT IF $c_i^0 = g_i(\mathbf{x}^0)$, THEN $\frac{\tilde{\eta}N}{S}\mathcal{C}_0$ CAN BE BOUNDED IN TERMS OF FUNCTION SUB-OPTIMALITY F . FOR THE **GENERAL CONVEX** SETTING, AVERAGING

OVER r IN LEMMA 69 WITH $\mu = 0$ GIVES

$$\begin{aligned} \frac{1}{R} \sum_{r=1}^R \mathbb{E}[f(\mathbf{x}^{r-1})] - f(\mathbf{x}^\star) &\leq \frac{1}{\tilde{\eta}R} \|\mathbf{x}^0 - \mathbf{x}^\star\|^2 + \frac{9N\tilde{\eta}}{SR} \mathcal{C}_0 + \frac{12\tilde{\eta}}{KS} (1 + \frac{S}{\eta_g^2}) \sigma^2 \\ &\leq 4 \|\mathbf{x}^0 - \mathbf{x}^\star\| \sigma \sqrt{\frac{3(1 + S/\eta_g^2)}{RKS}} \\ &\quad + \sqrt{\frac{N}{S} \frac{\|\mathbf{x}^0 - \mathbf{x}^\star\|^2 + 9\mathcal{C}_0}{R} + \frac{81\beta \|\mathbf{x}^0 - \mathbf{x}^\star\|^2}{R}} \dots \end{aligned}$$

THE LAST STEP FOLLOWS FROM USING A STEP SIZE OF $\tilde{\eta} = \min \left(\frac{1}{81\beta}, \sqrt{\frac{S}{N}}, \frac{\|\mathbf{x}^0 - \mathbf{x}^\star\|}{\sigma} \sqrt{\frac{KS}{12R(1 + \frac{S}{\eta_g^2})}} \right)$.

12.5.2 Convergence of SCAFFOLD FOR NON-CONVEX FUNCTIONS (THEOREM VIII)

WE NOW ANALYZE THE MOST GENERAL CASE OF SCAFFOLD WITH OPTION II ON FUNCTIONS WHICH ARE POTENTIALLY NON-CONVEX. JUST AS IN THE NON-CONVEX PROOF, WE WILL FIRST BOUND THE VARIANCE OF THE SERVER UPDATE IN LEMMA 70, THE CHANGE IN CONTROL LAG IN LEMMA 71 AND FINALLY WE BOUND THE CLIENT-DRIFT IN LEMMA 72. COMBINING THESE THREE TOGETHER GIVES US THE PROGRESS MADE IN ONE ROUND IN LEMMA 73. THE FINAL RATE IS DERIVED FROM THE PROGRESS MADE USING LEMMA 56.

ADDITIONAL NOTATION. RECALL THAT IN ROUND r , WE UPDATE THE CONTROL VARIATE AS (12.13)

$$\mathbf{c}_i^r = \begin{cases} \frac{1}{K} \sum_{k=1}^K \mathbf{g}_i(\mathbf{y}_{i,k-1}^r) & \text{IF } i \in \mathcal{S}^r, \\ \mathbf{c}_i^{r-1} & \text{OTHERWISE.} \end{cases}$$

WE INTRODUCE THE FOLLOWING NOTATION TO KEEP TRACK OF THE ‘LAG’ IN THE UPDATE OF THE CONTROL VARIATE: DEFINE A SEQUENCE OF PARAMETERS $\{\boldsymbol{\alpha}_{i,k-1}^{r-1}\}$ SUCH THAT FOR ANY $i \in [N]$ AND $k \in [K]$ WE HAVE $\boldsymbol{\alpha}_{i,k-1}^0 := \mathbf{x}^0$ AND FOR $r \geq 1$,

$$\boldsymbol{\alpha}_{i,k-1}^r := \begin{cases} \mathbf{y}_{i,k-1}^r & \text{IF } i \in \mathcal{S}^r, \\ \boldsymbol{\alpha}_{i,k-1}^{r-1} & \text{OTHERWISE.} \end{cases} \quad (12.20)$$

BY THE UPDATE RULE FOR CONTROL VARIATES (12.13) AND THE DEFINITION OF $\{\boldsymbol{\alpha}_{i,k-1}^{r-1}\}$ ABOVE, THE FOLLOWING PROPERTY ALWAYS HOLDS:

$$\mathbf{c}_i^r = \frac{1}{K} \sum_{k=1}^K \mathbf{g}_i(\boldsymbol{\alpha}_{i,k-1}^r).$$

WE CAN THEN DEFINE THE FOLLOWING Ξ_r TO BE THE ERROR IN CONTROL VARIATE FOR ROUND r :

$$\Xi_r := \frac{1}{KN} \sum_{k=1}^K \sum_{i=1}^N \mathbb{E} \|\boldsymbol{\alpha}_{i,k-1}^r - \mathbf{x}^r\|^2. \quad (12.21)$$

ALSO RECALL THE CLOSELY RELATED DEFINITION OF CLIENT DRIFT CAUSED BY LOCAL UPDATES:

$$\mathcal{E}_r := \frac{1}{KN} \sum_{k=1}^K \sum_{i=1}^N \mathbb{E}[\|\mathbf{y}_{i,k}^r - \mathbf{x}^{r-1}\|^2].$$

VARIANCE OF SERVER UPDATE. LET US ANALYZE HOW THE CONTROL VARIATES EFFECT THE VARIANCE OF THE AGGREGATE SERVER UPDATE.

Lemma 70. *For updates (12.12)—(12.15) and assumptions A4 and A5, the following holds true for any $\tilde{\eta} := \eta_l \eta_g K \in [0, 1/\beta]$:*

$$\begin{aligned} \mathbb{E}\|\mathbb{E}_{r-1}[\mathbf{x}^r] - \mathbf{x}^{r-1}\|^2 &\leq 2\tilde{\eta}^2 \beta^2 \mathcal{E}_r + 2\tilde{\eta}^2 \mathbb{E}\|\nabla f(\mathbf{x}^{r-1})\|^2, \text{ and} \\ \mathbb{E}\|\mathbf{x}^r - \mathbf{x}^{r-1}\|^2 &\leq 4\tilde{\eta}^2 \beta^2 \mathcal{E}_r + 8\tilde{\eta}^2 \beta^2 \Xi_{r-1} + 4\tilde{\eta}^2 \mathbb{E}\|\nabla f(\mathbf{x}^{r-1})\|^2 + \frac{9\tilde{\eta}^2 \sigma^2}{KS}. \end{aligned}$$

Proof. Recall that the server update satisfies

$$\mathbb{E}[\Delta \mathbf{x}] = -\frac{\tilde{\eta}}{KN} \sum_{k,i} \mathbb{E}[g_i(\mathbf{y}_{i,k-1})].$$

From the definition of $\alpha_{i,k-1}^{r-1}$ and dropping the superscript everywhere we have

$$\Delta \mathbf{x} = -\frac{\tilde{\eta}}{KS} \sum_{k,i \in \mathcal{S}} (g_i(\mathbf{y}_{i,k-1}) + \mathbf{c} - \mathbf{c}_i) \text{ where } \mathbf{c}_i = \frac{1}{K} \sum_k g_i(\alpha_{i,k-1}).$$

Taking norm on both sides and separating mean and variance, we proceed as

$$\begin{aligned} \mathbb{E}\|\Delta \mathbf{x}\|^2 &= \mathbb{E}\left\| -\frac{\tilde{\eta}}{KS} \sum_{k,i \in \mathcal{S}} (g_i(\mathbf{y}_{i,k-1}) - g_i(\alpha_{i,k-1}) + \mathbf{c} - \mathbf{c}_i) \right\|^2 \\ &\leq \mathbb{E}\left\| -\frac{\tilde{\eta}}{KS} \sum_{k,i \in \mathcal{S}} (\nabla f_i(\mathbf{y}_{i,k-1}) + \mathbb{E}[\mathbf{c}] - \mathbb{E}[\mathbf{c}_i]) \right\|^2 + \frac{9\tilde{\eta}^2 \sigma^2}{KS} \\ &\leq \mathbb{E}\left[\frac{\tilde{\eta}^2}{KS} \sum_{k,i \in \mathcal{S}} \left\| \nabla f_i(\mathbf{y}_{i,k-1}) + \mathbb{E}[\mathbf{c}] - \mathbb{E}[\mathbf{c}_i] \right\|^2 \right] + \frac{9\tilde{\eta}^2 \sigma^2}{KS} \\ &= \frac{\tilde{\eta}^2}{KN} \sum_{k,i} \mathbb{E}\left\| (\nabla f_i(\mathbf{y}_{i,k-1}) - \nabla f_i(\mathbf{x})) + (\mathbb{E}[\mathbf{c}] - \nabla f(\mathbf{x})) + \nabla f(\mathbf{x}) - (\mathbb{E}[\mathbf{c}_i] - \nabla f_i(\mathbf{x})) \right\|^2 + \frac{9\tilde{\eta}^2 \sigma^2}{KS} \\ &\leq \frac{4\tilde{\eta}^2}{KN} \sum_{k,i} \mathbb{E}\|\nabla f_i(\mathbf{y}_{i,k-1}) - \nabla f_i(\mathbf{x})\|^2 + \frac{8\tilde{\eta}^2}{KN} \sum_{k,i} \mathbb{E}\|\nabla f_i(\alpha_{i,k-1}) - \nabla f_i(\mathbf{x})\|^2 \\ &\quad + 4\tilde{\eta}^2 \mathbb{E}\|\nabla f(\mathbf{x})\|^2 + \frac{9\tilde{\eta}^2 \sigma^2}{KS} \\ &\leq 4\tilde{\eta}^2 \beta^2 \mathcal{E}_r + 8\beta^2 \tilde{\eta}^2 \Xi_{r-1} + 4\tilde{\eta}^2 \mathbb{E}\|\nabla f(\mathbf{x})\|^2 + \frac{9\tilde{\eta}^2 \sigma^2}{KS}. \end{aligned}$$

In the first inequality, note that the three random variables— $\frac{1}{KS} \sum_{k,i \in \mathcal{S}} g_i(\mathbf{y}_{i,k})$, $\frac{1}{S} \sum_{i \in \mathcal{S}} \mathbf{c}_i$, and \mathbf{c} —may not be independent but each have variance smaller than $\frac{\sigma^2}{KS}$ and so we can ap-

ply Lemma 58. The rest of the inequalities follow from repeated applications of the relaxed triangle inequality, β -Lipschitzness of f_i , and the definition of Ξ_{r-1} (12.21). This proves the second statement. The first statement follows from our expression of $\mathbb{E}_{r-1}[\Delta \mathbf{x}]$ and similar computations. \square

LAG IN THE CONTROL VARIATES. WE NOW ANALYZE THE ‘LAG’ IN THE CONTROL VARIATES DUE TO US SAMPLING ONLY A SMALL SUBSET OF CLIENTS EACH ROUND. BECAUSE WE CANNOT RELY ON CONVEXITY ANYMORE BUT ONLY ON THE LIPSCHITZNESS OF THE GRADIENTS, THE CONTROL-LAG INCREASES FASTER IN THE NON-CONVEX CASE.

Lemma 71. *For updates (12.12)—(12.15) and assumptions A4, A5, the following holds true for any $\tilde{\eta} \leq \frac{1}{24\beta}(\frac{S}{N})^\alpha$ for $\alpha \in [\frac{1}{2}, 1]$ where $\tilde{\eta} := \eta_l \eta_g K$:*

$$\Xi_r \leq (1 - \frac{17S}{36N})\Xi_{r-1} + \frac{1}{48\beta^2}(\frac{S}{N})^{2\alpha-1}\|\nabla f(\mathbf{x}^{r-1})\|^2 + \frac{97}{48}(\frac{S}{N})^{2\alpha-1}\mathcal{E}_r + (\frac{S}{N\beta^2})\frac{\sigma^2}{32KS}.$$

Proof. The proof proceeds similar to that of Lemma 67 except that we cannot rely on convexity. Recall that after round r , the definition of $\boldsymbol{\alpha}_{i,k-1}^r$ (12.20) implies that

$$\mathbb{E}_{\mathcal{S}^r}[\boldsymbol{\alpha}_{i,k-1}^r] = (1 - \frac{S}{N})\boldsymbol{\alpha}_{i,k-1}^{r-1} + \frac{S}{N}\mathbf{y}_{i,k-1}^r.$$

Plugging the above expression in the definition of Ξ_r we get

$$\begin{aligned} \Xi_r &= \frac{1}{KN} \sum_{i,k} \mathbb{E} \|\boldsymbol{\alpha}_{i,k-1}^r - \mathbf{x}^r\|^2 \\ &= \left(1 - \frac{S}{N}\right) \cdot \underbrace{\frac{1}{KN} \sum_i \mathbb{E} \|\boldsymbol{\alpha}_{i,k-1}^{r-1} - \mathbf{x}^r\|^2}_{\mathcal{T}_5} + \underbrace{\frac{S}{N} \cdot \frac{1}{KN} \sum_{k,i} \mathbb{E} \|\mathbf{y}_{i,k-1}^r - \mathbf{x}^r\|^2}_{\mathcal{T}_6}. \end{aligned}$$

We can expand the second term \mathcal{T}_6 with the relaxed triangle inequality to claim

$$\mathcal{T}_6 \leq 2(\mathcal{E}_r + \mathbb{E} \|\Delta \mathbf{x}^r\|^2).$$

We will expand the first term \mathcal{T}_5 to claim for a constant $b \geq 0$ to be chosen later

$$\begin{aligned} \mathcal{T}_5 &= \frac{1}{KN} \sum_i \mathbb{E} (\|\boldsymbol{\alpha}_{i,k-1}^{r-1} - \mathbf{x}^{r-1}\|^2 + \|\Delta \mathbf{x}^r\|^2 + \mathbb{E}_{r-1} \langle \Delta \mathbf{x}^r, \boldsymbol{\alpha}_{i,k-1}^{r-1} - \mathbf{x}^{r-1} \rangle) \\ &\leq \frac{1}{KN} \sum_i \mathbb{E} (\|\boldsymbol{\alpha}_{i,k-1}^{r-1} - \mathbf{x}^{r-1}\|^2 + \|\Delta \mathbf{x}^r\|^2 + \frac{1}{b} \|\mathbb{E}_{r-1}[\Delta \mathbf{x}^r]\|^2 + b \|\boldsymbol{\alpha}_{i,k-1}^{r-1} - \mathbf{x}^{r-1}\|^2) \end{aligned}$$

where we used Young’s inequality which holds for any $b \geq 0$. Combining the bounds for \mathcal{T}_5

and \mathcal{T}_6 ,

$$\begin{aligned}\Xi_r &\leq \left(1 - \frac{S}{N}\right)(1+b)\Xi_{r-1} + 2\frac{S}{N}\mathcal{E}_r + 2\mathbb{E}\|\Delta\mathbf{x}^r\|^2 + \frac{1}{b}\mathbb{E}\|\mathbb{E}_{r-1}[\Delta\mathbf{x}^r]\|^2 \\ &\leq \left(\left(1 - \frac{S}{N}\right)(1+b) + 16\tilde{\eta}^2\beta^2\right)\Xi_{r-1} + \left(\frac{2S}{N} + 8\tilde{\eta}^2\beta^2 + 2\frac{1}{b}\tilde{\eta}^2\beta^2\right)\mathcal{E}_r + \left(8 + 2\frac{1}{b}\right)\tilde{\eta}^2\mathbb{E}\|\nabla f(\mathbf{x})\|^2 + \frac{18\tilde{\eta}^2\sigma^2}{KS}\end{aligned}$$

The last inequality applied Lemma 70. Verify that with choice of $b = \frac{S}{2(N-S)}$, we have $\left(1 - \frac{S}{N}\right)(1+b) \leq \left(1 - \frac{S}{2N}\right)$ and $\frac{1}{b} \leq \frac{2N}{S}$. Plugging these values along with the bound on the step-size $16\beta^2\tilde{\eta}^2 \leq \frac{1}{36}\left(\frac{S}{N}\right)^{2\alpha} \leq \frac{S}{36N}$ completes the lemma. \square

BOUNDING THE DRIFT. WE WILL NEXT BOUND THE CLIENT DRIFT \mathcal{E}_r . FOR THIS, CONVEXITY IS NOT CRUCIAL AND WE WILL RECOVER A VERY SIMILAR RESULT TO LEMMA 68 ONLY USE THE LIPSCHITZNESS OF THE GRADIENT.

Lemma 72. Suppose our step-sizes satisfy $\eta_l \leq \frac{1}{24\beta K\eta_g}$ and f_i satisfies assumptions A4–A5. Then, for any global $\eta_g \geq 1$ we can bound the drift as

$$\frac{5}{3}\beta^2\tilde{\eta}\mathcal{E}_r \leq \frac{5}{3}\beta^3\tilde{\eta}^2\Xi_{r-1} + \frac{\tilde{\eta}}{24\beta\eta_g^2}\mathbb{E}\|\nabla f(\mathbf{x}^{r-1})\|^2 + \frac{\tilde{\eta}^2\beta}{4K\eta_g^2}\sigma^2.$$

Proof. First, observe that if $K = 1$, $\mathcal{E}_r = 0$ since $\mathbf{y}_{i,0} = \mathbf{x}$ for all $i \in [N]$ and that Ξ_{r-1} and the right hand side are both positive. Thus the Lemma is trivially true if $K = 1$ and we will henceforth assume $K \geq 2$. Starting from the update rule (12.12) for $i \in [N]$ and $k \in [K]$

$$\begin{aligned}\mathbb{E}\|\mathbf{y}_{i,k} - \mathbf{x}\|^2 &= \mathbb{E}\|\mathbf{y}_{i,k-1} - \eta_l(g_i(\mathbf{y}_{i,k-1}) + \mathbf{c} - \mathbf{c}_i) - \mathbf{x}\|^2 \\ &\leq \mathbb{E}\|\mathbf{y}_{i,k-1} - \eta_l(\nabla f_i(\mathbf{y}_{i,k-1}) + \mathbf{c} - \mathbf{c}_i) - \mathbf{x}\|^2 + \eta_l^2\sigma^2 \\ &\leq \left(1 + \frac{1}{K-1}\right)\mathbb{E}\|\mathbf{y}_{i,k-1} - \mathbf{x}\|^2 + K\eta_l^2\mathbb{E}\|\nabla f_i(\mathbf{y}_{i,k-1}) + \mathbf{c} - \mathbf{c}_i\|^2 + \eta_l^2\sigma^2 \\ &= \left(1 + \frac{1}{K-1}\right)\mathbb{E}\|\mathbf{y}_{i,k-1} - \mathbf{x}\|^2 + \eta_l^2\sigma^2 \\ &\quad + K\eta_l^2\mathbb{E}\|\nabla f_i(\mathbf{y}_{i,k-1}) - \nabla f_i(\mathbf{x}) + (\mathbf{c} - \nabla f(\mathbf{x})) + \nabla f(\mathbf{x}) - (\mathbf{c}_i - \nabla f_i(\mathbf{x}))\|^2 \\ &\leq \left(1 + \frac{1}{K-1}\right)\mathbb{E}\|\mathbf{y}_{i,k-1} - \mathbf{x}\|^2 + 4K\eta_l^2\mathbb{E}\|\nabla f_i(\mathbf{y}_{i,k-1}) - \nabla f_i(\mathbf{x})\|^2 + \eta_l^2\sigma^2 \\ &\quad + 4K\eta_l^2\mathbb{E}\|\mathbf{c} - \nabla f(\mathbf{x})\|^2 + 4K\eta_l^2\mathbb{E}\|\nabla f(\mathbf{x})\|^2 + 4K\eta_l^2\mathbb{E}\|\mathbf{c}_i - \nabla f_i(\mathbf{x})\|^2 \\ &\leq \left(1 + \frac{1}{K-1} + 4K\beta^2\eta_l^2\right)\mathbb{E}\|\mathbf{y}_{i,k-1} - \mathbf{x}\|^2 + \eta_l^2\sigma^2 + 4K\eta_l^2\mathbb{E}\|\nabla f(\mathbf{x})\|^2 \\ &\quad + 4K\eta_l^2\mathbb{E}\|\mathbf{c} - \nabla f(\mathbf{x})\|^2 + 4K\eta_l^2\mathbb{E}\|\mathbf{c}_i - \nabla f_i(\mathbf{x})\|^2\end{aligned}$$

The inequalities above follow from repeated application of the relaxed triangle inequalities and the β -Lipschitzness of f_i . Averaging the above over i , the definition of $\mathbf{c} = \frac{1}{N}\sum_i \mathbf{c}_i$ and

Ξ_{r-1} (12.21) gives

$$\begin{aligned}
 \frac{1}{N} \sum_i \mathbb{E} \|\mathbf{y}_{i,k} - \mathbf{x}\|^2 &\leq \left(1 + \frac{1}{K-1} + 4K\beta^2\eta_l^2\right) \frac{1}{N} \sum_i \mathbb{E} \|\mathbf{y}_{i,k-1} - \mathbf{x}\|^2 \\
 &\quad + \eta_l^2 \sigma^2 + 4K\eta_l^2 \mathbb{E} \|\nabla f(\mathbf{x})\|^2 + 8K\eta_l^2 \beta^2 \Xi_{r-1} \\
 &\leq \left(\eta_l^2 \sigma^2 + 4K\eta_l^2 \mathbb{E} \|\nabla f(\mathbf{x})\|^2 + 8K\eta_l^2 \beta^2 \Xi_{r-1}\right) \left(\sum_{\tau=0}^{k-1} \left(1 + \frac{1}{K-1} + 4K\beta^2\eta_l^2\right)^\tau\right) \\
 &= \left(\frac{\tilde{\eta}^2 \sigma^2}{K^2 \eta_g^2} + \frac{4\tilde{\eta}^2}{K\eta_g^2} \mathbb{E} \|\nabla f(\mathbf{x})\|^2 + \frac{8\tilde{\eta}^2 \beta^2}{K\eta_g^2} \Xi_{r-1}\right) \left(\sum_{\tau=0}^{k-1} \left(1 + \frac{1}{K-1} + \frac{4\beta^2 \tilde{\eta}^2}{K\eta_g^2}\right)^\tau\right) \\
 &\leq \left(\frac{\tilde{\eta} \sigma^2}{24\beta K^2 \eta_g^2} + \frac{1}{144\beta^2 K \eta_g^2} \mathbb{E} \|\nabla f(\mathbf{x})\|^2 + \frac{\tilde{\eta} \beta}{3K\eta_g^2} \Xi_{r-1}\right) 3K.
 \end{aligned}$$

The last inequality used the bound on the step-size $\beta\tilde{\eta} \leq \frac{1}{24}$. Averaging over k and multiplying both sides by $\frac{5}{3}\beta^2\tilde{\eta}$ yields the lemma statement. \square

PROGRESS MADE IN EACH ROUND. GIVEN THAT WE CAN BOUND ALL SOURCES OF ERROR, WE CAN FINALLY PROVE THE PROGRESS MADE IN EACH ROUND.

Lemma 73. *Suppose the updates (12.12)–(12.15) satisfy assumptions A4–A5. For any effective step-size $\tilde{\eta} := K\eta_g\eta_l$ satisfying $\tilde{\eta} \leq \frac{1}{24\beta} \left(\frac{S}{N}\right)^{\frac{2}{3}}$,*

$$\left(\mathbb{E}[f(\mathbf{x}^r)] + 12\beta^3\tilde{\eta}^2 \frac{N}{S} \Xi_r\right) \leq \left(\mathbb{E}[f(\mathbf{x}^{r-1})] + 12\beta^3\tilde{\eta}^2 \frac{N}{S} \Xi_{r-1}\right) + \frac{5\beta\tilde{\eta}^2\sigma^2}{KS} \left(1 + \frac{S}{\eta_g^2}\right) - \frac{\tilde{\eta}}{14} \mathbb{E} \|\nabla f(\mathbf{x}^{r-1})\|^2.$$

Proof. Starting from the smoothness of f and taking conditional expectation gives

$$\mathbb{E}_{r-1}[f(\mathbf{x} + \Delta\mathbf{x})] \leq f(\mathbf{x}) + \langle \nabla f(\mathbf{x}), \mathbb{E}_{r-1}[\Delta\mathbf{x}] \rangle + \frac{\beta}{2} \mathbb{E}_{r-1} \|\Delta\mathbf{x}\|^2.$$

We as usual dropped the superscript everywhere. Recall that the server update can be written as

$$\Delta\mathbf{x} = -\frac{\tilde{\eta}}{KS} \sum_{k,i \in \mathcal{S}} (g_i(\mathbf{y}_{i,k-1}) + \mathbf{c} - \mathbf{c}_i), \text{ and } \mathbb{E}_{\mathcal{S}}[\Delta\mathbf{x}] = -\frac{\tilde{\eta}}{KN} \sum_{k,i} g_i(\mathbf{y}_{i,k-1}).$$

Substituting this in the previous inequality and applying Lemma 70 to bound $\mathbb{E}[\|\Delta \mathbf{x}\|^2]$ gives

$$\begin{aligned}
 \mathbb{E}[f(\mathbf{x} + \Delta \mathbf{x})] - f(\mathbf{x}) &\leq -\frac{\tilde{\eta}}{KN} \sum_{k,i} \langle \nabla f(\mathbf{x}), \mathbb{E}[\nabla f_i(\mathbf{y}_{i,k-1})] \rangle + \frac{\beta}{2} \mathbb{E}\|\Delta \mathbf{x}\|^2 \\
 &\leq -\frac{\tilde{\eta}}{KN} \sum_{k,i} \langle \nabla f(\mathbf{x}), \mathbb{E}[\nabla f_i(\mathbf{y}_{i,k-1})] \rangle + \\
 &\quad 2\tilde{\eta}^2 \beta^3 \mathcal{E}_r + 4\tilde{\eta}^2 \beta^3 \Xi_{r-1} + 2\beta \tilde{\eta}^2 \mathbb{E}\|\nabla f(\mathbf{x})\|^2 + \frac{9\beta \tilde{\eta}^2 \sigma^2}{2KS} \\
 &\leq -\frac{\tilde{\eta}}{2} \|\nabla f(\mathbf{x})\|^2 + \frac{\tilde{\eta}}{2} \sum_{i,k} \mathbb{E} \left\| \frac{1}{KN} \sum \nabla f_i(\mathbf{y}_{i,k-1}) - \nabla f(\mathbf{x}) \right\|^2 + \\
 &\quad 2\tilde{\eta}^2 \beta^3 \mathcal{E}_r + 4\tilde{\eta}^2 \beta^3 \Xi_{r-1} + 2\beta \tilde{\eta}^2 \mathbb{E}\|\nabla f(\mathbf{x})\|^2 + \frac{9\beta \tilde{\eta}^2 \sigma^2}{2KS} \\
 &\leq -\frac{\tilde{\eta}}{2} \|\nabla f(\mathbf{x})\|^2 + \frac{\tilde{\eta}}{2KN} \sum_{i,k} \mathbb{E} \left\| \nabla f_i(\mathbf{y}_{i,k-1}) - \nabla f_i(\mathbf{x}) \right\|^2 + \\
 &\quad 2\tilde{\eta}^2 \beta^3 \mathcal{E}_r + 4\tilde{\eta}^2 \beta^3 \Xi_{r-1} + 2\beta \tilde{\eta}^2 \mathbb{E}\|\nabla f(\mathbf{x})\|^2 + \frac{9\beta \tilde{\eta}^2 \sigma^2}{2KS} \\
 &\leq -(\frac{\tilde{\eta}}{2} - 2\beta \tilde{\eta}^2) \|\nabla f(\mathbf{x})\|^2 + (\frac{\tilde{\eta}}{2} + 2\beta \tilde{\eta}^2) \beta^2 \mathcal{E}_r + 4\beta^3 \tilde{\eta}^2 \Xi_{r-1} + \frac{9\beta \tilde{\eta}^2 \sigma^2}{2KS}.
 \end{aligned}$$

The third inequality follows from the observation that $-ab = \frac{1}{2}((b-a)^2 - a^2) - \frac{1}{2}b^2 \leq \frac{1}{2}((b-a)^2 - a^2)$ for any $a, b \in \mathbb{R}$, and the last from the β -Lipschitzness of f_i . Now we use Lemma 71 to bound Ξ_r as

$$\begin{aligned}
 12\beta^3 \tilde{\eta}^2 \frac{N}{S} \Xi_r &\leq 12\beta^3 \tilde{\eta}^2 \frac{N}{S} \left((1 - \frac{17S}{36N}) \Xi_{r-1} + \frac{1}{48\beta^2} (\frac{S}{N})^{2\alpha-1} \|\nabla f(\mathbf{x}^{r-1})\|^2 + \frac{97}{48} (\frac{S}{N})^{2\alpha-1} \mathcal{E}_r + (\frac{S}{N\beta^2}) \frac{\sigma^2}{32KS} \right) \\
 &= 12\beta^3 \tilde{\eta}^2 \frac{N}{S} \Xi_{r-1} - \frac{17}{3} \beta^3 \tilde{\eta}^2 \Xi_{r-1} + \frac{1}{4} \beta \tilde{\eta}^2 (\frac{N}{S})^{2-2\alpha} \|\nabla f(\mathbf{x})\|^2 + \frac{97}{4} \beta^3 \tilde{\eta}^2 (\frac{N}{S})^{2-2\alpha} \mathcal{E}_r + \frac{3\beta \tilde{\eta}^2 \sigma^2}{8KS}.
 \end{aligned}$$

Also recall that Lemma 72 states that

$$\frac{5}{3} \beta^2 \tilde{\eta} \mathcal{E}_r \leq \frac{5}{3} \beta^3 \tilde{\eta}^2 \Xi_{r-1} + \frac{\tilde{\eta}}{24\eta_g^2} \mathbb{E}\|\nabla f(\mathbf{x}^{r-1})\|^2 + \frac{\tilde{\eta}^2 \beta}{4K\eta_g^2} \sigma^2.$$

Adding these bounds on Ξ_r and \mathcal{E}_r to that of $\mathbb{E}[f(\mathbf{x} + \Delta \mathbf{x})]$ gives

$$\begin{aligned}
 (\mathbb{E}[f(\mathbf{x} + \Delta \mathbf{x})] + 12\beta^3 \tilde{\eta}^2 \frac{N}{S} \Xi_r) &\leq (\mathbb{E}[f(\mathbf{x})] + 12\beta^3 \tilde{\eta}^2 \frac{N}{S} \Xi_{r-1}) + (\frac{5}{3} - \frac{17}{3}) \beta^3 \tilde{\eta}^2 \Xi_{r-1} \\
 &\quad - (\frac{\tilde{\eta}}{2} - 2\beta \tilde{\eta}^2 - \frac{1}{4} \beta \tilde{\eta}^2 (\frac{N}{S})^{2-2\alpha}) \|\nabla f(\mathbf{x})\|^2 + (\frac{\tilde{\eta}}{2} - \frac{5\tilde{\eta}}{3} + 2\beta \tilde{\eta}^2 + \frac{97}{4} \beta \tilde{\eta}^2 (\frac{N}{S})^{2-2\alpha}) \beta^2 \mathcal{E}_r + \frac{39\beta \tilde{\eta}^2 \sigma^2}{8KS} (1 + \frac{S}{\eta_g^2}).
 \end{aligned}$$

By our choice of $\alpha = \frac{2}{3}$ and plugging in the bound on step-size $\beta \tilde{\eta} (\frac{N}{S})^{2-2\alpha} \leq \frac{1}{24}$ proves the lemma. \square

THE **NON-CONVEX RATE** OF CONVERGENCE NOW FOLLOWS BY UNROLLING THE RECURSION IN LEMMA 73 AND SELECTING AN APPROPRIATE STEP-SIZE $\tilde{\eta}$ AS IN LEMMA 56. FINALLY NOTE THAT IF WE INITIALIZE $\mathbf{c}_i^0 = \mathbf{g}_i(\mathbf{x}^0)$ THEN WE HAVE $\Xi_0 = 0$.

12.6 Usefulness of local steps (Theorem IX)

LET US STATE OUR RATES OF CONVERGENCE FOR SCAFFOLD WHICH INTERPOLATES BETWEEN IDENTICAL AND COMPLETELY HETEROGENEOUS CLIENTS. IN THIS SECTION, WE ALWAYS SET $\eta_g = 1$ AND ASSUME ALL CLIENTS PARTICIPATE ($S = N$).

Theorem XXXIII. *Suppose that the functions $\{f_i\}$ are quadratic and satisfy assumptions A4, A5 and additionally A2. Then, for global step-size $\eta_g = 1$ in each of the following cases, there exist probabilities $\{p_k^r\}$ and local step-size η_l such that the output (12.23) of SCAFFOLD WHEN RUN WITH NO CLIENT SAMPLING ($S = N$) USING UPDATE (12.22) SATISFIES:*

- **STRONGLY CONVEX:** f_i SATISFIES (A3) FOR $\mu > 0$, $\eta_l \leq \min(\frac{1}{10\beta}, \frac{1}{22\delta K}, \frac{1}{10\mu K})$, $R \geq \max(\frac{20\beta}{\mu}, \frac{44\delta K + 20\mu K}{\mu}, 20K)$ THEN

$$\mathbb{E}[\|\nabla f(\bar{\mathbf{x}}^R)\|^2] \leq \tilde{\mathcal{O}}\left(\frac{\beta\sigma^2}{\mu RKN} + \mu D^2 \exp\left(-\frac{\mu}{20\beta + 44\delta K + 20\mu K} RK\right)\right).$$

- **GENERAL CONVEX:** f SATISFIES $\nabla^2 f \succeq -\delta I$, $\eta_l \leq \min(\frac{1}{10\beta}, \frac{1}{22\delta K})$, AND $R \geq 1$, THEN

$$\mathbb{E}[\|\nabla f(\bar{\mathbf{x}}^R)\|^2] \leq \mathcal{O}\left(\frac{\sigma\sqrt{\beta(f(\mathbf{x}_0) - f^*)}}{\sqrt{RKN}} + \frac{(\beta + \delta K)(f(\mathbf{x}_0) - f^*)}{RK}\right).$$

NOTE THAT IF $\delta = 0$, WE MATCH (UP TO ACCELERATION) THE LOWER BOUND IN (WOODWORTH ET AL., 2018). WHILE CERTAINLY $\delta = 0$ WHEN THE FUNCTIONS ARE IDENTICAL AS STUDIED IN (WOODWORTH ET AL., 2018), OUR UPPER-BOUND IS SIGNIFICANTLY STRONGER SINCE IT IS POSSIBLE THAT $\delta = 0$ EVEN FOR HIGHLY HETEROGENEOUS FUNCTIONS. FOR EXAMPLE, OBJECTIVE PERTURBATION (CHAUDHURI ET AL., 2011; KIFER ET AL., 2012) IS AN OPTIMAL MECHANISM TO ACHIEVE DIFFERENTIAL PRIVACY FOR SMOOTH CONVEX OBJECTIVES (BASSILY ET AL., 2014). INTUITIVELY, OBJECTIVE PERTURBATION RELIES ON MASKING EACH CLIENT’S GRADIENTS BY ADDING A LARGE RANDOM LINEAR TERM TO THE OBJECTIVE FUNCTION. IN SUCH A CASE, WE WOULD HAVE HIGH GRADIENT DISSIMILARITY BUT NO HESSIAN DISSIMILARITY.

OUR NON-CONVEX CONVERGENCE RATES ARE THE FIRST OF THEIR KIND AS FAR AS WE ARE AWARE—NO PREVIOUS WORK SHOWS HOW ONE CAN TAKE ADVANTAGE OF SIMILARITY FOR NON-CONVEX FUNCTIONS. HOWEVER, WE SHOULD NOTE THAT NON-CONVEX QUADRATICS DO NOT HAVE A GLOBAL LOWER-BOUND ON THE FUNCTION VALUE f^* . WE WILL INSTEAD ASSUME THAT f^* ALMOST SURELY LOWER-BOUNDS THE VALUE OF $f(\mathbf{x}^R)$, IMPLICITLY ASSUMING THAT THE ITERATES REMAIN BOUNDED.

OUTLINE. IN THE REST OF THIS SECTION, WE WILL FOCUS ON PROVING THEOREM XXXIII. WE WILL SHOW HOW TO BOUND VARIANCE IN LEMMA 77, BOUND THE AMOUNT OF DRIFT IN LEMMA 76, AND SHOW PROGRESS MADE IN ONE STEP IN LEMMA 78. IN ALL OF THESE WE DO NOT USE CONVEXITY, BUT STRONGLY RELY ON THE FUNCTIONS BEING QUADRATICS. THEN WE COMBINE THESE TO DERIVE THE PROGRESS MADE BY THE SERVER IN ONE ROUND—FOR THIS WE NEED *weak*-CONVEXITY TO ARGUE THAT AVERAGING THE PARAMETERS DOES NOT HURT CON-

VERGENCE TOO MUCH. AS BEFORE, IT IS STRAIGHT-FORWARD TO DERIVE RATES OF CONVERGENCE FROM THE ONE-ROUND PROGRESS USING LEMMAS 55 AND 56.

12.6.1 Additional notation and assumptions

FOR ANY MATRIX M AND VECTOR \mathbf{v} , LET $\|\mathbf{v}\|_M^2 := \mathbf{v}^\top M \mathbf{v}$. SINCE ALL FUNCTIONS IN THIS SECTION ARE QUADRATICS, WE CAN ASSUME W.L.O.G THEY ARE OF THE FOLLOWING FORM:

$$f_i(\mathbf{x}) - f_i(\mathbf{x}_i^*) = \frac{1}{2} \|\mathbf{x} - \mathbf{x}_i^*\|_{A_i}^2 \text{ FOR } i \in [N], \text{ AND } f(\mathbf{x}) = \frac{1}{2} \|\mathbf{x} - \mathbf{x}^*\|_A^2, \text{ FOR ALL } \mathbf{x},$$

FOR SOME $\{\mathbf{x}_i^*\}$ AND \mathbf{x}^* , $A := \frac{1}{N} \sum_{i=1}^N A_i$. WE ALSO ASSUME THAT A IS A SYMMETRIC MATRIX THOUGH THIS REQUIREMENT IS EASILY RELAXED. NOTE THAT THIS IMPLIES $f(\mathbf{x}^*) = 0$ AND THAT $\nabla f_i(\mathbf{x}) = A(\mathbf{x} - \mathbf{x}_i^*)$. IF $\{f_i\}$ ARE ADDITIONALLY CONVEX, WE HAVE THAT \mathbf{x}_i^* IS THE OPTIMUM OF f_i AND \mathbf{x}^* THE OPTIMUM OF f . HOWEVER, THIS IS NOT NECESSARILY TRUE IN GENERAL.

WE WILL ALSO FOCUS ON A SIMPLIFIED VERSION OF SCAFFOLD WHERE IN EACH ROUND r , CLIENT i PERFORMS THE FOLLOWING UPDATE STARTING FROM $\mathbf{y}_{i,0}^r \leftarrow \mathbf{x}^{r-1}$:

$$\begin{aligned} \mathbf{y}_{i,k}^r &= \mathbf{y}_{i,k-1}^r - \eta(g_i(\mathbf{y}_{i,k-1}^r) + \nabla f(\mathbf{x}^{r-1}) - \nabla f_i(\mathbf{x}^{r-1})), \text{ I.E.} \\ \mathbb{E}_{r-1,k-1}[\mathbf{y}_{i,k}^r] &= \mathbf{y}_{i,k-1}^r - \eta A(\mathbf{y}_{i,k-1}^r - \mathbf{x}^*) - \eta(A_i - A)(\mathbf{y}_{i,k-1}^r - \mathbf{x}^{r-1}), \end{aligned} \quad (12.22)$$

WHERE THE SECOND PART IS SPECIALIZED TO QUADRATICS AND THE EXPECTATION IS CONDITIONED OVER EVERYTHING BEFORE CURRENT STEP k OF ROUND r . AT THE END OF EACH ROUND, AS BEFORE, $\mathbf{x}^r = \frac{1}{N} \sum_{i=1}^N \mathbf{y}_{i,K}^r$. THE FINAL OUTPUT OF THE ALGORITHM IS CHOSEN USING PROBABILITIES $\{p_k^r\}$ AS

$$\bar{\mathbf{x}}^R = \mathbf{x}_k^r \text{ WITH PROBABILITY } p_k^r, \text{ WHERE } \mathbf{x}_k^r := \frac{1}{N} \sum_{i=1}^N \mathbf{y}_{i,k}^r. \quad (12.23)$$

NOTE THAT WE ARE NOW POSSIBLY OUTPUTTING ITERATES COMPUTED WITHIN A SINGLE ROUND AND THAT $\mathbf{x}^r = \mathbf{x}_K^r$. BEYOND THIS, THE UPDATE ABOVE DIFFERS FROM OUR USUAL SCAFFOLD IN TWO KEY ASPECTS: A) IT USES GRADIENTS COMPUTED AT \mathbf{x}^{r-1} AS CONTROL VARIATES INSTEAD OF THOSE AT EITHER \mathbf{x}^{r-2} (AS IN OPTION I) OR $\mathbf{y}_{i,k}^{r-1}$ (AS IN OPTION II), AND B) IT USES FULL BATCH GRADIENTS TO COMPUTE ITS CONTROL VARIATES INSTEAD OF STOCHASTIC GRADIENTS. THE FIRST ISSUE IS EASY TO FIX AND OUR PROOF EXTENDS TO USING BOTH OPTION I OR OPTION II USING TECHNIQUES IN SECTION 12.5. THE SECOND ISSUE IS MORE TECHNICAL—USING STOCHASTIC GRADIENTS FOR CONTROL VARIATES COUPLES THE RANDOMNESS ACROSS THE CLIENTS IN MAKING THE LOCAL-UPDATES *biased*. WHILE IT MAY BE POSSIBLE TO GET AROUND THIS (CF. (LEI AND JORDAN, 2017; NGUYEN ET AL., 2017; TRAN-DINH ET AL., 2019)), WE WILL NOT ATTEMPT TO DO SO IN THIS WORK. NOTE THAT IF K LOCAL UPDATE STEPS TYPICALLY REPRESENTS RUNNING MULTIPLE EPOCHS ON EACH CLIENT. HENCE ONE ADDITIONAL EPOCH TO COMPUTE THE CONTROL VARIATE $\nabla f_i(\mathbf{x})$ DOES NOT SIGNIFICANTLY ADD TO THE COST.

FINALLY, WE DEFINE THE FOLLOWING SEQUENCE OF POSITIVE NUMBERS FOR NOTATION CONVENIENCE:

$$\begin{aligned}\xi_{i,k}^r &:= \left(\mathbb{E}_{r-1}[f(\mathbf{y}_{i,k}^r)] - f(\mathbf{x}^*) + \delta(1 + \frac{1}{K})^{K-k} \mathbb{E}_{r-1} \|\mathbf{y}_{i,k}^r - \mathbf{x}^{r-1}\|^2 \right), \text{ AND} \\ \tilde{\xi}_{i,k}^r &:= \left([f(\mathbb{E}_{r-1}[\mathbf{y}_{i,k}^r])] - f(\mathbf{x}^*) + \delta(1 + \frac{1}{K})^{K-k} \mathbb{E}_{r-1,k-1} \|\mathbb{E}_{r-1}[\mathbf{y}_{i,k}^r] - \mathbf{x}^{r-1}\|^2 \right).\end{aligned}$$

OBSERVE THAT FOR $k = 0$, $\xi_{i,0}^r = \tilde{\xi}_{i,0}^r = f(\mathbf{x}^{r-1}) - f(\mathbf{x}^*)$.

12.6.2 Lemmas tracking errors

EFFECT OF AVERAGING. WE SEE HOW AVERAGING CAN REDUCE VARIANCE. A SIMILAR ARGUMENT WAS USED IN THE SPECIAL CASE OF ONE-SHOT AVERAGING IN (ZHANG ET AL., 2013B).

Lemma 74. *Suppose $\{f_i\}$ are quadratic functions and assumption A4 is satisfied. Then let \mathbf{x}_k^r and $\mathbf{y}_{i,k}^r$ be vectors in step k and round r generated using (12.22)–(12.23). Then,*

$$\mathbb{E}_{r-1} \|\nabla f(\mathbf{x}_k^r)\|^2 \leq \frac{1}{N} \sum_{i=1}^N \|\nabla f(\mathbb{E}_{r-1}[\mathbf{y}_{i,k}^r])\|^2 + \frac{1}{N^2} \sum_{i=1}^N \mathbb{E}_{r-1} [\|\nabla f(\mathbf{y}_{i,k}^r)\|^2].$$

Proof. Observe that the variables $\{\mathbf{y}_{i,k} - \mathbf{x}\}$ are independent of each other (the only source of randomness is the local gradient computations). The rest of the proof is exactly that of Lemma 58. Dropping superscripts everywhere,

$$\begin{aligned}\mathbb{E}_{r-1} \|A(\mathbf{x}_k^r - \mathbf{x}^*)\|^2 &= \mathbb{E}_{r-1} \left\| \frac{1}{N} \sum_i A(\mathbf{y}_{i,k} - \mathbf{x}^*) \right\|^2 \\ &= \mathbb{E}_{r-1} \left\| \frac{1}{N} \sum_i A(\mathbb{E}_{r-1}[\mathbf{y}_{i,k}] - \mathbf{x}^*) \right\|^2 + \mathbb{E}_{r-1} \left\| \frac{1}{N} \sum_i A(\mathbb{E}_{r-1}[\mathbf{y}_{i,k}] - \mathbf{y}_{i,k}) \right\|^2 \\ &= \mathbb{E}_{r-1} \left\| \frac{1}{N} \sum_i A(\mathbb{E}_{r-1}[\mathbf{y}_{i,k}] - \mathbf{x}^*) \right\|^2 + \frac{1}{N^2} \sum_i \mathbb{E}_{r-1} \|A(\mathbb{E}_{r-1}[\mathbf{y}_{i,k}] - \mathbf{y}_{i,k})\|^2 \\ &= \mathbb{E}_{r-1} \left\| \frac{1}{N} \sum_i A(\mathbb{E}_{r-1}[\mathbf{y}_{i,k}] - \mathbf{x}^*) \right\|^2 + \frac{1}{N^2} \sum_i \mathbb{E}_{r-1} \|A(\mathbf{y}_{i,k} - \mathbf{x}^* - \mathbb{E}_{r-1}[\mathbf{y}_{i,k} - \mathbf{x}^*])\|^2 \\ &\leq \mathbb{E}_{r-1} \left\| \frac{1}{N} \sum_i A(\mathbb{E}_{r-1}[\mathbf{y}_{i,k}] - \mathbf{x}^*) \right\|^2 + \frac{1}{N^2} \sum_i \mathbb{E}_{r-1} \|A(\mathbf{y}_{i,k} - \mathbf{x}^*)\|^2.\end{aligned}$$

The third equality was because $\{\mathbf{y}_{i,k}\}$ are independent of each other conditioned on everything before round r . \square

WE NEXT SEE THE EFFECT OF AVERAGING ON FUNCTION VALUES.

Lemma 75. *Suppose that f is δ general-convex, then we have:*

$$\frac{1}{N} \sum_{i=1}^n \xi_{i,k}^r \geq \mathbb{E}_{r-1}[f(\mathbf{x}_k^r)] - f(\mathbf{x}^*), \text{ and } \frac{1}{N} \sum_{i=1}^n \tilde{\xi}_{i,k}^r \geq f(\mathbb{E}_{r-1}[\mathbf{x}_k^r]) - f(\mathbf{x}^*).$$

Proof. Since f is δ -general convex, it follows that the function $f(\mathbf{z}) + \delta(1 + \frac{1}{K})^{K-k} \|\mathbf{z} - \mathbf{x}\|_2^2$ is convex in \mathbf{z} for any $k \in [K]$. The lemma now follows directly from using convexity and the definition of $\mathbf{x}_k^r = \frac{1}{N} \mathbf{y}_{i,k}^r$. \square

BOUNDING DRIFT OF ONE CLIENT. WE SEE HOW THE CLIENT DRIFT OF SCAFFOLD DEPENDS ON δ .

Lemma 76. *For the update (12.22), assuming (A2) and that $\{f_i\}$ are quadratics, the following holds for any $\eta \leq \frac{1}{21\delta K}$*

$$\mathbb{E}_{r-1,k-1} \|\mathbf{y}_{i,k}^r - \mathbf{x}^{r-1}\|^2 \leq (1 + \frac{1}{2K}) \|\mathbf{y}_{i,k-1}^r - \mathbf{x}^{r-1}\|^2 + 7K\eta^2 \|\nabla f(\mathbf{y}_{i,k-1}^r)\|^2 + \eta^2 \sigma^2.$$

Proof. Starting from the update step (12.22)

$$\begin{aligned} \mathbb{E}_{r-1,k-1} \|\mathbf{y}_i^+ - \mathbf{x}\|^2 &\leq \|\mathbf{y}_i - \mathbf{x} - \eta A(\mathbf{y}_i - \mathbf{x}^*) - \eta(A_i - A)(\mathbf{y}_i - \mathbf{x})\|^2 + \eta^2 \sigma^2 \\ &\leq (1 + \frac{1}{7(K-1)}) \|(I - \eta(A_i - A))(\mathbf{y}_i - \mathbf{x})\|^2 + 7K\eta^2 \|A(\mathbf{y}_i - \mathbf{x}^*)\|^2 + \eta^2 \sigma^2. \end{aligned}$$

Note that if $K = 1$, then the first inequality directly proves the lemma. For the second inequality, we assumed $K \geq 2$ and then applied our relaxed triangle inequality. By assumption A2, we have the following for $\eta\delta \leq 1$

$$\|(I - \eta(A_i - A))^2\| = \|I - \eta(A_i - A)\|^2 \leq (1 + \eta\delta)^2 \leq 1 + 3\eta\delta.$$

Using the bound on the step-size $\eta \leq \frac{1}{21\delta K}$ gives

$$\mathbb{E}_{r-1,k-1} \|\mathbf{y}_i^+ - \mathbf{x}\|^2 \leq (1 + \frac{1}{7K})(1 + \frac{1}{7(K-1)}) \|\mathbf{y}_i - \mathbf{x}\|^2 + 7K\eta^2 \|A(\mathbf{y}_i - \mathbf{x}^*)\|^2 + \eta^2 \sigma^2$$

Simple computations now give the Lemma statement for all $K \geq 1$. \square

TRACKING THE VARIANCE. WE WILL SEE HOW TO BOUND THE VARIANCE OF THE OUTPUT.

Lemma 77. *Consider the update (12.22) for quadratic $\{f_i\}$ with $\eta \leq \max(\frac{1}{2\delta K}, \frac{1}{\beta})$. Then, if further (A2), (A5) and (A4) are satisfied, we have*

$$\mathbb{E}_{r-1} f(\mathbf{x}^r) \leq f(\mathbb{E}_{r-1}[\mathbf{x}^r]) + 3K\beta \frac{\sigma^2}{N}.$$

Further if $\{f_i\}$ are strongly convex satisfying (A3), we have

$$\mathbb{E}_{r-1} f(\mathbf{x}^r) \leq f(\mathbb{E}_{r-1}[\mathbf{x}^r]) + \beta \frac{\sigma^2}{N} \sum_{k=1}^K (1 - \mu\eta)^{k-1}.$$

Proof. We can rewrite the update step (12.22) as below:

$$\mathbf{y}_{i,k} = \mathbf{y}_{i,k-1} - \eta(A_i(\mathbf{y}_{i,k-1} - \mathbf{x}^*) + (A - A_i)(\mathbf{x} - \mathbf{x}^*)) - \eta\zeta_{i,k},$$

where by the bounded variance assumption A4, $\zeta_{i,k}$ is a random variable satisfying $\mathbb{E}_{k-1,r-1}[\zeta_{i,k}] = 0$ and $\mathbb{E}_{k-1,r-1}\|\zeta_{i,k}\|^2 \leq \sigma^2$. Subtracting \mathbf{x}^\star from both sides and unrolling the recursion gives

$$\begin{aligned} \mathbf{y}_{i,K} - \mathbf{x}^\star &= (I - \eta A_i)(\mathbf{y}_{i,K-1} - \mathbf{x}^\star) - \eta((A - A_i)(\mathbf{x} - \mathbf{x}^\star) + \zeta_{i,K}) \\ &= (I - \eta A_i)^K(\mathbf{x} - \mathbf{x}^\star) - \sum_{k=1}^K \eta(I - \eta A_i)^{k-1}(\zeta_{i,k} + (A - A_i)(\mathbf{x} - \mathbf{x}^\star)). \end{aligned}$$

Similarly, the expected iterate satisfies the same equation without the $\zeta_{i,k}$

$$\mathbb{E}_{r-1}[\mathbf{y}_{i,K}] - \mathbf{x}^\star = (I - \eta A_i)^K(\mathbf{x} - \mathbf{x}^\star) - \sum_{k=1}^K \eta(I - \eta A_i)^{k-1}(A - A_i)(\mathbf{x} - \mathbf{x}^\star).$$

This implies that the difference satisfies

$$\mathbb{E}_{r-1}[\mathbf{y}_{i,K}] - \mathbf{y}_{i,K} = \eta \sum_{k=1}^K (I - \eta A_i)^{k-1} \zeta_{i,k}.$$

We can relate this to the function value as follows:

$$\begin{aligned} \mathbb{E}_{r-1}\|\mathbf{x}_K^r - \mathbf{x}^\star\|_A^2 &= \|\mathbb{E}_{r-1}[\mathbf{x}_K^r] - \mathbf{x}^\star\|_A^2 + \mathbb{E}_{r-1}\|\mathbb{E}_{r-1}[\mathbf{x}_K^r] - \mathbf{x}_K^r\|_A^2 \\ &= \|\mathbb{E}_{r-1}[\mathbf{x}_K^r] - \mathbf{x}^\star\|_A^2 + \mathbb{E}_{r-1}\|\frac{1}{N} \sum_i (\mathbb{E}_{r-1}[\mathbf{y}_{i,K}] - \mathbf{y}_{i,K})\|_A^2 \\ &= \|\mathbb{E}_{r-1}[\mathbf{x}_K^r] - \mathbf{x}^\star\|_A^2 + \eta^2 \mathbb{E}_{r-1}\|\frac{1}{N} \sum_{i,k} (I - \eta A_i)^{k-1} \zeta_{i,k}\|_A^2 \\ &= \|\mathbb{E}_{r-1}[\mathbf{x}_K^r] - \mathbf{x}^\star\|_A^2 + \frac{\eta^2}{N^2} \mathbb{E}_{r-1} \sum_{i,k} \|(I - \eta A_i)^{k-1} \zeta_{i,k}\|_A^2 \\ &\leq \|\mathbb{E}_{r-1}[\mathbf{x}_K^r] - \mathbf{x}^\star\|_A^2 + \frac{\beta \eta^2}{N^2} \mathbb{E}_{r-1} \sum_{i,k} \|(I - \eta A_i)^{k-1} \zeta_{i,k}\|_2^2. \end{aligned}$$

The last inequality used smoothness of f and the one before that relied on the independence of $\zeta_{i,k}$. Now, if f_i is general convex we have for $\eta \leq \frac{1}{2\delta K}$ that $I - \eta A_i \leq (1 + \frac{1}{2K})I$ and hence

$$\|(I - \eta A_i)^{k-1} \zeta_{i,k}\|_2^2 \leq \sigma^2 (1 + \frac{1}{2K})^{2(k-1)} \leq 3\sigma^2.$$

This proves our second statement of the lemma. For strongly convex functions, we have for $\eta \leq \frac{1}{\beta}$,

$$\|(I - \eta A_i)^{k-1} \zeta_{i,k}\|_2^2 \leq \sigma^2 (1 - \eta \mu)^{2(k-1)} \leq \sigma^2 (1 - \eta \mu)^{k-1}.$$

□

12.6.3 Lemmas showing progress

PROGRESS OF ONE CLIENT IN ONE STEP. NOW WE FOCUS ONLY ON A SINGLE CLIENT AND MONITOR THEIR PROGRESS.

Lemma 78. Suppose (A2), (A5) and (A4) hold, and $\{f_i\}$ are quadratics. Then, the following holds for the update (12.22) with $\eta \leq \min(\frac{1}{10\beta}, \frac{1}{22\delta K}, \frac{1}{\mu K})$ with $\mu = 0$ if f is non-convex or

general-convex

$$\begin{aligned}\xi_{i,k}^r &\leq (1 - \frac{\mu\eta}{6})\xi_{i,k-1}^r - \frac{\eta}{6}\mathbb{E}_{r-1}\|\nabla f(\mathbf{y}_{i,k-1}^r)\|^2 + 7\beta\eta^2\sigma^2, \text{ and} \\ \tilde{\xi}_{i,k}^r &\leq (1 - \frac{\mu\eta}{6})\tilde{\xi}_{i,k-1}^r - \frac{\eta}{6}\|\nabla f(\mathbb{E}_{r-1}[\mathbf{y}_{i,k-1}^r])\|^2.\end{aligned}$$

Proof. Recall that $\xi_{i,k}^r \geq 0$ is defined to be

$$\xi_{i,k}^r := \left(\mathbb{E}_{r-1}[f(\mathbf{y}_{i,k}^r)] - f(\mathbf{x}^\star) + \delta(1 + \frac{1}{K})^{K-k}\mathbb{E}_{r-1}\|\mathbf{y}_{i,k}^r - \mathbf{x}^{r-1}\|^2 \right).$$

Let us start from the local update step (12.22) (dropping unnecessary subscripts and superscripts)

$$\begin{aligned}\mathbb{E}_{r-1,k-1}\|\mathbf{y}_i^+ - \mathbf{x}^\star\|_A^2 &\leq \|\mathbf{y}_i - \mathbf{x}^\star\|_A^2 - 2\eta\langle A(\mathbf{y}_i - \mathbf{x}^\star), A(\mathbf{y}_i - \mathbf{x}^\star) \rangle + 2\eta\langle (A - A_i)(\mathbf{y}_i - \mathbf{x}), A(\mathbf{y}_i - \mathbf{x}^\star) \rangle \\ &\quad + \eta^2\|A(\mathbf{y}_i - \mathbf{x}^\star) + (A_i - A)(\mathbf{y}_i - \mathbf{x})\|_A^2 + \beta\eta^2\sigma^2 \\ &\leq \|\mathbf{y}_i - \mathbf{x}^\star\|_A^2 - \frac{3\eta}{2}\|A(\mathbf{y}_i - \mathbf{x}^\star)\|_2^2 + 2\eta\|(A - A_i)(\mathbf{y}_i - \mathbf{x})\|_2^2 \\ &\quad + 2\eta^2\|A(\mathbf{y}_i - \mathbf{x}^\star)\|_A^2 + 2\eta^2\|(A_i - A)(\mathbf{y}_i - \mathbf{x})\|_A^2 + \beta\eta^2\sigma^2 \\ &\leq \|\mathbf{y}_i - \mathbf{x}^\star\|_A^2 - (\frac{3\eta}{2} - 2\eta^2\beta)\|A(\mathbf{y}_i - \mathbf{x}^\star)\|_2^2 + \beta\eta^2\sigma^2 + \delta^2(2\eta^2\beta + 2\eta)\|\mathbf{y}_i - \mathbf{x}\|_2^2 \\ &\leq \|\mathbf{y}_i - \mathbf{x}^\star\|_A^2 - (\frac{3\eta}{2} - 2\eta^2\beta)\|A(\mathbf{y}_i - \mathbf{x}^\star)\|_2^2 + \beta\eta^2\sigma^2 + \frac{\delta}{10K}\|\mathbf{y}_i - \mathbf{x}\|_2^2.\end{aligned}$$

The second to last inequality used that $\|\cdot\|_A^2 \leq \beta\|\cdot\|_2^2$ by (A5) and that $\|(A - A_i)(\cdot)\|_2^2 \leq \delta^2\|\cdot\|_2^2$ by (A2). The final inequality used that $\eta \leq \max(\frac{1}{10\beta}, \frac{1}{22\delta K})$. Now, multiplying Lemma 76 by $\delta(1 + \frac{1}{K})^{K-k} \leq \frac{20\delta}{7}$ we have

$$\begin{aligned}\delta(1 + \frac{1}{K})^{K-k}\mathbb{E}_{r-1,k-1}\|\mathbf{y}_i^+ - \mathbf{x}\|^2 &\leq \delta(1 + \frac{1}{K})^{K-k}(1 + \frac{1}{2K})\|\mathbf{y}_i - \mathbf{x}\|^2 + 20\delta K\eta^2\|A(\mathbf{y}_i - \mathbf{x}^\star)\|^2 + 3\delta\eta^2\sigma^2 \\ &\leq \delta(1 + \frac{1}{K})^{K-k}(1 + \frac{1}{2K} + \frac{1}{10K})\|\mathbf{y}_i - \mathbf{x}\|^2 - \frac{\delta}{10K}\|\mathbf{y}_i - \mathbf{x}\|^2 \\ &\quad + 20\delta K\eta^2\|A(\mathbf{y}_i - \mathbf{x}^\star)\|^2 + 3\delta\eta^2\sigma^2 \\ &\leq (1 - \frac{1}{5K})\delta(1 + \frac{1}{K})^{K-k+1}(1 + \frac{1}{K})\|\mathbf{y}_i - \mathbf{x}\|^2 - \frac{\delta}{10K}\|\mathbf{y}_i - \mathbf{x}\|^2 \\ &\quad + 20\delta K\eta^2\|A(\mathbf{y}_i - \mathbf{x}^\star)\|^2 + 3\delta\eta^2\sigma^2.\end{aligned}$$

Adding this to our previous equation gives the following recursive bound:

$$\begin{aligned}&\left(\mathbb{E}_{r-1,k-1}\|\mathbf{y}_i^+ - \mathbf{x}^\star\|_A^2 + \delta(1 + \frac{1}{K})^{K-k}\mathbb{E}_{r-1,k-1}\|\mathbf{y}_i^+ - \mathbf{x}\|^2 \right) \leq \\ &\left(\|\mathbf{y}_i - \mathbf{x}^\star\|_A^2 + (1 - \frac{1}{5K})\delta(1 + \frac{1}{K})^{K-k+1}\|\mathbf{y}_i - \mathbf{x}\|^2 \right) - (\frac{3\eta}{2} - 2\eta^2\beta - 20\delta K\eta^2)\|A(\mathbf{y}_i - \mathbf{x}^\star)\|_2^2 + (3\delta + \beta)\eta^2\sigma^2\end{aligned}$$

The bound on our step-size $\eta \leq \min(\frac{1}{10\beta}, \frac{1}{22\delta K})$ implies that $\frac{3\eta}{2} - 2\eta^2\beta - 20\delta K\eta^2 \geq \frac{\eta}{3}$ and recall that $\delta \leq 2\beta$. This proves first statement of the lemma for non-strongly convex functions ($\mu = 0$). If additionally f is strongly-convex with $\mu > 0$, we have

$$\eta\|A(\mathbf{y}_i - \mathbf{x}^\star)\|_2^2 \geq \frac{\mu\eta}{2}\|\mathbf{y}_i - \mathbf{x}^\star\|_A^2 + \frac{\eta}{2}\|A(\mathbf{y}_i - \mathbf{x}^\star)\|_2^2.$$

This can be used to tighten the inequality as follows

$$\begin{aligned} & \left(\mathbb{E}_{r-1, k-1} \|\mathbf{y}_i^+ - \mathbf{x}^\star\|_A^2 + \delta(1 + \frac{1}{K})^{K-(k-1)} \mathbb{E}_{r-1, k-1} \|\mathbf{y}_i^+ - \mathbf{x}\|^2 \right) \leq \\ & \left((1 - \frac{\mu\eta}{6}) \|\mathbf{y}_i - \mathbf{x}^\star\|_A^2 + (1 - \frac{1}{5K}) \delta(1 + \frac{1}{K})^{K-k+1} \|\mathbf{y}_i - \mathbf{x}\|^2 \right) - \frac{\eta}{2} \|A(\mathbf{y}_i - \mathbf{x}^\star)\|_2^2 + 7\beta\eta^2\sigma^2 \end{aligned}$$

If $\eta \leq \frac{1}{\mu K}$, then $(1 - \frac{1}{5K}) \leq (1 - \frac{\mu\eta}{6})$ and we have the strongly-convex version of the first statement.

Now for the second statement, recall that $\tilde{\xi}_{i,k}^r \geq 0$ was defined to be

$$\tilde{\xi}_{i,k}^r := \left([f(\mathbb{E}_{r-1}[\mathbf{y}_{i,k}^r])] - f(\mathbf{x}^\star) + \delta(1 + \frac{1}{K})^{K-k} \mathbb{E}_{r-1} \|\mathbb{E}_{r-1}[\mathbf{y}_{i,k}^r] - \mathbf{x}^{r-1}\|^2 \right).$$

Observe that for quadratics, $\mathbb{E}_{r-1}[\nabla f(\mathbf{x})] = \nabla f(\mathbb{E}_{r-1}[\mathbf{x}])$. This implies that the analysis of $\tilde{\xi}_{i,k}^r$ is essentially of a deterministic process with $\sigma = 0$, proving the second statement. It is also straightforward to repeat exactly the above argument to formally verify the second statement. \square

SERVER PROGRESS IN ONE ROUND. NOW WE COMBINE THE PROGRESS MADE BY EACH CLIENT IN ONE STEP TO CALCULATE THE SERVER PROGRESS.

Lemma 79. Suppose (A2), (A5) and (A4) hold, and $\{f_i\}$ are quadratics. Then, the following holds for the update (12.22) with $\eta \leq \min(\frac{1}{10\beta}, \frac{1}{21\delta K}, \frac{1}{10\mu K})$ and weights $w_k := (1 - \frac{\mu\eta}{6})^{1-k}$:

$$\frac{\eta}{6} \sum_{k=1}^K w_k \mathbb{E}_{r-1} \|\nabla f(\mathbf{x}_k^r)\|^2 \leq (f(\mathbb{E}_{r-2}[\mathbf{x}^{r-1}]) - f^\star) - w_K (f(\mathbb{E}_{r-1}[\mathbf{x}^r]) - f^\star) + \sum_{k=1}^K w_k 8\eta \frac{\sigma^2}{N}.$$

Set $\mu = 0$ if $\{f_i\}$ s are not strongly-convex (is only general-convex).

Proof. Let us do the non-convex (and general convex) case first. By summing over Lemma 78 we have

$$\frac{\eta}{6} \sum_{k=1}^K \mathbb{E}_{r-1} \|\nabla f(\mathbf{y}_{i,k})\|^2 \leq \xi_{i,0}^r - \xi_{i,K}^r + 7K\beta\eta^2\sigma^2.$$

A similar result holds with $\sigma = 0$ for $\mathbb{E}_{r-1}[\mathbf{y}_{i,k}]$. Now, using Lemma 74 we have that

$$\frac{\eta}{6} \sum_{k=1}^K \mathbb{E}_{r-1} \|\nabla f(\mathbf{x}_k^r)\|^2 \leq \underbrace{\frac{1}{N} \sum_{i=1}^N (\tilde{\xi}_{i,0}^r + \frac{1}{N} \xi_{i,0})}_{=: \theta_+^r} - \underbrace{\frac{1}{N} \sum_{i=1}^N (\tilde{\xi}_{i,K}^r + \frac{1}{N} \xi_{i,K})}_{=: \theta_-^r} + 7K\beta\eta^2 \frac{\sigma^2}{N}.$$

Using Lemma 77, we have that

$$\theta_+^r = (1 + \frac{1}{N})(f(\mathbf{x}^{r-1}) - f(\mathbf{x}^\star)) \leq f(\mathbb{E}_{r-1}[\mathbf{x}^r]) + \frac{1}{N} \mathbb{E} f(\mathbf{x}^r) - (1 + \frac{1}{N})f(\mathbf{x}^\star) + 3K\beta \frac{\sigma^2}{N}.$$

Further, by Lemma 75, we have that

$$\theta_-^r \geq f(\mathbb{E}_{r-1}[\mathbf{x}^r]) + \frac{1}{N}f(\mathbf{x}^r) - (1 + \frac{1}{N})f(\mathbf{x}^\star).$$

Combining the above gives:

$$\frac{\eta}{6} \sum_{k=1}^K \mathbb{E}_{r-1} \|\nabla f(\mathbf{x}_k^r)\|^2 \leq f(\mathbb{E}_{r-2}[\mathbf{x}^{r-1}]) - f(\mathbb{E}_{r-1}[\mathbf{x}^r]) + 10\beta K \frac{\sigma^2}{N}.$$

proving the second part of the Lemma for weights $w_k = 1$. The proof of strongly convex follows a very similar argument. Unrolling Lemma 78 using weights $w_k := (1 - \frac{\mu\eta}{6})^{1-k}$ gives

$$\frac{\eta}{6} \sum_{k=1}^K w_k \mathbb{E}_{r-1} \|\nabla f(\mathbf{x}_k^r)\|^2 \leq \theta_+^r - w_K \theta_-^r + \sum_{k=1}^K w_k 7\eta \frac{\sigma^2}{N}.$$

As in the general-convex case, we can use Lemmas 75, 74 and 77 to prove that

$$\frac{\eta}{6} \sum_{k=1}^K w_k \mathbb{E}_{r-1} \|\nabla f(\mathbf{x}_k^r)\|^2 \leq (f(\mathbb{E}_{r-2}[\mathbf{x}^{r-1}]) - f^\star) - w_K(f(\mathbb{E}_{r-1}[\mathbf{x}^r]) - f^\star) + \sum_{k=1}^K w_k 8\eta \frac{\sigma^2}{N}.$$

□

DERIVING FINAL RATES. THE PROOF OF THEOREM XXXIII FOLLOWS BY APPROPRIATELY UNROLLING LEMMA 79. FOR GENERAL-CONVEX FUNCTIONS, WE CAN SIMPLY USE LEMMA 56 WITH THE PROBABILITIES SET AS $p_k^r = \frac{1}{KR}$. FOR STRONGLY-CONVEX FUNCTIONS, WE USE $p_k^r \propto (1 - \frac{\mu\eta}{6})^{1-rk}$ AND FOLLOW THE COMPUTATIONS IN LEMMA 55.

13 Appendix for Byzantine robust learning using history

13.1 Convergence of momentum SGD

HERE WE DESCRIBE THE CONVERGENCE PROOF OF THE NAIVE SGD WITH MOMENTUM ALGORITHM. STARTING FROM A GIVEN \mathbf{x}_0 AND WITH $\mathbf{m}_0 = \mathbf{0}$, WE RUN THE FOLLOWING UPDATES WITH A SEQUENCE OF MOMENTUM PARAMETERS $\alpha_t \in [0, 1]$ AND STEP-SIZES $\eta_t \geq 0$

$$\begin{aligned}\mathbf{m}_t &= \alpha_t \mathbf{g}(\mathbf{x}_{t-1}) + (1 - \alpha_t) \mathbf{m}_{t-1} \\ \mathbf{x}_t &= \mathbf{x}_{t-1} - \eta_t \mathbf{m}_t.\end{aligned}\tag{SGDM}$$

WHILE THERE EXIST NUMEROUS PREVIOUS ANALYSES OF SGD WITH MOMENTUM FOR SMOOTH NON-CONVEX OBJECTIVES, MOST OF THEM RELY ON VIEWING THE SGDM METHOD AS AN APPROXIMATION OF AN UNDERLYING SGD WITHOUT MOMENTUM ALGORITHM—SEE [YU ET AL. \(2019A\)](#); [LIU ET AL. \(2020\)](#) FOR RECENT EXAMPLES OF THIS VIEWPOINT. BECAUSE THEY VIEW MOMENTUM AS APPROXIMATING AN SGD PROCESS, THE RATES PROVED ARE NECESSARILY SLOWER FOR MOMENTUM AND FURTHER THEY CAN ONLY HANDLE CONSTANT VALUES OF α (I.E. THE MOMENTUM PARAMETER CANNOT DECREASE WITH T). IN THIS WORK, WE TAKE AN ALTERNATE VIEWPOINT TO MOMENTUM INSPIRED BY ([CUTKOSKY AND ORABONA, 2019](#); [KARIMIREDDY ET AL., 2020A](#)). WE VIEW THE MOMENTUM UPDATE AS A WAY TO REDUCE THE VARIANCE I.E. BY USING AN EXPONENTIAL AVERAGING OVER MANY INDEPENDENT STOCHASTIC GRADIENTS WE GET AN ESTIMATE OF THE TRUE FULL GRADIENT WHICH HAS MUCH LESSER VARIANCE (THOUGH HIGHER BIAS). THIS WAY, OUR METHOD CAN HANDLE MOMENTUM PARAMETER WHICH IS ALMOST 1 ($\alpha \approx \frac{1}{\sigma\sqrt{T}}$). THUS THE RESULTING UPDATE HAS VERY LOW VARIANCE WHICH WILL LATER BE CRUCIAL FOR DERIVING OPTIMAL ROBUST METHODS.

Theorem XXXIV (Convergence of SGDM). *The SGDM algorithm with step-size $\eta_t = \min\{\frac{1}{4\sigma} \sqrt{\frac{f(\mathbf{x}_0) - f^*}{LT}}, \frac{1}{4L}\}$ and momentum parameter $\alpha_1 = 1$ and $\alpha_t = 4L\eta_{t-1}$ for $t \geq 2$ satisfies*

$$\frac{1}{T} \sum_{t=1}^T \mathbb{E} \|\nabla f(\mathbf{x}_{t-1})\|^2 \leq 80 \cdot \sigma \sqrt{\frac{L(f(\mathbf{x}_0) - f^*)}{T}} + \frac{4L(f(\mathbf{x}_0) - f^*)}{T}$$

FIRST, NOTE THAT THE RATE FOR MOMENTUM ALGORITHM IS OF THE ORDER $\frac{\sigma}{\sqrt{T}}$ WHICH MATCHES THE OPTIMAL RATE OF SGD FOR SMOOTH NON-CONVEX FUNCTIONS (ARJEVANI ET AL., 2019). FURTHER, THIS RATE IS ACHIEVED USING VERY HIGH MOMENTUM WITH BOTH α (AND STEP-SIZES) OF THE ORDER $\frac{1}{\sigma\sqrt{T}}$. ALSO, WHEN $\sigma = 0$ I.E. IN THE DETERMINISTIC GRADIENT CASE, WE RECOVER THE OPTIMAL $\frac{1}{T}$ RATE (BUT WITH A CONSTANT STEP-SIZE AND MOMENTUM). THIS IS INTUITIVE SINCE WE DO NOT NEED TO REDUCE THE VARIANCE IN THE DETERMINISTIC CASE AND SO LARGE MOMENTUM IS UNNECESSARY.

Remark 80 (Large batch generalization). *There is some empirical evidence that momentum is also useful when using extremely large batch sizes (i.e. nearly deterministic gradient) and helps in closing the generalization gap (Shallue et al., 2018). In contrast, current theory claims that gradient descent (without momentum) is already optimal for non-convex optimization (Arjevani et al., 2019). We believe these differences occur because even if using large batches, there remains stochasticity in the gradient due to data-augmentation. Thus $\sigma > 0$ in practice even when using full batches.*

WE FIRST PROVE SOME SUPPORTING LEMMAS BEFORE PROVING THEOREM XXXIV.

Lemma 81. *For $\alpha_1 = 1$ and any $\alpha_t \in [0, 1]$ for $t \geq 2$, and an L -smooth function f we have that $\mathbb{E}_1[f(\mathbf{x}_1)] \leq f(\mathbf{x}_0) - \frac{\eta_1}{2} \|\nabla f(\mathbf{x}_0)\|^2 + \frac{\eta_1}{2} \sigma^2 - \frac{\eta_1}{2} (1 - L\eta_1) \|\mathbf{m}_1\|^2$ and for $t \geq 2$*

$$\mathbb{E}_t[f(\mathbf{x}_t)] \leq f(\mathbf{x}_{t-1}) + \frac{\eta_t}{2} \|\mathbf{m}_t - \nabla f(\mathbf{x}_{t-1})\|^2 - \frac{\eta_t}{2} \|\nabla f(\mathbf{x}_{t-1})\|^2 - \frac{\eta_t}{2} (1 - L\eta_t) \|\mathbf{m}_t\|^2.$$

Proof. By the smoothness of the function f and the SGDm update,

$$\begin{aligned} f(\mathbf{x}_t) &\leq f(\mathbf{x}_{t-1}) - \eta_t \langle \nabla f(\mathbf{x}_{t-1}), \mathbf{m}_t \rangle + \frac{L\eta_t^2}{2} \|\mathbf{m}_t\|^2 \\ &= f(\mathbf{x}_{t-1}) + \frac{\eta_t}{2} \|\mathbf{m}_t - \nabla f(\mathbf{x}_{t-1})\|^2 - \frac{\eta_t}{2} \|\nabla f(\mathbf{x}_{t-1})\|^2 - \frac{\eta_t}{2} (1 - L\eta_t) \|\mathbf{m}_t\|^2. \end{aligned}$$

Taking conditional expectation on both sides yields the second part of the lemma. The first part follows from standard descent analysis of SGD. \square

Lemma 82. *Define $\mathbf{e}_t := \mathbf{m}_t - \nabla f(\mathbf{x}_{t-1})$. Then, using any momentum and step-sizes such that $1 \geq \alpha_t \geq 4L\eta_{t-1}$ for $t \geq 2$, we have for an L -smooth function f that $\mathbb{E}\|\mathbf{e}_1\|^2 \leq \alpha_1 \sigma^2$ and for $t \geq 2$*

$$\mathbb{E}\|\mathbf{e}_t\|^2 \leq (1 - \frac{\alpha_t}{2}) \mathbb{E}\|\mathbf{e}_{t-1}\|^2 + L^2 \eta_{t-1}^2 (1 - \alpha_t) (1 + \frac{2}{\alpha_t}) \mathbb{E}\|\mathbf{m}_{t-1}\|^2 + \alpha_t^2 \sigma^2.$$

Proof. Starting from the definition of \mathbf{e}_t and \mathbf{m}_t ,

$$\begin{aligned}
 \mathbb{E}\|\mathbf{e}_t\|^2 &= \mathbb{E}\|\mathbf{m}_t - \nabla f(\mathbf{x}_{t-1})\|^2 \\
 &= \mathbb{E}\|\alpha_t \mathbf{g}(\mathbf{x}_{t-1}) + (1 - \alpha_t) \mathbf{m}_{t-1} - \nabla f(\mathbf{x}_{t-1})\|^2 \\
 &\leq (1 - \alpha_t)^2 \mathbb{E}\|\mathbf{m}_{t-1} - \nabla f(\mathbf{x}_{t-1})\|^2 + \alpha_t^2 \sigma^2 \\
 &= (1 - \alpha_t)^2 \mathbb{E}\|(\mathbf{m}_{t-1} - \nabla f(\mathbf{x}_{t-2})) + (\nabla f(\mathbf{x}_{t-2}) - \nabla f(\mathbf{x}_{t-1}))\|^2 + \alpha_t^2 \sigma^2 \\
 &\leq (1 - \alpha_t)(1 + \frac{\alpha_t}{2}) \mathbb{E}\|\mathbf{m}_{t-1} - \nabla f(\mathbf{x}_{t-2})\|^2 + (1 - \alpha_t)(1 + \frac{2}{\alpha_t}) \mathbb{E}\|\nabla f(\mathbf{x}_{t-2}) - \nabla f(\mathbf{x}_{t-1})\|^2 + \alpha_t^2 \sigma^2 \\
 &\leq (1 - \frac{\alpha_t}{2}) \mathbb{E}\|\mathbf{e}_{t-1}\|^2 + L^2(1 - \alpha_t)(1 + \frac{2}{\alpha_t}) \mathbb{E}\|\mathbf{x}_{t-2} - \mathbf{x}_{t-1}\|^2 + \alpha_t^2 \sigma^2 \\
 &\leq (1 - \frac{\alpha_t}{2}) \mathbb{E}\|\mathbf{e}_{t-1}\|^2 + L^2 \eta_{t-1}^2 (1 - \alpha_t)(1 + \frac{2}{\alpha_t}) \mathbb{E}\|\mathbf{m}_{t-1}\|^2 + \alpha_t^2 \sigma^2.
 \end{aligned}$$

Here the first inequality used the fact that $\mathbf{g}(\mathbf{x}_{t-1})$ is an unbiased and independent stochastic gradient with variance bounded by σ^2 . The second inequality follows from Fano's inequality i.e. $\|\mathbf{x} + \mathbf{y}\|^2 \leq (1 + a)\|\mathbf{x}\|^2 + (1 + \frac{1}{a})\|\mathbf{y}\|^2$ for any $a \geq 0$. \square

WE ARE NOW READY TO PROVE THE CONVERGENCE THEOREM.

PROOF OF THEOREM XXXIV. SCALING LEMMA 81 BY L AND ADDING IT TO LEMMA 82 WE HAVE FOR ANY $t \geq 2$

$$\begin{aligned}
 \mathbb{E} Lf(\mathbf{x}_t) + \mathbb{E}\|\mathbf{e}_t\|^2 &\leq \mathbb{E} Lf(\mathbf{x}_{t-1}) + \frac{L\eta_t}{2} \mathbb{E}\|\mathbf{e}_t\|^2 - \frac{L\eta_t}{2} \mathbb{E}\|\nabla f(\mathbf{x}_{t-1})\|^2 - \frac{L\eta_t}{2} (1 - L\eta_t) \|\mathbf{m}_t\|^2 \\
 &\quad + (1 - \frac{\alpha_t}{2}) \mathbb{E}\|\mathbf{e}_{t-1}\|^2 + L^2 \eta_{t-1}^2 (1 - \alpha_t)(1 + \frac{2}{\alpha_t}) \mathbb{E}\|\mathbf{m}_{t-1}\|^2 + \alpha_t^2 \sigma^2.
 \end{aligned}$$

BY TAKING $\eta_t = \eta_{t-1} = \eta$ AND $1 \geq \alpha_t \geq 4L\eta$

$$\begin{aligned}
 &\underbrace{\mathbb{E} L(f(\mathbf{x}_t) - f^*) + \left(1 - \frac{L\eta_t}{2}\right) \mathbb{E}\|\mathbf{e}_t\|^2 + \frac{L\eta_t}{2} (1 - L\eta_t) \|\mathbf{m}_t\|^2 + \frac{L\eta_t}{2} \mathbb{E}\|\nabla f(\mathbf{x}_{t-1})\|^2}_{=: \xi_t} \\
 &\leq \mathbb{E} L(f(\mathbf{x}_{t-1}) - f^*) + \left(1 - \frac{\alpha_t}{2}\right) \mathbb{E}\|\mathbf{e}_{t-1}\|^2 + L^2 \eta_{t-1}^2 (1 - \alpha_t)(1 + \frac{2}{\alpha_t}) \mathbb{E}\|\mathbf{m}_{t-1}\|^2 + \alpha_t^2 \sigma^2. \\
 &\leq \underbrace{\mathbb{E} L(f(\mathbf{x}_{t-1}) - f^*) + \left(1 - \frac{L\eta_{t-1}}{2}\right) \mathbb{E}\|\mathbf{e}_{t-1}\|^2 + \frac{L\eta_{t-1}}{2} (1 - L\eta_{t-1}) \mathbb{E}\|\mathbf{m}_{t-1}\|^2}_{=: \xi_{t-1}} + \alpha_t^2 \sigma^2.
 \end{aligned}$$

NOTE THAT FROM THE FIRST PARTS OF LEMMA 81 AND LEMMA 82, WE HAVE

$$\begin{aligned}
 \xi_1 &\leq \mathbb{E} L(f(\mathbf{x}_1) - f^*) + \left(1 - \frac{L\eta_1}{2}\right) \mathbb{E}\|\mathbf{e}_1\|^2 + \frac{L\eta_1}{2} (1 - L\eta_1) \|\mathbf{m}_1\|^2 \\
 &\leq L(f(\mathbf{x}_0) - f^*) + \sigma^2 - \frac{L\eta_1}{2} \mathbb{E}\|\nabla f(\mathbf{x}_0)\|^2.
 \end{aligned}$$

SUMMING OVER t AND AGAIN REARRANGING GIVES

$$\sum_{t=1}^{\ell} L\eta_t \mathbb{E}\|\nabla f(\mathbf{x}_{t-1})\|^2 \leq L(f(\mathbf{x}_0) - f^*) + \sum_{t=1}^{\ell} \alpha_t^2 \sigma^2.$$

BY TAKING $\eta_t = \eta_{t-1} = \eta$ AND $\alpha_t = 4L\eta$, THIS SIMPLIFIES THE ABOVE INEQUALITY TO

$$\frac{1}{T} \sum_{t=1}^T \mathbb{E} \|\nabla f(\mathbf{x}_{t-1})\|^2 \leq \frac{f(\mathbf{x}_0) - f^*}{\eta T} + 16L\eta\sigma^2.$$

BY TAKING $\eta = \min\{\frac{1}{4\sigma} \sqrt{\frac{f(\mathbf{x}_0) - f^*}{LT}}, \frac{1}{4L}\}$ WE PROVE THE THEOREM. \square

13.2 Proof of Theorem XV - Failure of permutation-invariant methods

OUR PROOF BUILDS TWO INSTANCES OF A δ -ROBUST OPTIMIZATION PROBLEM SATISFYING DEFINITION D AND SHOWS THAT THEY ARE INDISTINGUISHABLE, MEANING THAT WE MAKE A MISTAKE ON AT LEAST ONE OF THEM.

FOR THE FIRST PROBLEM, SET $f^{(1)}(x) = \frac{\mu}{2}x^2 - Gx$ WITH OPTIMUM AT $x^* = \frac{G}{\mu}$ FOR SOME G TO BE DEFINED LATER. IT HAS A GRADIENT $\nabla f^{(1)}(x) = \mu x - G$ AND WE SET THE STOCHASTIC GRADIENT FOR SOME $\tilde{\delta} \in [0, 1]$ TO BE DEFINED LATER AS

$$g^{(1)}(x) = \begin{cases} \mu x - \sigma \tilde{\delta}^{-1/2} & \text{WITH PROB. } \tilde{\delta} \\ \mu x & \text{O.W.} \end{cases}$$

DEFINING $G := \sigma \tilde{\delta}^{1/2}$, WE HAVE THAT $g^{(1)}(x)$ IS AN UNBIASED STOCHASTIC GRADIENT. FURTHER, ITS VARIANCE IS BOUNDED BY σ^2 SINCE $\mathbb{E}[(g^{(1)}(x) - \nabla f^{(1)}(x))^2] \leq \sigma^2$. IN EACH ROUND t , LET EACH WORKER $i \in [n]$ DRAW AN I.I.D. SAMPLE FROM THE DISTRIBUTION $g^{(1)}(x)$ AS THEIR STOCHASTIC GRADIENT. DEFINE $C_t \in [n]$ TO BE THE NUMBER OF WORKERS WHOSE STOCHASTIC GRADIENTS IS THE FIRST SETTING I.E.

$$C_t = \#\{i \in [n] \text{ s.t. } g_i^{(1)}(x_t) = \mu x_t - \sigma \tilde{\delta}^{-1/2}\}.$$

NOW WE DEFINE THE SECOND PROBLEM. LET $f_2(x) = \frac{\mu}{2}x^2$ WITH OPTIMUM AT $x^* = 0$. DEFINE ITS STOCHASTIC GRADIENT TO ALWAYS BE $g^{(2)}(x) = \mu x$. NOW, IN ROUND t EACH WORKER $i \in [n]$ COMPUTES $g_i^{(2)}(x_t) = x_t$. THEN, $\min(n\tilde{\delta}, C_t)$ BYZANTINE WORKERS CORRUPT THEIR GRADIENTS TO INSTEAD BE $g_j^{(2)}(x_t) = \mu x_t - \sigma \tilde{\delta}^{-1/2}$.

NOTE THAT C_t IS THE SUM n INDEPENDENT BERNOULLI TRIALS WITH PARAMETER $\tilde{\delta}$. THUS, WE HAVE VIA CHERNOFF'S BOUND THAT FOR ANY $\gamma \geq 2$,

$$\Pr[C_t > (1 + \gamma)n\tilde{\delta}] \leq \exp\left(-\frac{\gamma n\tilde{\delta}}{2}\right).$$

BY PICKING $\gamma = \max(2, 2(1 + \log(T))/(n\tilde{\delta}))$, WE HAVE THAT $\Pr[C_t > (1 + \gamma)n\tilde{\delta}] \leq \frac{1}{2T}$. BY SETTING $\tilde{\delta} = \delta/6$ AND ASSUMING THAT n IS LARGE ENOUGH SUCH THAT $4(1 + \log T) \leq \delta n$, WE CAN

13.3. Proof of Theorem XVI (Limits of robust aggregation)

SIMPLIFY $(1 + \gamma)n\tilde{\delta} \geq \delta n$. TAKING AN UNION BOUND OVER ALL VALUES OF t , WE HAVE THAT

$$\Pr[C_t \leq n\delta \text{ FOR ALL } t \in [T]] \geq \frac{1}{2}.$$

THUS, WITH PROBABILITY AT LEAST 0.5, WE HAVE THAT THE STOCHASTIC GRADIENTS IN PROBLEM 1 ARE EXACTLY THE SAME (UP TO PERMUTATION) TO PROBLEM 2. THIS IMPLIES THAT WITH PROBABILITY 0.5, NO PERMUTATION-INVARIANT ALGORITHM CAN DISTINGUISH BETWEEN THE TWO SETTINGS, IMPLYING THAT WE NECESSARILY INCUR AN ERROR OF THE ORDER OF THE DIFFERENCE BETWEEN THEIR MINIMA

$$\mu\left(\frac{G}{\mu}\right)^2 = \frac{\sigma^2 \tilde{\delta}}{\mu} = \frac{\sigma^2 \delta}{6\mu}.$$

□

13.3 Proof of Theorem XVI (Limits of robust aggregation)

IT IS EASY TO ESTABLISH THE SECOND RESULT SINCE IF $\delta \geq \frac{1}{2}$, IT IS IMPOSSIBLE TO DECIDE WHICH OF THE SUBSETS IS GOOD. E.G. IF HALF OF THE INPUTS ARE a AND THE OTHER ARE b , EVEN IF WE KNOW THAT $\rho = 0$, THE GOOD WORKERS MIGHT CORRESPOND TO EITHER THE a HALF OR THE b HALF EQUALLY LIKELY. ASSUMING $\delta \leq \frac{1}{2}$, DEFINE THE FOLLOWING BINOMIAL DISTRIBUTION:

$$\mathcal{P} := \begin{cases} \rho\delta^{-1/2} & \text{WITH PROB. } \delta/2 \\ 0 & \text{O.W.} \end{cases}$$

SUPPOSE THAT EACH x_i FOR ALL $i \in [n]$ IS AN IID SAMPLE DRAWN FROM \mathcal{P} . CLEARLY WE HAVE THAT $\mathbb{E}(x_i - x_j)^2 \leq \rho^2$. DEFINE $B_n \in [n]$ TO BE THE NUMBER OF SAMPLES WHICH ARE EQUAL TO $\rho\delta^{-1/2}$ (WITH THE REST BEING 0). NOW CONSIDER A SECOND SCENARIO FOR $\{x_i\}$: THE ADVERSARY SETS $\min(\delta n, B_n)$ OF THE VARIABLES TO $\rho\delta^{-1/2}$ AND THE REST OF THE GOOD VARIABLES ARE 0.

NOTE THAT $\mathbb{E}[B_n] = n\delta/2$ AND SO BY MARKOV'S INEQUALITY WE HAVE THAT $\Pr[B_n \leq n\delta] \geq \frac{1}{2}$. SO WITH AT LEAST PROBABILITY 1/2, THE TWO CASES ARE IMPOSSIBLE TO DISTINGUISH. HOWEVER IN THE FIRST CASE, ALL SAMPLES ARE GOOD WHEREAS IN THE SECOND CASE ONLY THE 0 SAMPLES ARE GOOD. HENCE, ANY OUTPUT WILL NECESSARILY HAVE AN ERROR OF THE ORDER OF THE DIFFERENCE BETWEEN THEIR RESPECTIVE $\tilde{\mathbf{x}}$ S:

$$(\mathbb{E}_{x \sim \mathcal{P}}[x] - 0)^2 = (\rho\delta^{1/2}/2)^2 = \frac{\delta\rho^2}{4}.$$

13.4 Proof of Theorem XVII- Robustness of iterative clipping

FIRST, SUPPOSE THAT $\delta = 0$. IN THIS CASE, OUR CHOICE OF CLIPPING RADIUS $\tau_l = \tilde{\mathcal{O}}(\rho/\sqrt{\delta}) = \infty$ MEANS THAT WE WILL SIMPLY AVERAGES ALL POINTS. HENCE, WE RECOVER $\tilde{\mathbf{x}}$ EXACTLY WITH

NO ERROR AS REQUIRED. NOW IF $\delta > 0$, THIS MEANS THAT AT LEAST ONE OF THE n WORKERS IS BYZANTINE AND HENCE $\delta \geq 1/n$. WE CONSIDER THIS CASE IN THE REST OF THE PROOF.

RECALL THAT $\bar{\mathbf{x}} = \frac{1}{|\mathcal{G}|} \sum_{i \in \mathcal{G}} \mathbf{x}_i$ AND LET US DEFINE $\boldsymbol{\mu} = \mathbb{E}[\mathbf{x}_j]$ FOR ANY FIXED $j \in \mathcal{G}$. NOW SINCE THE GOOD RANDOM VECTORS ARE IID, WE HAVE

$$\mathbb{E} \|\bar{\mathbf{x}} - \boldsymbol{\mu}\|^2 \leq \frac{\rho^2}{|\mathcal{G}|} \leq \frac{2\rho^2}{n} \leq 2\delta\rho^2.$$

WE WILL FIRST ANALYZE A SINGLE STEP OF CENTERED CLIPPING ASSUMING WE HAVE ACCESS TO \mathbf{v} SUCH THAT I) \mathbf{v} IS INDEPENDENT OF THE SAMPLES $\{\mathbf{x}_i | i \in \mathcal{G}\}$, AND II) $\mathbb{E} \|\mathbf{v} - \boldsymbol{\mu}\|^2 \leq \mathcal{O}(\rho^2)$. THEN, WE WILL NEXT SEE HOW TO CONSTRUCT SUCH A \mathbf{v} . OUR PROOF IS INSPIRED BY (ZHANG ET AL., 2019B; GORBUNOV ET AL., 2020) WHO ANALYZE THE BIAS OF CLIPPING UNDER HEAVY-TAILED NOISE.

13.4.1 Single iteration with good starting point

LET US SUPPOSE THAT AT SOME ROUND l , WE HAVE THE FOLLOWING PROPERTIES:

- WE HAVE A GOOD ESTIMATE OF THE MEAN SATISFYING $\mathbb{E} \|\mathbf{v}_l - \boldsymbol{\mu}\|^2 \leq B_l^2$ WHERE B_l IS A KNOWN DETERMINISTIC CONSTANT.
- THE STARTING POINT \mathbf{v}_l IS STATISTICALLY INDEPENDENT OF $\{\mathbf{x}_i | i \in \mathcal{G}\}$.

DEFINE INDICATOR VARIABLES $\mathbb{1}_{i,l} := \mathbb{1}\{\|\mathbf{v}_l - \mathbf{x}_i\| \geq \tau_l\}$ WHICH DEFINE THE EVENT THAT THE VECTOR \mathbf{x}_i IS CLIPPED, AS WELL THE RESULTING CLIPPED VECTOR

$$\mathbf{y}_{i,l} := \mathbf{v}_l + (\mathbf{x}_i - \mathbf{v}_l) \min\left(1, \frac{\tau_l}{\|\mathbf{x}_i - \mathbf{v}_l\|}\right).$$

THE OUTPUT CAN ALSO BE WRITTEN IN THIS NEW NOTATION AS

$$\mathbf{v}_{l+1} = \frac{1}{n} \sum_{i \in [n]} \mathbf{y}_{i,l} = (1 - \delta) \frac{1}{|\mathcal{G}|} \sum_{i \in \mathcal{G}} \mathbf{y}_{i,l} + \delta \frac{1}{|\mathcal{B}|} \sum_{j \in \mathcal{B}} \mathbf{y}_{j,l}.$$

THEN THE ERROR CAN BE DECOMPOSED AS FOLLOWS

$$\begin{aligned}
 \mathbb{E}\|\mathbf{v}_{l+1} - \boldsymbol{\mu}\|^2 &= \mathbb{E}\left\|\left(1 - \delta\right) \frac{1}{|\mathcal{G}|} \sum_{i \in \mathcal{G}} \mathbf{y}_{i,l} + \delta \frac{1}{|\mathcal{B}|} \sum_{j \in \mathcal{B}} \mathbf{y}_{j,l} - \frac{1}{|\mathcal{G}|} \sum_{i \in \mathcal{G}} \mathbb{E}[\mathbf{x}_i]\right\|^2 \\
 &= \mathbb{E}\left\|\left(1 - \delta\right) \frac{1}{|\mathcal{G}|} \sum_{i \in \mathcal{G}} (\mathbf{y}_{i,l} - \mathbb{E}[\mathbf{x}_i]) + \delta \frac{1}{|\mathcal{B}|} \sum_{j \in \mathcal{B}} (\mathbf{y}_{j,l} - \boldsymbol{\mu})\right\|^2 \\
 &\leq 2(1 - \delta)^2 \mathbb{E}\left\|\frac{1}{|\mathcal{G}|} \sum_{i \in \mathcal{G}} \mathbf{y}_{i,l} - \mathbb{E}[\mathbf{x}_i]\right\|^2 + 2\delta^2 \frac{1}{|\mathcal{B}|} \sum_{j \in \mathcal{B}} \mathbb{E}\|\mathbf{y}_{j,l} - \boldsymbol{\mu}\|^2 \\
 &= 2(1 - \delta)^2 \underbrace{\mathbb{E}\left\|\frac{1}{|\mathcal{G}|} \sum_{i \in \mathcal{G}} \mathbb{E}[\mathbf{y}_{i,l}] - \mathbb{E}[\mathbf{x}_i]\right\|^2}_{\mathcal{T}_1} + 2(1 - \delta)^2 \underbrace{\mathbb{E}\left\|\frac{1}{|\mathcal{G}|} \sum_{i \in \mathcal{G}} \mathbf{y}_{i,l} - \mathbb{E}[\mathbf{y}_{i,l}]\right\|^2}_{\mathcal{T}_2} + 2\delta^2 \underbrace{\frac{1}{|\mathcal{B}|} \sum_{j \in \mathcal{B}} \mathbb{E}\|\mathbf{y}_{j,l} - \boldsymbol{\mu}\|^2}_{\mathcal{T}_3}.
 \end{aligned}$$

THUS, THE ERROR CAN BE DECOMPOSED INTO 3 TERMS: \mathcal{T}_1 CORRESPONDS TO THE BIAS INTRODUCED BY OUR CLIPPING OPERATION IN THE GOOD WORKERS, \mathcal{T}_2 IS THE VARIANCE OF THE CLIPPED GOOD WORKERS, AND FINALLY \mathcal{T}_3 IS THE ERROR DUE TO THE BAD WORKERS. WE WILL ANALYZE EACH OF THE THREE ERRORS IN TURN,

\mathcal{T}_3 . FOR ANY BAD INDEX $j \in \mathcal{B}$, WE CAN BOUND THE ERROR USING OUR CLIPPING RADIUS AS FOR ANY PARAMETER $\gamma > 0$ AS

$$\mathbb{E}\|\mathbf{y}_{j,l} - \boldsymbol{\mu}\|^2 \leq (1 + \frac{1}{\gamma}) \mathbb{E}\|\mathbf{y}_{j,l} - \mathbf{v}_l\|^2 + (1 + \gamma) \mathbb{E}\|\mathbf{v}_l - \boldsymbol{\mu}\|^2 \leq (1 + \gamma) \tau_l^2 + (1 + \frac{1}{\gamma}) B_l^2.$$

THE FIRST STEP USED YOUNG'S INEQUALITY. FURTHER, THE ERROR DUE TO THE BAD BUYS IS ALSO SMALLER IF OUR INITIAL ESTIMATION ERROR B_l^2 IS SMALL.

\mathcal{T}_1 . WE THEN COMPUTE THE BIAS IN THE UPDATE OF A GOOD WORKER $i \in \mathcal{G}$ DUE TO THE CLIPPING OPERATION. LET $\mathbb{1}_{i,l}$ BE AN INDICATOR VARIABLE DENOTING IF THE i TH WORKER WAS CLIPPED (I.E. ITS DISTANCE FROM \mathbf{v}_l EXCEEDING τ_l). NOTE THAT IF $\mathbb{1}_{i,l} = 0$, WE HAVE THAT $\mathbf{y}_{i,l} = \mathbf{x}_i$. THEN,

$$\begin{aligned}
 \mathbb{E}\|\mathbf{y}_{i,l} - \mathbf{x}_i\| &= \mathbb{E}\mathbb{1}_{i,l} \|\mathbf{y}_{i,l} - \mathbf{x}_i\| \leq \mathbb{E}\mathbb{1}_{i,l} \|\mathbf{v}_l - \mathbf{x}_i\| \leq \frac{\mathbb{E}\mathbb{1}_{i,l} \|\mathbf{v}_l - \mathbf{x}_i\|^2}{\tau} \\
 &\leq \frac{\mathbb{E}\|\mathbf{v}_l - \mathbf{x}_i\|^2}{\tau} \leq \frac{(1 + \frac{1}{\gamma}) \mathbb{E}\|\mathbf{v}_l - \boldsymbol{\mu}\|^2 + (1 + \gamma) \mathbb{E}\|\mathbf{x}_i - \boldsymbol{\mu}\|^2}{\tau} \\
 &\leq \frac{(1 + \frac{1}{\gamma}) \rho^2 + (1 + \gamma) B_l^2}{\tau}.
 \end{aligned}$$

USING THIS, WE CAN COMPUTE THE ERROR AS

$$\mathcal{T}_1 \leq \frac{1}{|\mathcal{G}|} \sum_{i \in \mathcal{G}} \|\mathbb{E}[\mathbf{y}_{i,l}] - \mathbb{E}[\mathbf{x}_i]\|^2 \leq \frac{1}{|\mathcal{G}|} \sum_{i \in \mathcal{G}} (\mathbb{E}\|\mathbf{y}_{i,l} - \mathbf{x}_i\|)^2 \leq \frac{((1 + \frac{1}{\gamma}) \rho^2 + (1 + \gamma) B_l^2)^2}{\tau^2}. \quad (13.1)$$

* \mathcal{T}_2 . SINCE \mathbf{v}_l IS INDEPENDENT OF $\{\mathbf{x}_i | i \in \mathcal{G}\}$, THE RANDOM VECTORS $\{\mathbf{y}_i | i \in \mathcal{G}\}$ ARE ALSO INDEPENDENT OF EACH OTHER. WE THEN HAVE,

$$\begin{aligned}\mathcal{T}_2 &= \mathbb{E} \frac{1}{(|\mathcal{G}|)^2} \sum_{i \in \mathcal{G}} \|\mathbf{y}_{i,l} - \mathbb{E}[\mathbf{y}_{i,l}]\|^2 \\ &\leq \mathbb{E} \frac{1}{(|\mathcal{G}|)^2} \sum_{i \in \mathcal{G}} \|\mathbf{x}_i - \mathbb{E}[\mathbf{x}_i]\|^2 \\ &\leq \frac{\rho^2}{|\mathcal{G}|} \leq \frac{2\rho^2}{n} \leq 2\delta\rho^2.\end{aligned}$$

THE EQUALITY IN THE FIRST STEP USED THE FACT THAT THE QUANTITIES WERE INDEPENDENT, AND THE NEXT INEQUALITY FOLLOWS BECAUSE OF THE CONTRACTIVITY OF A CLIPPING (PROJECTION) STEP. THE LAST USED THE FACT THAT $|\mathcal{G}| \geq n/2$.

COMBINING THE THREE ERROR TERMS, WE HAVE

$$\begin{aligned}\mathbb{E}\|\mathbf{v}_{l+1} - \boldsymbol{\mu}\|^2 &\leq 2(1-\delta)^2 \frac{((1+\frac{1}{\gamma})\rho^2 + (1+\gamma)B_l^2)^2}{\tau_l^2} + 2(1-\delta)^2 2\delta\rho^2 + 2\delta^2 \left((1+\gamma)\tau_l^2 + (1+\frac{1}{\gamma})B_l^2 \right) \\ &= (4(1-\delta)\delta(1+\gamma)^{3/2} + 2(1+\frac{1}{\gamma})\delta^2)B_l^2 + 4(1-\delta)^2\delta\rho^2 + (4(1-\delta)(1+\frac{1}{\gamma})\sqrt{1+\gamma})\delta\rho^2 \\ &\leq (4(1-\delta)\delta(1+\frac{1}{3})^{3/2} + 8\delta^2)B_l^2 + 4\delta\rho^2 + (16\sqrt{1+\frac{1}{3}})\delta\rho^2 \\ &\leq (6.158\delta(1-\delta) + 8\delta^2)B_l^2 + 22\delta\rho^2.\end{aligned}$$

THE LAST STEP USED $\gamma = \frac{1}{3}$. THE EQUALITY IN THE SECOND STEP USED A CLIPPING RADIUS OF

$$\tau_l^2 = 4(1-\delta) \frac{(4\rho^2 + \frac{4}{3}B_l^2)}{\sqrt{3}\delta}.$$

THUS, WE HAVE

$$\|\mathbf{v}_{l+1} - \boldsymbol{\mu}\|^2 \leq (6.158\delta(1-\delta) + 8\delta^2)B_l^2 + 22\delta\rho^2 \leq 8\delta B_l^2 + 22\delta\rho^2. \quad (13.2)$$

IN MANY CASES, WE WILL HAVE ACCESS TO A GOOD STARTING POINT SATISFYING $B_l^2 = \mathcal{O}(\rho^2)$. FOR EXAMPLE, SUPPOSE WE KNEW THAT $\mathbb{E}\|\mathbf{x}_i\|^2 \leq b\rho^2$ FOR ANY FIXED $i \in \mathcal{G}$. THEN, THEN $B_l^2 = b\rho^2$ WITH $\mathbf{v}_l = \mathbf{0}$. IN SUCH CASES, THE ABOVE PROOF SHOWS THAT A SINGLE ITERATION OF CENTERED CLIPPING IS SUFFICIENT TO GIVE A ROBUST AGGREGATOR.

13.4.2 Robustness starting from arbitrary point

IN THIS SECTION, WE WILL GIVE AN ALGORITHM FOR THOSE CASES WHERE WE DO NOT HAVE ACCESS TO ANY GOOD STARTING POINT. THEN, WE PROCEED AS FOLLOWS: FIRST, WE PARTITION THE GIVEN DATASET $\mathcal{X} = \{\mathbf{x}_1, \dots, \mathbf{x}_n\}$ RANDOMLY INTO \mathcal{X}_1 AND \mathcal{X}_2 OF SIZES $|\mathcal{X}_1| = 2n/3$ AND $|\mathcal{X}_2| = n/3$. NOTE THAT THE FRACTION OF BYZANTINE WORKERS IN EACH OF THESE IS AT MOST $|\mathcal{B}| < \delta n = \underbrace{1.5\delta}_{=: \delta_1} |\mathcal{X}_1| = \underbrace{3\delta}_{=: \delta_2} |\mathcal{X}_2|$. OUR STRATEGY THEN IS TO COMPUTE \mathbf{v}_l WITH $\mathcal{O}(\rho^2)$ ERROR

13.4. Proof of Theorem XVII- Robustness of iterative clipping

USING SET \mathcal{X}_1 , AND THEN RUN A SINGLE STEP OF CENTERED CLIPPING USING DATA \mathcal{X}_2 . BY (13.2), WE CAN GUARANTEE THAT THE OUTPUT WILL HAVE ERROR $\mathcal{O}(\delta\rho^2)$.

COMPUTING A GOOD STARTING POINT. STARTING FROM AN ARBITRARY \mathbf{v}_0 WITH ERROR $\mathbb{E}\|\mathbf{v}_0 - \boldsymbol{\mu}\|^2 \leq B_0^2$, WE WILL REPEATEDLY APPLY CENTERED CLIPPING (CC). CONSIDER ITERATION $l \geq 0$ WITH ERROR $\mathbb{E}\|\mathbf{v}_l - \boldsymbol{\mu}\|^2 \leq B_l^2$. THEN, TO ANALYZE THE ERROR OF $\mathbb{E}\|\mathbf{v}_{l+1} - \boldsymbol{\mu}\|^2$, WE WILL PROCEED EXACTLY AS IN THE SINGLE ITERATION CASE UP TO EQUATION (13.1). HOWEVER, WHILE ANALYZING THE ERROR OF \mathcal{T}_2 , WE CAN NO LONGER RELY ON $\{\mathbf{y}_1, \dots, \mathbf{y}_n\}$ BEING INDEPENDENT. HENCE, THIS STEP INSTEAD BECOMES

$$\begin{aligned} \mathcal{T}_2 &\leq \mathbb{E} \frac{1}{|\mathcal{G}|} \sum_{i \in \mathcal{G}} \|\mathbf{y}_{i,l} - \mathbb{E}[\mathbf{y}_{i,l}]\|^2 \\ &\leq \mathbb{E} \frac{1}{|\mathcal{G}|^2} \sum_{i \in \mathcal{G}} \|\mathbf{x}_i - \mathbb{E}[\mathbf{x}_i]\|^2 \\ &\leq \rho^2. \end{aligned}$$

COMBINING THE PREVIOUS BOUNDS FOR THE ERRORS OF \mathcal{T}_1 AND \mathcal{T}_3 WITH THE ABOVE YIELDS

$$\begin{aligned} \mathbb{E}\|\mathbf{v}_{l+1} - \boldsymbol{\mu}\|^2 &\leq 2(1-\delta)^2 \frac{((1+\frac{1}{\gamma})\rho^2 + (1+\gamma)B_l^2)^2}{\tau_l^2} + 2(1-\delta)^2\rho^2 + 2\delta^2\left((1+\gamma)\tau_l^2 + (1+\frac{1}{\gamma})B_l^2\right) \\ &= (4(1-\delta)\delta(1+\gamma)^{3/2} + 2(1+\frac{1}{\gamma})\delta^2)B_l^2 + 2(1-\delta)^2\rho^2 + (4(1-\delta)(1+\frac{1}{\gamma})\sqrt{1+\gamma})\delta\rho^2 \\ &\leq (4(1-\delta)\delta(1+\frac{1}{3})^{3/2} + 8\delta^2)B_l^2 + 2\rho^2 + (16\sqrt{1+\frac{1}{3}})\delta\rho^2 \\ &\leq (6.158\delta(1-\delta) + 8\delta^2)B_l^2 + (20\delta + 2)\rho^2 \\ &\leq 6.45\delta B_l^2 + 5\rho^2. \end{aligned}$$

THE LAST STEP ASSUMED $\delta \leq 0.15$, AND THE STEP BEFORE THAT USED $\gamma = \frac{1}{3}$. THE EQUALITY IN THE SECOND STEP USED A CLIPPING RADIUS OF

$$\tau_l^2 = 4(1-\delta) \frac{(4\rho^2 + \frac{4}{3}B_l^2)}{\sqrt{3}\delta}.$$

NOTE THAT THIS HOLDS FOR *any iteration* l AND WE DID NOT MAKE ANY ASSUMPTIONS ON \mathbf{v}_l . HENCE, WE CAN DEFINE $B_{l+1}^2 = 6.45\delta B_l^2 + 5\rho^2$. WITH THIS, WE CAN GUARANTEE THAT FOR ANY $l \geq 0$, WE HAVE $\mathbb{E}\|\mathbf{v}_l - \boldsymbol{\mu}\|^2 \leq B_l^2$ WHERE

$$B_l^2 \leq (6.45\delta)^l B_0^2 + 154\rho^2 \text{ FOR } \delta \leq 0.15. \quad (13.3)$$

* PUTTING IT TOGETHER. LET US RUN THE ABOVE PROCEDURE FOR l STEPS ON \mathcal{X}_1 WITH $\delta_1 = 1.5\delta$. THEN, BY (13.3) WE CAN GUARANTEE THAT \mathbf{v}_l SATISFIES

$$B_l^2 \leq (9.7\delta)^l B_0^2 + 154\rho^2 \text{ FOR } \delta \leq 0.1.$$

SINCE \mathbf{v}_l WAS COMPUTED ONLY USING \mathcal{X}_1 , IT IS INDEPENDENT OF \mathcal{X}_2 AND HENCE BY (13.2) HAS AN ERROR WITH $\delta_2 = 3\delta$ FOR $\delta \leq 0.1$:

$$\begin{aligned}\mathbb{E}\|\mathbf{v}_{l+1} - \boldsymbol{\mu}\|^2 &\leq 24\delta B_l^2 + 66\delta\rho^2 \\ &\leq 24\delta\left((9.7\delta)^l B_0^2 + 154\rho^2\right) + 66\delta\rho^2 \\ &\leq (9.7\delta)^{l+1} 2.5B_0^2 + 3762\delta\rho^2.\end{aligned}$$

NOW, WE CAN FINISH THE PROOF OF THE THEOREM AS

$$\begin{aligned}\mathbb{E}\|\mathbf{v}_{l+1} - \bar{\mathbf{x}}\|^2 &\leq \left(1 + \frac{1}{99}\right)\mathbb{E}\|\mathbf{v}_{l+1} - \boldsymbol{\mu}\|^2 + 100\mathbb{E}\|\boldsymbol{\mu} - \bar{\mathbf{x}}\|^2 \\ &\leq \left(1 + \frac{1}{99}\right)\mathbb{E}\|\mathbf{v}_{l+1} - \boldsymbol{\mu}\|^2 + 100\delta\rho^2 \\ &\leq (9.7\delta)^{l+1} 3B_0^2 + 4000\delta\rho^2.\end{aligned}$$

OUR THEORY CONSIDERS THIS TWO STAGE PROCEDURE ONLY DUE TO A TECHNICALITY. WE BELIEVE THAT THE SINGLE STAGE METHOD ALSO YIELDS SIMILAR GUARANTEES, AND LEAVE ITS ANALYSIS (ALONG WITH OBTAINING A BETTER δ_{\max}) FOR FUTURE WORK.

□

13.5 Proof of Theorem XIX - Byzantine-Robust Convergence

WE STATE SEVERAL SUPPORTING LEMMAS BEFORE PROVING OUR MAIN THEOREM XIX.

Lemma 83 (Aggregation error). *Given that Definition F holds, and that we use momentum constant parameter with $\alpha_1 = 1$ and $\alpha_t = \alpha$ for $t \geq 2$, the error between the ideal average momentum $\bar{\mathbf{m}}_t$ and the output of the robust aggregation rule \mathbf{m}_t for any $t \geq 2$ can be bounded as*

$$\mathbb{E}\|\mathbf{m}_t - \bar{\mathbf{m}}_t\|^2 \leq 2c\delta\sigma^2(\alpha + (1 - \alpha)^{t-1}).$$

For $t = 1$ we can simplify the bound as $\mathbb{E}\|\mathbf{m}_1 - \bar{\mathbf{m}}_1\|^2 \leq 2c\delta\sigma^2$.

Proof. Expanding the definition of the worker momentum for any two good workers $i, j \in \mathcal{G}$,

$$\begin{aligned}\mathbb{E}\|\mathbf{m}_{i,t} - \mathbf{m}_{j,t}\|^2 &= \mathbb{E}\|\alpha_t(\mathbf{g}_i(\mathbf{x}_{t-1}) - \mathbf{g}_j(\mathbf{x}_{t-1})) + (1 - \alpha_t)(\mathbf{m}_{i,t-1} - \mathbf{m}_{j,t-1})\|^2 \\ &\leq \mathbb{E}\|(1 - \alpha_t)(\mathbf{m}_{i,t-1} - \mathbf{m}_{j,t-1})\|^2 + 2\alpha_t^2\sigma^2 \\ &\leq (1 - \alpha_t)\mathbb{E}\|\mathbf{m}_{i,t-1} - \mathbf{m}_{j,t-1}\|^2 + 2\alpha_t^2\sigma^2.\end{aligned}$$

Recall that we use $\alpha_1 = 1$ and a fixed momentum $\alpha_t = \alpha$ the rest of the steps. Unrolling the recursion above yields

$$\mathbb{E}\|\mathbf{m}_{i,t} - \mathbf{m}_{j,t}\|^2 \leq \left(\sum_{\ell=2}^t (1 - \alpha)^{t-\ell}\right) 2\alpha^2\sigma^2 + (1 - \alpha)^{t-1} 2\sigma^2 \leq 2\sigma^2(\alpha + (1 - \alpha)^{t-1}).$$

The previous computation shows that all the good vectors given to the server are close to each other with $\rho^2 = 2\sigma^2(\alpha + (1-\alpha)^{t-1})$. Hence, by Definition F the output of the aggregation rule $\text{AGG}(\mathbf{m}_{t,1}, \dots, \mathbf{m}_{t,n})$ satisfies the lemma statement. \square

Lemma 84 (Descent bound). *For $\alpha_1 = 1$ and any $\alpha_t \in [0, 1]$ for $t \geq 2$, $\eta_t \leq \frac{1}{L}$, and an L -smooth function f we have for any $t \geq 1$*

$$\mathbb{E}_t[f(\mathbf{x}_t)] \leq f(\mathbf{x}_{t-1}) - \frac{\eta_t}{2} \|\nabla f(\mathbf{x}_{t-1})\|^2 + \eta_t \mathbb{E}_t \|\bar{\mathbf{e}}_t\|^2 + \eta_t \mathbb{E}_t \|\mathbf{m}_t - \bar{\mathbf{m}}_t\|^2.$$

where $\bar{\mathbf{e}}_t := \bar{\mathbf{m}}_t - \nabla f(\mathbf{x}_{t-1})$.

Proof. By the smoothness of the function f and the server update,

$$\begin{aligned} f(\mathbf{x}_t) &\leq f(\mathbf{x}_{t-1}) - \eta_t \langle \nabla f(\mathbf{x}_{t-1}), \mathbf{m}_t \rangle + \frac{L\eta_t^2}{2} \|\mathbf{m}_t\|^2 \\ &\leq f(\mathbf{x}_{t-1}) - \eta_t \langle \nabla f(\mathbf{x}_{t-1}), \mathbf{m}_t \rangle + \frac{\eta_t}{2} \|\mathbf{m}_t\|^2 \\ &= f(\mathbf{x}_{t-1}) + \frac{\eta_t}{2} \|\mathbf{m}_t - \nabla f(\mathbf{x}_{t-1})\|^2 - \frac{\eta_t}{2} \|\nabla f(\mathbf{x}_{t-1})\|^2 \\ &= f(\mathbf{x}_{t-1}) + \frac{\eta_t}{2} \|\mathbf{m}_t \pm \bar{\mathbf{m}}_t - \nabla f(\mathbf{x}_{t-1})\|^2 - \frac{\eta_t}{2} \|\nabla f(\mathbf{x}_{t-1})\|^2 \\ &\leq f(\mathbf{x}_{t-1}) + \eta_t \|\bar{\mathbf{e}}_t\|^2 + \eta_t \|\mathbf{m}_t - \bar{\mathbf{m}}_t\|^2 - \frac{\eta_t}{2} \|\nabla f(\mathbf{x}_{t-1})\|^2. \end{aligned}$$

Taking conditional expectation on both sides yields the second part of the lemma. \square

Lemma 85 (Error bound). *Using any constant momentum and step-sizes such that $1 \geq \alpha \geq 8L\eta$ for $t \geq 2$, we have for an L -smooth function f that $\mathbb{E} \|\bar{\mathbf{e}}_1\|^2 \leq \frac{2\sigma^2}{n}$ and for $t \geq 2$*

$$\mathbb{E} \|\bar{\mathbf{e}}_t\|^2 \leq (1 - \frac{2\alpha}{5}) \mathbb{E} \|\bar{\mathbf{e}}_{t-1}\|^2 + \frac{\alpha}{10} \mathbb{E} \|\nabla f(\mathbf{x}_{t-2})\|^2 + \frac{\alpha}{10} \mathbb{E} \|\mathbf{m}_{t-1} - \bar{\mathbf{m}}_{t-1}\|^2 + \alpha^2 \frac{2\sigma^2}{n}.$$

Proof. Using the definitions (13.4) and proceeding as in Lemma 82, we have

$$\begin{aligned} \mathbb{E} \|\bar{\mathbf{e}}_t\|^2 &= \mathbb{E} \|\bar{\mathbf{m}}_t - \nabla f(\mathbf{x}_{t-1})\|^2 \\ &= \mathbb{E} \|\alpha_t \bar{\mathbf{g}}(\mathbf{x}_{t-1}) + (1 - \alpha_t) \bar{\mathbf{m}}_{t-1} - \nabla f(\mathbf{x}_{t-1})\|^2 \\ &\leq \mathbb{E} \|\alpha_t \nabla f(\mathbf{x}_{t-1}) + (1 - \alpha_t) \bar{\mathbf{m}}_{t-1} - \nabla f(\mathbf{x}_{t-1})\|^2 + \alpha_t^2 \frac{2\sigma^2}{n} \\ &= (1 - \alpha_t)^2 \mathbb{E} \|(\bar{\mathbf{m}}_{t-1} - \nabla f(\mathbf{x}_{t-2})) + (\nabla f(\mathbf{x}_{t-2}) - \nabla f(\mathbf{x}_{t-1}))\|^2 + \alpha_t^2 \frac{2\sigma^2}{n} \\ &\leq (1 - \alpha_t)(1 + \frac{\alpha_t}{2}) \mathbb{E} \|(\bar{\mathbf{m}}_{t-1} - \nabla f(\mathbf{x}_{t-2}))\|^2 + (1 - \alpha_t)(1 + \frac{2}{\alpha_t}) \mathbb{E} \|\nabla f(\mathbf{x}_{t-2}) - \nabla f(\mathbf{x}_{t-1})\|^2 + \alpha_t^2 \frac{2\sigma^2}{n} \\ &\leq (1 - \frac{\alpha_t}{2}) \mathbb{E} \|\bar{\mathbf{e}}_{t-1}\|^2 + \frac{2L^2}{\alpha_t} \mathbb{E} \|\mathbf{x}_{t-2} - \mathbf{x}_{t-1}\|^2 + \alpha_t^2 \frac{2\sigma^2}{n} \\ &= (1 - \frac{\alpha_t}{2}) \mathbb{E} \|\bar{\mathbf{e}}_{t-1}\|^2 + \frac{2L^2\eta_{t-1}^2}{\alpha_t} \mathbb{E} \|\mathbf{m}_{t-1}\|^2 + \alpha_t^2 \frac{2\sigma^2}{n}. \end{aligned}$$

Note that we have $\frac{2\sigma^2}{n}$ instead of simply σ^2 since we average the momentums (and hence also the stochastic gradients) over all the good workers (who number at least $n/2$). Another

difference is that in the last equality we have the robust aggregate \mathbf{m}_{t-1} instead of the average momentum $\bar{\mathbf{m}}_{t-1}$. We can proceed as

$$\begin{aligned}\mathbb{E}\|\bar{\mathbf{e}}_t\|^2 &\leq (1 - \frac{\alpha}{2})\mathbb{E}\|\bar{\mathbf{e}}_{t-1}\|^2 + \frac{2L^2\eta^2}{\alpha}\mathbb{E}\|\mathbf{m}_{t-1}\|^2 + \alpha^2\frac{2\sigma^2}{n} \\ &= (1 - \frac{\alpha}{2})\mathbb{E}\|\bar{\mathbf{e}}_{t-1}\|^2 + \frac{2L^2\eta^2}{\alpha}\mathbb{E}\|\mathbf{m}_{t-1} \pm \bar{\mathbf{m}}_{t-1} \pm \nabla f(\mathbf{x}_{t-2})\|^2 + \alpha^2\frac{2\sigma^2}{n} \\ &\leq (1 - \frac{\alpha}{2})\mathbb{E}\|\bar{\mathbf{e}}_{t-1}\|^2 + \frac{6L^2\eta^2}{\alpha}\mathbb{E}\|\bar{\mathbf{e}}_{t-1}\|^2 + \frac{6L^2\eta^2}{\alpha}\mathbb{E}\|\mathbf{m}_{t-1} - \bar{\mathbf{m}}_{t-1}\|^2 + \frac{6L^2\eta^2}{\alpha}\mathbb{E}\|\nabla f(\mathbf{x}_{t-2})\|^2 + \alpha^2\frac{2\sigma^2}{n}.\end{aligned}$$

Our choice of the momentum parameter α implies $64L^2\eta^2 \leq \alpha^2$ and yields the lemma statement. \square

PROOF OF THEOREM XIX. WE WILL LOOSELY FOLLOW THE PROOF OF VANILLA SGDM PROOF IN THEOREM XXXIV. RECALL THAT \mathcal{G} DENOTES THE GOOD SET AND \mathcal{B} DENOTES THE BAD BYZANTINE WORKERS WITH $|\mathcal{G}| \leq (1 - \delta)n$ AND $|\mathcal{B}| = n - |\mathcal{G}| \leq \delta n$. DEFINE THE IDEAL MOMENTUM AND ERROR AS

$$\bar{\mathbf{m}}_t := \frac{1}{|\mathcal{G}|} \sum_{j \in \mathcal{G}} \mathbf{m}_{t,j}, \quad \bar{\mathbf{e}}_t := \bar{\mathbf{m}}_t - \nabla f(\mathbf{x}_{t-1}), \quad \text{AND} \quad \bar{\mathbf{g}}(\mathbf{x}_{t-1}) = \frac{1}{|\mathcal{G}|} \sum_{j \in \mathcal{G}} \mathbf{g}_j(\mathbf{x}_{t-1}). \quad (13.4)$$

NOW SCALE THE MODIFIED ERROR BOUND LEMMA 85 BY $\frac{5\eta}{2\alpha}$ AND ADD IT TO THE MODIFIED DESCENT BOUND LEMMA 84 TAKING EXPECTATIONS ON BOTH SIDES TO GET FOR $t \geq 2$

$$\begin{aligned}\mathbb{E}[f(\mathbf{x}_t)] + \frac{5\eta}{2\alpha}\mathbb{E}\|\bar{\mathbf{e}}_t\|^2 &\leq \mathbb{E}[f(\mathbf{x}_{t-1})] - \frac{\eta}{2}\mathbb{E}\|\nabla f(\mathbf{x}_{t-1})\|^2 + \eta\mathbb{E}\|\bar{\mathbf{e}}_t\|^2 + \eta\mathbb{E}\|\mathbf{m}_t - \bar{\mathbf{m}}_t\|^2 + \\ &\quad \frac{5\eta}{2\alpha}\mathbb{E}\|\bar{\mathbf{e}}_{t-1}\|^2 - \eta\mathbb{E}\|\bar{\mathbf{e}}_{t-1}\|^2 + \frac{\eta}{4}\mathbb{E}\|\nabla f(\mathbf{x}_{t-2})\|^2 + \frac{\eta}{4}\mathbb{E}\|\mathbf{m}_{t-1} - \bar{\mathbf{m}}_{t-1}\|^2 + 5\eta\alpha\frac{\sigma^2}{n}\end{aligned}$$

REARRANGING THE ABOVE TERMS AND USING THE BOUND IN THE AGGREGATION ERROR LEMMA 83 YIELDS THE RECURSION

$$\begin{aligned}\underbrace{\mathbb{E}[f(\mathbf{x}_t) - f^* + (\frac{5\eta}{2\alpha} - \eta)\mathbb{E}\|\bar{\mathbf{e}}_t\|^2 + \frac{\eta}{4}\mathbb{E}\|\nabla f(\mathbf{x}_{t-1})\|^2]}_{=:\xi_t} &\leq \underbrace{\mathbb{E}[f(\mathbf{x}_{t-1}) - f^* + (\frac{5\eta}{2\alpha} - \eta)\mathbb{E}\|\bar{\mathbf{e}}_{t-1}\|^2 + \frac{\eta}{4}\mathbb{E}\|\nabla f(\mathbf{x}_{t-2})\|^2]}_{=:\xi_{t-1}} \\ &\quad - \frac{\eta}{4}\mathbb{E}\|\nabla f(\mathbf{x}_{t-1})\|^2 + \frac{5\eta\alpha}{n}\sigma^2 + \frac{5\eta}{4}\mathbb{E}\|\mathbf{m}_{t-1} - \bar{\mathbf{m}}_{t-1}\|^2 \\ &\leq \xi_{t-1} - \frac{\eta}{4}\mathbb{E}\|\nabla f(\mathbf{x}_{t-1})\|^2 \\ &\quad + \frac{5\eta\alpha\sigma^2}{2}\left(\frac{2}{n} + \delta(c + \frac{c}{\alpha}(1 - \alpha)^{t-2})\right).\end{aligned}$$

FURTHER, SPECIALIZING THE DESCENT BOUND LEMMA 84 AND ERROR BOUND LEMMA 85 FOR

13.6. Proof of Theorem XX (Momentum based variance reduction)

$t = 1$ WE HAVE

$$\begin{aligned}\xi_1 &\leq \mathbb{E} f(\mathbf{x}_1) - f^\star + \frac{3\eta}{2} \mathbb{E} \|\bar{\mathbf{e}}_1\|^2 + \frac{\eta}{4} \mathbb{E} \|\nabla f(\mathbf{x}_0)\|^2 \\ &\leq f(\mathbf{x}_0) - f^\star + \frac{5\eta}{2} \mathbb{E} \|\bar{\mathbf{e}}_1\|^2 - \frac{\eta}{4} \mathbb{E} \|\nabla f(\mathbf{x}_0)\|^2 + \eta \mathbb{E} \|\mathbf{m}_1 - \bar{\mathbf{m}}_1\|^2 \\ &\leq f(\mathbf{x}_0) - f^\star - \frac{\eta}{4} \mathbb{E} \|\nabla f(\mathbf{x}_0)\|^2 + \frac{5\eta\sigma^2}{n} + 2c\eta\delta\sigma^2.\end{aligned}$$

SUMMING OVER t AND AGAIN REARRANGING OUR RECURSION FOR ξ_t GIVES

$$\begin{aligned}\frac{1}{T} \sum_{t=1}^T \mathbb{E} \|\nabla f(\mathbf{x}_{t-1})\|^2 &\leq \frac{4(f(\mathbf{x}_0) - f^\star)}{\eta T} + \frac{20\sigma^2}{nT} + \frac{8c\delta\sigma^2}{T} \\ &\quad + \frac{10\alpha\sigma^2}{T} \sum_{t=1}^T \left(\frac{2}{n} + \delta(c + \frac{c}{\alpha}(1-\alpha)^{t-2}) \right) \\ &\leq \frac{4(f(\mathbf{x}_0) - f^\star)}{\eta T} + \frac{20\sigma^2}{nT} + \frac{8c\delta\sigma^2}{T} \\ &\quad + \frac{20\alpha\sigma^2}{n} + 10c\delta\alpha\sigma^2 + \frac{10\delta c\alpha\sigma^2}{\alpha^2 T} \\ &= \frac{4(f(\mathbf{x}_0) - f^\star)}{\eta T} + \frac{20\sigma^2}{nT} + \frac{8c\delta\sigma^2}{T} \\ &\quad + \frac{160L\eta\sigma^2}{n} + 80L\delta c\eta\sigma^2 + \frac{5c\delta\sigma^2}{4L\eta T} \\ &\leq 16\sqrt{\frac{5\sigma^2(2+c\delta n)}{nT}} \left(L(f(\mathbf{x}_0) - f^\star) + \frac{5c\delta}{16}\sigma^2 \right) \\ &\quad + \frac{32L(f(\mathbf{x}_0) - f^\star)}{T} + \frac{10c\delta\sigma^2}{T} + \frac{20\sigma^2}{nT} + \frac{8c\delta\sigma^2}{T}.\end{aligned}$$

SUBSTITUTING THE APPROPRIATE STEP-SIZE $\eta = \min\left(\sqrt{\frac{f(\mathbf{x}_0) - f^\star + \frac{5c\delta}{16L}\sigma^2}{20LT\sigma^2\left(\frac{2}{n} + c\delta\right)}}, \frac{1}{8L}\right)$ FINISHES THE PROOF OF THE THEOREM. \square

13.6 Proof of Theorem XX (Momentum based variance reduction)

WE NOW DESCRIBE HOW TO MODIFY THE MOMENTUM METHOD WITH A SMALL CORRECTION TERM TO IMPROVE ITS CONVERGENCE RATE (CUTKOSKY AND ORABONA, 2019). STARTING FROM A GIVEN \mathbf{x}_0 AND WITH $\mathbf{d}_0 = \mathbf{0}$, $\alpha_1 = 1$, WE RUN THE FOLLOWING UPDATES WITH A SEQUENCE OF MOMENTUM PARAMETERS $\alpha_t \in [0, 1]$ AND STEP-SIZES $\eta_t \geq 0$ FOR $t \geq 2$

$$\mathbf{d}_{t,i} = \alpha_t \mathbf{g}_{t,i}(\mathbf{x}_{t-1}) + (1 - \alpha_t) \mathbf{d}_{t-1,i} + (1 - \alpha_t)(\mathbf{g}_{t,i}(\mathbf{x}_{t-1}) - \mathbf{g}_{t,i}(\mathbf{x}_{t-2})) \quad (\text{MVR-WORKER})$$

NOTE THAT BOTH $\mathbf{g}_{t,i}(\mathbf{x}_{t-1})$ AND $\mathbf{g}_{t,i}(\mathbf{x}_{t-2})$ HERE ARE COMPUTED USING THE SAME STOCHASTIC FUNCTION (SAME BATCH) AS INDICATED BY THE SUBSCRIPT. THE GOOD WORKERS COMMU-

NICATE $\mathbf{d}_{t,i}$ WHEREAS THE BAD ONES SEND ARBITRARY VECTORS. THEN, THE SERVER PERFORMS

$$\begin{aligned}\mathbf{d}_t &= \text{AGG}(\mathbf{d}_{t,1}, \dots, \mathbf{d}_{t,n}) \\ \mathbf{x}_t &= \mathbf{x}_{t-1} - \eta_t \mathbf{d}_t.\end{aligned}\tag{MVR-SERVER}$$

DEFINE $\bar{\mathbf{d}}_t := \frac{1}{|\mathcal{G}|} \sum_{j \in \mathcal{G}} \mathbf{d}_{t,j}$ AND $\bar{\mathbf{e}}_t := \bar{\mathbf{d}}_t - \nabla f(\mathbf{x}_{t-1})$. NOTE THAT SINCE $\alpha_1 = 1$, THE FIRST STEP CAN BE SIMPLIFIED AS $\mathbf{d}_{1,i} = \mathbf{g}_{1,i}(\mathbf{x}_0)$. HERE WE ASSUME THAT THE STOCHASTIC GRADIENT CONDITIONED ON ALL PAST HISTORY IS UNBIASED $\mathbb{E}_t[\mathbf{g}_{t,i}(\mathbf{x}_{t-1})] = \nabla f(\mathbf{x}_{t-1})$ AND HAS BOUNDED VARIANCE σ^2 . FURTHER, WE ASSUME THAT THE STOCHASTIC GRADIENTS SATISFY $\mathbb{E}\|\mathbf{g}_{t,i}(\mathbf{x}_{t-1}) - \mathbf{g}_{t,i}(\mathbf{x}_{t-2})\|^2 \leq L^2 \|\mathbf{x}_{t-1} - \mathbf{x}_{t-2}\|^2$. THIS IS STRONGER THAN ASSUMING ONLY THAT THE FULL GRADIENT ∇f IS LIPSCHITZ.

Lemma 86. *For $\alpha_1 = 1$ and any $\alpha_t \in [0, 1]$ for $t \geq 2$, $\eta_t \leq \frac{1}{L}$, and an L -smooth function f we have that $E_1[f(\mathbf{x}_1)] \leq f(\mathbf{x}_0) - \frac{\eta_1}{2} \|\nabla f(\mathbf{x}_0)\|^2 + \frac{\eta_1^2 L}{2} \sigma^2$ and for $t \geq 2$ with $\bar{\mathbf{e}}_t := \bar{\mathbf{d}}_t - \nabla f(\mathbf{x}_{t-1})$ we have*

$$\mathbb{E}_t[f(\mathbf{x}_t)] \leq f(\mathbf{x}_{t-1}) - \frac{\eta_t}{2} \|\nabla f(\mathbf{x}_{t-1})\|^2 + \eta_t (\mathbb{E}_t \|\bar{\mathbf{e}}_t\|^2 + \mathbb{E}_t \|\mathbf{d}_t - \bar{\mathbf{d}}_t\|^2).$$

THE PROOF IS IDENTICAL TO THAT OF LEMMA 81.

Lemma 87. *Using any momentum and step-sizes such that $1 \geq \alpha \geq 16L^2\eta^2$ for $t \geq 2$, we have i) $\mathbb{E}[\mathbf{e}_t] = 0$, and ii) for an L -smooth function f that $\mathbb{E}\|\bar{\mathbf{e}}_1\|^2 \leq 2\sigma^2/n$ and for $t \geq 2$*

$$\mathbb{E}\|\bar{\mathbf{e}}_t\|^2 \leq (1 - \frac{\alpha}{2}) \mathbb{E}\|\bar{\mathbf{e}}_{t-1}\|^2 + 8L^2\eta^2 \|\nabla f(\mathbf{x}_{t-2})\|^2 + 2\alpha^2\sigma^2/n + 4L^2\eta^2 \mathbb{E}\|\mathbf{d}_{t-1} - \bar{\mathbf{d}}_{t-1}\|^2.$$

Proof. Starting from the definition of $\bar{\mathbf{e}}_{t+1}$ and $\bar{\mathbf{d}}_t$,

$$\begin{aligned}\bar{\mathbf{e}}_t &= \bar{\mathbf{d}}_t - \nabla f(\mathbf{x}_{t-1}) \\ &= \alpha \bar{\mathbf{g}}_t(\mathbf{x}_{t-1}) + (1 - \alpha) \bar{\mathbf{d}}_{t-1} + (1 - \alpha) (\bar{\mathbf{g}}_t(\mathbf{x}_{t-1}) - \bar{\mathbf{g}}_t(\mathbf{x}_{t-2})) - \nabla f(\mathbf{x}_{t-1}) \\ &= \underbrace{(1 - \alpha) (\bar{\mathbf{d}}_{t-1} - \nabla f(\mathbf{x}_{t-2}))}_{\mathcal{T}_1} + \\ &\quad \underbrace{\alpha (\bar{\mathbf{g}}_t(\mathbf{x}_{t-1}) - \nabla f(\mathbf{x}_{t-1}))}_{\mathcal{T}_2} + \underbrace{(1 - \alpha) (\bar{\mathbf{g}}_t(\mathbf{x}_{t-1}) - \bar{\mathbf{g}}_t(\mathbf{x}_{t-2}) - \nabla f(\mathbf{x}_{t-1}) + \nabla f(\mathbf{x}_{t-2}))}_{\mathcal{T}_3}.\end{aligned}$$

Note that $\mathcal{T}_1 = (1 - \alpha) \bar{\mathbf{e}}_{t-1}$ and that $\mathbb{E}[\mathcal{T}_2] = 0, \mathbb{E}[\mathcal{T}_3] = 0$. This proves that $\mathbb{E}[\bar{\mathbf{e}}_t] = 0$. Further, conditioned on all history \mathcal{F}_t (i.e. everything before step t), we have $\mathbb{E}_t[\mathcal{T}_2] = 0$ and $\mathbb{E}_t[\mathcal{T}_3] = 0$

and \mathcal{T}_1 is deterministic. Hence, we can take squared norms on both sides as expand as

$$\begin{aligned}
 \mathbb{E}\|\bar{\mathbf{e}}_t\|^2 &= (1-\alpha)^2 \mathbb{E}\|\bar{\mathbf{e}}_{t-1}\|^2 + \\
 &\quad \mathbb{E}\|\alpha(\bar{\mathbf{g}}_t(\mathbf{x}_{t-1}) - \nabla f(\mathbf{x}_{t-1})) + (1-\alpha)(\bar{\mathbf{g}}_t(\mathbf{x}_{t-1}) - \bar{\mathbf{g}}_t(\mathbf{x}_{t-2}) - \nabla f(\mathbf{x}_{t-1}) + \nabla f(\mathbf{x}_{t-2}))\|^2 \\
 &\leq (1-\alpha)^2 \mathbb{E}\|\bar{\mathbf{e}}_{t-1}\|^2 + \\
 &\quad 2\mathbb{E}\|\alpha(\bar{\mathbf{g}}_t(\mathbf{x}_{t-1}) - \nabla f(\mathbf{x}_{t-1}))\|^2 + 2\mathbb{E}\|(1-\alpha)(\bar{\mathbf{g}}_t(\mathbf{x}_{t-1}) - \bar{\mathbf{g}}_t(\mathbf{x}_{t-2}) - \nabla f(\mathbf{x}_{t-1}) + \nabla f(\mathbf{x}_{t-2}))\|^2 \\
 &\leq (1-\alpha) \mathbb{E}\|\bar{\mathbf{e}}_{t-1}\|^2 + 2\alpha^2\sigma^2/n + 2(1-\alpha)^2 \mathbb{E}\|\bar{\mathbf{g}}_t(\mathbf{x}_{t-1}) - \bar{\mathbf{g}}_t(\mathbf{x}_{t-2})\|^2 \\
 &\leq (1-\alpha) \mathbb{E}\|\bar{\mathbf{e}}_{t-1}\|^2 + 2\alpha^2\sigma^2/n + 2(1-\alpha)^2 L^2 \mathbb{E}\|\mathbf{x}_{t-1} - \mathbf{x}_{t-2}\|^2 \\
 &= (1-\alpha) \mathbb{E}\|\bar{\mathbf{e}}_{t-1}\|^2 + 2\alpha^2\sigma^2/n + 4(1-\alpha)^2 L^2 \eta^2 \mathbb{E}\|\bar{\mathbf{d}}_{t-1}\|^2 + 4(1-\alpha)^2 L^2 \eta^2 \mathbb{E}\|\mathbf{d}_{t-1} - \bar{\mathbf{d}}_{t-1}\|^2 \\
 &\leq (1-\alpha) \mathbb{E}\|\bar{\mathbf{e}}_{t-1}\|^2 + 2\alpha^2\sigma^2/n + 8L^2\eta^2 \mathbb{E}\|\bar{\mathbf{e}}_{t-1}\|^2 + 8L^2\eta^2 \mathbb{E}\|\nabla f(\mathbf{x}_{t-2})\|^2 + 4L^2\eta^2 \mathbb{E}\|\mathbf{d}_{t-1} - \bar{\mathbf{d}}_{t-1}\|^2.
 \end{aligned}$$

Here the the third inequality used the expected squared Lipschitzness of $g_t(\cdot)$, whereas the rest relied on Young's inequality and that $\alpha \in [0, 1]$. Now the condition on the momentum implies that $8L^2\eta^2 \leq \frac{\alpha}{2}$, yielding the second statement of the lemma for $t \geq 2$. The statement for \mathbf{e}_1 follows since $\bar{\mathbf{d}}_0 = 0$. \square

Lemma 88 (Aggregation error). *Given Definition F holds and we use a momentum constant parameter $\alpha_1 = 1$ and $\alpha_t = \alpha \geq 192L^2\eta^2(c\delta + 1)$ for $t \geq 2$, the error between the ideal average momentum $\bar{\mathbf{d}}_t$ and the robust aggregate \mathbf{d}_t for any $t \geq 2$ can be bounded as*

$$\begin{aligned}
 \mathbb{E}\|\bar{\mathbf{e}}_t\|^2 + c\delta \mathbb{E}\|\mathbf{d}_{i,t} - \mathbf{d}_{j,t}\|^2 &\leq (1 - \frac{\alpha}{4})(\mathbb{E}\|\bar{\mathbf{e}}_{t-1}\|^2 + c\delta \mathbb{E}\|\mathbf{d}_{i,t-1} - \mathbf{d}_{j,t-1}\|^2) \\
 &\quad + \frac{\alpha}{16} \|\nabla f(\mathbf{x}_{t-2})\|^2 + (c\delta + 1/n)4\alpha^2\sigma^2
 \end{aligned}$$

For $t = 1$, we can simplify the bound to $\mathbb{E}\|\bar{\mathbf{e}}_1\|^2 + c\delta \mathbb{E}\|\mathbf{d}_{i,2} - \mathbf{d}_{j,2}\|^2 \leq 2\sigma^2(c\delta + 1/n)$.

Proof. Expanding the definition of the worker momentum for any two good workers $i, j \in \mathcal{G}$ for $t \geq 2$,

$$\begin{aligned}
 \mathbb{E}\|\mathbf{d}_{i,t} - \mathbf{d}_{j,t}\|^2 &= \mathbb{E}\|\alpha(\mathbf{g}_i(\mathbf{x}_{t-1}) - \mathbf{g}_j(\mathbf{x}_{t-1})) + \\
 &\quad (1-\alpha)(\mathbf{d}_{i,t-1} - \mathbf{d}_{j,t-1}) + \\
 &\quad (1-\alpha)(\mathbf{g}_{t,i}(\mathbf{x}_{t-1}) - \mathbf{g}_{t,j}(\mathbf{x}_{t-1}) - \mathbf{g}_{t,i}(\mathbf{x}_{t-2}) + \mathbf{g}_{t,j}(\mathbf{x}_{t-2}))\|^2 \\
 &\leq \mathbb{E}\|(1-\alpha)(\mathbf{d}_{i,t-1} - \mathbf{d}_{j,t-1})\|^2 + 4\alpha^2\sigma^2 + 4L^2(1-\alpha)^2 \mathbb{E}\|\mathbf{x}_{t-1} - \mathbf{x}_{t-2}\|^2 \\
 &\leq (1-\alpha) \mathbb{E}\|\mathbf{d}_{i,t-1} - \mathbf{d}_{j,t-1}\|^2 + 4\alpha^2\sigma^2 + 4L^2\eta^2 \mathbb{E}\|\mathbf{d}_{t-1}\|^2 \\
 &\leq (1-\alpha) \mathbb{E}\|\mathbf{d}_{i,t-1} - \mathbf{d}_{j,t-1}\|^2 + 4\alpha^2\sigma^2 + 12L^2\eta^2 \mathbb{E}\|\mathbf{d}_{t-1} - \bar{\mathbf{d}}_{t-1}\|^2 \\
 &\quad + 12L^2\eta^2 \mathbb{E}\|\bar{\mathbf{e}}_{t-1}\|^2 + 12L^2\eta^2 \mathbb{E}\|\nabla f(\mathbf{x}_{t-2})\|^2 \\
 &\leq (1-\alpha + 12c\delta L^2\eta^2) \mathbb{E}\|\mathbf{d}_{i,t-1} - \mathbf{d}_{j,t-1}\|^2 + 4\alpha^2\sigma^2 \\
 &\quad + 12L^2\eta^2 \mathbb{E}\|\bar{\mathbf{e}}_{t-1}\|^2 + 12L^2\eta^2 \mathbb{E}\|\nabla f(\mathbf{x}_{t-2})\|^2.
 \end{aligned}$$

Scale this by $c\delta$ and then add the inequality from Lemma 87 to get

$$\begin{aligned}
 \mathbb{E}\|\bar{\mathbf{e}}_t\|^2 + c\delta\mathbb{E}\|\mathbf{d}_{i,t} - \mathbf{d}_{j,t}\|^2 &\leq (1 - \frac{\alpha}{2})\mathbb{E}\|\bar{\mathbf{e}}_{t-1}\|^2 + 8L^2\eta^2\|\nabla f(\mathbf{x}_{t-2})\|^2 + 2\alpha^2\sigma^2/n + 4L^2\eta^2\mathbb{E}\|\mathbf{d}_{t-1} - \bar{\mathbf{d}}_{t-1}\|^2 \\
 &\quad + (1 - \alpha + 12c\delta L^2\eta^2)c\delta\mathbb{E}\|\mathbf{d}_{i,t-1} - \mathbf{d}_{j,t-1}\|^2 + 4c\delta\alpha^2\sigma^2 \\
 &\quad + 12c\delta L^2\eta^2\mathbb{E}\|\bar{\mathbf{e}}_{t-1}\|^2 + 12c\delta L^2\eta^2\mathbb{E}\|\nabla f(\mathbf{x}_{t-2})\|^2 \\
 &\leq (1 - \frac{\alpha}{2})\mathbb{E}\|\bar{\mathbf{e}}_{t-1}\|^2 + 8L^2\eta^2\|\nabla f(\mathbf{x}_{t-2})\|^2 + 2\alpha^2\sigma^2/n \\
 &\quad + (1 - \alpha + 12c\delta L^2\eta^2 + 4L^2\eta^2)c\delta\mathbb{E}\|\mathbf{d}_{i,t-1} - \mathbf{d}_{j,t-1}\|^2 + 4c\delta\alpha^2\sigma^2 \\
 &\quad + 12c\delta L^2\eta^2\mathbb{E}\|\bar{\mathbf{e}}_{t-1}\|^2 + 12c\delta L^2\eta^2\mathbb{E}\|\nabla f(\mathbf{x}_{t-2})\|^2 \\
 &= (1 - \frac{\alpha}{2} + 12c\delta L^2\eta^2)\mathbb{E}\|\bar{\mathbf{e}}_{t-1}\|^2 + (8L^2\eta^2 + 12c\delta L^2\eta^2)\|\nabla f(\mathbf{x}_{t-2})\|^2 + (4c\delta + 2/n)\alpha^2\sigma^2 \\
 &\quad + (1 - \alpha + 12c\delta L^2\eta^2 + 4L^2\eta^2)c\delta\mathbb{E}\|\mathbf{d}_{i,t-1} - \mathbf{d}_{j,t-1}\|^2 \\
 &\leq (1 - \frac{\alpha}{4})(\mathbb{E}\|\bar{\mathbf{e}}_{t-1}\|^2 + c\delta\mathbb{E}\|\mathbf{d}_{i,t-1} - \mathbf{d}_{j,t-1}\|^2) + \frac{\alpha}{16}\|\nabla f(\mathbf{x}_{t-2})\|^2 + (4c\delta + 2/n)\alpha^2\sigma^2.
 \end{aligned}$$

Here we used $\alpha \geq 192L^2\eta^2(c\delta + 1)$, and Definition F that $\mathbb{E}\|\mathbf{d}_{t-1} - \bar{\mathbf{d}}_{t-1}\|^2 \leq c\delta\mathbb{E}\|\mathbf{d}_{i,t-1} - \mathbf{d}_{j,t-1}\|^2$.

□

WE ARE NOW READY TO PROVE THE CONVERGENCE THEOREM.

Theorem XXXV (Byzantine robust MVR). *Let us run the MVR algorithm combined with a robust aggregation rule AGG with step-size $\eta = \min\left(\sqrt[3]{\frac{f(\mathbf{x}_0) - f^*}{T(1536L^2\sigma^2(c\delta + 1)(c\delta + 1/n))}}, \frac{1}{4L}\right)$ and momentum parameter $\alpha = 192L^2\eta^2(1 + c\delta)$. Then,*

$$\frac{1}{T} \sum_{t=1}^T \mathbb{E}\|\nabla f(\mathbf{x}_{t-1})\|^2 \leq 120 \left(\frac{L\sigma\sqrt{(c\delta + 1/n)(c\delta + 1)}(f(\mathbf{x}_0) - f^*)}{T} \right)^{\frac{2}{3}} + \frac{16L(f(\mathbf{x}_0) - f^*) + 32\sigma^2(c\delta + 1/n)}{T}.$$

Proof. Scaling Lemma 88 by $\frac{4\eta}{\alpha}$ and adding it to Lemma 86 we have for any $t \geq 2$

$$\begin{aligned}
 (\mathbb{E} f(\mathbf{x}_t) - f^*) + \frac{4\eta}{\alpha}(\mathbb{E}\|\bar{\mathbf{e}}_t\|^2 + c\delta\mathbb{E}\|\mathbf{d}_{i,t} - \mathbf{d}_{j,t}\|^2) &\leq (\mathbb{E} f(\mathbf{x}_{t-1}) - f^*) + \frac{4\eta}{\alpha}(\mathbb{E}\|\bar{\mathbf{e}}_{t-1}\|^2 + c\delta\mathbb{E}\|\mathbf{d}_{i,t-1} - \mathbf{d}_{j,t-1}\|^2) \\
 &\quad - \eta(\mathbb{E}\|\bar{\mathbf{e}}_{t-1}\|^2 + c\delta\mathbb{E}\|\mathbf{d}_{i,t-1} - \mathbf{d}_{j,t-1}\|^2) \\
 &\quad - \frac{\eta}{2}\mathbb{E}\|\nabla f(\mathbf{x}_{t-1})\|^2 + \eta(\mathbb{E}\|\bar{\mathbf{e}}_t\|^2 + c\delta\mathbb{E}\|\mathbf{d}_{i,t} - \mathbf{d}_{j,t}\|^2) \\
 &\quad + \frac{\eta}{4}\mathbb{E}\|\nabla f(\mathbf{x}_{t-2})\|^2 + (c\delta + 1/n)16\eta\alpha\sigma^2.
 \end{aligned}$$

Define the constant

$$\xi_t := (\mathbb{E} f(\mathbf{x}_t) - f^*) + \left(\frac{4\eta}{\alpha} - \eta \right) (\mathbb{E}\|\bar{\mathbf{e}}_t\|^2 + c\delta\mathbb{E}\|\mathbf{d}_{i,t} - \mathbf{d}_{j,t}\|^2) + \frac{\eta}{4}\mathbb{E}\|\nabla f(\mathbf{x}_{t-1})\|^2.$$

Then the previously stated inequality can be rearranged as

$$\frac{\eta}{4}\mathbb{E}\|\nabla f(\mathbf{x}_{t-1})\|^2 \leq \xi_{t-1} - \xi_t + (c\delta + 1/n)16\eta\alpha\sigma^2.$$

Also note that $\xi_t \geq 0$ for any t and also for $t = 1$,

$$\begin{aligned}\xi_1 &= \mathbb{E} f(\mathbf{x}_1) - f^* + \left(\frac{4\eta}{\alpha} - \eta \right) (\mathbb{E} \|\bar{\mathbf{e}}_t\|^2 + c\delta \mathbb{E} \|\mathbf{d}_{i,t} - \mathbf{d}_{j,t}\|^2) + \frac{\eta}{4} \mathbb{E} \|\nabla f(\mathbf{x}_0)\|^2 \\ &\leq f(\mathbf{x}_0) - f^* - \frac{\eta}{4} \mathbb{E} \|\nabla f(\mathbf{x}_0)\|^2 + 8\eta\sigma^2(c\delta + 1/n).\end{aligned}$$

Note that here we assumed a batch size of T in the first step to simplify computations. This does not change the asymptotic rate (multiplies it by 2), similar to (Tran-Dinh et al., 2019). This is easy to work around by using changing step-sizes/momentum values as shown by (Cutkosky and Orabona, 2019). Now summing over t and again rearranging gives

$$\frac{1}{\sum_{t=1}^{\ell} \eta} \sum_{t=1}^{\ell} \eta \mathbb{E} \|\nabla f(\mathbf{x}_{t-1})\|^2 \leq \frac{4(f(\mathbf{x}_0) - f^*)}{\sum_{t=1}^{\ell} \eta} + \frac{1}{\sum_{t=1}^{\ell} \eta} \sum_{t=1}^{\ell} 32(c\delta + 1/n)\eta\alpha\sigma^2.$$

For simplicity, let us use a constant $\eta = \min\left(\sqrt[3]{\frac{f(\mathbf{x}_0) - f^*}{T(1536L^2\sigma^2(c\delta+1)^2)}}, \frac{1}{4L}\right)$ for $t \geq 1$ and momentum parameter $\alpha_1 = 1$ and $\alpha = 192L^2\eta^2(c\delta + 1)$ for $t \geq 2$. This simplifies the above inequality to

$$\frac{1}{T} \sum_{t=1}^T \mathbb{E} \|\nabla f(\mathbf{x}_{t-1})\|^2 \leq \frac{4(f(\mathbf{x}_0) - f^*)}{\eta T} + 6144L^2\eta^2(c\delta + 1)(c\delta + 1/n)\sigma^2 + \frac{32\sigma^2(c\delta + 1/n)}{T}.$$

Substituting the appropriate η yields the desired rate. \square

13.7 Additional Experiments

13.7.1 Experiment setups

General setup

THE DEFAULT EXPERIMENT SETUP IS LISTED IN TABLE 13.1. THE DEFAULT HYPERPARAMETERS OF THE AGGREGATORS ARE SUMMARIZED AS FOLLOWS

AGGREGATORS	HYPERPARAMETERS
KRUM	N/A
CM	N/A
RFA	$T = 3$
TM	$b = \delta$
CC	$\tau = 100$

ABOUT FIGURE 6.5. WE HAVE THE FOLLOWING SETUP

- FOR ALL AGGREGATORS EXCEPT MEAN, THERE ARE $n = 25$ WORKERS AND $n\delta = 11$ OF THEM ARE BYZANTINE.
- FOR AGGREGATOR MEAN, THERE ARE $n = 14$ WORKERS AND 0 BYZANTINE WORKERS.
- THE IPM ATTACK HAS STRENGTH OF $\epsilon = 0.1$.

- THE ALIE ATTACK HAS A HYPERPARAMETER z WHICH IS COMPUTED ACCORDING TO (BARUCH ET AL., 2019)

$$z = \max_z \left(\phi(z) < \frac{n - n\delta - s}{n - n\delta} \right)$$

WHERE $s = \lfloor \frac{n}{2} + 1 \rfloor - n\delta$ AND ϕ IS THE CUMULATIVE STANDARD NORMAL FUNCTION. IN OUR SETUP, THE $z \approx 1.06$.

Table 13.1 – Default experimental settings for CIFAR-10 and MNIST.

Dataset	CIFAR-10	MNIST
Architecture	ResNet-20 (He et al., 2016a)	CONV-CONV-DROPOUT-FC-DROPOUT-FC
Training objective	Cross entropy loss	Negative log likelihood loss
Evaluation objective	Top-1 accuracy	Top-1 accuracy
Batch size per worker	32	1
Momentum β	0 or 0.9 or 0.99	0
Learning rate	0.1	$\frac{0.1}{256}$
LR decay	0.1 at epoch 75	No
LR warmup	No	No
# Epochs / # Iterations	100 Epochs	800 Iterations
Weight decay	No	No
Repetitions	2, with varying seeds	2, with varying seeds

Constructing datasets

LONG-TAILNESS. THE MNIST DATASET HAS 10 CLASSES EACH WITH SIMILAR AMOUNT OF SAMPLES. THE LONG-TAILNESS IS ACHIEVED BY SAMPLING CLASS WITH EXPONENTIALLY DECREASING PORTIONS $\gamma \in (0, 1]$. THAT IS, FOR CLASS $i = 1, \dots, 10$, WE ONLY RANDOMLY SAMPLE γ^i PORTION OF ALL SAMPLES IN CLASS i . NOTE THAT THE SAME PROCEDURE HAS TO BE APPLIED TO THE TEST DATASET.

ABOUT DATASET ON BYZANTINE WORKERS. THE TRAINING SET IS DIVIDED BY THE NUMBER OF GOOD WORKERS. SO THE GOOD WORKERS HAS TO FULL INFORMATION OF TRAINING DATASET. THE BYZANTINE WORKER HAS ACCESS TO THE WHOLE TRAINING DATASET.

Running environment

WE SUMMARIZE THE RUNNING ENVIRONMENT OF THIS PAPER AS IN TABLE 13.2.

Table 13.2 – Runtime hardwares and softwares.

CPU	
Model name	Intel (R) Xeon (R) Gold 6132 CPU @ 2.60 GHz
# CPU(s)	56
NUMA node(s)	2
GPU	
Product Name	Tesla V100-SXM2-32GB
CUDA Version	11.0
PyTorch	
Version	1.7.1

13.7.2 Exploring local steps between aggregations

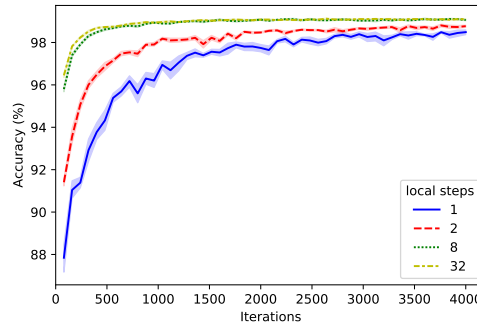


Figure 13.1 – CC with 1, 2, 8, 32, local steps for MNIST dataset.

IN THIS EXPERIMENT, WE COMBINE CC WITH LOCAL SGD AND BENCH MARKED ON MNIST WITHOUT ATTACKER. THE RESULTS IN FIG. 13.1 SHOWS THAT USING HIGHER LOCAL STEPS IMPROVES THE ACCURACY AND CONVERGENCE RATE. IT SUPPORTS THAT CC IS COMPATIBLE WITH LOCALSGD.

13.7.3 Comparison with (Allen-Zhu et al., 2021)

THE RECENT INDEPENDENT WORK SAFEGUARD (ALLEN-ZHU ET AL., 2021) ALSO USES HISTORICAL INFORMATION TO DETECT BYZANTINE WORKERS. HOWEVER, AS WE DISCUSSED EARLIER, THEY ASSUME THAT THE NOISE IN STOCHASTIC GRADIENTS IS BOUNDED ALMOST SURELY INSTEAD OF THE MORE STANDARD ASSUMPTION THAT ONLY THE VARIANCE IS BOUNDED. THEORETICALLY, SUCH STRONG ASSUMPTIONS ARE UNLIKELY TO HOLD (ZHANG ET AL., 2019B) AND EVEN GAUSSIAN NOISE IS EXCLUDED. FURTHER, THE LOWER-BOUNDS OF (ARJEVANI ET AL., 2019) NO LONGER APPLY, AND THUS THEIR ALGORITHM MAY BE SUB-OPTIMAL. PRACTICALLY, THEIR ALGORITHM REMOVES SUSPECTED WORKERS EITHER PERMANENTLY (A DECISION OF HIGH RISK), OR RESETS THE LIST OF SUSPECTS AT EACH WINDOW BOUNDARY (WHICH IS SENSITIVE TO

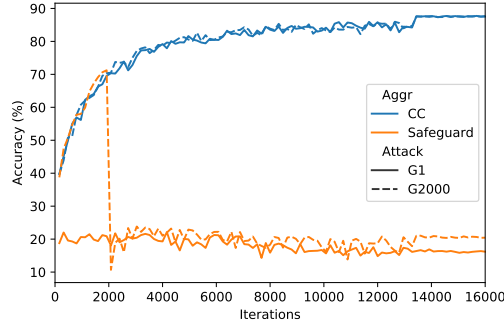


Figure 13.2 – Comparing **CC** ($\tau = 100$) with Safeguard (Allen-Zhu et al., 2021) ($T_0 = 1, T_1 = 6, \mathfrak{T}_0 = 20, \mathfrak{T}_1 = 50$). The Byzantine workers send to the server vectors from a Gaussian distribution with standard deviation of 10^8 . The “G1” attack inject attack at the 1st iteration while the “G2000” attack inject attack at the 2000th iteration. There are 10 nodes in total and 4 of them are Byzantine. The underlying dataset is Cifar10. We use batch size 32 and learning rate 0.1.

THE CHOICE OF HYPERPARAMETERS). HAVING SAID THAT, (ALLEN-ZHU ET AL., 2021) PROVE CONVERGENCE TO A LOCAL MINIMUM INSTEAD OF TO A SADDLE POINT AS WE DO HERE. IN THIS SECION, WE CONDUCT FURTHER EMPIRICAL COMPARISON OF SAFEGUARD AND CENTERED CLIP **CC**.

FIRST, NOTE THAT ALGORITHM 1 IN (ALLEN-ZHU ET AL., 2021) IS VULNERABLE TO SIMPLE ATTACKS, E.G. SENDING AN ARBITRARY VECTOR OF VERY LARGE MAGNITUDE, WHILE **CC** IS NOT. THIS IS BECAUSE SAFEGUARD USES INFORMATION FROM THE PREVIOUS STEP TO FILTER IN THE CURRENT STEP. THIS IS NECESSARY IN ORDER TO MAKE THE ALGORITHM AMENABLE TO ANALYSIS. THIS MEANS THAT EVEN IF A BYZANTINE WORKER SENDS A VERY LARGE BAD UPDATE, THE ALGORITHM WILL APPLY IT ONCE AND FILTER OUT THE WORKER ONLY FROM THE NEXT ROUND ONWARD. THUS, ALL BYZANTINE WORKERS CAN ENSURE THAT THEIR UPDATE IS INCORPORATED AT LEAST ONCE. WHILE THEORETICALLY THIS MIGHT NOT BE PROBLEMATIC SINCE THE INFLUENCE OF A SINGLE UPDATE IS LIMITED, IN PRACTICE THIS MEANS THAT THE BYZANTINE WORKERS CAN PUSH THE TRAINING PROCESS TO ENCOUNTER NaNs, ENSURING NO CHANCE OF RECOVERY.

TO DEMONSTRATE THE EFFECT, WE APPLY THE GAUSSIAN ATTACK TO SAFEGUARD AND **CC** AT $t = 1$ (G1) AND $t = 2000$ (G2000). THE GAUSSIAN ATTACKER SENDS TO THE SERVER VECTORS OF GAUSSIAN DISTRIBUTION OF STANDARD DEVIATION 10^8 . SINCE THE WORKERS BEHAVE CORRECTLY UNTIL $t - 1$, THEY ALL BELONG TO \mathbf{GOOD}_t AND THEIR UPDATES ARE INCORPORATED. WHILE THE BYZANTINE WORKER IS REMOVED FROM \mathbf{GOOD}_{t+1} , THE ATTACK ALREADY SUCCEEDED AND THERE IS NO CHANCE OF RECOVERY. WE SHOW THE EXPERIMENTAL RESULTS IN FIGURE 13.2. IN CONTRAST, **CC** (EVEN WITHOUT MOMENTUM) EASILY DEFENDS AGAINST SUCH ATTACKS.

SECONDLY, SAFEGUARD REQUIRES TUNING ADDITIONAL PARAMETERS (E.G. $\mathfrak{T}_0, \mathfrak{T}_1$) FOR EACH

KIND OF ATTACK WHILE CC DOES NOT. FOR EXAMPLE, SAFEGUARD USES $\mathfrak{T}_0 = 1$, $\mathfrak{T}_1 = 2$ FOR BIT-FLIPPING ATTACK (ALLEN-ZHU ET AL., 2021, APPENDIX C.2.1) AND $\mathfrak{T}_0 = 2$, $\mathfrak{T}_1 = 7$ FOR LABEL-FLIPPING ATTACK (ALLEN-ZHU ET AL., 2021, APPENDIX C.2.3). HOWEVER, BY THE DEFINITION OF BYZANTINE ATTACK, THE ATTACKER IS ALLOWED TO ADAPTIVELY CHANGE ATTACKS *after* TUNING. THIS MAKES IT CRUCIAL TO ENSURE THAT ANY BYZANTINE ROBUST ALGORITHM WORKS WITHOUT ADDITIONAL TUNING. IN CONTRAST, CC USES $\tau = 100$ AND $l = 1$ FOR ALL EXPERIMENTS IN THE PAPER UNLESS OTHERWISE CLARIFIED.

14 Appendix for heterogeneous Byzantine robust learning using resampling

14.1 Experiment setup and additional experiments

14.1.1 Experiment setup

General setup

THE DEFAULT EXPERIMENT SETUP IS LISTED IN TABLE 14.1.

Table 14.1 – Default experimental settings for MNIST

Dataset	MNIST
Architecture	CONV-CONV-DROPOUT-FC-DROPOUT-FC
Training objective	Negative log likelihood loss
Evaluation objective	Top-1 accuracy
Batch size	$32 \times \text{number of workers}$
Momentum	0 or 0.9
Learning rate	0.01
LR decay	No
LR warmup	No
# Iterations	600 or 4500
Weight decay	No
Repetitions	3, with varying seeds
Reported metric	Mean test accuracy over the last 150 iterations

BY DEFAULT THE HYPERPARAMETERS OF THE AGGREGATORS ARE SUMMARIZED AS FOLLOWS

AGGREGATORS	HYPERPARAMETERS
KRUM	N/A
CM	N/A
RFA	$T = 8$
TM	$b = f$
CCLIP	$\tau = \frac{10}{1-\beta}$

Constructing datasets

THE MNIST DATASET HAS 10 CLASSES EACH WITH SIMILAR AMOUNT OF SAMPLES. IN THIS PART, WE DISCUSS HOW TO PROCESS AND DISTRIBUTE MNIST TO EACH WORKERS IN ORDER TO ACHIEVE LONG-TAILNESS AND HETEROGENEITY.

LONG-TAILNESS. THE LONG-TAILNESS (*-LT) IS ACHIEVED BY SAMPLING CLASS WITH EXPONENTIALLY DECREASING PORTIONS $\gamma \in (0, 1]$. THAT IS, FOR CLASS $i \in [10]$, WE ONLY RANDOMLY SAMPLE γ^i PORTION OF ALL SAMPLES IN CLASS i . WE DEFINE α AS THE RATIO OF THE LARGEST CLASS OVER THE SMALLEST CLASS, WHICH CAN BE WRITTEN AS $\alpha = \frac{1}{\gamma^9}$. FOR EXAMPLE, IF $\gamma = 1$, THEN ALL CLASSES HAVE SAME AMOUNT OF SAMPLES AND THUS $\alpha = 1$; IF $\gamma = 0.5$ THEN

14.1. Experiment setup and additional experiments

$\alpha = 2^9 = 512$. NOTE THAT THE SAME PROCEDURE HAS TO BE APPLIED TO THE TEST DATASET.

HETEROGENEITY. STEPS TO CONSTRUCT IID/NON-IID DATASET FROM MNIST DATASET

1. SORT THE TRAINING DATASET BY ITS LABELS.
2. EVENLY DIVIDE THE SORTED TRAINING DATASET INTO CHUNKS OF SAME SIZE. THE NUMBER OF CHUNKS EQUALS THE NUMBER OF GOOD WORKERS. IF THE LAST CHUNK HAS FEWER SAMPLES, WE AUGMENT IT WITH SAMPLES FROM ITSELF.
3. SHUFFLE THE SAMPLES WITHIN THE SAME WORKER.

HETEROGENEITY + LONG-TAILNESS. FIRST TRANSFORM THE TRAINING DATASET INTO LONG-TAIL DATASET, THEN FEED IT TO THE PREVIOUS PROCEDURE TO INTRODUCE HETEROGENEITY.

ABOUT DATASET ON BYZANTINE WORKERS. THE TRAINING SET IS DIVIDED BY THE NUMBER OF GOOD WORKERS. SO THE GOOD WORKERS HAS TO FULL INFORMATION OF TRAINING DATASET. THE BYZANTINE WORKER HAS ACCESS TO THE WHOLE TRAINING DATASET.

Setup for each experiment

IN TABLE 14.2, WE LIST THE HYPERPARAMETERS FOR THE EXPERIMENTS.

Table 14.2 – Setups for each experiment.

	n	f	momentum	Iters	LT	NonIID
Table 7.1	20	0	0	4500	$\alpha = 1, \alpha = 500$	iid/ non-iid
Table 7.2	25	5	0	TBD	$\alpha = 1$ (balanced)	iid/ non-iid
Table 7.3	20	0	0	4500	$\alpha = 1, \alpha = 500$	iid/ non-iid
Table 7.4	25	5	0	TBD	$\alpha = 1$ (balanced)	iid/ non-iid
Figure 7.2	25	5	0 / 0.9	600	$\alpha = 1$ (balanced)	non-iid
Figure 7.3	53	5	0 / 0.9	600	$\alpha = 1$ (balanced)	non-iid
Figure 14.1	25	5	0 / 0.5 / 0.9 / 0.99	600	$\alpha = 1$ (balanced)	non-iid
Figure 14.2	25	5	0 / 0.5 / 0.9 / 0.99	1200	$\alpha = 1$ (balanced)	non-iid
Figure 14.3	20	3	0	1200	$\alpha = 1$ (balanced)	non-iid
Figure 14.4	25	5	0	3000	$\alpha = 1$ (balanced)	non-iid
Figure 14.5	24	3	0	1200	$\alpha = 1$ (balanced)	non-iid

IPM ATTACK IN FIGURE 7.2 AND FIGURE 7.3. WE SET THE STRENGTH OF THE ATTACK $\epsilon = 0.1$.

ALIE ATTACK IN IN FIGURE 7.2. THE HYPERPARAMETER z FOR ALIE IS COMPUTED ACCORDING TO (BARUCH ET AL., 2019)

$$z = \max_z \left(\phi(z) < \frac{n-f-s}{n-f} \right)$$

Chapter 14. Appendix for heterogeneous Byzantine robust learning using resampling

WHERE $s = \lfloor \frac{n}{2} + 1 \rfloor - f$ AND ϕ IS THE CUMULATIVE STANDARD NORMAL FUNCTION. IN OUR SETUP, THE $z \approx 0.25$.

Running environment

WE SUMMARIZE THE RUNNING ENVIRONMENT OF THIS PAPER AS IN TABLE 14.3.

Table 14.3 – Runtime hardwares and softwares.

CPU	
Model name	Intel (R) Xeon (R) Gold 6132 CPU @ 2.60 GHz
# CPU(s)	56
NUMA node(s)	2
GPU	
Product Name	Tesla V100-SXM2-32GB
CUDA Version	11.0
PyTorch	
Version	1.7.1

THE GPU-HOUR SPENT ON THE EXPERIMENTS ARE ESTIMATED AS FOLLOWS. EACH RUN TAKES 4 GPU MINUTES AND THERE ARE 15 RUNS IN PARALLEL ON GPU.

	# RUNS	GPU HOURS
TABLE 7.1:	15 * 3 (REPEATS) = 45	0.2
TABLE 7.2:	15 * 3 (REPEATS) = 45	0.2
TABLE 7.3:	15 * 3 (REPEATS) = 45	0.2
TABLE 7.4:	15 * 3 (REPEATS) = 45	0.2
FIGURE 7.2	25 * 4 * 3 (REPEATS) = 300	1.33
FIGURE 7.3	2 * 6 * 3 (REPEATS) = 36	0.16
TOTAL	516	2.29

14.1.2 Additional experiments

Clipping radius scaling

THE RADIUS τ OF CCLIP DEPENDS ON THE NORM OF GOOD GRADIENTS. HOWEVER, PYTORCH IMPLEMENTS SGD WITH MOMENTUM USING THE FOLLOWING FORMULA

$$\mathbf{m}_i^t = \beta \mathbf{m}_i^{t-1} + \mathbf{g}_i(\mathbf{x}^{t-1}) \quad \text{FOR EVERY } i \in \mathcal{G}$$

WHICH MAY LEADS TO THE INCREASE IN THE GRADIENT NORM.

GRADIENT NORMS. IN FIGURE 14.1 WE PRESENT THE AVERAGED GRADIENT NORM FROM ALL GOOD WORKERS. HERE WE USE CCLIP AS THE AGGREGATOR AND $\tau = \frac{1}{1-\beta}$. THE NORM OF GRADIENTS ARE COMPUTED BEFORE AGGREGATION. EVEN THOUGH THE DATASET ON WORKERS ARE NON-IID, THE GRADIENT NORMS ARE ROUGHLY OF SAME ORDER. THE GRADIENT DISSIMILARITY G^2 ALSO INCREASES ACCORDINGLY.

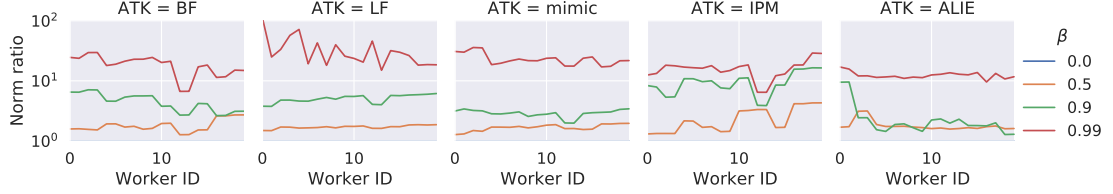


Figure 14.1 – The ratio of norm of good gradients with momentum β over no momentum under different attacks.

SCALED CLIPPING RADIUS. AS THE GRADIENT NORM INCREASES WITH MOMENTUM β , THE CLIPPING RADIUS SHOULD INCREASE ACCORDINGLY. IN FIGURE 14.2 WE COMPARE 3 SCHEMES: 1) NO SCALING ($\tau = 10, \beta = 0$); 2) *linear* SCALING $\frac{10}{1-\beta}$; 3) *sqrt* SCALING $\frac{10}{\sqrt{1-\beta}}$. THE NO SCALING SCHEME CONVERGES BUT SLOWER WHILE WITH MOMENTUM. THE LINEAR SCALING IS USUALLY BETTER THAN *sqrt* SCALING AND WITH RESAMPLING IT BECOMES MORE STABLE. HOWEVER, THE SCALED CLIPPING RADIUS FAILS FOR $\beta = 0.99$ UNDER LABEL FLIPPING ATTACK. THIS IS BECAUSE THE GRADIENT CAN BE VERY LARGE AND G^2 DOMINATES. SO IN GENERAL, A LINEAR SCALING OF CLIPPING RADIUS WITH MOMENTUM $\beta = 0.9$ WOULD BE A GOOD CHOICE.

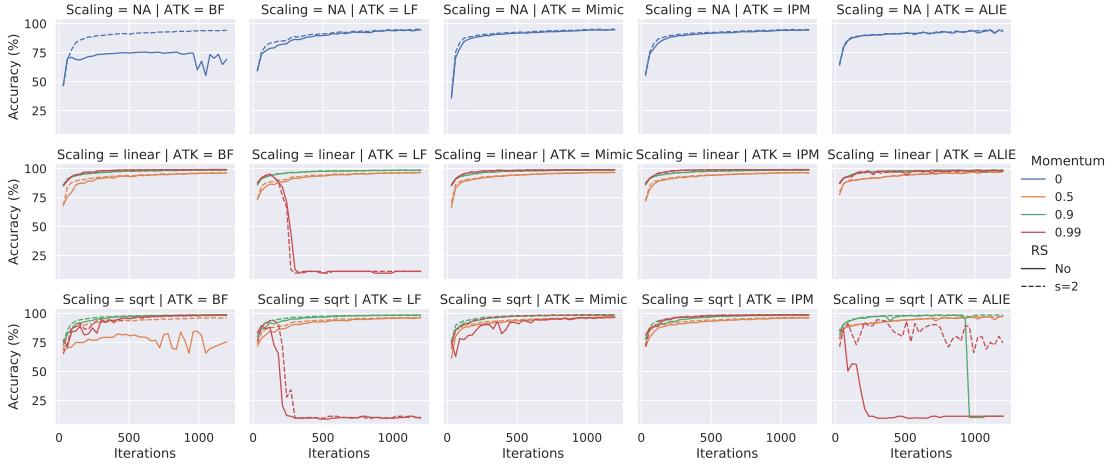


Figure 14.2 – Convergence of CCLIP with $\tau = 10, \frac{10}{1-\beta}, \frac{10}{\sqrt{1-\beta}}$ for $\beta = 0, 0.5, 0.9, 0.99$. The RS stands for resampling and s is the resampling hyperparameter.

Demonstration of effects of resampling through the selections of Krum

IN THE MAIN TEXT WE HAVE THEORETICALLY SHOW THAT RESAMPLING HELPS AGGREGATORS ALLEVIATE THE IMPACT OF NON-IID. IN THIS SECTION WE EMPIRICALLY SHOW THAT AFTER

RESAMPLING AGGREGATORS CAN INCORPORATE UPDATES MORE EVENLY FROM GOOD WORKERS AND THEREFORE THE PROBLEM OF NON-IID AMONG GOOD WORKERS IS LESS SIGNIFICANT. SINCE KRUM OUTPUTS THE ID OF THE SELECTED DEVICE, IT IS VERY CONVENIENT TO RECORD THE FREQUENCY OF EACH WORKER BEING SELECTED. SINCE RESAMPLING REPLICATES EACH WORKER FOR s TIMES, WE DIVIDE THEIR FREQUENCIES BY s FOR NORMALIZATION. FROM FIGURE 14.3, WE CAN SEE THAT WITHOUT RESAMPLING KRUM BASICALLY ALWAYS SELECTS UPDATES FROM BYZANTINE WORKERS WHILE WITH LARGER s , THE SELECTION BECOMES MORE EVENLY DISTRIBUTED.

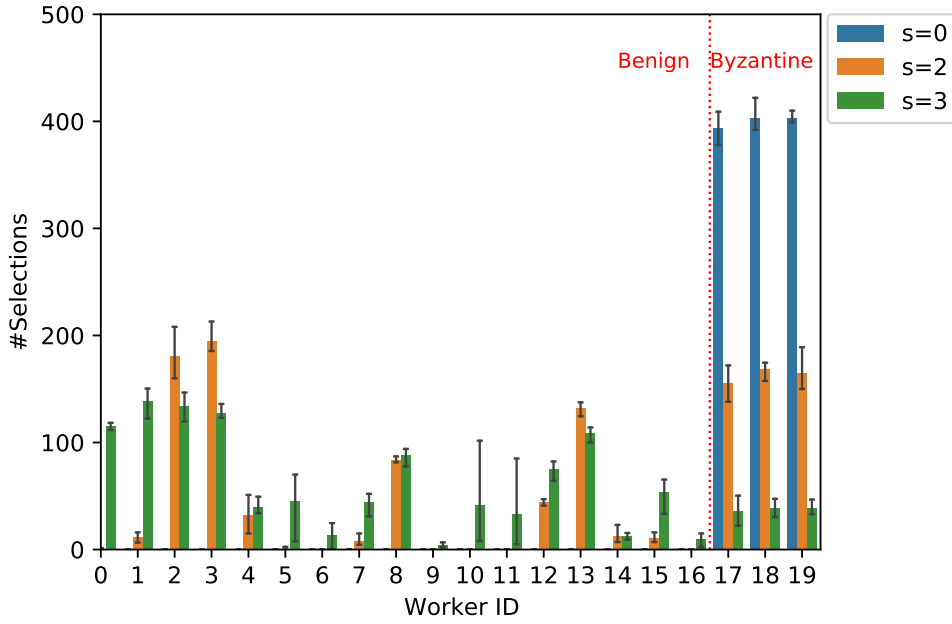


Figure 14.3 – The selected workers of KRUM for resampling coefficient $s = 0, 2, 3$. There are 20 workers and the last 3 workers (worker id=17,18,19) are Byzantine with label-flipping attack.

Overparameterization

THE ARCHITECTURE OF THE NEURAL NET USED IN THE EXPERIMENTS CAN BE SCALED TO MAKE IT OVERPARAMETERIZED. WE ADD MORE PARAMETERS TO THE MODEL BY MULTIPLYING THE CHANNELS OF 2D Conv layer and fully connected layer by a factor of ‘scale’. SO THE ORIGINAL MODEL HAS A SCALE OF 1. WE SHOW THE TRAINING LOSSES DECREASE FASTER FOR OVERPARAMETERIZED MODELS IN FIGURE 14.4. AS WE CAN SEE, THE CONVERGENCE BEHAVIORS ARE SIMILAR FOR DIFFERENT MODEL SCALES WITH OVERPARAMETERIZED MODELS HAVING SMALLER TRAINING LOSS DESPITE THE EXISTENCE OF BYZANTINE WORKERS.

14.1. Experiment setup and additional experiments

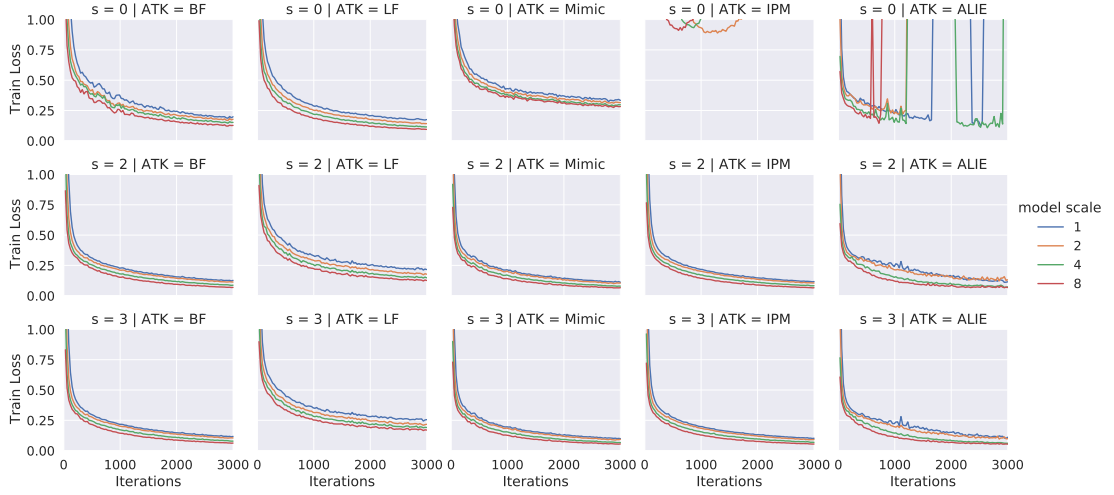


Figure 14.4 – The training loss of models of different levels of overparameterization.

Bucketing - variant of resampling

A VARIANT OF ALGORITHM 9 IS TO NOT REPEAT THE GRADIENTS FOR s TIMES, BUT RATHER n GRADIENTS INTO $\lceil n/s \rceil$ BUCKETS. THE RESULTS IN FIGURE 14.5 SUGGEST THAT THE CONVERGENCE RATE OF BUCKETING AND RESAMPLING IS ALMOST THE SAME. SO AGGREGATORS CAN BENEFIT MORE FROM BUCKETING AS IT REDUCES THE NUMBER OF INPUT GRADIENTS AND THEREFORE REDUCE THE COMPLEXITY.

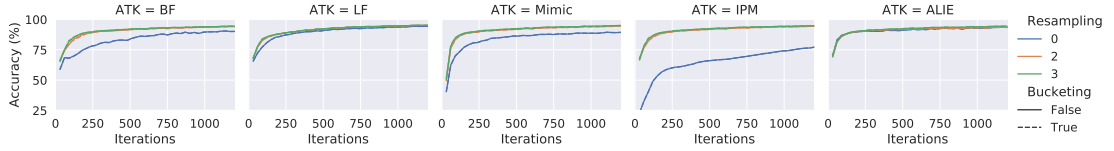


Figure 14.5 – The convergence SGD with bucketing and resampling under different attacks. The underlying aggregator is RFA.

Fashion-MNIST

IN THIS SUBSECTION, WE DEMONSTRATE THAT OUR ALGORITHM ALSO WORKS ON MODERN DATASET LIKE FASHION-MNIST (XIAO ET AL., 2017). SINCE THE FASHION-MNIST IS DESIGNED TO BE A DROP-IN REPLACEMENT OF MNIST, WE CONDUCT EXPERIMENTS ON FASHION-MNIST WITH 12 WORKERS WHERE 2 OF THEM ARE BYZANTINE. THE RESULTS ARE PRESENTED IN FIGURE 14.6.

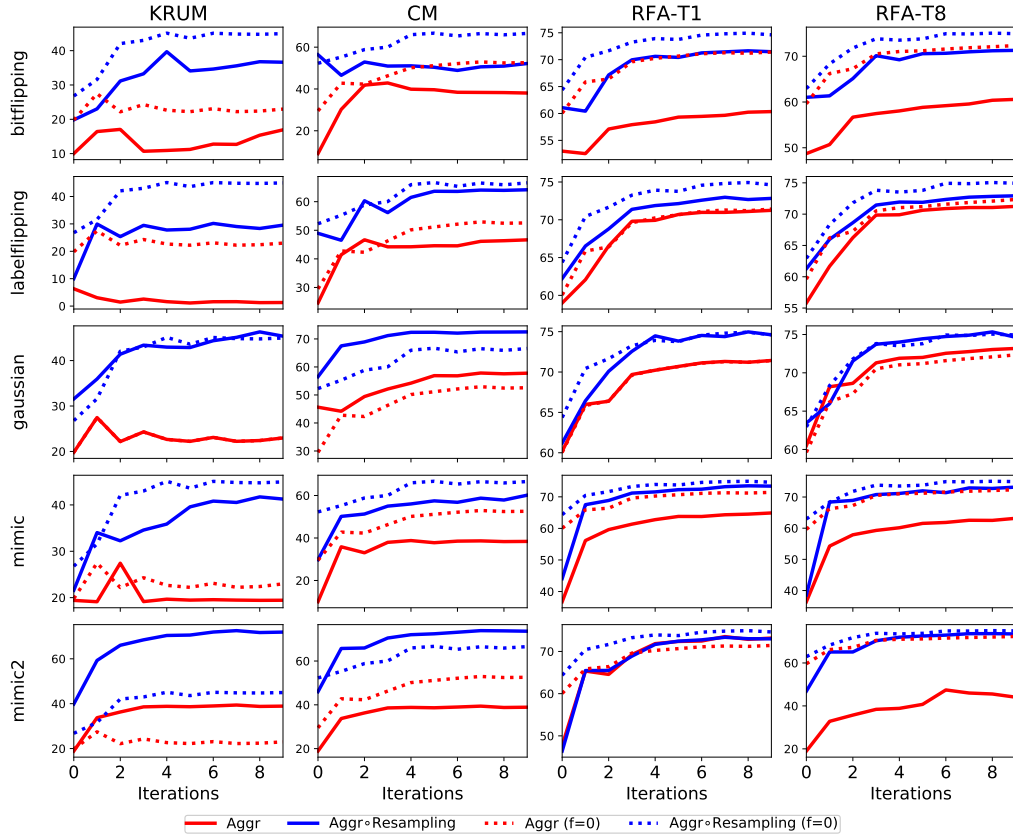


Figure 14.6 – Test accuracies of KRUM, CM, RFA under 5 kinds of attacks (and without attack) on non-iid Fashion-MNIST datasets. There are 12 workers and 2 of them are Byzantine according to each attack row. Columns show each aggregation rule applied without (red) and with resampling (blue). Dotted lines for comparison are showing the same method without any Byzantine workers ($f = 0$). For RFA, T1, T8 refers to the number of inner iterations of Weiszfeld's algorithm.

14.1. Experiment setup and additional experiments

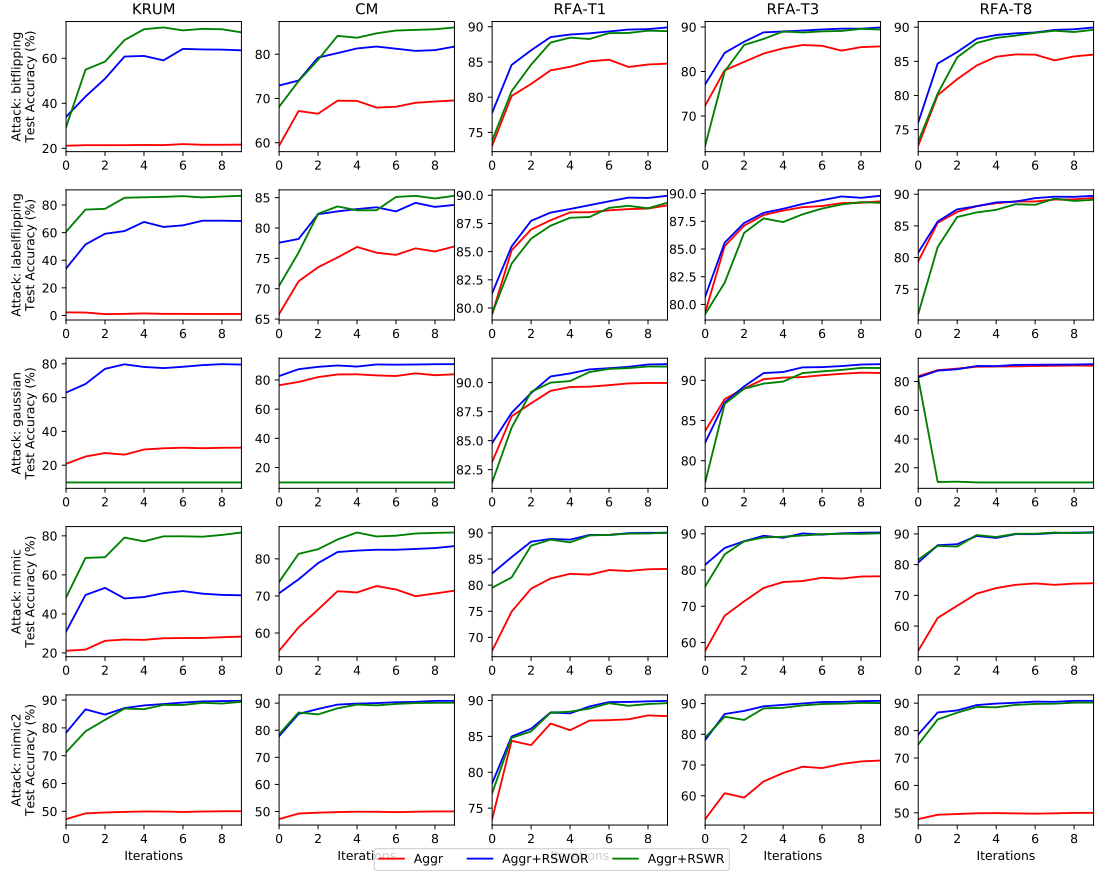


Figure 14.7 – Comparing 3 aggregation rules under 5 kinds of attacks on non-iid datasets. There are 10 workers and 2 of them are Byzantine. In the grid of experiments, same aggregation rules are used in the same column and same attacks are applied to the same row. The aggregation rules are KRUM (Blanchard et al., 2017), CM (Yin et al., 2018a), RFA (Pillutla et al., 2019). The RFA-T1, T3, T8 refers to the number of inner iterations.

Resampling or fixed grouping

IN (CHEN ET AL., 2017), WORKERS ARE GROUPED AT THE BEGINNING OF TRAINING, AND THEY ARE TRAINED ON IID DATASETS. IN CONTRAST, RESAMPLING IS PERFORMED EVERY ROUND, AND APPLIES TO NON-IID DATASETS. IF A BYZANTINE WORKER CAN PREDICT THE RANDOM BITS ON SERVER, RESAMPLING BECOMES GROUPING IN EACH ROUND WHICH IS STILL STRONGER THAN (CHEN ET AL., 2017).

IN FIGURE 14.8, WE COMPARE KRUM ◦RESAMPLING WITH VANILLA KRUM AND KRUM WITH FIXED GROUPING. AS WE CAN SEE, THE FIXED GROUPING HAS BETTER ACCURACY THAN VANILLA KRUM, BUT WEAKER THAN RESAMPLING AS WE EXPECTED.

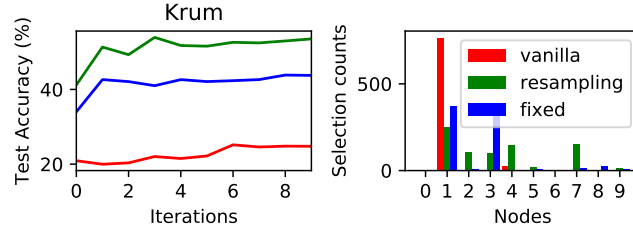


Figure 14.8 – Comparison with no resampling, and fixed grouping for Krum on non-iid datasets.

14.2 Implementing the mimic attack

THE SECTION 7.4.2 DESCRIBES THE IDEA AND FORMULATION OF THE MIMIC ATTACK. IN THIS SECTION, WE DISCUSS HOW TO IMPLEMENT THE MIMIC ATTACK EFFICIENTLY. FIRST, REWRITE THE MIMIC ATTACK IN ITS ONLINE VERSION AT TIME $t \in \mathcal{T}_0$

$$\mathbf{z}^t = \underset{\|\mathbf{z}\|=1}{\operatorname{argmax}} h^t(\mathbf{z})$$

WHERE $\boldsymbol{\mu}^t = \frac{1}{|\mathcal{G}|t} \sum_{\tau \leq t} \sum_{i \in \mathcal{G}} \mathbf{x}_i^\tau$ AND

$$h^t(\mathbf{z}) = \mathbf{z}^\top \left(\sum_{\tau \leq t} \sum_{i \in \mathcal{G}} (\mathbf{x}_i^\tau - \boldsymbol{\mu}^t)(\mathbf{x}_i^\tau - \boldsymbol{\mu}^t)^\top \right) \mathbf{z}.$$

THUS WE CAN ITERATIVELY UPDATE $\boldsymbol{\mu}^t$ BY

$$\boldsymbol{\mu}^{t+1} = \frac{t}{1+t} \boldsymbol{\mu}^t + \frac{1}{1+t} \frac{1}{|\mathcal{G}|} \sum_{i \in \mathcal{G}} \mathbf{x}_i^{t+1},$$

AND THEN

$$\begin{aligned} \underset{\|\mathbf{z}\|=1}{\operatorname{argmax}} h^{t+1}(\mathbf{z}) &\approx \frac{t}{1+t} \mathbf{z}^t + \frac{1}{1+t} \underset{\|\mathbf{z}\|=1}{\operatorname{argmax}} \mathbf{z}^\top \left(\sum_{i \in \mathcal{G}} (\mathbf{x}_i^{t+1} - \boldsymbol{\mu}^{t+1})(\mathbf{x}_i^{t+1} - \boldsymbol{\mu}^{t+1})^\top \right) \mathbf{z} \\ &\approx \frac{t}{1+t} \mathbf{z}^t + \frac{1}{1+t} \left(\sum_{i \in \mathcal{G}} (\mathbf{x}_i^{t+1} - \boldsymbol{\mu}^{t+1})(\mathbf{x}_i^{t+1} - \boldsymbol{\mu}^{t+1})^\top \right) \mathbf{z}^t \end{aligned}$$

THE ABOVE ALGORITHM COORESPONDS TO OJA'S METHOD FOR COMPUTING THE TOP EIGEN-VECTOR IN A STREAMING FASHION (OJA, 1982). THEN, IN EACH SUBSEQUENT ITERATION t , WE PICK

$$i_\star^t = \underset{i \in \mathcal{G}}{\operatorname{argmax}} \mathbf{z}^\top \mathbf{x}_i^t.$$

14.3 Constructing a robust aggregator using resampling

14.3.1 Supporting lemmas

WE FIRST START WITH PROVING THE MAIN RESAMPLING LEMMA 19 RESTATED BELOW.

Lemma' 19. *Suppose we are given n independent (but not identical) random vectors $\{\mathbf{x}_1, \dots, \mathbf{x}_n\}$ such that a good subset $\mathcal{G} \subseteq [n]$ of size at least $|\mathcal{G}| \geq n(1 - \delta)$ satisfies:*

$$\mathbb{E}[\langle \mathbf{x}_i, \mathbf{x}_j \rangle] = \langle \mathbb{E}[\mathbf{x}_i], \mathbb{E}[\mathbf{x}_j] \rangle \quad \text{and} \quad \mathbb{E}\|\mathbf{x}_i - \mathbf{x}_j\|^2 \leq \rho^2, \quad \text{for any fixed } i, j \in \mathcal{G}.$$

Define $\bar{\mathbf{x}} := \frac{1}{|\mathcal{G}|} \sum_{j \in \mathcal{G}} \mathbf{x}_j$. Let the outputs after s -resampling without replacement be $\{\mathbf{y}_1, \dots, \mathbf{y}_n\}$. Then, a subset of the outputs $\tilde{\mathcal{G}} \subseteq [n]$ of size at least $|\tilde{\mathcal{G}}| \geq n(1 - \delta s)$ satisfy

$$\mathbb{E}[\mathbf{y}_i] = \mathbb{E}[\bar{\mathbf{x}}] \quad \text{and} \quad \mathbb{E}\|\mathbf{y}_i - \mathbf{y}_j\| \leq \frac{\rho^2}{s} \quad \text{for any fixed } i, j \in \tilde{\mathcal{G}}.$$

Proof. Recall that the duplicated vectors $\mathbf{v}_1, \dots, \mathbf{v}_{s \cdot n}$ were defined to be $\mathbf{v}_k = \mathbf{x}_{\lceil k/s \rceil}$ and further

$$\mathbf{y}_i = \frac{1}{s} \sum_{k=s(i-1)+1}^{s \cdot i} \mathbf{v}_{\pi(k)},$$

for some permutation π over $[s \cdot n]$. Then, define the *new* good set

$$\tilde{\mathcal{G}} = \{i \in [n] \mid \{\pi(s(i-1)+1), \dots, \pi(s \cdot i)\} \subseteq \mathcal{G}\}.$$

$\tilde{\mathcal{G}}$ contains the set of all the resampled vectors which are made up of only good vectors i.e. are uninfluenced by any Byzantine vector. Since $|\mathcal{B}| \leq \delta n$ and each vector is replicated at most s times, we have that $|\tilde{\mathcal{G}}| \geq (1 - \delta s)n$. Now, for any fixed $i \in \tilde{\mathcal{G}}$, let us look at the conditional expectation over the random permutation π we have

$$\mathbb{E}_{\pi}[\mathbf{y}_i \mid i \in \tilde{\mathcal{G}}] = \frac{1}{|\mathcal{G}|} \sum_{j \in \mathcal{G}} \mathbf{x}_j = \bar{\mathbf{x}}.$$

This yields the first part of the lemma. Now we analyze the variance. Let us define the indices of \mathbf{x} used to compute \mathbf{y}_i as

$$B_i := \{\lceil \pi(s(i-1)+1)/s \rceil, \dots, \lceil \pi(s \cdot i)/s \rceil\}.$$

Thus, we can write $\mathbf{y}_i = \frac{1}{s} \sum_{k \in B_i} \mathbf{x}_k$. Further, $|B_i| = s$ for any i , and $B_i \subseteq \mathcal{G}$ if $i \in \tilde{\mathcal{G}}$. With this,

for any fixed $i, j \in \tilde{\mathcal{G}}$ the variance can be written as

$$\begin{aligned}\mathbb{E}\|\mathbf{y}_i - \mathbf{y}_j\|^2 &= \mathbb{E}\left\|\frac{1}{s} \sum_{k \in B_i} \mathbf{x}_k - \frac{1}{s} \sum_{l \in B_j} \mathbf{x}_l\right\|^2 \\ &= \frac{1}{s^2} \mathbb{E}\left\|\sum_{k \in B_i \setminus B_j} \mathbf{x}_k - \sum_{l \in B_j \setminus B_i} \mathbf{x}_l\right\|^2 \\ &= \frac{s - |B_i \cap B_j|}{s^2} \rho^2 \\ &\leq \frac{\rho^2}{s}.\end{aligned}$$

□

THIS ADDITIONAL LEMMA ABOUT THE MAXIMUM EXPECTED DISTANCE BETWEEN GOOD WORKERS WILL ALSO BE USEFUL LATER.

Lemma 89 (maximum good distance). *Suppose we are given the output of resampling $\mathbf{y}_1, \dots, \mathbf{y}_n$ which satisfy for any fixed $i \in \tilde{\mathcal{G}}$, $\mathbb{E}[\mathbf{y}_i] = \boldsymbol{\mu}$ and $\mathbb{E}\|\mathbf{y}_i - \boldsymbol{\mu}\|^2 \leq \rho^2/s$. Then, we have*

$$\mathbb{E}\left[\max_{i \in \tilde{\mathcal{G}}} \|\mathbf{y}_i - \boldsymbol{\mu}\|^2\right] \leq n\rho^2/s.$$

Further, there exist instances where

$$\mathbb{E}\left[\max_{i \in \tilde{\mathcal{G}}} \|\mathbf{y}_i - \boldsymbol{\mu}\|^2\right] \geq \Omega(n\rho^2/s).$$

Proof. For the upper bound, we simply use

$$\mathbb{E}\left[\max_{i \in \tilde{\mathcal{G}}} \|\mathbf{y}_i - \boldsymbol{\mu}\|^2\right] \leq \sum_{i \in \tilde{\mathcal{G}}} \mathbb{E}\|\mathbf{y}_i - \boldsymbol{\mu}\|^2 \leq n\rho^2/s.$$

For the lower bound, let $\tilde{\mathcal{G}} = [n]$ and consider $\mathbf{y}_i \sim \tilde{\rho}\sqrt{n}\text{Bern}(p = \frac{1}{n})$. This means \mathbf{y}_i is either 0 or $\tilde{\rho}\sqrt{n}$. Further, its variance is clearly bounded by $\tilde{\rho}^2$. Upon drawing n samples, the probability of seeing at least 1 $\mathbf{y}_j = \tilde{\rho}\sqrt{n}$ is

$$1 - \Pr(\mathbf{y}_i = 0 \ \forall i \in [n]) = 1 - (1 - \frac{1}{n})^n \geq 1 - 1/e \geq 0.5.$$

Thus, with probability at least 0.5 we have

$$\max_{i \in [n]} \|\mathbf{y}_i - \boldsymbol{\mu}\|^2 \geq n\tilde{\rho}^2/2.$$

This directly proves our lower bound by defining $\tilde{\rho}^2 := \rho^2/s$. Note that this lemma can be tightened if we make stronger assumptions on the noise such as $\mathbb{E}\|\mathbf{y}_i - \boldsymbol{\mu}\|^r \leq (\rho/\sqrt{s})^r$ for some

large $r \geq 2$. However, we focus on using standard stochastic assumptions ($r = 2$) in this work. \square

14.3.2 Proofs of robustness

LET $\{\mathbf{y}_1, \dots, \mathbf{y}_n\}$ BE THE RESAMPLED VECTORS WITH RESAMPLING USING $s = \frac{\delta_{\max}}{\delta}$. BY LEMMA 19, WE HAVE THAT THERE IS A $\tilde{\mathcal{G}} \subseteq [n]$ OF SIZE $|\tilde{\mathcal{G}}| > n(1 - \delta_{\max})$ WHICH SATISFIES FOR ANY FIXED $i, j \in \tilde{\mathcal{G}}$

$$\mathbb{E}\|\mathbf{y}_i - \mathbf{y}_j\|^2 \leq \frac{\delta \rho^2}{\delta_{\max}} =: \tilde{\rho}^2.$$

THIS OBSERVATION WILL BE COMBINED WITH EACH OF THE ALGORITHMS TO OBTAIN ROBUSTNESS GUARANTEES.

ROBUSTNESS OF KRUM. WE NOW PROVE THAT KRUM WHEN COMBINED WITH RESAMPLING IS A ROBUST AGGREGATOR. WE CAN REWRITE THE OUTPUT OF KRUM AS THE FOLLOWING FOR $\delta_{\max} = 1/4 - \nu$ FOR SOME ARBITRARILY SMALL POSITIVE NUMBER $\nu \in (0, 1/4)$:

$$\text{KRUM}(\mathbf{y}_1, \dots, \mathbf{y}_n) = \underset{\mathbf{y}_i}{\operatorname{argmin}} \min_{|\mathcal{S}|=3n/4} \max_{j \in \mathcal{S}} \|\mathbf{y}_i - \mathbf{y}_j\|.$$

LET \mathcal{S}^* AND k^* BE THE QUANTITIES WHICH MINIMIZE THE OPTIMIZATION PROBLEM SOLVED BY KRUM.

THE MAIN DIFFICULTY OF ANALYZING KRUM IS THAT EVEN IF WE SUCCEED IN SELECTING A $k^* \in \tilde{\mathcal{G}}$, k^* DEPENDS ON THE SAMPLING. HENCE, WE **CANNOT** CLAIM THAT THE ERROR IS BOUNDED BY $\tilde{\rho}^2$ I.E ¹

$$\mathbb{E}\|\mathbf{y}_{k^*} - \mathbf{y}_j\|^2 \not\leq \tilde{\rho}^2 \text{ FOR SOME FIXED } j \in \tilde{\mathcal{G}}.$$

THIS IS BECAUSE THE VARIANCE IS BOUNDED BY $\tilde{\rho}^2$ ONLY FOR A *fixed* i , AND NOT A DATA DEPENDENT k^* . INSTEAD, WE WILL HAVE TO RELY ON LEMMA 89 THAT

$$\mathbb{E}\|\mathbf{y}_{k^*} - \mathbf{y}_j\|^2 \leq \mathbb{E} \max_{i \in \tilde{\mathcal{G}}} \|\mathbf{y}_i - \mathbf{y}_j\|^2 \leq n \tilde{\rho}^2.$$

LEMMA 89 SHOWS THAT THIS INEQUALITY IS ESSENTIALLY TIGHT AND HENCE RELYING ON IT NECESSARILY INCURS AN EXTRA FACTOR OF n WHICH CAN BE VERY LARGE. INSTEAD, WE SHOW AN ALTERNATE ANALYSIS WHICH WORKS FOR A SMALLER BREAKDOWN POINT OF $\delta_{\max} = 1/4$, BUT *does not* INCUR THE EXTRA n FACTOR.

FOR ANY GOOD INPUT $i \in \tilde{\mathcal{G}}$, WE HAVE

$$\begin{aligned} \|\mathbf{y}_{k^*} - \bar{\mathbf{x}}\|^2 &\leq 2\|\mathbf{y}_{k^*} - \mathbf{y}_i\|^2 + 2\|\mathbf{y}_i - \bar{\mathbf{x}}\|^2 \\ \Rightarrow 2\|\mathbf{y}_{k^*} - \mathbf{y}_i\|^2 &\geq \|\mathbf{y}_{k^*} - \bar{\mathbf{x}}\|^2 - 2\|\mathbf{y}_i - \bar{\mathbf{x}}\|^2. \end{aligned}$$

¹This issue was incorrectly overlooked in the original analysis of Krum (Blanchard et al., 2017)

FURTHER, FOR A BAD WORKER $j \in \tilde{\mathcal{B}}$ WE CAN WRITE

$$2\|\mathbf{y}_{k^*} - \mathbf{y}_j\|^2 \geq \|\mathbf{y}_j - \bar{\mathbf{x}}\|^2 - 2\|\mathbf{y}_{k^*} - \bar{\mathbf{x}}\|^2.$$

COMBINING BOTH AND SUMMING OVER \mathcal{S}^* ,

$$\begin{aligned} \sum_{i \in \mathcal{S}^*} 2\|\mathbf{y}_{k^*} - \mathbf{y}_i\|^2 &= \sum_{i \in \tilde{\mathcal{G}} \cap \mathcal{S}^*} 2\|\mathbf{y}_{k^*} - \mathbf{y}_i\|^2 + \sum_{j \in \tilde{\mathcal{B}} \cap \mathcal{S}^*} 2\|\mathbf{y}_{k^*} - \mathbf{y}_j\|^2 \\ &\geq \sum_{j \in \tilde{\mathcal{B}} \cap \mathcal{S}^*} \|\mathbf{y}_j - \bar{\mathbf{x}}\|^2 - 2 \sum_{i \in \tilde{\mathcal{G}} \cap \mathcal{S}^*} \|\mathbf{y}_i - \bar{\mathbf{x}}\|^2 \\ &\quad + (|\tilde{\mathcal{G}} \cap \mathcal{S}^*| - 2|\tilde{\mathcal{B}} \cap \mathcal{S}^*|) \|\mathbf{y}_{k^*} - \bar{\mathbf{x}}\|^2. \end{aligned}$$

WE CAN REARRANGE THE ABOVE EQUATION AS

$$\begin{aligned} \|\mathbf{y}_{k^*} - \bar{\mathbf{x}}\|^2 &\leq \frac{1}{(|\tilde{\mathcal{G}} \cap \mathcal{S}^*| - 2|\tilde{\mathcal{B}} \cap \mathcal{S}^*|)} \left(\sum_{i \in \mathcal{S}^*} 2\|\mathbf{y}_{k^*} - \mathbf{y}_i\|^2 + \sum_{i \in \tilde{\mathcal{G}} \cap \mathcal{S}^*} 2\|\mathbf{y}_i - \bar{\mathbf{x}}\|^2 \right) \\ &\leq \frac{1}{(|\mathcal{S}^*| - 3|\tilde{\mathcal{B}}|)} \left(\sum_{i \in \mathcal{S}^*} 2\|\mathbf{y}_{k^*} - \mathbf{y}_i\|^2 + \sum_{i \in \tilde{\mathcal{G}} \cap \mathcal{S}^*} 2\|\mathbf{y}_i - \bar{\mathbf{x}}\|^2 \right) \\ &\leq \frac{1}{(|\mathcal{S}^*| - 3|\tilde{\mathcal{B}}|)} \left(2 \min_{k, \mathcal{S}} \sum_{i \in \mathcal{S}} \|\mathbf{y}_k - \mathbf{y}_i\|^2 + \sum_{i \in \tilde{\mathcal{G}}} 2\|\mathbf{y}_i - \bar{\mathbf{x}}\|^2 \right) \end{aligned}$$

TAKING EXPECTATION NOW ON BOTH SIDES YIELDS

$$\mathbb{E} \|\mathbf{y}_{k^*} - \bar{\mathbf{x}}\|^2 \leq \frac{3n\tilde{\rho}^2}{|\mathcal{S}^*| - 3|\tilde{\mathcal{B}}|}.$$

NOW, RECALL THAT WE USED A RESAMPLING VALUE OF $s = \delta_{\max}/\delta$ WHERE FOR KRM WE HAVE $\delta_{\max} = 1/4 - \nu$. THEN, THE NUMBER OF BYZANTINE WORKERS CAN BE BOUNDED AS $|\tilde{\mathcal{B}}| \leq n(1/4 - \delta)$. THIS GIVES THE FINAL RESULT THAT

$$\mathbb{E} \|\mathbf{y}_{k^*} - \bar{\mathbf{x}}\|^2 \leq \frac{3n\tilde{\rho}^2}{3n/4 - 3(n/4 - \nu n)} = \frac{\tilde{\rho}^2}{\nu} \leq \frac{1}{\nu(1/4 - \nu)} \delta \rho^2.$$

THUS, KRM WITH RESAMPLING INDEED SATISFIES DEFINITION G WITH $\delta_{\max} = (1/4 - \nu)$ AND $c = 1/(\nu(1/4 - \nu))$.

ROBUSTNESS OF GEOMETRIC MEDIAN. GEOMETRIC MEDIAN COMPUTES THE MINIMUM OF THE FOLLOWING OPTIMIZATION PROBLEM

$$\mathbf{y}^* = \operatorname{argmin}_{\mathbf{y}} \sum_{i \in [n]} \|\mathbf{y} - \mathbf{y}_i\|_2.$$

WE WILL ADAPT LEMMA 24 OF COHEN ET AL. (2016), WHICH ITSELF IS BASED ON (MINSKER

ET AL., 2015). FOR A GOOD WORKER $i \in \tilde{\mathcal{G}}$ AND BAD WORKER $j \in \tilde{\mathcal{B}}$:

$$\begin{aligned} \|\mathbf{y}^\star - \mathbf{y}_i\|_2 &\geq \|\mathbf{y}^\star - \bar{\mathbf{x}}\|_2 - \|\mathbf{y}_i - \bar{\mathbf{x}}\|_2 \text{ FOR } i \in \tilde{\mathcal{G}}, \text{ AND} \\ \|\mathbf{y}^\star - \mathbf{y}_j\|_2 &\geq \|\mathbf{y}_j - \bar{\mathbf{x}}\|_2 - \|\mathbf{y}^\star - \bar{\mathbf{x}}\|_2 \text{ FOR } j \in \tilde{\mathcal{B}}. \end{aligned}$$

SUMMING THIS OVER ALL WORKERS WE HAVE

$$\begin{aligned} \sum_{i \in [n]} \|\mathbf{y}^\star - \mathbf{y}_i\| &\geq (|\tilde{\mathcal{G}}| - |\tilde{\mathcal{B}}|) \|\mathbf{y}^\star - \bar{\mathbf{x}}\| + \sum_{j \in \tilde{\mathcal{B}}} \|\mathbf{y}_j - \bar{\mathbf{x}}\| - \sum_{i \in \tilde{\mathcal{G}}} \|\mathbf{y}_i - \bar{\mathbf{x}}\| \\ \Rightarrow \|\mathbf{y}^\star - \bar{\mathbf{x}}\| &\leq \frac{1}{(|\tilde{\mathcal{G}}| - |\tilde{\mathcal{B}}|)} \left(\sum_{i \in [n]} \|\mathbf{y}^\star - \mathbf{y}_i\| - \sum_{j \in \tilde{\mathcal{B}}} \|\mathbf{y}_j - \bar{\mathbf{x}}\| + \sum_{i \in \tilde{\mathcal{G}}} \|\mathbf{y}_i - \bar{\mathbf{x}}\| \right) \\ &= \frac{1}{(|\tilde{\mathcal{G}}| - |\tilde{\mathcal{B}}|)} \left(\min_{\mathbf{y}} \sum_{i \in [n]} \|\mathbf{y} - \mathbf{y}_i\| - \sum_{j \in \tilde{\mathcal{B}}} \|\mathbf{y}_j - \bar{\mathbf{x}}\| + \sum_{i \in \tilde{\mathcal{G}}} \|\mathbf{y}_i - \bar{\mathbf{x}}\| \right) \\ &\leq \frac{2}{(|\tilde{\mathcal{G}}| - |\tilde{\mathcal{B}}|)} \left(\sum_{i \in \tilde{\mathcal{G}}} \|\mathbf{y}_i - \bar{\mathbf{x}}\| \right). \end{aligned}$$

THE LAST STEP WE SUBSTITUTED $\mathbf{y} = \bar{\mathbf{x}}$. SQUARING BOTH SIDES, EXPANDING, AND THEN TAKING EXPECTATION GIVES

$$\begin{aligned} \mathbb{E} \|\mathbf{y}^\star - \bar{\mathbf{x}}\|^2 &\leq \frac{4}{(|\tilde{\mathcal{G}}| - |\tilde{\mathcal{B}}|)^2} \mathbb{E} \left(\sum_{i \in \tilde{\mathcal{G}}} \mathbb{E} \|\mathbf{y}_i - \bar{\mathbf{x}}\| \right)^2 \\ &\leq \frac{4}{(|\tilde{\mathcal{G}}| - |\tilde{\mathcal{B}}|)^2} \left(|\tilde{\mathcal{G}}| \sum_{i \in \tilde{\mathcal{G}}} \mathbb{E} \|\mathbf{y}_i - \bar{\mathbf{x}}\|^2 \right) \\ &\leq \frac{4|\tilde{\mathcal{G}}|^2}{(n - 2|\tilde{\mathcal{B}}|)^2} \bar{\rho}^2. \end{aligned}$$

NOW, RECALL THAT WE USED A RESAMPLING VALUE OF $s = \delta_{\max}/\delta$ WHERE FOR KRUM WE HAVE $\delta_{\max} = 1/2 - \nu$. THEN, THE NUMBER OF BYZANTINE WORKERS CAN BE BOUNDED AS $|\tilde{\mathcal{B}}| \leq n(1/2 - \nu)$. THIS GIVES THE FINAL RESULT THAT

$$\mathbb{E} \|\mathbf{y}^\star - \bar{\mathbf{x}}\|^2 \leq \frac{4n^2}{4n^2\nu^2} \bar{\rho}^2 \leq \frac{\bar{\rho}^2}{\nu^2} \leq \frac{1}{\nu(1/2 - \nu)} \delta \rho^2.$$

THUS, GEOMETRIC MEDIAN WITH RESAMPLING INDEED SATISFIES DEFINITION G WITH $\delta_{\max} = (1/2 - \nu)$ AND $c = 1/(\nu(1/2 - \nu))$. NOTE THAT GEOMETRIC MEDIAN HAS BETTER THEORETICAL PERFORMANCE THAN KRUM.

ROBUSTNESS OF COORDINATE-WISE MEDIAN. THE PROOF OF COORDINATE-WISE MEDIAN LARGELY FOLLOWS THAT OF THE GEOMETRIC MEDIAN. FIRST, WE NOTE THAT WE CAN SEPARATE OUT THE OBJECTIVE BY COORDINATES

$$\mathbb{E} \|\text{CM}(\mathbf{y}_1, \dots, \mathbf{y}_n) - \bar{\mathbf{x}}\|^2 = \sum_{l=1}^d \mathbb{E} (\text{CM}([y_1]_l, \dots, [y_n]_l) - [\bar{\mathbf{x}}]_l)^2.$$

THEN NOTE THAT, FOR ANY FIXED COORDINATE $l \in [d]$ AND FIXED GOOD WORKER $i \in \mathcal{G}$, WE HAVE $\mathbb{E}([y_i]_l - [\bar{x}]_l)^2 \leq \mathbb{E}\|y_i - \bar{x}\|^2 \leq \tilde{\rho}^2$. THUS, WE CAN SIMPLY ANALYZE COORDINATE-WISE MEDIAN AS d SEPARATE (GEOMETRIC) MEDIAN PROBLEMS ON SCALARS. THUS FOR ANY FIXED COORDINATE $l \in [d]$, WE HAVE

$$\mathbb{E}(\text{CM}([y_1]_l, \dots, [y_n]_l) - [\bar{x}]_l)^2 \leq \frac{\tilde{\rho}^2}{v^2} \Rightarrow \mathbb{E}\|\text{CM}(y_1, \dots, y_n) - \bar{x}\|^2 \leq \frac{d\tilde{\rho}^2}{v^2} \leq \frac{d}{v(1/2 - v)}\delta\rho^2.$$

THUS, COORDINATE-WISE MEDIAN WITH RESAMPLING INDEED SATISFIES DEFINITION G WITH $\delta_{\max} = (1/2 - v)$ AND $c = d/(v(1/2 - v))$.

14.4 Lower bounds on non-iid data (Proof of Theorem XXIII)

OUR PROOF BUILDS TWO SETS OF FUNCTIONS $\{f_i^1(x) \mid i \in \mathcal{G}^1\}$ AND $\{f_i^2(x) \mid i \in \mathcal{G}^2\}$ AND WILL SHOW THAT IN THE PRESENCE OF δ -FRACTION OF BYZANTINE WORKERS, NO ALGORITHM CAN DISTINGUISH BETWEEN THEM. SINCE THE PROBLEMS HAVE DIFFERENT OPTIMA, THIS MEANS THAT THE ALGORITHM NECESSARILY HAS AN ERROR ON AT LEAST ONE OF THEM.

FOR THE FIRST SET OF FUNCTIONS, LET THERE BE *no* BAD WORKERS AND HENCE $\mathcal{G}^1 = [n]$. THEN, WE DEFINE THE FOLLOWING FUNCTIONS FOR ANY $i \in [n]$:

$$f_i^1(x) = \begin{cases} \frac{\mu}{2}x^2 - G\hat{\delta}^{-1/2}x & \text{FOR } i \in \{1, \dots, \delta n\} \\ \frac{\mu}{2}x^2 & \text{FOR } i \in \{\delta n + 1, \dots, n\}. \end{cases}$$

DEFINING $G := G\delta^{1/2}$, THE AVERAGE FUNCTION WHICH IS OUR OBJECTIVE IS

$$f^1(x) = \frac{1}{n} \sum_{i=1}^n f_i^1(x) = \frac{\mu}{2}x^2 - Gx.$$

THE OPTIMUM OF OUR $f^1(x)$ IS AT $x = \frac{G}{\mu}$. NOTE THAT THE GRADIENT HETEROGENEITY AMONGST THESE WORKERS IS BOUNDED AS

$$\mathbb{E}_{i \sim [n]} \|\nabla f_i^1(x) - \nabla f^1(x)\|^2 \leq \mathbb{E}_{i \sim [n]} \|\nabla f_i^1(x) - \mu x\|^2 = \delta(G\delta^{-1/2})^2 = G^2.$$

NOW, WE DEFINE THE SECOND SET OF FUNCTIONS. HERE, SUPPOSE THAT WE HAVE δn BYZANTINE ATTACKERS WITH $\mathcal{B}^2 = \{1, \dots, \delta n\}$. THEN, THE FUNCTIONS OF THE GOOD WORKERS ARE DEFINED AS

$$f_i^2(x) = \frac{\mu}{2}x^2 \text{ FOR } i \in \mathcal{G}^2 = \{\delta n + 1, \dots, n\}.$$

WE THEN HAVE THAT THE SECOND AVERAGE OBJECTIVE IS

$$f^2(x) = \frac{1}{|\mathcal{G}^2|} \sum_{i \in \mathcal{G}^2} f_i^2(x) = \frac{\mu}{2}x^2.$$

HERE, WE HAVE GRADIENT HETEROGENEITY OF 0 AND HENCE IS SMALLER THAN G^2 . THE OPTI-

14.5. Convergence of robust optimization on non-iid data (Theorems XXII and XXIV)

MUM OF $f^2(x)$ IS AT $x = 0$. THE BYZANTINE ATTACKERS SIMPLY IMITATE AS IF THEY HAVE THE FUNCTIONS:

$$f_j^2(x) = \frac{\mu}{2}x^2 - G\delta^{-1/2}x \text{ FOR } j \in \mathcal{B}^2 = \{1, \dots, \delta n\}.$$

NOTE THAT THE SET OF FUNCTIONS, $\{f_1^1(\mathbf{x}), \dots, f_n^1(\mathbf{x})\}$ IS EXACTLY IDENTICAL TO THE SET $\{f_1^2(\mathbf{x}), \dots, f_n^2(\mathbf{x})\}$. ONLY THE IDENTITY OF THE GOOD WORKERS \mathcal{G}^1 AND \mathcal{G}^2 ARE DIFFERENT, LEADING TO DIFFERENT OBJECTIVE FUNCTIONS $f^1(x)$ AND $f^2(x)$. HOWEVER, SINCE THE ALGORITHM DOES NOT HAVE ACCESS TO \mathcal{G} , ITS OUTPUT ON EACH OF THEM IS IDENTICAL I.E.

$$\mathbf{x}^{\text{OUT}} = \text{ALG}(f_1^1(\mathbf{x}), \dots, f_n^1(\mathbf{x})) = \text{ALG}(f_1^2(\mathbf{x}), \dots, f_n^2(\mathbf{x})).$$

HOWEVER, THIS LEADS TO MAKING A LARGE ERROR IN LEAST ONE OF f^1 AND f^2 SINCE THEY HAVE DIFFERENT OPTIMUM. THIS PROVES A LOWER BOUND ERROR OF

$$\max_{k \in \{1,2\}} f^k(\mathbf{x}^{\text{OUT}}) - f^k(\mathbf{x}^*) \geq \mu \left(\frac{G}{2\mu} \right)^2 = \frac{\delta G^2}{4\mu}.$$

SIMILARLY, WE CAN ALSO BOUND THE GRADIENT NORM ERROR BOUND AS

$$\max_{k \in \{1,2\}} \|\nabla f^k(\mathbf{x}^{\text{OUT}})\|^2 \geq \mu^2 \left(\frac{G}{2\mu} \right)^2 = \frac{\delta G^2}{4}.$$

□

14.5 Convergence of robust optimization on non-iid data (Theorems XXII and XXIV)

WE WILL PROVE A MORE GENERAL CONVERGENCE THEOREM WHICH GENERALIZES THEOREMS XXII AND XXIV.

Theorem XXXVI. Suppose we are given a (δ_{\max}, c) -ARAGG satisfying Definition G, and n workers of which a subset \mathcal{G} of size at least $|\mathcal{G}| \geq n(1-\delta)$ faithfully follow the algorithm for $\delta \leq \delta_{\max}$. Further, for any good worker $i \in \mathcal{G}$ let f_i be a possibly non-convex function with L -Lipschitz gradients, and the stochastic gradients on each worker be independent, unbiased and satisfy

$$\mathbb{E}_{\xi_i} \|\mathbf{g}_i(\mathbf{x}) - \nabla f_i(\mathbf{x})\|^2 \leq \sigma^2 \text{ and } \mathbb{E}_{j \sim \mathcal{G}} \|\nabla f_j(\mathbf{x}) - \nabla f(\mathbf{x})\|^2 \leq G^2 + B^2 \|\nabla f(\mathbf{x})\|^2, \quad \forall \mathbf{x},$$

where $\delta \leq 1/(3cB^2)$. Then, for $F^0 := f(\mathbf{x}^0) - f^*$, the output of Algorithm 10 with step-size $\eta = \min\left(\mathcal{O}\left(\sqrt{\frac{LF^0 + c\delta(G^2 + \sigma^2)}{TL^2\sigma^2(n^{-1} + c\delta)}}\right), \frac{1}{8L}\right)$ and momentum parameter $\beta = (1 - 8L\eta)$ satisfies

$$\frac{1}{T} \sum_{t=1}^T \mathbb{E} \|\nabla f(\mathbf{x}^{t-1})\|^2 \leq \mathcal{O}\left(\frac{1}{1-3c\delta B^2} \cdot \left(c\delta G^2 + \sigma \sqrt{\frac{LF^0}{T}(c\delta + 1/n)} + \frac{LF^0}{T}\right)\right).$$

DEFINITIONS.

RECALL OUR ALGORITHM WHICH PERFORMS FOR $t \geq 2$ THE FOLLOWING UPDATE WITH $(1 - \beta) = \alpha$:

$$\begin{aligned} \mathbf{m}_i^t &= (1 - \alpha)\mathbf{m}_i^{t-1} + \alpha \mathbf{g}_i(\mathbf{x}^{t-1}) \quad \text{FOR EVERY } i \in \mathcal{G}, \\ \mathbf{x}^t &= \mathbf{x}^{t-1} - \eta \text{ARAGG}(\mathbf{m}_1^t, \dots, \mathbf{m}_n^t). \end{aligned}$$

FOR $t = 1$, WE USE $\alpha = 0$ I.E. $\mathbf{m}_i^1 = \mathbf{g}_i(\mathbf{x}^0)$. LET US ALSO DEFINE THE ACTUAL AND IDEAL MOMENTUM AGGREGATES AS

$$\mathbf{m}^t := \text{ARAGG}(\mathbf{m}_1^t, \dots, \mathbf{m}_n^t) \quad \text{AND} \quad \bar{\mathbf{m}}^t := \frac{1}{|\mathcal{G}|} \sum_{i \in \mathcal{G}} \mathbf{m}_i^t.$$

WE STATE SEVERAL SUPPORTING LEMMAS BEFORE PROVING OUR MAIN THEOREM XXXVI. WE WILL LOOSELY FOLLOW THE PROOF OF BYZANTINE ROBUSTNESS IN THE IID CASE BY KARIM-IREDDY ET AL. (2021B), WITH THE KEY DIFFERENCE OF LEMMA 90 WHICH ACCOUNTS FOR THE NON-IID ERROR.

Lemma 90 (Aggregation error). *Given that ARAGG satisfies Definition G holds, the error between the ideal average momentum $\bar{\mathbf{m}}^t$ and the output of the robust aggregation rule \mathbf{m}^t for any $t \geq 2$ can be bounded as*

$$\mathbb{E} \|\mathbf{m}^t - \bar{\mathbf{m}}^t\|^2 \leq c\delta\rho_t^2,$$

where we define for $t \geq 2$

$$\rho_t^2 := (6\alpha\sigma^2 + 3G^2) + (6\sigma^2 + 3G^2)(1 - \alpha)^t + 3 \sum_{k=1}^t (1 - \alpha)^{t-k} \alpha B^2 \mathbb{E} \|\nabla f(\mathbf{x}^{k-1})\|^2.$$

For $t = 1$ we can simplify the bound as $\rho_1^2 := 6c\delta\sigma^2 + 3c\delta G^2 + 3c\delta B^2 \|\nabla f(\mathbf{x}^0)\|^2$.

Proof. Expanding the definition of the worker momentum for a fixed good workers $i \in \mathcal{G}$,

$$\begin{aligned} \mathbb{E} \|\mathbf{m}_i^t - \mathbb{E}[\mathbf{m}_i^t | i]\|^2 &= \mathbb{E} \|\alpha(\mathbf{g}_i(\mathbf{x}^{t-1}) - \nabla f_i(\mathbf{x}^{t-1})) + (1 - \alpha)(\mathbf{m}_i^{t-1} - \mathbb{E}[\mathbf{m}_i^{t-1} | i])\|^2 \\ &\leq \mathbb{E} \|(1 - \alpha)(\mathbf{m}_i^{t-1} - \mathbb{E}[\mathbf{m}_i^{t-1} | i])\|^2 + \alpha^2 \sigma^2 \\ &\leq (1 - \alpha) \mathbb{E} \|\mathbf{m}_i^{t-1} - \mathbb{E}[\mathbf{m}_i^{t-1} | i]\|^2 + \alpha^2 \sigma^2. \end{aligned}$$

Unrolling the recursion above yields

$$\mathbb{E} \|\mathbf{m}_i^t - \mathbb{E}[\mathbf{m}_i^t | i]\|^2 \leq \left(\sum_{\ell=2}^t (1 - \alpha)^{t-\ell} \right) \alpha^2 \sigma^2 + (1 - \alpha)^{t-1} \sigma^2 \leq \sigma^2 (\alpha + (1 - \alpha)^{t-1}).$$

Similar computation also shows

$$\mathbb{E} \|\bar{\mathbf{m}}^t - \mathbb{E}[\bar{\mathbf{m}}^t]\|^2 \leq \frac{\sigma^2}{n} (\alpha + (1 - \alpha)^{t-1}).$$

14.5. Convergence of robust optimization on non-iid data (Theorems XXII and XXIV)

So far, the expectation was only over the stochasticity of the gradients of worker i . Note that we have $\mathbb{E}[\mathbf{m}_i^t | i] = (1 - \alpha)\nabla f_i(\mathbf{x}^{t-1}) + \alpha\mathbb{E}[\mathbf{m}_i^{t-1} | i]$. Now, suppose we sample i uniformly at random from \mathcal{G} . Then,

$$\begin{aligned}\mathbb{E}_i \|\mathbb{E}[\mathbf{m}_i^t | i] - \mathbb{E}[\bar{\mathbf{m}}^t]\|^2 &= \mathbb{E} \|\alpha(\nabla f_i(\mathbf{x}^{t-1}) - \nabla f(\mathbf{x}^{t-1})) + (1 - \alpha)(\mathbb{E}[\mathbf{m}_i^{t-1} | i] - \mathbb{E}[\bar{\mathbf{m}}^{t-1}])\|^2 \\ &\leq (1 - \alpha)\mathbb{E}_i \|\mathbb{E}[\mathbf{m}_i^{t-1} | i] - \mathbb{E}[\bar{\mathbf{m}}^{t-1}]\|^2 + \alpha\mathbb{E}_i \|\nabla f_i(\mathbf{x}^{t-1}) - \nabla f(\mathbf{x}^{t-1})\|^2 \\ &\leq (1 - \alpha)\mathbb{E}_i \|\mathbb{E}[\mathbf{m}_i^{t-1} | i] - \mathbb{E}[\bar{\mathbf{m}}^{t-1}]\|^2 + \alpha G^2 + \alpha B^2 \mathbb{E} \|\nabla f(\mathbf{x}^{t-1})\|^2\end{aligned}$$

Note that here we only get α instead of α^2 as before. This is because the randomness in the sampling of i $\nabla f_i(\mathbf{x}^{t-1})$ is not independent of the second term $\mathbb{E}[\mathbf{m}_i^{t-1} | i] - \mathbb{E}[\bar{\mathbf{m}}^{t-1}]$. Expanding this we get,

$$\mathbb{E}_i \|\mathbb{E}[\mathbf{m}_i^t | i] - \mathbb{E}[\bar{\mathbf{m}}^t]\|^2 \leq G^2(1 + (1 - \alpha)^{t-1}) + \sum_{k=1}^t (1 - \alpha)^{t-k} \alpha B^2 \mathbb{E} \|\nabla f(\mathbf{x}^{k-1})\|^2.$$

We can combine all three bounds above as

$$\begin{aligned}\mathbb{E}_i \|\mathbf{m}_i^t - \bar{\mathbf{m}}^t\|^2 &\leq 3\mathbb{E} \|\mathbf{m}_i^t - \mathbb{E}[\mathbf{m}_i^t | i]\|^2 + 3\mathbb{E} \|\bar{\mathbf{m}}^t - \mathbb{E}[\bar{\mathbf{m}}^t]\|^2 + 3\mathbb{E}_i \|\mathbb{E}[\mathbf{m}_i^t | i] - \mathbb{E}[\bar{\mathbf{m}}^t]\|^2 \\ &\leq (6\alpha\sigma^2 + 3G^2) + (6\sigma^2 + 3G^2)(1 - \alpha)^t + 3 \sum_{k=1}^t (1 - \alpha)^{t-k} \alpha B^2 \mathbb{E} \|\nabla f(\mathbf{x}^{k-1})\|^2.\end{aligned}$$

Recall that the right hand side was defined to be ρ_t^2 . Using Definition G, we can show that the output of the aggregation rule ARAGG satisfies the condition in the lemma. \square

ONE MAJOR CAVEAT IN THE ABOVE LEMMA IS THAT HERE ρ^2 CANNOT BE KNOWN TO THE ROBUST AGGREGATION SINCE IT INVOLVES $\mathbb{E} \|\nabla f(\mathbf{x}^{k-1})\|^2$ WHOSE VALUE WE DO NOT HAVE ACCESS TO. HOWEVER, THIS DOES NOT PRESENT A HURDLE TO *agnostic* AGGREGATION RULES WHICH ARE AUTOMATICALLY ADAPTIVE TO THE VALUE OF ρ^2 . DERIVING A SIMILARLY PROVABLE ADAPTIVE CLIPPING METHOD IS A VERY IMPORTANT OPEN PROBLEM.

Lemma 91 (Descent bound). *For any $\alpha \in [0, 1]$ for $t \geq 2$, $\eta \leq \frac{1}{L}$, and an L -smooth function f we have for any $t \geq 1$*

$$\mathbb{E}_t[f(\mathbf{x}_t)] \leq f(\mathbf{x}^{t-1}) - \frac{\eta}{2} \|\nabla f(\mathbf{x}^{t-1})\|^2 + \eta \mathbb{E}_t \|\bar{\mathbf{e}}_t\|^2 + \eta \mathbb{E}_t \|\mathbf{m}_t - \bar{\mathbf{m}}_t\|^2.$$

where $\bar{\mathbf{e}}_t := \bar{\mathbf{m}}_t - \nabla f(\mathbf{x}^{t-1})$.

Proof. By the smoothness of the function f and the server update,

$$\begin{aligned}
 f(\mathbf{x}_t) &\leq f(\mathbf{x}^{t-1}) - \eta \langle \nabla f(\mathbf{x}^{t-1}), \mathbf{m}_t \rangle + \frac{L\eta^2}{2} \|\mathbf{m}_t\|^2 \\
 &\leq f(\mathbf{x}^{t-1}) - \eta \langle \nabla f(\mathbf{x}^{t-1}), \mathbf{m}_t \rangle + \frac{\eta}{2} \|\mathbf{m}_t\|^2 \\
 &= f(\mathbf{x}^{t-1}) + \frac{\eta}{2} \|\mathbf{m}_t - \nabla f(\mathbf{x}^{t-1})\|^2 - \frac{\eta}{2} \|\nabla f(\mathbf{x}^{t-1})\|^2 \\
 &= f(\mathbf{x}^{t-1}) + \frac{\eta}{2} \|\mathbf{m}_t \pm \bar{\mathbf{m}}_t - \nabla f(\mathbf{x}^{t-1})\|^2 - \frac{\eta}{2} \|\nabla f(\mathbf{x}^{t-1})\|^2 \\
 &\leq f(\mathbf{x}^{t-1}) + \eta \|\bar{\mathbf{e}}_t\|^2 + \eta \|\mathbf{m}_t - \bar{\mathbf{m}}_t\|^2 - \frac{\eta}{2} \|\nabla f(\mathbf{x}^{t-1})\|^2.
 \end{aligned}$$

Here we used the identities that $-2ab = (a-b)^2 - a^2 - b^2$, and Young's inequality that $(a+b)^2 \leq (1+\gamma)a^2 + (1+\frac{1}{\gamma})b^2$ for any positive constant $\gamma \geq 0$ (here we used $\gamma = 1$). Taking conditional expectation on both sides yields the lemma. \square

Lemma 92 (Error bound). *Using any constant momentum and step-sizes such that $1 \geq \alpha \geq 8L\eta$ for $t \geq 2$, we have for an L -smooth function f that $\mathbb{E}\|\bar{\mathbf{e}}_1\|^2 \leq \frac{2\sigma^2}{n}$ and for $t \geq 2$*

$$\mathbb{E}\|\bar{\mathbf{e}}_t\|^2 \leq (1 - \frac{2\alpha}{5}) \mathbb{E}\|\bar{\mathbf{e}}^{t-1}\|^2 + \frac{\alpha}{10} \mathbb{E}\|\nabla f(\mathbf{x}_{t-2})\|^2 + \frac{\alpha}{10} \mathbb{E}\|\mathbf{m}^{t-1} - \bar{\mathbf{m}}^{t-1}\|^2 + \alpha^2 \frac{2\sigma^2}{n}.$$

Proof. Let us define $\bar{\mathbf{g}}(\mathbf{x}) := \frac{1}{|\mathcal{G}|} \sum_{i \in \mathcal{G}} \mathbf{g}_i(\mathbf{x})$. This implies that

$$\mathbb{E}\|\bar{\mathbf{g}}(\mathbf{x}) - \nabla f(\mathbf{x})\|^2 \leq \frac{\sigma^2}{|\mathcal{G}|} \leq \frac{2\sigma^2}{n}.$$

Then by definition of $\bar{\mathbf{m}}$, we can expand the error as:

$$\begin{aligned}
 \mathbb{E}\|\bar{\mathbf{e}}_t\|^2 &= \mathbb{E}\|\bar{\mathbf{m}}_t - \nabla f(\mathbf{x}^{t-1})\|^2 \\
 &= \mathbb{E}\|\alpha \bar{\mathbf{g}}(\mathbf{x}^{t-1}) + (1-\alpha) \bar{\mathbf{m}}^{t-1} - \nabla f(\mathbf{x}^{t-1})\|^2 \\
 &\leq \mathbb{E}\|\alpha \nabla f(\mathbf{x}^{t-1}) + (1-\alpha) \bar{\mathbf{m}}^{t-1} - \nabla f(\mathbf{x}^{t-1})\|^2 + \frac{2\alpha^2\sigma^2}{n} \\
 &= (1-\alpha)^2 \mathbb{E}\|(\bar{\mathbf{m}}^{t-1} - \nabla f(\mathbf{x}_{t-2})) + (\nabla f(\mathbf{x}_{t-2}) - \nabla f(\mathbf{x}^{t-1}))\|^2 + \frac{2\alpha^2\sigma^2}{n} \\
 &\leq (1-\alpha)(1 + \frac{\alpha}{2}) \mathbb{E}\|(\bar{\mathbf{m}}^{t-1} - \nabla f(\mathbf{x}_{t-2}))\|^2 \\
 &\quad + (1-\alpha)(1 + \frac{2}{\alpha}) \mathbb{E}\|\nabla f(\mathbf{x}_{t-2}) - \nabla f(\mathbf{x}^{t-1})\|^2 + \frac{2\alpha^2\sigma^2}{n} \\
 &\leq (1 - \frac{\alpha}{2}) \mathbb{E}\|\bar{\mathbf{e}}^{t-1}\|^2 + \frac{2L^2}{\alpha} \mathbb{E}\|\mathbf{x}_{t-2} - \mathbf{x}^{t-1}\|^2 + \frac{2\alpha^2\sigma^2}{n} \\
 &= (1 - \frac{\alpha}{2}) \mathbb{E}\|\bar{\mathbf{e}}^{t-1}\|^2 + \frac{2L^2\eta^2}{\alpha} \mathbb{E}\|\mathbf{m}^{t-1}\|^2 + \frac{2\alpha^2\sigma^2}{n} \\
 &\leq (1 - \frac{\alpha}{2}) \mathbb{E}\|\bar{\mathbf{e}}^{t-1}\|^2 + \frac{6L^2\eta^2}{\alpha} \mathbb{E}\|\bar{\mathbf{e}}^{t-1}\|^2 \\
 &\quad + \frac{6L^2\eta^2}{\alpha} \mathbb{E}\|\mathbf{m}^{t-1} - \bar{\mathbf{m}}^{t-1}\|^2 + \frac{6L^2\eta^2}{\alpha} \mathbb{E}\|\nabla f(\mathbf{x}_{t-2})\|^2 + \frac{2\alpha^2\sigma^2}{n}.
 \end{aligned}$$

14.5. Convergence of robust optimization on non-iid data (Theorems XXII and XXIV)

Our choice of the momentum parameter α implies $64L^2\eta^2 \leq \alpha^2$ and yields the lemma statement. □

PROOF OF THEOREM XXXVI. SCALE THE ERROR BOUND LEMMA 92 BY $\frac{5\eta}{2\alpha}$ AND ADD IT TO THE DESCENT BOUND LEMMA 91 TAKING EXPECTATIONS ON BOTH SIDES TO GET FOR $t \geq 2$

$$\begin{aligned} \mathbb{E}[f(\mathbf{x}_t)] + \frac{5\eta}{2\alpha} \mathbb{E}\|\bar{\mathbf{e}}_t\|^2 &\leq \mathbb{E}[f(\mathbf{x}^{t-1})] - \frac{\eta}{2} \mathbb{E}\|\nabla f(\mathbf{x}^{t-1})\|^2 + \eta \mathbb{E}\|\bar{\mathbf{e}}_t\|^2 + \eta \mathbb{E}\|\mathbf{m}_t - \bar{\mathbf{m}}_t\|^2 + \\ &\quad \frac{5\eta}{2\alpha} \mathbb{E}\|\bar{\mathbf{e}}^{t-1}\|^2 - \eta \mathbb{E}\|\bar{\mathbf{e}}^{t-1}\|^2 + \frac{\eta}{4} \mathbb{E}\|\nabla f(\mathbf{x}_{t-2})\|^2 \\ &\quad + \frac{\eta}{4} \mathbb{E}\|\mathbf{m}^{t-1} - \bar{\mathbf{m}}^{t-1}\|^2 + 5\eta\alpha \frac{\sigma^2}{n} \end{aligned}$$

NOW, LET USE THE AGGREGATION ERROR LEMMA 90 TO BOUND $\mathbb{E}\|\mathbf{m}^{t-1} - \bar{\mathbf{m}}^{t-1}\|^2$ IN THE ABOVE EXPRESSION TO GET

$$\begin{aligned} \mathbb{E}[f(\mathbf{x}_t)] + \frac{5\eta}{2\alpha} \mathbb{E}\|\bar{\mathbf{e}}_t\|^2 &\leq \mathbb{E}[f(\mathbf{x}^{t-1})] - \frac{\eta}{2} \mathbb{E}\|\nabla f(\mathbf{x}^{t-1})\|^2 + \eta \mathbb{E}\|\bar{\mathbf{e}}_t\|^2 + \eta \mathbb{E}\|\mathbf{m}_t - \bar{\mathbf{m}}_t\|^2 \\ &\quad + \frac{5\eta}{2\alpha} \mathbb{E}\|\bar{\mathbf{e}}^{t-1}\|^2 - \eta \mathbb{E}\|\bar{\mathbf{e}}^{t-1}\|^2 + \frac{\eta}{4} \mathbb{E}\|\nabla f(\mathbf{x}_{t-2})\|^2 + 5\eta\alpha \frac{\sigma^2}{n} \\ &\quad + \frac{\eta c \delta}{4} \left((6\alpha\sigma^2 + 3G^2) + (6\sigma^2 + 3G^2)(1-\alpha)^{t-2} \right. \\ &\quad \left. + 3 \sum_{k=1}^{t-1} (1-\alpha)^{t-1-k} \alpha B^2 \mathbb{E}\|\nabla f(\mathbf{x}^{k-1})\|^2 \right) \end{aligned}$$

LET US DEFINE $S_t := \sum_{k=1}^t (1-\alpha)^{t-k} \alpha B^2 \mathbb{E}\|\nabla f(\mathbf{x}^{k-1})\|^2$. THEN, S_t SATISFIES THE RECURSION:

$$\frac{1}{\alpha} S_t = \left(\frac{1}{\alpha} - 1\right) S_{t-1} + B^2 \mathbb{E}\|\nabla f(\mathbf{x}^{t-1})\|^2.$$

ADDING $\frac{3\eta c \delta}{4\alpha} S_t$ ON BOTH SIDES TO THE BOUND ABOVE AND REARRANGING GIVES THE FOLLOWING FOR $t \geq 2$

$$\begin{aligned}
 & \underbrace{\mathbb{E} f(\mathbf{x}_t) - f^\star + \left(\frac{5\eta}{2\alpha} - \eta\right) \mathbb{E} \|\bar{\mathbf{e}}_t\|^2 + \frac{\eta}{4} \mathbb{E} \|\nabla f(\mathbf{x}^{t-1})\|^2 + \frac{3\eta c \delta}{4\alpha} S_t}_{=:\mathcal{E}_t} \\
 & \leq \underbrace{\mathbb{E} f(\mathbf{x}^{t-1}) - f^\star + \left(\frac{5\eta}{2\alpha} - \eta\right) \mathbb{E} \|\bar{\mathbf{e}}^{t-1}\|^2 + \frac{\eta}{4} \mathbb{E} \|\nabla f(\mathbf{x}_{t-2})\|^2 + \frac{3\eta c \delta}{4\alpha} S_{t-1}}_{=:\mathcal{E}_{t-1}} \\
 & \quad + \left(-\frac{\eta}{4} + \frac{3\eta c \delta B^2}{4}\right) \mathbb{E} \|\nabla f(\mathbf{x}^{t-1})\|^2 \\
 & \quad + \frac{5\eta \alpha}{n} \sigma^2 + \frac{\eta c \delta}{4} \left((6\alpha \sigma^2 + 3G^2) + (6\sigma^2 + 3G^2)(1-\alpha)^{t-1}\right) \\
 & \leq \mathcal{E}_{t-1} - \frac{\eta}{4} (1 - 3c\delta B^2) \mathbb{E} \|\nabla f(\mathbf{x}^{t-1})\|^2 \\
 & \quad + \underbrace{5\eta \alpha \sigma^2 \left(\frac{1}{n} + \frac{3}{10} c\delta \left(1 + \frac{1}{\alpha} (1-\alpha)^{t-2}\right)\right) + \frac{3\eta c \delta G^2}{4} (1 + (1-\alpha)^{t-2})}_{=:\eta \xi_{t-1}^2}.
 \end{aligned}$$

FURTHER, SPECIALIZING THE DESCENT BOUND LEMMA 91 AND ERROR BOUND LEMMA 92 FOR $t = 1$ WE HAVE

$$\begin{aligned}
 \mathcal{E}_1 &= \mathbb{E} f(\mathbf{x}_1) - f^\star + \frac{3\eta}{2} \mathbb{E} \|\bar{\mathbf{e}}_1\|^2 + \frac{\eta}{4} \mathbb{E} \|\nabla f(\mathbf{x}_0)\|^2 + \frac{3\eta c \delta B^2}{4} \|\nabla f(\mathbf{x}^0)\|^2 \\
 &\leq f(\mathbf{x}_0) - f^\star + \frac{5\eta}{2} \mathbb{E} \|\bar{\mathbf{e}}_1\|^2 - \frac{\eta}{4} (1 - 3c\delta B^2) \mathbb{E} \|\nabla f(\mathbf{x}_0)\|^2 + \eta \mathbb{E} \|\mathbf{m}_1 - \bar{\mathbf{m}}_1\|^2 \\
 &\leq f(\mathbf{x}_0) - f^\star - \frac{\eta}{4} (1 - 3c\delta B^2) \mathbb{E} \|\nabla f(\mathbf{x}_0)\|^2 + \frac{5\eta \sigma^2}{n} + 3c\delta \eta (2\sigma^2 + G^2 + B^2 \|\nabla f(\mathbf{x}^0)\|^2) \\
 &= f(\mathbf{x}_0) - f^\star - \frac{\eta}{4} (1 - 3c\delta B^2) \mathbb{E} \|\nabla f(\mathbf{x}_0)\|^2 + \eta \xi_0^2.
 \end{aligned}$$

ABOVE, WE DEFINED $\xi_0^2 := \frac{5\sigma^2}{n} + 3c\delta (2\sigma^2 + G^2 + B^2 \|\nabla f(\mathbf{x}^0)\|^2)$. SUMMING OVER t FROM 2 UNTIL T , AGAIN REARRANGING OUR RECURSION FOR \mathcal{E}_t , AND ADDING $(1 - 3c\delta B^2) \mathbb{E} \|\nabla f(\mathbf{x}^0)\|^2$ ON

14.5. Convergence of robust optimization on non-iid data (Theorems XXII and XXIV)

BOTH SIDES GIVES

$$\begin{aligned}
(1 - 3c\delta B^2) \frac{1}{T} \sum_{t=1}^T \mathbb{E} \|\nabla f(\mathbf{x}^{t-1})\|^2 &\leq \frac{4\mathcal{E}_1}{\eta T} + (1 - 3c\delta B^2) \mathbb{E} \|\nabla f(\mathbf{x}^0)\|^2 + \frac{1}{T} \sum_{t=2}^T 4\xi_{t-1}^2 \\
&\leq \frac{4(f(\mathbf{x}_0) - f^*)}{\eta T} + \frac{1}{T} \sum_{t=1}^T 4\xi_{t-1}^2 \\
&= \frac{4(f(\mathbf{x}_0) - f^*)}{\eta T} + \frac{4\xi_0^2}{T} \\
&\quad + \frac{1}{T} \sum_{t=2}^T 20\alpha\sigma^2 \left(\frac{1}{n} + \frac{3}{10}c\delta(1 + \frac{1}{\alpha}(1 - \alpha)^{t-2}) \right) \\
&\quad + \frac{1}{T} \sum_{t=2}^T 3c\delta G^2(1 + (1 - \alpha)^{t-2}) \\
&\leq \frac{4(f(\mathbf{x}_0) - f^*)}{\eta T} + \frac{4\xi_0^2}{T} + 3c\delta G^2 + \frac{3c\delta G^2}{\alpha T} \\
&\quad + 20\alpha\sigma^2 \left(\frac{1}{n} + \frac{3}{10}c\delta \right) + \frac{6c\delta\sigma^2}{\alpha T} \\
&= \frac{4(f(\mathbf{x}_0) - f^*)}{\eta T} + \frac{3c\delta(G^2 + 2\sigma^2)}{\eta 8LT} + \eta 160L\sigma^2 \left(\frac{1}{n} + \frac{3}{10}c\delta \right) \\
&\quad + \frac{4\xi_0^2}{T} + 3c\delta G^2
\end{aligned}$$

THE LAST EQUALITY SUBSTITUTED THE VALUE OF $\alpha = 8L\eta$. NEXT, LET US USE THE APPROPRIATE STEP-SIZE OF

$$\eta = \min \left(\sqrt{\frac{4(f(\mathbf{x}_0) - f^*) + \frac{3c\delta}{8L}(G^2 + 2\sigma^2)}{T(160L\sigma^2)(\frac{1}{n} + \frac{3}{10}c\delta)}}, \frac{1}{8L} \right).$$

THIS GIVES THE FOLLOWING FINAL RATE OF CONVERGENCE:

$$\begin{aligned}
&\frac{1}{T} \sum_{t=1}^T \mathbb{E} \|\nabla f(\mathbf{x}^{t-1})\|^2 \\
&\leq \frac{1}{1 - 3c\delta B^2} \cdot \left(3c\delta G^2 + \sqrt{\frac{160L\sigma^2(\frac{1}{n} + \frac{3}{10}c\delta)}{T}} \cdot \sqrt{4(f(\mathbf{x}_0) - f^*) + \frac{3c\delta}{8L}(G^2 + 2\sigma^2)} \right. \\
&\quad + \frac{L(f(\mathbf{x}_0) - f^*)}{2T} + \frac{3c\delta(G^2 + 2\sigma^2)}{T} \\
&\quad \left. + \frac{\frac{20\sigma^2}{n} + 12c\delta(2\sigma^2 + G^2 + B^2 \|\nabla f(\mathbf{x}^0)\|^2)}{T} \right)
\end{aligned}$$

□

Bibliography

- DURMUS ALP EMRE ACAR, YUE ZHAO, RAMON MATAS NAVARRO, MATTHEW MATTINA, PAUL N WHATMOUGH, AND VENKATESH SALIGRAMA. FEDERATED LEARNING BASED ON DYNAMIC REGULARIZATION. IN *International Conference on Learning Representations*, 2021.
- ANISH ACHARYA, ABOLFAZL HASHEMI, PRATEEK JAIN, SUJAY SANGHAVI, INDERJIT S DHILLON, AND UFUK TOPCU. ROBUST TRAINING IN HIGH DIMENSIONS VIA BLOCK COORDINATE GEOMETRIC MEDIAN DESCENT. *arXiv preprint arXiv:2106.08882*, 2021.
- NAMAN AGARWAL, ANANDA THEERTHA SURESH, FELIX X. YU, SANJIV KUMAR, AND BRENDAN MCMAHAN. CPSGD: COMMUNICATION-EFFICIENT AND DIFFERENTIALLY-PRIVATE DISTRIBUTED SGD. IN *Proceedings of NeurIPS*, PAGES 7575–7586, 2018.
- SAURABH AGARWAL, HONGYI WANG, SHIVARAM VENKATARAMAN, AND DIMITRIS PAPAIIOPOULOS. ON THE UTILITY OF GRADIENT COMPRESSION IN DISTRIBUTED TRAINING SYSTEMS. *arXiv preprint arXiv:2103.00543*, 2021.
- OPEN AI. AI AND COMPUTE. <https://openai.com/blog/ai-and-compute/>, 2018.
- MARUAN AL-SHEDIVAT, JENNIFER GILLENWATER, ERIC XING, AND AFSHIN ROSTAMIZADEH. FEDERATED LEARNING VIA POSTERIOR AVERAGING: A NEW PERSPECTIVE AND PRACTICAL ALGORITHMS. *arXiv preprint arXiv:2010.05273*, 2020.
- DAN ALISTARH, DEMJAN GRUBIC, JERRY LI, RYOTA TOMIOKA, AND MILAN VOJNOVIC. QSGD: COMMUNICATION-EFFICIENT SGD VIA GRADIENT QUANTIZATION AND ENCODING. IN *Advances in Neural Information Processing Systems (NeurIPS)*, 2017.
- DAN ALISTARH, ZEYUAN ALLEN-ZHU, AND JERRY LI. BYZANTINE STOCHASTIC GRADIENT DESCENT. IN *NeurIPS - Advances in Neural Information Processing Systems*, PAGES 4613–4623, 2018.
- ZEYUAN ALLEN-ZHU, FAEZE EBRAHIMIAN, JERRY LI, AND DAN ALISTARH. BYZANTINE-RESILIENT NON-CONVEX STOCHASTIC GRADIENT DESCENT. IN *ICLR 2021 - International Conference of Learning Representations*, 2021.
- ROHAN ANIL, VINEET GUPTA, TOMER KOREN, AND YORAM SINGER. MEMORY-EFFICIENT ADAPTIVE OPTIMIZATION. *arXiv preprint arXiv:1901.11150*, 2019.

Bibliography

- PETER ARBENZ. LECTURE NOTES ON SOLVING LARGE SCALE EIGENVALUE PROBLEMS. *D-MATH, ETH Zürich*, 2, 2016.
- YOSSI ARJEVANI AND OHAD SHAMIR. COMMUNICATION COMPLEXITY OF DISTRIBUTED CONVEX LEARNING AND OPTIMIZATION. IN *Advances in neural information processing systems*, PAGES 1756–1764, 2015.
- YOSSI ARJEVANI, YAIR CARMON, JOHN C DUCHI, DYLAN J FOSTER, NATHAN SREBRO, AND BLAKE WOODWORTH. LOWER BOUNDS FOR NON-CONVEX STOCHASTIC OPTIMIZATION. *arXiv 1912.02365*, 2019.
- SANJEEV ARORA, RONG GE, BEHNAM NEYSHABUR, AND YI ZHANG. STRONGER GENERALIZATION BOUNDS FOR DEEP NETS VIA A COMPRESSION APPROACH. IN *International Conference on Machine Learning (ICML)*, 2018.
- DEVANSH ARPIT, STANISŁAW JASTRZEBSKI, NICOLAS BALLAS, DAVID KRUEGER, EMMANUEL BENGIO, MAXINDER S KANWAL, TEGAN MAHARAJ, ASJA FISCHER, AARON COURVILLE, YOSHUA BENGIO, ET AL. A CLOSER LOOK AT MEMORIZATION IN DEEP NETWORKS. IN *International Conference on Machine Learning (ICML)*, 2017.
- AMMAR AHMAD AWAN, CHING-HSIANG CHU, HARI SUBRAMONI, AND DHABALESWAR K PANDA. OPTIMIZED BROADCAST FOR DEEP LEARNING WORKLOADS ON DENSE-GPU INFINIBAND CLUSTERS: MPI OR NCCL? IN *European MPI Users’ Group Meeting (EuroMPI)*, 2018.
- ALEXEI BAEVSKI AND MICHAEL AULI. ADAPTIVE INPUT REPRESENTATIONS FOR NEURAL LANGUAGE MODELING. IN *International Conference on Learning Representations (ICLR)*, 2019.
- EUGENE BAGDASARYAN, ANDREAS VEIT, YIQING HUA, DEBORAH ESTRIN, AND VITALY SHMATIKOV. HOW TO BACKDOOR FEDERATED LEARNING. IN *AISTATS - Proceedings of the Twenty Third International Conference on Artificial Intelligence and Statistics*, VOLUME 108, PAGES 2938–2948, 2020.
- LUKAS BALLES AND PHILIPP HENNIG. DISSECTING ADAM: THE SIGN, MAGNITUDE AND VARIANCE OF STOCHASTIC GRADIENTS. IN *International Conference on Machine Learning (ICML)*, 2018.
- MORAN BARUCH, GILAD BARUCH, AND YOAV GOLDBERG. A LITTLE IS ENOUGH: CIRCUMVENTING DEFENSES FOR DISTRIBUTED LEARNING. *NeurIPS*, 2019.
- RAEF BASSILY, ADAM SMITH, AND ABHRADEEP THAKURTA. PRIVATE EMPIRICAL RISK MINIMIZATION: EFFICIENT ALGORITHMS AND TIGHT ERROR BOUNDS. IN *2014 IEEE 55th Annual Symposium on Foundations of Computer Science*, PAGES 464–473. IEEE, 2014.
- DEBRAJ BASU, DEEPESH DATA, CAN KARAKUS, AND SUHAS DIGGAVI. QSPARSE-LOCAL-SGD: DISTRIBUTED SGD WITH QUANTIZATION, SPARSIFICATION, AND LOCAL COMPUTATIONS. *arXiv preprint arXiv:1906.02367*, 2019.

- JEREMY BERNSTEIN, JIAWEI ZHAO, KAMYAR AZIZZADENESHELI, AND ANIMA ANANDKUMAR. SIGNSGD WITH MAJORITY VOTE IS COMMUNICATION EFFICIENT AND FAULT TOLERANT. *arXiv preprint arXiv:1810.05291*, 2018.
- JEREMY BERNSTEIN, JIAWEI ZHAO, KAMYAR AZIZZADENESHELI, AND ANIMA ANANDKUMAR. SIGNSGD WITH MAJORITY VOTE IS COMMUNICATION EFFICIENT AND FAULT TOLERANT. IN *International Conference on Learning Representations (ICLR)*, 2019.
- ARJUN NITIN BHAGOJI, SUPRIYO CHAKRABORTY, PRATEEK MITTAL, AND SERAPHIN CALO. ANALYZING FEDERATED LEARNING THROUGH AN ADVERSARIAL LENS, 2018.
- PEVA BLANCHARD, EL MAHDI EL MHAMDI, RACHID GUERRAOU, AND JULIEN STAINER. MACHINE LEARNING WITH ADVERSARIES: BYZANTINE TOLERANT GRADIENT DESCENT. IN *NeurIPS - Advances in Neural Information Processing Systems 30*, PAGES 119–129, 2017.
- KEITH BONAWITZ, VLADIMIR IVANOV, BEN KREUTER, ANTONIO MARCEDONE, H BRENDAN MCMAHAN, SARVAR PATEL, DANIEL RAMAGE, AARON SEGAL, AND KARN SETH. PRACTICAL SECURE AGGREGATION FOR PRIVACY-PRESERVING MACHINE LEARNING. IN *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*, PAGES 1175–1191. ACM, 2017.
- KEITH BONAWITZ, HUBERT EICHNER, WOLFGANG GRIESKAMP, DZMITRY HUBA, ALEX INGERMAN, VLADIMIR IVANOV, CHLOE KIDDON, JAKUB KONECNY, STEFANO MAZZOCCHI, H BRENDAN MCMAHAN, ET AL. TOWARDS FEDERATED LEARNING AT SCALE: SYSTEM DESIGN. IN *SysML - Proceedings of the 2nd SysML Conference, Palo Alto, CA, USA*, 2019.
- LÉON BOTTOU. LARGE-SCALE MACHINE LEARNING WITH STOCHASTIC GRADIENT DESCENT. IN *Proceedings of COMPSTAT’2010*, PAGES 177–186. SPRINGER, 2010.
- AMY BRAND, LIZ ALLEN, MICAH ALTMAN, MARJORIE HLAVA, AND JO SCOTT. BEYOND AUTHORSHIP: ATTRIBUTION, CONTRIBUTION, COLLABORATION, AND CREDIT. *Learned Publishing*, 28(2):151–155, 2015.
- THEODORA S. BRISIMI, RUIDI CHEN, THEOFANIE MELA, ALEX OLSHEVSKY, IOANNIS CH. PASCHALIDIS, AND WEI SHI. FEDERATED LEARNING OF PREDICTIVE MODELS FROM FEDERATED ELECTRONIC HEALTH RECORDS. *International journal of medical informatics*, 112: 59–67, 2018.
- TOM B BROWN, BENJAMIN MANN, NICK RYDER, MELANIE SUBBIAH, JARED KAPLAN, PRAFULLA DHARIWAL, ARVIND NEELAKANTAN, PRANAV SHYAM, GIRISH SASTRY, AMANDA ASKELL, ET AL. LANGUAGE MODELS ARE FEW-SHOT LEARNERS. *arXiv preprint arXiv:2005.14165*, 2020.
- DAVID CARLSON, VOLKAN CEVHER, AND LAWRENCE CARIN. STOCHASTIC SPECTRAL DESCENT FOR RESTRICTED BOLTZMANN MACHINES. IN *International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2015.

Bibliography

- SHICONG CEN, HUIHUAI ZHANG, YUEJIE CHI, WEI CHEN, AND TIE-YAN LIU. CONVERGENCE OF DISTRIBUTED STOCHASTIC VARIANCE REDUCED METHODS WITHOUT SAMPLING EXTRA DATA. *arXiv preprint arXiv:1905.12648*, 2019.
- ZACHARY CHARLES AND JAKUB KONEČNÝ. ON THE OUTSIZED IMPORTANCE OF LEARNING RATES IN LOCAL UPDATE METHODS. *arXiv preprint arXiv:2007.00878*, 2020.
- KAMALIKA CHAUDHURI, CLAIRE MONTELEONI, AND ANAND D SARWATE. DIFFERENTIALLY PRIVATE EMPIRICAL RISK MINIMIZATION. *Journal of Machine Learning Research*, 12(MAR): 1069–1109, 2011.
- FEI CHEN, MI LUO, ZHENHUA DONG, ZHENGUO LI, AND XIUQIANG HE. FEDERATED META-LEARNING WITH FAST CONVERGENCE AND EFFICIENT COMMUNICATION. *arXiv preprint arXiv:1802.07876*, 2018A.
- JIANMIN CHEN, XINGHAO PAN, RAJAT MONGA, SAMY BENGIO, AND RAFAL JOZEFOWICZ. RE-VISITING DISTRIBUTED SYNCHRONOUS SGD. *arXiv preprint arXiv:1604.00981*, 2016.
- JINGHUI CHEN AND QUANQUAN GU. PADAM: CLOSING THE GENERALIZATION GAP OF ADAPTIVE GRADIENT METHODS IN TRAINING DEEP NEURAL NETWORKS. IN *International Conference on Learning Representations (ICLR)*, 2019.
- LINGJIAO CHEN, HONGYI WANG, ZACHARY CHARLES, AND DIMITRIS PAPAILIOPOULOS. DRACO: BYZANTINE-RESILIENT DISTRIBUTED TRAINING VIA REDUNDANT GRADIENTS. *arXiv preprint arXiv:1803.09877*, 2018B.
- MINGQING CHEN, RAJIV MATHEWS, TOM OUYANG, AND FRANÇOISE BEAUFAYS. FEDERATED LEARNING OF OUT-OF-VOCABULARY WORDS. *arXiv preprint arXiv:1903.10635*, 2019A.
- MINGQING CHEN, ANANDA THEERTHA SURESH, RAJIV MATHEWS, ADELINE WONG, FRANÇOISE BEAUFAYS, CYRIL ALLAUZEN, AND MICHAEL RILEY. FEDERATED LEARNING OF N-GRAM LANGUAGE MODELS. IN *Proceedings of the 23rd Conference on Computational Natural Language Learning (CoNLL)*, 2019B.
- XIANGYI CHEN, TIANCONG CHEN, HAORAN SUN, ZHIWEI STEVEN WU, AND MINGYI HONG. DISTRIBUTED TRAINING WITH HETEROGENEOUS DATA: BRIDGING MEDIAN- AND MEAN-BASED ALGORITHMS. *arXiv 1906.01736*, 2019C.
- YUDONG CHEN, LILI SU, AND JIAMING XU. DISTRIBUTED STATISTICAL MACHINE LEARNING IN ADVERSARIAL SETTINGS. *Proceedings of the ACM on Measurement and Analysis of Computing Systems*, 1(2):1–25, DEC 2017. ISSN 2476-1249. DOI: 10.1145/3154503.
- TRISHUL M CHILIMBI, YUTAKA SUZUE, JOHNSON APACIBLE, AND KARTHIK KALYANARAMAN. PROJECT ADAM: BUILDING AN EFFICIENT AND SCALABLE DEEP LEARNING TRAINING SYSTEM. IN *OSDI*, VOLUME 14, PAGES 571–582, 2014.

- GREGORY COHEN, SAEED AFSHAR, JONATHAN TAPSON, AND ANDRE VAN SCHAIK. EMNIST: EXTENDING MNIST TO HANDWRITTEN LETTERS. IN *2017 International Joint Conference on Neural Networks (IJCNN)*, PAGES 2921–2926. IEEE, 2017.
- MICHAEL B COHEN, YIN TAT LEE, GARY MILLER, JAKUB PACHOCKI, AND AARON SIDFORD. GEOMETRIC MEDIAN IN NEARLY LINEAR TIME. IN *Proceedings of the forty-eighth annual ACM symposium on Theory of Computing*, PAGES 9–21, 2016.
- EDO COLLINS, SIAVASH ARJOMAND BIGDELI, AND SABINE SÜSTRUNK. DETECTING MEMORIZATION IN RELU NETWORKS. *arXiv*, ABS/1810.03372, 2018.
- CORINNA CORTES AND VLADIMIR VAPNIK. SUPPORT-VECTOR NETWORKS. *Machine learning*, 20(3):273–297, 1995.
- CORINNA CORTES, MEHRYAR MOHRI, AND ANDRÉS MUNOZ MEDINA. ADAPTATION BASED ON GENERALIZED DISCREPANCY. *The Journal of Machine Learning Research*, 20(1):1–30, 2019.
- COMMON CRAWL. COMMON CRAWL DATASET. <https://commoncrawl.org/>, 2020.
- ASHOK CUTKOSKY AND FRANCESCO ORABONA. MOMENTUM-BASED VARIANCE REDUCTION IN NON-CONVEX SGD. IN *Advances in Neural Information Processing Systems*, PAGES 15236–15245, 2019.
- GEORGIOS DAMASKINOS, EL MAHDI EL MHAMDI, RACHID GUERRAOU, ARSANY HANY ABDELMESSIH GUIRGUIS, AND SÉBASTIEN LOUIS ALEXANDRE ROUAULT. AGGREGATHOR: BYZANTINE MACHINE LEARNING VIA ROBUST GRADIENT AGGREGATION. *Conference on Systems and Machine Learning (SysML) 2019, Stanford, CA, USA*, PAGE 19, 2019.
- DEEPESH DATA AND SUHAS DIGGAVI. BYZANTINE-RESILIENT SGD IN HIGH DIMENSIONS ON HETEROGENEOUS DATA. *arXiv preprint arXiv:2005.07866*, 2020.
- DEEPESH DATA AND SUHAS DIGGAVI. BYZANTINE-RESILIENT HIGH-DIMENSIONAL SGD WITH LOCAL ITERATIONS ON HETEROGENEOUS DATA. IN *ICML 2021 - 37th International Conference on Machine Learning*, 2021.
- PYTORCH DDP. POWERSGD HOOK. [HTTPS://PYTORCH.ORG/DOCS/STABLE/DDP_COMM_HOOKS.HTML#POWERSGD-COMMUNICATION-HOOK](https://pytorch.org/docs/stable/DDP_COMM_HOOKS.HTML#POWERSGD-COMMUNICATION-HOOK). ACCESSED: 2021-07-21.
- JEFFREY DEAN, GREG CORRADO, RAJAT MONGA, KAI CHEN, MATTHIEU DEVIN, MARK MAO, MARC’AURELIO RANZATO, ANDREW SENIOR, PAUL TUCKER, KE YANG, QUOC V. LE, AND ANDREW Y. NG. LARGE SCALE DISTRIBUTED DEEP NETWORKS. IN *Advances in neural information processing systems*, PAGES 1223–1231, 2012.
- AARON DEFAZIO AND LÉON BOTTOU. ON THE INEFFECTIVENESS OF VARIANCE REDUCED OPTIMIZATION FOR DEEP LEARNING. IN *Advances in Neural Information Processing Systems*, PAGES 1753–1763, 2019.

Bibliography

- AARON DEFAZIO, FRANCIS BACH, AND SIMON LACOSTE-JULIEN. SAGA: A FAST INCREMENTAL GRADIENT METHOD WITH SUPPORT FOR NON-STRONGLY CONVEX COMPOSITE OBJECTIVES. IN *Advances in neural information processing systems*, PAGES 1646–1654, 2014.
- ILIAS DIAKONIKOLAS, GAUTAM KAMATH, DANIEL M. KANE, JERRY LI, JACOB STEINHARDT, AND ALISTAIR STEWART. SEVER: A ROBUST META-ALGORITHM FOR STOCHASTIC OPTIMIZATION. *arXiv preprint arXiv:1803.02815*, 2018.
- LAURENT DINH, RAZVAN PASCANU, SAMY BENGIO, AND YOSHUA BENGIO. SHARP MINIMA CAN GENERALIZE FOR DEEP NETS. *arXiv preprint arXiv:1703.04933*, 2017.
- YANJIE DONG, GEORGIOS B. GIANNAKIS, TIANYI CHEN, JULIAN CHENG, MD. JAHANGIR HOS-SAIN, AND VICTOR C. M. LEUNG. COMMUNICATION-EFFICIENT ROBUST FEDERATED LEARN-ING OVER HETEROGENEOUS DATASETS. *arXiv 2006.09992*, 2020.
- JOHN DUCHI, ELAD HAZAN, AND YORAM SINGER. ADAPTIVE SUBGRADIENT METHODS FOR ONLINE LEARNING AND STOCHASTIC OPTIMIZATION. *Journal of Machine Learning Research*, 12(JUL):2121–2159, 2011.
- CELESTINE DÜNNER, AURELIEN LUCCHI, MATILDE GARGIANI, AN BIAN, THOMAS HOFMANN, AND MARTIN JAGGI. A DISTRIBUTED SECOND-ORDER ALGORITHM YOU CAN TRUST. IN *Inter-national Conference on Machine Learning*, PAGES 1358–1366. PMLR, 2018.
- EL-MAHDI EL-MHAMDI AND RACHID GUERRAOUI. FAST AND SECURE DISTRIBUTED LEARNING IN HIGH DIMENSION. *arXiv 1905.04374*, 2019.
- EL-MAHDI EL-MHAMDI, RACHID GUERRAOUI, ARSANY GUIRGUIS, LÊ NGUYỄN HOANG, AND SÉBASTIEN ROUAULT. COLLABORATIVE LEARNING AS AN AGREEMENT PROBLEM. *arXiv 2008.00742*, 2020.
- ANDRE ESTEVA, BRETT KUPREL, ROBERTO A NOVOA, JUSTIN KO, SUSAN M SWETTER, HE-LEN M BLAU, AND SEBASTIAN THRUN. DERMATOLOGIST-LEVEL CLASSIFICATION OF SKIN CANCER WITH DEEP NEURAL NETWORKS. *nature*, 542(7639):115–118, 2017.
- CONG FANG, CHRIS JUNCHI LI, ZHOUCHE LIN, AND TONG ZHANG. SPIDER: NEAR-OPTIMAL NON-CONVEX OPTIMIZATION VIA STOCHASTIC PATH-INTEGRATED DIFFERENTIAL ESTIMA-TOR. IN *Advances in Neural Information Processing Systems*, PAGES 689–699, 2018.
- JIASHI FENG, HUAN XU, AND SHIE MANNOR. DISTRIBUTED ROBUST LEARNING. *arXiv preprint arXiv:1409.5937*, 2014.
- DYLAN J FOSTER, AYUSH SEKHARI, OHAD SHAMIR, NATHAN SREBRO, KARTHIK SRIDHARAN, AND BLAKE WOODWORTH. THE COMPLEXITY OF MAKING THE GRADIENT SMALL IN STOCHAS-TIC CONVEX OPTIMIZATION. IN *Conference on Learning Theory*, PAGES 1319–1345. PMLR, 2019.
- SHUHAO FU, CHULIN XIE, BO LI, AND QIFENG CHEN. ATTACK-RESISTANT FEDERATED LEARN-ING WITH RESIDUAL-BASED REWEIGHTING. *arXiv 1912.11464*, 2019.

- ROBIN C GEYER, TASSILO KLEIN, AND MOIN NABI. DIFFERENTIALLY PRIVATE FEDERATED LEARNING: A CLIENT LEVEL PERSPECTIVE. *arXiv preprint arXiv:1712.07557*, 2017.
- SAEED GHADIMI AND GUANGHUI LAN. STOCHASTIC FIRST-AND ZERO-ORDER METHODS FOR NONCONVEX STOCHASTIC PROGRAMMING. *SIAM Journal on Optimization*, 23(4):2341–2368, 2013.
- SAEED GHADIMI AND GUANGHUI LAN. ACCELERATED GRADIENT METHODS FOR NONCONVEX NONLINEAR AND STOCHASTIC PROGRAMMING. *Mathematical Programming*, 156(1-2):59–99, 2016.
- AVISHEK GHOSH, JUSTIN HONG, DONG YIN, AND KANNAN RAMCHANDRAN. ROBUST FEDERATED LEARNING IN A HETEROGENEOUS ENVIRONMENT. *arXiv preprint arXiv:1906.06629*, 2019.
- PAUL GLASSERMAN. *Monte Carlo methods in financial engineering*, VOLUME 53. SPRINGER SCIENCE & BUSINESS MEDIA, 2013.
- EDUARD GORBUNOV, MARINA DANILOVA, AND ALEXANDER GASNIKOV. STOCHASTIC OPTIMIZATION WITH HEAVY-TAILED NOISE VIA ACCELERATED GRADIENT CLIPPING. *NeurIPS - Advances in Neural Information Processing Systems*, 2020.
- PRIYA GOYAL, PIOTR DOLLÁR, ROSS GIRSHICK, PIETER NOORDHUIS, LUKASZ WESOŁOWSKI, AAPO KYROLA, ANDREW TULLOCH, YANGQING JIA, AND KAIMING HE. ACCURATE, LARGE MINIBATCH SGD: TRAINING IMAGENET IN 1 HOUR. *arXiv 1706.02677*, 2017.
- SYLVAIN GUGGER AND JEREMY HOWARD. ADAMW AND SUPER-CONVERGENCE IS NOW THE FASTEST WAY TO TRAIN NEURAL NETS. [HTTPS://WWW.FAST.AI/2018/07/02/ADAM-WEIGHT-DECAY/](https://www.fast.ai/2018/07/02/adam-weight-decay/), 2018. ACCESSED: 2019-01-17.
- SURIYA GUNASEKAR, JASON LEE, DANIEL SOUDRY, AND NATHAN SREBRO. CHARACTERIZING IMPLICIT BIAS IN TERMS OF OPTIMIZATION GEOMETRY. IN *International Conference on Machine Learning (ICML)*, 2018.
- NIRUPAM GUPTA AND NITIN H. VAIDYA. RANDOMIZED REACTIVE REDUNDANCY FOR BYZANTINE FAULT-TOLERANCE IN PARALLELIZED LEARNING. *arXiv 1912.09528*, 2019.
- FARZIN HADDADPOUR AND MEHRDAD MAHDAVI. ON THE CONVERGENCE OF LOCAL DESCENT METHODS IN FEDERATED LEARNING. *arXiv preprint arXiv:1910.14425*, 2019.
- FILIP HANZELY AND PETER RICHTÁRIK. ONE METHOD TO RULE THEM ALL: VARIANCE REDUCTION FOR DATA, PARAMETERS AND MANY NEW METHODS. *arXiv preprint arXiv:1905.11266*, 2019.
- ANDREW HARD, KANISHKA RAO, RAJIV MATHEWS, FRANÇOISE BEAUFAYS, SEAN AUGENSTEIN, HUBERT EICHNER, CHLOÉ KIDDON, AND DANIEL RAMAGE. FEDERATED LEARNING FOR MOBILE KEYBOARD PREDICTION. *arXiv preprint arXiv:1811.03604*, 2018.

Bibliography

- KAIMING HE, XIANGYU ZHANG, SHAOQING REN, AND JIAN SUN. DEEP RESIDUAL LEARNING FOR IMAGE RECOGNITION. IN *Proceedings of the IEEE conference on computer vision and pattern recognition*, PAGES 770–778, 2016A.
- KAIMING HE, XIANGYU ZHANG, SHAOQING REN, AND JIAN SUN. IDENTITY MAPPINGS IN DEEP RESIDUAL NETWORKS. IN *European Conference on Computer Vision (ECCV)*, PAGES 630–645. SPRINGER, 2016B.
- LIE HE, SAI PRANEETH KARIMIREDDY, AND MARTIN JAGGI. SECURE BYZANTINE-ROBUST MACHINE LEARNING. *arXiv 2006.04747*, 2020.
- ELAD HOFFER, ITAY HUBARA, AND DANIEL SOUDRY. TRAIN LONGER, GENERALIZE BETTER: CLOSING THE GENERALIZATION GAP IN LARGE BATCH TRAINING OF NEURAL NETWORKS. IN *Advances in Neural Information Processing Systems*, PAGES 1731–1741, 2017.
- ARNE HOLST. SMARTPHONE USERS WORLDWIDE 2016-2021. *Statista* <https://web.archive.org/web/20210608080335/https://www.statista.com/statistics/330695/number-of-smartphone-users-worldwide/>, 2019.
- KEVIN HSIEH, AMAR PHANISHAYEE, ONUR MUTLU, AND PHILLIP B GIBBONS. THE NON-IID DATA QUAGMIRE OF DECENTRALIZED MACHINE LEARNING. *arXiv preprint arXiv:1910.00189*, 2019.
- TZU-MING HARRY HSU, HANG QI, AND MATTHEW BROWN. MEASURING THE EFFECTS OF NON-IDENTICAL DATA DISTRIBUTION FOR FEDERATED VISUAL CLASSIFICATION. *arXiv preprint arXiv:1909.06335*, 2019.
- MIA HUBERT, PETER J ROUSSEEUW, AND STEFAN VAN AELST. HIGH-BREAKDOWN ROBUST MULTIVARIATE METHODS. *Statistical science*, PAGES 92–119, 2008.
- FORREST N. IANDOLA, MATTHEW W MOSKEWICZ, KHALID ASHRAF, AND KURT KEUTZER. FIRECAFFE: NEAR-LINEAR ACCELERATION OF DEEP NEURAL NETWORK TRAINING ON COMPUTE CLUSTERS. IN *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, PAGES 2592–2600, 2016.
- ICO. GUIDE TO UK GDPR. <https://ico.org.uk/for-organisations/guide-to-data-protection/guide-to-the-general-data-protection-regulation-gdpr/principles/data-minimisation/>, 2021.
- DANIEL JIWOONG IM, MICHAEL TAO, AND KRISTIN BRANSON. AN EMPIRICAL ANALYSIS OF THE OPTIMIZATION OF DEEP NETWORK LOSS SURFACES. *arXiv preprint arXiv:1612.04010*, 2016.
- YIHAN JIANG, JAKUB KONEČNÝ, KEITH RUSH, AND SREERAM KANNAN. IMPROVING FEDERATED LEARNING PERSONALIZATION VIA MODEL AGNOSTIC META LEARNING. *arXiv preprint arXiv:1909.12488*, 2019.

- RICHENG JIN, YUFAN HUANG, XIAOFAN HE, TIANFU WU, AND HUAIYU DAI. STOCHASTIC-SIGN SGD FOR FEDERATED LEARNING WITH THEORETICAL GUARANTEES. *arXiv 2002.10940*, 2020.
- RIE JOHNSON AND TONG ZHANG. ACCELERATING STOCHASTIC GRADIENT DESCENT USING PREDICTIVE VARIANCE REDUCTION. IN *Advances in neural information processing systems*, PAGES 315–323, 2013.
- PETER KAIROUZ, H. BRENDAN MCMAHAN, BRENDAN AVENT, AURÉLIEN BELLET, MEHDI BENNIS, ARJUN NITIN BHAGOJI, KEITH BONAWITZ, ZACHARY CHARLES, GRAHAM CORMODE, RACHEL CUMMINGS, RAFAEL G. L. D’OLIVEIRA, SALIM EL ROUAYHEB, DAVID EVANS, JOSH GARDNER, ZACHARY GARRETT, ADRIÀ GASCÓN, BADIH GHAZI, PHILLIP B. GIBBONS, MARCO GRUTESER, ZAÏD HARCHAOU, CHAOYANG HE, LIE HE, ZHOUYUAN HUO, BEN HUTCHINSON, JUSTIN HSU, MARTIN JAGGI, TARA JAVIDI, GAURI JOSHI, MIKHAIL KHODAK, JAKUB KONECNY, ALEKSANDRA KOROLOVA, FARINAZ KOUSHANFAR, SANMI KOYEJO, TANCÈRE LEPOINT, YANG LIU, PRATEEK MITTAL, MEHRYAR MOHRI, RICHARD NOCK, AYFER ÖZGÜR, RASMUS PAGH, MARIANA RAYKOVA, HANG QI, DANIEL RAMAGE, RAMESH RASKAR, DAWN SONG, WEIKANG SONG, SEBASTIAN U. STICH, ZITENG SUN, ANANDA THEERTHA SURESH, FLORIAN TRAMÈR, PRANEETH VEPAKOMMA, JIANYU WANG, LI XIONG, ZHENG XU, QIANG YANG, FELIX X. YU, HAN YU, AND SEN ZHAO. ADVANCES AND OPEN PROBLEMS IN FEDERATED LEARNING. *arXiv preprint arXiv:1912.04977*, 2019.
- JARED KAPLAN, SAM MCCANDLISH, TOM HENIGHAN, TOM B BROWN, BENJAMIN CHESSE, REWON CHILD, SCOTT GRAY, ALEC RADFORD, JEFFREY WU, AND DARIO AMODEI. SCALING LAWS FOR NEURAL LANGUAGE MODELS. *arXiv preprint arXiv:2001.08361*, 2020.
- HAMED KARIMI, JULIE NUTINI, AND MARK SCHMIDT. LINEAR CONVERGENCE OF GRADIENT AND PROXIMAL-GRADIENT METHODS UNDER THE POLYAK-ŁOJASIEWICZ CONDITION. IN *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*, PAGES 795–811. SPRINGER, 2016.
- SAI PRANEETH KARIMIREDDY, SEBASTIAN STICH, AND MARTIN JAGGI. OPTIMAL INEXACT ACCELERATED GRADIENT METHODS. IN *ISMP - International Symposium on Mathematical Programming*, 2018A.
- SAI PRANEETH KARIMIREDDY, SEBASTIAN U STICH, AND MARTIN JAGGI. GLOBAL LINEAR CONVERGENCE OF NEWTON’S METHOD WITHOUT STRONG-CONVEXITY OR LIPSCHITZ GRADIENTS. *arXiv preprint arXiv:1806.00413*, 2018B.
- SAI PRANEETH KARIMIREDDY, QUENTIN REBJOCK, SEBASTIAN U STICH, AND MARTIN JAGGI. ERROR FEEDBACK FIXES SIGNSGD AND OTHER GRADIENT COMPRESSION SCHEMES. IN *ICML 2019 - Proceedings of the 36th International Conference on Machine Learning*, 2019.
- SAI PRANEETH KARIMIREDDY, MARTIN JAGGI, SATYEN KALE, MEHRYAR MOHRI, SASHANK J REDDI, SEBASTIAN U STICH, AND ANANDA THEERTHA SURESH. MIME: MIMICK-

Bibliography

- ING CENTRALIZED STOCHASTIC ALGORITHMS IN FEDERATED LEARNING. *arXiv preprint arXiv:2008.03606*, 2020A.
- SAI PRANEETH KARIMIREDDY, SATYEN KALE, MEHRYAR MOHRI, SASHANK REDDI, SEBASTIAN STICH, AND ANANDA THEERTHA SURESH. SCAFFOLD: STOCHASTIC CONTROLLED AVERAGING FOR FEDERATED LEARNING. IN *ICML 2020 - International Conference on Machine Learning*, 2020B.
- SAI PRANEETH KARIMIREDDY, LIE HE, , AND MARTIN JAGGI. BYZANTINE-ROBUST LEARNING ON HETEROGENEOUS DATASETS VIA RESAMPLING. *arXiv 2006.09365*, 2021A.
- SAI PRANEETH KARIMIREDDY, LIE HE, AND MARTIN JAGGI. LEARNING FROM HISTORY FOR BYZANTINE ROBUST OPTIMIZATION. IN *ICML 2021 - 37th International Conference on Machine Learning*, 2021B.
- SAI PRANEETH REDDY KARIMIREDDY, SEBASTIAN STICH, AND MARTIN JAGGI. ADAPTIVE BALANCING OF GRADIENT AND UPDATE COMPUTATION TIMES USING GLOBAL GEOMETRY AND APPROXIMATE SUBPROBLEMS. IN *International Conference on Artificial Intelligence and Statistics*, PAGES 1204–1213. PMLR, 2018C.
- KENJI KAWAGUCHI, LESLIE PACK KAEHLING, AND YOSHUA BENGIO. GENERALIZATION IN DEEP LEARNING. *arXiv preprint arXiv:1710.05468*, 2017.
- AHMED KHALED, KONSTANTIN MISHCHENKO, AND PETER RICHTÁRIK. TIGHTER THEORY FOR LOCAL SGD ON IDENTICAL AND HETEROGENEOUS DATA. IN *AISTATS 2020 - International Conference on Artificial Intelligence and Statistics*, 2020.
- DANIEL KIFER, ADAM SMITH, AND ABHRADEEP THAKURTA. PRIVATE CONVEX EMPIRICAL RISK MINIMIZATION AND HIGH-DIMENSIONAL REGRESSION. IN *Conference on Learning Theory*, PAGES 25–1, 2012.
- DIEDERIK P KINGMA AND JIMMY BA. ADAM: A METHOD FOR STOCHASTIC OPTIMIZATION. *arXiv preprint arXiv:1412.6980*, 2014.
- ANASTASIA KOLOSKOVA, NICOLAS LOIZOU, SADRA BOREIRI, MARTIN JAGGI, AND SEBASTIAN U STICH. A UNIFIED THEORY OF DECENTRALIZED SGD WITH CHANGING TOPOLOGY AND LOCAL UPDATES. IN *37th International Conference on Machine Learning (ICML)*, 2020.
- JAKUB KONEČNÝ, H BRENDAN MCMAHAN, FELIX X YU, PETER RICHTÁRIK, ANANDA THEERTHA SURESH, AND DAVE BACON. FEDERATED LEARNING: STRATEGIES FOR IMPROVING COMMUNICATION EFFICIENCY. *arXiv*, abs/1610.05492, 2016.
- KONSTANTINOS KONSTANTINIDIS AND ADITYA RAMAMOORTHY. BYZSHIELD: AN EFFICIENT AND ROBUST SYSTEM FOR DISTRIBUTED TRAINING. *arXiv 2010.04902*, 2020.
- ALEX KRIZHEVSKY, GEOFFREY HINTON, ET AL. LEARNING MULTIPLE LAYERS OF FEATURES FROM TINY IMAGES. 2009.

- ALEX KRIZHEVSKY, ILYA SUTSKEVER, AND GEOFFREY E HINTON. IMAGENET CLASSIFICATION WITH DEEP CONVOLUTIONAL NEURAL NETWORKS. IN *Advances in Neural Information Processing Systems (NIPS)*, PAGES 1097–1105, 2012.
- ANDREI KULUNCHAKOV AND JULIEN MAIRAL. ESTIMATE SEQUENCES FOR STOCHASTIC COMPOSITE OPTIMIZATION: VARIANCE REDUCTION, ACCELERATION, AND ROBUSTNESS TO NOISE. *arXiv preprint arXiv:1901.08788*, 2019.
- KEVIN A LAI, ANUP B RAO, AND SANTOSH VEMPALA. AGNOSTIC ESTIMATION OF MEAN AND COVARIANCE. IN *2016 IEEE 57th Annual Symposium on Foundations of Computer Science (FOCS)*, PAGES 665–674. IEEE, 2016.
- LESLIE LAMPORT, ROBERT SHOSTAK, AND MARSHALL PEASE. THE BYZANTINE GENERALS PROBLEM. IN *Concurrency: the Works of Leslie Lamport*, PAGES 203–226. 2019.
- YANN LECUN, LÉON BOTTOU, YOSHUA BENGIO, AND PATRICK HAFFNER. GRADIENT-BASED LEARNING APPLIED TO DOCUMENT RECOGNITION. *Proceedings of the IEEE*, 86(11):2278–2324, 1998.
- YANN LECUN, YOSHUA BENGIO, AND GEOFFREY HINTON. DEEP LEARNING. *nature*, 521(7553):436, 2015.
- JASON D LEE, QIHANG LIN, TENG YU MA, AND TIANBAO YANG. DISTRIBUTED STOCHASTIC VARIANCE REDUCED GRADIENT METHODS AND A LOWER BOUND FOR COMMUNICATION COMPLEXITY. *arXiv preprint arXiv:1507.07595*, 2015.
- LIHUA LEI AND MICHAEL JORDAN. LESS THAN A SINGLE PASS: STOCHASTICALLY CONTROLLED STOCHASTIC GRADIENT. IN *AISTATS*, PAGES 148–156, 2017.
- HAO LI, ZHENG XU, GAVIN TAYLOR, CHRISTOPH STUDER, AND TOM GOLDSTEIN. VISUALIZING THE LOSS LANDSCAPE OF NEURAL NETS. IN *Advances in Neural Information Processing Systems (NeurIPS)*, 2018A.
- LIPING LI, WEI XU, TIANYI CHEN, GEORGIOS B GIANNAKIS, AND QING LING. RSA: BYZANTINE-ROBUST STOCHASTIC AGGREGATION METHODS FOR DISTRIBUTED LEARNING FROM HETEROGENEOUS DATASETS. IN *Proceedings of the AAAI Conference on Artificial Intelligence*, VOLUME 33, PAGES 1544–1551, 2019A.
- MU LI, DAVID G ANDERSEN, JUN WOO PARK, ALEXANDER J SMOLA, AMR AHMED, VANJA JOSIFOVSKI, JAMES LONG, EUGENE J SHEKITA, AND BOR-YIING SU. SCALING DISTRIBUTED MACHINE LEARNING WITH THE PARAMETER SERVER. IN *OSDI*, VOLUME 14, PAGES 583–598, 2014.
- TIAN LI, ANIT KUMAR SAHU, MAZIAR SANJABI, MANZIL ZAHEER, AMEET TALWALKAR, AND VIRGINIA SMITH. ON THE CONVERGENCE OF FEDERATED OPTIMIZATION IN HETEROGENEOUS NETWORKS. *arXiv preprint arXiv:1812.06127*, 2018B.

Bibliography

- TIAN LI, MAZIAR SANJABI, AND VIRGINIA SMITH. FAIR RESOURCE ALLOCATION IN FEDERATED LEARNING. *arXiv preprint arXiv:1905.10497*, 2019b.
- TIAN LI, ANIT KUMAR SAHU, MANZIL ZAHEER, MAZIAR SANJABI, AMEET TALWALKAR, AND VIRGINIA SMITH. FEDDANE: A FEDERATED NEWTON-TYPE METHOD. *arXiv preprint arXiv:2001.01920*, 2020.
- XIANG LI, KAIXUAN HUANG, WENHAO YANG, SHUSEN WANG, AND ZHIHUA ZHANG. ON THE CONVERGENCE OF FEDAVG ON NON-IID DATA. *arXiv preprint arXiv:1907.02189*, 2019c.
- YUANZHI LI, TENGJU MA, AND HONGYANG ZHANG. ALGORITHMIC REGULARIZATION IN OVER-PARAMETERIZED MATRIX SENSING AND NEURAL NETWORKS WITH QUADRATIC ACTIVATIONS. IN *Conference on Learning Theory (COLT)*, 2018c.
- XIANFENG LIANG, SHUHEN SHEN, JINGCHANG LIU, ZHEN PAN, ENHONG CHEN, AND YIFEI CHENG. VARIANCE REDUCED LOCAL SGD WITH LOWER COMMUNICATION COMPLEXITY. *arXiv preprint arXiv:1912.12844*, 2019.
- YUJUN LIN, SONG HAN, HUIZI MAO, YU WANG, AND WILLIAM J DALLY. DEEP GRADIENT COMPRESSION: REDUCING THE COMMUNICATION BANDWIDTH FOR DISTRIBUTED TRAINING. IN *International Conference on Learning Representations (ICLR)*, 2018.
- LIYUAN LIU, HAOMING JIANG, PENGCHENG HE, WEIZHU CHEN, XIAODONG LIU, JIANFENG GAO, AND JIAWEI HAN. ON THE VARIANCE OF THE ADAPTIVE LEARNING RATE AND BEYOND. *arXiv preprint arXiv:1908.03265*, 2019.
- SIJIA LIU, PIN-YU CHEN, XIANGYI CHEN, AND MINGYI HONG. SIGNSGD VIA ZERO-TH-ORDER ORACLE. IN *International Conference on Learning Representations (ICLR)*, 2018.
- YANLI LIU, YUAN GAO, AND WOTAO YIN. AN IMPROVED ANALYSIS OF STOCHASTIC GRADIENT DESCENT WITH MOMENTUM. *arXiv 2007.07989*, 2020.
- HAIHAO LU, SAI PRANEETH KARIMIREDDY, NATALIA PONOMAREVA, AND VAHAB MIRROKNI. ACCELERATING GRADIENT BOOSTING MACHINES. IN *International Conference on Artificial Intelligence and Statistics*, PAGES 516–526. PMLR, 2020.
- LIANGCHEN LUO, YUANHAO XIONG, AND YAN LIU. ADAPTIVE GRADIENT METHODS WITH DYNAMIC BOUND OF LEARNING RATE. IN *International Conference on Learning Representations (ICLR)*, 2019.
- YISHAY MANSOUR, MEHRYAR MOHRI, AND AFSHIN ROSTAMIZADEH. DOMAIN ADAPTATION: LEARNING BOUNDS AND ALGORITHMS. *arXiv preprint arXiv:0902.3430*, 2009.
- YISHAY MANSOUR, MEHRYAR MOHRI, JAE RO, AND ANANDA THEERTHA SURESH. THREE APPROACHES FOR PERSONALIZATION WITH APPLICATIONS TO FEDERATED LEARNING. *arXiv preprint arXiv:2002.10619*, 2020.

- CHARLES H MARTIN AND MICHAEL W MAHONEY. IMPLICIT SELF-REGULARIZATION IN DEEP NEURAL NETWORKS: EVIDENCE FROM RANDOM MATRIX THEORY AND IMPLICATIONS FOR LEARNING. *arXiv*, ABS/1810.01075, 2018.
- RAHUL MAZUMDER, TREVOR HASTIE, AND ROBERT TIBSHIRANI. SPECTRAL REGULARIZATION ALGORITHMS FOR LEARNING LARGE INCOMPLETE MATRICES. *Journal of Machine Learning Research*, 11(AUG):2287–2322, 2010.
- BRENDAN MCMAHAN, EIDER MOORE, DANIEL RAMAGE, SETH HAMPSON, AND BLAISE AGÜERA Y ARCAS. COMMUNICATION-EFFICIENT LEARNING OF DEEP NETWORKS FROM DECENTRALIZED DATA. IN *Proceedings of AISTATS*, PAGES 1273–1282, 2017.
- ADITYA KRISHNA MENON, ANKIT SINGH RAWAT, SASHANK J REDDI, SEUNGYEON KIM, AND SANJIV KUMAR. WHY DISTILLATION HELPS: A STATISTICAL PERSPECTIVE. *arXiv preprint arXiv:2005.10419*, 2020.
- EL MAHDI EL MHAMDI, RACHID GUERRAOUI, AND SÉBASTIEN ROUAULT. THE HIDDEN VULNERABILITY OF DISTRIBUTED LEARNING IN BYZANTIUM. *arXiv preprint arXiv:1802.07927*, 2018.
- EL MAHDI EL MHAMDI, RACHID GUERRAOUI, AND SÉBASTIEN ROUAULT. DISTRIBUTED MOMENTUM FOR BYZANTINE-RESILIENT STOCHASTIC GRADIENT DESCENT. IN *ICLR 2021 - International Conference on Learning Representations*, 2021.
- STANISLAV MINSKER ET AL. GEOMETRIC MEDIAN AND ROBUST ESTIMATION IN BANACH SPACES. *Bernoulli*, 21(4):2308–2335, 2015.
- KONSTANTIN MISHCHENKO, EDUARD GORBUNOV, MARTIN TAKÁČ, AND PETER RICHTÁRIK. DISTRIBUTED LEARNING WITH COMPRESSED GRADIENT DIFFERENCES. *arXiv preprint arXiv:1901.09269*, 2019.
- KEN MIURA AND TATSUYA HARADA. IMPLEMENTATION OF A PRACTICAL DISTRIBUTED CALCULATION SYSTEM WITH BROWSERS AND JAVASCRIPT, AND APPLICATION TO DISTRIBUTED DEEP LEARNING. *arXiv preprint arXiv:1503.05743*, 2015.
- HOSSEIN MOBAHI, MEHRDAD FARAJTABAR, AND PETER L BARTLETT. SELF-DISTILLATION AMPLIFIES REGULARIZATION IN HILBERT SPACE. *arXiv preprint arXiv:2002.05715*, 2020.
- MEHRYAR MOHRI, GARY SIVEK, AND ANANDA THEERTHA SURESH. AGNOSTIC FEDERATED LEARNING. *arXiv preprint arXiv:1902.00146*, 2019.
- MILAD NASR, REZA SHOKRI, AND AMIR HOUMANSADR. COMPREHENSIVE PRIVACY ANALYSIS OF DEEP LEARNING: PASSIVE AND ACTIVE WHITE-BOX INFERENCE ATTACKS AGAINST CENTRALIZED AND FEDERATED LEARNING. IN *2019 IEEE symposium on security and privacy (SP)*, PAGES 739–753. IEEE, 2019.

Bibliography

- ANGELIA NEDICH, ALEX OLSHEVSKY, AND WEI SHI. A GEOMETRICALLY CONVERGENT METHOD FOR DISTRIBUTED OPTIMIZATION OVER TIME-VARYING GRAPHS. IN *2016 IEEE 55th Conference on Decision and Control (CDC)*, PAGES 1023–1029. IEEE, 2016.
- YURI NESTEROV. *Lectures on convex optimization*, VOLUME 137. SPRINGER, 2018.
- LAM M NGUYEN, JIE LIU, KATYA SCHEINBERG, AND MARTIN TAKÁČ. SARAH: A NOVEL METHOD FOR MACHINE LEARNING PROBLEMS USING STOCHASTIC RECURSIVE GRADIENT. IN *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, PAGES 2613–2621. JMLR. ORG, 2017.
- LAM M. NGUYEN, KATYA SCHEINBERG, AND MARTIN TAKÁČ. INEXACT SARAH ALGORITHM FOR STOCHASTIC OPTIMIZATION. *arXiv preprint arXiv:1811.10105*, 2018.
- ERKKI OJA. SIMPLIFIED NEURON MODEL AS A PRINCIPAL COMPONENT ANALYZER. *Journal of mathematical biology*, 15(3):267–273, 1982.
- MYLE OTT, SERGEY EDUNOV, ALEXEI BAEVSKI, ANGELA FAN, SAM GROSS, NATHAN NG, DAVID GRANGIER, AND MICHAEL AULI. FAIRSEQ: A FAST, EXTENSIBLE TOOLKIT FOR SEQUENCE MODELING. IN *Proceedings of NAACL-HLT 2019: Demonstrations*, 2019.
- DHABALESWAR K DK PANDA, HARI SUBRAMONI, AND AMMAR AHMAD AWAN. HIGH PERFORMANCE DISTRIBUTED DEEP LEARNING: A BEGINNER’S GUIDE. IN *Symposium on Principles and Practice of Parallel Programming (PPoPP)*, 2019.
- ADAM PASZKE, SAM GROSS, FRANCISCO MASSA, ADAM LERER, JAMES BRADBURY, GREGORY CHANAN, TREVOR KILLEEN, ZEMING LIN, NATALIA GIMELSHEIN, LUCA ANTIGA, ET AL. PYTORCH: AN IMPERATIVE STYLE, HIGH-PERFORMANCE DEEP LEARNING LIBRARY. IN *Advances in Neural Information Processing Systems*, PAGES 8024–8035, 2019.
- KUMAR KSHITIJ PATEL AND AYMERIC DIEULEVEUT. COMMUNICATION TRADE-OFFS FOR SYNCHRONIZED DISTRIBUTED SGD WITH LARGE STEP SIZE. IN *33rd Conference on Neural Information Processing Systems (NeurIPS)*, 2019.
- JIE PENG, ZHAOXIAN WU, AND QING LING. BYZANTINE-ROBUST VARIANCE-REDUCED FEDERATED LEARNING OVER DISTRIBUTED NON-I.I.D. DATA. *arXiv 2009.08161*, 2020.
- VENKATADHEERAJ PICHAPATI, ANANDA THEERTHA SURESH, FELIX X YU, SASHANK J REDDI, AND SANJIV KUMAR. ADACLIP: ADAPTIVE CLIPPING FOR PRIVATE SGD. *arXiv preprint arXiv:1908.07643*, 2019.
- KRISHNA PILLUTLA, SHAM M. KAKADE, AND ZAID HARCHAOUI. ROBUST AGGREGATION FOR FEDERATED LEARNING. *arXiv preprint arXiv:1912.13445*, 2019.
- SHASHANK RAJPUT, HONGYI WANG, ZACHARY CHARLES, AND DIMITRIS PAPAILIOPOULOS. DETOX: A REDUNDANCY-BASED FRAMEWORK FOR FASTER AND MORE ROBUST GRADIENT AGGREGATION. *arXiv preprint arXiv:1907.12205*, 2019.

- SWAROOP RAMASWAMY, RAJIV MATHEWS, KANISHKA RAO, AND FRANÇOISE BEAUFAYS. FEDERATED LEARNING FOR EMOJI PREDICTION IN A MOBILE KEYBOARD. *arXiv preprint arXiv:1906.04329*, 2019.
- ADITYA RAMESH, MIKHAIL PAVLOV, GABRIEL GOH, SCOTT GRAY, CHELSEA VOSS, ALEC RADFORD, MARK CHEN, AND ILYA SUTSKEVER. ZERO-SHOT TEXT-TO-IMAGE GENERATION. *arXiv preprint arXiv:2102.12092*, 2021.
- SASHANK REDDI, ZACHARY CHARLES, MANZIL ZAHEER, ZACHARY GARRETT, KEITH RUSH, JAKUB KONEČNÝ, SANJIV KUMAR, AND H BRENDAN MCMAHAN. ADAPTIVE FEDERATED OPTIMIZATION. *arXiv preprint arXiv:2003.00295*, 2020.
- SASHANK J. REDDI, AHMED HEFNY, SUVRIT SRA, BARNABAS PO CZOS, AND ALEX SMOLA. STOCHASTIC VARIANCE REDUCTION FOR NONCONVEX OPTIMIZATION. IN *International conference on machine learning*, PAGES 314–323, 2016A.
- SASHANK J. REDDI, JAKUB KONEČNÝ, PETER RICHTÁRIK, BARNABÁS PÓCZÓS, AND ALEX SMOLA. AIDE: FAST AND COMMUNICATION EFFICIENT DISTRIBUTED OPTIMIZATION. *arXiv preprint arXiv:1608.06879*, 2016B.
- SASHANK J. REDDI, SUVRIT SRA, BARNABÁS PÓCZOS, AND ALEX SMOLA. FAST INCREMENTAL METHOD FOR SMOOTH NONCONVEX OPTIMIZATION. IN *2016 IEEE 55th Conference on Decision and Control (CDC)*, PAGES 1971–1977. IEEE, 2016C.
- SASHANK J REDDI, SATYEN KALE, AND SANJIV KUMAR. ON THE CONVERGENCE OF ADAM AND BEYOND. *International Conference on Learning Representations (ICLR)*, 2018.
- JAYANTH REGATTI AND ABHISHEK GUPTA. BEFRIENDING THE BYZANTINES THROUGH REPUTATION SCORES. *arXiv 2006.13421*, 2020.
- MARTIN RIEDMILLER AND HEINRICH BRAUN. A DIRECT ADAPTIVE METHOD FOR FASTER BACK-PROPAGATION LEARNING: THE RPROP ALGORITHM. IN *Neural Networks, 1993., IEEE International Conference on*, PAGES 586–591. IEEE, 1993.
- HERBERT ROBBINS AND SUTTON MONRO. A STOCHASTIC APPROXIMATION METHOD. *The Annals of Mathematical Statistics*, 22(3):400–407, SEPTEMBER 1951.
- NURIA RODRÍGUEZ-BARROSO, EUGENIO MARTÍNEZ-CÁMARA, M. VICTORIA LUZÓN, GERARDO GONZÁLEZ SECO, MIGUEL ÁNGEL VEGANZONES, AND FRANCISCO HERRERA. DYNAMIC FEDERATED LEARNING MODEL FOR IDENTIFYING ADVERSARIAL CLIENTS. *arXiv 2007.15030*, 2020.
- ITAY SAFRAN AND OHAD SHAMIR. HOW GOOD IS SGD WITH RANDOM SHUFFLING? *arXiv preprint arXiv:1908.00045*, 2019.
- SUMUDU SAMARAKOON, MEHDI BENNIS, WALID SAAD, AND MEROUANE DEBBAH. FEDERATED LEARNING FOR ULTRA-RELIABLE LOW-LATENCY V2V COMMUNICATIONS. IN *2018 IEEE Global Communications Conference (GLOBECOM)*, PAGES 1–7. IEEE, 2018.

Bibliography

- F. SATTLER, K. MÜLLER, T. WIEGAND, AND W. SAMEK. ON THE BYZANTINE ROBUSTNESS OF CLUSTERED FEDERATED LEARNING. IN *ICASSP 2020 - 2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, PAGES 8861–8865, 2020.
- JÜRGEN SCHMIDHUBER. DEEP LEARNING IN NEURAL NETWORKS: AN OVERVIEW. *Neural networks*, 61:85–117, 2015.
- MARK SCHMIDT, NICOLAS LE ROUX, AND FRANCIS BACH. MINIMIZING FINITE SUMS WITH THE STOCHASTIC AVERAGE GRADIENT. *Mathematical Programming*, 162(1-2):83–112, 2017.
- FRANK SEIDE, HAO FU, JASHA DROPPA, GANG LI, AND DONG 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.
- ANDREW W SENIOR, RICHARD EVANS, JOHN JUMPER, JAMES KIRKPATRICK, LAURENT SIFRE, TIM GREEN, CHONGLI QIN, AUGUSTIN ŽÍDEK, ALEXANDER WR NELSON, ALEX BRIDGLAND, ET AL. IMPROVED PROTEIN STRUCTURE PREDICTION USING POTENTIALS FROM DEEP LEARNING. *Nature*, 577(7792):706–710, 2020.
- CHRISTOPHER J SHALLUE, JAEHOON LEE, JOSEPH ANTOGNINI, JASCHA SOHL-DICKSTEIN, ROY FROSTIG, AND GEORGE E DAHL. MEASURING THE EFFECTS OF DATA PARALLELISM ON NEURAL NETWORK TRAINING. *arXiv 1811.03600*, 2018.
- OHAD SHAMIR, NATI SREBRO, AND TONG ZHANG. COMMUNICATION-EFFICIENT DISTRIBUTED OPTIMIZATION USING AN APPROXIMATE NEWTON-TYPE METHOD. IN *International conference on machine learning*, PAGES 1000–1008, 2014.
- NOAM SHAZEER AND MITCHELL STERN. ADAFACTOR: ADAPTIVE LEARNING RATES WITH SUB-LINEAR MEMORY COST. IN *International Conference on Machine Learning*, PAGES 4596–4604. PMLR, 2018.
- WEI SHI, QING LING, GANG WU, AND WOTAO YIN. EXTRA: AN EXACT FIRST-ORDER ALGORITHM FOR DECENTRALIZED CONSENSUS OPTIMIZATION. *SIAM Journal on Optimization*, 25(2):944–966, 2015.
- KAREN SIMONYAN AND ANDREW ZISSERMAN. VERY DEEP CONVOLUTIONAL NETWORKS FOR LARGE-SCALE IMAGE RECOGNITION. *arXiv preprint arXiv:1409.1556*, 2014.
- VIRGINIA SMITH, SIMONE FORTE, CHENXIN MA, MARTIN TAKÁČ, MICHAEL JORDAN, AND MARTIN JAGGI. CoCoA: A GENERAL FRAMEWORK FOR COMMUNICATION-EFFICIENT DISTRIBUTED OPTIMIZATION. *Journal of Machine Learning Research*, 18(230):1–49, 2018.
- JINHYUN SO, BASAK GULER, AND A. SALMAN AVESTIMEHR. BYZANTINE-RESILIENT SECURE FEDERATED LEARNING. *arXiv 2007.11115*, 2020A.

- JINHYUN SO, BASAK GULER, AND A. SALMAN AVESTIMEHR. TURBO-AGGREGATE: BREAKING THE QUADRATIC AGGREGATION BARRIER IN SECURE FEDERATED LEARNING. *arXiv 2002.04156*, 2020B.
- DANIEL SOUDRY, ELAD HOFFER, MOR SHPIGEL NACSON, SURIYA GUNASEKAR, AND NATHAN SREBRO. THE IMPLICIT BIAS OF GRADIENT DESCENT ON SEPARABLE DATA. *Journal of Machine Learning Research*, 19(70), 2018.
- GW STEWART. SIMULTANEOUS ITERATION FOR COMPUTING INVARIANT SUBSPACES OF NON-HERMITIAN MATRICES. *Numerische Mathematik*, 25(2):123–136, 1976.
- GW STEWART AND JH MILLER. METHODS OF SIMULTANEOUS ITERATION FOR CALCULATING EIGENVECTORS OF MATRICES. *Topics in Numerical Analysis II*, PAGES 169–185, 1975.
- SEBASTIAN U. STICH. LOCAL SGD CONVERGES FAST AND COMMUNICATES LITTLE. *International Conference on Learning Representations (ICLR)*, 2019A.
- SEBASTIAN U. STICH. UNIFIED OPTIMAL ANALYSIS OF THE (STOCHASTIC) GRADIENT METHOD. *arXiv preprint arXiv:1907.04232*, 2019B.
- SEBASTIAN U. STICH AND SAI PRANEETH KARIMIREDDY. THE ERROR-FEEDBACK FRAMEWORK: BETTER RATES FOR SGD WITH DELAYED GRADIENTS AND COMPRESSED COMMUNICATION. *arXiv preprint arXiv:1909.05350*, 2019.
- SEBASTIAN U STICH AND SAI PRANEETH KARIMIREDDY. THE ERROR-FEEDBACK FRAMEWORK: BETTER RATES FOR SGD WITH DELAYED GRADIENTS AND COMPRESSED UPDATES. *Journal of Machine Learning Research*, 21:1–36, 2020.
- SEBASTIAN U. STICH, JEAN-BAPTISTE CORDONNIER, AND MARTIN JAGGI. SPARSIFIED SGD WITH MEMORY. IN *Advances in Neural Information Processing Systems*, PAGES 4447–4458, 2018.
- NIKKO STROM. SCALABLE DISTRIBUTED DNN TRAINING USING COMMODITY GPU CLOUD COMPUTING. IN *Sixteenth Annual Conference of the International Speech Communication Association*, 2015.
- LILI SU AND JIAMING XU. SECURING DISTRIBUTED GRADIENT DESCENT IN HIGH DIMENSIONAL STATISTICAL LEARNING. *arXiv preprint arXiv:1804.10140*, 2018.
- ZITENG SUN, PETER KAIROUZ, ANANDA THEERTHA SURESH, AND H BRENDAN MCMAHAN. CAN YOU REALLY BACKDOOR FEDERATED LEARNING? *arXiv preprint arXiv:1911.07963*, 2019.
- ANANDA THEERTHA SURESH, FELIX X. YU, SANJIV KUMAR, AND H. BRENDAN MCMAHAN. DISTRIBUTED MEAN ESTIMATION WITH LIMITED COMMUNICATION. IN *Proceedings of the 34th International Conference on Machine Learning-Volume 70*, PAGES 3329–3337. JMLR. ORG, 2017.

Bibliography

- ILYA SUTSKEVER, JAMES MARTENS, GEORGE DAHL, AND GEOFFREY HINTON. ON THE IMPORTANCE OF INITIALIZATION AND MOMENTUM IN DEEP LEARNING. IN *International conference on machine learning*, PAGES 1139–1147, 2013.
- TFF. TENSORFLOW FEDERATED DATASETS. https://www.tensorflow.org/federated/api_docs/python/tff/simulation/datasets, 2020.
- OM THAKKAR, SWAROOP RAMASWAMY, RAJIV MATHEWS, AND FRANÇOISE BEAUFAYS. UNDERSTANDING UNINTENDED MEMORIZATION IN FEDERATED LEARNING. *arXiv preprint arXiv:2006.07490*, 2020.
- QUOC TRAN-DINH, NHAN H. PHAM, DZUNG T. PHAN, AND LAM M. NGUYEN. HYBRID STOCHASTIC GRADIENT DESCENT ALGORITHMS FOR STOCHASTIC NONCONVEX OPTIMIZATION. *arXiv preprint arXiv:1905.05920*, 2019.
- NILESH TRIPURANENI, MICHAEL I JORDAN, AND CHI JIN. ON THE THEORY OF TRANSFER LEARNING: THE IMPORTANCE OF TASK DIVERSITY. *NeurIPS*, 2020.
- LESLIE G VALIANT. A THEORY OF THE LEARNABLE. *Communications of the ACM*, 27(11): 1134–1142, 1984.
- SHARAN VASWANI, FRANCIS BACH, AND MARK SCHMIDT. FAST AND FASTER CONVERGENCE OF SGD FOR OVER-PARAMETERIZED MODELS AND AN ACCELERATED PERCEPTRON. IN *AISTATS 2019 - The 22nd International Conference on Artificial Intelligence and Statistics*, 2019.
- THIJS VOGELS, SAI PRANEETH KARIMIREDDY, AND MARTIN JAGGI. POWERSGD: PRACTICAL LOW-RANK GRADIENT COMPRESSION FOR DISTRIBUTED OPTIMIZATION. IN *Advances in Neural Information Processing Systems (NeurIPS)*, 2019.
- HONGYI WANG, SCOTT SIEVERT, SHENGCHAO LIU, ZACHARY CHARLES, DIMITRIS PAPALIOPOULOS, AND STEPHEN WRIGHT. ATOMO: COMMUNICATION-EFFICIENT LEARNING VIA ATOMIC SPARSIFICATION. IN *Advances in Neural Information Processing Systems (NeurIPS)*, 2018.
- HONGYI WANG, KARTIK SREENIVASAN, SHASHANK RAJPUT, HARIT VISHWAKARMA, SAURABH AGARWAL, JY-YONG SOHN, KANGWOOK LEE, AND DIMITRIS PAPALIOPOULOS. ATTACK OF THE TAILS: YES, YOU REALLY CAN BACKDOOR FEDERATED LEARNING. *arXiv preprint arXiv:2007.05084*, 2020A.
- JIANYU WANG, QINGHUA LIU, HAO LIANG, GAURI JOSHI, AND H VINCENT POOR. TACKLING THE OBJECTIVE INCONSISTENCY PROBLEM IN HETEROGENEOUS FEDERATED OPTIMIZATION. *arXiv preprint arXiv:2007.07481*, 2020B.
- JIANYU WANG, VINAYAK TANTIA, NICOLAS BALLAS, AND MICHAEL RABBAT. SLOWMO: IMPROVING COMMUNICATION-EFFICIENT DISTRIBUTED SGD WITH SLOW MOMENTUM. *International Conference on Learning Representations (ICLR)*, 2020C.

- JIANYU WANG, ZACHARY CHARLES, ZHENG XU, GAURI JOSHI, H BRENDAN MCMAHAN, MARUAN AL-SHEDIVAT, GALEN ANDREW, SALMAN AVESTIMEHR, KATHARINE DALY, DEEPESH DATA, ET AL. A FIELD GUIDE TO FEDERATED OPTIMIZATION. *arXiv preprint arXiv:2107.06917*, 2021.
- SHIQIANG WANG, TIFFANY TUOR, THEODOROS SALONIDIS, KIN K. LEUNG, CHRISTIAN MAKAYA, TING HE, AND KEVIN CHAN. ADAPTIVE FEDERATED LEARNING IN RESOURCE CONSTRAINED EDGE COMPUTING SYSTEMS. *IEEE Journal on Selected Areas in Communications*, 37(6):1205–1221, 2019.
- JIANQIAO WANGNI, JIALEI WANG, JI LIU, AND TONG ZHANG. GRADIENT SPARSIFICATION FOR COMMUNICATION-EFFICIENT DISTRIBUTED OPTIMIZATION. IN *Advances in Neural Information Processing Systems (NeurIPS)*, PAGES 1306–1316, 2018.
- WEI WEN, CONG XU, FENG YAN, CHUNPENG WU, YANDAN WANG, YIRAN CHEN, AND HAI LI. TERNGRAD: TERNARY GRADIENTS TO REDUCE COMMUNICATION IN DISTRIBUTED DEEP LEARNING. IN *Advances in Neural Information Processing Systems (NIPS)*, PAGES 1509–1519, 2017.
- ASHIA C WILSON, REBECCA ROELOFS, MITCHELL STERN, NATI SREBRO, AND BENJAMIN RECHT. THE MARGINAL VALUE OF ADAPTIVE GRADIENT METHODS IN MACHINE LEARNING. IN *Advances in Neural Information Processing Systems (NIPS)*, PAGES 4148–4158, 2017.
- BLAKE WOODWORTH, KUMAR KSHITIJ PATEL, AND NATHAN SREBRO. MINIBATCH VS LOCAL SGD FOR HETEROGENEOUS DISTRIBUTED LEARNING. *arXiv preprint arXiv:2006.04735*, 2020A.
- BLAKE WOODWORTH, KUMAR KSHITIJ PATEL, SEBASTIAN U STICH, ZHEN DAI, BRIAN BULLINS, H BRENDAN MCMAHAN, OHAD SHAMIR, AND NATHAN SREBRO. IS LOCAL SGD BETTER THAN MINIBATCH SGD? IN *37th International Conference on Machine Learning (ICML)*, 2020B.
- BLAKE E WOODWORTH, JIALEI WANG, ADAM SMITH, H. BRENDAN MCMAHAN, AND NATI SREBRO. GRAPH ORACLE MODELS, LOWER BOUNDS, AND GAPS FOR PARALLEL STOCHASTIC OPTIMIZATION. IN *Advances in neural information processing systems*, PAGES 8496–8506, 2018.
- JIAXIANG WU, WEIDONG HUANG, JUNZHOU HUANG, AND TONG ZHANG. ERROR COMPENSATED QUANTIZED SGD AND ITS APPLICATIONS TO LARGE-SCALE DISTRIBUTED OPTIMIZATION. IN *International Conference on Machine Learning (ICML)*, PAGES 5321–5329, 2018.
- HAN XIAO, KASHIF RASUL, AND ROLAND VOLLGRAF. FASHION-MNIST: A NOVEL IMAGE DATASET FOR BENCHMARKING MACHINE LEARNING ALGORITHMS, 2017.
- CONG XIE, OLUWASANMI KOYEJO, AND INDRANIL GUPTA. ZENO: DISTRIBUTED STOCHASTIC GRADIENT DESCENT WITH SUSPICION-BASED FAULT-TOLERANCE. IN *ICML 2019 - 35th International Conference on Machine Learning*, 2019.

Bibliography

- CONG XIE, OLUWASANMI KOYEJO, AND INDRANIL GUPTA. FALL OF EMPIRES: BREAKING BYZANTINE-TOLERANT SGD BY INNER PRODUCT MANIPULATION. IN *UAI - Proceedings of The 35th Uncertainty in Artificial Intelligence Conference*, 2020.
- HANG XU, CHEN-YU HO, AHMED M ABDELMONIEM, ARITRA DUTTA, EL HOUCINE BERGOU, KONSTANTINOS KARATSENIDIS, MARCO CANINI, AND PANOS KALNIS. COMPRESSED COMMUNICATION FOR DISTRIBUTED DEEP LEARNING: SURVEY AND QUANTITATIVE EVALUATION. TECHNICAL REPORT, 2020.
- TIMOTHY YANG, GALEN ANDREW, HUBERT EICHNER, HAICHENG SUN, WEI LI, NICHOLAS KONG, DANIEL RAMAGE, AND FRANÇOISE BEAUFAYS. APPLIED FEDERATED LEARNING: IMPROVING GOOGLE KEYBOARD QUERY SUGGESTIONS. *arXiv preprint arXiv:1812.02903*, 2018.
- DONG YIN, YUDONG CHEN, KANNAN RAMCHANDRAN, AND PETER BARTLETT. BYZANTINE-ROBUST DISTRIBUTED LEARNING: TOWARDS OPTIMAL STATISTICAL RATES. *arXiv preprint arXiv:1803.01498*, 2018A.
- DONG YIN, YUDONG CHEN, KANNAN RAMCHANDRAN, AND PETER BARTLETT. DEFENDING AGAINST SADDLE POINT ATTACK IN BYZANTINE-ROBUST DISTRIBUTED LEARNING. *arXiv preprint arXiv:1806.05358*, 2018B.
- YUICHI YOSHIDA AND TAKERU MIYATO. SPECTRAL NORM REGULARIZATION FOR IMPROVING THE GENERALIZABILITY OF DEEP LEARNING. *arXiv*, ABS/1705.10941, 2017.
- YANG YOU, IGOR GITMAN, AND BORIS GINSBURG. LARGE BATCH TRAINING OF CONVOLUTIONAL NETWORKS. *arXiv preprint arXiv:1708.03888*, 2017.
- YANG YOU, JING LI, SASHANK REDDI, JONATHAN HSEU, SANJIV KUMAR, SRINADH BHOJANAPALLI, XIAODAN SONG, JAMES DEMMEL, KURT KEUTZER, AND CHO-JUI HSIEH. LARGE BATCH OPTIMIZATION FOR DEEP LEARNING: TRAINING BERT IN 76 MINUTES. IN *International Conference on Learning Representations*, 2019.
- HAO YU, RONG JIN, AND SEN YANG. ON THE LINEAR SPEEDUP ANALYSIS OF COMMUNICATION EFFICIENT MOMENTUM SGD FOR DISTRIBUTED NON-CONVEX OPTIMIZATION. *arXiv 1905.03817*, 2019A.
- HAO YU, SEN YANG, AND SHENGHUO ZHU. PARALLEL RESTARTED SGD WITH FASTER CONVERGENCE AND LESS COMMUNICATION: DEMYSTIFYING WHY MODEL AVERAGING WORKS FOR DEEP LEARNING. IN *AAAI 2019 - Conference on Artificial Intelligence*, 2019B.
- MINGCHAO YU, ZHIFENG LIN, KRISHNA NARRA, SONGZE LI, YOUJIE LI, NAM SUNG KIM, ALEXANDER G. SCHWING, MURALI ANNAVARAM, AND SALMAN AVESTIMEHR. GRADIENT-VEQ: VECTOR QUANTIZATION FOR BANDWIDTH-EFFICIENT GRADIENT AGGREGATION IN DISTRIBUTED CNN TRAINING. IN *Advances in Neural Information Processing Systems (NeurIPS)*, 2018.

- TAO YU, EUGENE BAGDASARYAN, AND VITALY SHMATIKOV. SALVAGING FEDERATED LEARNING BY LOCAL ADAPTATION. *arXiv preprint arXiv:2002.04758*, 2020.
- XIAO-TONG YUAN AND PING LI. ON CONVERGENCE OF DISTRIBUTED APPROXIMATE NEWTON METHODS: GLOBALIZATION, SHARPER BOUNDS AND BEYOND. *arXiv preprint arXiv:1908.02246*, 2019.
- ALP YURTSEVER, MADELEINE UDELL, JOEL A TROPP, AND VOLKAN CEVHER. SKETCHY DECISIONS: CONVEX LOW-RANK MATRIX OPTIMIZATION WITH OPTIMAL STORAGE. IN *International Conference on Artificial Intelligence and Statistics (AISTATS)*, 2017.
- MANZIL ZAHEER, SASHANK REDDI, DEVENDRA SACHAN, SATYEN KALE, AND SANJIV KUMAR. ADAPTIVE METHODS FOR NONCONVEX OPTIMIZATION. IN *Advances in neural information processing systems*, PAGES 9793–9803, 2018.
- CHIYUAN ZHANG, SAMY BENGIO, MORITZ HARDT, BENJAMIN RECHT, AND ORIOL VINYALS. UNDERSTANDING DEEP LEARNING REQUIRES RETHINKING GENERALIZATION. IN *International Conference on Learning Representations (ICLR)*, 2017.
- CHIYUAN ZHANG, ORIOL VINYALS, REMI MUNOS, AND SAMY BENGIO. A STUDY ON OVERFITTING IN DEEP REINFORCEMENT LEARNING. *arXiv preprint arXiv:1804.06893*, 2018.
- HONGYANG ZHANG, YAODONG YU, JIANTAO JIAO, ERIC XING, LAURENT EL GHAOU, AND MICHAEL JORDAN. THEORETICALLY PRINCIPLED TRADE-OFF BETWEEN ROBUSTNESS AND ACCURACY. IN *International Conference on Machine Learning*, PAGES 7472–7482. PMLR, 2019A.
- JINGZHAO ZHANG, SAI PRANEETH KARIMIREDDY, ANDREAS VEIT, SEUNGYEON KIM, SASHANK J REDDI, SANJIV KUMAR, AND SUVRIT SRA. WHY ADAM BEATS SGD FOR ATTENTION MODELS. *arXiv 1912.03194*, 2019B.
- JINGZHAO ZHANG, TIANXING HE, SUVRIT SRA, AND ALI JADBABAIE. WHY GRADIENT CLIPPING ACCELERATES TRAINING: A THEORETICAL JUSTIFICATION FOR ADAPTIVITY. IN *International Conference on Learning Representations*, 2020.
- LIJUN ZHANG, MEHRDAD MAHDAVI, AND RONG JIN. LINEAR CONVERGENCE WITH CONDITION NUMBER INDEPENDENT ACCESS OF FULL GRADIENTS. IN *Advances in Neural Information Processing Systems*, PAGES 980–988, 2013A.
- SIXIN ZHANG, ANNA E CHOROMANSKA, AND YANN LECUN. DEEP LEARNING WITH ELASTIC AVERAGING SGD. IN *Advances in neural information processing systems*, PAGES 685–693, 2015.
- YUCHEN ZHANG, JOHN C DUCHI, AND MARTIN J WAINWRIGHT. COMMUNICATION-EFFICIENT ALGORITHMS FOR STATISTICAL OPTIMIZATION. *The Journal of Machine Learning Research*, 14(1):3321–3363, 2013B.

Bibliography

- JIawei ZHAO. SIGNSGD WITH MAJORITY VOTE. [GITHUB.COM/PERMIJW/SIGNSGD-WITH-MAJORITY-VOTE](https://github.com/PERMIJW/SIGNSGD-WITH-MAJORITY-VOTE), 2019. [ONLINE; ACCESSED 12-MAY-2019].
- YUE ZHAO, MENG LI, LIANGZHEN LAI, NAVEEN SUDA, DAMON CIVIN, AND VIKAS CHANDRA. FEDERATED LEARNING WITH NON-IID DATA. *arXiv preprint arXiv:1806.00582*, 2018.
- MARTIN ZINKEVICH, MARKUS WEIMER, LIHONG LI, AND ALEX J SMOLA. PARALLELIZED STOCHASTIC GRADIENT DESCENT. IN *Advances in neural information processing systems*, PAGES 2595–2603, 2010.

CURRICULUM VITAE – KARIMIREDDY

Personal information

Name: Sai Praneeth Reddy KARIMIREDDY **Website:** spkreddy.org
Google Scholar ID: [wKJeOQoAAAAJ](https://scholar.google.com/citations?user=wKJeOQoAAAAJ) (url) **Email:** sai.karimireddy@epfl.ch
Address: INJ 338, EPFL, Lausanne CH 1015 **Phone:** (+41)786851439
Research Interests: Optimization, Collaborative (federated, decentralized) learning.

Education

EPFL Lausanne 09.2016–08.2021
PhD candidate in Computer Science
Advised by Prof. Martin Jaggi. PhD thesis defense scheduled on 15.08.2021.
Thesis title: *Optimization Algorithms for Collaborative Machine Learning*.

Indian Institute of Technology, New Delhi 07.2011–06.2016
Bachelor and Master of Technology in Computer Science
Advised by Prof. Sandeep Sen. Master thesis defended in 06.2016.
Thesis title: *Sampling Techniques in Computational Geometry*.

Employment

EPFL, Lausanne 09.2016–08.2021
PhD in optimization for machine learning advised by Prof. Martin Jaggi.

Google Research, Remote 06.2020–08.2020
Developed new scalable algorithms and software for federated learning.

Google Research, New York 07.2019–11.2019
Developed new optimization methods for NLP and for federated learning.

The Broadline, New Delhi (now rebranded as Loki.ai) 05.2016–08.2016
Developed core recommendation engine and tools to assist journalists using NLP.

Microsoft Research, Bangalore 04.2015–06.2015
Worked on Multi-armed bandit problems with Shipra Agrawal.

Xerox Research Center India, Bangalore 04.2014–06.2014
Worked on distributed and privacy preserving data mining with Shailesh Vaya.

Awards

Swiss National Science Foundation PostDoc Fellowship. 2021–2022
Best paper award in FL ICML workshop 2021. 2021
Outstanding performance bonus (thrice) awarded by EPFL. 2018, 2019, 2020
Top Reviewer (twice) awarded at ICML and NeurIPS 2019, 2020
Travel Awards (thrice) to present research at ICML and NeurIPS 2018, 2019, 2019
EDIC Fellowship awarded by EPFL to selected PhD students 2016–2017
Xerox Travel Award (twice) to present at CSCW, and PODC 2016, 2016²⁹³
Ministry of Human Resource Development Scholarship 2015–2016
Semester Merit Award given to the top 7% of the batch 2015
Kishore Vaigyanik Protsahana Yojana scholarship 2009–2011

Student projects supervised

Master Theses:

1. Eloise Berthier, *Differential privacy of cyclic non-convex SGD*. 04.2019–07.2019
2. Ignacio Aleman, *Scalable causal inference from noise residuals*. 03.2020–08.2020
3. Felix Grimberg, *Optimal model averaging for collaborative learning*. 09.2020–01.2021
4. William Cappelletti, *Byzantine robust decentralized optimization*. 09.2020–01.2021

Semester projects and internships:

1. Anastasiia Koloskova, *Efficient greedy methods for optimization*. 06.2017–09.2017
2. Quentin Rebjock, *Error feedback for gradient compression*. 09.2018–01.2019
3. Fedor Moiseev, *Bias correction for non-iid federated learning data*. 09.2019–01.2020
4. Felix Grimberg, *Incentivizing decentralized learning*. 02.2020–05.2020
5. Ilyas Fatkhullin, *Accelerated inexact gradient descent*. 06.2020–10.2020
6. Mahmoud Hegazy, *Transfer learning via parameter sharing*. 06.2020–10.2020
7. Andrei Afonin, *Bias correction for semi-supervised learning*. 09.2020–01.2021
8. Andrei Afonin, *Model agnostic communication protocols*. 02.2021–05.2021
9. Usman A. Khan, *Federated image segmentation for science*. 02.2021–05.2021

Junior PhD students mentored who became my co-authors:

1. Thijs Vogels, *Low rank methods for practical gradient compression*. 09.2018–present
2. Lie He, *Scalable private, secure and robust machine learning*. 03.2020–present

Teaching activities

Teaching assistant for the following courses

1. *Sublinear algorithms for big data analysis* (CS-448) Spring 2017.
Along with helping grading and tutorial sessions, calibrated the difficulty of exams.
2. *Machine learning* (CS-443) Fall 2017, 2018, 2019, 2020.
Developed several practical exercises, held tutorial sessions, and set exam questions.
3. *Optimization for machine learning* (CS-439) Spring 2018, 2019, 2020.
Developed most of the exercises and exam questions. Also held tutorial sessions.

Service

Co-organized the AutoTrain challenge at AMLD 2020.

Reviewing activity:

- ICML 2018, 2019, 2020. Awarded top reviewer 2019, 2020.
- Expert reviewer at ICML 2021.
- NeurIPS 2018, 2019, 2020. Awarded top reviewer 2019.
- AISTATS 2018, 2019, 2020.
- Expert external reviewer for ALT 2020, CDC 2021, and PODC 2021.
- Journal of Machine Learning (JMLR)
- IEEE/ACM Transactions on Networking (ToN)
- European Journal of Operational Research (EJOR)
- Optimization Methods and Software

Public outreach and knowledge transfer

1. **Advisory member** at the Infectious Diseases Data Observatory (IDDO), Oxford.
Developing a collaborative data sharing platform for health crises. 06.2020–12.2021

2. **Interviewed** by ZettaBytes, EPFL's public outreach channel, on the challenges and opportunities in federated learning (watch here).
3. Helped Facebook implement our algorithm PowerSGD in **PyTorch**, the most popular open-source deep learning framework. This potentially reduces the communication costs up to $100\times$ and energy requirements up to $2\times$ for deep learning.
4. Helped **Google** implement our federated learning algorithms SCAFFOLD and MIME. These are currently being tested for production and potentially become the backbone of Google's future machine learning platform.

Open source software

Released and maintain open source ready to use software for open science.

1. Running fast and flexible federated learning simulations: ([FedJAX](#))
2. Real world decentralized learning across devices: ([DecentralizedAI](#))
3. Efficient compressed communication for deep learning: ([PowerSGD](#))
4. Compressed communication for decentralized deep learning: ([PowerGossip](#))
5. Implementing sign based gradient compression: ([Error feedback SGD](#))

Invited Talks

- **SIAM MDS Mathematics of Data Science** 06.2020: *Optimization for deep learning.*
- **Federated learning one world seminar** 06.2020: *SCAFFOLD: Stochastic controlled averaging for federated learning.*
- **ZettaBytes** (interview) 11.2019: *Challenges and opportunities in federated learning*
- **Google** Federated Learning Talks 11.2019: *SCAFFOLD for federated learning.*
- **Google** 08.2019, New York: *Error-feedback fixes gradient compression methods*
- **MIT** 11.2018, Boston: *Optimal inexact accelerated algorithms*
- **ETH Zurich** 08.2018: *Accelerated first order methods with approximate subproblems*
- **International Symposia on Mathematical Programming** 07.2018, Bordeaux: *Accelerated first order methods with approximate subproblems*
- **Swiss Machine Learning Day** 11.2017, Lausanne: *Making optimization algorithms adaptive to system conditions*

Publications List

*Note: * indicates that the authors with equal contributions or alphabetical ordering.*

Peer-reviewed conference and journal publications

1. Learning from History for Byzantine Robust Optimization.
ICML 2021 (url)
Sai Praneeth Karimireddy, Lie He, Martin Jaggi.
2. Quasi-global Momentum: Decentralized Deep Learning on Heterogeneous Data.
ICML 2021 (url)
Tao Lin, **Sai Praneeth Karimireddy**, Sebastian Stich, Martin Jaggi.
3. Practical Communication Compression in Decentralized Deep Learning.
NeurIPS 2020 (url).
Thijs Vogels, **Sai Praneeth Karimireddy**, Martin Jaggi.
4. Why are Adaptive Methods Good for Attention Models?
NeurIPS 2020 (url).
Jingzhao Zhang, **Sai Praneeth Karimireddy**, Andreas Veit, Seungyeon Kim, Sashank Reddi, Sanjiv Kumar
5. Weight Erosion: An Update Aggregation Scheme for Personalized Collaborative Machine Learning.
DART 2020 (url).
Felix Grimberg, Mary-Anne Hartley, Martin Jaggi, **Sai Praneeth Karimireddy**.
6. Accelerated Gradient Boosted Machines.
AISTATS 2020 (url).
Haihao Lu*, **Sai Praneeth Karimireddy***, Natalia Ponomareva, Vahab Mirrokni.
7. SCAFFOLD: Stochastic Controlled Averaging for Federated Learning.
ICML 2020 (url).
Sai Praneeth Karimireddy, Satyen Kale, Mehryar Mohri, Sashank Reddi, Sebastian Stich, Ananda Theertha Suresh.
8. The Error-Feedback Framework: Better Rates for SGD with Delayed Gradients and Compressed Communication.
JMLR 2020 (url).
Sebastian Stich, **Sai Praneeth Karimireddy**.
9. PowerSGD: Practical Low-rank Gradient Compression for Distributed Opt.
NeurIPS 2019 (url).
Thijs Vogels, **Sai Praneeth Karimireddy**, Martin Jaggi.
10. Error Feedback fixes SignSGD and other Gradient Compression Schemes.
ICML 2019 (url) **Long talk**.
Sai Praneeth Karimireddy, Quentin Rebjock, Sebastian Stich, Martin Jaggi.
- 296 11. Efficient greedy coordinate descent for composite problems.
AISTATS 2019 (url)
Sai Praneeth Karimireddy*, Anastasia Koloskova*, S Stich, Martin Jaggi.

12. On Matching Pursuit and Coordinate Descent.
ICML 2018 (url)
Francesco Locatello*, Anant Raj*, **Sai Praneeth Karimireddy**, Sebastian Stich, Martin Jaggi.
13. Adaptive Balancing of Gradient and Update Computation Times using Approximate Subproblem Solvers.
AISTATS 2018 (url) Oral.
Sai Praneeth Karimireddy, Sebastian Stich, Martin Jaggi.
14. Some results on a class of mixed van der Waerden numbers.
Rocky Mountain Journal of Mathematics Vol.48 2018 (url). (Pre-PhD work)
Kaushik Maran*, **Sai Praneeth Reddy***, Dravyansh Sharma*, Amitabha Tripathi*.
15. Assignment Techniques for Crowdsourcing Sensitive Tasks.
CSCW 2016 (url) (pre-PhD work).
Elisa Celis*, **Sai Praneeth Reddy***, Ishaan Singh*, Shailesh Vaya*.
16. Brief Announcement: Multi-Broadcasting under the SINR Model.
PODC 2016 (url) (pre-PhD work).
Sai Praneeth Reddy, Shailesh Vaya.

Peer-reviewed workshop papers

1. Optimal Model Averaging: Towards Personalized Collaborative Learning.
FL ICML workshop 2021 Best paper.
Felix Grimberg, Mary-Anne Hartley, **Sai Praneeth Karimireddy**, Martin Jaggi.
2. Byzantine-Robust Learning on Heterogeneous Datasets via Resampling.
NeurIPS workshop 2020 (SPICY FL) (url)
Lie He*, **Sai Praneeth Karimireddy***, Martin Jaggi.
3. Secure Byzantine Machine Learning.
NeurIPS workshop 2020 (SPICY FL) (url)
Lie He, **Sai Praneeth Karimireddy**, Martin Jaggi.

Submitted/under review publications

1. Mime: Mimicking Centralized Stochastic Algorithms in Federated Learning.
Arxiv 2020 (url).
Sai Praneeth Karimireddy, Martin Jaggi, Satyen Kale, Mehryar Mohri, Sashank Reddi, Sebastian Stich, Ananda Theertha Suresh.
2. Byzantine-Robust Learning on Heterogeneous Datasets via Resampling.
Arxiv 2020 (url)
Sai Praneeth Karimireddy*, Lie He*, Martin Jaggi.
3. RelaySum for Decentralized Deep Learning on Heterogeneous Data.
Under submission.
Thijs Vogels*, Lie He*, Anastasia Koloskova, **Sai Praneeth Karimireddy**, Sebastian Stich, Martin Jaggi.
4. Secure Byzantine Machine Learning.
Submitted to TIST Special Issue on Federated Learning (url)
Lie He, **Sai Praneeth Karimireddy**, Martin Jaggi.

Patents

1. Methods and systems for creating tasks.
Applied in US 2013, Granted in US 2017 ([url](#)).
Shailesh Vaya*, Akshayaram Srinivasan*, **Sai Praneeth Reddy K***.
2. Methods and systems for recognizing handwriting in handwritten documents.
Applied in US 2013, Granted in US 2015 ([url](#)).
Shailesh Vaya*, Akshayaram Srinivasan*, **Sai Praneeth Reddy K***.
3. Apparatus and method for secure digital coupon verification.
Applied in US 2015, Granted ([url](#))
Sai Praneeth Reddy K*, Ishaan Preet Singh*, Shailesh Vaya*