

UC Irvine

UC Irvine Electronic Theses and Dissertations

Title

Collaborative Learning under Communication Constraints

Permalink

<https://escholarship.org/uc/item/07p462nx>

ISBN

9798247990529

Author

Ortega Sanchez Colomer, Tomas

Publication Date

2026-06-10

Copyright Information

This work is made available under the terms of a Creative Commons Attribution-NonCommercial-NoDerivatives License, available at

<https://creativecommons.org/licenses/by-nc-nd/4.0/>

Peer reviewed|Thesis/dissertation

UNIVERSITY OF CALIFORNIA,
IRVINE

Collaborative Learning under Communication Constraints

DISSERTATION

submitted in partial satisfaction of the requirements
for the degree of

DOCTOR OF PHILOSOPHY

in Electrical and Computer Engineering

by

Tomàs Ortega Sanchez-Colomer

Dissertation Committee:
Professor Hamid Jafarkhani, Chair
Professor A. Lee Swindlehurst
Associate Professor Yanning Shen

TABLE OF CONTENTS

	Page
LIST OF FIGURES	vi
LIST OF TABLES	x
LIST OF ALGORITHMS	xi
ACKNOWLEDGMENTS	xii
VITA	xiii
ABSTRACT OF THE DISSERTATION	xviii
 1 Introduction	 1
 2 Gossiped and Quantized Online Multi-Kernel Learning	 5
2.1 Introduction	5
2.2 Problem Formulation	6
2.3 Algorithm and Regret Analysis	10
2.4 Experimental Results and Conclusions	15
 3 Quantized and Asynchronous Federated Learning	 18
3.1 Introduction	18
3.1.1 Related Work	19
3.1.2 Contributions	23
3.2 System Model	25
3.2.1 General Quantizer Model	25
3.2.2 Proposed Algorithm	25
3.3 Algorithm Analysis	28
3.3.1 Problem Formulation	28
3.3.2 Convergence Analysis for QAFEL	30
3.4 Results and Discussion	35
3.4.1 Logistic Regression Experiments	35
3.4.2 Neural Network Experiments	37
3.5 Conclusion	44

4	Communication Compression for Distributed Learning with Aggregate and Server-Guided Feedback	45
4.1	Introduction	45
4.2	Related Work	49
4.3	Proposed Frameworks	49
4.3.1	CAFe Overview	51
4.3.2	CAFe-S Overview	53
4.4	Analysis	54
4.4.1	Analysis of CAFe	55
4.4.2	Analysis of CAFe-S	56
4.4.3	Discussion of the Theoretical Results	57
4.5	Experimental Results	58
4.5.1	Experimental Setup	59
4.5.2	Validation of Working Principle	59
4.5.3	Performance of CAFe	60
4.5.4	Validation of CAFe-S	64
4.6	Conclusion	65
5	Decentralized Parameter-Free Online Learning	66
5.1	Introduction	66
5.2	The Decentralized Coin-Betting Algorithm	69
5.2.1	High-Level Idea: Local Betting and Global Gossiping	69
5.2.2	Algorithm Description	70
5.2.3	Summary of Main Results	72
5.3	Preliminaries for the Analysis	73
5.3.1	Subgradients and Online Convex Optimization	73
5.3.2	The Coin-Betting Framework	73
5.3.3	Gossip Algorithms for Consensus	76
5.4	Analysis	77
5.4.1	Multi-Agent Betting Game	77
5.4.2	Regret Analysis	82
5.5	Operational Details	92
5.6	Experiments	95
5.6.1	Experimental Setup	96
5.6.2	The Challenge of Hyperparameter Tuning	98
5.6.3	Impact of Network Connectivity	99
5.6.4	The Communication-Regret Trade-Off	101
5.6.5	Performance on Real-World Datasets	102
5.7	Conclusion	103
6	Decentralized Parameter-Free Online Learning with Compressed Gossip	104
6.1	Introduction	104
6.2	Related Work	107
6.3	Model and Setup	109
6.3.1	Problem Model	110

6.3.2	Communication Graph and Compression	110
6.4	Algorithm	112
6.4.1	High-Level Idea	112
6.4.2	Coin-Betting Potentials and Predictions	113
6.4.3	DECO-EF State and Update	116
6.4.4	Notation Used in the Analysis	117
6.5	Main Results	117
6.6	Analysis	121
6.6.1	Regret Decomposition	121
6.6.2	Centralized Coin-Betting	122
6.6.3	Compressed Tracking	123
6.6.4	From State Disagreement to Regret	124
6.7	Operational Details	125
6.7.1	Choosing the Potential	125
6.7.2	Choosing the Gossip Schedule	125
6.7.3	Communication/Contraction Trade-Off	125
6.7.4	Compression and Communication	126
6.8	Experiments	127
6.9	Conclusion	130
7	Conclusion	131
	Bibliography	133
	Appendix A Supporting Material for QAFEL	144
A.1	Additional Derivations	144
A.2	Algorithm Pseudocode	151
	Appendix B Supporting Material for CAFe	153
B.1	Experimental Details	153
B.1.1	Compression Parameter Details	154
B.1.2	Learning Rate Sensitivity	154
B.2	Proofs of Convergence Theorems	155
B.2.1	Proof of Theorem 4.6 (DCGD)	156
B.2.2	Proof of Theorem 4.7 (CAFe)	157
B.2.3	Proof of Theorem 4.9 (CAFe-S)	163
	Appendix C Supporting Material for DECO-EF	165
C.1	Appendix Roadmap and Notation	165
C.2	Coin-Betting Reductions	167
C.3	Network Dynamics	172
C.4	Disagreement and Network Regret	178
C.5	KT Specialization	180
C.6	Shifted-Exponential Specialization	183
C.7	Global-Loss and Prediction-Disagreement Details	186

C.8 Pathwise Network-Regret Add-On	190
--	-----

LIST OF FIGURES

	Page
2.1 Comparison between the gossiped OMKL, OMKL from [1] and the single-kernel SGD, labelled Koloskova, from [2]. We have used $\sigma = 1$ for the latter. The labels indicate the used topology for each algorithm.	14
2.2 The effect of graph topology, without quantization. Average loss function at each iteration, with the Credit-Card dataset. Since the Credit-Card dataset has $n = 30000$ data observations, and $J = 20$ nodes, we have run the algorithm 1500 iterations.	16
3.1 Flowchart comparing synchronous and asynchronous FL.	20
3.2 Numerical example of the effect of direct quantization (Naive) vs. the unquantized counterpart. No client quantization is performed. We show unbiased (QSGD [3]) and biased (top-k [4]) server quantizer examples. We consider a logistic regression problem with ℓ_2 regularization on the <i>mushrooms</i> dataset from LIBSVM [5]. The simulation parameters are: 100 clients, delays following a half-normal distribution, server buffer size of 10, client learning rate of 2, server learning rate of 0.1, and ℓ_2 regularization strength of $1/8124$, where 8124 is the number of samples in the dataset. The y -axis illustrates the difference $f(x) - f^*$, where $f(x)$ is the global model cost at a given iteration, and f^* is the optimal cost.	22
3.3 Block diagram for updating the hidden-state.	23
3.4 QAFEL block diagram. Shaded parts occur on the client side. Dashed lines indicate communication between the clients and the server.	26
3.5 Numerical example of the effect of naive quantization vs. our proposed algorithm (QAFEL). No client quantization is performed. The y -axis illustrates the difference $f(x) - f^*$, where $f(x)$ is the global model cost at a given iteration, and f^* is the optimal cost. Subfigure (a) illustrates the unbiased server quantizer case. Subfigure (b) illustrates the biased server quantizer case. . .	36
3.6 Numerical example of the effect of local steps. The y -axis illustrates the difference $f(x) - f^*$, where $f(x)$ is the global model cost at a given iteration, and f^* is the optimal cost. QAFEL is run without quantization and a varying quantity of local steps. More local steps lead to larger client drift and faster convergence, but to a less optimal solution.	37

3.7	Comparison of communication metrics for QAFeL and FedBuff across varying concurrency levels (clients training in parallel) to reach 60% target accuracy on the CIFAR-10 dataset. QAFeL employs 4-bit QSGD quantization at both the server and client: (a) the number of client updates in thousands, (b) the total GB uploaded by the clients, and (c) the GB broadcasted by the server.	40
3.8	Comparison of communication metrics for QAFeL and FedBuff across varying concurrency levels (clients training in parallel) to reach 90% target accuracy on the CelebA dataset. QAFeL employs 4-bit QSGD quantization at both the server and client: (a) the number of client updates in thousands, (b) the total GB uploaded by the clients, and (c) the GB broadcasted by the server. . . .	40
3.9	Comparison of communication metrics for QAFeL, FedBuff, FedAsync, FedBuff with direct quantization, and FedAsync with direct quantization across varying concurrency levels (clients training in parallel) to reach 54% target accuracy on the Shakespeare dataset. QAFeL employs 4-bit QSGD quantization at both the server and client: (a) the number of client updates in thousands, (b) the total GB uploaded by the clients, and (c) the GB broadcasted by the server. FedBuff with direct quantization only has one data-point because for concurrencies 500 and 1000 the algorithm diverged for at least one of the three seeds.	41
3.10	Evaluation curves for the Shakespeare experiments with concurrency 500 and FedBuff with direct 4-bit quantization for both client and server. We observe that one of the three runs diverges.	42
4.1	Error feedback block diagram. Typically, each client stores its own control variate and must update it after each training round. The server stores its own copy of each client’s control variate.	46
4.2	Analysis of the CAFe working principle on a synthetic logistic regression task. Left: Global training loss showing convergence. Middle: The Compression Gain Ratio (ρ) remains below 1.0 for the majority of training, indicating that the difference vector has a smaller norm than the raw update. The ratio approaches 1.0 as the model converges. Right: The log-density histogram shows that CAFe updates are more peaked at zero (sparser) than direct updates.	61
4.3	Learning curves comparing direct compression (Direct) and CAFe with low-rank compression in the CIFAR-10 task for iid and non-iid settings. CAFe results are presented with solid lines, and Direct compression results are presented with dashed lines. CAFe demonstrates faster convergence and higher accuracy.	61
4.4	Accuracy of CAFe-S as a function of server’s data representativeness (β) on CIFAR-100. While CAFe-S improves as the server’s data become more representative, CAFe remains superior due to the stability of the aggregated predictor.	64
5.1	Operational loop for an agent using DECO-i. Key elements specific to this variant, related to tracking and communicating wealth, are highlighted in red.	94

5.2	Operational loop for DECO-ii. The process is simplified by decoupling from wealth, requiring only the accumulated gradients to be maintained and gossiped. Differences from DECO-i are highlighted in green.	95
5.3	Final cumulative network loss of DOGD as a function of its initial learning rate (η_0) on synthetic data, compared to the constant performance of our parameter-free DECO algorithms. The U-shaped curve for DOGD highlights its extreme sensitivity to hyperparameter choice, where a suboptimal setting leads to loss that is orders of magnitude worse. Our DECO algorithms achieve good performance without any tuning, regardless of the potential function used.	98
5.4	Performance of DECO-ii (KT) on synthetic data for Erdős-Rényi random graphs with varying edge probability p . Top: Cumulative network loss. Cumulative loss growth improves with better connectivity, as faster information mixing reduces the disagreement among agents. Bottom: Per-round network loss (smoothed with a 250-round moving average). The smoothed loss shows that more connected networks not only accumulate less total loss but also converge faster to the centralized ideal's steady-state per-round loss.	100
5.5	Cumulative network loss of DECO-ii (KT) on a cycle graph with different gossip schedules, $q(t)$. Increasing the communication budget from constant to logarithmic and linear progressively reduces loss, allowing the algorithm to approach the centralized oracle's performance. This highlights the explicit trade-off between communication cost and learning performance.	101
5.6	Final cumulative network loss on four real-world LIBSVM datasets. Each panel shows the sensitivity of DOGD to its initial learning rate (η_0), producing a characteristic U-shaped curve. Note that we do not plot loss values that are five times the minimum value to allow for good visualization. The horizontal lines show the performance of the parameter-free DECO variants, which achieve competitive loss without any tuning. This highlights the unreliability of DOGD when its hyperparameters are not properly set for a specific dataset.	102
6.1	System model for DECO-EF. Agents make local predictions, observe local convex losses, and exchange compressed state differences over the communication graph. The analysis evaluates the averaged network loss against a single comparator u	112
6.2	LIBSVM online-regression experiments, showing decentralized online gradient descent (DOGD) tuning sensitivity on the ring graph against parameter-free methods.	127

6.3 Synthetic quantized-communication diagnostics for DECO-EF. All decentralized runs use $\gamma = 1/2$, and the compressed runs use b -bit quantization. Each plotted point or curve is the average over one hundred independent synthetic streams, with error bars or shaded bands showing one standard deviation. Panels (a) and (b) use the complete graph with $q(t) = 1$, the topology that most clearly separates quantization resolution from slow graph mixing. Panel (a) reports the paired final cumulative network loss difference $L_b - L_{\text{exact}}$ against the no-quantization reference run with the same stream seed and relaxation. Panel (b) reports cumulative prediction disagreement against transmitted Mbits. Panel (c) fixes $b = 3$ and $q(t) = 1$, and compares the disagreement on the ring, 4×5 toroidal grid, and complete graph. . . . 128

LIST OF TABLES

	Page
3.1 Related work comparison. The number of algorithm rounds is denoted by T .	24
3.2 Summary of notation.	31
3.3 Chosen hyperparameters	39
3.4 Communication metrics of QAF _e L to reach the CelebA target validation accuracy (90%) with different QSGD combinations.	43
4.1 Feature comparison with state-of-the-art distributed learning frameworks . .	50
4.2 Accuracy for CAFe and Direct compression (MNIST, EMNIST).	62
4.3 Accuracy for CAFe and Direct compression (CIFAR-10, CIFAR-100).	62
4.4 Communication cost (bits per parameter) for CAFe and Direct compression (MNIST, EMNIST).	63
4.5 Communication cost (bits per parameter) for CAFe and Direct compression (CIFAR-10, CIFAR-100).	63
5.1 Mappings from $F_t(x)$ to $\beta_t(x)$	75
6.1 Positioning of representative algorithm families. Comparator-adaptive (CA) denotes explicit dependence on $\ u\ $	109
6.2 Representative mean-square compressor coefficients.	112

LIST OF ALGORITHMS

	Page
1 Gossiped and Quantized OMKL at Node j	11
2 Distributed Compressed Gradient Descent (DCGD)	51
3 Compressed Aggregate Feedback (CAFe)	52
4 Server-Guided Compressed Aggregate Feedback (CAFe-S)	53
5 Decentralized Coin-Betting (DECO), Variants (i) and (ii)	71
6 DECO-EF: One Round at Agent n	116
7 QAFEL-server	152
8 QAFEL-client	152
9 QAFEL-client-background	152

ACKNOWLEDGMENTS

I would like to express my gratitude to my advisor, Professor Hamid Jafarkhani, for his guidance, patience, and support throughout my PhD journey. I am also grateful to my committee members, Professor Lee Swindlehurst and Professor Yanning Shen, for their time and feedback.

My thanks also go to my colleagues and collaborators at the Jafarkhani Lab for their support: Carles Diaz Vilor, Berkay Guler, Hossein Maleki, Saeed Karimi Bidhendi, Mojtaba Ahmadi Almasi, Kemeng Han, Ruizhan Shen, Zhaoxu Luo, and Reza Barzegaran.

I am deeply grateful to the mentors and collaborators who put their trust in me and gave me the opportunity to work with them: Marc Sanchez-Net, Dariush Divsalar, Kar-Ming Cheung, and everyone at JPL; Xiaoxiao Li, Wenlong Deng, Chun-Yin Huang, and my Vector Institute colleagues; as well as Sahil Tikale, Jordan Schmerge, Kedar Namjoshi, and many others at Bell Labs. I would also like to extend a special thanks to Professor Stephan Mandt and Yibo Yang, among many other colleagues at UC Irvine.

This work would not have been possible without the warmth of my friends and family. I cannot thank Nora, As, Joaquim, Miquel, and everyone in my family enough. Thank you as well to Tomeu, Mar, Irene, Laura Mendoza, David, Mateo, Raquel, Berta, Julieta, and all my friends back in Catalonia, as well as Laura, Anna, Manel, Jordi, Noe, Olga, and all the Spanish friends in California who made me feel at home.

I also extend my thanks to Carles Balsells, Carles Domingo, Pau, Pol, and all the UPC friends in the US and back home, and of course to Nick, Caro, Daniel, Daniela, Michael, Chris, Schuyler, Ariel, Melissa, and all the friends I have made during my time in California. While I am certain I have omitted some names, please know that I am deeply thankful to everyone who offered their support and friendship along the way.

Finally, I gratefully acknowledge the financial support that made this research possible, including the NSF Award ECCS-2207457, the Balsells Graduate Fellowship, the EECS Department Fellowship, the Henry Samueli Endowed Fellowship, the IEEE Signal Processing Society Scholarship, and the JPL Graduate Fellowship.

VITA

Tomàs Ortega Sanchez-Colomer

EDUCATION

Doctor of Philosophy in Electrical and Computer Engineering	2026
University of California, Irvine	<i>Irvine, California</i>
Master of Science in Mathematics	2021
Universitat Politècnica de Catalunya (UPC)	<i>Barcelona, Spain</i>
Bachelor of Science in Mathematics	2020
CFIS, Universitat Politècnica de Catalunya (UPC)	<i>Barcelona, Spain</i>
Bachelor of Engineering in Telecommunications Engineering	2020
CFIS, Universitat Politècnica de Catalunya (UPC)	<i>Barcelona, Spain</i>

RESEARCH EXPERIENCE

Research Intern	Summer 2025
Nokia Bell Labs	<i>Murray Hill, New Jersey</i>
Research Intern	Summer 2024
Vector Institute	<i>Toronto, Canada</i>
Graduate Fellow / JVS RP Intern	2020, 2022
NASA Jet Propulsion Laboratory	<i>Pasadena, California</i>
Research Assistant	Fall 2020
Universitat Politècnica de Catalunya (UPC)	<i>Barcelona, Spain</i>
Research Project Collaborator	Winter 2019
HP	<i>Barcelona, Spain</i>

TEACHING EXPERIENCE

Teaching Assistant, Probability for Engineers (EECS 55)	2024–2026
University of California, Irvine	<i>Irvine, California</i>

HONORS AND AWARDS

Henry Samueli Endowed Fellowship Samueli School of Engineering, UC Irvine	2026 <i>Irvine, California</i>
EECS Graduate Student of the Year Engineering Student Council, UC Irvine	2024 <i>Irvine, California</i>
Signal Processing Scholarship IEEE Signal Processing Society	2023 <i>Online</i>
Early Career Invitation ICML Federated Learning Workshop	2023 <i>Honolulu, Hawaii</i>
Honorable Mention, Évariste Galois Prize for best MSc thesis Catalan Society of Mathematics	2023 <i>Catalonia, Spain</i>
JPL Graduate Fellowship NASA Jet Propulsion Laboratory	2022 <i>Pasadena, California</i>
EECS Department Fellowship Samueli School of Engineering, UC Irvine	2021 <i>Irvine, California</i>
Balsells Graduate Fellowship Balsells Fellowship Program	2021 <i>Irvine, California</i>
CFIS Scholarship CFIS, Universitat Politècnica de Catalunya	2015–2020 <i>Barcelona, Spain</i>

REFEREED JOURNAL PUBLICATIONS

Quantized and Asynchronous Federated Learning 2024
IEEE Transactions on Communications

Correlating electrochemical stimulus to structural change in liquid electron microscopy videos using the structural dissimilarity metric 2024
Ultramicroscopy

Gossiped and Quantized Online Multi-Kernel Learning 2023
IEEE Signal Processing Letters

LOS/NLOS Estimators for mmWave Cellular Systems With Blockages 2022
IEEE Wireless Communications Letters

REFEREED CONFERENCE PUBLICATIONS

Decentralized Optimization in Networks with Arbitrary Delays 2024
IEEE International Conference on Communications (ICC)

Geometric Quasi-Cyclic Low Density Parity Check Codes 2024
Discrete Mathematics Days 2024

Asynchronous Federated Learning with Bidirectional Quantized Communications and Buffered Aggregation 2023
International Conference on Machine Learning (ICML)–FL Workshop

Acquisition and tracking of high dynamics Doppler profiles for space applications 2021
IEEE Aerospace Conference

Adaptive-Sweep Algorithm for Spacecraft Carrier Acquisition and Tracking: System Analysis and Implementation 2021
IEEE Aerospace Conference

PREPRINTS AND WORKING PAPERS

Federated Rate-Distortion Bottlenecks: Optimal Communication Allocation for Personalized Federated Learning **2026**

In preparation for submission

Decentralized Parameter-Free Online Learning with Compressed Gossip **2026**

arXiv preprint arXiv:2605.27831

Illusory evidence asymmetrically impacts visual performance and metacognition **2026**

In preparation for submission

Communication Compression for Distributed Learning with Aggregate and Server-Guided Feedback **2025**

arXiv preprint arXiv:2512.22623

Decentralized Parameter-Free Online Learning **2025**

arXiv preprint arXiv:2510.15644

Communication Compression for Distributed Learning without Control Variates **2024**

arXiv preprint arXiv:2412.04538

Decentralized Optimization in Time-Varying Networks with Arbitrary Delays **2024**

arXiv preprint arXiv:2405.19513

Practical implementation of geometric quasi-cyclic LDPC codes **2024**

arXiv preprint arXiv:2405.20524

PATENTS

Systems and Methods for Quantized Machine Learning, Federated Learning and Bidirectional Network Communication **2024**

US Patent Application (Pending)

SOFTWARE

FLSim: Federated Learning Simulator

<https://github.com/TomasOrtega/FLSim>

Python simulator reproducing experiments from “Quantized and Asynchronous Federated Learning”.

DT-GO Decentralized Optimization

<https://github.com/TomasOrtega/DT-GO>

Python prototype with performance benchmarks for decentralized logistic regression and FL tasks.

Noisy Lattice Denoising

<https://github.com/TomasOrtega/Noisy-lattice-problem>

MATLAB, Python and AMPL solver for a noisy lattice point recovery problem.

DECO: Decentralized Coin-Betting for Online Learning

<https://github.com/TomasOrtega/Deco>

Python simulator reproducing experiments from “Decentralized Parameter-Free Online Learning”.

Quasi Cyclic Representations of Classical Generalized Quadrangles

<https://github.com/TomasOrtega/QuasiCyclicGQs>

Largest repository of known QC representations of point-line incidence matrices of the classical GQs.

Contributions to formalizing mathematics proofs in Lean

<https://github.com/dwrensha/compfiles>, <https://github.com/thefundamentaltheor3m/Sphere-Packing-Lean>, <https://github.com/ericluap/OrderedSemigroups>

Contributions to the compfiles project, the Sphere Packing project and the Ordered Semigroups project in the Lean proof assistant.

UC Irvine’s Samueli School of Engineering L^AT_EX

<https://www.overleaf.com/latex/templates/tagged/uci>, <https://www.overleaf.com/latex/templates/unofficial-uci-engineering-beamer-template/qvqfvstgkgn>

Designed UCI Engineering L^AT_EX beamer template for presentations, now hosted on Overleaf.

ABSTRACT OF THE DISSERTATION

Collaborative Learning under Communication Constraints

By

Tomàs Ortega Sanchez-Colomer

Doctor of Philosophy in Electrical and Computer Engineering

University of California, Irvine, 2026

Professor Hamid Jafarkhani, Chair

This dissertation studies collaborative learning when communication is limited. The challenge is: data are spread across many devices, but the model should improve as if the network were working together. Moreover, the devices may be connected by a network with limited bandwidth, and links may be delayed or unavailable. The algorithms in this dissertation are designed to address these issues.

The first part considers online kernel learning over a graph. We extend distributed online multi-kernel learning beyond complete graphs by combining gossip with a hidden-state quantization scheme, and we prove sublinear regret with finite-bit messages.

We then study asynchronous federated learning with bidirectional quantization. The resulting algorithm uses a shared hidden state to prevent quantization errors from accumulating between the server and clients, and it retains the standard ergodic convergence rate for non-convex stochastic optimization.

The next part focuses on compression in distributed learning. Previous literature has shown that error-feedback can be used to correct for compression errors, but it requires clients to store local hidden states. We propose two methods that remove this requirement and allow clients to be stateless while still using error-feedback.

The final part moves to decentralized online learning. We show that we can use a coin-betting framework, previously proposed for the single-agent case, to remove learning-rate tuning from decentralized online convex optimization, and we give comparator-adaptive network-regret guarantees. We also extend this framework to compressed gossip, giving parameter-free network-regret bounds under compressed communication.

Together, these results improve our understanding of how compression, hidden states, gossip, and parameter-free online learning can affect collaborative learning in communication-constrained networks, and how we can co-design algorithms to work around communication bottlenecks.

Chapter 1

Introduction

Machine learning is no longer a centralized endeavor. Models are trained and used on phones, sensors, hospitals, laboratories, and data centers, to name a few examples. The data are local, the devices are different from each other, and the network is often the bottleneck. On top of this, often times there are multiple stakeholders involved in the learning process, and this discourages the use of a central server that can coordinate the learning process. In these settings, learning algorithms need to take into account the communication constraints to guarantee good performance.

This dissertation uses the term *collaborative learning* for this family of problems. The common thread is that many agents contribute to one learning task. Sometimes there is a server, as in federated learning. Sometimes there is no server, and agents only communicate with their neighbors. In both cases, the basic tension is the same: the learning method needs information from the network, but communication is expensive, delayed, compressed, or all of the above.

Two constraints appear throughout the dissertation. The first is communication. Sending full-precision models or gradients at every round is often unrealistic, especially at the edge.

Quantization, sparsification, and other compressed messages help, but they also introduce error that can accumulate if the protocol is not designed carefully. We take inspiration from video compression, where instead of compressing frame by frame, the encoder and decoder maintain a shared hidden state that they update with compressed differences. Several chapters therefore use hidden states or feedback variables to make compressed communication behave well.

The second constraint is tuning. Many online and decentralized methods require learning rates or schedules that depend on quantities that are not known in advance. This is inconvenient in centralized learning and worse in networks, where the right choice can depend on the graph, the losses, the horizon, and the scale of the comparator. Chapters 5 and 6 use parameter-free online learning to reduce this dependence on manual tuning. This enables drop-in algorithms that can be deployed without worrying about learning rates.

Chapter 2 studies gossiped and quantized online multi-kernel learning [6]. Earlier distributed online multi-kernel learning guarantees relied on complete communication graphs. We replace this requirement with a gossip mechanism and a hidden-state quantization scheme, allowing agents to communicate only with their neighbors while still achieving sublinear regret.

Chapter 3 studies Quantized and Asynchronous Federated Learning (QAFEL) [7, 8]. Asynchrony helps avoid waiting for slow clients, but direct quantization can create persistent disagreement between the server and clients. QAFEL uses bidirectional quantization with a shared hidden state, so both sides track the same compressed model state. The analysis shows that the method keeps the usual non-convex convergence order while reducing communication.

Chapter 4 studies communication compression for distributed learning with aggregate and server-guided feedback [9, 10]. The main idea is to avoid per-client error-feedback memory.

We propose an algorithm, Compressed Aggregate Feedback (CAFe), and its server-guided variant CAFe-S. CAFe uses the previous aggregate update as a shared predictor, and CAFe-S uses a server-side data set to build a more targeted predictor. Both methods let clients compress differences from a common reference, which is useful when clients should remain stateless.

Chapter 5 studies Decentralized Coin-Betting (DECO) [11]. DECO addresses decentralized online convex optimization without learning-rate tuning. The algorithm combines gossip with coin-betting ideas from parameter-free online learning and gives comparator-adaptive network-regret guarantees.

Chapter 6 studies DECO-EF, which is DECO with Error Feedback. It adds compressed gossip to the parameter-free decentralized setting. Each agent maintains a clean accumulated state and a compressed tracker, and communicates compressed state differences during gossip. The result is a parameter-free decentralized online learning method with compressed communication and network-regret guarantees.

Some research completed during the Ph.D. is listed in the curriculum vitae but is not developed as dissertation chapters. This includes work on decentralized optimization with delays [12, 13], quasi-cyclic LDPC codes [14–16], and liquid electron microscopy video analysis [17]. These projects fall outside the main narrative of this dissertation, whose chapters focus on collaborative learning under communication constraints.

Together, the chapters develop a set of tools for learning over networks where communication is not free. The tools are different in each setting, but the design principle is the same: the communication protocol and the learning rule have to be designed together.

We also acknowledge the limitations of the theory presented in this work, which often relies on (standard) simplifying assumptions, such as convexity, bounded gradients, or bounded loss heterogeneity across agents. While these assumptions do not necessarily hold in practice,

especially for learning tasks involving neural nets, they provide a starting point for understanding the behavior of these algorithms and help guide designs in practice. Each chapter includes experiments that show the practical performance of the proposed methods, and we hope that the theoretical insights can inspire further research.

Chapter 2

Gossiped and Quantized Online Multi-Kernel Learning

2.1 Introduction

Many of the latest efforts in machine learning are focused on bringing learning as close to data collection as possible. This is of practical interest in a diverse array of applications, sensor networks in particular. In sensor networks, nodes are often communication-constrained, operate distributively, and can send only a few bits to their neighbors.

There are several papers in the literature that consider the problem of designing distributed learning methods for Online Multi-Kernel Learning (OMKL). In particular, [1] contains the design and regret analysis of quantized and distributed OMKL for a fully connected (complete) graph. Although the fully connected case is important, it is not applicable in many sensor networks. This chapter designs a new algorithm that expands the theoretical guarantees of the distributed and quantized OMKL in [1] to an arbitrary general non-fully connected graph. Unlike [1], our method uses a novel hidden state quantization scheme

in conjunction with a gossip algorithm [18]. One major advantage, compared with [1], is that our algorithm does not need to communicate the loss function. Another example of a distributed OMKL that works for a non-fully connected graph is presented in [19]. It requires communicating an unlimited number of bits between neighbors, and its assumptions are stricter than those of [1]. To manage the required communication throughput, which is often a performance bottleneck [20, 21], this chapter provides the properties of [19] under looser assumptions and while communicating only a limited number of bits between neighboring nodes.

In [2], a distributed stochastic gradient descent scheme with quantized communication is discussed. The proposed algorithm uses an extra shared “hidden” state and can be used for the single-kernel case, but not multi-kernel settings. In addition to extending [1] to general non-fully connected networks, our work can be considered as the extension of [2] to multi-kernel cases.

There are a variety of learning tasks that OMKL is suited for, such as multi-kernel classification, regression, and clustering [22–24]. For a more in-depth review of the literature, see [1].

2.2 Problem Formulation

Let us consider a network with J nodes and model it as an undirected connected graph, where two nodes are connected if and only if they can communicate. Our set of nodes V , and the set of edges E , define a communication graph $G = (V, E)$. At each instant of time t , Node j receives the data string $\mathbf{x}_t^j \in \mathbb{R}^d$ and the desired response $y_t^j \in \mathbb{R}$. Our task is to find an approximating function $f(\mathbf{x}_t^j)$ for y_t^j . As in [1], f belongs to a Reproducing Kernel Hilbert Space (RKHS) $\mathcal{H} = \left\{ f \mid f(\mathbf{x}) = \sum_{t=1}^{\infty} \sum_{j=1}^J \alpha_t^j \kappa(\mathbf{x}, \mathbf{x}_t^j) \right\}$, where $\kappa(\mathbf{x}, \mathbf{x}_t^j) : \mathbb{R}^d \times \mathbb{R}^d \rightarrow \mathbb{R}$ is a kernel function that measures the similarity between x and $\mathbf{x}_t^j$. Let us consider the following

optimization problem

$$\min_{f \in \mathcal{H}} \sum_{t=1}^T \sum_{j=1}^J \mathcal{C}(f(\mathbf{x}_t^j), y_t^j) + \lambda \Omega(\|f\|_{\mathcal{H}}^2), \quad (2.1)$$

where $\mathcal{C}$ is a cost function and $\lambda > 0$ is a regularization parameter that controls an increasing function Ω . An optimal solution for this problem exists in the form $\hat{f}(\mathbf{x}) = \sum_{t=1}^T \sum_{j=1}^J \alpha_t^j \kappa(\mathbf{x}, \mathbf{x}_t^j)$, where α_t^j are real numbers [25].

In multi-kernel learning, a weighted combination of several kernels is selected to improve the performance, compared to single-kernel learning. A convex combination of the kernels $\{\kappa_p\}_{p=1}^P$, where $\kappa_p \in \mathcal{H}_p$ is an RKHS, is also an RKHS, denoted by $\bar{\mathcal{H}} = \mathcal{H}_1 \oplus \cdots \oplus \mathcal{H}_P$ [25]. Here, $\oplus$ indicates the direct sum of Hilbert spaces. Using $\bar{\mathcal{H}}$ instead of $\mathcal{H}$, we turn problem Equation (2.1) into:

$$\min_{\{\bar{w}_p^j\}, \{f_p\}} \sum_{t=1}^T \sum_{j=1}^J \mathcal{C}\left(\sum_{p=1}^P \bar{w}_p^j f_p(\mathbf{x}_t^j), y_t^j\right) + \lambda \Omega\left(\left\|\sum_{p=1}^P \bar{w}_p^j f_p\right\|_{\bar{\mathcal{H}}}^2\right), \quad (2.2a)$$

$$\text{s.t. } \sum_{p=1}^P \bar{w}_p^j = 1, \bar{w}_p^j \geq 0, f_p \in \mathcal{H}_p. \quad (2.2b)$$

To evade the curse of dimensionality, a Random Feature (RF) approximation is used [1, 26]. For normalized shift-invariant kernels, i.e., $\kappa(\mathbf{x}, \mathbf{x}') = \kappa(\mathbf{x} - \mathbf{x}')$, their Fourier transforms $\pi_\kappa(v)$ exist, and due to normalization, $\pi_\kappa(0) = 1$. Also, viewing π_κ as a pdf, $\kappa(\mathbf{x} - \mathbf{x}') = \int \pi_\kappa(v) e^{jv^\top(\mathbf{x} - \mathbf{x}')} dv = \mathbb{E}_v[e^{jv^\top(\mathbf{x} - \mathbf{x}')}]$. Gaussian, Laplacian, and Cauchy kernels are three examples of normalized shift-invariant kernels that we can use [27]. However, any family of shift invariant kernels can be used for the RF approximation we describe next. Drawing D

i.i.d. samples $\{\mathbf{v}_i\}_{i=1}^D$ from $\pi_\kappa(v)$, we define

$$z_V(\mathbf{x}) = \frac{1}{\sqrt{D}} \begin{bmatrix} \sin(\mathbf{v}_1^\top \mathbf{x}) \\ \vdots \\ \sin(\mathbf{v}_D^\top \mathbf{x}) \\ \cos(\mathbf{v}_1^\top \mathbf{x}) \\ \vdots \\ \cos(\mathbf{v}_D^\top \mathbf{x}) \end{bmatrix}, \quad (2.3)$$

which has the property that $\kappa(\mathbf{x}, \mathbf{x}') = \mathbb{E}_v [z_V(\mathbf{x}')^\top z_V(\mathbf{x})]$. So, given a fixed $\{\mathbf{v}_i\}_{i=1}^D$, we can approximate $\kappa(\mathbf{x}, \mathbf{x}') \approx z_V(\mathbf{x}')^\top z_V(\mathbf{x})$. This is called the RF approximation, and it is an unbiased and consistent estimation of $\kappa(\mathbf{x}, \mathbf{x}')$ [26]. Thus, a weight vector $\boldsymbol{\theta} \in \mathbb{R}^{2D}$ can be constructed such that

$$\begin{aligned} \hat{f}(\mathbf{x}) &= \sum_{t=1}^T \sum_{j=1}^J \alpha_t^j \kappa(\mathbf{x}, \mathbf{x}_t^j) \approx \sum_{t=1}^T \sum_{j=1}^J \alpha_t^j z_V(\mathbf{x}_t^j)^\top z_V(\mathbf{x}) \\ &= \boldsymbol{\theta}^\top z_V(\mathbf{x}). \end{aligned} \quad (2.4)$$

Then, the loss function is defined as

$$\mathcal{L}(f(\mathbf{x})) = \mathcal{C}(\boldsymbol{\theta}^\top z_V(\mathbf{x}), y) + \lambda \Omega(\|\boldsymbol{\theta}\|^2), \quad (2.5)$$

and the following equations are obtained for Kernel p and Node j :

$$\hat{f}_{p,t}^j(\mathbf{x}_t^j) = \boldsymbol{\theta}_{p,t}^j{}^\top z_{V_p}(\mathbf{x}_t^j), \quad (2.6)$$

$$\boldsymbol{\theta}_{p,t+1}^j = \boldsymbol{\theta}_{p,t}^j - \eta \nabla \mathcal{L}(\boldsymbol{\theta}_{p,t}^j{}^\top z_{V_p}(\mathbf{x}_t^j), y_t^j), \quad (2.7)$$

$$w_{p,t+1}^j = w_{p,t}^j \exp -\eta \mathcal{L}(\hat{f}_{p,t}^j(\mathbf{x}_t^j)), \quad (2.8)$$

where $\eta \in (0, 1)$ is a learning rate. The weights are normalized as $\bar{w}_{p,t}^j = w_{p,t}^j / \sum_{p=1}^P w_{p,t}^j$ to

have

$$\hat{f}_t^j(x) = \sum_{p=1}^P \bar{w}_{p,t}^j \hat{f}_{p,t}^j(\mathbf{x}_t^j). \quad (2.9)$$

The above equations represent how local online multi-kernel learning models can be built [1, 26]. In contrast to previous research, each node j in our scenario can only communicate with its neighboring nodes in the set $\mathcal{S}_j \subseteq V$. Thus, the local model $\boldsymbol{\theta}_{p,t}^j$ of each node may be different from node to node. Therefore, our algorithm must ensure that all nodes converge to the same model. For this purpose, the local models are propagated using a gossip algorithm. The nodes calculate the weighted average of the information provided by their neighbors at each communication round and use it to update their local information. The weight Node j assigns to the information coming from Node i is defined as w'_{ij} . Notice that the sum of weights $\sum_{i \in \mathcal{S}_j} w'_{ij} = 1$ for all j since the weights of a weighted average must add up to one. We impose $w'_{ij} = w'_{ji}$ such that we can associate the weights to the edges of the undirected communication graph G . Thus, the weights define the gossip matrix $\mathbf{W}'$, which is $J \times J$ and doubly stochastic [28–30]. The spectral gap of $\mathbf{W}'$ is denoted by $\rho = 1 - \lambda_2(\mathbf{W}')$ and satisfies $0 < \rho \leq 1$, where $\lambda_2(\mathbf{W}')$ represents the second eigenvalue of $\mathbf{W}'$ in descending order.

Such a gossip algorithm works very well when the nodes communicate their states perfectly with their neighbors. However, practically, only a few bits can be communicated, i.e., information needs to be quantized before being shared with the neighbors. We use a random quantizer $Q : \mathbb{R}^n \rightarrow \mathbb{R}^n$, to represent an arbitrary vector $\mathbf{x} \in \mathbb{R}^n$ with $Q(\mathbf{x})$ in an efficient way. Our results work for any random quantizer that satisfies

$$\mathbb{E}_Q \|Q(\mathbf{x}) - \mathbf{x}\|^2 \leq (1 - \delta) \|\mathbf{x}\|^2, \quad \forall \mathbf{x} \in \mathbb{R}^n, \quad (2.10)$$

for some $\delta > 0$, which we will call the compression parameter. Here, $\mathbb{E}_Q$ denotes the expected

value with respect to the internal randomness of $Q(\cdot)$. In simulations, we use the randomized quantizer of [3]. Each element of a non-zero vector $\mathbf{v} \in \mathbb{R}^n$, i.e., v_i , is quantized by

$$Q_M(v_i) = \|v\| \text{sign}(v_i) \xi_i(\mathbf{v}, M), \quad (2.11)$$

where $M = 2^b - 1$ is the number of quantization levels and defining $l = \lfloor M \frac{v_i}{\|v\|} \rfloor$,

$$\xi_i(\mathbf{v}, M) = \begin{cases} \frac{l}{M} & \text{with probability } 1 - M \frac{v_i}{\|v\|} + l, \\ \frac{l+1}{M} & \text{otherwise,} \end{cases} \quad (2.12)$$

is represented by b bits. Following [3, Lemma 3.1], Equation (2.10) holds for $\delta = 1 - \min\left(\frac{2D}{M^2}, \frac{\sqrt{2D}}{M}\right) > 0$.

2.3 Algorithm and Regret Analysis

We present Algorithm 1 for gossiped and quantized OMKL. For ease of notation, we will denote $\mathcal{L}(\boldsymbol{\theta}_{p,t}^j \top z_{V_p}(\mathbf{x}_t^j), y_t)$ as $\mathcal{L}(\boldsymbol{\theta}_{p,t}^j)$ from now on. In Algorithm 1, at each time instance, nodes collect their local data and transform them according to the RF approximation. Then, the kernel losses are computed and used to update the kernel weights. For the gossip step, we define a hidden state $\mathbf{h}_{p,t}^j \in \mathbb{R}^{2D}$ for each node j that is known for all neighbors in $\mathcal{S}_j$ because it is updated by the same quantized values known to all neighbors. The new local state, $\boldsymbol{\theta}_{p,t}^j$, will be a sum of $\boldsymbol{\theta}_{p,t-1}^j$, which is an auxiliary variable where we store the local learning, plus a gossip step of size γ with the information from the hidden states. Subsequently, each node j prepares the update to the hidden state by quantizing the difference between its local state $\boldsymbol{\theta}_{p,t}^j$ and the common hidden state $\mathbf{h}_{p,t}^j$. This quantized difference is sent to the neighbors and is used by them to collectively update the hidden state. Finally, each node performs local learning with a step size η , and stores it in the auxiliary variable $\boldsymbol{\theta}_{p,t}^j$.

The role of a hidden state and quantized update is to have an accurate representation of the neighbor's states without the need for broadcasting the full state at each time instance.

Algorithm 1 Gossiped and Quantized OMKL at Node j

```

1: Initialize  $w_p^j = 1/P$  and  $\mathbf{h}_p^j = \boldsymbol{\theta}_p^j = \boldsymbol{\theta}'_p^j = 0$  for all  $p$ .
2: for  $t = 1, \dots, T$  do
3:   Obtain data  $\mathbf{x}_t^j$  and construct  $z_p(\mathbf{x}_t^j)$  for all  $p$ .
4:   Compute  $\mathcal{L}(\hat{f}_{p,t}^j(\mathbf{x}_t^j))$  for all  $p$ .
5:   Update  $w_{p,t}^j$  according to Equation (2.8).
6:   Compute  $\mathcal{L}(\hat{f}_t^j(\mathbf{x}_t^j))$ 
7:   for  $p = 1, \dots, P$  do
8:     Update using the hidden states  $\boldsymbol{\theta}_{p,t}^j = \boldsymbol{\theta}_{p,t-1}^j + \gamma \sum_{i=1}^J w'_{ij}(\mathbf{h}_{p,t}^i - \mathbf{h}_{p,t}^j)$ .
9:     Compress  $\mathbf{q}_{p,t}^j = Q(\boldsymbol{\theta}_{p,t}^j - \mathbf{h}_{p,t}^j)$ .
10:    for  $i$  neighbor of  $j$ , including  $j$  do
11:      Send  $\mathbf{q}_{p,t}^j$  to  $i$  and receive  $\mathbf{q}_{p,t}^i$ .
12:      Update  $\mathbf{h}_{p,t+1}^i = \mathbf{h}_{p,t}^i + \mathbf{q}_{p,t}^i$ .
13:    end for
14:    As in Equation (2.7), update  $\boldsymbol{\theta}_{p,t}^j = \boldsymbol{\theta}_{p,t}^j - \eta \nabla \mathcal{L}(\boldsymbol{\theta}_{p,t}^j)$ .
15:  end for
16: end for

```

In what follows, we provide regret analysis to study the performance of our algorithm. First, we make the following assumptions similar to what is done in the literature.

ASSUMPTION 2.1. *Let us assume a K -smooth loss function*

$$\|\nabla \mathcal{L}(\boldsymbol{\theta}_1) - \nabla \mathcal{L}(\boldsymbol{\theta}_2)\| \leq K \|\boldsymbol{\theta}_1 - \boldsymbol{\theta}_2\| \quad \forall \boldsymbol{\theta}_1, \boldsymbol{\theta}_2 \in \mathbb{R}^{2D}, \quad (2.13)$$

with a bounded gradient $\|\nabla \mathcal{L}(\boldsymbol{\theta})\| \leq L$.

ASSUMPTION 2.2. *Let us assume a convex loss function with respect to $\boldsymbol{\theta}$, i.e.,*

$$\mathcal{L}(\boldsymbol{\theta}_{p,t}^j) - \mathcal{L}(\boldsymbol{\theta}) \leq \nabla \mathcal{L}(\boldsymbol{\theta}_{p,t}^j)^\top (\boldsymbol{\theta}_{p,t}^j - \boldsymbol{\theta}) \quad \forall \boldsymbol{\theta} \in \mathbb{R}^{2D}. \quad (2.14)$$

ASSUMPTION 2.3. *Let us assume that we have a connected communication graph and a doubly stochastic gossip matrix $\mathbf{W}'$ whose entries are zero if and only if their corresponding edges*

are not present in the communication graph.

THEOREM 2.4. *Under Assumptions 2.1 to 2.3, the sequences $\{\hat{f}_{p,t}^j\}$ and $\{\bar{w}_{p,t}\}$ generated by the gossiped and quantized OMKL satisfy*

$$\sum_{j=1}^J \sum_{t=1}^T \mathbb{E} \left[\mathcal{L} \left(\sum_{p=1}^P \bar{w}_{p,t}^j \hat{f}_{p,t}^j(\mathbf{x}_t^j) \right) \right] - \sum_{j=1}^J \sum_{t=1}^T \mathcal{L} \left(\hat{f}_p^*(\mathbf{x}_t^j) \right) \leq J \frac{\|\boldsymbol{\theta}_p^*\|^2}{2\eta} + \eta \frac{JTL^2}{2} + \eta \frac{JTL\sqrt{12}}{c} + \frac{J \ln P}{\eta} + \eta JT, \quad (2.15)$$

where $\boldsymbol{\theta}_p^*$ is the optimal solution to the problem for the p -th kernel, $c = \frac{\rho^2 \delta}{82}$, ρ is the spectral gap of the gossip matrix $\mathbf{W}'$, and $\delta > 0$ is the quantizer's compression parameter.

To prove Theorem 2.4, we first need to present Lemma 2.5, which bounds the difference between a single kernel loss and the optimal loss, and Lemma 2.6, which bounds the error added by the kernel weights.

LEMMA 2.5. *Let us choose the consensus step size as in [31, Thm. 4.1], i.e., $\gamma = \frac{\rho^2 \delta}{16\rho + \rho^2 + 4\beta^2 + 2\rho\beta^2 - 8\rho\delta}$, where we have defined $\beta = \|\mathbf{I} - \mathbf{W}'\|_2 \in [0, 2]$. Then, for $0 < \eta \leq 1$ and under Assumptions 2.1 to 2.3, we have*

$$\sum_{j=1}^J \sum_{t=1}^T \mathbb{E} \left[\mathcal{L} \left(\hat{f}_{p,t}^j(\mathbf{x}_t^j) \right) \right] - \sum_{j=1}^J \sum_{t=1}^T \mathcal{L} \left(\hat{f}_p^*(\mathbf{x}_t^j) \right) \leq J \frac{\|\boldsymbol{\theta}_p^*\|^2}{2\eta} + \eta \frac{JTL^2}{2} + \eta \frac{JTL\sqrt{12}}{c}. \quad (2.16)$$

Proof. Similar to [1], let us consider an arbitrary $\boldsymbol{\theta} \in \mathbb{R}^{2D}$. Define $\bar{\boldsymbol{\theta}}_{p,t} = \frac{1}{J} \sum_{j=1}^J \boldsymbol{\theta}_{p,t}^j$, and $\bar{\boldsymbol{\theta}}'_{p,t}$ and $\nabla \bar{\mathcal{L}}(\bar{\boldsymbol{\theta}}_{p,t})$ analogously. Since our gossip scheme *preserves averages*, we can deduce

$\bar{\boldsymbol{\theta}}_{p,t+1} = \bar{\boldsymbol{\theta}}'_{p,t} = \bar{\boldsymbol{\theta}}_{p,t} - \eta \nabla \bar{\mathcal{L}}(\boldsymbol{\theta}_{p,t})$. Thus,

$$\|\bar{\boldsymbol{\theta}}_{p,t+1} - \boldsymbol{\theta}\|^2 = \|\bar{\boldsymbol{\theta}}_{p,t} - \eta \nabla \bar{\mathcal{L}}(\boldsymbol{\theta}_{p,t}) - \boldsymbol{\theta}\|^2 \quad (2.17)$$

$$= \|\bar{\boldsymbol{\theta}}_{p,t} - \boldsymbol{\theta}\|^2 + \eta^2 \|\nabla \bar{\mathcal{L}}(\boldsymbol{\theta}_{p,t})\|^2 \quad (2.18)$$

$$- 2\eta \nabla \bar{\mathcal{L}}(\boldsymbol{\theta}_{p,t})^\top (\bar{\boldsymbol{\theta}}_{p,t} - \boldsymbol{\theta}). \quad (2.19)$$

By Assumption 2.2, we have

$$\begin{aligned} \mathcal{L}(\boldsymbol{\theta}_{p,t}^j) - \mathcal{L}(\boldsymbol{\theta}) &\leq \nabla \mathcal{L}(\boldsymbol{\theta}_{p,t}^j)^\top (\boldsymbol{\theta}_{p,t}^j + \bar{\boldsymbol{\theta}}_{p,t} - \bar{\boldsymbol{\theta}}_{p,t} - \boldsymbol{\theta}) \\ &\leq \nabla \mathcal{L}(\boldsymbol{\theta}_{p,t}^j)^\top (\bar{\boldsymbol{\theta}}_{p,t} - \boldsymbol{\theta}) + L \|\boldsymbol{\theta}_{p,t}^j - \bar{\boldsymbol{\theta}}_{p,t}\|. \end{aligned} \quad (2.20)$$

Inserting Equation (2.20) into Equation (2.19) and summing over j , we obtain

$$\begin{aligned} \sum_{j=1}^J (\mathcal{L}(\boldsymbol{\theta}_{p,t}^j) - \mathcal{L}(\boldsymbol{\theta})) &\leq \frac{\eta}{2} J \|\nabla \bar{\mathcal{L}}(\boldsymbol{\theta}_{p,t})\|^2 + \\ &\quad J \frac{\|\bar{\boldsymbol{\theta}}_{p,t} - \boldsymbol{\theta}\|^2 - \|\bar{\boldsymbol{\theta}}_{p,t+1} - \boldsymbol{\theta}\|^2}{2\eta} + L \sum_{j=1}^J \|\boldsymbol{\theta}_{p,t}^j - \bar{\boldsymbol{\theta}}_{p,t}\|, \end{aligned} \quad (2.21)$$

where we have used Assumption 2.1 to bound the gradient. Summing over t and bounding again, we derive

$$\begin{aligned} \sum_{t=1}^T \sum_{j=1}^J (\mathcal{L}(\boldsymbol{\theta}_{p,t}^j) - \mathcal{L}(\boldsymbol{\theta})) &\leq \frac{\eta}{2} J T L^2 + L \sum_{t=1}^T \sum_{j=1}^J \|\boldsymbol{\theta}_{p,t}^j - \bar{\boldsymbol{\theta}}_{p,t}\| \\ &\quad + J \frac{\|\bar{\boldsymbol{\theta}}_{p,1} - \boldsymbol{\theta}\|^2 - \|\bar{\boldsymbol{\theta}}_{p,T+1} - \boldsymbol{\theta}\|^2}{2\eta}. \end{aligned} \quad (2.22)$$

Applying [31, Lemma A.2] and Cauchy-Schwarz results in

$$\sum_{j=1}^J \mathbb{E} \|\boldsymbol{\theta}_{p,t}^j - \bar{\boldsymbol{\theta}}_{p,t}\|^2 \leq \eta^2 \frac{12JL^2}{c^2} \implies \quad (2.23)$$

$$\sum_{j=1}^J \mathbb{E} \|\boldsymbol{\theta}_{p,t}^j - \bar{\boldsymbol{\theta}}_{p,t}\| \leq \eta \frac{JL\sqrt{12}}{c}. \quad (2.24)$$

The proof is complete, by using Equation (2.24) to bound Equation (2.22) and choosing

$\boldsymbol{\theta}_{p,1}^j = \mathbf{0}$ and $\boldsymbol{\theta} = \boldsymbol{\theta}^*$. $\square$

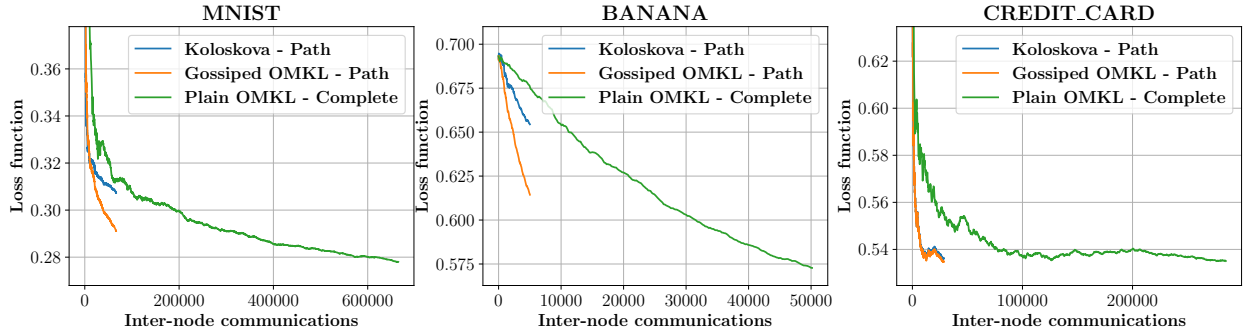


Figure 2.1: Comparison between the gossiped OMKL, OMKL from [1] and the single-kernel SGD, labelled Koloskova, from [2]. We have used $\sigma = 1$ for the latter. The labels indicate the used topology for each algorithm.

LEMMA 2.6. *Under the same conditions as Theorem 2.4,*

$$\sum_{j=1}^J \sum_{t=1}^T \sum_{p=1}^P \bar{w}_{p,t}^j \mathcal{L} \left(\hat{f}_{p,t}^j(\mathbf{x}_t^j) \right) - \sum_{j=1}^J \sum_{t=1}^T \mathcal{L} \left(\hat{f}_{p',t}^j(\mathbf{x}_t^j) \right) \leq \frac{J \ln P}{\eta} + \eta JT. \quad (2.25)$$

The proof of Lemma 2.6 follows the same steps as in [1, Lemma 3].

From the convexity of $\mathcal{L}(\cdot)$,

$$\mathcal{L} \left(\sum_{p=1}^P \bar{w}_{p,t} \hat{f}_{p,t}^j(\mathbf{x}_t) \right) \leq \sum_{p=1}^P \bar{w}_{p,t} \mathcal{L} \left(\hat{f}_{p,t}^j(\mathbf{x}_t) \right), \quad (2.26)$$

and Theorem 2.4 follows immediately using Lemma 2.5 and Lemma 2.6.

2.4 Experimental Results and Conclusions

To validate Theorem 2.4, we have tested Algorithm 1 with real datasets for binary classification, with different topologies, and using different values of quantization level. We have used the well-known Kernel Logistic Regression (KLR) loss function [25], i.e., the loss function Equation (2.5) is defined as

$$\ln(1 + \exp(-y \cdot \boldsymbol{\theta}^\top z_V(\mathbf{x}))) + \lambda \|\boldsymbol{\theta}\|^2, \quad (2.27)$$

where we have used $\Omega(\|\boldsymbol{\theta}\|^2) = \|\boldsymbol{\theta}\|^2$. We have conducted our experiments with three datasets: Banana, Credit-Card, and MNIST. The synthetic data from the Banana dataset [32] ($n = 5300, d = 2$) are two non-linearly separable clusters. The Credit-Card dataset [33] ($n = 30000, d = 2$) contains data from credit card users, such as their credit history and whether they have defaulted or not. It is obtained from the UCI machine learning repository [34]. The MNIST dataset [35] ($n = 70000, d = 784$) is a labeled set of handwritten digits from 0 to 9. We divide them into two classes, those that are number 8 and those that are not.

Our experimental setup has $J = 20$ nodes, dimension $D = 20$ for our RF approximation, regularization parameter $\lambda = 0.001$, and three Gaussian kernels with $\sigma \in \{1, 3, 5\}$, where a Gaussian kernel with parameter σ is defined as $\kappa(\mathbf{x}, \mathbf{x}') = \exp(-\|\mathbf{x} - \mathbf{x}'\|^2 / (2\sigma^2))$. Simulations have been performed 100 times with different sets of Random Features and the corresponding mean of the loss function is plotted. We report the loss instead of the classification accuracy because the difference in the latter is minimal. We have used the quantizer described in Equation (2.12) with 7 levels of quantization, that is, 3 bits per element in any transmitted array. Our learning rates are $\eta = 0.01$ and $\gamma = 0.9\eta = 0.009$.

Figure 2.1 compares the performance of our algorithm versus two benchmarks. Since Algorithm 1 can be viewed as an extension of [1] to non-complete graphs, the first benchmark for comparison is the OMKL algorithm from [1]. There is a small difference between the performance of OMKL and that of our gossiped OMKL although OMKL requires many more inter-node communications. This is despite the fact that OMKL is run over a complete topology, but the gossiped OMKL is executed over the worst-case scenario, i.e., a path topology that includes much smaller number of connections and as a result requires much less communication.

Gossiped OMKL can also be considered as an extension of [2] to multi-kernel learning, which justifies using the single-kernel SGD from [2] as our second benchmark. Gossiped OMKL using three kernels clearly outperforms the single-kernel approach, named Koloskova in Figure 2.1.

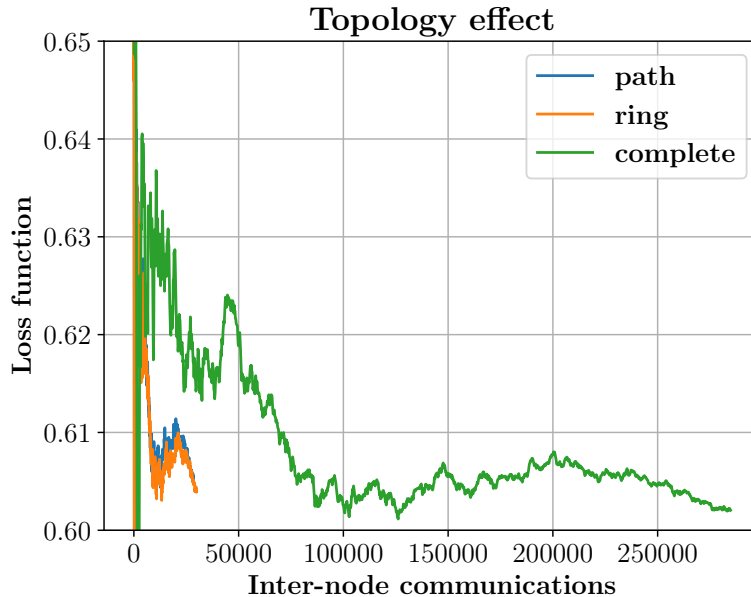


Figure 2.2: The effect of graph topology, without quantization. Average loss function at each iteration, with the Credit-Card dataset. Since the Credit-Card dataset has $n = 30000$ data observations, and $J = 20$ nodes, we have run the algorithm 1500 iterations.

Although not shown in the figure, we have observed that, for values of η and γ chosen in Figure 2.1, or smaller values, the choice of topology does not affect the performance of our

algorithm. To observe the effect of topology on algorithm performance, larger step sizes ($\eta = 0.1$ and $\gamma = 0.09$) are chosen in Figure 2.2. The figure shows simulations using the Credit-Card dataset and without quantization on three different topologies: the complete graph, the ring, and the path. We can observe that more densely connected communication graphs lead to better performance. We have also tested the algorithm for $M \geq 7$ levels of quantization, that satisfies the condition $\delta > 0$ in Equation (2.12), and all of them perform as well as the non-quantized version, with the loss function differences in the order of 10^{-6} .

The results show that our gossiped OMKL algorithm can successfully extend the OMKL algorithm [1] to non-complete graphs and the distributed learning algorithm in [2] to multi-kernel learning.

Chapter 3

Quantized and Asynchronous Federated Learning

3.1 Introduction

In a traditional machine learning (ML) pipeline, data is collected from clients at a central server. Then, a model is trained on the collected data and is deployed for use. This has two major drawbacks: (i) it requires a large amount of storage at a central server, and more importantly, (ii) it raises privacy concerns when collecting sensitive data. Decentralized learning techniques can deal with these concerns [1, 2, 36]. One of the mechanisms to inherently address these drawbacks is federated learning (FL) where clients train local models and send them to the server for aggregation [37]. In FL, client data is used exclusively to train local models, ensuring it never leaves the client side. Instead, the clients send their local models' updates to the server and the server aggregates the updates to create a global model. Including new updates improves the global model's accuracy. FL is gaining momentum in healthcare, finance, and natural language processing, to name a few areas [21, 38].

FL characteristics are different from those of the traditional distributed optimization. First, the data originates from the clients and cannot be shared with the server. Second, clients are heterogeneous, i.e., they have access to different amount of data and operate with different speeds and communication bandwidths [39]. Since the size of ML models is large, and becoming larger with time (particularly with language models) [40], communicating model updates from edge devices with bandwidth constraints is a costly operation.

3.1.1 Related Work

Multiple FL algorithms have been proposed in the literature such as FedSGD [37], FedAvg [41], and FedProx [42], to name a few. In FedSGD, first, each client completes a stochastic gradient descent (SGD) step with its local data and sends the update to the server. Then, the updated local models are averaged at the server to create a global model. Finally, the global averaged model is sent back to clients and the process is repeated. To reduce communication costs, FedAvg allows multiple SGD steps at each client before exchanging the model, proving effective in real-world scenarios. FedProx builds upon FedAvg, introducing a proximal term to account for disparities in local data distributions and device characteristics.

Synchronous vs. Asynchronous Federated Learning

A common assumption is that these methods operate in rounds, leading to synchronous federated learning (FL), where the server waits for a predetermined time to receive updates from all clients. Should a client miss the time window, its update is considered stale and is discarded. While FedProx allows clients to perform different number of local steps, it still requires them to communicate before the round ends. Since FL is designed for massive-scale networks, it is natural for clients to have different update times [43]. As a result, interest has grown in asynchronous FL, which allows the server to update the global model without waiting for all clients. There is no idle time in asynchronous FL and the clients restart

calculating a new update after each transmission. A comparison between synchronous and asynchronous FL is shown in Figure 3.1.

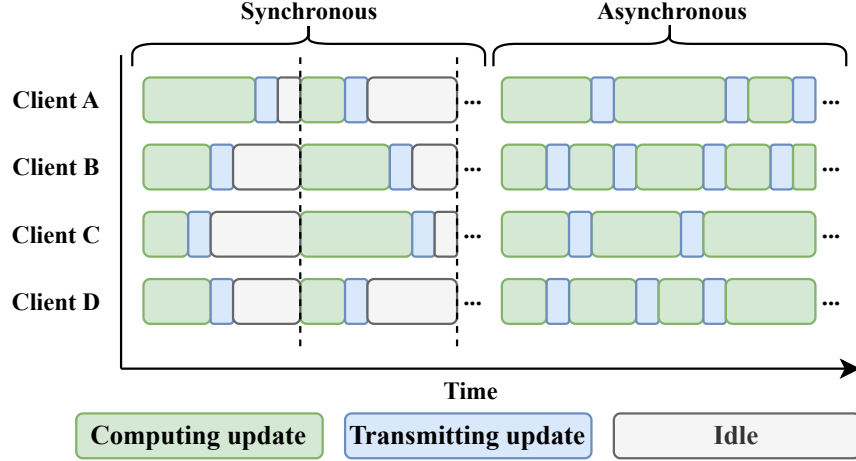


Figure 3.1: Flowchart comparing synchronous and asynchronous FL.

Despite its challenges, like the need to handle stragglers and stale gradients, asynchronous FL is attractive because it eliminates the burden of fitting clients into time slots. This enables slow clients to participate in the training process and enables the use of larger training cohorts [44, 45]. A naive approach to asynchronous FL is to have the server update the global model every time it receives a client update. However, this method is not scalable, as the communication cost of broadcasting the global model grows too much with the number of clients. To address this issue, client updates are buffered at the server before performing a global model update. FedBuff [46, 47], an asynchronous and scalable version of FedAvg, is an example of such an algorithm suitable for heterogeneous clients. It allows multiple local steps at each client and is compatible with privacy-preserving mechanisms. It is also fairer and more efficient compared to synchronous FL methods [45].

Other asynchronous FL works include algorithms that consider time-weighted schemes for stale model aggregation [48, 49]. In another approach, [50] proposes a client uploading policy, which avoids sending updates where the previous and current models are sufficiently similar. However, these solutions do not consider quantization, which is indispensable to reduce the

communication bottleneck in many practical FL systems.

Quantization in Federated Learning

Apart from allowing multiple local steps, quantization can further reduce the communication overhead. There is a vast literature on synchronous FL with quantized communications [4, 51–53]. Quantization can reduce the number of transmitted bits in both directions: (i) the server can quantize the global model prior to sending it to clients, and (ii) clients can quantize the model updates before transmitting them to the server. Moreover, quantizing client updates enhances the privacy guarantees [54]. On the other hand, directly quantizing the models results in error propagation over time. For example, since clients only access the quantized global model, there will be a drift between the global models at the server and clients. This phenomenon is illustrated in Figure 3.2. We observe that a naive use of quantization results in suboptimal loss. This is particularly egregious in the biased quantization case, where FedBuff diverges when a server sends the top 50% of coordinates in absolute value, setting the rest to zero. This is a popular quantizer with a low compression ratio of $1/2$ and the fact that it diverges is devastating.

The use of quantization in asynchronous FL remains unexplored. It is of particular interest to investigate the error produced by the compounded effects of model staleness due to asynchrony and quantization error, since the former is absent in the synchronous FL setting.

Managing Error Propagation

To manage the error propagation while quantizing the model, the server and clients should operate on the same model [18]. Therefore, we define a common model state and keep it at all nodes. The difference between the updated server model and the common model state is quantized and communicated after every server update. Similar ideas exist in other signal processing fields, for example in handling the drifts caused by motion compensation in video coding [55–57]. In quantization theory, the general framework to manage error propagation

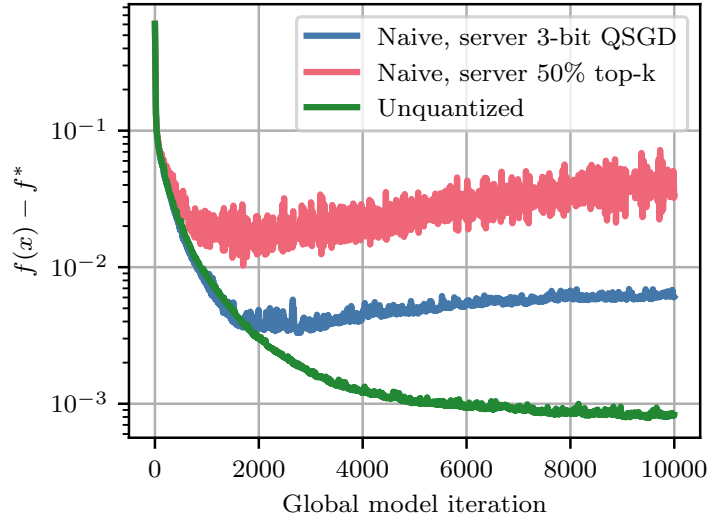


Figure 3.2: Numerical example of the effect of direct quantization (Naive) vs. the unquantized counterpart. No client quantization is performed. We show unbiased (QSGD [3]) and biased (top-k [4]) server quantizer examples. We consider a logistic regression problem with ℓ_2 regularization on the *mushrooms* dataset from LIBSVM [5]. The simulation parameters are: 100 clients, delays following a half-normal distribution, server buffer size of 10, client learning rate of 2, server learning rate of 0.1, and ℓ_2 regularization strength of $1/8124$, where 8124 is the number of samples in the dataset. The y -axis illustrates the difference $f(x) - f^*$, where $f(x)$ is the global model cost at a given iteration, and f^* is the optimal cost.

is discussed under predictive coding [58]. There has been some efforts in managing error propagation in synchronous FL [2, 6, 59–62], but, to the best of our knowledge, this is the first effort in handling the error propagation in asynchronous FL.

3.1.2 Contributions

Motivated by the need to reduce communication overhead and the appeal of asynchronous FL, we propose a new algorithm, called Quantized Asynchronous Federated Learning (QAFEL), which includes a bidirectional quantization scheme for asynchronous FL with buffered aggregation. To address the error propagation, we introduce a common hidden-state by aggregating all communicated messages as shown in Figure 3.3. The server quantizes and broadcasts

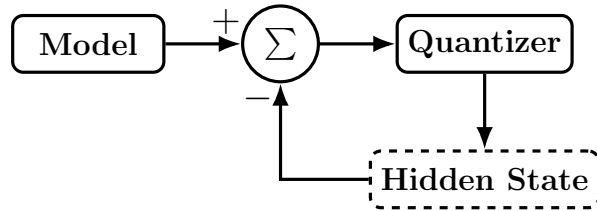


Figure 3.3: Block diagram for updating the hidden-state.

the difference between the hidden-state and its updated model. Similarly, clients quantize the difference between their updated model and the corresponding hidden-state version. Using the mechanism in Figure 3.3, QAFEL avoids error propagation and is scalable as it only needs to track one hidden-state. Furthermore, it is a privacy-aware system since it does not track client states. We also investigate the QAFEL’s compound error produced by staleness and quantization to gain insights not present in separate analysis of the staleness and quantization effects. Table 3.1 compares the characteristics of our work with respect to others.

The main contributions of the chapter are:

1. We introduce QAFEL, an asynchronous FL algorithm with multiple local steps and limited number of communication bits that avoids error propagation using hidden-state

Table 3.1: Related work comparison. The number of algorithm rounds is denoted by T .

Algorithm	Work	Characteristics				Analysis		
		Asynchronous	Buffered Aggregation	Quantized Communication	Mitigates Error Propagation	Unbounded Gradient	Arbitrary Client Update Distribution	Non-convex objective $\mathcal{O}(1/\sqrt{T})$ rate
FedAvg	[37]	✗	✓	✗	✗	✗	-	✗
	[63]	✗	✓	✗	✗	✓	-	✓
MCM	[62]	✗	✓	✓	✓	✓	-	✗
FedAsync	[64]	✓	✗	✗	✗	✗	✗	✓
FedBuff	[46]	✓	✓	✗	✗	✗	✗	✓
	[47]	✓	✓	✗	✗	✓	✗	✓
QAFEL	This work	✓	✓	✓	✓	✓	✓	✓

updates.

2. We analytically prove that quantization using QAFEL does not affect the complexity order. More precisely, QAFEL’s convergence rate achieves the optimal $\mathcal{O}(1/\sqrt{T})$ complexity order for non-convex objectives [65] even without assuming uniform client arrival.
3. We present FedBuff as a special case of QAFEL and fix an error from the original FedBuff manuscript. We analytically show that asymptotically the convergence rate of QAFEL is the same as that of FedBuff, without assuming bounded gradients.
4. We show that the cross-term error caused by staleness and quantization is of smaller order than the errors introduced by each of these factors alone and does not affect the complexity order.

Finally, we validate our theoretical results through experimental evaluation standard benchmarks for FL [66, 67].

The rest of the chapter is organized as follows. Section 3.2 presents the system model as well as a description of our proposed algorithm. Section 3.3 presents the analysis of our algorithm and discusses the derived convergence guarantees. The auxiliary proofs are relegated to Appendix A.1. Section 3.4 includes experimental results and Section 3.5 concludes the chapter.

3.2 System Model

In this section, first, we describe our general quantizer model and then present QAFeL in detail.

3.2.1 General Quantizer Model

A quantizer is composed of an encoder, which receives blocks of information and outputs blocks of bits, and a decoder, which receives blocks of bits and reconstructs blocks of information. If the encoder and decoder are designed carefully, the mean square error between the original and the reconstructed symbols is small, for example smaller than the norm of the original symbols. Obviously, the mean square error grows as the compression ratio increases. Let us denote the combination of the encoder and decoder as a single function Q which is in agreement with the following standard definition of a quantizer in the FL literature [4, 52, 61].

DEFINITION 3.1. *A quantizer $Q : \mathbb{R}^d \rightarrow \mathbb{R}^d$ (which is a combination of an encoder and a decoder) with a compression parameter $\delta \in (0, 1]$ is a (possibly random) function that satisfies*

$$\mathbb{E}_Q [\|x - Q(x)\|^2] \leq (1 - \delta) \|x\|^2, \quad (3.1)$$

where $\mathbb{E}_Q$ is the expectation with respect to the possible internal randomness of the quantizer.

3.2.2 Proposed Algorithm

QAFeL is comprised of three processes that run concurrently: QAFeL-server, QAFeL-client and QAFeL-client-background. The three processes are depicted in Figure 3.4 as a block diagram. The algorithm works as follows. First, clients initialize the hidden-states with $\hat{x}^0$, which is the same as the initial server model, x^0 . Then, the server waits for a client update.

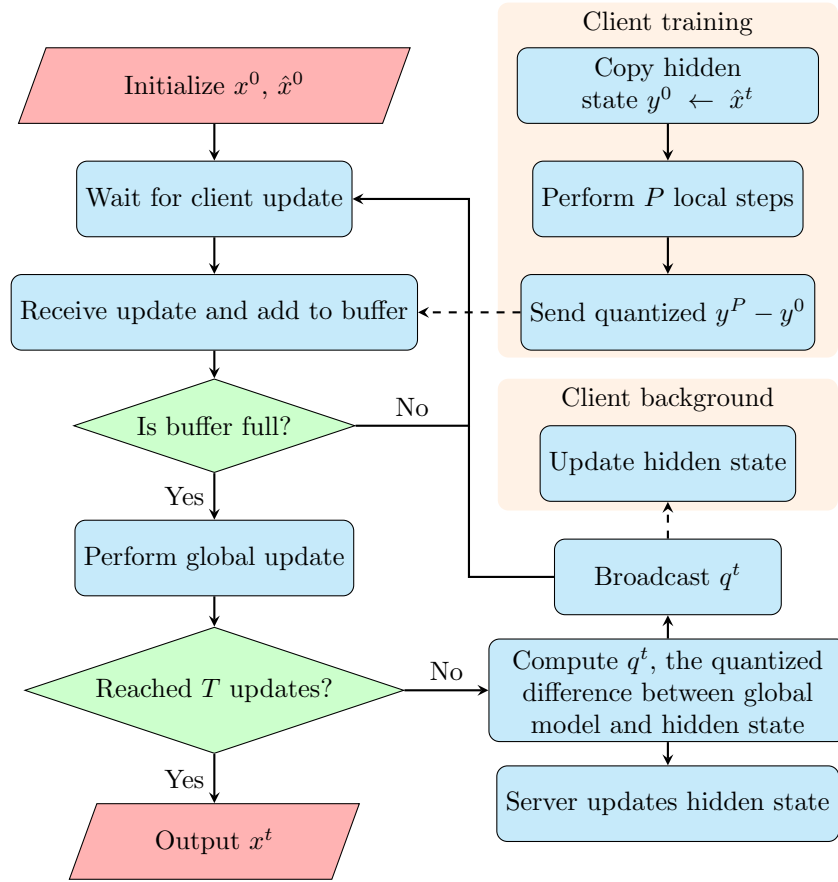


Figure 3.4: QAFEL block diagram. Shaded parts occur on the client side. Dashed lines indicate communication between the clients and the server.

To start training, the client copies the locally stored hidden-state into a variable $y_0 \leftarrow \hat{x}^t$, and performs P local model update steps of the type

$$y_{p+1} = y_p - \eta_\ell \cdot g(y_p), \quad (3.2)$$

where $g(y_p)$ is a noisy, unbiased estimator of the gradient at y_p based on the local dataset and η_ℓ is a local step-size. Then, the client sends the quantized update $Q_c(y_0 - y_P)$ to the server, where Q_c is the client's quantizer. The server adds the received updates to its buffer. We denote the k -th update in the buffer as Δ_k . The server keeps receiving updates until its buffer is full, i.e., the server has received K updates. Then, the server updates the global model by averaging the received updates:

$$x^{t+1} = x^t - \eta_g \bar{\Delta}^t, \quad \bar{\Delta}^t := \frac{1}{K} \sum_{k=0}^{K-1} \Delta_k. \quad (3.3)$$

When the pre-defined number of iterations T is reached, the server outputs the model and the training stops. Otherwise, the server computes the quantized difference between its updated model and the hidden-state,

$$q^t = Q_s(x^{t+1} - \hat{x}^t), \quad (3.4)$$

where Q_s is the server's quantizer. The server broadcasts q^t to all clients. Finally, the clients and the server update their copies of the hidden-state using the same equation, $\hat{x}^{t+1} = \hat{x}^t + q^t$.

For further details on the algorithm design see Appendix A.2, where we have included the pseudocode for all components of QAFEL. The highlighted lines in the pseudocode represent the novel hidden-state mechanism.

3.3 Algorithm Analysis

In this section, we analyze the convergence of QAFEL. First, we formulate the non-convex optimization problem that QAFEL solves. Then, we present convergence upper bounds for QAFEL.

3.3.1 Problem Formulation

We consider the weighted FL problem setting to find a model $x \in \mathbb{R}^d$ such that

$$\min_{x \in \mathbb{R}^d} f(x) := \sum_{n=1}^N w_n F_n(x), \quad (3.5)$$

where F_n is Client n 's loss function, w_n is the client's weight, and N is the total number of clients. Note that each function F_n implicitly depends on the local data at Client n . The weights w_n can be any arbitrary set of numbers chosen to weigh the importance of each client or to strategically differentiate different clients. For example, one can assign a larger weight to clients that have a larger dataset. Obviously, this setting is more general than the equal-weight setting, which is also covered in our setup by choosing $w_n = 1/N$. Alternatively, one can enforce a uniform request for updates from the server, a technique employed in FedBuff. While enforcing uniform client participation guarantees that all weights are identical, it slows down the fastest clients' update frequency. Our analysis considers the case that f is non-convex, which is prevalent in FL applications.

Let us make the following standard assumptions:

ASSUMPTION 3.2. *Unbiased Stochastic Gradients Assumption:* We have an unbiased stochastic estimator $g_n(x)$ of the true gradient $\nabla F_n(x)$, i.e., $\mathbb{E}[g_n(x)] = \nabla F_n(x)$ for all x , at every client.

ASSUMPTION 3.3. *Bounded Local Variance Assumption: At every client,*

$$\mathbb{E} [\|g_n(x) - \nabla F_n(x)\|^2] \leq \sigma_\ell^2, \quad \forall x \in \mathbb{R}^d. \quad (3.6)$$

ASSUMPTION 3.4. *L-smoothness Assumption: At every client, F_n is L-smooth, i.e., F_n is differentiable and its gradient is Lipschitz continuous. Thus, ∇F_n satisfies*

$$\|\nabla F_n(x) - \nabla F_n(x')\| \leq L \|x - x'\|, \quad \forall x, x' \in \mathbb{R}^d. \quad (3.7)$$

ASSUMPTION 3.5. *Bounded Client Heterogeneity Assumption: The gradients ∇F_n at every client satisfy*

$$\|\nabla f(x) - \nabla F_n(x)\|^2 \leq B, \quad \forall x \in \mathbb{R}^d. \quad (3.8)$$

In the proof, we only use bounded heterogeneity at each buffer set $\mathcal{S}_t$, i.e., each set of clients that fills the buffer. Therefore, we can relax the assumption to

$$\frac{1}{K} \sum_{k \in \mathcal{S}_t} \|\nabla f(x) - \nabla F_k(x)\|^2 \leq B, \quad \forall x \in \mathbb{R}^d, \quad (3.9)$$

and the results still hold. However, for the clearness of exposition, we will use the first version in the proof.

ASSUMPTION 3.6. *Lower Bounded Objective Assumption: There exists a lower bound $f^* \leq f(x)$ for all $x \in \mathbb{R}^d$.*

In an asynchronous setting, different nodes may operate on different versions of the model causing staleness [46]. The definition of the staleness for QAFEL and a bounded staleness assumption similar to that of FedBuff are presented below.

DEFINITION. For Client n , at server step t , the difference between the current hidden-state $\hat{x}^t$ and the hidden-state version that was used to start the current local training is called staleness, and is denoted $\tau_n(t)$.

ASSUMPTION 3.7. *Bounded Staleness Assumption:* At every server step t , and for each client n , the staleness $\tau_n(t)$ is less than or equal to a maximum allowed staleness, $\tau_{\max,K}$, where K is the server buffer size.

Note that as the buffer size increases, the server updates less frequently, which reduces the number of server steps between when a client starts training and when its updates are applied at the server. If Assumption 3.7 is met for any $K > 1$, the maximum delay, $\tau_{\max,K}$, is at most $\lceil \tau_{\max,1}/K \rceil$; this is proven in [46, Appendix A].

In most ML applications, the objective function is highly non-convex [68]. With such functions, we cannot guarantee the existence of a global minimum. Instead, as is standard in non-convex optimization, our goal is to find a *first-order ε -stationary point*, that is, $x \in \mathbb{R}^d$ such that $\|\nabla f(x)\|^2 \leq \varepsilon$. With such a goal in mind, we study the ergodic squared norm of the gradient after T iterations:

$$\frac{1}{T} \sum_{t=0}^{T-1} \mathbb{E} \left[\|\nabla f(x^t)\|^2 \right]. \quad (3.10)$$

We call this quantity the *ergodic convergence rate* of our algorithm. Upper bounding the rate by ε ensures that our algorithm finds a first-order ε -stationary point in expectation.

Throughout the chapter, we use the notation from Table 3.2.

3.3.2 Convergence Analysis for QAFEL

In this section, we present our main result, as well as a corollary that analyzes the convergence rate in detail.

Table 3.2: Summary of notation.

$x^t, \hat{x}^t$	server, shared hidden-state at time t
L	L -smoothness constant of the loss function
P, p	number, index of local steps at client
K, k	number, index of clients at the buffer
N, n	number, index of total clients
η_g, η_ℓ	server, client learning rates
Q_s, Q_c	server, client quantizers
$\bar{\Delta}^t, \Delta_k^t$	server, client k 's update at time t
$\mathcal{S}^t$	set of client indices at the buffer at time t
$y_{k,p}^t$	local state at client k , during local step p at time t
$\pm$	plus and minus a quantity, i.e., $a \pm b = a + b - b = a$

THEOREM 3.8. *Consider the optimization problem in Equation (3.5) satisfying Assumptions 3.2 to 3.5 and 3.7. Then, QAFEL's iterations satisfy*

$$\begin{aligned}
\frac{1}{T} \sum_{t=0}^{T-1} \mathbb{E} \left[\|\nabla f(x^t)\|^2 \right] &\leq 4 \frac{f(0) - f(x^T)}{T\eta_g P \eta_\ell} \\
&+ 8L^2 \frac{\eta_g^2 P \eta_\ell^2 (2 - \delta_c) (\tau_{\max, K}^2 + \frac{8}{\delta_s^2}) \sigma_\ell^2}{K} \\
&+ 80L^2 P^2 \eta_\ell^2 (\sigma_\ell^2 + B) \\
&+ 2L \frac{\eta_g \eta_\ell (2 - \delta_c) \sigma_\ell^2}{K},
\end{aligned} \tag{3.11}$$

as long as

$$\eta_g^2 \left(\tau_{\max, K}^2 + \frac{8}{\delta_s^2} \right) + \left(1 + \frac{1 - \delta_s}{K} \right) \eta_\ell \eta_g L \leq \frac{1}{P} \tag{3.12}$$

and

$$\eta_\ell^2 \leq \frac{1}{80L^2 P^2 \tau_{\max, K}}, \quad \eta_\ell \leq \frac{1}{4L(P+1)}. \tag{3.13}$$

Proof. QAFEL's iterations are

$$x^{t+1} = x^t - \eta_g \bar{\Delta}^t, \quad (3.14)$$

$$\hat{x}^{t+1} = \hat{x}^t + Q_s(x^t - \hat{x}^t). \quad (3.15)$$

A special case of QAFEL is when the server quantizer Q_s is the identity, for which the hidden-state is the same as the server's state. This special case covers FedBuff. In other words, our results are applicable to FedBuff as a special case.

Assumption 3.4 implies

$$f(y) \leq f(x) + \langle \nabla f(x), y - x \rangle + \frac{L}{2} \|y - x\|^2, \quad (3.16)$$

for all $x, y \in \mathbb{R}^d$. This follows immediately from [69, Lemma 1.2.3]. Plugging Equation (3.14) into Equation (3.16) yields

$$\begin{aligned} f(x^{t+1}) &\leq f(x^t) + \langle \nabla f(x^t), x^{t+1} - x^t \rangle + \frac{L}{2} \|x^{t+1} - x^t\|^2 \\ &= f(x^t) - \eta_g \langle \nabla f(x^t), \bar{\Delta}^t \rangle + \eta_g^2 \frac{L}{2} \|\bar{\Delta}^t\|^2. \end{aligned} \quad (3.17)$$

Using Lemma A.2 and Corollary A.4 from Appendix A.1, we can add the terms for $t = 0, \dots, T-1$ and obtain

$$\begin{aligned} f(x^T) &\leq f(0) - \eta_g \frac{\eta_\ell P}{4} \sum_{t=0}^{T-1} \mathbb{E} \left[\|\nabla f(x^t)\|^2 \right] \\ &\quad + \eta_g \frac{\eta_\ell P}{2} 4L^2 \sum_{t=0}^{T-1} \eta_g^2 (\tau_{\max, K}^2 + \frac{8}{\delta_s^2}) \frac{P\eta_\ell^2}{K} (2 - \delta_c) \sigma_\ell^2 \\ &\quad + \eta_g \frac{\eta_\ell P}{2} 4L^2 \sum_{t=0}^{T-1} 10P^2 \eta_\ell^2 (\sigma_\ell^2 + B) \\ &\quad + \sum_{t=0}^{T-1} \eta_g^2 \frac{L}{2} \frac{P\eta_\ell^2}{K} (2 - \delta_c) \sigma_\ell^2, \end{aligned} \quad (3.18)$$

as long as

$$P\eta_g^2(\tau_{\max,K}^2 + \frac{8}{\delta_s^2}) + (1 + \frac{1 - \delta_s}{K})P\eta_\ell\eta_gL \leq 1 \quad (3.19)$$

and

$$40L^2P^2\eta_\ell^2\tau_{\max,K} \leq \frac{1}{2}, \quad \eta_\ell \leq \frac{1}{4L(P+1)}. \quad (3.20)$$

The theorem statement is obtained by re-arranging Equation (3.18). $\square$

COROLLARY 3.9 (QAFEL's Order of Complexity). *Consider the optimization problem Equation (3.5) satisfying Assumptions 3.2 to 3.7 and define $F^* := f(x^0) - f^*$, where f^* minimizes f . Choosing $\eta_\ell = \mathcal{O}(K^{-1}P^{-1/2}T^{-1/3})$, $\eta_g = \mathcal{O}(KT^{-1/6})$, and a large enough T , QAFEL's iterations satisfy*

$$\begin{aligned} \frac{1}{T} \sum_{t=0}^{T-1} \mathbb{E} \left[\|\nabla f(x^t)\|^2 \right] &\leq \underbrace{\mathcal{O}\left(\frac{F^*}{\sqrt{PT}}\right) + \mathcal{O}\left(\frac{L^2\sigma_\ell^2(2 - \delta_c)}{K\sqrt{PT}}\right)}_{\text{main error terms}} \\ &+ \underbrace{\mathcal{O}\left(\frac{L^2P(\sigma_\ell^2 + B)}{K^2T^{2/3}}\right)}_{\text{heterogeneity term}} \\ &+ \underbrace{\mathcal{O}\left(\frac{L(2 - \delta_c)(\tau_{\max,K}^2 + \frac{1}{\delta_s^2})\sigma_\ell^2}{KT}\right)}_{\text{staleness term}}. \end{aligned} \quad (3.21)$$

Note that the terms with $(2 - \delta_c)$ could have been absorbed by the $\mathcal{O}(\cdot)$ notation, but we have purposefully kept them to highlight the minimal effect of the client quantization.

Corollary 3.9 yields several insights:

- The main error term is $\mathcal{O}(1/\sqrt{T})$, which is the optimal ergodic convergence rate of SGD for non-convex objectives [65].

- The effect of the client quantizer, controlled by δ_c , is relatively small compared to the main error term, since it is divided by the buffer size. Intuitively, we average K client updates at the buffer and it is reasonable that the quantization error order is divided by K .
- Our theory corroborates the well-established observation in the context of synchronous and unquantized scenarios: increasing the number of local steps, P , leads to a faster convergence when the model is far from a good solution, but introduces more drift [70]. Such a drift is also exacerbated by a higher heterogeneity (higher B). This is reflected in the heterogeneity error term, as a larger P is beneficial as long as the main error terms dominate. Experiments that showcase the effect of the client drift are presented in Section 3.4.1.
- The effect of the server quantizer is a term of order $\mathcal{O}(1/T)$, which is negligible for even a moderate number of server steps T .
- The error added by staleness, controlled by $\tau_{\max,K}$, and the cross-term error between staleness and quantization are of order $\mathcal{O}(1/T)$, which are also negligible compared to the $\mathcal{O}(1/\sqrt{T})$ terms.

As a special case, taking the limit $\delta_c, \delta_s \rightarrow 1$ provides the complexity order of FedBuff, i.e., QAFEL without any quantization scheme, as follows:

$$\begin{aligned}
\frac{1}{T} \sum_{t=0}^{T-1} \mathbb{E} \left[\|\nabla f(x^t)\|^2 \right] &\leq \underbrace{\mathcal{O} \left(\frac{F^*}{\sqrt{PT}} \right) + \mathcal{O} \left(\frac{L^2 \sigma_\ell^2}{K \sqrt{PT}} \right)}_{\text{main error terms}} \\
&\quad + \underbrace{\mathcal{O} \left(\frac{L^2 P (\sigma_\ell^2 + B)}{K^2 T^{2/3}} \right)}_{\text{heterogeneity term}} \\
&\quad + \underbrace{\mathcal{O} \left(\frac{L (\tau_{\max,K}^2 + 1) \sigma_\ell^2}{KT} \right)}_{\text{staleness term}}.
\end{aligned} \tag{3.22}$$

Note that our assumptions are weaker than those in FedBuff. The above corrected convergence rate¹ includes a higher-order ($T^{-2/3}$) heterogeneity term because of our weaker assumptions.

3.4 Results and Discussion

We present results with two methods of compression. The first method, QSGD, is an unbiased stochastic quantizer that uses a fixed number of bits per dimension [3]. A higher bit-rate results in a more precise quantizer. The second method, top-k, is biased and only sends the k largest coordinates in absolute value, and does not send the rest [60]. The receiver sets the unsent missing coordinates to zero.

3.4.1 Logistic Regression Experiments

To illustrate the fact that QAFEL addresses the error propagation caused by naive direct quantization, we present a set of experiments on a standard logistic regression task and run both algorithms for a comparison.

We consider a logistic regression problem with ℓ_2 regularization on the *mushrooms* dataset from LIBSVM [5]. The simulation parameters are: 100 clients, delays following a half-normal distribution, server buffer size of 10, client learning rate of 2, server learning rate of 0.1, and ℓ_2 regularization strength of $1/8124$, where 8124 is the number of samples in the dataset.

Figure 3.5 illustrates both the unbiased server quantizer case (Figure 3.5a, QSGD quantization) and the more drastic biased server quantization case (Figure 3.5b, top-k quantization). In Figure 3.5a, a quantizer with 3 bits is used for both direct quantization and QAFEL. While QAFEL provides results very close to that of the unquantized case, the direct quantization approach does not converge. Similarly, in the case of top-k quantization in Figure 3.5b,

¹The rate that appears in the original AISTATS 2022 paper [46] has a minor error resulted from inaccuracy in Eq. (20) of that paper.

QAFeL elegantly manages the error propagation while a direct quantization suffers from error propagation. More specifically, with direct quantization, the algorithm diverges when the server sends the top 50% of coordinates in absolute value. Meanwhile, with QAFeL, a server that sends only the top 1% of coordinates in absolute value converges.

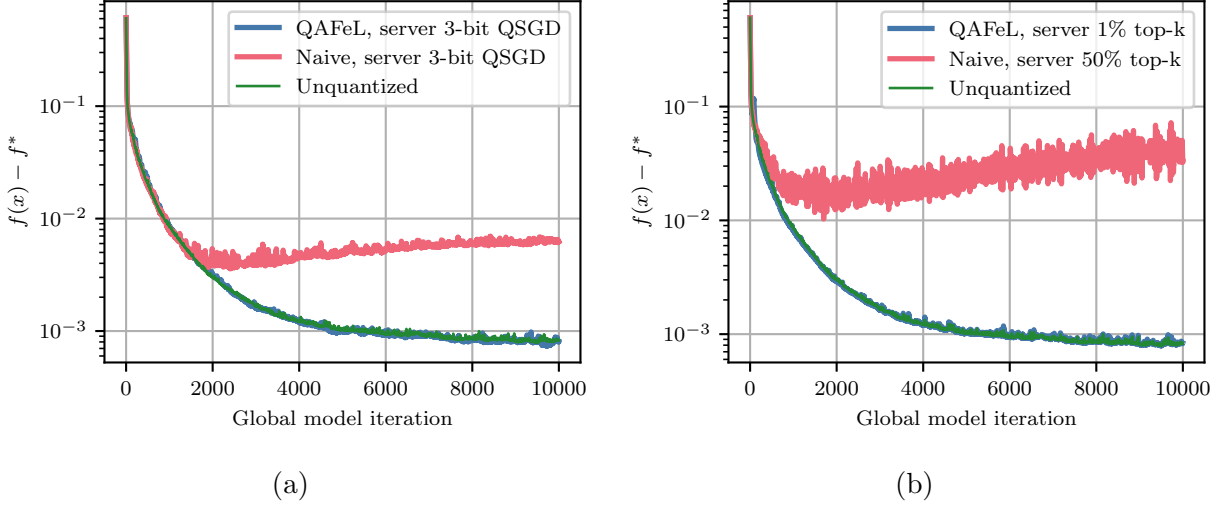


Figure 3.5: Numerical example of the effect of naive quantization vs. our proposed algorithm (QAFeL). No client quantization is performed. The y -axis illustrates the difference $f(x) - f^*$, where $f(x)$ is the global model cost at a given iteration, and f^* is the optimal cost. Subfigure (a) illustrates the unbiased server quantizer case. Subfigure (b) illustrates the biased server quantizer case.

Figure 3.6 presents a numerical illustration of the effect of the local step values, without quantization. For an analogous figure in the synchronous and unquantized case, see [71, Figure 6]. Observe that a larger amount of local steps implies faster convergence, but higher suboptimality. The noise around the optimum is due to the randomness introduced by the delays. Also, note that the noise does not increase for smaller local steps. Such an appearance is because of the y -axis logarithmic scale.

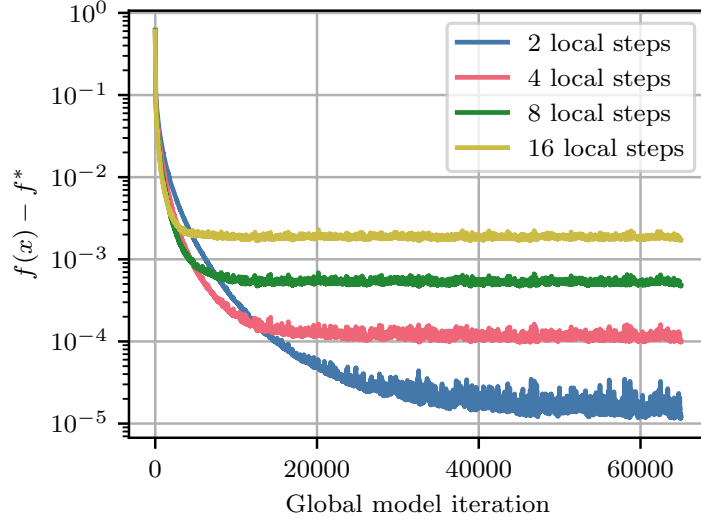


Figure 3.6: Numerical example of the effect of local steps. The y -axis illustrates the difference $f(x) - f^*$, where $f(x)$ is the global model cost at a given iteration, and f^* is the optimal cost. QAFeL is run without quantization and a varying quantity of local steps. More local steps lead to larger client drift and faster convergence, but to a less optimal solution.

3.4.2 Neural Network Experiments

Experiment Setup

We have implemented QAFeL using FL Simulator (FLSim) [72], a library written in PyTorch [73]. To verify our theoretical results, we consider two image classification tasks using the CIFAR-10 [67] and CelebA [74] datasets, as well as a next-word prediction task using the Shakespeare dataset [37].

The CIFAR-10 dataset consists of labeled images pertaining to one of 10 classes. As is standard in image classification, we first normalize the images using the dataset mean and standard deviation such that each color channel has mean 0 and standard deviation 1. To simulate a FL scenario, we follow the approach of [75] to synthesize non-identical clients. Specifically, we split the dataset into 5,000 clients where each client has a number of samples that follows a symmetric Dirichlet distribution with parameter 0.1. This parameter controls the correlation of the data distribution, i.e., how close the distribution is to an i.i.d. random

process. When the parameter is 0, each client holds data from only one class chosen at random. On the other hand, when it tends to ∞ , all clients have identical distributions.

The CelebA dataset contains labeled images of celebrities. We detect whether they are smiling or not. We resize and crop the images to 32×32 pixels before normalizing them to have 0.5 mean and 0.5 standard deviation, as done in previous work [46]. We also use the default data-heterogeneous client partition from LEAF.

The Shakespeare dataset contains Shakespeare’s plays, separated by play and character. This was proposed as a good FL scenario in [37] as different characters have different word distributions. We use LEAF’s default client partition for this task as well.

For both image datasets, we train the image classification model used in LEAF, that is, a four-layer convolutional neural network (CNN) classifier, slightly modified by replacing batch normalization layers with group normalization layers [76] following the approach of [77]. We use a 0.1 dropout regularization rate, stride of 1, and padding of 2. We compare FedBuff and QAFEL with the hyperparameter selection from [46], which we re-state in Table 3.3 for completeness. Note that we have used server momentum β in two of our experiment settings, which we have found to help convergence in practice.

In our image experiments, we simulate QAFEL training until we reach a pre-specified target evaluation accuracy, which is 60% for CIFAR-10 and 90% for CelebA. These quantities are the same as previous benchmarks [46, 66], i.e., the pre-defined architectures from our LEAF benchmark converge to these percentages on the centralized setting.

For the Shakespeare dataset, we use the 2-layer stacked LSTM model proposed in [37, 66], as well as their proposed target accuracy of 54%. We use Bayesian optimization [78] to obtain the hyperparameters reported in Table 3.3.

We simulate client arrival times and training durations based on Meta’s production FL

Table 3.3: Chosen hyperparameters

	FedBuff and QAFEL	FedAsync
CelebA	$\eta_\ell = 4.7 \cdot 10^{-6}$	$\eta_\ell = 5.7$
	$\eta_g = 1.0 \cdot 10^3$	$\eta_g = 2.8 \cdot 10^{-3}$
	$\beta = 3.0 \cdot 10^{-1}$	$\beta = 0$
CIFAR-10	$\eta_\ell = 1.95 \cdot 10^{-4}$	$\eta_\ell = 1.0 \cdot 10^2$
	$\eta_g = 4.09 \cdot 10^1$	$\eta_g = 6.4 \cdot 10^{-5}$
	$\beta = 0$	$\beta = 0$
Shakespeare	$\eta_\ell = 1.17$	$\eta_\ell = 1.7 \cdot 10^1$
	$\eta_g = 2.64 \cdot 10^{-1}$	$\eta_g = 2.03 \cdot 10^{-1}$
	$\beta = 0$	$\beta = 5.0 \cdot 10^{-1}$

system [46, Appendix C]. In this model, clients arrive at a constant rate and their training durations are sampled from a half-normal distribution denoted as Y , where $Y = |X|$ and $X \sim \mathcal{N}(0, 1)$. To achieve varying concurrencies (the maximum number of users training in parallel) of 100, 500, and 1,000 users, we adjust the arrival rates of clients to 125, 627, and 1,253 clients per unit of time, respectively. These rates are determined based on the expected value $\sqrt{2/\pi}$ of the half-normal distribution Y . As done in previous work [46, 64], we use weight scaling to penalize staleness. More precisely, an update with staleness τ is scaled down by a $1/\sqrt{1 + \tau}$ factor. For all scenarios, we use a batch size of 32 and a buffer size $K = 10$.

To justify our choice of 4-bit QSGD quantizer for server and client in our concurrency experiments, we perform a set of experiments displayed in Table 3.4, where we assume a constant client arrival rate of 100 clients per unit of time, without performing any weight scaling. These experiments show that using a 4-bit QSGD at both client and server does not significantly alter the number of client uploads to reach a target accuracy, yet approximately provides an eight-fold reduction in the number of communicated bits.

Experimental Results

We conduct experiments three times and provide the mean and standard deviation of the results. Alongside the conventional metric of comparing synchronous and asynchronous FL methods, which is the number of client trips, we also report the number of bytes sent per message to emphasize QAFeL’s advantages.

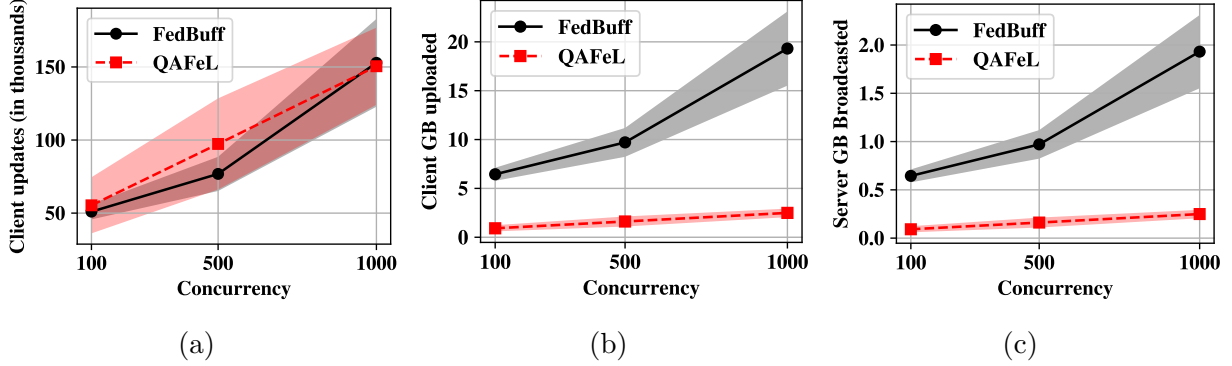


Figure 3.7: Comparison of communication metrics for QAFeL and FedBuff across varying concurrency levels (clients training in parallel) to reach 60% target accuracy on the CIFAR-10 dataset. QAFeL employs 4-bit QSGD quantization at both the server and client: (a) the number of client updates in thousands, (b) the total GB uploaded by the clients, and (c) the GB broadcasted by the server.

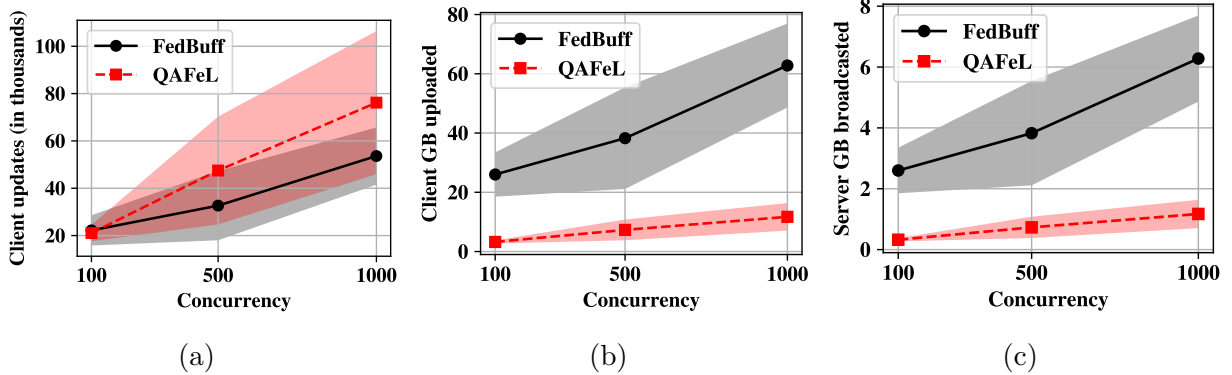


Figure 3.8: Comparison of communication metrics for QAFeL and FedBuff across varying concurrency levels (clients training in parallel) to reach 90% target accuracy on the CelebA dataset. QAFeL employs 4-bit QSGD quantization at both the server and client: (a) the number of client updates in thousands, (b) the total GB uploaded by the clients, and (c) the GB broadcasted by the server.

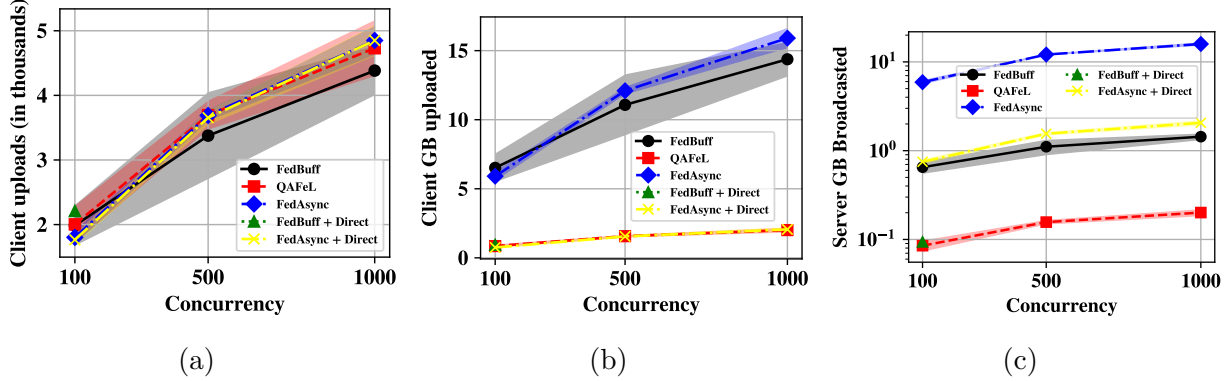


Figure 3.9: Comparison of communication metrics for QAFeL, FedBuff, FedAsync, FedBuff with direct quantization, and FedAsync with direct quantization across varying concurrency levels (clients training in parallel) to reach 54% target accuracy on the Shakespeare dataset. QAFeL employs 4-bit QSGD quantization at both the server and client: (a) the number of client updates in thousands, (b) the total GB uploaded by the clients, and (c) the GB broadcasted by the server. FedBuff with direct quantization only has one data-point because for concurrencies 500 and 1000 the algorithm diverged for at least one of the three seeds.

Figures 3.7 and 3.8 illustrate the benefits of QAFeL for various concurrency values over CIFAR-10 and CelebA datasets, respectively. As shown in Figures 3.7b and 3.8b, for all concurrencies, QAFeL clients use less uploaded bytes compared to FedBuff. For example, reduction factors of 5.2 to 8 for the CelebA dataset and 6 to 7.7 for the CIFAR-10 dataset are observed. Similar reductions for the number of broadcasted bytes by the server are seen in Figures 3.7c and 3.8c. Note that the total number of bytes includes the extra client updates that occur by adding quantization. Notably, the number of client updates remains similar for QAFeL and FedBuff for both datasets. For example, the maximum increase is about 1.5 times in the CelebA case in Figure 3.8a. For both tasks, a 4-bit QSGD quantizer [3] was used in both upload and download.

Figure 3.9 illustrates similar results by comparing QAFeL with FedBuff, FedAsync [64], and FedBuff with direct quantization (4-bit QSGD for both client and server quantizers). The latter is not plotted for concurrencies 500 and 1000 as it diverged for at least one of the three experiments that we average over. Figure 3.10 shows the results for FedBuff with direct quantization at concurrency 500. The figure clearly shows that FedBuff with direct

quantization does not converge for the random seed 0. This is consistent with the theory and

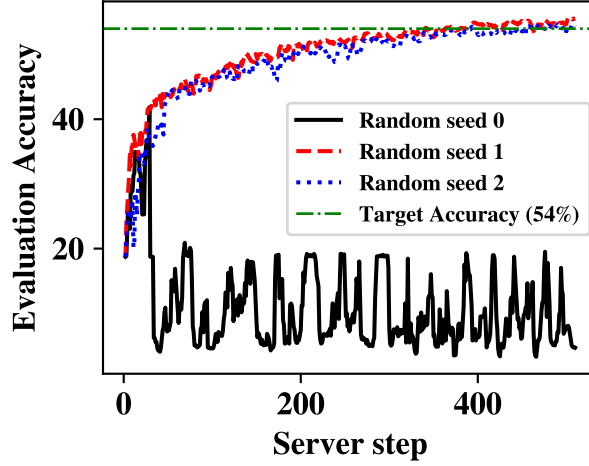


Figure 3.10: Evaluation curves for the Shakespeare experiments with concurrency 500 and FedBuff with direct 4-bit quantization for both client and server. We observe that one of the three runs diverges.

the results shown in the logistic regression experiments of Figure 3.2, which also showed a divergence for the direct quantization case. Moreover, Figure 3.9 also corroborates the results shown in Figures 3.7 and 3.8, as QAFeL performs markedly better than existing algorithms in terms of both uploaded and broadcasted bytes and maintains approximately the same amount of client uploads. Note that FedAsync is not massively scalable as it does not have a buffer to accumulate client updates, which is reflected in the number of broadcasted bytes. With the three random seeds that we picked, FedAsync with direct quantization performs close to the unquantized FedAsync in number of client uploads. However, note that using FedAsync, the number of server broadcasts is ten times more than those of FedBuff and QAFeL, due to the lack of buffer. Also, even with quantization, FedAsync still broadcasts more bytes compared to FedBuff without quantization and an order of magnitude more than QAFeL. If one were to combine a hidden-state scheme with FedAsync, this would require more computations at the client side, as they would update their hidden states more often due to the high frequency of server updates.

Results for a simple scenario with client arrival rate of 100 and different number of quantiza-

tion bits are presented in Table 3.4. Using fewer quantization bits at the server always leads

Table 3.4: Communication metrics of QAFeL to reach the CelebA target validation accuracy (90%) with different QSGD combinations.

	Algorithm	Uploads	kB/upload	kB/download
	(client, server) QSGD bits	(thousands)		
QAFeL	(8,8)	27.6 ± 4.82	29.924	29.924
	(8,4)	29.8 ± 2.76	29.924	15.380
	(8,2)	35.7 ± 14.5	29.924	8.108
	(4,8)	39.7 ± 3.32	15.380	29.924
	(4,4)	27.8 ± 10.4	15.380	15.380
	(4,2)	37.7 ± 9.10	15.380	8.108
	(2,8)	58.6 ± 7.16	8.108	29.924
	(2,4)	74.6 ± 35.7	8.108	15.380
	(2,2)	91.9 ± 34.4	8.108	8.108
	FedBuff	26.1 ± 6.7	117.128	117.128

to fewer total downloaded bytes. However, using less precision at the client-side, for example reducing the number of quantization bits from 4 to 2 while keeping the server’s quantizer at 2 bits per sample, may result in more total uploaded bytes. Moreover, Table 3.4 shows an increase in the number of uploads by switching from 4 to 2 bits per sample at the client-side resulting in similar total upload bytes. This situation highlights a trade-off between the level of quantization and how quickly the system converges.

In simpler terms, while reducing the compression ratio means transmitting fewer bytes, it may require more messages to achieve the desired accuracy. Therefore, it is important to find the right trade-off between the two factors. Nevertheless, QAFeL generally uses fewer total communication bits compared to FedBuff. Also, our simulation results indicate that the best compression ratio depends on the chosen quantization method.

3.5 Conclusion

We propose a practical, scalable, and communication-efficient FL algorithm that allows client asynchrony and quantized communications. Using a hidden-state scheme, we avoid error propagation. We provide optimal ergodic convergence rates for SGD on non-convex objectives, without assuming bounded gradients or uniform client arrival. We also show that the cross-term error between staleness and quantization is negligible compared to each of the error terms. These are achieved without assuming uniform client arrival or imposing restrictions on step-size choices. The presented theoretical analysis can guide the design of quantizers to achieve a specific convergence rate and has been corroborated empirically. A common application is the design of FL systems with bandwidth constraints. Our simulation results show that the effects of the client quantization are larger than those of the server quantization.

Chapter 4

Communication Compression for Distributed Learning with Aggregate and Server-Guided Feedback

4.1 Introduction

In distributed learning, a central server coordinates the training of a global model using data distributed across multiple clients. The objective is to minimize a global loss function f , which is the average of the local loss functions of the clients

$$f(x) = \frac{1}{N} \sum_{n=1}^N f_n(x), \tag{4.1}$$

where $x \in \mathbb{R}^d$ is the global model, $f_n: \mathbb{R}^d \rightarrow \mathbb{R}$ is the loss function of Client n , and N is the total number of clients. This formulation is prevalent in Federated Learning (FL) [37], a distributed learning paradigm designed for privacy preservation, where clients train the global model on their local data and send updates to the server for aggregation.

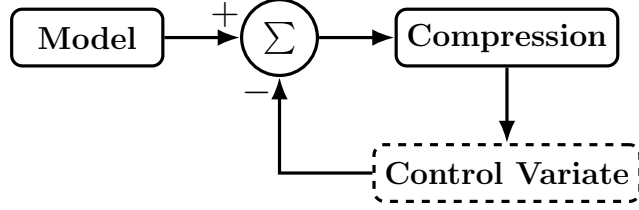


Figure 4.1: Error feedback block diagram. Typically, each client stores its own control variate and must update it after each training round. The server stores its own copy of each client’s control variate.

A primary bottleneck in distributed learning is communication overhead. Given that communication at edge devices is not symmetric, with the upload (client to server) link being significantly more constrained [79], the download cost is considered negligible in the distributed learning literature. Thus, the primary concern is the cost of uploading model updates from clients to the server [21]. To mitigate this, various lossy compression techniques, such as quantization [3], sparsification [80], and low-rank approximation [81], have been proposed. While unbiased compressors are easier to analyze, biased compressors are frequently preferred in practice for their superior empirical performance and computational efficiency [21, 82].

To guarantee the convergence of algorithms using biased compression, a mechanism known as error feedback (or error compensation) is typically required [4]. This involves each client tracking the accumulated compression error locally and correcting for it in subsequent updates. Figure 4.1 shows the block diagram for an error feedback mechanism. The control variate shown in the figure is precisely the variable responsible for accumulating the compression error at each client. However, standard error feedback requires both parties (clients and server) to store these error vectors in their respective control variates. In other words, the clients and the server must maintain a state between rounds to be able to compensate for past compression errors. This introduces two problems: (i) storing control variates at the clients and (ii) storing control variates at the server. For the first issue, in many large-scale distributed learning scenarios, such as cross-device FL, clients are very unreliable and

are thus continuously added to and dropped from the training pool throughout the rounds. Therefore, in most deployments, clients are assumed to be stateless and cannot maintain the state between communication rounds [21]. For the latter issue, having the server store a control variate for each client introduces privacy risks by tracking individual client behavior over time, which completely defeats the purpose of many distributed training applications.

This chapter addresses these challenges by proposing two novel frameworks that enable biased compression without requiring client-side state or per-client server-side tracking. The core idea is to provide clients with a globally consistent predictor vector. Clients then compress the **difference** between their local update and this predictor. If the predictor is a good approximation of the local update, the difference vector will be sparse or will have a small magnitude, making it highly compressible.

We propose two distinct sources for this predictor, **Compressed Aggregate Feedback (CAFe)** and **Server-Guided Compressed Aggregate Feedback (CAFe-S)**. The first, **CAFe**, uses the aggregated and decoded update from the *previous* communication round as a predictor for the current round k . This approach is entirely self-contained, requires no additional data, and is suitable for any FL setting. Clients receive this shared vector from the server along with the new model, use it for compression, and then discard it, thus remaining stateless. The second framework, **CAFe-S**, is designed for scenarios where the server has access to a small private dataset relevant to the global learning task. The server uses this dataset to compute a “candidate update” which serves as a more timely and potentially more accurate predictor than the lagged aggregate from **CAFe**.

Both frameworks are inspired by the use of temporal prediction in video coding, where compressing the difference between consecutive frames yields significant gains [57]. We analyze both methods in the **Distributed Gradient Descent (DGD)** setting for non-convex objectives. Our theoretical contributions show that **CAFe** improves the convergence rate over standard compressed DGD, while **CAFe-S** achieves a convergence rate that depends on the similarity

between the server’s data and the clients’ data. Experimental results on standard FL benchmarks confirm that our proposed methods outperform existing compression schemes.

To the best of our knowledge, this is the first work to propose a globally shared error feedback mechanism that eliminates the need for client-side memory. Our main contributions are summarized as follows:

- **Stateless Error Feedback:** We propose **CAFe**, a framework that leverages the aggregated update from the previous round as a global predictor. This allows clients to transmit compressed difference vectors without maintaining local error accumulation vectors, making it uniquely suitable for cross-device FL with stateless clients.
- **Server-Guided Optimization:** We introduce **CAFe-S**, which utilizes a server-side auxiliary dataset to generate candidate updates. This transforms the server from a passive aggregator into an active participant that guides the compression process, reducing uplink variance.
- **Theoretical Guarantees:** We provide a rigorous convergence analysis for both frameworks in the non-convex setting. We prove that **CAFe** achieves a convergence rate superior to standard DCGD by a factor of $(1 - \omega)$, and **CAFe-S** achieves a rate that improves linearly with the representativeness of the server’s data.

The remainder of this chapter is organized as follows. We review related work in communication compression and error feedback in Section 4.2. We then formally introduce the **CAFe** and **CAFe-S** frameworks in Section 4.3. In Section 4.4, we provide a detailed convergence analysis for both methods, comparing their theoretical guarantees with the DCGD baseline. Finally, we present our experimental validation in Section 4.5 and conclude in Section 4.6.

4.2 Related Work

Communication compression is a well-studied topic in distributed learning, and error feedback is often suggested to improve convergence guarantees [83]. In [4], the authors study the error feedback mechanism for one-bit per coordinate biased compression. For general sparse compressors, error feedback was studied in [59, 60]. For the decentralized setting, [2, 6] proposed variants of error feedback with general compression operators. For asynchronous methods, [7, 8] also showed that a modified error feedback with general compression operators has good convergence guarantees. In the non-convex setting, [61] showed that error feedback can be used in arbitrarily heterogeneous settings, which was later extended to the stochastic and convex settings in [82]. All these methods, however, adhere to the traditional model of per-client error accumulation, making them unsuitable for stateless FL environments. Our work departs from this by using a global, shared predictor, eliminating the need for client-specific states.

Table 4.1 positions our work against key state-of-the-art baselines. While MARINA [84] offers variance reduction for unbiased compression, and recent methods like EF21-P [85] and ADEF [86] push the boundaries of convergence speed through bidirectional feedback and acceleration, they all share a common limitation: the requirement for *client-side state*. Whether storing error vectors, gradient estimators, or momentum buffers, these requirements hinder participation in stateless, cross-device Federated Learning. Our frameworks, **CAFe** and **CAFe-S**, eliminate this dependency by using global, server-side predictors.

4.3 Proposed Frameworks

To discuss the algorithm design, first, we must cover some compression preliminaries. When clients send a message to the server, they first encode it using a function E . The server decodes the received information using a function D . We call these functions the encoder and

Table 4.1: Feature comparison with state-of-the-art distributed learning frameworks

Algorithm	Compression	Error Feedback	Client State	Server Data
<i>Standard Baselines</i>				
DGD [37]	None	N/A	No	No
DCGD [3]	Possibly Biased	None	No	No
<i>State-of-the-Art (Stateful)</i>				
EF21 [61]	Possibly Biased	Local Control Variate	Yes	No
MARINA [84]	Unbiased	Gradient Estimator	Yes	No
EF21-P [85]	Possibly Biased	Bidirectional EF	Yes	No
ADEF [86]	Possibly Biased	Accelerated Local EF	Yes	No
<i>Proposed Frameworks (Stateless)</i>				
CAFe (Ours)	Possibly Biased	Global Aggregated	No	No
CAFe-S (Ours)	Possibly Biased	Global Server-Guided	No	Yes

Client State: Indicates if clients must persist a vector (e.g., error accumulator e_n^k , gradient estimator h_n^k , or momentum buffer) between rounds.

ADEF integrates Nesterov acceleration with error feedback, which requires maintaining local momentum states.

decoder, respectively. For a general compression mechanism, the composition $D(E(x)) := \mathcal{C}(x)$ is called a compression operator [59].

DEFINITION 4.1. *A compression operator is a function $\mathcal{C}: \mathbb{R}^d \rightarrow \mathbb{R}^d$, paired with a positive compression parameter $\omega < 1$, such that for any vector x ,*

$$\mathbb{E} [\|\mathcal{C}(x) - x\|^2] \leq \omega \|x\|^2. \quad (4.2)$$

EXAMPLE 4.2 (Top-k Compression [3]). *The top-k operator zeroes out all but the k largest-magnitude elements of a vector. It satisfies Equation (4.2) with $\omega = 1 - k/d$.*

Next, we describe how compression operators are used when minimizing the global loss function from Equation (4.1) in a distributed learning setting. The fundamental algorithm for this purpose is **Distributed Compressed Gradient Descent (DCGD)** — see Algorithm 2. The pseudocode shows how, at each round, the global model is sent to the clients, which train it using gradients computed with local data. Clients then compress these gradients and send them to the server, which averages them to update the global model. This process is repeated for any desired number of rounds. Note that DCGD is a specific instance of the

Algorithm 2 Distributed Compressed Gradient Descent (DCGD)

Input: Initial model x^0 , Rounds K , Encoder-Decoder (E, D) pair for compression, learning rate γ

- 1: Initialize global model x^0 , and aggregate $\Delta_s^0 \leftarrow 0$
- 2: **for** round k from 0 to $K - 1$ **do**
- 3: Send x^k to all clients
- 4: **for** each client n in parallel **do**
- 5: $y_n^k \leftarrow x^k - \gamma \nabla f_n(x^k)$ *// Train x^k using local data, store the output in y_n^k*
- 6: $\Delta_n^k \leftarrow y_n^k - x^k = -\gamma \nabla f_n(x^k)$ *// Compute local update*
- 7: Send $E(\Delta_n^k)$ to server *// Upload local update*
- 8: **end for**
- 9: Server decodes each client n via $q_n^k \leftarrow D(E(\Delta_n^k))$
- 10: Server aggregates client updates $\Delta_s^k := \frac{1}{N} \sum q_n^k$
- 11: Server updates model $x^{k+1} := x^k + \Delta_s^k$.
- 12: **end for**

general distributed learning framework, where we have chosen **Gradient Descent (GD)** as the optimizer for local models, and equal-weight averaging for the aggregation strategy. We can derive a general strategy by not determining the aggregation strategy for client updates, nor the optimizer for on-client training.

4.3.1 CAFe Overview

Our framework, **CAFe**, leverages the previous aggregated update Δ_s^{k-1} to help clients compute a more compressible update. Namely, clients compress the difference between their local update Δ_n^k and the previous aggregated update

$$E(\Delta_n^k - \Delta_s^{k-1}).$$

On the server side, the server will add the previous aggregated update when decoding the received messages

$$q_n^k \leftarrow D(E(\Delta_n^k - \Delta_s^{k-1})) + \Delta_s^{k-1}. \quad (4.3)$$

The pseudocode for this procedure is in Algorithm 3, where the novelty with respect to the general distributed learning framework is highlighted in green boxes.

Algorithm 3 Compressed Aggregate Feedback (CAFe)

Input: Initial model x^0 , Rounds K , Encoder-Decoder (E, D) pair for compression, learning rate γ

- 1: Initialize global model x^0 , and aggregate $\Delta_s^{-1} \leftarrow 0$
- 2: **for** round k from 0 to $K - 1$ **do**
- 3: Send x^k and Δ_s^{k-1} to all clients *// In the stateful version Δ_s^{k-1} may be omitted*
- 4: **for** each client n in parallel **do**
- 5: Compute local update $\Delta_n^k \leftarrow -\gamma \nabla f_n(x^k)$
- 6: Upload difference: send $E(\Delta_n^k - \Delta_s^{k-1})$ to server
- 7: **end for**
- 8: Server decodes each client n via $q_n^k \leftarrow D(E(\Delta_n^k - \Delta_s^{k-1})) + \Delta_s^{k-1}$
- 9: Server aggregates $\Delta_s^k := \frac{1}{N} \sum_{n=1}^N q_n^k$
- 10: Server updates model $x^{k+1} := x^k + \Delta_s^k$
- 11: **end for**

The intuition is that if the client gradients do not change drastically between rounds, Δ_n^k will be close to the average update Δ_s^{k-1} , making their difference highly compressible.

This method allows for stateless participation, as the predictor Δ_s^{k-1} is provided by the server. However, if a client participated in the previous round and retained the model x^{k-1} , they can locally recover the predictor via $\Delta_s^{k-1} = x^k - x^{k-1}$ without additional communication. Explicit transmission of the predictor is therefore only strictly necessary for clients that are dropping in, returning after absence, or strictly memory-constrained.

In many popular distributed learning algorithms, x^k and x^{k-1} determine Δ_s^{k-1} , like Distributed Gradient Descent, FedAvg, etc. In other words, since the predictor Δ_s^{k-1} represents the net global model update, this framework naturally accommodates server-side momentum or adaptive optimizers (e.g., Adam). In such cases, Δ_s^{k-1} captures the momentum-accumulated velocity, which stabilizes the update trajectory and often serves as a more accurate predictor of the next step than raw gradients alone.

Note that the error feedback mechanism in [61] is a special case of **CAFe** with a single client. In this case, the aggregated update at the server is simply the client update, and we can analyze it as a control variate. However, in the multi-client setting, the aggregated update is a combination of all client updates, which acts as a proxy for client-specific control variates and requires novel analysis, shown in Section 4.4.

4.3.2 CAFe-S Overview

Our second proposal, **CAFe-S**, is designed for scenarios where the server holds a small private dataset, with a corresponding loss function $f_s(x)$. Instead of using the lagged update Δ_s^{k-1} , the server computes a candidate update, $\Delta_c^k = -\gamma \nabla f_s(x^k)$, using its own data and the current model x^k . This candidate update is sent to clients as the predictor. The procedure is detailed in Algorithm 4.

Algorithm 4 Server-Guided Compressed Aggregate Feedback (CAFe-S)

Input: Initial model x^0 , rounds K , compression (E, D) , learning rate γ , server loss f_s

- 1: **for** round k from 0 to $K - 1$ **do**
- 2: Server computes candidate $\Delta_c^k \leftarrow -\gamma \nabla f_s(x^k)$
- 3: Server sends x^k and Δ_c^k to all clients
- 4: **for** each client n in parallel **do**
- 5: Compute local update $\Delta_n^k \leftarrow -\gamma \nabla f_n(x^k)$
- 6: Upload difference: send $E(\Delta_n^k - \Delta_c^k)$ to server
- 7: **end for**
- 8: Server decodes each client n via $q_n^k \leftarrow D(E(\Delta_n^k - \Delta_c^k)) + \Delta_c^k$
- 9: Server aggregates $\Delta_s^k := \frac{1}{N} \sum_{n=1}^N q_n^k$
- 10: Server updates model $x^{k+1} := x^k + \Delta_s^k$
- 11: **end for**

If the server's data distribution is similar to the average distribution of clients, Δ_c^k will be a highly accurate predictor of Δ_n^k , leading to greater compression gains.

REMARK 4.3 (Communication Trade-Off). *In the stateless configuration, the server must transmit the predictor Δ_s^{k-1} (for **CAFe**) or Δ_c^k (for **CAFe-S**) to the clients alongside the global*

model x^k . While this effectively doubles the downlink payload compared to standard *DGD*, the trade-off is favorable in Federated Learning for several reasons. First, edge client connections (e.g., LTE/5G, residential broadband) are typically highly asymmetric, with downlink bandwidth significantly exceeding uplink bandwidth [79]. Second, the predictor is a single global vector broadcast to all participants, whereas client updates are unique unicast transmissions. Therefore, maximizing the compression of the scarce unicast uplink capacity at the expense of the abundant broadcast downlink capacity is potentially a net-positive strategy.

4.4 Analysis

Having formally defined both frameworks, we now proceed to analyze their convergence properties. We choose *GD* as the optimizer for local training, as it is representative of a wide range of distributed learning algorithms [21]. We proceed with the following standard assumptions [21, 83]:

ASSUMPTION 4.4 (L-Smoothness). *See [69, Lemma 1.2.3.].*

The objective function f is L -smooth, which implies that it is differentiable, ∇f is L -Lipschitz, and

$$f(y) \leq f(x) + \langle \nabla f(x), y - x \rangle + \frac{L}{2} \|y - x\|^2. \quad (4.4)$$

Also, the objective function f is lower-bounded by f^* .

ASSUMPTION 4.5 (Bounded Local Dissimilarity). *The local gradients have bounded dissimilarity, that is, there exists a $B^2 \geq 1$ such that*

$$\frac{1}{N} \sum_{n=1}^N \|\nabla f_n(x)\|^2 \leq B^2 \|\nabla f(x)\|^2. \quad (4.5)$$

For reference, we first state the convergence of the baseline DCGD algorithm.

THEOREM 4.6. *[DCGD Convergence] Under Assumptions 4.4 and 4.5, with a learning rate $\gamma \leq \frac{1}{L}$ and a compression parameter ω such that $\omega B^2 < 1$, DCGD satisfies*

$$\frac{1}{K} \sum_{k=0}^{K-1} \mathbb{E} \left[\|\nabla f(x^k)\|^2 \right] \leq \frac{2(f(x^0) - f^*)}{\gamma K} \cdot \frac{1}{1 - \omega B^2}. \quad (4.6)$$

Proof Sketch. The proof relies on a standard descent lemma [69, Section 1.2.3]. The main challenge is to bound the variance introduced by compression, $\mathbb{E} \left[\|\bar{e}^k\|^2 \right]$, which represents the average compression error. Using Assumption 4.5, we can show this error is bounded by the gradient norm: $\mathbb{E} \left[\|\bar{e}^k\|^2 \right] \leq \omega B^2 \mathbb{E} \left[\|\nabla f(x^k)\|^2 \right]$. Plugging this bound into the descent lemma and telescoping the sum over K iterations yields the final result. The full proof with details is in Appendix B.2 □

4.4.1 Analysis of CAFe

The key benefit of **CAFe** is that it improves the dependency on the compression parameter ω . The required learning-rate condition is

$$\gamma \leq \frac{1 - \omega}{L(1 + \omega)}. \quad (4.7)$$

THEOREM 4.7. *[CAFe + DGD Convergence] Under Assumptions 4.4 and 4.5, with a learning rate satisfying Equation (4.7) and $\omega B^2 < 1$, CAFe with DGD satisfies*

$$\frac{1}{K} \sum_{k=0}^{K-1} \mathbb{E} \left[\|\nabla f(x^k)\|^2 \right] \leq \frac{2(f(x^0) - f^*)}{\gamma K} \cdot \frac{1 - \omega}{1 - \omega B^2}. \quad (4.8)$$

Proof Sketch. The analysis for **CAFe** is more involved because the predictor Δ_s^{k-1} creates a

temporal dependency: the compression error at Round k is linked to the aggregate update (and thus the error) from Round $k - 1$. We cannot bound the error at Step k purely by the state at Step k , as we do for DCGD.

The proof, detailed in Appendix B.2, proceeds in two main parts. We first derive a bound that links the expected error at Step $k + 1$, $\mathbb{E} [\|\bar{e}^{k+1}\|^2]$, to the error at Step k , $\mathbb{E} [\|\bar{e}^k\|^2]$ and the model gradients. Second, we define a Lyapunov function $\Psi^k := \mathbb{E} [f(x^k) + C\|\bar{e}^k\|^2]$ for a carefully chosen constant $C = \frac{\gamma}{2(1-\omega)}$. By combining our recursive error bound with the standard descent lemma, we show that this Lyapunov function decreases at each step ($\Psi^{k+1} \leq \Psi^k - (\text{progress term})$). Telescoping this inequality proves convergence. $\square$

Comparing Theorem 4.7 to Theorem 4.6, **CAFe** introduces an improvement factor of $(1 - \omega)$. This is significant for aggressive compression where ω is close to 1.

4.4.2 Analysis of CAFe-S

The analysis for **CAFe-S** requires an additional assumption that quantifies the similarity between the server's data and the clients' data.

ASSUMPTION 4.8 (Bounded Server-Client Dissimilarity). *The local gradients and the server gradient have bounded dissimilarity, that is, there exists a $G^2 \geq 0$ such that*

$$\frac{1}{N} \sum_{n=1}^N \|\nabla f_n(x) - \nabla f_s(x)\|^2 \leq G^2 \frac{1}{N} \sum_{n=1}^N \|\nabla f_n(x)\|^2. \quad (4.9)$$

A small G^2 implies that the server's data is a good proxy for the aggregate client data.

THEOREM 4.9. *[CAFe-S + DGD Convergence] Under Assumptions 4.4, 4.5 and 4.8, with a learning rate $\gamma \leq \frac{1}{L}$, and a compression parameter ω such that $\omega G^2 B^2 < 1$, CAFe-S with DGD*

satisfies

$$\frac{1}{K} \sum_{k=0}^{K-1} \mathbb{E} \left[\|\nabla f(x^k)\|^2 \right] \leq \frac{2(f(x^0) - f^*)}{\gamma K(1 - \omega G^2 B^2)}. \quad (4.10)$$

Proof Sketch. The proof for **CAFe-S** has a structure very similar to that of **DCGD** (Theorem 4.6). The key difference is in how we bound the compression error $\mathbb{E} \left[\|\bar{e}^k\|^2 \right]$.

Instead of compressing the raw update Δ_n^k , clients compress the difference $\Delta_n^k - \Delta_c^k$. We bound the error of this compression using Assumptions 4.5 and 4.8, which shows that $\mathbb{E} \left[\|\bar{e}^k\|^2 \right] \leq (\omega G^2 B^2) \mathbb{E} \left[\|\nabla f(x^k)\|^2 \right]$. This error bound has the exact same form as the **DCGD** bound, but with the factor ωB^2 replaced by the (potentially much smaller) factor $\omega G^2 B^2$. The rest of the proof is identical to that of Theorem 4.6. The full proof is in Appendix B.2. $\square$

Theorem 4.9 shows that **CAFe-S** achieves a convergence rate that depends on the similarity between the server’s data and the clients’ data. This result is intuitive: if the server’s data is a perfect match ($G^2 = 0$), the algorithm achieves the same convergence rate as **DCGD**. For all values of $G^2 < 1$, **CAFe-S** outperforms **DCGD**, which reflects the advantage of having a representative dataset at the server.

4.4.3 Discussion of the Theoretical Results

Let us highlight the implications of our theoretical findings. Comparing **CAFe** (Theorem 4.7) with the baseline **DCGD** (Theorem 4.6), we observe an improvement factor of $(1 - \omega)$ in the convergence bound for the same learning rate, which becomes significant as compression becomes more aggressive (i.e., as ω approaches 1). This gain comes at the cost of a stricter learning rate condition, $\gamma \leq \frac{1-\omega}{L(1+\omega)}$, which is common for error-feedback methods. In practice, however, we usually chose learning rates much smaller than $\frac{1}{L}$, so this condition is not a significant limitation.

Comparing **CAFe-S** (Theorem 4.9) with DCGD (Theorem 4.6), we see a different kind of improvement. The convergence rate of **CAFe-S** depends on $1 - \omega G^2 B^2$ instead of $1 - \omega B^2$. The new term G^2 , defined in Assumption 4.8, quantifies the dissimilarity between the server’s gradient and the average of the clients’ gradients. If the server’s data represent the global data distribution, G^2 will be small ($G^2 < 1$), resulting in a faster convergence rate compared to DCGD. In the ideal case where the server’s data is a perfect proxy ($G^2 \rightarrow 0$), the negative impact of both compression (ω) and client heterogeneity (B^2) on the convergence rate is eliminated. Importantly, **CAFe-S** achieves this improvement without imposing additional constraints on the learning rate, keeping the standard DCGD condition, $\gamma \leq \frac{1}{L}$.

In short, the main take-aways from this analysis are as follows: **CAFe** is a *data-free* method. It provides a *guaranteed* convergence improvement of $(1 - \omega)$ over standard compression, making it desirable for aggressive compression, but it requires a smaller learning rate. In contrast, **CAFe-S** is a *data-dependent* method. It provides a *conditional* improvement if the server’s data represent the population (low G^2). It has no learning rate penalty, but its performance depends on the quality of the server-side dataset.

4.5 Experimental Results

We now empirically validate the theoretical findings of Section 4.4. Our experiments are designed to: (i) demonstrate the core working principle that our difference-based compression is effective, (ii) compare the end-to-end performance of **CAFe** against baselines with various datasets and compression methods, and (iii) validate the **CAFe-S** hypothesis (Theorem 4.9) that convergence performance is directly tied to the quality of the server’s data. The code to reproduce all experiments is publicly available [87].

4.5.1 Experimental Setup

We conducted FL experiments with 10 clients on MNIST [88], EMNIST [89], CIFAR-10 and CIFAR-100 [67] datasets. We used `CONV4` models for MNIST/EMNIST, a `Dense6` model for CIFAR-10, and a `ResNet-20` model for CIFAR-100, following the setup in [90]. We present results for both homogeneous and heterogeneous data cases, denoted *iid* and *non-iid*, respectively. For the latter, we randomly sampled 40% of the total classes for each client. For each round of communication, which we refer to as a global training round, all clients participated in training. Clients performed one local GD training epoch with a batch size of 64 per global training round. We varied the number of global training rounds for each experiment. We ran each experiment with 3 random seeds and report the accuracy means and standard deviations.

The learning rates for each dataset and model combination were tuned by performing a sweep over a set of seven log-spaced candidate values, ranging from 10^{-3} to 10^0 . We selected the rate that yielded the highest accuracy during the specified number of rounds. This process was performed independently for each method (combination of task and compressor), and we found that the optimal rates were approximately consistent across both frameworks. The exact learning rates used in each experiment are tabulated in Table B.1. We also provide a detailed sensitivity analysis for the learning rate and further details on the compressor setup in Appendix B.1.

4.5.2 Validation of Working Principle

The premise of our proposed frameworks is that the difference between the local update and a shared predictor is more compressible than the raw update itself. To quantify this, we define the *Compression Gain Ratio* for a predictor P at Round k as

$$\frac{\|\Delta_n^k - P\|}{\|\Delta_n^k\|}. \quad (4.11)$$

A ratio smaller than one indicates that the predictor reduces the magnitude of the vector to be compressed.

To visualize this principle, we simulated a Federated Logistic Regression task using synthetic data ($D = 200$, 10 clients, iid data, and a server-side dataset of the same size) over 500 communication rounds, with no compression. We log the updates that would be sent with no modification, with **CAFe**, and with **CAFe-S**. Figure 4.2 presents the analysis of the training dynamics and the corresponding logs. The left panel displays the global training loss, confirming that the model converges to the optimum. The middle panel plots the Compression Gain Ratio. During the active training phase, **CAFe** consistently achieves a ratio below 1.0, confirming that the difference vector’s magnitude is smaller than that of the raw gradient. We observe that as the model converges and the gradients vanish (approaching Round 500), the ratio trends toward 1.0. This occurs because both the update Δ_n^k and the predictor Δ_s^{k-1} approach zero, and the relative benefit of the predictor diminishes as the updates become dominated by stochastic noise. The right panel shows the log-density histogram of the update values aggregated across the entire training process. The **CAFe** distribution (blue) is significantly narrower compared to the direct updates (red). For clarity, we omit the **CAFe-S** distribution, which closely resembles that of **CAFe**. This increased sparsity suggests that the difference vectors are more efficient to compress.

4.5.3 Performance of CAFe

Having validated the premise, we now evaluate the end-to-end performance of **CAFe** against direct compression in FL. We test four biased compression methods. First, we use *Top-k* (see Example 4.2), which retains only the k elements with the largest magnitudes. Second, we use *Low-rank approximation* via PowerSGD [81], which compresses the update matrix by decomposing it into the product of two smaller matrices of rank r (which we vary from 1 to 3), effectively projecting the update onto a lower-dimensional subspace. We also evaluate

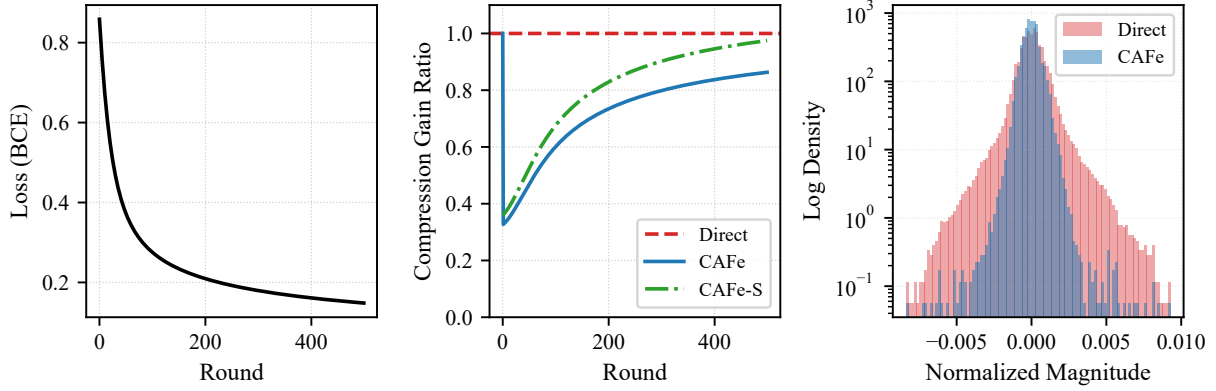


Figure 4.2: Analysis of the **CAFe** working principle on a synthetic logistic regression task. **Left:** Global training loss showing convergence. **Middle:** The Compression Gain Ratio (ρ) remains below 1.0 for the majority of training, indicating that the difference vector has a smaller norm than the raw update. The ratio approaches 1.0 as the model converges. **Right:** The log-density histogram shows that **CAFe** updates are more peaked at zero (sparser) than direct updates.

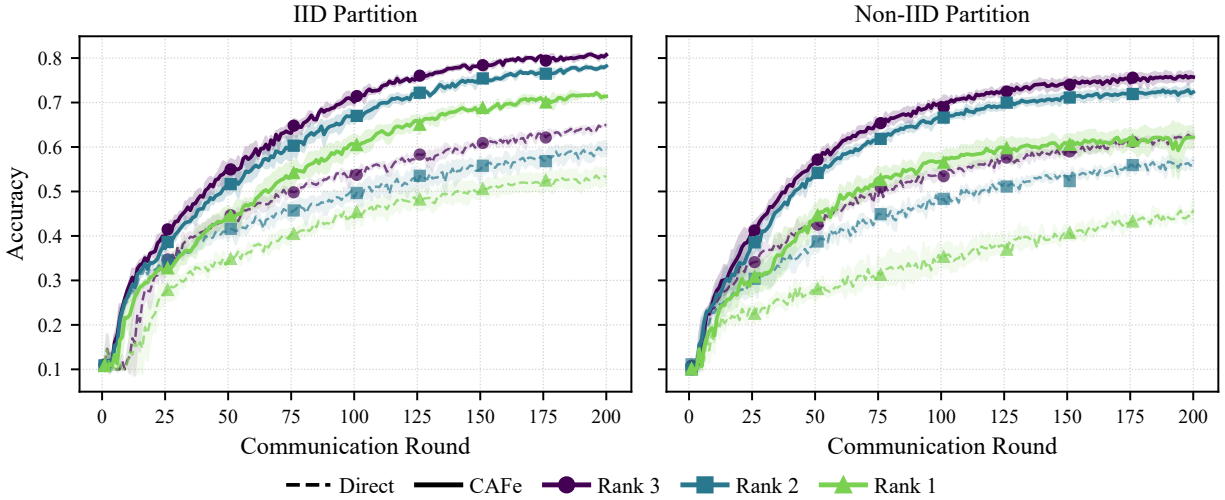


Figure 4.3: Learning curves comparing direct compression (Direct) and **CAFe** with low-rank compression in the CIFAR-10 task for iid and non-iid settings. **CAFe** results are presented with solid lines, and Direct compression results are presented with dashed lines. **CAFe** demonstrates faster convergence and higher accuracy.

Table 4.2: Accuracy for CAFE and Direct compression (MNIST, EMNIST).

Compressor	Param.	MNIST (CONV4, 50 rounds)				EMNIST (CONV4, 50 rounds)			
		iid		non-iid		iid		non-iid	
		10 classes/client		4 classes/client		47 classes/client		4 classes/client	
		Direct	CAFe	Direct	CAFe	Direct	CAFe	Direct	CAFe
None	—	99.3 $\pm$ 0.0		99.0 $\pm$ 0.0		88.9 $\pm$ 0.1		82.2 $\pm$ 0.2	
Top-k	k=10%	99.4 $\pm$ 0.0	99.4 $\pm$ 0.0	99.0 $\pm$ 0.1	99.0 $\pm$ 0.1	88.9 $\pm$ 0.0	88.9 $\pm$ 0.1	81.8 $\pm$ 0.6	81.7 $\pm$ 0.1
	k=1%	99.2 $\pm$ 0.0	99.3 $\pm$ 0.0	98.7 $\pm$ 0.1	98.9 $\pm$ 0.1	88.0 $\pm$ 0.1	88.3 $\pm$ 0.1	79.5 $\pm$ 0.2	81.4 $\pm$ 0.3
	k=0.1%	98.7 $\pm$ 0.1	99.0 $\pm$ 0.1	97.6 $\pm$ 0.3	98.1 $\pm$ 0.1	85.6 $\pm$ 0.3	86.9 $\pm$ 0.2	69.0 $\pm$ 2.3	75.2 $\pm$ 0.2
Top-1% + Quantizer	6 bits	99.3 $\pm$ 0.0	99.2 $\pm$ 0.1	98.7 $\pm$ 0.1	98.9 $\pm$ 0.1	88.1 $\pm$ 0.1	88.3 $\pm$ 0.1	79.7 $\pm$ 0.7	81.3 $\pm$ 0.2
	5 bits	99.3 $\pm$ 0.0	99.3 $\pm$ 0.0	98.6 $\pm$ 0.1	98.9 $\pm$ 0.1	88.2 $\pm$ 0.1	88.5 $\pm$ 0.1	80.6 $\pm$ 0.2	81.7 $\pm$ 0.8
	4 bits	99.3 $\pm$ 0.0	99.2 $\pm$ 0.1	98.5 $\pm$ 0.1	98.8 $\pm$ 0.1	88.0 $\pm$ 0.2	88.4 $\pm$ 0.1	79.1 $\pm$ 0.6	81.2 $\pm$ 0.3
Low-rank	rank=3	99.0 $\pm$ 0.1	99.2 $\pm$ 0.1	98.4 $\pm$ 0.1	98.8 $\pm$ 0.0	86.5 $\pm$ 0.1	88.3 $\pm$ 0.1	74.6 $\pm$ 1.1	79.7 $\pm$ 0.6
	rank=2	98.9 $\pm$ 0.1	99.2 $\pm$ 0.1	98.2 $\pm$ 0.1	98.8 $\pm$ 0.0	85.8 $\pm$ 0.3	87.9 $\pm$ 0.0	67.3 $\pm$ 2.5	76.7 $\pm$ 0.2
	rank=1	98.5 $\pm$ 0.0	99.2 $\pm$ 0.1	97.4 $\pm$ 0.1	98.5 $\pm$ 0.2	83.5 $\pm$ 0.2	87.0 $\pm$ 0.1	50.0 $\pm$ 3.2	71.0 $\pm$ 0.4
Rank-1 + Quantizer	5 bits	98.5 $\pm$ 0.0	99.2 $\pm$ 0.0	97.2 $\pm$ 0.4	98.5 $\pm$ 0.2	83.4 $\pm$ 0.5	87.0 $\pm$ 0.1	49.6 $\pm$ 3.4	71.6 $\pm$ 0.9
	4 bits	98.6 $\pm$ 0.1	99.2 $\pm$ 0.0	97.4 $\pm$ 0.1	98.6 $\pm$ 0.0	83.6 $\pm$ 0.3	87.0 $\pm$ 0.0	48.1 $\pm$ 3.4	71.3 $\pm$ 0.6
	3 bits	98.6 $\pm$ 0.1	99.2 $\pm$ 0.0	97.3 $\pm$ 0.2	98.6 $\pm$ 0.0	83.7 $\pm$ 0.2	87.0 $\pm$ 0.2	44.7 $\pm$ 3.1	70.3 $\pm$ 0.7

Table 4.3: Accuracy for CAFE and Direct compression (CIFAR-10, CIFAR-100).

Compressor	Param.	CIFAR-10 (CONV6, 200 rounds)				CIFAR-100 (ResNet-20, 200 rounds)			
		iid		non-iid		iid		non-iid	
		10 classes/client		4 classes/client		100 classes/client		40 classes/client	
		Direct	CAFe	Direct	CAFe	Direct	CAFe	Direct	CAFe
None	—	88.5 $\pm$ 0.3		84.4 $\pm$ 0.3		41.3 $\pm$ 1.6		39.3 $\pm$ 1.3	
Top-k	k=10%	86.3 $\pm$ 0.2	87.5 $\pm$ 0.5	81.5 $\pm$ 1.0	83.9 $\pm$ 0.6	33.1 $\pm$ 0.8	37.0 $\pm$ 0.4	32.1 $\pm$ 1.0	34.4 $\pm$ 2.3
	k=1%	74.1 $\pm$ 0.7	80.5 $\pm$ 0.8	70.3 $\pm$ 0.3	76.8 $\pm$ 0.6	19.8 $\pm$ 0.8	23.1 $\pm$ 0.7	19.9 $\pm$ 0.4	21.7 $\pm$ 2.0
	k=0.1%	52.9 $\pm$ 1.6	61.9 $\pm$ 2.4	48.2 $\pm$ 1.5	60.8 $\pm$ 2.8	12.1 $\pm$ 0.7	12.3 $\pm$ 1.2	11.0 $\pm$ 0.4	10.7 $\pm$ 0.3
Top-1% + Quantizer	6 bits	74.5 $\pm$ 0.8	77.5 $\pm$ 1.5	69.4 $\pm$ 1.4	77.0 $\pm$ 0.9	20.1 $\pm$ 0.1	23.4 $\pm$ 1.2	20.0 $\pm$ 0.6	22.0 $\pm$ 0.5
	5 bits	72.9 $\pm$ 0.3	75.3 $\pm$ 0.6	69.7 $\pm$ 2.1	76.5 $\pm$ 1.1	20.0 $\pm$ 0.8	24.0 $\pm$ 0.5	20.7 $\pm$ 0.7	22.0 $\pm$ 0.7
	4 bits	13.1 $\pm$ 2.7	72.5 $\pm$ 2.0	69.2 $\pm$ 0.3	77.0 $\pm$ 0.9	20.7 $\pm$ 0.0	24.3 $\pm$ 0.2	19.1 $\pm$ 0.7	21.7 $\pm$ 0.4
Low-rank	rank=3	65.0 $\pm$ 0.4	80.8 $\pm$ 0.4	62.7 $\pm$ 0.9	76.0 $\pm$ 0.8	19.3 $\pm$ 0.3	32.4 $\pm$ 0.5	19.5 $\pm$ 0.7	26.6 $\pm$ 0.6
	rank=2	59.9 $\pm$ 1.2	78.2 $\pm$ 0.1	57.0 $\pm$ 0.7	72.8 $\pm$ 1.1	16.5 $\pm$ 1.2	29.8 $\pm$ 0.2	17.4 $\pm$ 0.2	24.6 $\pm$ 1.0
	rank=1	53.6 $\pm$ 1.9	72.2 $\pm$ 0.4	45.8 $\pm$ 2.6	62.2 $\pm$ 2.4	12.9 $\pm$ 0.5	24.1 $\pm$ 0.6	13.3 $\pm$ 0.8	17.6 $\pm$ 1.2
Rank-1 + Quantizer	5 bits	52.5 $\pm$ 0.8	71.8 $\pm$ 0.4	43.6 $\pm$ 1.4	62.8 $\pm$ 1.8	13.1 $\pm$ 0.7	24.3 $\pm$ 0.5	12.2 $\pm$ 0.3	17.4 $\pm$ 0.9
	4 bits	53.5 $\pm$ 1.4	70.7 $\pm$ 0.5	46.2 $\pm$ 2.3	62.7 $\pm$ 0.9	13.2 $\pm$ 0.2	23.5 $\pm$ 0.7	14.0 $\pm$ 0.5	16.5 $\pm$ 1.1
	3 bits	53.6 $\pm$ 1.7	68.4 $\pm$ 0.7	44.1 $\pm$ 1.5	61.7 $\pm$ 1.4	13.4 $\pm$ 0.3	22.6 $\pm$ 0.8	13.6 $\pm$ 1.1	17.3 $\pm$ 0.7

variants where these compressed outputs are further quantized to lower bit-widths. Sparsification is performed before quantization since it is optimal for FL [91]. We also include an uncompressed baseline as a reference. The results for various compression parameter settings are reported in Tables 4.2 and 4.3. The communication cost in bits per parameter (bpp) for each compression method and parameter setting is reported in Tables 4.4 and 4.5.

The results align with our theory: CAFE consistently outperforms or matches existing direct

Table 4.4: Communication cost (bits per parameter) for **CAFe** and Direct compression (MNIST, EMNIST).

		MNIST (CONV4, 50 rounds)				EMNIST (CONV4, 50 rounds)			
Compressor	Param.	iid 10 classes/client		non-iid 4 classes/client		iid 47 classes/client		non-iid 4 classes/client	
		Direct	CAFe	Direct	CAFe	Direct	CAFe	Direct	CAFe
None	—	$3.20 \cdot 10^1$		$3.20 \cdot 10^1$		$3.20 \cdot 10^1$		$3.20 \cdot 10^1$	
Top-k	k=10%	$5.40 \cdot 10^0$	$5.40 \cdot 10^0$	$5.40 \cdot 10^0$	$5.40 \cdot 10^0$	$5.40 \cdot 10^0$	$5.40 \cdot 10^0$	$5.40 \cdot 10^0$	$5.40 \cdot 10^0$
	k=1%	$5.40 \cdot 10^{-1}$	$5.40 \cdot 10^{-1}$	$5.40 \cdot 10^{-1}$	$5.40 \cdot 10^{-1}$	$5.40 \cdot 10^{-1}$	$5.40 \cdot 10^{-1}$	$5.40 \cdot 10^{-1}$	$5.40 \cdot 10^{-1}$
	k=0.1%	$5.40 \cdot 10^{-2}$	$5.40 \cdot 10^{-2}$	$5.40 \cdot 10^{-2}$	$5.40 \cdot 10^{-2}$	$5.40 \cdot 10^{-2}$	$5.40 \cdot 10^{-2}$	$5.40 \cdot 10^{-2}$	$5.40 \cdot 10^{-2}$
Top-1% + Quantizer	6 bits	$2.80 \cdot 10^{-1}$	$2.80 \cdot 10^{-1}$	$2.77 \cdot 10^{-1}$	$2.76 \cdot 10^{-1}$	$2.80 \cdot 10^{-1}$	$2.80 \cdot 10^{-1}$	$2.76 \cdot 10^{-1}$	$2.78 \cdot 10^{-1}$
	5 bits	$2.66 \cdot 10^{-1}$	$2.64 \cdot 10^{-1}$	$2.55 \cdot 10^{-1}$	$2.59 \cdot 10^{-1}$	$2.68 \cdot 10^{-1}$	$2.68 \cdot 10^{-1}$	$2.46 \cdot 10^{-1}$	$2.55 \cdot 10^{-1}$
	4 bits	$1.96 \cdot 10^{-1}$	$2.55 \cdot 10^{-1}$	$1.38 \cdot 10^{-1}$	$2.13 \cdot 10^{-1}$	$2.47 \cdot 10^{-1}$	$2.51 \cdot 10^{-1}$	$1.57 \cdot 10^{-1}$	$1.83 \cdot 10^{-1}$
Low-rank	rank=3	$4.79 \cdot 10^{-1}$	$4.79 \cdot 10^{-1}$	$4.79 \cdot 10^{-1}$	$4.79 \cdot 10^{-1}$	$4.80 \cdot 10^{-1}$	$4.80 \cdot 10^{-1}$	$4.80 \cdot 10^{-1}$	$4.80 \cdot 10^{-1}$
	rank=2	$3.22 \cdot 10^{-1}$	$3.22 \cdot 10^{-1}$	$3.22 \cdot 10^{-1}$	$3.22 \cdot 10^{-1}$	$3.22 \cdot 10^{-1}$	$3.22 \cdot 10^{-1}$	$3.22 \cdot 10^{-1}$	$3.22 \cdot 10^{-1}$
	rank=1	$1.64 \cdot 10^{-1}$	$1.64 \cdot 10^{-1}$	$1.64 \cdot 10^{-1}$	$1.64 \cdot 10^{-1}$	$1.65 \cdot 10^{-1}$	$1.65 \cdot 10^{-1}$	$1.65 \cdot 10^{-1}$	$1.65 \cdot 10^{-1}$
Rank-1 + Quantizer	5 bits	$2.59 \cdot 10^{-2}$	$2.59 \cdot 10^{-2}$	$2.59 \cdot 10^{-2}$	$2.59 \cdot 10^{-2}$	$2.60 \cdot 10^{-2}$	$2.60 \cdot 10^{-2}$	$2.60 \cdot 10^{-2}$	$2.60 \cdot 10^{-2}$
	4 bits	$2.08 \cdot 10^{-2}$	$2.08 \cdot 10^{-2}$	$2.08 \cdot 10^{-2}$	$2.08 \cdot 10^{-2}$	$2.08 \cdot 10^{-2}$	$2.08 \cdot 10^{-2}$	$2.08 \cdot 10^{-2}$	$2.08 \cdot 10^{-2}$
	3 bits	$1.56 \cdot 10^{-2}$	$1.56 \cdot 10^{-2}$	$1.56 \cdot 10^{-2}$	$1.56 \cdot 10^{-2}$	$1.57 \cdot 10^{-2}$	$1.57 \cdot 10^{-2}$	$1.57 \cdot 10^{-2}$	$1.57 \cdot 10^{-2}$

Table 4.5: Communication cost (bits per parameter) for **CAFe** and Direct compression (CIFAR-10, CIFAR-100).

		CIFAR-10 (CONV6, 200 rounds)				CIFAR-100 (ResNet-20, 200 rounds)			
Compressor	Param.	iid 10 classes/client		non-iid 4 classes/client		iid 100 classes/client		non-iid 40 classes/client	
		Direct	CAFe	Direct	CAFe	Direct	CAFe	Direct	CAFe
None	—	$3.20 \cdot 10^1$		$3.20 \cdot 10^1$		$3.20 \cdot 10^1$		$3.20 \cdot 10^1$	
Top-k	k=10%	$5.40 \cdot 10^0$	$5.40 \cdot 10^0$	$5.40 \cdot 10^0$	$5.40 \cdot 10^0$	$5.10 \cdot 10^0$	$5.10 \cdot 10^0$	$5.10 \cdot 10^0$	$5.10 \cdot 10^0$
	k=1%	$5.40 \cdot 10^{-1}$	$5.40 \cdot 10^{-1}$	$5.40 \cdot 10^{-1}$	$5.40 \cdot 10^{-1}$	$5.10 \cdot 10^{-1}$	$5.10 \cdot 10^{-1}$	$5.10 \cdot 10^{-1}$	$5.10 \cdot 10^{-1}$
	k=0.1%	$5.40 \cdot 10^{-2}$	$5.40 \cdot 10^{-2}$	$5.40 \cdot 10^{-2}$	$5.40 \cdot 10^{-2}$	$5.11 \cdot 10^{-2}$	$5.11 \cdot 10^{-2}$	$5.11 \cdot 10^{-2}$	$5.11 \cdot 10^{-2}$
Top-1% + Quantizer	6 bits	$2.59 \cdot 10^{-1}$	$2.17 \cdot 10^{-1}$	$2.71 \cdot 10^{-1}$	$2.75 \cdot 10^{-1}$	$2.50 \cdot 10^{-1}$	$2.50 \cdot 10^{-1}$	$2.49 \cdot 10^{-1}$	$2.48 \cdot 10^{-1}$
	5 bits	$2.40 \cdot 10^{-1}$	$2.41 \cdot 10^{-1}$	$2.30 \cdot 10^{-1}$	$2.57 \cdot 10^{-1}$	$2.38 \cdot 10^{-1}$	$2.40 \cdot 10^{-1}$	$1.91 \cdot 10^{-1}$	$2.12 \cdot 10^{-1}$
	4 bits	$6.60 \cdot 10^{-4}$	$1.64 \cdot 10^{-1}$	$1.88 \cdot 10^{-1}$	$2.08 \cdot 10^{-1}$	$2.21 \cdot 10^{-1}$	$2.22 \cdot 10^{-1}$	$1.07 \cdot 10^{-1}$	$1.74 \cdot 10^{-1}$
Low-rank	rank=3	$5.21 \cdot 10^{-1}$	$5.21 \cdot 10^{-1}$	$5.21 \cdot 10^{-1}$	$5.21 \cdot 10^{-1}$	$2.43 \cdot 10^0$	$2.43 \cdot 10^0$	$2.43 \cdot 10^0$	$2.43 \cdot 10^0$
	rank=2	$3.54 \cdot 10^{-1}$	$3.54 \cdot 10^{-1}$	$3.54 \cdot 10^{-1}$	$3.54 \cdot 10^{-1}$	$1.68 \cdot 10^0$	$1.68 \cdot 10^0$	$1.68 \cdot 10^0$	$1.68 \cdot 10^0$
	rank=1	$1.87 \cdot 10^{-1}$	$1.87 \cdot 10^{-1}$	$1.87 \cdot 10^{-1}$	$1.87 \cdot 10^{-1}$	$9.26 \cdot 10^{-1}$	$9.26 \cdot 10^{-1}$	$9.26 \cdot 10^{-1}$	$9.26 \cdot 10^{-1}$
Rank-1 + Quantizer	5 bits	$2.96 \cdot 10^{-2}$	$2.96 \cdot 10^{-2}$	$2.96 \cdot 10^{-2}$	$2.96 \cdot 10^{-2}$	$1.54 \cdot 10^{-1}$	$1.54 \cdot 10^{-1}$	$1.54 \cdot 10^{-1}$	$1.54 \cdot 10^{-1}$
	4 bits	$2.38 \cdot 10^{-2}$	$2.38 \cdot 10^{-2}$	$2.38 \cdot 10^{-2}$	$2.38 \cdot 10^{-2}$	$1.25 \cdot 10^{-1}$	$1.25 \cdot 10^{-1}$	$1.25 \cdot 10^{-1}$	$1.25 \cdot 10^{-1}$
	3 bits	$1.79 \cdot 10^{-2}$	$1.79 \cdot 10^{-2}$	$1.79 \cdot 10^{-2}$	$1.79 \cdot 10^{-2}$	$9.59 \cdot 10^{-2}$	$9.59 \cdot 10^{-2}$	$9.59 \cdot 10^{-2}$	$9.59 \cdot 10^{-2}$

compression methods. This is particularly stark in aggressive compression settings (low bpp). For instance, in the CIFAR-10 iid task with Top-1% + 4-bit Quantization, Direct compression collapses to random chance (13.0%), whereas **CAFe** recovers a performance of 72.5%. Similarly, with Rank-1 Low-rank compression on CIFAR-10 (non-iid), **CAFe** improves accuracy from 45.8% to 62.2%. We plot the learning curves using Low-rank compression

in Figure 4.3 to provide a visual comparison of convergence speed and accuracy.

4.5.4 Validation of CAFE-S

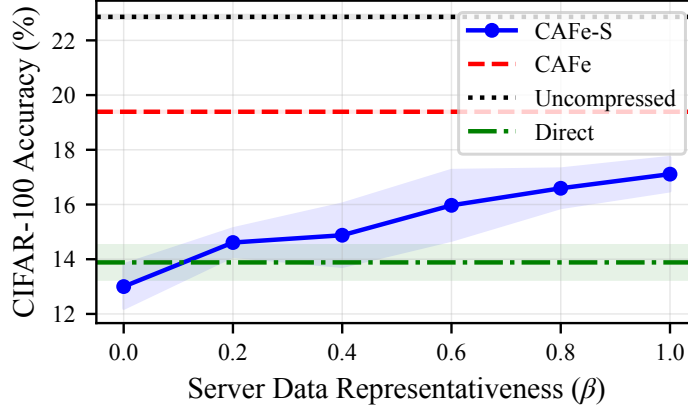


Figure 4.4: Accuracy of **CAFe-S** as a function of server’s data representativeness (β) on CIFAR-100. While **CAFe-S** improves as the server’s data become more representative, **CAFe** remains superior due to the stability of the aggregated predictor.

To validate our second proposed framework, **CAFe-S**, we designed an experiment to test the hypothesis of Theorem 4.9: that the convergence rate depends on the server-client dissimilarity, G^2 . We designed a sensitivity experiment using CIFAR-100. Note that CIFAR-100 is an image classification dataset that has 100 classes grouped into 20 superclasses, making it suitable for creating related but distinct data distributions.

Clients received data from superclasses 0–9. We then created a server-side dataset and tuned its “representativeness” using a parameter $\beta \in [0, 1]$. For $\beta = 1.0$, the server’s data were drawn entirely from superclasses 0–9 (the same as the clients), which makes the server a good proxy for the clients’ data distribution ($G^2 \approx 0$). For $\beta = 0.0$, the server’s data were drawn entirely from superclasses 10–19, which makes the server a non-representative proxy. For $\beta \in (0, 1)$, the server’s data were drawn from both superclasses, creating a mixed proxy. In all cases, the server’s dataset size was fixed at 10% of the total data.

We plot the accuracy of **CAFe-S** with rank-3 approximation compression against β in Fig-

ure 4.4, with horizontal baselines corresponding to the **CAFe** and uncompressed methods. As predicted by our theory, the performance of **CAFe-S** improves monotonically as the server’s data become more representative (increasing β). However, we observe that **CAFe** outperforms **CAFe-S** even when $\beta = 1.0$. This reveals a trade-off between *staleness* and *variance*. Although the **CAFe-S** predictor Δ_c^k is “fresh” (computed at Round k), it is generated from a small server split (10% of the total data), leading to higher variance. In contrast, the **CAFe** predictor Δ_s^{k-1} is “stale” (from Round $k - 1$), but is aggregated over all clients (90% of the data), making it a significantly more stable estimator of the global gradient direction. This result suggests that while **CAFe-S** effectively leverages server’s data, its advantage over the data-free **CAFe** is conditional on having a large enough dataset at the server to reduce variance below that of the aggregated history.

4.6 Conclusion

We proposed two novel frameworks, **CAFe** and **CAFe-S**, to improve the efficiency of biased communication compression in distributed learning without relying on stateful clients or per-client error tracking. **CAFe** leverages the previous global update as a shared predictor, making client updates more compressible. **CAFe-S** extends this concept by using a candidate update generated from a server-side dataset, which can serve as a more accurate predictor. We provided convergence guarantees for both methods in the non-convex setting, highlighting a trade-off between the data-free convergence improvement of **CAFe** and the potentially faster convergence of **CAFe-S**, which leverages a representative server-side dataset to mitigate the impact of inter-client heterogeneity. Our experimental results for **CAFe** confirmed its practicality, showing significant improvements in convergence speed and model accuracy over standard compressed methods.

Chapter 5

Decentralized Parameter-Free Online Learning

5.1 Introduction

Online learning algorithms are essential for adapting to streaming data and changing environments, such as in online routing, ad selection, and spam filtering [92, 93]. In the standard Online Convex Optimization (OCO) framework, a learner makes a decision x_t from a convex set $\mathcal{X}$ at each round t . It then incurs a loss $l_t(x_t)$, determined by a convex loss function l_t that is unknown in advance. The goal is to minimize the **regret** after T rounds, which measures the cumulative loss difference between the learner's decisions and those of a single decision in hindsight

$$R_T(u) := \sum_{t=1}^T l_t(x_t) - \sum_{t=1}^T l_t(u), \tag{5.1}$$

where $u \in \mathcal{X}$ is an arbitrary comparator point.

The Challenge of Parameter Tuning

Classic algorithms like Online Gradient Descent (OGD) achieve optimal regret, but this performance hinges on careful **hyperparameter tuning** [92]. Typically, an algorithm’s learning rate must be set based on prior knowledge of problem parameters, such as the total time horizon T , bounds on the comparator’s norm $\|u\|$, or properties of the loss functions. This dependency is a significant practical drawback. In many real-world applications, these parameters are unknown, or the environment is non-stationary, making a single, fixed learning rate suboptimal. Incorrect tuning can lead to poor performance, requiring costly trial-and-error to find a suitable setting [94].

Parameter-Free Learning

To overcome this limitation, **parameter-free algorithms** have been developed [94–97]. These methods are based on an analogy to **coin-betting**, where minimizing regret is reframed as maximizing a gambler’s wealth in a sequential betting game. These algorithms achieve excellent bounds (which are optimal in some settings) without requiring any hyperparameter tuning. They automatically adapt to the characteristics of the problem, giving regret guarantees that depend on the unknown comparator u , hence why they are also known as comparator-adaptive algorithms [98, 99]. When the decision set $\mathcal{X}$ is unbounded and losses are L -Lipschitz, it has been shown that no algorithm can guarantee a regret better than $\Omega\left(\|u\| L \sqrt{T \ln(\|u\| LT + 1)}\right)$. Parameter-free algorithms can achieve this bound [95].

The Decentralized Setting

While parameter-free methods solve the tuning problem for a single learner, many modern applications require collaboration among multiple agents in a network. For example, today’s large-scale systems, from sensor networks to collaborative machine learning, are inherently **decentralized** [100, 101]. In these systems, a network of N agents must collaborate to solve a global learning task without a central coordinator. Each agent n makes its own decisions

$x_{n,t}$, incurs a local loss $l_{n,t}(x_{n,t})$, and can only communicate with its immediate neighbors over a network graph $\mathcal{G}$. The collective goal is to minimize the **network regret**¹

$$R_T^{\text{net}}(u) := \sum_{t=1}^T \frac{1}{N} \sum_{n=1}^N l_t(x_{n,t}) - \sum_{t=1}^T l_t(u), \quad (5.2)$$

where $l_t(x) = \frac{1}{N} \sum_{m=1}^N l_{m,t}(x)$ is the global loss function. This formulation measures the performance of the entire network's decisions against a single best-in-hindsight comparator u .

There have been recent efforts to solve variants of this problem in graphs where agents are available intermittently [102], or where a single agent is activated per round [99], but to the best of our knowledge, no parameter-free decentralized approach with *network regret guarantees* has been proposed to date. Note that the decentralized formulation described above also encompasses distributed scenarios with a central coordinator and various computing nodes. This is simply a special case of the topology of $\mathcal{G}$.

Contributions

This chapter introduces the first decentralized online learning algorithms that are parameter-free and come with network regret guarantees. We bridge the gap between adaptive single-agent learning and multi-agent coordination by combining the coin-betting framework with decentralized gossip protocols.

Our main contributions are:

- We propose a novel family of decentralized algorithms that allow agents to collaboratively minimize cumulative loss without any hyperparameter tuning.
- We provide theoretical analysis showing that our algorithms achieve sublinear regret,

¹We use the nomenclature from [102], but similar notions have been previously called *collective regret* [103].

comparable to centralized methods.

- We introduce a new *betting function* formulation that simplifies the analysis of coin-betting algorithms in the multi-agent context.

We also validate our approach with experiments on synthetic and real-world datasets, demonstrating its effectiveness in decentralized networks.

The remainder of the chapter is organized as follows. Section 5.2 introduces our proposed algorithm and its underlying intuition. Section 5.3 provides the necessary technical background for our analysis. Section 5.4 presents the main theoretical results and proofs. Section 5.5 provides operational details to ease implementation. Finally, Section 5.6 discusses our experimental findings, and Section 5.7 concludes the chapter.

5.2 The Decentralized Coin-Betting Algorithm

In this section, we present our decentralized parameter-free algorithm. We begin with a high-level and intuitive explanation of its mechanics before providing the formal algorithm description and performance analysis.

5.2.1 High-Level Idea: Local Betting and Global Gossiping

The main idea behind our approach is that each agent in the network plays a **personalized betting game**. The goal is to maximize wealth, which, through the lens of online learning theory, is equivalent to minimizing regret.

At each round, every agent n uses its accumulated knowledge of past losses (summarized in a vector $G_{n,t-1}$) to decide on its next action, $x_{n,t}$. This decision is analogous to a gambler **placing a bet**. The size and direction of the bet are determined by a parameter-free strategy derived from the coin-betting framework, which automatically adapts to observed

data without needing a learning rate.

After making a decision, each agent observes a local subgradient $g_{n,t}$ of its loss function. This subgradient acts as the “outcome” of the betting round, informing the agent how good its bet was. The agent then updates its internal state (its accumulated knowledge $G_{n,t}$) based on this new information.

If agents only acted on their local information, they might never achieve a globally good solution. To promote consensus, agents communicate with their neighbors after each round. They engage in a **gossip protocol**, where they exchange and average their internal states. This process spreads information across the network, ensuring that over time, all agents’ decisions are guided by the collective knowledge of the entire system.

The core of our contribution lies in exploiting the best properties of both tools. The coin-betting mechanism provides local, adaptive, and parameter-free decisions, while the gossip protocol provides a decentralized mechanism for global consensus.

5.2.2 Algorithm Description

Inspired by the coin-betting aspect, we name our algorithms DECO, for Decentralized Coin-betting. The formal procedure, performed at each agent, is detailed in Algorithm 5. Each agent n maintains a state vector $G_{n,t}$, which accumulates the subgradients it has observed, updated by the gossip process. At the start of each round t , the agent uses this state to compute its decision $x_{n,t}$.

We propose two closely related variants for making this decision:

- **DECO-i**: This version simply adds a gossip step to the standard coin-betting approach from [94], where the decision $x_{n,t}$ depends on both the agent’s state $G_{n,t-1}$ and its current wealth, $Wealth_{n,t-1}$. While effective, tracking and gossiping wealth adds

Algorithm 5 Decentralized Coin-Betting (DECO), Variants (i) and (ii)

Input: For each agent $n = 1, \dots, N$:

```
1:  $G_{n,0} = \mathbf{0}$ .
2: (i)  $\text{Wealth}_{n,0} = \epsilon > 0$ .
3: for  $t = 1, 2, \dots, T$  do
4:   for each agent  $n = 1, \dots, N$  in parallel do
5:     (i) Compute bet:  $x_{n,t} = \beta_t(G_{n,t-1}) \cdot \text{Wealth}_{n,t-1}$ .
6:     (ii) Compute bet using betting function:  $x_{n,t} = h_t(G_{n,t-1})$ .
7:     Play  $x_{n,t}$  and receive subgradient  $g_{n,t}$ .
8:     Update local accumulated gradient:  $\hat{G}_{n,t} = G_{n,t-1} - g_{n,t}$ .
9:     (i) Update local wealth:  $\text{Wealth}_{n,t} = \text{Wealth}_{n,t-1} - \langle g_{n,t}, x_{n,t} \rangle$ .
10:   end for
11:   Gossip Step: Perform  $q(t)$  rounds of gossip.
12:   for each agent  $n = 1, \dots, N$  in parallel do
13:      $G_{n,t} = \sum_{m=1}^N (W^{q(t)})_{nm} \hat{G}_{m,t}$ .
14:     (i)  $\text{Wealth}_{n,t} = \sum_{m=1}^N (W^{q(t)})_{nm} \text{Wealth}_{m,t}$ .
15:   end for
16: end for
```

complexity to the algorithm and analysis.

- **DECO-ii:** This is a novel formulation, which uses a **betting function** $h_t(\cdot)$ (defined in Section 5.3) to compute the decision $x_{n,t}$ solely from the state $G_{n,t-1}$. This simplifies the algorithm, as only the state vectors need to be gossiped, and makes the network regret analysis more tractable.

After receiving the subgradient $g_{n,t}$, each agent performs a local update on its state. Then, all agents execute one or more rounds of gossip by applying a gossip matrix W . This matrix is doubly stochastic and reflects the network topology, meaning agents only need to communicate with their direct neighbors. This gossip step averages the states across neighbors, driving the entire system toward consensus.

5.2.3 Summary of Main Results

Before diving into the detailed description and mathematical proofs, we discuss our main theoretical guarantees.

The Average Local Regret Is Independent of the Topology

Our first key result (Theorem 5.10) shows that the *average* of local regrets across all agents is sublinear and matches the bounds for centralized parameter-free algorithms. Remarkably, this guarantee holds regardless of the network’s topology or connectivity. Even if the agents never communicate ($W = I$), their average performance is still optimal. The only necessary condition is that the gossip matrix W must be doubly stochastic.

Network Regret and Disagreement

Our second key result (Theorem 5.14) characterizes the more challenging **network regret**. We prove that it consists of two components: the sublinear average local regret mentioned above, plus a non-negative **disagreement term**. The latter term measures the cumulative cost incurred because agents’ decisions differ from one another. For a complete graph, $\mathcal{G}$, the disagreement term is null, recovering *both* the single-agent scenario and the regret guarantees.

Controlling Disagreement with Gossip

Finally, we show that the disagreement term is controlled by the rate of information mixing in the network, which is determined by the spectral properties of the gossip matrix W and the number of gossip steps performed. By performing a number of gossip steps that grows slowly with time (e.g. linearly with the round number t), we can ensure that the disagreement term also grows sublinearly, leading to an overall sublinear network regret (Lemma 5.15). This shows that given enough communication, we can achieve sublinear network regret.

5.3 Preliminaries for the Analysis

To formally prove the guarantees stated in the previous section, we now introduce the key mathematical concepts upon which our algorithm is built. We describe each component (subgradients, coin-betting, and gossip) that plays a role in our framework separately.

5.3.1 Subgradients and Online Convex Optimization

Our analysis applies to the general problem of Online Convex Optimization (OCO). A standard technique for analyzing OCO problems is to linearize the convex loss functions using subgradients. This allows us to bound the regret of the convex problem by analyzing an equivalent Online Linear Optimization (OLO) problem.

DEFINITION 5.1 (Subgradient). *For a convex function $f : \mathcal{X} \rightarrow \mathbb{R}$, a vector $g \in \mathcal{X}$ is a subgradient of f at a point $x \in \mathcal{X}$ if for all $y \in \mathcal{X}$:*

$$f(y) \geq f(x) + \langle g, y - x \rangle.$$

This inequality implies that the regret is upper-bounded by the sum of linear losses

$$R_T(u) \leq \sum_{t=1}^T \langle g_t, x_t - u \rangle. \tag{5.3}$$

Our algorithm uses the subgradients $g_{n,t}$ of the local loss functions as the primary feedback signal for learning. Note that through online-to-batch conversion results [104], OLO algorithms can also solve convex stochastic optimization problems.

5.3.2 The Coin-Betting Framework

The coin-betting framework is the engine that drives the local, adaptive learning of each agent. It provides a powerful method for designing parameter-free algorithms by establishing

a duality between online learning and a sequential betting game. The formal connection is established by the reward-regret relationship, which states that an upper bound on regret is mathematically equivalent to a lower bound on cumulative reward (or wealth in the betting analogy).

LEMMA 5.2 (Reward-Regret Relationship [96]). *Let V, V^* be a pair of dual vector spaces. Let $F : V \rightarrow \mathbb{R} \cup \{+\infty\}$ be a proper convex lower semi-continuous function and let $F^* : V^* \rightarrow \mathbb{R} \cup \{+\infty\}$ be its Fenchel conjugate. Let $x_1, x_2, \dots, x_T \in V$ and $g_1, g_2, \dots, g_T \in V^*$. For any $\epsilon > 0$,*

$$\underbrace{\sum_{t=1}^T \langle g_t, x_t \rangle}_{\text{Reward}} \geq F\left(-\sum_{t=1}^T g_t\right) - \epsilon \quad (5.4)$$

if and only if

$$\forall u \in V^*, \quad \underbrace{\sum_{t=1}^T \langle g_t, x_t - u \rangle}_{\text{Regret}} \leq F^*(u) + \epsilon. \quad (5.5)$$

This lemma tells us that if we can guarantee that our cumulative earnings (reward) are lower-bounded by some function F of the cumulative outcomes, then our regret is automatically upper-bounded by its conjugate F^* .

The function F is constructed using a sequence of **coin-betting potentials**. These are carefully designed functions that represent a guaranteed lower bound on a gambler's wealth.

DEFINITION 5.3 (Coin-Betting Potential [94]). *Let $\epsilon > 0$. Let $\{F_t\}_{t=0}^\infty$ be a sequence of functions $F_t : (-a_t, a_t) \rightarrow \mathbb{R}_+$ where $a_t > t$. The sequence $\{F_t\}_{t=0}^\infty$ is called a **sequence of coin betting potentials for initial endowment ϵ** if it satisfies the following conditions:*

(a) $F_0(0) = \epsilon$.

(b) For every $t \geq 0$, $F_t(x)$ is even, logarithmically convex, strictly increasing on $[0, a_t)$, and $\lim_{x \rightarrow a_t} F_t(x) = +\infty$.

(c) For every $t \geq 1$, every $x \in [-(t-1), t-1]$, and every $c \in [-1, 1]$, it holds that

$$(1 + c\beta_t) F_{t-1}(x) \geq F_t(x + c), \quad (5.6)$$

where

$$\beta_t = \frac{F_t(x+1) - F_t(x-1)}{F_t(x+1) + F_t(x-1)}. \quad (5.7)$$

Moreover, it is an **excellent** sequence of coin betting potentials if it also satisfies

(d) For every $t \geq 0$, F_t is twice-differentiable and satisfies $x \cdot F_t''(x) \geq F_t'(x)$ for every $x \in [0, a_t)$.

Popular choices for these potentials, such as the Krichevsky-Trofimov (KT) potential or the exponential potential (see Table 5.1), lead to betting strategies that produce near-optimal regret bounds of the form $\mathcal{O}(\|u\| \sqrt{T \ln(1+T)})$.

Table 5.1: Mappings from $F_t(x)$ to $\beta_t(x)$.

Name	$F_t(x)$	$\beta_t(x)$
Exponential	$\frac{\epsilon}{\sqrt{t}} \exp\left(\frac{x^2}{2t}\right)$	$\tanh\left(\frac{x}{t}\right)$
Krichevsky-Trofimov	$\frac{2^t \cdot \Gamma\left(\frac{t+1}{2} + \frac{x}{2}\right) \cdot \Gamma\left(\frac{t+1}{2} - \frac{x}{2}\right)}{\pi \cdot t!}$	$\frac{x}{t}$

Betting Functions

For our decentralized analysis, we introduce a new tool called a **betting function**. This function, derived from the potentials, maps the accumulated subgradients directly to a decision (a bet). This decouples the betting decision from the agent’s wealth, simplifying the algorithm and its analysis.

DEFINITION 5.4 (Betting Function). *Let $\{F_t\}$ be a sequence of coin-betting potentials. The betting function at time $t > 0$ is defined as $h_t(x) = \beta_t(x) \cdot F_{t-1}(x)$, where $\beta_t(x)$ is the betting fraction derived from F_t .*

This function h_t is used in DECO-ii to determine $x_{n,t}$ based only on the gossiped state $G_{n,t-1}$.

5.3.3 Gossip Algorithms for Consensus

Gossip algorithms are the communication backbone of our method, enabling agents to coordinate and reach a consensus. They are decentralized protocols for averaging information across a network.

An agent’s communication with its neighbors is defined by a **gossip matrix** W .

DEFINITION 5.5 (Gossip Matrix). *Given a connected graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, a real $N \times N$ matrix W is a **gossip matrix** if it is doubly stochastic ($\sum_m W_{mn} = \sum_n W_{mn} = 1$), its non-zero entries correspond to the edges of the graph ($W_{mn} > 0$ only if $(m, n) \in \mathcal{E}$ or $m = n$), and its spectral radius $\rho(W - \frac{1}{N}\mathbf{1}\mathbf{1}^\top)$ is less than 1.*

Crucially, such a matrix can be constructed using only local information (e.g., via the Metropolis-Hastings method [105]), making it suitable for decentralized systems. When a vector of N values is multiplied by W , each agent’s value becomes a weighted average of its own and its neighbors’ values. Repeated application of W causes all agents’ values to converge to the network-wide average. The rate of this convergence is geometric and is

determined by the spectral properties of the network graph, captured by ρ .

LEMMA 5.6 (Gossip Convergence Rate [105]). *Let W be a gossip matrix and let $\bar{x} = \frac{1}{N} \sum_{n=1}^N x_{n,0}$. After k applications of W , the vector $x_k = W^k x_0$ converges to the average value $\bar{x} \otimes \mathbf{1}$:*

$$\|x_k - \bar{x} \otimes \mathbf{1}\| \leq \rho^k \|x_0 - \bar{x} \otimes \mathbf{1}\|. \quad (5.8)$$

This exponential convergence is what allows our algorithm to efficiently control the disagreement between agents and ensure the entire network collaborates effectively on the global learning task.

For our analysis, we will need the following corollary, proved using the triangle inequality

$$\|x_{n,k} - \bar{x}\| \leq \|x_k - \bar{x} \otimes \mathbf{1}\|.$$

COROLLARY 5.7 (Gossip Component-Wise Convergence Rate). *Given the conditions in Lemma 5.6, the components of the gossip iterations converge to the mean as*

$$\|x_{n,k} - \bar{x}\| \leq \rho^k \|x_0 - \bar{x}\|. \quad (5.9)$$

To ease the analysis, without loss of generality, we will often write that DECO does one round of gossip per round of coin-betting. The reasoning is identical for $q(t)$ rounds of gossip per round of coin-betting, since the multiplication of two gossip matrices is also a gossip matrix.

5.4 Analysis

5.4.1 Multi-Agent Betting Game

In this subsection, we extend the single-agent coin-betting framework to a multi-agent setting suitable for decentralized online learning.

Single-Agent Coin-Betting Recap

In the single-agent coin-betting game [94], a gambler starts with an initial wealth $\text{Wealth}_0 = \epsilon > 0$. At each round t , the gambler decides how much to bet x_t on the outcome of a coin flip. The coin outcome $c_t \in [-1, 1]$, which is a real number² between -1 and 1, satisfies $c_t = -g_t$, where g_t is the subgradient from the OLO formulation, assuming g_t is a scalar.³ The wealth updates according to

$$\text{Wealth}_t = \text{Wealth}_{t-1} + c_t x_t \quad (5.10)$$

$$= \text{Wealth}_{t-1} - g_t x_t. \quad (5.11)$$

By using a betting strategy based on coin-betting potentials F_t as in Definition 5.3, we can lower-bound the wealth and thereby upper-bound the regret through Lemma 5.2.

Extension to Multi-Agent Setting

In our multi-agent setting, each agent n acts as a gambler with initial wealth $\text{Wealth}_{n,0} = \epsilon > 0$. At each round t , each agent decides its own betting amount $x_{n,t}$ based on its local information. After the betting decision, agents exchange information with their neighbors via a gossip protocol.

More formally, for the *local betting strategy*, each agent n computes its betting fraction using the coin-betting potential

$$\beta_{n,t} = \frac{F_t(C_{n,t-1} + 1) - F_t(C_{n,t-1} - 1)}{F_t(C_{n,t-1} + 1) + F_t(C_{n,t-1} - 1)}, \quad (5.12)$$

²Although the outcome is not binary, we use the terminology of flipping a coin to be consistent with the existing literature.

³We refer the reader to [94] for the extension to vectors in a Hilbert space.

and sets

$$x_{n,t} = \beta_{n,t} \text{Wealth}_{n,t-1}. \quad (5.13)$$

Next, upon receiving the coin outcome $c_{n,t} \in [-1, 1]$, each agent updates its wealth and accumulated coin output

$$\hat{\text{Wealth}}_{n,t} = \text{Wealth}_{n,t-1} + c_{n,t} x_{n,t}, \quad (5.14)$$

$$\hat{C}_{n,t} = C_{n,t-1} + c_{n,t}. \quad (5.15)$$

Finally, agents exchange their wealth $\hat{\text{Wealth}}_{n,t}$, and accumulated coin outputs $\hat{C}_{n,t}$ with their neighbors. The updated variables after gossip are

$$\text{Wealth}_{n,t} = \sum_{m=1}^N W_{mn} \hat{\text{Wealth}}_{m,t}, \quad (5.16)$$

$$C_{n,t} = \sum_{m=1}^N W_{mn} \hat{C}_{m,t}. \quad (5.17)$$

Notice how the gossip exchange step ensures that agents' decisions and coin outputs are mixed and averaged across the network, preserving the network-wide accumulated coin output $\sum_{s=1}^t \bar{c}_s$, where $\bar{c}_t$ denotes the average coin outcome at time t .

Wealth Lower Bound in Multi-Agent Setting

Our goal is to establish a wealth lower bound similar to the single-agent case. Specifically, we aim to show that

$$\text{Wealth}_{n,t} \geq F_t(C_{n,t}). \quad (5.18)$$

By averaging this equation across all agents, we obtain

$$\overline{\text{Wealth}}_t = \frac{1}{N} \sum_{n=1}^N \text{Wealth}_{n,t} \quad (5.19)$$

$$\geq \frac{1}{N} \sum_{n=1}^N F_t(C_{n,t}) \quad (5.20)$$

$$\geq F_t \left(\frac{1}{N} \sum_{n=1}^N C_{n,t} \right) \quad (5.21)$$

$$= F_t \left(\sum_{s=1}^t \bar{c}_s \right), \quad (5.22)$$

where we have used the log-convexity of F_t and the fact that the average of the accumulated coin outputs is equal to the accumulated average coin output.

To prove Equation (5.18), we proceed by induction.

LEMMA 5.8 (Wealth Lower Bound in Multi-Agent Setting). *Under the proposed multi-agent betting game and assuming the coin-betting potential F_t satisfies the conditions in Definition 5.3, the wealth of each agent satisfies:*

$$\text{Wealth}_{n,t} \geq F_t(C_{n,t}), \quad (5.23)$$

as long as the gossip matrix W is doubly stochastic.

Proof. We use induction on t . For the base case, $t = 0$, for any agent n , we have $\text{Wealth}_{n,0} = \epsilon = F_0(0)$ and $C_{n,0} = 0$, so the inequality holds.

For the inductive step, we assume the inequality holds at time $t - 1$. At time t , the wealth

update at Agent n is given by:

$$\text{Wealth}_{n,t} = \sum_{m=1}^N W_{mn} \hat{\text{Wealth}}_{m,t} \quad (5.24)$$

$$= \sum_{m=1}^N W_{mn} (\text{Wealth}_{m,t-1} + c_{m,t} x_{m,t}) \quad (5.25)$$

$$= \sum_{m=1}^N W_{mn} \text{Wealth}_{m,t-1} (1 + \beta_{m,t} c_{m,t}) \quad (5.26)$$

$$\geq \sum_{m=1}^N W_{mn} F_{t-1}(C_{m,t-1}) (1 + \beta_{m,t} c_{m,t}). \quad (5.27)$$

The last step is derived by the induction hypothesis, $\text{Wealth}_{m,t-1} \geq F_{t-1}(C_{m,t-1})$. Using the coin-betting potential property (c) in Definition 5.3, we have:

$$F_{t-1}(C_{m,t-1}) (1 + \beta_{m,t} c_{m,t}) \geq F_t(C_{m,t-1} + c_{m,t}) \quad (5.28)$$

$$= F_t(\hat{C}_{m,t}). \quad (5.29)$$

Since F_t is logarithmically convex and W is doubly stochastic, applying Jensen's inequality yields:

$$\text{Wealth}_{n,t} \geq \sum_{m=1}^N W_{mn} F_t(\hat{C}_{m,t}) \quad (5.30)$$

$$\geq F_t \left(\sum_{m=1}^N W_{mn} \hat{C}_{m,t} \right) \quad (5.31)$$

$$= F_t(C_{n,t}). \quad (5.32)$$

□

This lemma shows that each agent's wealth is lower bounded by the coin-betting potential evaluated at its accumulated coin output. This relationship forms the basis for the regret bounds of our algorithm.

REMARK 5.9. *Note that Lemma 5.8 also applies for a betting-function based decision. The proof follows identically.*

5.4.2 Regret Analysis

Using Lemma 5.8, we can derive regret bounds for our algorithms. Note that we have shown its proof for the case where $\{g_t\}$ are scalars, but the proof can be extended to elements of any Hilbert space $\mathcal{H}$ as in [94, Theorem 3].

THEOREM 5.10 (Average Local Regret Bound). *Assuming that the coin-betting potential F_t satisfies the conditions in Definition 5.3, both versions of DECO achieve the following regret bound for any comparator $u \in \mathcal{H}$:*

$$R_T^{local}(u) := \frac{1}{N} \sum_{n=1}^N \sum_{t=1}^T l_{n,t}(x_{n,t}) - l_{n,t}(u) \quad (5.33)$$

$$\leq F_T^*(u) + \epsilon, \quad (5.34)$$

where F_T^* is the Fenchel conjugate of F_T , and $\epsilon > 0$ is the initial endowment.

Proof. Notice that, due to convexity, we have:

$$\frac{1}{N} \sum_{n=1}^N \sum_{t=1}^T l_{n,t}(x_{n,t}) - l_{n,t}(u) \leq \frac{1}{N} \sum_{n=1}^N \sum_{t=1}^T \langle g_{n,t}, x_{n,t} - u \rangle.$$

Now, similar to the reward-regret relationship (Lemma 5.2) and the wealth lower bounds established in Lemma 5.8, we can relate the cumulative wealth to the linear regret. Specifically, since Equation (5.22) holds and the gossip step preserves averages, we set $c_t = -g_t$

and obtain

$$\frac{1}{N} \sum_{n=1}^N \sum_{t=1}^T \langle g_{n,t}, x_{n,t} \rangle = -\overline{\text{Wealth}}_T \quad (5.35)$$

$$\leq -F_T \left(-\sum_{t=1}^T \bar{g}_t \right) + \epsilon. \quad (5.36)$$

Therefore,

$$\frac{1}{N} \sum_{n=1}^N \sum_{t=1}^T \langle g_{n,t}, x_{n,t} - u \rangle \quad (5.37)$$

$$\leq -\frac{1}{N} \sum_{n=1}^N \sum_{t=1}^T \langle g_{n,t}, u \rangle - F_T \left(-\sum_{t=1}^T \bar{g}_t \right) + \epsilon \quad (5.38)$$

$$= \left\langle -\sum_{t=1}^T \bar{g}_t, u \right\rangle - F_T \left(-\sum_{t=1}^T \bar{g}_t \right) + \epsilon \quad (5.39)$$

$$\leq \sup_{\theta \in \mathcal{H}} \langle \theta, u \rangle - F_T(\theta) + \epsilon \quad (5.40)$$

$$= F_T^*(u) + \epsilon. \quad (5.41)$$

□

REMARK 5.11. Applying [94, Corollary 5], any potential from Table 5.1 satisfies

$$R_T^{\text{local}}(u) \leq \|u\| \sqrt{T \ln \left(1 + \frac{24T^2 \|u\|^2}{\epsilon^2} \right)} + \epsilon,$$

which gives sublinear regret when $\epsilon = \mathcal{O}(\sqrt{T})$.

REMARK 5.12. Note that the regret bound in Theorem 5.10 only uses the double stochasticity of the gossip matrix W , so it holds for any gossip algorithm that preserves averages. Notably, this includes the case where agents do not communicate with each other, that is, $W = I_N$. This means that **no matter the network topology**, the average local loss is minimized.

The effect of the network topology is reflected in the average network regret, which is analyzed in Theorem 5.14.

We now prove that the network regret is related to the local regret, with a disagreement term that depends on the network topology. This disagreement will become larger when the output decisions at each node differ more, which in turn is a consequence of the difference between the node losses. The output decisions are a function of the accumulated subgradient variable at each node. These are gossiped at each round and their difference is bounded as follows.

LEMMA 5.13. *Let W be a gossip matrix as in Definition 5.5. Let $G_{n,t}$ be the accumulated subgradient at Agent n , and let $g_t = (g_{1,t}, \dots, g_{N,t})$ be the vector of subgradients at time t . If we perform $q(t)$ gossip steps at each round t , the error with respect to the average accumulated subgradient at time t evolves as*

$$\|G_{n,t} - \bar{G}_t\| \leq \sum_{s=1}^t \rho^{Q(s,t)} \|g_s - \bar{g}_s \otimes \mathbf{1}\|, \quad (5.42)$$

where $\rho < 1$ is the spectral radius of $W - \frac{1}{N}\mathbf{1}\mathbf{1}^\top$, and

$$Q(s, t) = \sum_{r=s}^t q(r). \quad (5.43)$$

Since each $g_{n,t} \in (-1, 1)$, we can further bound

$$\|G_{n,t} - \bar{G}_t\| \leq \sqrt{N} \sum_{s=1}^t \rho^{Q(s,t)}. \quad (5.44)$$

Proof. The lemma immediately follows from the triangle inequality and Lemma 5.6. $\square$

THEOREM 5.14 (Network loss regret bound). *Let $\{F_t\}$ be a family of excellent coin-betting potentials as in Definition 5.3, and let W be a gossip matrix as in Definition 5.5. Then, the*

proposed DECO-ii achieves the following regret bound for any comparator $u \in \mathcal{H}$:

$$R_T^{\text{net}}(u) = \frac{1}{N} \sum_{n=1}^N R_T^{\text{local}}(u) + \underbrace{2\sqrt{N} \sum_{t=1}^T L_{h_t} \sum_{s=1}^{t-1} \rho^{Q(s,t)}}_{\text{Disagreement}},$$

where L_{h_t} is the Lipschitz constant of h_t in $[-(t-1), (t-1)]$.

Proof. Using the definition of network regret in Equation (5.2), we have:

$$R_T^{\text{net}}(u) = \sum_{t=1}^T \frac{1}{N} \sum_{n=1}^N l_t(x_{n,t}) - \sum_{t=1}^T l_t(u). \quad (5.45)$$

Next, we add and subtract the average local loss to obtain:

$$R_T^{\text{net}}(u) = \sum_{t=1}^T \frac{1}{N} \sum_{n=1}^N l_t(x_{n,t}) \pm \sum_{t=1}^T \frac{1}{N} \sum_{n=1}^N l_{n,t}(x_{n,t}) - \sum_{t=1}^T l_t(u),$$

which we can re-arrange to

$$R_T^{\text{net}}(u) = \frac{1}{N} \sum_{n=1}^N R_T^{\text{local}}(u) + \underbrace{\sum_{t=1}^T \frac{1}{N} \sum_{n=1}^N (l_t(x_{n,t}) - l_{n,t}(x_{n,t}))}_{\text{Disagreement}}.$$

Note that the local regret term can be bounded using Theorem 5.10, so we just have to bound the disagreement term, which we will denote as D . We can further re-arrange the

disagreement term as

$$D := \sum_{t=1}^T \frac{1}{N} \sum_{n=1}^N \frac{1}{N} \sum_{m=1}^N (l_{n,t}(x_{m,t}) - l_{n,t}(x_{n,t})) . \quad (5.46)$$

Note that we can now use the 1-Lipschitz property to bound each term, that is,

$$\begin{aligned} l_{n,t}(x_{m,t}) - l_{n,t}(x_{n,t}) &\leq \|x_{m,t} - x_{n,t}\| \\ &= \|h_t(G_{m,t-1}) - h_t(G_{n,t-1})\| , \end{aligned}$$

where we have used the definition of $x_{m,t}$ according to DECO-**ii**. Next, we use the property that h_t is L_{h_t} -Lipschitz in its domain, and we obtain

$$\|x_{m,t} - x_{n,t}\| \leq L_{h_t} \|G_{m,t-1} - G_{n,t-1}\| . \quad (5.47)$$

Now, using the triangle inequality and Lemma 5.13, we derive

$$\|x_{m,t} - x_{n,t}\| \leq L_{h_t} \cdot 2\sqrt{N} \sum_{s=1}^{t-1} \rho^{Q(s,t)} . \quad (5.48)$$

Finally, we plug this back into Equation (5.46) to show

$$D \leq 2\sqrt{N} \sum_{t=1}^T L_{h_t} \sum_{s=1}^{t-1} \rho^{Q(s,t)} . \quad (5.49)$$

□

LEMMA 5.15 (Square-Root Disagreement Bound). *Let W be a gossip matrix as in Definition 5.5, where ρ is the spectral radius of $W - \frac{1}{N}\mathbf{1}\mathbf{1}^\top$, and let $\{F_t\}$ be a family of excellent coin-betting potentials, as in Definition 5.3, where $\{L_{h_t}\}$ are the Lipschitz constants of $\{h_t\}$*

derived from $\{F_t\}$ in $[-(t-1), (t-1)]$. If we perform $q(t)$ gossip steps at round t , such that

$$\forall t \geq 1, \quad L_{h_t} \sum_{s=1}^{t-1} \rho^{Q(s,t)} \leq C \cdot \frac{1}{\sqrt{t}}, \quad (5.50)$$

where $Q(s, t) := \sum_{r=s}^t q(r)$, we obtain sublinear regret and the disagreement is upper-bounded by $4C\sqrt{N}\sqrt{T}$.

Proof. From Theorem 5.14, the disagreement term D is upper bounded as

$$D \leq 2\sqrt{N} \sum_{t=1}^T L_{h_t} \sum_{s=1}^{t-1} \rho^{Q(s,t)}. \quad (5.51)$$

Given the condition (5.50), we can upper bound

$$D \leq 2\sqrt{N} \sum_{t=1}^T C \cdot \frac{1}{\sqrt{t}}. \quad (5.52)$$

Note that

$$\sum_{t=1}^T \frac{1}{\sqrt{t}} := H_{T,1/2} \quad (5.53)$$

is the generalized harmonic number $H_{T,1/2}$. From the classic square root reciprocals sum inequality,

$$H_{T,1/2} \leq 2\sqrt{T} - 1, \quad (5.54)$$

and, thus, we obtain that

$$D \leq 4C\sqrt{N}\sqrt{T}. \quad (5.55)$$

□

THEOREM 5.16 (Exponential Potentials and Sublinear Disagreement). *DECO-ii*, choosing $\{F_t\}$ to be the exponential potentials from Table 5.1, defined as

$$F_t(x) = \frac{\epsilon}{\sqrt{t}} \exp\left(\frac{x^2}{2t}\right), \quad (5.56)$$

and choosing the gossip step function to be

$$q(t) = \lceil c \cdot t \rceil, \quad (5.57)$$

for any $c \geq \frac{-3}{2 \ln \rho}$ has sublinear regret and disagreement.

Proof. The betting function derived from the exponential potential is

$$h_t(x) = \tanh\left(\frac{x}{t}\right) \cdot F_{t-1}(x), \quad (5.58)$$

which is odd and has derivative

$$h'_t(x) = F_t(x) \frac{\operatorname{sech}^2 \frac{x}{t} + x \tanh \frac{x}{t}}{t}. \quad (5.59)$$

For $t \geq 1$, and $|x| \leq t - 1$, we can derive that

$$|h'_t(x)| \leq F_t(t-1) \left(1 - \frac{2}{1+e^2}\right) \quad (5.60)$$

$$\leq \frac{\epsilon}{\sqrt{t}} \exp\left(\frac{(t-1)^2}{2t}\right) \cdot \left(1 - \frac{2}{1+e^2}\right), \quad (5.61)$$

and we can use this quantity as L_{h_t} , needed in Lemma 5.15. The proof is complete if Lemma 5.15 holds and we apply it with $C := \epsilon \cdot \left(1 - \frac{2}{1+e^2}\right)$. To use Lemma 5.15, we need to show the

validity of Ineq. (5.50), i.e.,

$$\forall t \geq 1, \quad \exp \left(\frac{(t-1)^2}{2t} \right) \sum_{s=1}^{t-1} \rho^{Q(s,t)} \leq 1.$$

For $t = 1$, Ineq. (5.50) is trivial. Next, for $t \geq 2$, we can ensure that

$$\exp \left(\frac{(t-1)^2}{2t} \right) \sum_{s=1}^{t-1} \rho^{Q(s,t)} \leq \exp \left(\frac{t}{2} \right) \sum_{s=1}^{t-1} \rho^{Q(s,t)},$$

so the condition is satisfied if

$$e^{t/2} \sum_{s=1}^{t-1} \rho^{Q(s,t)} \leq 1.$$

Now, let $q(t) = c \cdot t$. In practice, we can take the ceiling of this quantity, since the number of gossip iterations has to be an integer. For ease of analysis, we will disregard the ceiling, but the logic follows analogously. Notice that we can express

$$Q(s, t) = c \cdot \frac{(t-s+1)(s+t)}{2}, \tag{5.62}$$

which is smallest when $s = t - 1$. Thus, we can upper bound

$$e^{t/2} \sum_{s=1}^{t-1} \rho^{Q(s,t)} \leq e^{t/2} (t-1) \rho^{c(2t-1)}.$$

We want to ensure that

$$e^{t/2} (t-1) \rho^{c(2t-1)} \leq 1$$

for all $t \geq 2$. Taking logarithms, we obtain

$$\frac{t}{2} + \ln(t-1) + \frac{c \ln \rho}{2} (2t-1) \leq 0 \tag{5.63}$$

Notice that if the LHS is non-increasing, we only need to verify Ineq. (5.63) at $t = 2$. The derivative of the LHS is

$$\frac{1}{2} + \frac{1}{t-1} + c \ln(\rho),$$

and ensuring

$$c \geq \frac{-3}{2 \ln(\rho)}, \quad (5.64)$$

it is non-positive for all $t \geq 2$, which ensures the non-increasing property. If Ineq. (5.64) is satisfied, we can substitute $t = 2$ in Ineq. (5.63) and verify that the bound holds, which concludes our proof. $\square$

THEOREM 5.17 (KT Potentials and Sublinear Disagreement). *DECO-ii*, choosing $\{F_t\}$ to be the KT potentials from Table 5.1, defined as

$$F_t(x) = \epsilon \frac{2^t \cdot \Gamma\left(\frac{t+1}{2} + \frac{x}{2}\right) \cdot \Gamma\left(\frac{t+1}{2} - \frac{x}{2}\right)}{\pi \cdot t!}, \quad (5.65)$$

and choosing the gossip step function to be

$$q(t) = \lceil c \cdot t \rceil, \quad (5.66)$$

for any $c \geq \frac{-2 \ln(2)}{\ln(\rho)}$ has sublinear regret and disagreement.

Proof. The betting function derived from the KT potential is

$$h_t(x) = \frac{x}{t} \cdot F_{t-1}(x). \quad (5.67)$$

Following a similar procedure to the one in Theorem 5.16, we obtain that for $t \geq 1$, and

$|x| \leq t - 1$, $|h'_t|$ is upper bounded by

$$\frac{2^{t-2}(\ln t - \psi(1/2))}{\sqrt{\pi t}}, \quad (5.68)$$

where ψ is the digamma function. Thus, we can set $C := 4\sqrt{\pi}$, and apply Lemma 5.15. For a valid Lemma 5.15, we need to verify Ineq. (5.50), which in this case means ensuring

$$\forall t \geq 1, \quad 2^t(\ln t - \psi(1/2)) \sum_{s=1}^{t-1} \rho^{Q(s,t)} \leq 1. \quad (5.69)$$

We set $q(t) = c \cdot t$ and check that

$$2^t(\ln t - \psi(1/2))(t - 1)\rho^{c(2t-1)} \leq 1. \quad (5.70)$$

For $t = 1$, this is trivial. For $t \geq 2$, we can take logarithms and obtain

$$t \ln(2) + \ln(\ln t - \psi(1/2)) + \ln(t - 1) + \frac{c \ln \rho}{2}(2t - 1) \leq 0.$$

Next, we use the bound

$$t \ln(2) + \ln(\ln t - \psi(1/2)) + \ln(t - 1) \leq 2t \ln(2)$$

for all $t \geq 2$, and the condition simplifies to

$$2t \ln(2) + \frac{c \ln \rho}{2}(2t - 1) \leq 0,$$

which is satisfied when $c \geq \frac{-2 \ln(2)}{\ln(\rho)}$. This concludes our proof. $\square$

REMARK 5.18. *Theorems 5.16 and 5.17 provide a clear trade-off between communication and performance. They establish that by performing $\mathcal{O}(t)$ gossip steps at each round, a*

communication cost that grows linearly with time, the disagreement term can be controlled to be at most in the order of the local regret term, thus ensuring sublinear network regret.

5.5 Operational Details

To bridge the gap between our theoretical analysis and practical implementation, this section details the operational flow of DECO. We illustrate how the algorithm operates in a fully decentralized manner using an online linear regression task, which is common in applications like distributed sensing, as an example.

Consider a weather sensor network deployed in a large agricultural field. Each sensor acts as an agent, forming a wireless mesh network with nearby sensors. The collective goal is to learn a single robust model to predict a value, such as air temperature or soil moisture, based on other local readings.

At each time step t , every sensor n makes a prediction using its current model parameters $x_{n,t}$ and a vector of local features $z_{n,t}$ (e.g., [humidity, pressure]). After observing the true value of the temperature or soil moisture, $y_{n,t}$, it computes its loss. For this example, we use the absolute loss

$$\ell_{n,t}(x) = |\langle x, z_{n,t} \rangle - y_{n,t}|, \quad (5.71)$$

with subgradient

$$g_{n,t} = \text{sign}(\langle x, z_{n,t} \rangle - y_{n,t}) \cdot z_{n,t}. \quad (5.72)$$

The sensors then use this subgradient to update their internal state and communicate with neighbors to collaboratively refine their models.

The operational flow at each agent is a four-step loop. First, the agent makes a **prediction**

by computing its decision $x_{n,t}$ from its current internal state. Following this, it **observes** feedback from the environment (e.g., by measuring the true temperature or soil moisture) and computes the local subgradient $g_{n,t}$. This subgradient is then used to perform a **local update** on the agent's internal state, creating a temporary version. The round concludes by **communicating** this temporary state with the agent's direct neighbors through a gossip protocol, averaging their information to produce the final state for the next round.

The key to decentralization lies in the communication step. This is a multi-round process where information propagates across the network. For $q(t)$ rounds of gossip, each agent n only exchanges its current state with its immediate neighbors $\mathcal{N}(n)$. The state is updated using a weighted average, with weights defined by the gossip matrix W . These weights can be constructed locally using, for example, the Metropolis-Hastings rule, which only requires an agent to know its own degree and the degrees of its neighbors:

$$W_{mn} = \begin{cases} 1 - \sum_{j \in \mathcal{N}(n) \setminus \{n\}} W_{jn}, & \text{if } m = n, \\ \frac{1}{\max\{d_n, d_m\} + 1}, & \text{if } m \in \mathcal{N}(n), \\ 0, & \text{otherwise.} \end{cases} \quad (5.73)$$

After $q(t)$ local exchanges, the agent has its final state for round t , ready for the next prediction.

In practice, the two variants of DECO differ in what constitutes the agent's state and how the decision is computed, which in turn affects the local update and communication steps.

In Figure 5.1, we illustrate the operational loop for DECO-**i**, which directly adapts the standard single-agent coin-betting algorithm. In this variant, an agent's state consists of its wealth and accumulated gradients, $(\text{Wealth}_{n,t-1}, G_{n,t-1})$. The prediction is computed from

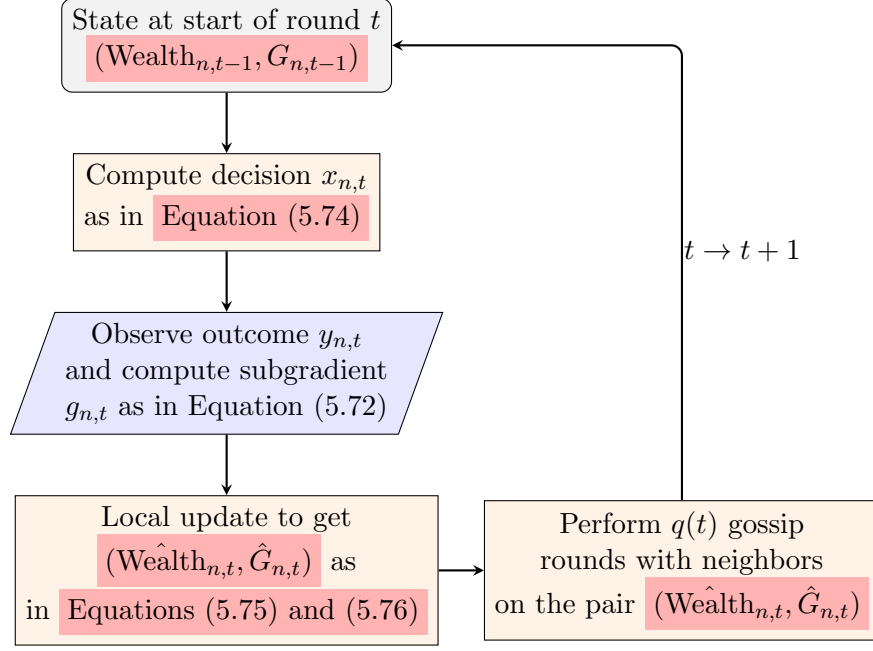


Figure 5.1: Operational loop for an agent using DECO-**i**. Key elements specific to this variant, related to tracking and communicating wealth, are highlighted in red.

this pair. For the exponential potential, it is

$$x_{n,t} = \tanh\left(\frac{\|G_{n,t-1}\|}{t}\right) \cdot \text{Wealth}_{n,t-1} \cdot \frac{G_{n,t-1}}{\|G_{n,t-1}\|}. \quad (5.74)$$

The local update then affects both state components:

$$\hat{\text{Wealth}}_{n,t} = \text{Wealth}_{n,t-1} - \langle g_{n,t}, x_{n,t} \rangle, \quad (5.75)$$

$$\hat{G}_{n,t} = G_{n,t-1} + g_{n,t}. \quad (5.76)$$

For communication, the agent gossips the pair $(\hat{\text{Wealth}}_{n,t}, \hat{G}_{n,t})$. Although this increases the communication load, the wealth is a scalar, which is often negligible compared to the dimension of the gradient vector.

Figure 5.2 depicts the operational loop for DECO-**ii**, which is designed to be more communication-efficient and analytically tractable. However, the performance of DECO-**ii** is often worse than that of DECO-**i**. Here, the agent's state only consists of the accumulated gradients $G_{n,t-1}$.

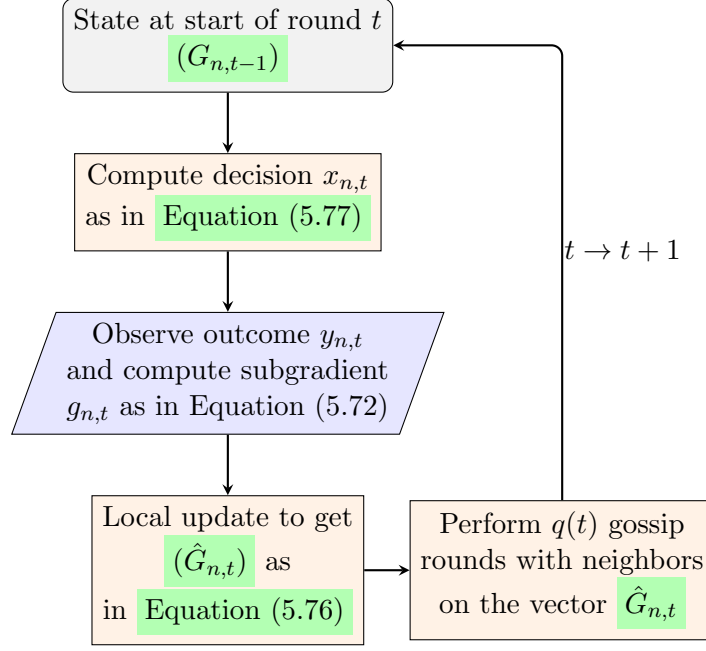


Figure 5.2: Operational loop for DECO-ii. The process is simplified by decoupling from wealth, requiring only the accumulated gradients to be maintained and gossiped. Differences from DECO-i are highlighted in green.

The prediction is computed directly from this state using the betting function $h_t(\cdot)$. For the exponential potential, this is

$$x_{n,t} = \underbrace{\tanh\left(\frac{\|G_{n,t-1}\|}{t}\right)}_{h_t(\|G_{n,t-1}\|)} \cdot \frac{\epsilon e^{\frac{\|G_{n,t-1}\|^2}{2(t-1)}}}{\sqrt{t-1}} \cdot \frac{G_{n,t-1}}{\|G_{n,t-1}\|}. \quad (5.77)$$

Then, the local update is only applied to the accumulated gradient vector as in Equation (5.76). Thus, the agent only needs to gossip the vector $\hat{G}_{n,t}$.

5.6 Experiments

To validate our theoretical findings, we conduct a series of experiments on both synthetic and real-world datasets. Our implementation is publicly available.⁴ All experiments compare our decentralized coin-betting algorithms, which we will refer to as **DECO-i** and **DECO-ii**,

⁴Code available at: <https://github.com/TomasOrtega/Deco>

against standard baselines.

5.6.1 Experimental Setup

Algorithms Compared

We evaluate the performance of our two proposed variants from DECO, with two different potential functions from Table 5.1. Therefore, we have four distinct parameter-free algorithms:

- **DECO-i (exp)**: Version (i) using the exponential potential, generating a tanh betting fraction.
- **DECO-ii (exp)**: Version (ii) using the exponential potential.
- **DECO-i (KT)**: Version (i) using the KT potential, based on the Krichevsky-Trofimov bettor, which produces a linear betting fraction, $\beta_t(x) = x/t$.
- **DECO-ii (KT)**: Version (ii) using the KT potential.

We compare them against two baselines:

- **Decentralized Online Gradient Descent (DOGD)**: A standard decentralized algorithm where each agent performs a gradient step followed by a gossip step. The learning rate follows a decreasing schedule $\eta_t = \eta_0/\sqrt{t}$.
- **Centralized**: An idealized oracle in which a single coin-betting agent (using the KT potential) has access to the average gradient $\bar{g}_t = \frac{1}{N} \sum_{n=1}^N g_{n,t}$ at each round.

Network and Evaluation

Unless otherwise specified, all experiments are conducted in a network of $N = 20$ agents. Performance is measured by **cumulative network loss**, defined as $\sum_{t=1}^T \frac{1}{N} \sum_{n=1}^N l_t(x_{n,t})$.

For the gossip step, we use a single round of communication ($q(t) = 1$) by default to represent a communication-constrained scenario.

Datasets

We use both a synthetic dataset for controlled experiments and several real-world datasets to test for robustness.

- **Synthetic data:** We simulate an online linear regression task. First, a true parameter vector $u^* \in \mathbb{R}^{10}$ is drawn from a standard normal distribution and fixed. To model data heterogeneity, each agent n is assigned a unique feature center $\mu_n \in \mathbb{R}^{10}$, drawn from a normal distribution. At each round t , a common base feature vector z'_t is drawn from a standard normal distribution. Agent n then observes a personalized feature vector $z_{n,t} = z'_t + \mu_n$, which is subsequently normalized to have a unit ℓ_2 -norm ($\|z_{n,t}\|_2 = 1$). Each agent n observes a label $y_{n,t} = \langle u^*, z_{n,t} \rangle + \nu_{n,t}$, where $\nu_{n,t}$ is a small Gaussian noise term. The loss function is the absolute loss, $\ell_{n,t}(x) = |\langle x, z_{n,t} \rangle - y_{n,t}|$.
- **Real-world data:** We use four regression datasets from the LIBSVM library: `cadata`, `cpusmall`, `space_ga`, and `abalone` [5]. The features of these datasets are pre-normalized.

Implementation Note

A notable challenge while computing quantities involving factorial or Gamma functions, such as in the KT potential shown in Table 5.1, is numerical stability. Direct computation of the Gamma function, $\Gamma(z)$, can easily lead to numerical overflow for even moderately large arguments.

To address this, the KT potential can be expressed in a more stable form using the Beta function, $B(a, b) = \frac{\Gamma(a)\Gamma(b)}{\Gamma(a+b)}$. Namely,

$$F_t(x) = \frac{\epsilon \cdot 2^t}{\pi} B\left(\frac{t+1+|x|}{2}, \frac{t+1-|x|}{2}\right). \quad (5.78)$$

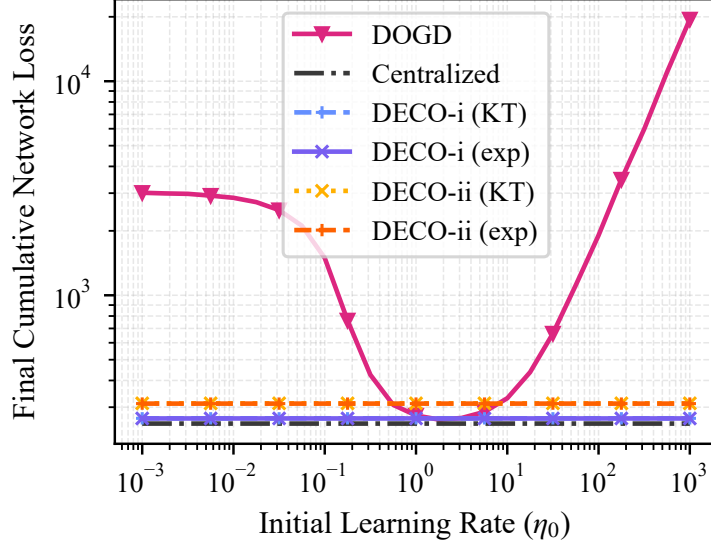


Figure 5.3: Final cumulative network loss of DOGD as a function of its initial learning rate (η_0) on synthetic data, compared to the constant performance of our parameter-free DECO algorithms. The U-shaped curve for DOGD highlights its extreme sensitivity to hyperparameter choice, where a suboptimal setting leads to loss that is orders of magnitude worse. Our DECO algorithms achieve good performance without any tuning, regardless of the potential function used.

However, even the Beta function can underflow to zero. Our implementation avoids these issues by performing calculations in the log-domain:

$$\ln F_t(x) = \ln(\epsilon/\pi) + t \ln(2) + \ln B(a, b). \quad (5.79)$$

The term $\ln B(a, b)$ is computed using a highly optimized log-beta function, Python’s `scipy.special.betaln`, which sidesteps the intermediate calculation of large or small numbers. The final value of $F_t(x)$ is then recovered by exponentiating the result.

5.6.2 The Challenge of Hyperparameter Tuning

A primary motivation for our work is the brittleness of traditional algorithms that require manual tuning. To illustrate this, we first conduct an experiment focusing on the sensitivity of DOGD to its initial learning rate, η_0 . We run the DOGD algorithm on our synthetic

dataset over a wide range of learning rates, from 10^{-3} to 10^3 , and plot the final cumulative network loss after $T = 3000$ iterations.

As shown in Figure 5.3, the performance of DOGD highly depends on η_0 . The loss forms a sharp “U” shape, indicating that only a narrow range of values leads to good performance. In contrast, the performance of all four DECO variants is represented by flat horizontal lines, as they do not require such tuning. This plot showcases the advantage of a parameter-free approach, which reliably achieves low loss.

Since KT and exponential potentials provide very similar performance, for clarity of the plots, we only show the centralized results using the KT potential. Also, since the difference between the two potential functions for DECO is small, in what follows, we only focus on results using the KT potential.

5.6.3 Impact of Network Connectivity

Our theoretical analysis in Theorem 5.14 shows that network regret is influenced by a disagreement term, which should decrease as network connectivity improves. To test this empirically, we evaluate the DECO-ii (KT) algorithm on several Erdős-Rényi (ER) random graphs [106] with different edge probabilities, p . A higher p results in a denser graph with a smaller spectral radius ρ , facilitating faster information mixing.

The results are presented in Figure 5.4. The top panel shows that, as predicted by the theory, the cumulative network loss decreases with better connectivity. On the sparsely connected graph ($p = 0.1$), the cumulative loss is highest due to slower consensus. Note that there is an initial jump that results in the deviation of the gossiped algorithms from the centralized oracle, which is due to the lag time of the gossiped algorithms in achieving consensus. As connectivity increases ($p = 0.3$ and $p = 1.0$), the growth of the cumulative loss curves approaches that of the centralized oracle, demonstrating that improved communication directly

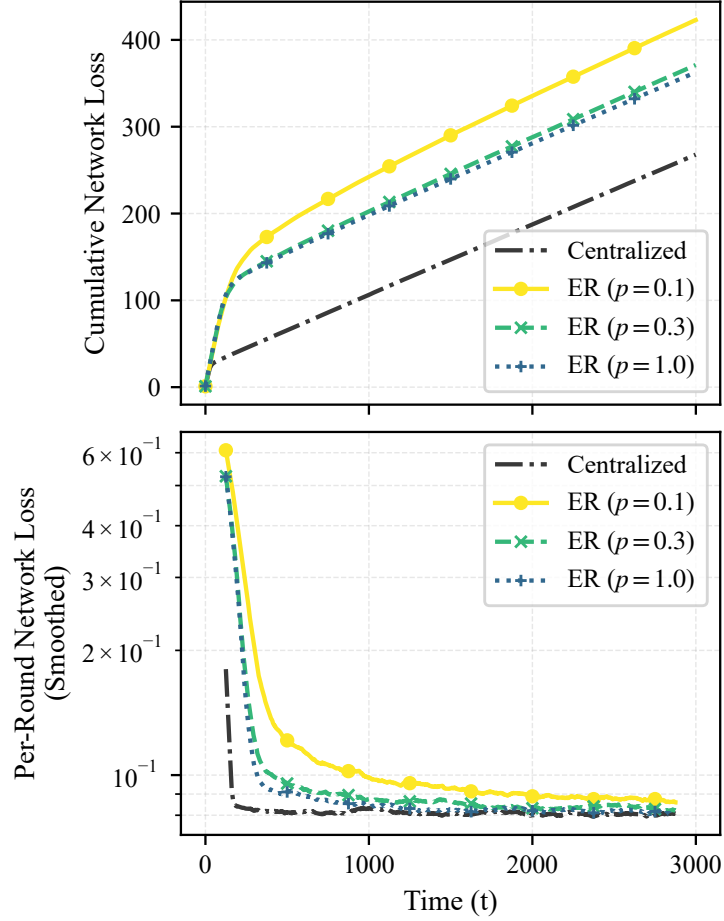


Figure 5.4: Performance of DECO-ii (KT) on synthetic data for Erdős-Rényi random graphs with varying edge probability p . **Top:** Cumulative network loss. Cumulative loss growth improves with better connectivity, as faster information mixing reduces the disagreement among agents. **Bottom:** Per-round network loss (smoothed with a 250-round moving average). The smoothed loss shows that more connected networks not only accumulate less total loss but also converge faster to the centralized ideal’s steady-state per-round loss.

mitigates the cost of decentralization.

The bottom panel, which displays the smoothed per-round network loss, provides further insight. It illustrates that not only do more connected networks accumulate less total loss, but they also converge faster to the steady-state loss rate of the centralized ideal.

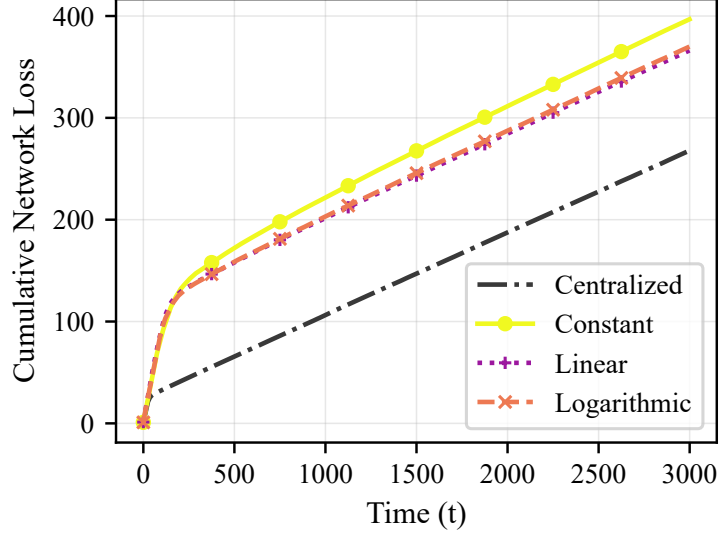


Figure 5.5: Cumulative network loss of DECO-ii (KT) on a cycle graph with different gossip schedules, $q(t)$. Increasing the communication budget from constant to logarithmic and linear progressively reduces loss, allowing the algorithm to approach the centralized oracle’s performance. This highlights the explicit trade-off between communication cost and learning performance.

5.6.4 The Communication-Regret Trade-Off

Our theory suggests that performing more gossip steps per round, $q(t)$, can further reduce disagreement and improve regret, at the cost of increased communication. We investigate this trade-off by running DECO-ii (KT) on a cycle graph with three different gossip schedules:

- **Constant:** $q(t) = 1$ (minimal communication).
- **Logarithmic:** $q(t) = \lceil \ln(t + 1) \rceil$.
- **Linear:** $q(t) = \lceil 0.1 \cdot t \rceil$.

Figure 5.5 shows a clear trade-off between communication and performance. While a single gossip step already performs well, increasing the number of communications over time systematically lowers the cumulative loss curves. A linear schedule, as suggested by our theory, allows the decentralized algorithm to closely track the performance of the centralized oracle.

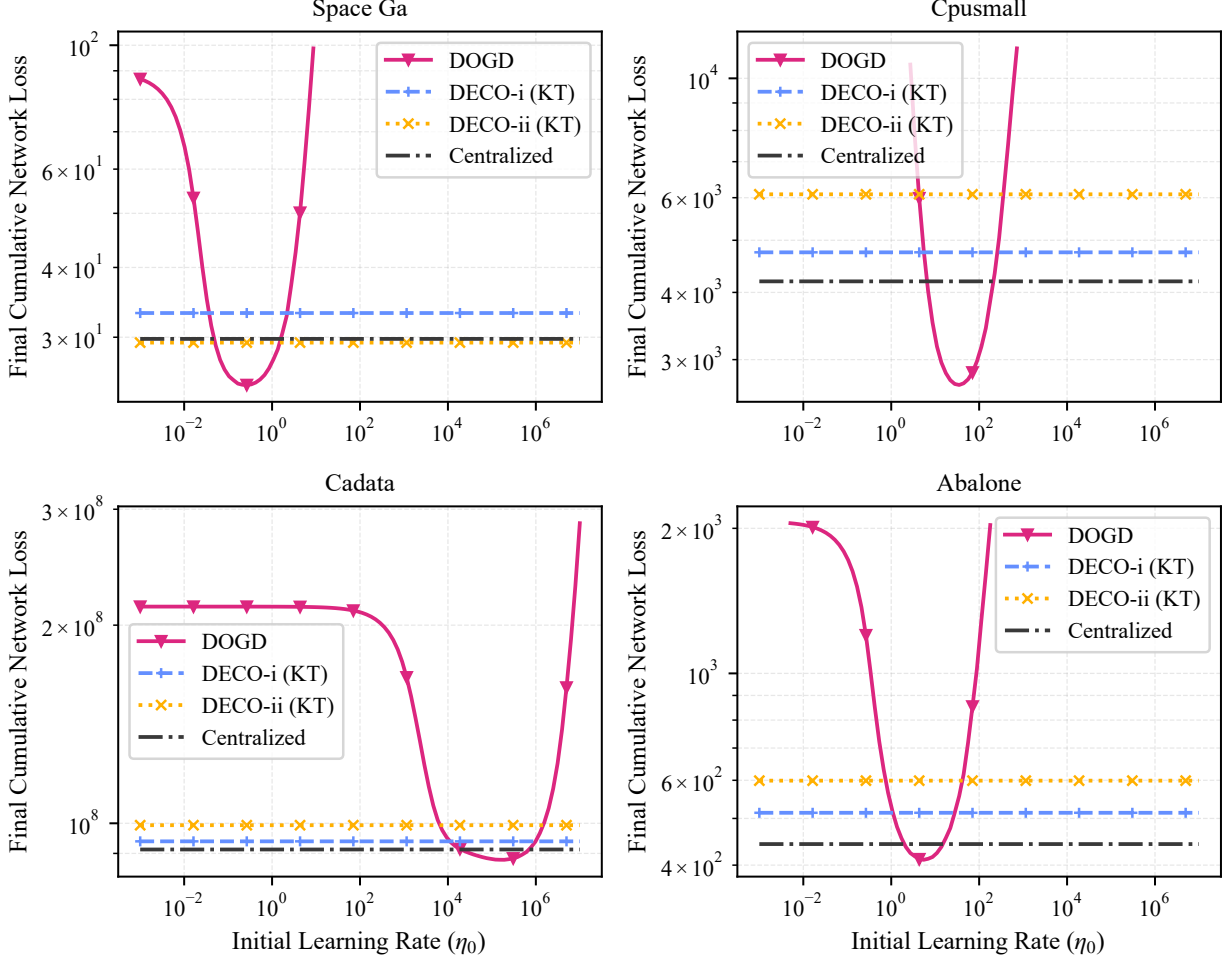


Figure 5.6: Final cumulative network loss on four real-world LIBSVM datasets. Each panel shows the sensitivity of DOGD to its initial learning rate (η_0), producing a characteristic U-shaped curve. Note that we do not plot loss values that are five times the minimum value to allow for good visualization. The horizontal lines show the performance of the parameter-free DECO variants, which achieve competitive loss without any tuning. This highlights the unreliability of DOGD when its hyperparameters are not properly set for a specific dataset.

5.6.5 Performance on Real-World Datasets

Finally, to assess robustness, we evaluate all variants of the algorithm on four real-world LIBSVM regression datasets. These experiments are run on a cycle graph with a constant gossip schedule ($q(t) = 1$). For the DOGD baseline, we perform a hyperparameter sweep over a wider range of initial learning rates, from 10^{-3} to 10^7 , to account for the diverse characteristics of these datasets.

The results in Figure 5.6 reinforce our findings. For each dataset, DOGD’s performance exhibits a sharp U-shaped dependence on its initial learning rate η_0 . Note that we do not plot loss values that are five times the minimum value to allow good visualization. The optimal value for η_0 varies significantly from one dataset to another, making it impossible to choose a single reliable setting. In contrast, the DECO variants (using the KT potential) consistently achieve low cumulative loss across all datasets without requiring any parameter selection. This shows that our framework is applicable to the diverse characteristics of real-world data.

5.7 Conclusion

This chapter introduced a family of decentralized, parameter-free online learning algorithms, termed DECO, designed to address the challenge of hyperparameter tuning in multi-agent systems. By combining the comparator-adaptive mechanism of coin-betting with information mixing via gossip protocols, our approach removes the need for setting a learning rate, a significant practical barrier for standard methods like Decentralized Online Gradient Descent (DOGD). A novel betting function formulation was also proposed to facilitate the theoretical analysis in the multi-agent context.

We provided a theoretical analysis showing that our algorithms achieve sublinear network regret. The regret bounds characterize the influence of network connectivity and communication effort through a disagreement term. Theoretically, a key open question is whether a constant number of gossip steps per round, $q(t) = \mathcal{O}(1)$, is sufficient to guarantee sublinear network regret. We conjecture that this is the case. Our theoretical findings were supported by a series of experiments on both synthetic and real-world datasets. The results demonstrated that DECO achieves performance comparable to a well-tuned DOGD baseline across various datasets and network topologies without requiring any parameter selection.

Chapter 6

Decentralized Parameter-Free Online Learning with Compressed Gossip

6.1 Introduction

Networked signal processing and learning systems increasingly make decisions from data streams instead of stored data sets. Online Convex Optimization (OCO) provides a standard adversarial model for such sequential decisions: at each round, a learner chooses a point before observing the current convex loss and performance is measured by regret with respect to the best fixed comparator in hindsight [92, 93].

Two practical issues arise in this context. First, classical OCO methods such as online gradient descent (OGD) and follow-the-regularized-leader (FTRL) require learning-rate choices that depend on the horizon, the comparator norm, or loss-scale parameters [92]. These quantities are usually unknown before the stream is observed, and a mis-specified learning rate can dominate the realized performance. Parameter-free OCO addresses this issue by replacing tuned learning rates with comparator-adaptive regret guarantees [107]. A parameter-free

guarantee is especially useful in streaming and edge deployments: the same algorithm can be run without a horizon estimate, without a comparator-scale guess, and without an expensive learning-rate sweep for every graph, device class, or data stream. A prominent mechanism is coin betting, which proves regret by lower-bounding a sequential wealth process and then converting this wealth certificate through a Fenchel conjugate [94–98]. For unbounded domains with Lipschitz losses, these methods obtain the optimal dependence on the comparator norm up to logarithmic factors [96].

Second, modern learning systems are often decentralized. Each agent observes only a local loss stream, communicates only with neighbors in a graph, and nevertheless contributes decisions that should be good for the network as a whole [100, 101, 108–111]. Communication is also lossy in many signal-processing and edge-learning deployments. The difficulty is that parameter-free learning and compressed decentralized communication interact in a nontrivial way. Coin-betting predictions are nonlinear functions of accumulated gradients, so small disagreement in the accumulated state can be amplified by the prediction map. At the same time, biased compressors create tracking error that ordinary gossip does not remove unless the communication protocol explicitly corrects it.

This chapter studies whether comparator-adaptive OCO guarantees can be retained when all communication is decentralized and compressed. We consider N agents on a connected graph. At round t , agent n predicts $x_{n,t}$, observes a local convex loss $\ell_{n,t}$, and exchanges compressed messages with its neighbors. The objective is to minimize the *network regret*, defined as follows. For a comparator u in a Hilbert space $\mathcal{H}$, the network regret after T OCO rounds is

$$R_T^{\text{net}}(u) = \sum_{t=0}^{T-1} \frac{1}{N} \sum_{n=1}^N \bar{\ell}_t(x_{n,t}) - \sum_{t=0}^{T-1} \bar{\ell}_t(u), \quad (6.1)$$

where

$$\bar{\ell}_t(x) = \frac{1}{N} \sum_{n=1}^N \ell_{n,t}(x) \quad (6.2)$$

is defined as the *network loss* at round t . The network regret evaluates all local predictions under the averaged network loss and compares them with one fixed comparator u . This is the natural collaborative benchmark for a network that learns a shared decision from distributed local observations.¹

We propose DECO-EF (DEcentralized COin-betting with Error Feedback), a decentralized parameter-free OCO algorithm with compressed gossip. Each agent maintains a clean accumulated state and a compressed tracker. Predictions are generated by a radial Hilbert-space coin-betting rule, while communication sends only compressed differences between the clean state and the tracker. The clean state is mixed through a relaxed gossip step applied to the updated trackers, so the clean network average remains the cumulative sum of the averaged negative subgradients used by the centralized comparison learner. The term parameter-free is used in the standard OCO sense: the algorithm does not tune a learning rate to the horizon or comparator norm. The graph-dependent gossip step size and the number of compressed gossip substeps remain communication parameters that depend only on the graph and compressor properties, not on the horizon, loss sequence, or comparator.

The main contributions are as follows.

- We introduce DECO-EF, a family of decentralized coin-betting algorithms in which agents communicate compressed state differences.
- We prove a network-regret decomposition that separates the centralized Hilbert-space coin-betting term from the prediction disagreement caused by decentralization and compression.

¹We take the name from [102], related work uses *collective* regret [103].

- We establish expected compressed-gossip tracking bounds for randomized mean-square compressors.
- We prove that DECO-EF with Krichevsky-Trofimov (KT) and shifted-exponential coin-betting potentials achieves expected $\tilde{O}(\max\{1, \|u\|\}\sqrt{T})$ network-regret bounds under a linear compressed-gossip schedule.
- We validate our results on both synthetic and real data, illustrating the trade-offs between communication and accuracy for DECO-EF.

To the best of our knowledge, we present the first algorithm family to achieve expected sublinear network-regret guarantees for parameter-free decentralized online learning with compressed gossip messages for the synchronous gossip setting.

The remainder of the chapter is organized as follows. Section 6.2 reviews related work. Section 6.3 defines the problem model and communication assumptions. Section 6.4 describes the high-level algorithm logic, prediction rule, and DECO-EF update. Section 6.5 states the main regret guarantees. Section 6.6 provides the proof idea for the main regret guarantees, with technical details in the appendices. Section 6.7 records implementation choices such as the potential, gossip schedule, step size, and compressor. Section 6.8 describes the empirical evaluation and Section 6.9 concludes.

6.2 Related Work

Online Convex Optimization and Parameter-Free Learning OCO formalizes the worst-case sequential prediction through regret minimization [92, 93]. Classical algorithms such as OGD, mirror descent, dual averaging, and FTRL achieve sublinear regret with appropriately tuned step sizes, but their guarantees depend on parameters such as the horizon, gradient scale, domain diameter, or comparator norm. Parameter-free online learning re-

moves this tuning by proving comparator-adaptive bounds. Coin-betting methods are a central approach: they maintain a wealth process, select predictions as bets, and obtain regret by duality from the wealth lower bound [94–98]. Our analysis uses this machinery, but the regret is for a network of learners rather than a single agent.

Decentralized Online Optimization Distributed and decentralized optimization over graphs is a long-standing topic in signal processing, control, and machine learning [101, 108, 112–114]. In online variants, agents observe time-varying losses and communicate through graph-supported averaging or dual-averaging steps, and the resulting regret bounds depend on the spectral properties of the graph and problem-specific parameters [100, 115]. Recent parameter-free distributed online-learning results address related but different models, including delayed information, activated agents, and the availability of random agents [99, 102, 103]. The present work focuses on the synchronous and fully decentralized setting in which every agent acts each round, communication is graph-local, and the benchmark is network regret.

Compressed Decentralized Online Learning Communication-efficient distributed learning uses quantization, sparsification, and other lossy compressors to reduce message sizes [3, 21, 59]. Because biased compressors such as Top- k and sign compression can accumulate error, error-feedback (EF) methods track residuals or differences rather than compressing full updates [4, 7, 8, 53, 61]. In decentralized optimization, several methods combine this difference-tracking idea with compressed proxies and gossip dynamics to obtain consensus and optimization convergence under lossy communication [31, 116]. Related online-learning work has studied collaborative regret under limited communication, including quantized online multi-kernel learning and gossiped online prediction [1, 6], as well as compressed federated or distributed optimization variants [9, 10]. The uncompressed DECO (DEcentralized COin-betting) algorithm gave parameter-free decentralized OCO with network-regret guarantees [11], while parameter-free compressed algorithms were previously obtained for the weaker joint-regret criterion [99, 103]. Other communication-constrained online-learning

Table 6.1: Positioning of representative algorithm families. Comparator-adaptive (CA) denotes explicit dependence on $\|u\|$.

Method family	Decentralized	Compressed gossip	CA	Sublinear	Network Regret	Representative guarantee
OGD/FTRL [92, 93]	×	×	×	✓		tuned network regret
Coin betting [94, 95]	×	×	✓		N/A	single-agent CA regret
Distributed CA OCO [99]	✓	✓	✓	×		joint regret
Decentralized OGD [100, 101, 113]	✓	×	×	✓		tuned network regret
EF-style compression [61, 116]	✓	✓	×		N/A	optimization convergence
DECO [11]	✓	×	✓	✓		CA network regret
DECO-EF	✓	✓	✓	✓		expected CA network regret

methods do not provide comparator-adaptive network regret [117, 118]. DECO-EF combines these directions by importing the difference-tracked compressed gossip into adversarial decentralized OCO, preserving the comparator-adaptive network regret while allowing compressed messages.

Table 6.1 summarizes the related work. The rows differ in the feedback model, synchrony, and objective. Its purpose is to isolate the three properties combined here: decentralized communication, compressed gossip, and comparator-adaptive sublinear network regret.

The network-regret notion in Equation (6.1) is a global-performance criterion. It differs from per-agent regret, which certifies each node only against its own observed loss stream, and from consensus error, which does not by itself measure predictive performance. As noted in [102], it is also different from the average of per-agent regrets, which is a weaker criterion that can be sublinear even if the network regret is not. The uncompressed DECO algorithm [11] is recovered from DECO-EF by taking the identity compressor and the exact-message communication.

6.3 Model and Setup

This section provides the mathematical model, network objective, communication graph, and compressor assumptions.

6.3.1 Problem Model

There are N agents that learn from local data streams while communicating over a fixed graph. Rounds are indexed by $t = 0, \dots, T-1$, and all horizon-dependent statements below are for positive integer horizons T . At each round, agent n chooses $x_{n,t} \in \mathcal{H}$, observes a convex 1-Lipschitz local loss $\ell_{n,t} : \mathcal{H} \rightarrow \mathbb{R}$, and receives a subgradient $g_{n,t} \in \partial \ell_{n,t}(x_{n,t})$. We aim to bound the network regret defined in Equation (6.1). This is an evaluation criterion for collaborative prediction: agent n observes and differentiates its own local loss $\ell_{n,t}$, but the regret score evaluates every local prediction under the averaged network loss $\bar{\ell}_t$.

6.3.2 Communication Graph and Compression

Agents communicate on a fixed undirected communication graph $\mathcal{G} = ([N], E)$, which we assume to be connected. A valid gossip matrix for $\mathcal{G}$ is a nonnegative, symmetric, doubly stochastic matrix W whose off-diagonal support is exactly the edge set:

$$W_{ij} > 0 \text{ if } \{i, j\} \in E, \quad W_{ij} = 0 \text{ if } i \neq j \text{ and } \{i, j\} \notin E.$$

We also require positive self-weights $W_{ii} > 0$. For $Z \in \mathcal{H}^N$, we define the product network norm

$$\|Z\|_{2,\mathcal{H}} = \left(\sum_{n=1}^N \|Z_n\|^2 \right)^{1/2}, \tag{6.3}$$

and we define

$$(WZ)_n = \sum_{j=1}^N W_{nj} Z_j.$$

For such a gossip matrix, the usual finite-dimensional spectral-gap theorem [105] gives a zero-mean contraction factor satisfying $0 \leq \rho < 1$. Equivalently, the spectral gap $\mu = 1 - \rho$

satisfies $0 < \mu \leq 1$, with $\rho = 0$ for exact averaging. The Hilbert lift of W is nonexpansive in this product ℓ_2 norm and contracts the zero-mean subspace:

$$\begin{aligned} \|WZ\|_{2,\mathcal{H}} &\leq \|Z\|_{2,\mathcal{H}} \quad \text{for all } Z \in \mathcal{H}^N, \\ \|WZ\|_{2,\mathcal{H}} &\leq \rho \|Z\|_{2,\mathcal{H}} \quad \text{if } \sum_{n=1}^N Z_n = 0. \end{aligned} \tag{6.4}$$

Messages sent in our graph are compressed.

DEFINITION 6.1 (Compressor). *A randomized compressor on $\mathcal{H}$ consists of a sample space Ω , a map $\mathcal{C} : \Omega \times \mathcal{H} \rightarrow \mathcal{H}$, and a parameter $0 \leq \alpha^{\text{stoch}} < 1$ such that for every input $v \in \mathcal{H}$,*

$$\mathbf{E}_{\omega} \|\mathcal{C}(\omega, v) - v\|^2 \leq \alpha^{\text{stoch}} \|v\|^2. \tag{6.5}$$

Fix a compressor in the sense of Definition 6.1. Table 6.2 lists representative mean-square residual coefficients α^{stoch} for compressors that take inputs in $v \in \mathbb{R}^d$. For the low-rank approximation entry, v is reshaped to an $a \times b$ matrix with $d = ab$, $p = \min\{a, b\}$, and the compressor keeps the best rank- r approximation with $r < p$. For b -bit uniform scalar quantization, one bit stores the sign and the remaining $b - 1$ bits store the magnitude: after normalizing by $\|v\|_{\infty}$, each coordinate magnitude is rounded to one of 2^{b-1} the equally spaced values in $[0, 1]$. For top- k and random- k sparsification, k coordinates are kept and the rest are zeroed out. Top- k keeps the k largest-magnitude coordinates, and the random- k chooses k coordinates uniformly at random. For the scaled sign compressor, the output is $\frac{\|v\|_1}{d}$ times the sign vector of v .

Table 6.2: Representative mean-square compressor coefficients.

Compressor	α^{stoch}
Identity [11]	0
Top- k [3, 59]	$1 - k/d$
Scaled sign [4]	$1 - 1/d$
Low-rank, rank $r < p$ [81]	$1 - r/p$
b -bit quant. [3]	$\frac{d}{4(2^{b-1}-1)^2}$
Random- k , unscaled [59]	$1 - k/d$

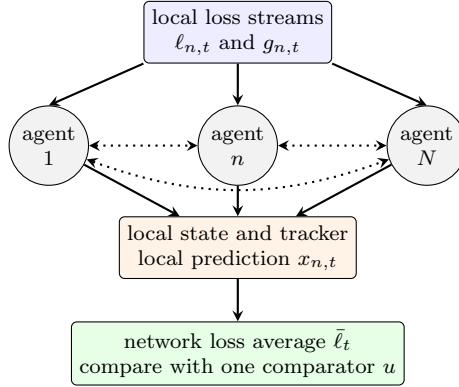


Figure 6.1: System model for DECO-EF. Agents make local predictions, observe local convex losses, and exchange compressed state differences over the communication graph. The analysis evaluates the averaged network loss against a single comparator u .

6.4 Algorithm

This section describes the algorithmic idea and the concrete DECO-EF update analyzed in this chapter.

6.4.1 High-Level Idea

DECO-EF combines parameter-free coin-betting prediction with compressed error-feedback gossip. The full one-round update appears in Algorithm 6, and its high-level system model is sketched in Figure 6.1. At the beginning of a round, each agent predicts from its local accumulated state using the coin-betting map defined in Section 6.4.2. After observing a local subgradient, the agent updates a clean state and then performs $q(t)$ compressed gossip substeps. Each communication substep sends the compressed difference between the clean

state and a tracker, rather than compressing the clean state itself. This design preserves the clean network average while the tracker absorbs compression error through repeated difference updates. The *communication step size* γ is a communication parameter that should be chosen according to the graph and compressor, but it does not depend on the loss sequence or comparator. A larger γ gives faster consensus but can amplify compression error, while a smaller γ reduces compression error but slows consensus. The gossip steps reduce clean-state disagreement, while the compressed difference updates keep the trackers close enough to the clean states for the Lyapunov contraction used in the proof.

The remaining subsections define the prediction map and state update.

6.4.2 Coin-Betting Potentials and Predictions

The algorithm is parameter-free in the usual OCO sense: it does not choose a learning rate as a function of T , or $\|u\|$. It still has a coin-betting endowment $\varepsilon > 0$, which fixes the initial wealth scale and communication parameters determined by the graph and compressor. The prediction rule converts accumulated signed gradients into decisions through a coin-betting potential [11, 94, 107].

DEFINITION 6.2 (Coin-Betting Potential). *Let $\varepsilon > 0$. A family of functions F_t , indexed by integers $t \geq 0$, with domains $F_t : (-a_t, a_t) \rightarrow \mathbb{R}$ is a coin-betting potential if:*

- (a) $F_0(0) = \varepsilon$ and $a_t > t$.
- (b) Each F_t is even, positive in $(-a_t, a_t)$, and $\log F_t$ is convex in $(-a_t, a_t)$.
- (c) For every $t \geq 1$, $|x| \leq t - 1$, and $|c| \leq 1$,

$$(1 + c\beta_t(x)) F_{t-1}(x) \geq F_t(x + c), \tag{6.6}$$

where

$$\beta_t(x) = \frac{F_t(x+1) - F_t(x-1)}{F_t(x+1) + F_t(x-1)}. \quad (6.7)$$

It is excellent if, additionally, for every t and every $0 < R < a_t$, the square-root lift

$$z \mapsto F_t(\sqrt{z}) \quad (6.8)$$

is convex on $[0, R^2]$. This extra condition is needed to lift scalar coin betting to a Hilbert space.

REMARK 6.3. In [94], the excellence condition required F_t to be twice-differentiable and $xF_t''(x) \geq F_t'(x)$. When potentials are twice-differentiable, both conditions are equivalent. However, we argue that the square-root formulation is better since it only requires convexity of (6.8), not differentiability.

The coin-betting function [11] converts wealth into a one-dimensional wager through

$$h_t(x) = \beta_t(x)F_{t-1}(x). \quad (6.9)$$

For a Hilbert-space state $G \in \mathcal{H}$, we use the radial lift of this function: the prediction points in the direction of G and has magnitude $h_t(\|G\|)$,

$$\text{Bet}_t(G) = \begin{cases} 0, & G = 0, \\ \frac{h_t(\|G\|)}{\|G\|}G, & G \neq 0. \end{cases} \quad (6.10)$$

The potential formulas are only used on their reachable domain. To keep the prediction input in that domain, the algorithm clips the norm of the state at radius t while preserving

its direction:

$$\text{clip}_t(G) = \begin{cases} G, & \|G\| \leq t, \\ \frac{t}{\|G\|}G, & \|G\| > t. \end{cases} \quad (6.11)$$

Thus $\|\text{clip}_t(G)\| \leq t$ and clip_t is the metric projection onto the closed ball of radius t . In the one-dimensional case, norm clipping is exactly the usual scalar clipping map

$$\text{clip}_t(z) = \min\{t, \max\{-t, z\}\}. \quad (6.12)$$

Both maps are 1-Lipschitz. This clipping affects only the prediction query $\text{Bet}_{t+1}(\text{clip}_t(G_{n,t}))$, the clean state used for gradient accumulation and compressed gossip remains the unclipped $G_{n,t}$. To give explicit examples of potentials, we introduce the two that we use throughout this chapter:

$$F_t^{\text{KT}}(x) = \varepsilon \frac{2^t \Gamma\left(\frac{t+1}{2} + \frac{x}{2}\right) \Gamma\left(\frac{t+1}{2} - \frac{x}{2}\right)}{\pi t!}, \quad (6.13)$$

$$F_t^{\text{SE}}(x) = \frac{\varepsilon}{\sqrt{t+1}} \exp\left(\frac{x^2}{2(t+1)}\right). \quad (6.14)$$

Here KT abbreviates Krichevsky-Trofimov and SE abbreviates shifted-exponential. The verification that these choices are excellent coin-betting potentials follows the standard KT and exponential-potential calculations, see [11, 94]. The KT betting fraction is $\beta_t^{\text{KT}}(x) = x/t$ for $t \geq 1$, and the SE betting fraction is $\beta_t^{\text{SE}}(x) = \tanh(x/(t+1))$.²

²The SE potential in Equation (6.14) is a shifted version of the original exponential potential [94], which was defined as $F_t(x) = \varepsilon \exp(x^2/(2t))$ for $t \geq 1$ and $F_0(0) = \varepsilon$. However, the reader might notice that the unshifted exponential potential is not a coin-betting potential according to Definition 6.2, since condition (c) fails at $t = 1$, $x = 0$, and any $c \neq 0$. Indeed, $\beta_1(0) = 0$, so the required inequality would imply $\varepsilon \geq \varepsilon \exp(c^2/2)$, which is false for $c \neq 0$.

Algorithm 6 DECO-EF: One Round at Agent n

Input: State $(G_{n,t}, \hat{G}_{n,t})$, potential F , gossip matrix W , step size γ , gossip count $q(t)$.

- 1: Predict $x_{n,t} = \text{Bet}_{t+1}(\text{clip}_t(G_{n,t}))$.
 - 2: Observe $g_{n,t} \in \partial \ell_{n,t}(x_{n,t})$ and set $\delta_{n,t} = -g_{n,t}$.
 - 3: Local movement: $Z_{n,t}^0 = G_{n,t} + \delta_{n,t}$ and $\hat{Z}_{n,t}^0 = \hat{G}_{n,t}$.
 - 4: **for** $k = 0, \dots, q(t) - 1$ **do**
 - 5: Difference payload: $v_{n,t}^k = Z_{n,t}^k - \hat{Z}_{n,t}^k$.
 - 6: Compress and broadcast to neighbors:
$$m_{n,t}^k = \mathcal{C}(\omega_{n,t}^k, v_{n,t}^k).$$
 - 7: Tracker update: $\hat{Z}_{n,t}^{k+1} = \hat{Z}_{n,t}^k + m_{n,t}^k$.
 - 8: Relaxed gossip update:
$$Z_{n,t}^{k+1} = Z_{n,t}^k + \gamma \sum_{j=1}^N W_{nj} \left(\hat{Z}_{j,t}^{k+1} - \hat{Z}_{n,t}^{k+1} \right).$$
 - 9: **end for**
 - 10: State update: $G_{n,t+1} = Z_{n,t}^{q(t)}$ and $\hat{G}_{n,t+1} = \hat{Z}_{n,t}^{q(t)}$.
-

6.4.3 DECO-EF State and Update

The DECO-EF update gossips compressed difference messages inside every communication substep. Each agent stores two vectors:

- $G_{n,t}$: the true local accumulated state after the consensus correction;
- $\hat{G}_{n,t}$: the compressed tracker shared through gossip.

Both are initialized to zero.

At round t , agent n applies the following update with $\delta_{n,t} = -g_{n,t}$. The communication step size $0 < \gamma \leq 1$ controls how aggressively agents mix compressed tracker information, and the gossip schedule $q : \mathbb{N} \rightarrow \mathbb{N}$ specifies how many compressed communication substeps are performed after the local gradient observation in round t .

The communication model is synchronous and reliable. At substep k , node n computes one compressed payload $m_{n,t}^k$ and broadcasts the same payload to all neighbors with $W_{jn} > 0$.

Receivers update their local copy of node n 's tracker by the same increment. Equivalently, the matrix update is the shared-tracker view after all messages in that substep have arrived.

The Laplacian form of the last line is important: since W is doubly stochastic,

$$\sum_{n=1}^N \sum_{j=1}^N W_{nj} \left(\widehat{Z}_{j,t}^{k+1} - \widehat{Z}_{n,t}^{k+1} \right) = 0,$$

so every compressed gossip substep preserves the average of the clean state Z^k .

6.4.4 Notation Used in the Analysis

Individual vectors in the decision space use the Hilbert norm. For $Z \in \mathcal{H}^N$, we define the product network norm as in Equation (6.3). We write JZ for the consensus vector whose coordinates are all $N^{-1} \sum_m Z_m$, and $P_{\perp}Z = Z - JZ$ for centered disagreement. The compression parameter α^{stoch} is the mean-square residual coefficient from Definition 6.1. We also use the spectral gap $\mu = 1 - \rho$, the Lyapunov weight $\chi = \mu/4$, and the expected compressed-gossip contraction rate $\bar{\eta} = 1 - \gamma\mu/2$. A longer reference table appears in Table C.1.

6.5 Main Results

The analysis uses two reusable statements. First, the network argument only needs a Hilbert-space coin-betting regret bound for the radial prediction map. For a terminal potential, namely the potential function at time T , we write its restricted Fenchel conjugate on the reachable interval as

$$F_T^*(\theta) = \sup_{0 \leq s \leq T} \{\theta s - F_T(s)\}. \quad (6.15)$$

Square-root convex excellence (see Definition 6.2) turns the scalar wealth telescope into a Hilbert-space reward telescope, and the final comparator conversion is exactly the conjugate in (6.15). The main theorems are stated in terms of this exact conjugate. The elementary envelope, used only to turn the KT and shifted-exponential conjugates into closed-form corollaries, is discussed in Lemma C.6.

THEOREM 6.4 (General Network Regret Template). *Let $\mathcal{H}$ be a real Hilbert space. Suppose a DECO-EF run uses a coin-betting potential whose radial Hilbert-space regret is controlled by its restricted terminal conjugate. Let $\bar{G}_t = N^{-1} \sum_n G_{n,t}$ and $y_t = \text{Bet}_{t+1}(\bar{G}_t)$. If its prediction disagreement satisfies*

$$\sum_{t=0}^{T-1} \frac{1}{N} \sum_{n=1}^N \|x_{n,t} - y_t\| \leq C_{\text{net}} \sqrt{T},$$

then for all $u \in \mathcal{H}$ and all $T > 0$,

$$R_T^{\text{net}}(u) \leq F_T^*(\|u\|) + \varepsilon + 3C_{\text{net}} \sqrt{T}. \quad (6.16)$$

Second, for both concrete potentials with a linear gossip schedule, Theorem 6.5 uses the effective tracker rate

$$\bar{\eta} = 1 - \frac{\gamma\mu}{2} \quad (6.17)$$

rather than the clean relaxed-gossip rate $1 - \gamma\mu$, which is the ideal zero-mean contraction rate without compression.

THEOREM 6.5 (Expected Network Regret). *Let DECO-EF run over a real Hilbert space $\mathcal{H}$ with either the KT potential (6.13) or the shifted-exponential potential (6.14). Assume W is a valid gossip matrix for a connected communication graph, and let ρ be the graph-derived product- ℓ_2 zero-mean contraction factor satisfying $0 \leq \rho < 1$. Let the compressor, as*

in Definition 6.1, have mean-square residual parameter α^{stoch} . Put

$$\mu = 1 - \rho, \quad \bar{\sigma} = \sqrt{\alpha^{\text{stoch}}}.$$

Here, μ is the spectral gap. Let $P \in \{\text{KT}, \text{SE}\}$ denote the chosen potential, and let F_T^* be its restricted terminal conjugate. Choose

$$0 < \gamma \leq \min \left\{ 1, \frac{1 - \bar{\sigma}}{2\bar{\sigma} \left(1 + \frac{4}{\mu} \right) + \frac{\mu}{2}} \right\},$$

define $\bar{\eta} = 1 - \gamma\mu/2$, and use a linear gossip schedule $q(t) = \lceil ct \rceil$, where $c \geq -\frac{2\log 2}{\log \bar{\eta}}$ for KT and $c \geq -\frac{3}{2\log \bar{\eta}}$ for the shifted-exponential potential. Put $\chi = \mu/4$ and

$$C_{\text{sched}}^{\text{KT}} = 48 \left((\varepsilon + 1) \left(1 + \frac{2}{\log 2} \right) + 1 \right),$$

$$C_{\text{sched}}^{\text{SE}} = 4e^{3/2}(2\varepsilon + 1).$$

With C_{sched}^P selecting the matching constant above, set

$$C_{\text{net}}^P = (2 + \chi) \frac{C_{\text{sched}}^P}{2} \sqrt{N}, \quad C_{\text{reg}}^{P, \text{stoch}} = 3C_{\text{net}}^P.$$

Then, for every comparator $u \in \mathcal{H}$ and every $T > 0$,

$$\mathbf{E} R_T^{\text{net}}(u) \leq F_T^*(\|u\|) + \varepsilon + C_{\text{reg}}^{P, \text{stoch}} \sqrt{T}, \quad (6.18)$$

where the expectation is over the compressor randomness. The constants are independent of u and T .

The proof uses the conditional compressed-tracker contraction in Theorem C.13, the expected state-disagreement unrolling in Lemma C.15, and the potential-specific Lipschitz and

schedule-tail estimates in Appendices C.5 and C.6.

COROLLARY 6.6 (Closed-Form KT and Shifted-Exponential Bounds). *In the setting of Theorem 6.5, the KT potential satisfies*

$$\begin{aligned} \mathbf{E} R_T^{\text{net}}(u) &\leq \|u\| \sqrt{2(T+1)} \\ &\quad \times \sqrt{\log \left(1 + \frac{2e^2\pi T(T+1) \|u\|^2}{\varepsilon^2} \right)} \\ &\quad + \varepsilon \left(1 - \frac{1}{e\sqrt{\pi T}} \right) + C_{\text{reg}}^{\text{KT, stoch}} \sqrt{T}, \end{aligned} \tag{6.19}$$

and the shifted-exponential potential satisfies

$$\begin{aligned} \mathbf{E} R_T^{\text{net}}(u) &\leq \|u\| \sqrt{(T+1) \log \left(1 + \frac{(T+1)^2 \|u\|^2}{\varepsilon^2} \right)} \\ &\quad + \varepsilon \left(1 - \frac{1}{\sqrt{T+1}} \right) + C_{\text{reg}}^{\text{SE, stoch}} \sqrt{T}. \end{aligned} \tag{6.20}$$

COROLLARY 6.7 (Polylogarithmic Comparator Scale). *The two expected regret bounds in Equations (6.19) and (6.20) are $\tilde{O}(\max\{1, \|u\|\}\sqrt{T})$, where $\tilde{O}$ suppresses polylogarithmic factors in T and $\|u\|$.*

COROLLARY 6.8 (Communication Cost of the Proved Schedule). *For the linear schedule $q(t) = \lceil ct \rceil$, the total number of compressed gossip substeps through horizon T is*

$$M_T = \sum_{t=0}^{T-1} q(t), \quad \frac{cT(T-1)}{2} \leq M_T \leq \frac{cT(T-1)}{2} + T.$$

Thus, the linear schedule in Theorem 6.5 uses $\Theta(T^2)$ compressed gossip substeps.

6.6 Analysis

This section provides the proof idea for the main regret guarantees. The auxiliary inequalities, Lyapunov algebra, and potential-specific estimates are proved in the appendices. The main argument is as follows: reduce network regret to a centralized coin-betting term plus a prediction-disagreement term, bound the centralized term by Hilbert-space coin betting, and then use compressed-gossip tracking to show that the disagreement term is only $O(\sqrt{T})$ under the stated schedule.

6.6.1 Regret Decomposition

Let

$$\bar{g}_t = \frac{1}{N} \sum_{n=1}^N g_{n,t}, \quad \bar{G}_t = \frac{1}{N} \sum_{n=1}^N G_{n,t},$$

and define the centralized prediction

$$y_t = \text{Bet}_{t+1}(\bar{G}_t).$$

Since $\bar{G}_t$ is the sum of t averaged negative subgradients of norm at most one, $\|\bar{G}_t\| \leq t$. Hence, the centralized prediction lies in the reachable domain and no clipping is needed. The average clean state is exactly the centralized cumulative negative gradient,

$$\bar{G}_t = \sum_{s=0}^{t-1} (-\bar{g}_s),$$

because every gossip correction has zero network average. Thus, the network can be compared with a fictitious centralized coin-betting learner that sees the averaged gradients. This centralized learner is the bridge between the parameter-free OCO analysis and the decentralized protocol: all remaining network effects enter only through the distance from $x_{n,t}$ to

y_t .

Convexity and Lipschitzness give, round by round,

$$\frac{1}{N} \sum_{n=1}^N \bar{\ell}_t(x_{n,t}) - \bar{\ell}_t(u) \leq \langle \bar{g}_t, y_t - u \rangle + \frac{3}{N} \sum_{n=1}^N \|x_{n,t} - y_t\|. \quad (6.21)$$

The constant 3 comes from comparing the global loss at $x_{n,t}$ to the centralized prediction y_t , replacing local subgradients by their network average, and using the 1-Lipschitz assumption. The first term is the regret of the centralized radial coin-betting learner. The second term is the price of decentralization: it vanishes if all agents predict exactly at the average state. Summing (6.21) results in

$$R_T^{\text{net}}(u) \leq \sum_{t=0}^{T-1} \langle \bar{g}_t, y_t - u \rangle + 3 \sum_{t=0}^{T-1} \frac{1}{N} \sum_{n=1}^N \|x_{n,t} - y_t\|. \quad (6.22)$$

This is the central proof split used throughout this chapter. The detailed one-round comparison is given in Lemma C.27. We analyze these two terms in what follows.

6.6.2 Centralized Coin-Betting

We first handle the term that would remain even if every agent predicted at the network average. Set $c_t = -\bar{g}_t$. Since each local loss is 1-Lipschitz, $\|c_t\| \leq 1$, and the centralized state above is $\bar{G}_t = \sum_{s < t} c_s$. Square-root convex excellence lets the scalar coin-betting potential lift radially to a Hilbert space: the one-dimensional wealth telescoping argument applies to $\|\bar{G}_t\|$, while Jensen's inequality controls the change in squared norm after adding the vector c_t . The terminal Fenchel conjugate (6.15) then converts the wealth lower bound into

$$\sum_{t=0}^{T-1} \langle \bar{g}_t, y_t - u \rangle \leq F_T^*(\|u\|) + \varepsilon. \quad (6.23)$$

The scalar telescoping, Fenchel conversion, and Hilbert-space lift are proved in Appendix C.2. It remains to show that the actual local predictions stay close enough to this centralized trajectory.

6.6.3 Compressed Tracking

The disagreement term in (6.22) is controlled first at the state level. At a fixed outer round, suppress the round index and write Z^k and $\widehat{Z}^k$ for the inner-loop clean state and tracker. Formally, let us define the *inner-loop* index k to run from 0 to $q(t) - 1$ for the $q(t)$ gossip substeps at outer-loop time t . The inner gossip loop tracks two quantities: the clean-state disagreement D^k and the compression residual E^k , defined by

$$D^k = \|P_{\perp} Z^k\|_{2,\mathcal{H}}, \quad E^k = \|Z^k - \widehat{Z}^k\|_{2,\mathcal{H}}. \quad (6.24)$$

With $\mu = 1 - \rho$, $\bar{\sigma} = \sqrt{\alpha^{\text{stoch}}}$, and $\chi = \mu/4$, the Lyapunov quantity

$$V^k = D^k + \chi E^k$$

contracts in conditional expectation by the effective rate

$$\bar{\eta} = 1 - \frac{\gamma\mu}{2}$$

whenever γ satisfies the step-size condition in Theorem 6.5. The important point is that the rate is the compressed-tracker rate, not the cleaner relaxed-gossip rate one would obtain without residuals. More precisely, conditionally on the history $\mathcal{F}_{t,k}$ before gossip substep k ,

$$\mathbf{E}[V^{k+1} \mid \mathcal{F}_{t,k}] \leq \bar{\eta} V^k.$$

Each outer round injects a bounded gradient perturbation, and then $q(t)$ gossip substeps

contract it. Unrolling these injections in expectation gives

$$\begin{aligned} \mathbf{E} \frac{1}{N} \sum_{n=1}^N \|G_{n,t} - \bar{G}_t\| &\leq (2 + \chi) \sqrt{N} \sum_{s < t} \bar{\eta}^{Q(s,t)}, \\ Q(s, t) &= \sum_{r=s}^{t-1} q(r). \end{aligned} \tag{6.25}$$

The Lyapunov contraction and unrolling details are collected in Appendix C.3. The final step is to pass this state estimate through the nonlinear coin-betting prediction map.

6.6.4 From State Disagreement to Regret

The prediction map is Lipschitz on the clipped ball of radius t . If $L(t)$ denotes a valid envelope for the potential in use, then (6.25) implies

$$\mathbf{E} \frac{1}{N} \sum_{n=1}^N \|x_{n,t} - y_t\| \leq (2 + \chi) \sqrt{N} L(t) \sum_{s < t} \bar{\eta}^{Q(s,t)}.$$

For the KT potential, the envelope grows like $2^t \log(t+2)/\sqrt{t+1}$. For the shifted-exponential potential, it grows like $\exp(t/2)/\sqrt{t+1}$. For $P \in \{\text{KT}, \text{SE}\}$ denoting the chosen potential, the stated linear schedules make the gossip tail small enough that

$$\mathbf{E} \sum_{t=0}^{T-1} \frac{1}{N} \sum_{n=1}^N \|x_{n,t} - y_t\| \leq C_{\text{net}}^P \sqrt{T}. \tag{6.26}$$

Taking expectations in (6.22) and substituting (6.23) and (6.26) proves Theorem 6.5. The centralized coin-betting term pays the comparator-adaptive part, while compressed tracking contributes only the stated network constant times $\sqrt{T}$.

The KT and shifted-exponential analytic checks, including square-root convexity and the schedule-tail estimates, appear in Appendices C.5 and C.6.

6.7 Operational Details

To help practitioners act on the theoretical results, this section guides the operational choices required by DECO-EF.

6.7.1 Choosing the Potential

The above theory applies to both potentials in Equations (6.13) and (6.14). The shifted-exponential option is attractive computationally because it avoids Gamma-function evaluations and has $\beta_t^{\text{SE}}(x) = \tanh(x/(t+1))$.

6.7.2 Choosing the Gossip Schedule

Theorem 6.5 uses a linear schedule, $q(t) = \lceil ct \rceil$, with c chosen as a function of the compressed effective rate from Equation (6.17). The KT proof requires $c \geq -\frac{2\log 2}{\log \bar{\eta}}$. The shifted-exponential proof uses $c \geq -\frac{3}{2\log \bar{\eta}}$. Operationally, larger c increases communication but reduces disagreement faster. By Corollary 6.8, the proved schedule costs $\Theta(T^2)$ compressed gossip substeps through horizon T , so the guarantee should be read as a high-communication theorem rather than a constant communication theorem. Constant gossip schedules are empirically appealing, but Theorem 6.5 certifies the linear schedule.

6.7.3 Communication/Contraction Trade-Off

The step size γ controls the compressed-tracker contraction from Equation (6.17). Smaller γ makes each compressed gossip step smoother, but it also makes $\bar{\eta}$ closer to one, so the schedule must use more gossip rounds. Write $\mu = 1 - \rho$ for the spectral gap. The expected stability proof uses the sufficient condition

$$0 < \gamma \leq \min \left\{ 1, \frac{1 - \sqrt{\alpha^{\text{stoch}}}}{2\sqrt{\alpha^{\text{stoch}}} \left(1 + \frac{4}{\mu} \right) + \frac{\mu}{2}} \right\}.$$

This is the product- ℓ_2 sufficient condition used in the contraction argument, provided the residual vector bound is available in expectation for the agent-wise randomness. When $\gamma\mu$ is small,

$$\begin{aligned} -\log \bar{\eta} &\approx \frac{\gamma\mu}{2}, \\ c^{\text{KT}} &\approx \frac{4\log 2}{\gamma\mu}, \quad c^{\text{SE}} \approx \frac{3}{\gamma\mu}. \end{aligned}$$

Thus choosing γ is primarily a communication/contraction tradeoff. For parameter choices certified by Theorem 6.5, one should take γ as large as the graph/compressor bound permits, since this makes $\bar{\eta} = 1 - \gamma\mu/2$ smaller and therefore lowers the required linear-schedule constant c . If γ is chosen smaller, the compressed tracker changes more cautiously and the proof still applies, but the schedule must compensate with more gossip substeps. Conversely, choosing γ above the sufficient bound may work empirically but is outside the stated Lyapunov contraction argument. The constant- $q(t)$ experiments in Section 6.8 therefore use a fixed relaxation as an empirical low-communication regime, not as the schedule certified by Theorem 6.5.

6.7.4 Compression and Communication

The compressor interface is exactly Definition 6.1: a mean-square residual bound with $\alpha^{\text{stoch}} < 1$. Theorem 6.5 uses α^{stoch} after taking expectation over the compressed messages. Each outer round first moves the clean state by the new local gradient. Then, each of the $q(t)$ communication substeps sends only the compressed difference $\mathcal{C}(\omega_n^k, Z_n^k - \hat{Z}_n^k)$, updates the tracker $\hat{Z}$, and applies the Laplacian correction to Z using the updated trackers. The analysis above shows that, under the stated small- γ condition, these compressed-message substeps imply the prediction-disagreement estimate needed for the KT and shifted-exponential regret bounds.

6.8 Experiments

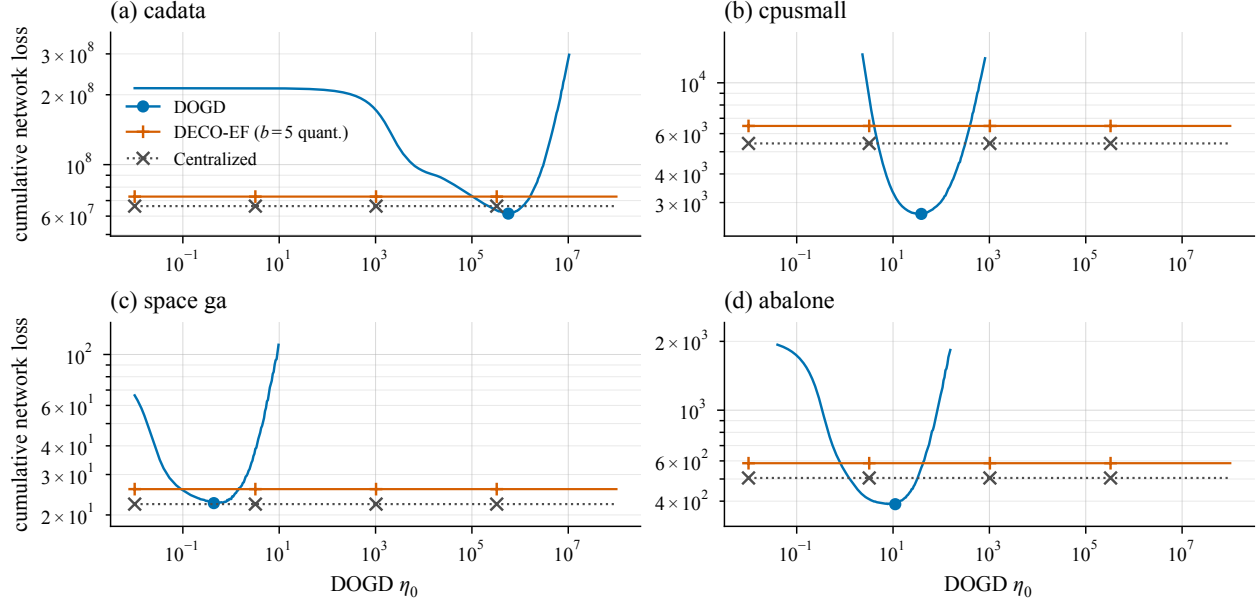


Figure 6.2: LIBSVM online-regression experiments, showing decentralized online gradient descent (DOGD) tuning sensitivity on the ring graph against parameter-free methods.

We evaluate two complementary effects. First, real LIBSVM [5] streams test the tuning sensitivity of DOGD against the parameter-free coin-betting methods. Second, a controlled heterogeneous synthetic stream isolates the effect of quantized communication on loss, bit budget, and consensus. Following the previous DECO evaluation protocol, we use the absolute linear prediction loss

$$\ell_{n,t}(x) = |\langle x, z_{n,t} \rangle - y_{n,t}|, \quad \|z_{n,t}\| \leq 1,$$

where the norm of every subgradient is at most one. The real-data tasks use the LIBSVM regression datasets `cadata`, `cpusmall`, `space_ga`, and `abalone`. Each dataset is row-scaled so that $\|z_{n,t}\| \leq 1$. The ordered stream is not reshuffled, samples are assigned to $N = 20$ agents in round-robin order, and each run uses the maximum number of complete synchronous rounds available without reusing samples. The communication graph in the LIBSVM panels is the ring, with a single communication round $q(t) = 1$. DOGD is swept over 500 logarithmically spaced initial learning rates from 10^{-2} to 10^8 . The DECO-EF curves use the

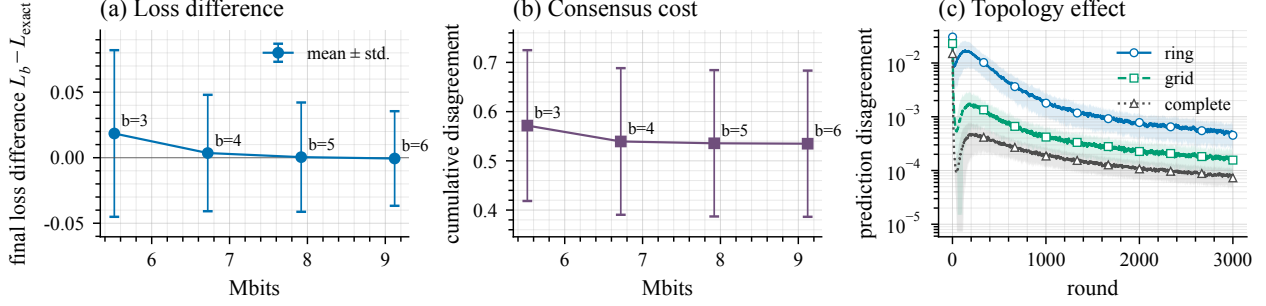


Figure 6.3: Synthetic quantized-communication diagnostics for DECO-EF. All decentralized runs use $\gamma = 1/2$, and the compressed runs use b -bit quantization. Each plotted point or curve is the average over one hundred independent synthetic streams, with error bars or shaded bands showing one standard deviation. Panels (a) and (b) use the complete graph with $q(t) = 1$, the topology that most clearly separates quantization resolution from slow graph mixing. Panel (a) reports the paired final cumulative network loss difference $L_b - L_{\text{exact}}$ against the no-quantization reference run with the same stream seed and relaxation. Panel (b) reports cumulative prediction disagreement against transmitted Mbits. Panel (c) fixes $b = 3$ and $q(t) = 1$, and compares the disagreement on the ring, 4×5 toroidal grid, and complete graph.

shifted-exponential potential with $b = 5$ -bit uniform scalar quantization. The centralized coin-betting reference uses the KT potential. For this centralized baseline, all agents use the common centralized prediction. The baseline averages the $N = 20$ local subgradients evaluated at that common prediction, runs a single KT coin-betting learner, and evaluates the resulting common prediction under the same network loss. It removes graph mixing and disagreement effects, so it should be read as a non-decentralized parameter-free reference rather than a communication-limited method. All decentralized coin-betting runs in the plots use the same $\gamma = 1/2$.

Since our topologies are regular graphs, we use the uniform gossip matrix: $W_{ij} = 1/\text{deg}$ when j is in the neighborhood of i , and $W_{ij} = 0$ otherwise, where deg includes the self-loop. This results in a valid gossip matrix. For the synthetic compression diagnostics, all runs use shifted-exponential DECO-EF, and the compressed runs use b -bit uniform scalar quantization with $b \in \{3, 4, 5, 6\}$. Communication costs count raw dense uniform-scalar-quantization codes, i.e., one 32-bit scale plus b bits per coordinate. They are reported as the total payload emitted by all agents over all gossip substeps, not multiplied by the number

of graph edges. Thus, for a fixed dataset, method, compressor, and $q(t)$, the Mbits value is the same across ring, grid, and complete graphs. Topology changes the mixing operator and disagreement, not this payload-count convention.

The synthetic stream has dimension 20 and $T = 3000$ rounds. For each seed $r \in \{0, \dots, 99\}$, a pseudorandom generator with seed r draws $u^* \sim \mathcal{N}(0, I_{20})$ and normalizes it. We then draw shared base vectors $b_t \sim \mathcal{N}(0, I_{20})$ and agent centers $\mu_n \sim \mathcal{N}(0, 2.0^2 I_{20})$, and set $z_{n,t} = a_{n,t} \min\{1, 1/\|a_{n,t}\|\}$ with $a_{n,t} = b_t + \mu_n$. Labels are

$$y_{n,t} = \langle u^*, z_{n,t} \rangle + \nu_{n,t}, \quad \nu_{n,t} \sim \mathcal{N}(0, 0.03^2),$$

with independent label noise across agents and rounds. This construction preserves a common regression signal while making the local streams heterogeneous enough for consensus effects to be visible.

Figure 6.2 shows DOGD tuning sensitivity across the four ordered LIBSVM streams. The DOGD curves confirm that the best learning rate varies widely across datasets, and that a poorly chosen learning rate can lead to much worse performance than the parameter-free methods. DECO-EF and the centralized coin-betting baseline show consistent performance. Figure 6.3 illustrates communication and topology aspects. The complete-graph loss-difference panel shows that coarse quantization has the largest observable effect on final cumulative loss variance. The disagreement panel reveals the expected resolution effect: $b = 3$ leaves more cumulative disagreement than $b = 4, 5, 6$. The topology panel recovers the expected network effect: the grid and complete graph remove prediction disagreement much faster than the ring.

6.9 Conclusion

This chapter introduced DECO-EF, a decentralized online learning algorithm family that avoids OCO learning-rate tuning while communicating compressed state differences. Each node maintains both a clean accumulated state and a compressed tracker. The analysis shows that, under a linear gossip schedule, DECO-EF achieves the same expected $\tilde{O}(\max\{1, \|u\|\}\sqrt{T})$ regret rate as the best known centralized parameter-free methods, with a constant factor that depends on the network and compressor but not on T or $\|u\|$. Synthetic and real-data experiments illustrate the tuning sensitivity of a standard decentralized OCO method and the communication/disagreement tradeoffs of DECO-EF. A natural next target is the gap between the linear-schedule guarantee and the attractive constant communication regime $q(t) = O(1)$.

Chapter 7

Conclusion

This dissertation studied collaborative learning under communication constraints. The setting changed from chapter to chapter: online kernel learning over a graph, asynchronous federated learning, server-client distributed learning, and fully decentralized online optimization. The main point stayed the same. When data and computation are spread across a network, the learning rule and the communication rule cannot be designed separately.

The first part of the dissertation focused on compression. For online multi-kernel learning, gossip and hidden-state quantization removed the need for a complete communication graph while preserving sublinear regret. For asynchronous federated learning, QAFEL used a shared hidden state between the server and clients so that quantized communication did not accumulate uncontrolled error. These results show that compression can be built into the protocol without simply accepting the loss of accuracy that direct quantization may cause.

The next part studied feedback mechanisms for distributed learning. CAFEL and CAFEL-S used shared predictors to make client updates easier to compress. The important feature is that clients do not need to store their own error-feedback variables. This is useful in cross-device settings, where clients may appear only briefly and should not be assumed to maintain state

across rounds.

The last part considered decentralized online learning. DECO used coin-betting ideas to remove learning-rate tuning from decentralized online convex optimization. DECO-EF then added compressed gossip by tracking compressed state differences. Together, these chapters show that parameter-free online learning can be made compatible with decentralized communication, and that compressed messages can be handled without giving up network-regret guarantees.

There are still clear gaps between the theory and the regimes one would like to use in practice. The compressed DECO-EF guarantees use a growing gossip schedule, while the most attractive implementations use only a small number of communication rounds per step. The experiments suggest that these low-communication regimes can work, but the current proof does not fully explain them. Closing that gap is a natural next problem.

Another open direction is to combine the constraints treated separately here. Real systems can be asynchronous, compressed, decentralized, heterogeneous, and unreliable at the same time. The algorithms in this dissertation isolate parts of that picture. The longer-term goal is a theory and practice of collaborative learning that treats these constraints together.

The work in this dissertation is a step in that direction. It shows that careful state tracking, feedback, gossip, and parameter-free prediction can make distributed learning methods more tolerant of the networks they run on.

Bibliography

- [1] Y. Shen, S. Karimi-Bidhendi, and H. Jafarkhani, “Distributed and quantized online multi-kernel learning,” *IEEE Transactions on Signal Processing*, vol. 69, pp. 5496–5511, 2021.
- [2] A. Koloskova, S. Stich, and M. Jaggi, “Decentralized stochastic optimization and gossip algorithms with compressed communication,” in *Proceedings of the 36th International Conference on Machine Learning*. PMLR, May 2019, pp. 3478–3487.
- [3] D. Alistarh, D. Grubic, J. Li, R. Tomioka, and M. Vojnovic, “QSGD: Communication-efficient SGD via gradient quantization and encoding,” in *Advances in Neural Information Processing Systems*, I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, Eds., vol. 30. Curran Associates, Inc., 2017.
- [4] S. P. Karimireddy, Q. Rebjock, S. Stich, and M. Jaggi, “Error feedback fixes signSGD and other gradient compression schemes,” in *Proceedings of the 36th International Conference on Machine Learning*. PMLR, May 2019, pp. 3252–3261.
- [5] C.-C. Chang and C.-J. Lin, “LIBSVM: A library for support vector machines,” *ACM Transactions on Intelligent Systems and Technology*, vol. 2, no. 3, pp. 1–27, Apr. 2011. [Online]. Available: <https://dl.acm.org/doi/10.1145/1961189.1961199>
- [6] T. Ortega and H. Jafarkhani, “Gossiped and quantized online multi-kernel learning,” *IEEE Signal Processing Letters*, vol. 30, pp. 468–472, 2023.
- [7] —, “Asynchronous federated learning with bidirectional quantized communications and buffered aggregation,” *arXiv*, no. arXiv:2308.00263, Jul. 2023.
- [8] —, “Quantized and asynchronous federated learning,” *IEEE Transactions on Communications*, vol. 73, no. 4, pp. 2361–2374, 2025.
- [9] T. Ortega, C.-Y. Huang, X. Li, and H. Jafarkhani, “Communication compression for distributed learning without control variates,” *arXiv*, no. arXiv:2412.04538, Dec. 2024.
- [10] —, “Communication compression for distributed learning with aggregate and server-guided feedback,” *arXiv*, no. arXiv:2512.22623, 2025.
- [11] T. Ortega and H. Jafarkhani, “Decentralized parameter-free online learning,” *arXiv*, no. arXiv:2510.15644, Oct. 2025.

- [12] —, “Decentralized optimization in networks with arbitrary delays,” in *ICC 2024 - IEEE International Conference on Communications*, 2024, pp. 794–799.
- [13] —, “Decentralized optimization in time-varying networks with arbitrary delays,” *arXiv*, no. arXiv:2405.19513, Oct. 2024.
- [14] S. Ball and T. Ortega, “Geometric quasi-cyclic low density parity check codes Discrete Mathematics Days 2024,” *Discrete Mathematics Days 2024*, p. 228, 2024.
- [15] —, “Practical implementation of geometric quasi-cyclic ldpc codes,” *arXiv*, no. arXiv:2405.20524, 2024.
- [16] S. M. Ball Marks and T. Ortega Sánchez Colomer, “Quasi-Cyclic Ldpc Codes Based on Generalised Quadrangles,” WO Patent WO2023218050, Nov., 2023. [Online]. Available: <https://patentscope.wipo.int/search/en/detail.jsf?docId=WO2023218050&.cid=P11-LP8MC2-94041-1>
- [17] J. T. Mulvey, K. P. Iyer, T. Ortega, J. G. Merham, Y. Pivak, H. Sun, A. I. Hochbaum, and J. P. Patterson, “Correlating electrochemical stimulus to structural change in liquid electron microscopy videos using the structural dissimilarity metric,” *Ultramicroscopy*, vol. 257, p. 113894, 2024, publisher: North-Holland.
- [18] H. Jafarkhani and T. Ortega, “Systems and methods for quantized machine learning, federated learning and bidirectional network communication,” US Patent US Patent App. 18/636,406, Oct., 2024.
- [19] S. Hong and J. Chae, “Distributed online learning with multiple kernels,” *IEEE Transactions on Neural Networks and Learning Systems*, 2021.
- [20] S. Savazzi, M. Nicoli, M. Bennis, S. Kianoush, and L. Barbieri, “Opportunities of Federated Learning in Connected, Cooperative, and Automated Industrial Systems,” *IEEE Communications Magazine*, vol. 59, no. 2, pp. 16–21, Feb. 2021.
- [21] P. Kairouz, H. B. McMahan, B. Avent, A. Bellet, M. Bennis, A. Nitin Bhagoji *et al.*, “Advances and open problems in federated learning,” *Foundations and Trends® in Machine Learning*, vol. 14, no. 1-2, pp. 1–210, 2021.
- [22] S. C. H. Hoi, R. Jin, P. Zhao, and T. Yang, “Online Multiple Kernel Classification,” *Machine Learning*, vol. 90, no. 2, pp. 289–316, Feb. 2013. [Online]. Available: <https://doi.org/10.1007/s10994-012-5319-2>
- [23] D. Sahoo, S. C. Hoi, and B. Li, “Online multiple kernel regression,” in *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*. New York New York USA: ACM, Aug. 2014, pp. 293–302. [Online]. Available: <https://dl.acm.org/doi/10.1145/2623330.2623712>
- [24] C. Tang, Z. Li, J. Wang, X. Liu, W. Zhang, and E. Zhu, “Unified One-step Multi-view Spectral Clustering,” *IEEE Transactions on Knowledge and Data Engineering*, pp. 1–1, 2022.

- [25] B. Schölkopf, D. Schölkopf, A. J. Smola, and F. Bach, *Learning with Kernels: Support Vector Machines, Regularization, Optimization, and Beyond*. MIT Press, 2002, series Title: Adaptive computation and machine learning. [Online]. Available: <https://books.google.com/books?id=y8ORL3DWt4sC>
- [26] Y. Shen, T. Chen, and G. B. Giannakis, “Random feature-based online multi-kernel learning in environments with unknown dynamics,” *The Journal of Machine Learning Research*, vol. 20, no. 1, pp. 773–808, Jan. 2019.
- [27] A. Rahimi and B. Recht, “Random Features for Large-Scale Kernel Machines,” in *Advances in Neural Information Processing Systems*, vol. 20. Curran Associates, Inc., 2007. [Online]. Available: <https://proceedings.neurips.cc/paper/2007/hash/013a006f03dbc5392effeb8f18fda755-Abstract.html>
- [28] L. Xiao and S. Boyd, “Fast linear iterations for distributed averaging,” *Systems & Control Letters*, vol. 53, no. 1, pp. 65–78, Sep. 2004. [Online]. Available: <https://www.sciencedirect.com/science/article/pii/S0167691104000398>
- [29] S. Boyd, P. Diaconis, and L. Xiao, “Fastest mixing markov chain on a graph,” *SIAM review*, vol. 46, no. 4, pp. 667–689, 2004.
- [30] B. Gharesifard and J. Cortés, “When does a digraph admit a doubly stochastic adjacency matrix?” in *Proceedings of the 2010 American Control Conference*. IEEE, 2010, pp. 2440–2445.
- [31] A. Koloskova, T. Lin, S. U. Stich, and M. Jaggi, “Decentralized Deep Learning with Arbitrary Communication Compression,” in *International Conference on Learning Representations*, Apr. 2020. [Online]. Available: https://iclr.cc/virtual_2020/poster_SkgGCKrKvH.html
- [32] G. Rätsch, T. Onoda, and K.-R. Müller, “Soft margins for adaboost,” *Machine learning*, vol. 42, no. 3, pp. 287–320, 2001.
- [33] I.-C. Yeh and C.-h. Lien, “The comparisons of data mining techniques for the predictive accuracy of probability of default of credit card clients,” *Expert systems with applications*, vol. 36, no. 2, pp. 2473–2480, 2009.
- [34] D. Dua and C. Graff, “UCI machine learning repository,” 2017. [Online]. Available: <http://archive.ics.uci.edu/ml>
- [35] Y. LeCun, C. Cortes, and C. J.C. Burgess. (1998) The mnist database of handwritten digits. [Online]. Available: <http://yann.lecun.com/exdb/mnist/>
- [36] J. F. C. Mota, J. M. F. Xavier, P. M. Q. Aguiar, and M. Püschel, “A communication-efficient distributed algorithm for separable optimization,” *IEEE Transactions on Signal Processing*, vol. 61, no. 10, pp. 2718–2723, 2013.

- [37] B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y. Arcas, “Communication-efficient learning of deep networks from decentralized data,” in *Proceedings of the 20th International Conference on Artificial Intelligence and Statistics*. PMLR, Apr. 2017, pp. 1273–1282.
- [38] J. Guo, Z. Liu, S. Tian, F. Huang, J. Li, X. Li, K. K. Igorevich, and J. Ma, “TFL-DT: A trust evaluation scheme for federated learning in digital twin for mobile networks,” *IEEE Journal on Selected Areas in Communications*, vol. 41, no. 11, pp. 3548–3560, 2023. [Online]. Available: <https://ieeexplore.ieee.org/document/10234616>
- [39] M. Chen, Z. Yang, W. Saad, C. Yin, H. V. Poor, and S. Cui, “A joint learning and communications framework for federated learning over wireless networks,” *IEEE Transactions on Wireless Communications*, vol. 20, no. 1, pp. 269–283, Jan. 2021. [Online]. Available: <https://ieeexplore.ieee.org/abstract/document/9210812>
- [40] P. Villalobos, J. Sevilla, T. Besiroglu, L. Heim, A. Ho, and M. Hobbhahn, “Machine learning model sizes and the parameter gap,” *arXiv*, no. arXiv:2207.02852, Jul. 2022.
- [41] X. Li, K. Huang, W. Yang, S. Wang, and Z. Zhang, “On the convergence of fedavg on non-iid data,” in *ICLR 2020*, Jul. 2019. [Online]. Available: <https://arxiv.org/abs/1907.02189v4>
- [42] T. Li, A. K. Sahu, M. Zaheer, M. Sanjabi, A. Talwalkar, and V. Smith, “Federated optimization in heterogeneous networks,” in *MLSys 2020*. arXiv, Apr. 2020, arXiv:1812.06127 [cs, stat]. [Online]. Available: <http://arxiv.org/abs/1812.06127>
- [43] C. Xu, Y. Qu, Y. Xiang, and L. Gao, “Asynchronous federated learning on heterogeneous devices: A survey,” *arXiv*, Aug. 2022, arXiv:2109.04269 [cs]. [Online]. Available: <http://arxiv.org/abs/2109.04269>
- [44] Y. Chen, Y. Ning, M. Slawski, and H. Rangwala, “Asynchronous online federated learning for edge devices with non-iid data,” in *2020 IEEE International Conference on Big Data (Big Data)*. Los Alamitos, CA, USA: IEEE Computer Society, Dec. 2020, pp. 15–24. [Online]. Available: <https://doi.ieeecomputersociety.org/10.1109/BigData50022.2020.9378161>
- [45] D. Huba, J. Nguyen, K. Malik, R. Zhu, M. Rabbat, A. Yousefpour, C.-J. Wu, H. Zhan, P. Ustinov, H. Srinivas, K. Wang, A. Shoumikhin, J. Min, and M. Malek, “Papaya: Practical, private, and scalable federated learning,” *Proceedings of Machine Learning and Systems*, vol. 4, pp. 814–832, Apr. 2022. [Online]. Available: <https://proceedings.mlsys.org/paper/2022/hash/f340f1b1f65b6df5b5e3f94d95b11daf-Abstract.html>
- [46] J. Nguyen, K. Malik, H. Zhan, A. Yousefpour, M. Rabbat, M. Malek, and D. Huba, “Federated learning with buffered asynchronous aggregation,” in *Proceedings of The 25th International Conference on Artificial Intelligence and Statistics*. PMLR, May 2022, pp. 3581–3607. [Online]. Available: <https://proceedings.mlr.press/v151/nguyen22b.html>

- [47] M. T. Toghani and C. A. Uribe, “Unbounded gradients in federated learning with buffered asynchronous aggregation,” in *2022 58th Annual Allerton Conference on Communication, Control, and Computing (Allerton)*. IEEE, 2022, pp. 1–8.
- [48] Y. Miao, Z. Liu, X. Li, M. Li, H. Li, K.-K. R. Choo, and R. H. Deng, “Robust asynchronous federated learning with time-weighted and stale model aggregation,” *IEEE Transactions on Dependable and Secure Computing*, pp. 1–15, 2023.
- [49] Y. Miao, R. Xie, X. Li, Z. Liu, K.-K. R. Choo, and R. H. Deng, “Efficient and secure federated learning against backdoor attacks,” *IEEE Transactions on Dependable and Secure Computing*, pp. 1–18, 2024.
- [50] S. Zhou, Y. Huo, S. Bao, B. Landman, and A. Gokhale, “Fedaca: An adaptive communication-efficient asynchronous framework for federated learning,” in *2022 IEEE International Conference on Autonomic Computing and Self-Organizing Systems (ACSOS)*. IEEE, 2022, pp. 71–80.
- [51] N. Shlezinger, M. Chen, Y. C. Eldar, H. V. Poor, and S. Cui, “UveQFed: Universal vector quantization for federated learning,” *IEEE Transactions on Signal Processing*, vol. 69, pp. 500–514, 2021.
- [52] A. Albasyoni, M. Safaryan, L. Condat, and P. Richtárik, “Optimal gradient compression for distributed and federated learning,” in *SpicyFL 2020*. arXiv, 2020. [Online]. Available: <https://arxiv.org/abs/2010.03246>
- [53] S. Horváth and P. Richtarik, “A better alternative to error feedback for communication-efficient distributed learning,” in *International Conference on Learning Representations*, 2021. [Online]. Available: <https://openreview.net/forum?id=vYVI1CHPaQg>
- [54] N. Lang, E. Sofer, T. Shaked, and N. Shlezinger, “Joint privacy enhancement and quantization in federated learning,” *IEEE Transactions on Signal Processing*, vol. 71, pp. 295–310, 2023.
- [55] B. Girod and N. Farber, “Feedback-based error control for mobile video transmission,” *Proceedings of the IEEE*, vol. 87, no. 10, pp. 1707–1723, 1999.
- [56] I. E. G. Richardson, *Video Codec Design: Developing Image and Video Compression Systems*, 1st ed. Wiley, Apr. 2002.
- [57] A. Reibman, H. Jafarkhani, Y. Wang, M. Orchard, and R. Puri, “Multiple-description video coding using motion-compensated temporal prediction,” *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 12, no. 3, pp. 193–204, 2002.
- [58] R. Gray and D. Neuhoff, “Quantization,” *IEEE Transactions on Information Theory*, vol. 44, no. 6, pp. 2325–2383, Oct. 1998.

- [59] S. U. Stich, J.-B. Cordonnier, and M. Jaggi, “Sparsified SGD with memory,” in *Advances in Neural Information Processing Systems*, vol. 31. Curran Associates, Inc., 2018.
- [60] D. Alistarh, T. Hoefer, M. Johansson, N. Konstantinov, S. Khirirat, and C. Renggli, “The convergence of sparsified gradient methods,” in *Advances in Neural Information Processing Systems*, vol. 31. Curran Associates, Inc., 2018.
- [61] P. Richtarik, I. Sokolov, and I. Fatkhullin, “EF21: A new, simpler, theoretically better, and practically faster error feedback,” in *Advances in Neural Information Processing Systems*, vol. 34. Curran Associates, Inc., 2021, pp. 4384–4396.
- [62] C. Philippenko and A. Dieuleveut, “Preserved central model for faster bidirectional compression in distributed settings,” *Advances in Neural Information Processing Systems*, vol. 34, pp. 2387–2399, 2021.
- [63] J. Wang, Q. Liu, H. Liang, G. Joshi, and H. V. Poor, “Tackling the objective inconsistency problem in heterogeneous federated optimization,” in *Advances in Neural Information Processing Systems*, H. Larochelle, M. Ranzato, R. Hadsell, M. Balcan, and H. Lin, Eds., vol. 33. Curran Associates, Inc., 2020, pp. 7611–7623. [Online]. Available: https://proceedings.neurips.cc/paper_files/paper/2020/file/564127c03caab942e503ee6f810f54fd-Paper.pdf
- [64] C. Xie, S. Koyejo, and I. Gupta, “Asynchronous federated optimization,” in *OPT2020: 12th Annual Workshop on Optimization for Machine Learning*. arXiv, Dec. 2020, arXiv:1903.03934 [cs]. [Online]. Available: <http://arxiv.org/abs/1903.03934>
- [65] Y. Drori and O. Shamir, “The complexity of finding stationary points with stochastic gradient descent,” in *Proceedings of the 37th International Conference on Machine Learning*. PMLR, Nov. 2020, pp. 2658–2667. [Online]. Available: <https://proceedings.mlr.press/v119/drori20a.html>
- [66] S. Caldas, S. M. K. Duddu, P. Wu, T. Li, J. Konečný, H. B. McMahan, V. Smith, and A. Talwalkar, “LEAF: A benchmark for federated settings,” in *NeurIPS 2019*, Dec. 2018. [Online]. Available: <https://arxiv.org/abs/1812.01097v3>
- [67] A. Krizhevsky and G. Hinton, “Learning multiple layers of features from tiny images,” Master’s thesis, Department of Computer Science, University of Toronto, 2009.
- [68] A. Choromanska, M. Henaff, M. Mathieu, G. Ben Arous, and Y. LeCun, “The loss surfaces of multilayer networks,” in *Proceedings of the Eighteenth International Conference on Artificial Intelligence and Statistics*, ser. Proceedings of Machine Learning Research, G. Lebanon and S. V. N. Vishwanathan, Eds., vol. 38. San Diego, California, USA: PMLR, May 2015, pp. 192–204. [Online]. Available: <https://proceedings.mlr.press/v38/choromanska15.html>
- [69] Y. Nesterov, *Lectures on Convex Optimization*, ser. Springer Optimization and Its Applications. Cham: Springer International Publishing, 2018, vol. 137.

- [70] M. K. Nori, S. Yun, and I.-M. Kim, “Fast federated learning by balancing communication trade-offs,” *IEEE Transactions on Communications*, vol. 69, no. 8, pp. 5168–5182, 2021.
- [71] A. Khaled, K. Mishchenko, and P. Richtarik, “Tighter theory for local SGD on identical and heterogeneous data,” in *Proceedings of the Twenty Third International Conference on Artificial Intelligence and Statistics*. PMLR, Jun. 2020, pp. 4519–4529. [Online]. Available: <https://proceedings.mlr.press/v108/bayoumi20a.html>
- [72] L. Li, J. Wang, and C. Xu, “Flsim: An extensible and reusable simulation framework for federated learning,” in *Simulation Tools and Techniques*, H. Song and D. Jiang, Eds. Cham: Springer International Publishing, 2021, pp. 350–369. [Online]. Available: https://doi.org/10.1007/978-3-030-72792-5_30
- [73] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan *et al.*, “Pytorch: An imperative style, high-performance deep learning library,” in *Advances in Neural Information Processing Systems*, vol. 32. Curran Associates, Inc., 2019. [Online]. Available: <https://proceedings.neurips.cc/paper/2019/hash/bdbca288fee7f92f2bfa9f7012727740-Abstract.html>
- [74] Z. Liu, P. Luo, X. Wang, and X. Tang, “Deep learning face attributes in the wild,” in *2015 IEEE International Conference on Computer Vision (ICCV)*. Los Alamitos, CA, USA: IEEE Computer Society, Dec. 2015, pp. 3730–3738. [Online]. Available: <https://doi.ieeecomputersociety.org/10.1109/ICCV.2015.425>
- [75] T.-M. H. Hsu, H. Qi, and M. Brown, “Measuring the effects of non-identical data distribution for federated visual classification,” *arXiv*, no. arXiv:1909.06335, Sep. 2019. [Online]. Available: <http://arxiv.org/abs/1909.06335>
- [76] Y. Wu and K. He, “Group normalization,” in *Computer Vision – ECCV 2018*, ser. Lecture Notes in Computer Science, V. Ferrari, M. Hebert, C. Sminchisescu, and Y. Weiss, Eds. Cham: Springer International Publishing, 2018, pp. 3–19. [Online]. Available: <https://doi.org/10.1007/978-3-030-01261-8>
- [77] K. Hsieh, A. Phanishayee, O. Mutlu, and P. Gibbons, “The non-iid data quagmire of decentralized machine learning,” in *Proceedings of the 37th International Conference on Machine Learning*. PMLR, Nov. 2020, pp. 4387–4398. [Online]. Available: <https://proceedings.mlr.press/v119/hsieh20a.html>
- [78] J. Snoek, H. Larochelle, and R. P. Adams, “Practical bayesian optimization of machine learning algorithms,” in *Advances in Neural Information Processing Systems*, F. Pereira, C. Burges, L. Bottou, and K. Weinberger, Eds., vol. 25. Curran Associates, Inc., 2012. [Online]. Available: https://proceedings.neurips.cc/paper_files/paper/2012/file/05311655a15b75fab86956663e1819cd-Paper.pdf
- [79] Ookla, “Speedtest global index – internet speed around the world,” <https://www.speedtest.net/global-index>, 2025, accessed: 2025-12-16.

- [80] A. Aji and K. Heafield, “Sparse communication for distributed gradient descent,” in *EMNLP 2017: Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics (ACL), 2017.
- [81] T. Vogels, S. P. Karimireddy, and M. Jaggi, “PowerSGD: Practical low-rank gradient compression for distributed optimization,” *Advances in Neural Information Processing Systems*, vol. 32, 2019.
- [82] A. Beznosikov, S. Horváth, P. Richtárik, and M. Safaryan, “On biased compression for distributed learning,” *Journal of Machine Learning Research*, vol. 24, no. 276, pp. 1–50, 2023.
- [83] S. P. Karimireddy, S. Kale, M. Mohri, S. Reddi, S. Stich, and A. T. Suresh, “SCAF-FOLD: Stochastic controlled averaging for federated learning,” in *Proceedings of the 37th International Conference on Machine Learning*. PMLR, Nov. 2020, pp. 5132–5143.
- [84] E. Gorbunov, K. P. Burlachenko, Z. Li, and P. Richtarik, “Marina: Faster non-convex distributed learning with compression,” in *Proceedings of the 38th International Conference on Machine Learning*. PMLR, Jul. 2021, pp. 3788–3798.
- [85] K. Gruntkowska, A. Tyurin, and P. Richtárik, “Ef21-p and friends: Improved theoretical communication complexity for distributed optimization with bidirectional compression,” in *Proceedings of the 40th International Conference on Machine Learning*. PMLR, Jul. 2023, pp. 11 761–11 807.
- [86] Y. Gao, A. Rodomanov, J. Rack, and S. U. Stich, “Accelerated distributed optimization with compression and error feedback,” *arXiv*, no. arXiv:2503.08427, Mar. 2025.
- [87] T. Ortega, “CAFe,” <https://github.com/TomasOrtega/CAFe>, 2025.
- [88] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, “Gradient-based learning applied to document recognition,” *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.
- [89] G. Cohen, S. Afshar, J. Tapson, and A. Van Schaik, “EMNIST: Extending mnist to handwritten letters,” in *2017 international joint conference on neural networks (IJCNN)*. IEEE, 2017, pp. 2921–2926.
- [90] B. Isik, F. Pase, D. Gunduz, S. Koyejo, T. Weissman, and M. Zorzi, “Adaptive compression in federated learning via side information,” in *International Conference on Artificial Intelligence and Statistics*. PMLR, 2024, pp. 487–495.
- [91] S. B. Harma, A. Chakraborty, E. Kostenok, D. Mishin, D. Ha, B. Falsafi, M. Jaggi, M. Liu, Y. Oh, S. Subramanian *et al.*, “Effective interplay between sparsity and quantization: From theory to practice,” *arXiv preprint arXiv:2405.20935*, 2024.
- [92] E. Hazan, “Introduction to Online Convex Optimization,” *arXiv*, Aug. 2023, arXiv:1909.05207 [cs]. [Online]. Available: <http://arxiv.org/abs/1909.05207>

- [93] S. Shalev-Shwartz, “Online Learning and Online Convex Optimization,” *Foundations and Trends® in Machine Learning*, vol. 4, no. 2, pp. 107–194, 2011. [Online]. Available: <http://www.nowpublishers.com/article/Details/MAL-018>
- [94] F. Orabona and D. Pál, “Coin Betting and Parameter-Free Online Learning,” *arXiv*, Nov. 2016, arXiv:1602.04128 [cs]. [Online]. Available: <http://arxiv.org/abs/1602.04128>
- [95] B. McMahan and M. Streeter, “No-Regret Algorithms for Unconstrained Online Convex Optimization,” in *Advances in Neural Information Processing Systems*, vol. 25. Curran Associates, Inc., 2012. [Online]. Available: https://proceedings.neurips.cc/paper_files/paper/2012/hash/38ca89564b2259401518960f7a06f94b-Abstract.html
- [96] H. B. McMahan and F. Orabona, “Unconstrained Online Linear Learning in Hilbert Spaces: Minimax Algorithms and Normal Approximations,” in *Proceedings of The 27th Conference on Learning Theory*. PMLR, May 2014, pp. 1020–1039, iSSN: 1938-7228. [Online]. Available: <https://proceedings.mlr.press/v35/mcmahan14.html>
- [97] A. Cutkosky and K. Boahen, “Online Convex Optimization with Unconstrained Domains and Losses,” *arXiv*, Mar. 2017, arXiv:1703.02622 [cs]. [Online]. Available: <http://arxiv.org/abs/1703.02622>
- [98] D. van der Hoeven, A. Cutkosky, and H. Luo, “Comparator-adaptive convex bandits,” in *Proceedings of the 34th International Conference on Neural Information Processing Systems*, ser. NIPS ’20. Red Hook, NY, USA: Curran Associates Inc., Dec. 2020, pp. 19 795–19 804.
- [99] D. v. d. Hoeven, H. Hadiji, and T. v. Erven, “Distributed Online Learning for Joint Regret with Communication Constraints,” *arXiv*, Oct. 2021, arXiv:2102.07521 [cs]. [Online]. Available: <http://arxiv.org/abs/2102.07521>
- [100] A. Nedic and A. Ozdaglar, “On the rate of convergence of distributed subgradient methods for multi-agent optimization,” in *2007 46th IEEE Conference on Decision and Control*, Dec. 2007, pp. 4711–4716, iSSN: 0191-2216. [Online]. Available: <https://ieeexplore.ieee.org/document/4434693>
- [101] B. Johansson, M. Rabi, and M. Johansson, “A simple peer-to-peer algorithm for distributed optimization in sensor networks,” in *2007 46th IEEE Conference on Decision and Control*, Dec. 2007, pp. 4705–4710, iSSN: 0191-2216. [Online]. Available: <https://ieeexplore.ieee.org/document/4434888>
- [102] J. Achddou, N. Cesa-Bianchi, and H. Qiu, “Distributed Online Optimization with Stochastic Agent Availability,” *arXiv*, Nov. 2024, arXiv:2411.16477 [cs]. [Online]. Available: <http://arxiv.org/abs/2411.16477>
- [103] Y.-G. Hsieh, F. Iutzeler, J. Malick, and P. Mertikopoulos, “Multi-agent online optimization with delays: asynchronicity, adaptivity, and optimism,” *J. Mach. Learn. Res.*, vol. 23, no. 1, pp. 78:3377–78:3425, Jan. 2022.

- [104] N. Cesa-Bianchi, A. Conconi, and C. Gentile, “On the generalization ability of on-line learning algorithms,” *IEEE Transactions on Information Theory*, vol. 50, no. 9, pp. 2050–2057, Sep. 2004. [Online]. Available: <https://ieeexplore.ieee.org/document/1327806>
- [105] S. Boyd, A. Ghosh, B. Prabhakar, and D. Shah, “Randomized gossip algorithms,” *IEEE Transactions on Information Theory*, vol. 52, no. 6, pp. 2508–2530, Jun. 2006. [Online]. Available: <http://ieeexplore.ieee.org/document/1638541/>
- [106] P. Erdős and A. Rényi, “On random graphs. I.” *Publicationes Mathematicae Debrecen*, vol. 6, no. 3-4, pp. 290–297, 1959. [Online]. Available: https://publi.math.unideb.hu/load_doi.php?doi=10.5486_PMD.1959_6.3.4.12
- [107] F. Orabona, “A Modern Introduction to Online Learning,” *arXiv*, May 2023, arXiv:1912.13213 [cs]. [Online]. Available: <http://arxiv.org/abs/1912.13213>
- [108] M. Rabbat and R. Nowak, “Distributed optimization in sensor networks,” in *Third International Symposium on Information Processing in Sensor Networks, 2004. IPSN 2004*, Apr. 2004, pp. 20–27. [Online]. Available: <https://ieeexplore.ieee.org/document/1307319>
- [109] A. Nedić and J. Liu, “Distributed Optimization for Control,” *Annual Review of Control, Robotics, and Autonomous Systems*, vol. 1, no. 1, pp. 77–103, May 2018. [Online]. Available: <https://www.annualreviews.org/doi/10.1146/annurev-control-060117-105131>
- [110] N. Cesa-Bianchi, T. Cesari, and C. Monteleoni, “Cooperative online learning: Keeping your neighbors updated,” in *Proceedings of the 31st International Conference on Algorithmic Learning Theory*. PMLR, Jan. 2020, pp. 234–250. [Online]. Available: <https://proceedings.mlr.press/v117/cesa-bianchi20a.html>
- [111] N. Cesa-Bianchi, T. R. Cesari, and R. Della Vecchia, “Cooperative online learning with feedback graphs,” *arXiv preprint arXiv:2106.04982*, 2021.
- [112] J. C. Duchi, A. Agarwal, and M. J. Wainwright, “Dual Averaging for Distributed Optimization: Convergence Analysis and Network Scaling,” *IEEE Transactions on Automatic Control*, vol. 57, no. 3, pp. 592–606, Mar. 2012. [Online]. Available: <https://ieeexplore.ieee.org/abstract/document/5936104>
- [113] A. Nedić, S. Lee, and M. Raginsky, “Decentralized online optimization with global objectives and local communication,” in *2015 American Control Conference (ACC)*, Jul. 2015, pp. 4497–4503, iSSN: 2378-5861. [Online]. Available: <https://ieeexplore.ieee.org/document/7172037>
- [114] A. Nedić, A. Olshevsky, and M. G. Rabbat, “Network Topology and Communication-Computation Tradeoffs in Decentralized Optimization,” *Proceedings of the IEEE*, vol. 106, no. 5, pp. 953–976, May 2018. [Online]. Available: <https://ieeexplore.ieee.org/document/8340193>

- [115] S. Hosseini, A. Chapman, and M. Mesbahi, “Online distributed optimization via dual averaging,” in *52nd IEEE Conference on Decision and Control*, Dec. 2013, pp. 1484–1489, iSSN: 0191-2216. [Online]. Available: <https://ieeexplore.ieee.org/document/6760092>
- [116] A. Koloskova, S. Stich, and M. Jaggi, “Decentralized Stochastic Optimization and Gossip Algorithms with Compressed Communication,” in *Proceedings of the 36th International Conference on Machine Learning*. PMLR, May 2019, pp. 3478–3487, iSSN: 2640-3498. [Online]. Available: <https://proceedings.mlr.press/v97/koloskova19a.html>
- [117] S. Yang, W. Yang, W. Jiang, and L. Zhang, “Improved regret for decentralized online convex optimization with compressed communication,” Oct. 2025. [Online]. Available: <https://openreview.net/forum?id=QTh3aRcbTt>
- [118] X. Cao and T. Başar, “Decentralized online convex optimization with compressed communications,” *Automatica*, vol. 156, p. 111186, Oct. 2023.

Appendix A

Supporting Material for Chapter 3

A.1 Additional Derivations

This appendix contains the derivations of the necessary lemmas for the main result. First, we present some useful remarks:

REMARK A.1. *Given an unbiased quantizer Q with parameter δ , for any set of $N \geq 1$ vectors $\{x_n \in \mathbb{R}^d, n = 1, \dots, N\}$, we have*

$$\mathbb{E}_Q \left[\left\| \sum_{n=1}^N Q(x_n) \right\|^2 \right] \leq \left\| \sum_{n=1}^N x_n \right\|^2 + (1 - \delta) \sum_{n=1}^N \|x_n\|^2.$$

The remark follows immediately from the unbiasedness of Q and our definition of a quantizer, see Definition 3.1.

LEMMA A.2. *Consider the optimization problem in Equation (3.5) satisfying Assumptions 3.2*

to 3.5 and 3.7. Then, QAFEL's iterations satisfy

$$\begin{aligned}
\mathbb{E} \left[-\langle \nabla f(x^t), \bar{\Delta}^t \rangle \right] &\leq -\frac{P\eta_\ell}{2} \mathbb{E} \left[\|\nabla f(x^t)\|^2 \right] - \frac{\eta_\ell}{2} r_t \\
&\quad + P\eta_\ell L^2 \mathbb{E} \left[2\tau_{\max, K} \eta_g^2 \sum_{s=t-\tau_{\max, K}}^{t-1} \|\bar{\Delta}^s\|^2 \right. \\
&\quad + \frac{12}{\delta_s} \eta_g^2 \sum_{s=1}^{t-1} \left(1 - \frac{\delta_s}{2} \right)^s \left\| \bar{\Delta}^{t-s-1} \right\|^2 \\
&\quad \left. + 20P^2 \eta_\ell^2 (\sigma_\ell^2 + B + \sum_{s=1}^{\tau_{\max, K}} \|\nabla f(x^{t-s})\|^2) \right],
\end{aligned} \tag{A.1}$$

where $r_t = \sum_{p=0}^{P-1} \mathbb{E} \left[\frac{1}{K} \sum_{k \in \mathcal{S}_t} \|\nabla F_k(y_{k,p}^{t-\tau_k})\|^2 \right]$.

Proof. Let us start by expanding the definition of the server update, which is the average of client updates,

$$\bar{\Delta}^t = \frac{1}{K} \sum_{k \in \mathcal{S}_t} \Delta_k^{t-\tau_k}. \tag{A.2}$$

We have defined the set of clients that participate in the update at time t as $\mathcal{S}_t$ and the update from Client k as $\Delta_k^{t-\tau_k}$, where $t - \tau_k$ represents the time index of the hidden-state used for the update. Also, $\Delta_k^{t-\tau_k} = Q_c(y_{k,0}^{t-\tau_k} - y_{k,P}^{t-\tau_k})$, where $y_{k,p}^{t-\tau_k}$ is the local model after p local updates. The initial model used for updating $\Delta_k^{t-\tau_k}$ is $y_{k,0}^{t-\tau_k} = \hat{x}^{t-\tau_k}$. Now, since the client quantizer Q_c is unbiased, we have

$$\begin{aligned}
\mathbb{E} [\Delta_k^{t-\tau_k}] &= \mathbb{E} [y_{k,0}^{t-\tau_k} - y_{k,P}^{t-\tau_k}] \\
&= \mathbb{E} \left[\sum_{p=0}^{P-1} \eta_\ell g_k(y_{k,p}^{t-\tau_k}) \right] \\
&= \mathbb{E} \left[\sum_{p=0}^{P-1} \eta_\ell \nabla F_k(y_{k,p}^{t-\tau_k}) \right],
\end{aligned} \tag{A.3}$$

where the last equality is derived from Assumption 3.2. Then, applying the linearity of the

expectation and using the last two equations show that $\mathbb{E} \left[\langle \nabla f(x^t), \bar{\Delta}^t \rangle \right]$ is

$$\mathbb{E} \left[\left\langle \nabla f(x^t), \frac{1}{K} \sum_{k \in \mathcal{S}_t} \sum_{p=0}^{P-1} \eta_\ell \nabla F_k(y_{k,p}^{t-\tau_k}) \right\rangle \right]. \quad (\text{A.4})$$

We re-arrange and obtain

$$\sum_{p=0}^{P-1} \eta_\ell \mathbb{E} \left[\frac{1}{K} \sum_{k \in \mathcal{S}_t} \langle \nabla f(x^t), \nabla F_k(y_{k,p}^{t-\tau_k}) \rangle \right]. \quad (\text{A.5})$$

Using the well-known identity $\langle a, b \rangle = \frac{1}{2}(\|a\|^2 + \|b\|^2 - \|a - b\|^2)$, and the definition of r_t , we obtain

$$\begin{aligned} \mathbb{E} \left[-\langle \nabla f(x^t), \bar{\Delta}^t \rangle \right] &= -\frac{P\eta_\ell}{2} \mathbb{E} \left[\|\nabla f(x^t)\|^2 \right] - \frac{\eta_\ell}{2} r_t \\ &\quad + \sum_{p=0}^{P-1} \frac{\eta_\ell}{2} \mathbb{E} \left[\frac{1}{K} \sum_{k \in \mathcal{S}_t} \|\nabla f(x^t) - \nabla F_k(y_{k,p}^{t-\tau_k})\|^2 \right]. \end{aligned} \quad (\text{A.6})$$

For the last term, we can add and subtract $\nabla f(y_{k,p}^{t-\tau_k})$, apply Cauchy-Schwarz and use Assumption 3.4 to obtain

$$\mathbb{E} \|\nabla f(x^t) - \nabla F_k(y_{k,p}^{t-\tau_k})\|^2 \leq 2L^2 \mathbb{E} \|x^t - y_{k,p}^{t-\tau_k}\|^2. \quad (\text{A.7})$$

Applying Cauchy-Schwarz once again, we obtain a decomposition in two error terms

$$\|x^t - y_{k,p}^{t-\tau_k}\|^2 \leq 2 \underbrace{\|x^t - x^{t-\tau_k}\|^2}_{\text{staleness}} + 2 \underbrace{\|x^{t-\tau_k} - y_{k,p}^{t-\tau_k}\|^2}_{\text{local drift}}.$$

We bound the staleness term with Assumption 3.7,

$$\begin{aligned}
\|x^t - x^{t-\tau_k}\|^2 &\leq \tau_{\max,K} \sum_{s=t-\tau_{\max,K}}^{t-1} \|x^{s+1} - x^s\|^2 \\
&= \tau_{\max,K} \eta_g^2 \sum_{s=t-\tau_{\max,K}}^{t-1} \|\overline{\Delta}^s\|^2.
\end{aligned} \tag{A.8}$$

Using Lemma A.5 with $z = x^{t-\tau_k}$ we bound the local drift term with

$$\begin{aligned}
\mathbb{E} \left[\|y_{k,p}^{t-\tau_k} - y_{k,0}^{t-\tau_k}\|^2 \right] &\leq 10P^2 \eta_\ell^2 \left(\sigma_\ell^2 + \mathbb{E} \left[\|\nabla f(x^{t-\tau_k})\|^2 \right] \right. \\
&\quad \left. + \mathbb{E} \left[\|\nabla F_k(x^{t-\tau_k}) - \nabla f(x^{t-\tau_k})\|^2 \right] \right) \\
&\quad + 3 \underbrace{\|\hat{x}^{t-\tau_k} - x^{t-\tau_k}\|^2}_{\text{server quantization}}.
\end{aligned}$$

Then, using Assumptions 3.5 and 3.7, we obtain

$$\begin{aligned}
\mathbb{E} \left[\|y_{k,p}^{t-\tau_k} - y_{k,0}^{t-\tau_k}\|^2 \right] &\leq 10P^2 \eta_\ell^2 (\sigma_\ell^2 + B) + 10P^2 \eta_\ell^2 \sum_{s=1}^{\tau_{\max,K}} \mathbb{E} \left[\|\nabla f(x^{t-s})\|^2 \right] \\
&\quad + 3 \underbrace{\|\hat{x}^{t-\tau_k} - x^{t-\tau_k}\|^2}_{\text{server quantization}}.
\end{aligned}$$

We can bound the server quantization term as

$$\mathbb{E}_Q \|x^{t-\tau_k} - \hat{x}^{t-\tau_k}\|^2 \leq (1 + c^{-1}) \eta_g^2 \left\| \overline{\Delta}^{t-\tau_k-1} \right\|^2 + (1 + c)(1 - \delta_s) \|x^{t-\tau_k-1} - \hat{x}^{t-\tau_k-1}\|^2.$$

Selecting $c = \delta_s/2$, we obtain

$$\mathbb{E}_Q \|x^{t-\tau_k} - \hat{x}^{t-\tau_k}\|^2 \leq \frac{2}{\delta_s} \eta_g^2 \left\| \overline{\Delta}^{t-\tau_k-1} \right\|^2 + \left(1 - \frac{\delta_s}{2}\right) \|x^{t-\tau_k-1} - \hat{x}^{t-\tau_k-1}\|^2. \tag{A.9}$$

Inductively, this yields

$$\mathbb{E}_Q \left\| x^{t-\tau_k} - \hat{x}^{t-\tau_k} \right\|^2 \leq \frac{2}{\delta_s} \eta_g^2 \sum_{s=1}^{t-1} \left(1 - \frac{\delta_s}{2} \right)^s \left\| \overline{\Delta}^{t-s-1} \right\|^2.$$

Substituting the staleness, server quantization, and local drift terms provides

$$\begin{aligned} \mathbb{E} \left[\left\| x^t - y_{k,p}^{t-\tau_k} \right\|^2 \right] &\leq 2\tau_{\max,K} \eta_g^2 \sum_{s=t-\tau_{\max,K}}^{t-1} \mathbb{E} \left[\left\| \overline{\Delta}^s \right\|^2 \right] \\ &\quad + \frac{12}{\delta_s} \eta_g^2 \sum_{s=1}^{t-1} \left(1 - \frac{\delta_s}{2} \right)^s \mathbb{E} \left[\left\| \overline{\Delta}^{t-s-1} \right\|^2 \right] \\ &\quad + 20P^2 \eta_\ell^2 (\sigma_\ell^2 + B) \\ &\quad + 20P^2 \eta_\ell^2 \sum_{s=1}^{\tau_{\max,K}} \mathbb{E} \left[\left\| \nabla f(x^{t-s}) \right\|^2 \right]. \end{aligned}$$

Substituting in Equation (A.7) and then Equation (A.6) proves the lemma. $\square$

LEMMA A.3. *Consider the optimization problem in Equation (3.5) satisfying Assumptions 3.2 and 3.3. Then, QAFEL's iterations satisfy*

$$\mathbb{E} \left[\left\| \overline{\Delta}^t \right\|^2 \right] \leq \frac{P\eta_\ell^2}{K} (2 - \delta_c) \sigma_\ell^2 + \left(1 + \frac{1 - \delta_c}{K} \right) \mathbb{E} \left[\frac{1}{K} \sum_{k \in \mathcal{S}_t} \left\| \sum_{p=0}^{P-1} \eta_\ell \nabla F_k(y_{k,p}^{t-\tau_k}) \right\|^2 \right]. \quad (\text{A.10})$$

Proof. First, using the unbiasedness of the client quantizer, we apply Remark A.1 as follows

$$\begin{aligned} \mathbb{E} \left[\left\| \overline{\Delta}^t \right\|^2 \right] &= \mathbb{E} \left[\left\| \frac{1}{K} \sum_{k \in \mathcal{S}_t} Q_c \left(\sum_{p=0}^{P-1} \eta_\ell g_k(y_{k,p}^{t-\tau_k}) \right) \right\|^2 \right] \\ &\leq \mathbb{E} \left[\left\| \frac{1}{K} \sum_{k \in \mathcal{S}_t} \sum_{p=0}^{P-1} \eta_\ell g_k(y_{k,p}^{t-\tau_k}) \right\|^2 \right] \\ &\quad + (1 - \delta_c) \mathbb{E} \left[\frac{1}{K^2} \sum_{k \in \mathcal{S}_t} \left\| \sum_{p=0}^{P-1} \eta_\ell g_k(y_{k,p}^{t-\tau_k}) \right\|^2 \right]. \end{aligned}$$

Then, using the unbiasedness of the stochastic gradient on both terms completes the proof. $\square$

COROLLARY A.4. *Consider the optimization problem in Equation (3.5) satisfying Assumptions 3.2 and 3.3. Then, QAFEL's iterations satisfy*

$$\mathbb{E} \left[\left\| \overline{\Delta}^t \right\|^2 \right] \leq \frac{P\eta_\ell^2}{K} (2 - \delta_c) \sigma_\ell^2 + \left(1 + \frac{1 - \delta_c}{K} \right) P\eta_\ell^2 r_t, \quad (\text{A.11})$$

where $r_t = \sum_{p=0}^{P-1} \mathbb{E} \left[\frac{1}{K} \sum_{k \in \mathcal{S}_t} \left\| \nabla F_k(y_{k,p}^{t-\tau_k}) \right\|^2 \right]$.

The following Lemma is a generalization of [47, Equations (52) to (64)].

LEMMA A.5. *Given any $z \in \mathbb{R}^d$, if Assumptions 3.3 and 3.4 are satisfied and $\eta_\ell \leq \frac{1}{4L(P+1)}$, QAFEL's k -th local step satisfies*

$$\begin{aligned} \mathbb{E} \left[\left\| y_{k,p}^s - z \right\|^2 \right] &\leq 3\mathbb{E} \left[\left\| y_{k,0}^s - z \right\|^2 \right] \\ &+ 10P^2\eta_\ell^2 \mathbb{E} \left[\sigma_\ell^2 + \left\| \nabla f(z) \right\|^2 + \left\| \nabla F_k(z) - \nabla f(z) \right\|^2 \right]. \end{aligned} \quad (\text{A.12})$$

Proof. We start by expanding the p -th local step:

$$\left\| y_{k,p}^s - z \right\|^2 = \left\| y_{k,p-1}^s - \eta_\ell g_k(y_{k,p-1}^s) - z \right\|^2. \quad (\text{A.13})$$

Next, we add and subtract $\eta_\ell \nabla F_k(y_{k,p-1}^s)$, $\eta_\ell \nabla F_k(z)$, and $\eta_\ell \nabla f(z)$, and apply Cauchy-

Schwarz:

$$\begin{aligned}
\|y_{k,p}^s - z\|^2 &\leq (1+c) \|y_{k,p-1}^s - z\|^2 \\
&\quad + 4\eta_\ell^2(1+c^{-1}) \|g_k(y_{k,p-1}^s) - \nabla F_k(y_{k,p-1}^s)\|^2 \\
&\quad + 4\eta_\ell^2(1+c^{-1}) \|\nabla F_k(y_{k,p-1}^s) - \nabla F_k(z)\|^2 \\
&\quad + 4\eta_\ell^2(1+c^{-1}) \|\nabla F_k(z) - \nabla f(z)\|^2 \\
&\quad + 4\eta_\ell^2(1+c^{-1}) \|\nabla f(z)\|^2.
\end{aligned}$$

Selecting $c = \frac{1}{2P}$, and applying Assumptions 3.3 to 3.5, we obtain

$$\begin{aligned}
\mathbb{E} [\|y_{k,p}^s - z\|^2] &\leq \left(1 + \frac{1}{2P}\right) \mathbb{E} [\|y_{k,p-1}^s - z\|^2] \\
&\quad + 4\eta_\ell^2(1+2P)\sigma_\ell^2 \\
&\quad + 4\eta_\ell^2(1+2P)L^2\mathbb{E} [\|y_{k,p-1}^s - z\|^2] \\
&\quad + 4\eta_\ell^2(1+2P)\mathbb{E} [\|\nabla F_k(z) - \nabla f(z)\|^2] \\
&\quad + 4\eta_\ell^2(1+2P) \|\nabla f(z)\|^2.
\end{aligned}$$

Now, using that $\eta_\ell \leq \frac{1}{4L(P+1)} \implies 4\eta_\ell^2(1+2P)L^2 \leq \frac{1}{2P}$, we define the U_p series and R term for ease of notation in the next steps as follows:

$$\begin{aligned}
\underbrace{\mathbb{E} [\|y_{k,p}^s - z\|^2]}_{U_p} &\leq \left(1 + \frac{1}{P}\right) \underbrace{\mathbb{E} [\|y_{k,p-1}^s - z\|^2]}_{U_{p-1}} \\
&\quad + \underbrace{4\eta_\ell^2(1+2P)(\sigma_\ell^2 + \mathbb{E} [\|\nabla F_k(z) - \nabla f(z)\|^2 + \|\nabla f(z)\|^2])}_R.
\end{aligned}$$

Finally, we use this recursion p times and obtain

$$\begin{aligned}
U_p &\leq U_0 \left(1 + \frac{1}{P}\right)^p + R \sum_{i=0}^{p-1} \left(1 + \frac{1}{P}\right)^i \\
&\leq U_0 e + R(e-1)P,
\end{aligned} \tag{A.14}$$

where we have used the geometric series sum and basic properties of the exponential function to obtain the last inequality. To finish the proof, we just apply the definitions of U_0 and R , and bound $e < 3$, as well as $(e-1)4(1+2P)P < 10P^2$, given that P is no less than 1. $\square$

A.2 Algorithm Pseudocode

This appendix contains the pseudocode for all the components of QAFEL, see Algorithms 7 to 9. The highlighted lines are the key novelties of the algorithm, which constitute the hidden-state mechanism.

Algorithm 7 QAFEL-server

Input: server learning rate η_g , client learning rate η_ℓ , client SGD steps P , buffer size K , initial model x^0

Output: FL-trained global model

```
1:  $\hat{x}^0 \leftarrow x^0$  // Initialize shared hidden state
2: while not converged do
3:   Run QAFEL-client( $\eta_\ell, P$ ) on clients // Asynchronous execution
4:   if received client update from client  $n$  then
5:      $\Delta_n \leftarrow$  received quantized update
6:      $\bar{\Delta}^t \leftarrow \bar{\Delta}^t + \Delta_n$ 
7:      $k \leftarrow k + 1$  // Increment clients in buffer
8:   end if
9:   if  $k == K$  then
10:     $\bar{\Delta}^t \leftarrow \frac{\bar{\Delta}^t}{K}$ 
11:     $x^{t+1} \leftarrow x^t + \eta_g \bar{\Delta}^t$ 
12:    Broadcast  $q^t \leftarrow Q_s(x^{t+1} - \hat{x}^t)$ 
13:     $\hat{x}^{t+1} \leftarrow \hat{x}^t + q^t$  // Update (shared) hidden state
14:     $\bar{\Delta}^t \leftarrow 0, k \leftarrow 0, t \leftarrow t + 1$  // Reset buffer
15:   end if
16: end while
```

Algorithm 8 QAFEL-client

Input: client learning rate η_ℓ , number of client SGD steps P

Output: client update Δ

```
1:  $y_0 \leftarrow \hat{x}^t$  // Ensured by Algorithm 9
2: for  $p \leftarrow 1$  to  $P$  do
3:    $y_p \leftarrow y_{p-1} - \eta_\ell g_p(y_{p-1})$ 
4: end for
5:  $\Delta \leftarrow Q_c(y_0 - y_p)$  // Using an unbiased quantizer
6: Send  $\Delta$  to server
```

Algorithm 9 QAFEL-client-background

Input: initial model x^0

Output: updated shared hidden state $\hat{x}$

```
1:  $\hat{x}^0 \leftarrow x^0$ 
2: repeat
3:   wait for quantized update  $q^t$  from server
4:    $\hat{x}^{t+1} \leftarrow \hat{x}^t + q^t$ 
5: until shutdown
```

Table B.1: Optimal learning rates for CAFE and Direct compression (MNIST, EMNIST, CIFAR-10, CIFAR-100).

Compressor	Param.	MNIST (CONV4, 50 rounds)				EMNIST (CONV4, 50 rounds)				CIFAR-10 (CONV6, 200 rounds)				CIFAR-100 (ResNet-20, 200 rounds)			
		iid		non-iid		iid		non-iid		iid		non-iid		iid		non-iid	
		10 classes/client		4 classes/client		47 classes/client		4 classes/client		10 classes/client		4 classes/client		100 classes/client		40 classes/client	
		Direct	CAFe	Direct	CAFe	Direct	CAFe	Direct	CAFe	Direct	CAFe	Direct	CAFe	Direct	CAFe	Direct	CAFe
None	—	$10^{-0.5}$		0.1		$10^{-0.5}$		0.1		$10^{-0.5}$		0.1		$10^{-0.5}$		1	
Top-k	k=10%	$10^{-0.5}$	$10^{-0.5}$	$10^{-0.5}$	0.1	$10^{-0.5}$	$10^{-0.5}$	0.1	0.1	$10^{-0.5}$	0.1	$10^{-0.5}$	0.1	$10^{-0.5}$	$10^{-0.5}$	1	1
	k=1%	$10^{-0.5}$	$10^{-0.5}$	0.1	0.1	$10^{-0.5}$	$10^{-0.5}$	0.1	0.1	$10^{-0.5}$	0.1	$10^{-0.5}$	0.1	1	$10^{-0.5}$	1	1
	k=0.1%	0.1	0.1	$10^{-0.5}$	0.1	0.1	0.1	0.1	0.1	$10^{-0.5}$	0.1	0.1	0.1	1	0.1	1	$10^{-0.5}$
Top-1% + Quantizer	6 bits	$10^{-0.5}$	$10^{-0.5}$	0.1	0.1	$10^{-0.5}$	$10^{-0.5}$	0.1	0.1	$10^{-0.5}$	0.1	$10^{-0.5}$	0.1	$10^{-0.5}$	1	1	1
	5 bits	$10^{-0.5}$	$10^{-0.5}$	0.1	0.1	$10^{-0.5}$	$10^{-0.5}$	$10^{-0.5}$	0.1	$10^{-0.5}$	$10^{-0.5}$	$10^{-0.5}$	0.1	1	$10^{-0.5}$	1	1
	4 bits	$10^{-0.5}$	0.1	0.1	0.1	$10^{-0.5}$	$10^{-0.5}$	0.1	0.1	0.01	$10^{-0.5}$	0.1	0.1	$10^{-0.5}$	$10^{-0.5}$	1	$10^{-0.5}$
Low-rank	rank=3	0.1	$10^{-0.5}$	0.1	0.1	0.1	0.1	0.1	0.1	$10^{-0.5}$	0.1	0.1	0.1	$10^{-0.5}$	$10^{-0.5}$	1	$10^{-0.5}$
	rank=2	0.1	0.1	0.1	0.1	$10^{-0.5}$	0.1	$10^{-1.5}$	$10^{-1.5}$	0.1	0.1	0.1	0.1	$10^{-0.5}$	$10^{-0.5}$	1	$10^{-0.5}$
	rank=1	0.1	0.1	0.1	0.1	0.1	0.1	$10^{-1.5}$	$10^{-1.5}$	$10^{-0.5}$	0.1	0.1	0.1	$10^{-0.5}$	$10^{-0.5}$	1	$10^{-0.5}$
Rank-1 + Quantizer	5 bits	0.1	0.1	0.1	0.1	0.1	0.1	$10^{-1.5}$	$10^{-1.5}$	$10^{-0.5}$	0.1	0.1	0.1	$10^{-0.5}$	$10^{-0.5}$	1	$10^{-0.5}$
	4 bits	0.1	0.1	0.1	0.1	0.1	0.1	$10^{-1.5}$	$10^{-1.5}$	$10^{-0.5}$	0.1	$10^{-0.5}$	0.1	$10^{-0.5}$	$10^{-0.5}$	1	$10^{-0.5}$
	3 bits	0.1	0.1	0.1	0.1	0.1	0.1	$10^{-1.5}$	$10^{-1.5}$	0.1	0.1	0.1	$10^{-1.5}$	$10^{-0.5}$	0.1	1	$10^{-0.5}$

Appendix B

Supporting Material for CAFE

B.1 Experimental Details

Our experimental setup, including model architectures, best learning rate per scenario, and data distribution, is summarized in Table B.1.

B.1.1 Compression Parameter Details

As reported in Tables 4.2 and 4.3, we select $k = 10\%$, 1% , and 0.1% for Top-k methods. We also choose uniform quantization with 4, 5, and 6 bits for Top-k + Quantization and uniform quantization with 3, 4, and 5 bits for Low-rank approximation + Quantization. For our Quantization experiments, we fix k to 1% for Top-k and rank to 1 for Low-rank approximation. For Top-k + Quantization compression, we aim to test the lower limit of the choice for the number of bits per coordinate. Notice in Tables 4.2 and 4.3 that when the number of bits is 4, Direct compression results in performance collapse on CIFAR-10, whereas **CAFe** maintains robustness. For Low-rank compression, **CAFe** consistently outperforms direct compression by a large margin, regardless of the model architecture and dataset. This is due to Low-rank approximation’s low compression error, even when using rank 1. With Low-rank approximation + Quantization, we also test the lower limit and find that **CAFe** allows for aggressive quantization (down to 3 bits) where Direct compression typically degrades.

B.1.2 Learning Rate Sensitivity

Table B.1 hints that both **CAFe** and direct compression tend to select similar learning rates. In this section, we provide an ablation study on its sensitivity to the learning rate. Theorem 4.7 requires a stricter learning rate condition $\gamma \leq \frac{1-\omega}{L(1+\omega)}$ compared to DCGD’s $\gamma \leq \frac{1}{L}$. In Figure B.1, we plot the accuracy of both methods on CIFAR-10 (iid) as a function of the learning rate γ . Using direct compression or **CAFe**, we find that both methods have similar performance curve shapes across a range of learning rates, albeit **CAFe**’s is always on top. This suggests that the stricter theoretical bound, required for the Lyapunov proof, may not reflect the practical reality, and **CAFe** does not impose an additional hyperparameter tuning burden.

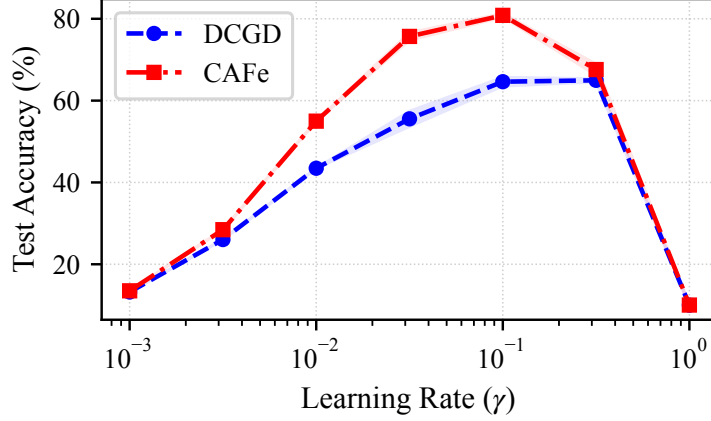


Figure B.1: Learning rate sensitivity analysis on CIFAR-10 (iid) with rank-3 low-rank approximation compression. Both approaches achieve optimal performance around $\gamma = 10^{-1}$ and $\gamma = 10^{-0.5}$ and show similar shapes, suggesting the stricter theoretical bound for CAFe may not be a practical limitation.

B.2 Proofs of Convergence Theorems

We now present the detailed proofs for the convergence theorems stated in Section 4.4, restated for clarity. All three convergence proofs analyze the iterates of the form $x^{k+1} = x^k + \Delta_s^k$. We first establish a common notation. Let the local update be

$$\Delta_n^k = -\gamma \nabla f_n(x^k) \tag{B.1}$$

and the server's candidate update (for CAFe-S) be

$$\Delta_c^k = -\gamma \nabla f_s(x^k). \tag{B.2}$$

The compression error for a client n at step k is e_n^k , and the re-scaled error is $\hat{e}_n^k = e_n^k/\gamma$. The average re-scaled error is $\bar{e}^k = \frac{1}{N} \sum_n \hat{e}_n^k$. Then, the aggregate update Δ_s^k can always be written as:

$$\Delta_s^k = \frac{1}{N} \sum_n (\Delta_n^k + e_n^k) = -\gamma \nabla f(x^k) + \gamma \bar{e}^k.$$

This gives the unified iteration for all algorithms:

$$x^{k+1} = x^k + \Delta_s^k = x^k - \gamma (\nabla f(x^k) - \bar{e}^k). \quad (\text{B.3})$$

We define the effective gradient as $g^k := \nabla f(x^k) - \bar{e}^k$, so the iteration is $x^{k+1} = x^k - \gamma g^k$.

All proofs start from the standard descent lemma, which follows from Assumption 4.4 and a learning rate $\gamma \leq \frac{1}{L}$:

$$\begin{aligned} f(x^{k+1}) &\leq f(x^k) + \langle \nabla f(x^k), x^{k+1} - x^k \rangle + \frac{L}{2} \|x^{k+1} - x^k\|^2 \\ &= f(x^k) - \gamma \langle \nabla f(x^k), g^k \rangle + \frac{L\gamma^2}{2} \|g^k\|^2 \\ &\leq f(x^k) - \frac{\gamma}{2} \|\nabla f(x^k)\|^2 - \left(\frac{\gamma}{2} - \frac{L\gamma^2}{2} \right) \|g^k\|^2 \\ &\quad + \frac{\gamma}{2} \|\nabla f(x^k) - g^k\|^2 \end{aligned} \quad (\text{B.4})$$

$$\leq f(x^k) - \frac{\gamma}{2} \|\nabla f(x^k)\|^2 + \frac{\gamma}{2} \|\bar{e}^k\|^2. \quad (\text{B.5})$$

The final inequality holds for $\gamma \leq \frac{1}{L}$ (which makes the $\|g^k\|^2$ term negative) and uses $\|\nabla f(x^k) - g^k\|^2 = \|\nabla f(x^k) - (\nabla f(x^k) - \bar{e}^k)\|^2 = \|\bar{e}^k\|^2$. The proofs for the three theorems differ only in how they bound the $\mathbb{E} [\|\bar{e}^k\|^2]$ term.

B.2.1 Proof of Theorem 4.6 (DCGD)

THEOREM 4.6. *[DCGD Convergence] Under Assumptions 4.4 and 4.5, with a learning rate $\gamma \leq \frac{1}{L}$ and a compression parameter ω such that $\omega B^2 < 1$, DCGD satisfies*

$$\frac{1}{K} \sum_{k=0}^{K-1} \mathbb{E} [\|\nabla f(x^k)\|^2] \leq \frac{2(f(x^0) - f^*)}{\gamma K} \cdot \frac{1}{1 - \omega B^2}. \quad (4.6)$$

Proof. For standard DCGD (Algorithm 2), clients compress their raw local update Δ_n^k . The error is $e_n^k = \mathcal{C}(\Delta_n^k) - \Delta_n^k$. We bound the expectation of its squared norm using Jensen's

inequality

$$\begin{aligned}
\mathbb{E} [\|\bar{e}^k\|^2] &= \mathbb{E} \left[\left\| \frac{1}{N} \sum_n \hat{e}_n^k \right\|^2 \right] \leq \frac{1}{N} \sum_n \mathbb{E} [\|\hat{e}_n^k\|^2] \\
&= \frac{1}{N\gamma^2} \sum_n \mathbb{E} [\|e_n^k\|^2] \\
&\leq \frac{\omega}{N\gamma^2} \sum_n \mathbb{E} [\|\Delta_n^k\|^2] && \text{(by Equation (4.2))} \\
&= \omega \frac{1}{N} \sum_n \mathbb{E} [\|\nabla f_n(x^k)\|^2] \\
&\leq \omega B^2 \mathbb{E} [\|\nabla f(x^k)\|^2]. && \text{(by Assumption 4.5)}
\end{aligned}$$

Next, we substitute this error bound into our descent lemma (Equation (B.5)) and take the expectation:

$$\begin{aligned}
\mathbb{E} [f(x^{k+1})] &\leq \mathbb{E} \left[f(x^k) - \frac{\gamma}{2} \|\nabla f(x^k)\|^2 + \frac{\gamma}{2} \|\bar{e}^k\|^2 \right] \\
&= \mathbb{E} [f(x^k)] - \frac{\gamma}{2} (1 - \omega B^2) \mathbb{E} [\|\nabla f(x^k)\|^2].
\end{aligned}$$

Re-arranging and telescoping from $k = 0$ to $K - 1$ yields

$$\begin{aligned}
\frac{\gamma}{2} (1 - \omega B^2) \sum_{k=0}^{K-1} \mathbb{E} [\|\nabla f(x^k)\|^2] &\leq f(x^0) - \mathbb{E} [f(x^K)] \\
&\leq f(x^0) - f^*.
\end{aligned}$$

Dividing by K and re-arranging finishes the proof. $\square$

B.2.2 Proof of Theorem 4.7 (CAFe)

The proof for **CAFe** is more complex as the error $\bar{e}^k$ depends on the previous update Δ_s^{k-1} , which itself contains $\bar{e}^{k-1}$. This requires a Lyapunov analysis. We first need three supporting lemmas.

LEMMA B.1. *Given an L -smooth function f , and iterations of the form $x^{k+1} = x^k - \gamma g^k$, we have*

$$-\langle \nabla f(x^{k+1}), g^k \rangle \leq -\langle \nabla f(x^k), g^k \rangle + \gamma L \|g^k\|^2. \quad (\text{B.6})$$

Proof. We have

$$\begin{aligned} -\langle \nabla f(x^{k+1}), g^k \rangle &= -\langle \nabla f(x^k), g^k \rangle \\ &\quad + \langle \nabla f(x^k) - \nabla f(x^{k+1}), g^k \rangle, \end{aligned}$$

and this can be bounded by

$$\begin{aligned} \langle \nabla f(x^{k+1}) - \nabla f(x^k), g^k \rangle &\leq \|\nabla f(x^{k+1}) - \nabla f(x^k)\| \|g^k\| \\ &\leq \gamma L \|g^k\|^2, \end{aligned}$$

where we have used the L -smoothness of f in the last step. Re-arranging, we obtain the desired result. $\square$

LEMMA B.2. *Given Assumptions 4.4 and 4.5, the compression error for DGD + CAFe (with $g^k = \nabla f(x^k) - \bar{e}^k$) satisfies*

$$\begin{aligned} \mathbb{E} \left[\|\bar{e}^{k+1}\|^2 \right] &\leq \omega \mathbb{E} \left[B^2 \|\nabla f(x^{k+1})\|^2 - \|\nabla f(x^k)\|^2 \right] \\ &\quad + \gamma 2\omega L \mathbb{E} \left[\|g^k\|^2 \right] + \omega \mathbb{E} \left[\|\bar{e}^k\|^2 \right]. \end{aligned} \quad (\text{B.7})$$

Proof. We start by bounding the next-step error:

$$\begin{aligned} \mathbb{E} \left[\|\bar{e}^{k+1}\|^2 \right] &\leq \frac{1}{N} \sum_n \mathbb{E} \left[\|\hat{e}_n^{k+1}\|^2 \right] && (\text{by Jensen's}) \\ &= \frac{1}{N\gamma^2} \sum_n \mathbb{E} \left[\|e_n^{k+1}\|^2 \right] \end{aligned}$$

In **CAFe**, clients compress $(\Delta_n^{k+1} - \Delta_s^k)$. So, by Equation (4.2), the last is upper bounded by

$$\begin{aligned}
& \frac{\omega}{N\gamma^2} \sum_n \mathbb{E} \left[\|\Delta_n^{k+1} - \Delta_s^k\|^2 \right] \\
&= \frac{\omega}{N\gamma^2} \sum_n \mathbb{E} \left[\|\gamma \nabla f_n(x^{k+1}) - \gamma g^k\|^2 \right] \\
&= \frac{\omega}{N} \sum_n \mathbb{E} \left[\|\nabla f_n(x^{k+1}) - g^k\|^2 \right].
\end{aligned}$$

We add and subtract $\nabla f(x^{k+1})$ inside the norm, and expand. Note that the cross-terms $\langle \nabla f_n - \nabla f, \nabla f - g^k \rangle$ sum over n is zero, so the last expression equals

$$\begin{aligned}
& \omega \mathbb{E} \left[\frac{1}{N} \sum_n \|\nabla f_n(x^{k+1}) - \nabla f(x^{k+1})\|^2 \right] \\
& \quad + \omega \mathbb{E} \left[\|\nabla f(x^{k+1}) - g^k\|^2 \right]
\end{aligned} \tag{B.8}$$

Using Assumption 4.5 we can bound the first term in Equation (B.8) as

$$\begin{aligned}
\frac{1}{N} \sum_n \|\nabla f_n - \nabla f\|^2 &= \left(\frac{1}{N} \sum_n \|\nabla f_n\|^2 \right) - \|\nabla f\|^2 \\
&\leq (B^2 - 1) \|\nabla f\|^2.
\end{aligned}$$

Thus, to finish we expand the last term in Equation (B.8),

$$\begin{aligned}
\|\nabla f(x^{k+1}) - g^k\|^2 &= \|\nabla f(x^{k+1})\|^2 + \|g^k\|^2 \\
&\quad - 2 \langle \nabla f(x^{k+1}), g^k \rangle,
\end{aligned}$$

and apply Lemma B.1 to bound the inner product term. This gives us

$$\begin{aligned}\|\nabla f(x^{k+1}) - g^k\|^2 &\leq \left(\|\nabla f(x^{k+1})\|^2 - \|\nabla f(x^k)\|^2\right) \\ &\quad + \left(\|\nabla f(x^k)\|^2 - 2\langle \nabla f(x^k), g^k \rangle + \|g^k\|^2\right) \\ &\quad + 2\gamma L \|g^k\|^2\end{aligned}$$

The middle term is $\|\nabla f(x^k) - g^k\|^2 = \|\bar{e}^k\|^2$, so plugging this back into our main inequality gives the lemma. $\square$

LEMMA B.3. *Let $f: \mathbb{R}^d \rightarrow \mathbb{R}$ be an L -smooth function with a lower bound f^* . Then, for any $x \in \mathbb{R}^d$,*

$$\|\nabla f(x)\|^2 \leq 2L(f(x) - f^*).$$

Proof. By Assumption 4.4, for any y , we have

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

We choose $y = x - \frac{1}{L} \nabla f(x)$, and obtain

$$f(y) \leq f(x) - \frac{1}{2L} \|\nabla f(x)\|^2.$$

Since $f(y) \geq f^*$, we re-arrange and obtain the result. $\square$

THEOREM 4.7. *[CAFe + DGD Convergence] Under Assumptions 4.4 and 4.5, with a learning rate satisfying Equation (4.7) and $\omega B^2 < 1$, CAFe with DGD satisfies*

$$\frac{1}{K} \sum_{k=0}^{K-1} \mathbb{E} \left[\|\nabla f(x^k)\|^2 \right] \leq \frac{2(f(x^0) - f^*)}{\gamma K} \cdot \frac{1 - \omega}{1 - \omega B^2}. \quad (4.8)$$

Proof. First, we define a Lyapunov function to capture both the function value and the compression error. Let

$$\Psi^k := \mathbb{E} \left[f(x^k) + \frac{\gamma}{2(1-\omega)} \|\bar{e}^k\|^2 \right]. \quad (\text{B.9})$$

Our goal is to show that $\Psi^{k+1} \leq \Psi^k - (\text{progress term})$. We will combine the descent lemma (Equation (B.5)) and the compression error bound from Lemma B.2. We start with the expectation of the descent lemma (Equation (B.4))

$$\begin{aligned} \mathbb{E} [f(x^{k+1})] &\leq \mathbb{E} [f(x^k)] - \frac{\gamma}{2} \mathbb{E} [\|\nabla f(x^k)\|^2] \\ &\quad + \frac{\gamma}{2} \mathbb{E} [\|\bar{e}^k\|^2] \\ &\quad - \left(\frac{1}{2\gamma} - \frac{L}{2} \right) \mathbb{E} [\|x^{k+1} - x^k\|^2] \end{aligned}$$

and $\frac{\gamma}{2(1-\omega)} \cdot \mathbb{E} [\text{Lemma B.2}]$, which yields

$$\begin{aligned} \frac{\gamma \mathbb{E} [\|\bar{e}^{k+1}\|^2]}{2(1-\omega)} &\leq \frac{\gamma \omega \mathbb{E} [\|\bar{e}^k\|^2]}{2(1-\omega)} \\ &\quad + \frac{\gamma \omega}{2(1-\omega)} \mathbb{E} [B^2 \|\nabla f^{k+1}\|^2 - \|\nabla f^k\|^2] \\ &\quad + \frac{\gamma^2 \omega L}{1-\omega} \mathbb{E} [\|g^k\|^2]. \end{aligned}$$

Summing these two inequalities gives

$$\begin{aligned}
\Psi^{k+1} &\leq \underbrace{\left(\mathbb{E} [f(x^k)] + \frac{\gamma\omega\mathbb{E} [\|\bar{e}^k\|^2]}{2(1-\omega)} + \frac{\gamma}{2}\mathbb{E} [\|\bar{e}^k\|^2] \right)}_{=\Psi^k} \\
&\quad + \underbrace{\left(-\frac{\gamma}{2}\mathbb{E} [\|\nabla f^k\|^2] - \frac{\gamma\omega}{2(1-\omega)}\mathbb{E} [\|\nabla f^k\|^2] \right)}_{=-\frac{\gamma}{2(1-\omega)}\mathbb{E} [\|\nabla f^k\|^2]} \\
&\quad - \underbrace{\left(\frac{1}{2\gamma} - \frac{L}{2} - \frac{\gamma\omega L}{1-\omega} \right) \mathbb{E} [\|x^{k+1} - x^k\|^2]}_{\leq 0 \text{ by Equation (4.7)}} \\
&\quad + \frac{\gamma\omega B^2}{2(1-\omega)} \mathbb{E} [\|\nabla f(x^{k+1})\|^2].
\end{aligned}$$

The Ψ^k term simplifies because $\frac{\gamma\omega}{2(1-\omega)} + \frac{\gamma}{2} = \frac{\gamma}{2(1-\omega)}$. The $\|\nabla f^k\|^2$ term simplifies to $-\frac{\gamma(1-\omega)+\gamma\omega}{2(1-\omega)} = -\frac{\gamma}{2(1-\omega)}$. The $\|x^{k+1} - x^k\|^2$ term is negative due to Equation (4.7), so we can drop it. Thus,

$$\begin{aligned}
\Psi^{k+1} &\leq \Psi^k - \frac{\gamma}{2(1-\omega)} \mathbb{E} [\|\nabla f(x^k)\|^2] \\
&\quad + \frac{\gamma\omega B^2}{2(1-\omega)} \mathbb{E} [\|\nabla f(x^{k+1})\|^2].
\end{aligned}$$

Given that $\omega B^2 < 1$, this almost gives us our desired Lyapunov decrease. We have taken the liberty of calling Ψ^k the Lyapunov function, but it does not strictly decrease due to the last term, which depends on the next iterate x^{k+1} , instead of x^k . We will handle the extra term by first telescoping the recursion from $k = 0$ to $K - 1$,

$$\begin{aligned}
\Psi^K &\leq \Psi^0 - \frac{\gamma}{2(1-\omega)} \sum_{k=0}^{K-1} \mathbb{E} [\|\nabla f(x^k)\|^2] \\
&\quad + \frac{\gamma\omega B^2}{2(1-\omega)} \sum_{k=0}^{K-1} \mathbb{E} [\|\nabla f(x^{k+1})\|^2].
\end{aligned}$$

Given $\Psi^0 = f(x^0)$, since $\bar{e}^0 = 0$, and $\Psi^K \geq f(x^K)$, we obtain

$$\begin{aligned} f(x^K) &\leq f(x^0) - \frac{\gamma(1 - \omega B^2)}{2(1 - \omega)} \sum_{k=0}^{K-1} \mathbb{E} \left[\|\nabla f(x^k)\|^2 \right] \\ &\quad + \frac{\gamma\omega B^2}{2(1 - \omega)} \mathbb{E} \left[\|\nabla f(x^K)\|^2 \right], \end{aligned}$$

where we have dropped a negative $\|\nabla f(x^0)\|^2$ term. Next, we use Lemma B.3 to bound the $\mathbb{E} \left[\|\nabla f(x^K)\|^2 \right]$ term,

$$\begin{aligned} \frac{\gamma(1 - \omega B^2)}{2(1 - \omega)} \sum_{k=0}^{K-1} \mathbb{E} \left[\|\nabla f(x^k)\|^2 \right] &\leq \\ f(x^0) + \frac{\gamma\omega B^2}{2(1 - \omega)} 2L (f(x^K) - f^*) - f(x^K). \end{aligned}$$

To obtain the theorem statement, it suffices to show that

$$\frac{\gamma\omega B^2}{2(1 - \omega)} 2L (f(x^K) - f^*) - f(x^K) \leq -f^*, \quad (\text{B.10})$$

which holds if

$$\frac{\gamma\omega B^2}{1 - \omega} L < 1. \quad (\text{B.11})$$

Notice that this is equivalent to Equation (4.7), given that $\omega B^2 < 1$. Thus, we have completed the proof. $\square$

B.2.3 Proof of Theorem 4.9 (CAFe-S)

THEOREM 4.9. *[CAFe-S + DGD Convergence] Under Assumptions 4.4, 4.5 and 4.8, with a learning rate $\gamma \leq \frac{1}{L}$, and a compression parameter ω such that $\omega G^2 B^2 < 1$, CAFe-S with DGD*

satisfies

$$\frac{1}{K} \sum_{k=0}^{K-1} \mathbb{E} \left[\|\nabla f(x^k)\|^2 \right] \leq \frac{2(f(x^0) - f^*)}{\gamma K(1 - \omega G^2 B^2)}. \quad (4.10)$$

Proof. This proof follows the same structure as the DCGD proof. For **CAFe-S**, clients compress the difference $\Delta_n^k - \Delta_c^k$. The error is $e_n^k = \mathcal{C}(\Delta_n^k - \Delta_c^k) - (\Delta_n^k - \Delta_c^k)$.

$$\begin{aligned} \mathbb{E} \left[\|\bar{e}^k\|^2 \right] &\leq \frac{1}{N} \sum_n \mathbb{E} \left[\|\hat{e}_n^k\|^2 \right] && \text{(by Jensen's)} \\ &= \frac{1}{N\gamma^2} \sum_n \mathbb{E} \left[\|e_n^k\|^2 \right] \\ &\leq \frac{\omega}{N\gamma^2} \sum_n \mathbb{E} \left[\|\Delta_n^k - \Delta_c^k\|^2 \right] && \text{(by Equation (4.2))} \\ &= \omega \frac{1}{N} \sum_n \mathbb{E} \left[\|\nabla f_n(x^k) - \nabla f_s(x^k)\|^2 \right] \\ &\leq \omega G^2 \frac{1}{N} \sum_n \mathbb{E} \left[\|\nabla f_n(x^k)\|^2 \right] && \text{(by Assumption 4.8)} \\ &\leq \omega G^2 B^2 \mathbb{E} \left[\|\nabla f(x^k)\|^2 \right]. && \text{(by Assumption 4.5)} \end{aligned}$$

Now, notice that this error bound has the *exact same form* as the DCGD error bound, with the constant ωB^2 simply replaced by $\omega G^2 B^2$. The remainder of the proof (telescoping the sum and re-arranging) is identical to the proof for Theorem 4.6. $\square$

Appendix C

Supporting Material for Chapter 6

C.1 Appendix Roadmap and Notation

The appendices collect the inequalities, coin-betting reductions, compressed-tracking estimates, and potential-specific tail calculations used in Section 6.6. They are organized in the same order as the main proof: scalar and Hilbert-space coin betting, shared compressed-gossip recurrences, expected compressed network dynamics, the regret decomposition, the KT and shifted-exponential specializations, and the final prediction-disagreement details. We also include a pathwise regret guarantee in Appendix C.8, which is not needed for the expected-regret theorem but may be of independent interest. For the closed-form corollaries, we use the following quadratic-exponential envelope:

$$\Phi_{\tau,\kappa}(\theta) = \theta \sqrt{\tau \log \left(1 + \frac{\tau \theta^2}{\kappa^2} \right)} - \kappa. \tag{C.1}$$

Table C.1: Frequently used notation.

Symbol	Definition
$N, T, \mathcal{H}$	Number of agents, horizon, real Hilbert space
W, ρ, μ	Symmetric gossip matrix, zero-mean contraction factor ρ , spectral gap $\mu = 1 - \rho$
$\ell_{n,t}, g_{n,t}$	Local convex loss and subgradient
$\bar{\ell}_t, R_T^{\text{net}}$	Network loss (6.2) and network regret (6.1)
F_t, ε	Coin-betting potential and initial endowment
$\beta_t, h_t, \text{Bet}_t$	Betting fraction, scalar bet, and lifted Hilbert-space prediction
$\mathcal{C}, \alpha^{\text{stoch}}$	Compressor and mean-square residual coefficient
$G_{n,t}, \hat{G}_{n,t}$	Clean state and compressed tracker at agent n
$J, P_\perp, \ \cdot\ _{2,\mathcal{H}}$	Consensus projection, centered projection $P_\perp = I - J$, and product network norm $\ Z\ _{2,\mathcal{H}} = (\sum_n \ Z_n\ ^2)^{1/2}$
$q(t), Q(s, t)$	Number of compressed gossip substeps and $Q(s, t) = \sum_{r=s}^{t-1} q(r)$
$\gamma, \chi, \eta, \bar{\eta}$	Consensus step size, Lyapunov weight $\chi = \mu/4$, generic contraction rate η , and expected rate $\bar{\eta}$
$F_T^*, \Phi_{\tau,\kappa}$	Restricted Fenchel conjugate (6.15) and quadratic-exponential envelope (C.1)
$\Psi_T^{\text{KT}}, \Psi_T^{\text{SE}}$	KT and shifted-exponential envelope specializations

For the two concrete potentials, set

$$\Psi_T^{\text{KT}}(\theta) = \Phi_{2(T+1), \varepsilon/(e\sqrt{\pi T})}(\theta),$$

$$\Psi_T^{\text{SE}}(\theta) = \Phi_{T+1, \varepsilon/\sqrt{T+1}}(\theta).$$

We will need the following facts.

LEMMA C.1 (Subgradient Regret Inequality). *If $g \in \partial f(x)$, then for every comparator u ,*

$$f(x) - f(u) \leq \langle g, x - u \rangle.$$

Proof. By the definition of subgradient, for every y ,

$$f(y) \geq f(x) + \langle g, y - x \rangle.$$

Substitute $y = u$ and rearrange:

$$f(x) - f(u) \leq -\langle g, u - x \rangle = \langle g, x - u \rangle.$$

□

LEMMA C.2 (Subgradient Norm Bound). *If f is 1-Lipschitz and $g \in \partial f(x)$, then $\|g\| \leq 1$.*

Proof. Suppose, toward a contradiction, that $\|g\| > 1$. Let $\tau = 1/\|g\| > 0$, so that $\|\tau g\| = 1$. Lipschitzness gives

$$f(x + \tau g) - f(x) \leq \|\tau g\| = 1.$$

Using the subgradient inequality at $y = x + \tau g$ gives

$$f(x + \tau g) - f(x) \geq \langle g, \tau g \rangle = \tau \|g\|^2 = \|g\| > 1,$$

a contradiction. □

C.2 Coin-Betting Reductions

Let $P = \{F_t\}$ be a coin-betting potential and $h_t(x) = \beta_t(x)F_{t-1}(x)$. For a scalar coin sequence $c_t \in [-1, 1]$, define $S_t = \sum_{s=0}^{t-1} c_s$.

LEMMA C.3 (Betting Fraction Is Bounded). *Whenever $F_t(x+1) > 0$ and $F_t(x-1) > 0$,*

$$|\beta_t(x)| \leq 1.$$

Proof. Set $a = F_t(x + 1)$ and $b = F_t(x - 1)$. Since $a, b > 0$,

$$|\beta_t(x)| = \left| \frac{a - b}{a + b} \right| = \frac{|a - b|}{a + b} \leq \frac{a + b}{a + b} = 1.$$

□

LEMMA C.4 (Wealth Lower-Bounds the Potential). *For any $T \geq 0$,*

$$\varepsilon \prod_{t=0}^{T-1} (1 + c_t \beta_{t+1}(S_t)) \geq F_T(S_T).$$

Proof. We argue by induction on T . For $T = 0$, $S_0 = 0$ and the claim is $F_0(0) = \varepsilon$. Assume the claim at time T . Since $|c_T| \leq 1$ and $|S_T| \leq T$ (by the triangle inequality), the potential inequality with time $T + 1$, state S_T , and coin c_T gives

$$\begin{aligned} (1 + c_T \beta_{T+1}(S_T)) F_T(S_T) \\ \geq F_{T+1}(S_T + c_T) = F_{T+1}(S_{T+1}). \end{aligned}$$

The factor $1 + c_T \beta_{T+1}(S_T)$ is nonnegative by Lemma C.3. Multiplying the induction hypothesis by this factor gives the claim at $T + 1$. □

LEMMA C.5 (Reward Telescoping). *For every $T \geq 0$,*

$$\sum_{t=0}^{T-1} c_t h_{t+1}(S_t) \geq F_T(S_T) - \varepsilon. \tag{C.2}$$

Proof. The defining potential inequality implies the one-step reward bound

$$c_t h_{t+1}(S_t) \geq F_{t+1}(S_t + c_t) - F_t(S_t) = F_{t+1}(S_{t+1}) - F_t(S_t).$$

Indeed, expanding h_{t+1} in the inequality

$$(1 + c_t \beta_{t+1}(S_t)) F_t(S_t) \geq F_{t+1}(S_t + c_t)$$

and moving $F_t(S_t)$ to the right gives exactly the one-step bound. Summing from $t = 0$ to $T - 1$ telescopes:

$$\begin{aligned} & \sum_{t=0}^{T-1} (F_{t+1}(S_{t+1}) - F_t(S_t)) \\ &= F_T(S_T) - F_0(S_0) = F_T(S_T) - \varepsilon. \end{aligned}$$

□

LEMMA C.6 (Quadratic-Exponential Fenchel Envelope). *Let $\tau, \kappa > 0$ and $x, y \geq 0$. Then,*

$$xy - \kappa \exp\left(\frac{x^2}{2\tau}\right) \leq y \sqrt{\tau \log\left(1 + \frac{\tau y^2}{\kappa^2}\right)} - \kappa.$$

Proof. Set

$$a = \frac{y\sqrt{\tau}}{\kappa}, \quad v = \frac{x}{\sqrt{\tau}}, \quad r = \sqrt{\log(1 + a^2)}.$$

It suffices to prove

$$av - \exp(v^2/2) \leq ar - 1.$$

If $v \leq r$, then $av \leq ar$ and $\exp(v^2/2) \geq 1$. If $v > r$, write $d = v - r \geq 0$. The inequality $q \log q \geq q - 1$ for $q \geq 1$, applied to $q = 1 + a^2$, gives $a \leq r \exp(r^2/2)$. Together with $\exp z \geq 1 + z$ and $r^2/2 + rd \leq v^2/2$, this yields $\exp(v^2/2) \geq 1 + ad$, which is equivalent to the desired bound. □

LEMMA C.7 (Concrete Terminal Conjugate Envelopes). *For every integer $T \geq 1$ and every $\theta \geq 0$, the KT and shifted-exponential terminal conjugates satisfy*

$$(F_T^{\text{KT}})^*(\theta) \leq \Psi_T^{\text{KT}}(\theta), \quad (F_T^{\text{SE}})^*(\theta) \leq \Psi_T^{\text{SE}}(\theta).$$

Proof. For the shifted-exponential potential, the bound follows directly from Lemma C.6 with $\tau = T + 1$ and $\kappa = \varepsilon/\sqrt{T + 1}$.

For the KT potential, one can show using standard Gamma properties, that on $0 \leq s \leq T$,

$$F_T^{\text{KT}}(s) \geq \frac{\varepsilon}{e\sqrt{\pi T}} \exp\left(\frac{s^2}{4(T + 1)}\right).$$

Applying Lemma C.6 with $\tau = 2(T + 1)$ and $\kappa = \varepsilon/(e\sqrt{\pi T})$ gives the KT envelope. $\square$

THEOREM C.8 (One-Dimensional Coin-Betting Regret). *Suppose the restricted conjugate (6.15) is finite. Then, for every $\theta \geq 0$ and every $T \geq 1$,*

$$\theta S_T - \sum_{t=0}^{T-1} c_t h_{t+1}(S_t) \leq F_T^*(\theta) + \varepsilon. \tag{C.3}$$

Proof. By Lemma C.5,

$$-\sum_{t=0}^{T-1} c_t h_{t+1}(S_t) \leq -F_T(S_T) + \varepsilon.$$

Since $|S_T| \leq T$ and the potential is even,

$$\theta S_T - F_T(S_T) \leq \theta |S_T| - F_T(|S_T|) \leq F_T^*(\theta).$$

Combining this with the telescoping inequality proves the claim. $\square$

The following is the only place where excellence enters the proof. Once this radial bound is

available, the decentralized and scalar reductions use only the ordinary coin-betting potential and the stated Lipschitz estimates.

THEOREM C.9 (Radial Hilbert-Space Coin-Betting Regret). *Let $\mathcal{H}$ be a real Hilbert space and let $c_t \in \mathcal{H}$ satisfy $\|c_t\| \leq 1$. Put $S_t = \sum_{s=0}^{t-1} c_s$ and predict $y_t = \text{Bet}_{t+1}(S_t)$ using (6.10). If P is excellent in the square-root sense and has finite restricted conjugates, then for every $u \in \mathcal{H}$ and every integer $T \geq 1$,*

$$\sum_{t=0}^{T-1} (\langle u, c_t \rangle - \langle c_t, y_t \rangle) \leq F_T^*(\|u\|) + \varepsilon.$$

Proof. The lifted betting map is radial:

$$\text{Bet}_t(G) = \frac{h_t(\|G\|)}{\|G\|} G \quad (G \neq 0).$$

The square-root convexity condition is used only to compare squared norms. When $S_t \neq 0$, set $r = \|S_t\|$ and $\zeta = \langle c_t, S_t \rangle / r$. Then, $|\zeta| \leq 1$, and

$$\|S_t + c_t\|^2 \leq \frac{1 + \zeta}{2} (r + 1)^2 + \frac{1 - \zeta}{2} (r - 1)^2.$$

Convexity of $z \mapsto F_{t+1}(\sqrt{z})$ and the scalar coin-betting inequality at r therefore give the one-step reward inequality

$$\langle c_t, \text{Bet}_{t+1}(S_t) \rangle \geq F_{t+1}(\|S_t + c_t\|) - F_t(\|S_t\|).$$

The case $S_t = 0$ follows directly from the same scalar inequality and evenness of the potential. Summing over t telescopes results in a cumulative reward that is at least $F_T(\|S_T\|) - \varepsilon$. Since $\|S_T\| \leq T$, the same Fenchel-conjugate argument as in the scalar theorem, now with $\theta = \|u\|$ and $\langle u, S_T \rangle \leq \|u\| \|S_T\|$, yields the bound. $\square$

C.3 Network Dynamics

We next record the deterministic tracking estimates used by DECO-EF.

LEMMA C.10 (Average State Tracks the Average Cumulative Gradient). *For every run and every t ,*

$$\frac{1}{N} \sum_{n=1}^N G_{n,t} = \sum_{s=0}^{t-1} \frac{1}{N} \sum_{n=1}^N (-g_{n,s}).$$

Proof. At $t = 0$ both sides are zero. Suppose the identity holds at time t . The local movement forms $G_t + \delta_t$, where $\delta_{n,t} = -g_{n,t}$. Each compressed gossip substep changes G by

$$\gamma \sum_j W_{nj} (\hat{Z}_j - \hat{Z}_n).$$

Summing this correction over n gives zero by the row and column stochasticity of W . Hence, every compressed gossip substep preserves the network average of G . Thus, the average at time $t + 1$ is the previous average plus $\frac{1}{N} \sum_n (-g_{n,t})$, proving the induction step. $\square$

LEMMA C.11 (Lyapunov Tail Unrolling). *Let $q : \mathbb{N} \rightarrow \mathbb{N}$, $P \geq 0$, and $\eta \geq 0$. Define $Q(s, t) = \sum_{r=s}^{t-1} q(r)$. If $a_0 = 0$ and*

$$a_{t+1} \leq \eta^{q(t)} (a_t + P)$$

for every t , then

$$a_t \leq P \sum_{s < t} \eta^{Q(s,t)}$$

for every t .

Proof. The claim is immediate at $t = 0$. For the induction step, use $Q(s, t+1) = Q(s, t) + q(t)$

for $s < t$ and $Q(t, t+1) = q(t)$:

$$\begin{aligned} a_{t+1} &\leq \eta^{q(t)} \left(P \sum_{s < t} \eta^{Q(s,t)} + P \right) \\ &= P \sum_{s < t} \eta^{Q(s,t+1)} + P \eta^{Q(t,t+1)} = P \sum_{s < t+1} \eta^{Q(s,t+1)}. \end{aligned}$$

□

LEMMA C.12 (Raw Compressed-Gossip Recurrences). *Let W be a valid gossip matrix for a connected communication graph, and let $L = I - W$. Suppose the Hilbert lift of W contracts the zero-mean subspace by $0 \leq \rho < 1$, and put $\mu = 1 - \rho$. Fix one compressed gossip substep with $0 < \gamma \leq 1$. Define*

$$e^k = Z^k - \widehat{Z}^k, \quad m^k = \mathcal{C}(\omega^k, e^k), \quad r^k = e^k - m^k,$$

and let

$$D^k = \|P_{\perp} Z^k\|_{2,\mathcal{H}}, \quad E^k = \|e^k\|_{2,\mathcal{H}}.$$

Then the inner-loop update satisfies

$$D^{k+1} \leq (1 - \gamma\mu)D^k + 2\gamma \|r^k\|_{2,\mathcal{H}}, \tag{C.4}$$

$$E^{k+1} \leq 2\gamma D^k + (1 + 2\gamma) \|r^k\|_{2,\mathcal{H}}. \tag{C.5}$$

Proof. Because W is doubly stochastic, $LJ = 0$, so $LZ = LP_{\perp}Z$. The product- ℓ_2 nonexpansiveness of W gives $\|LU\|_{2,\mathcal{H}} \leq 2\|U\|_{2,\mathcal{H}}$ for every network vector U . The tracker update gives

$$\widehat{Z}^{k+1} = \widehat{Z}^k + m^k = Z^k - r^k.$$

The clean-state update is

$$Z^{k+1} = Z^k - \gamma L \widehat{Z}^{k+1} = (I - \gamma L) Z^k + \gamma L r^k.$$

Using $I - \gamma L = (1 - \gamma)I + \gamma W$, the zero-mean contraction for $0 < \gamma \leq 1$ gives

$$\|(I - \gamma L)v\|_{2,\mathcal{H}} \leq (1 - \gamma + \gamma\rho) \|v\|_{2,\mathcal{H}} = (1 - \gamma\mu) \|v\|_{2,\mathcal{H}}$$

whenever v is mean zero. Since $\|Lr^k\|_{2,\mathcal{H}} \leq 2\|r^k\|_{2,\mathcal{H}}$, this yields Equation (C.4). Also,

$$Z^{k+1} - \widehat{Z}^{k+1} = r^k - \gamma L(Z^k - r^k).$$

Using $LZ^k = LP_\perp Z^k$ and $\|LU\|_{2,\mathcal{H}} \leq 2\|U\|_{2,\mathcal{H}}$ gives Equation (C.5). $\square$

THEOREM C.13 (Expected Compressed Tracker Contraction). *Let W be a valid gossip matrix for a connected communication graph. Let $0 \leq \rho < 1$ be the resulting product- ℓ_2 zero-mean contraction factor. Its Hilbert lift satisfies*

$$\|WZ\|_{2,\mathcal{H}} \leq \|Z\|_{2,\mathcal{H}},$$

$$\|WZ\|_{2,\mathcal{H}} \leq \rho \|Z\|_{2,\mathcal{H}} \quad \text{for zero-mean } Z, \quad 0 \leq \rho < 1.$$

Put

$$\mu = 1 - \rho, \quad \bar{\sigma} = \sqrt{\alpha^{\text{stoch}}}.$$

Assume

$$0 < \gamma \leq \min \left\{ 1, \frac{1 - \bar{\sigma}}{2\bar{\sigma} \left(1 + \frac{4}{\mu}\right) + \frac{\mu}{2}} \right\}.$$

Let

$$\chi = \frac{\mu}{4}, \quad \bar{\eta} = 1 - \frac{\gamma\mu}{2}.$$

For one compressed gossip substep, use D^k and E^k from Equation (6.24) and define

$$V^k = D^k + \chi E^k. \tag{C.6}$$

Suppose the compressor randomness is fresh conditionally on the current payloads, so that

$$\mathbf{E} \left[\|r^k\|_{2,\mathcal{H}} \mid Z^k, \widehat{Z}^k \right] \leq \sqrt{\alpha^{\text{stoch}}} \|e^k\|_{2,\mathcal{H}}.$$

Then one step satisfies

$$\mathbf{E} \left[V^{k+1} \mid Z^k, \widehat{Z}^k \right] \leq \bar{\eta} V^k.$$

Proof. Apply the raw recurrences in Lemma C.12 to the current inner-loop substep. The mean-square compressor condition gives the stated vector residual estimate. Indeed, conditionally on e^k ,

$$\mathbf{E} \left[\|r^k\|_{2,\mathcal{H}}^2 \mid Z^k, \widehat{Z}^k \right] = \sum_n \mathbf{E} \|r_n^k\|^2 \leq \alpha^{\text{stoch}} \|e^k\|_{2,\mathcal{H}}^2,$$

and Jensen's inequality gives

$$\mathbf{E} \left[\|r^k\|_{2,\mathcal{H}} \mid Z^k, \widehat{Z}^k \right] \leq \bar{\sigma} E^k.$$

Taking conditional expectations in Equations (C.4) and (C.5) gives

$$\mathbf{E}[D^{k+1} \mid Z^k, \widehat{Z}^k] \leq (1 - \gamma\mu)D^k + 2\gamma\bar{\sigma}E^k, \quad (\text{C.7})$$

$$\mathbf{E}[E^{k+1} \mid Z^k, \widehat{Z}^k] \leq 2\gamma D^k + \bar{\sigma}(1 + 2\gamma)E^k. \quad (\text{C.8})$$

Combining these recurrences with $V^k = D^k + \chi E^k$,

$$\begin{aligned} \mathbf{E}[V^{k+1} \mid Z^k, \widehat{Z}^k] &\leq (1 - \gamma\mu + 2\chi\gamma) D^k \\ &\quad + (2\gamma\bar{\sigma} + \chi\bar{\sigma}(1 + 2\gamma)) E^k. \end{aligned}$$

By the choice $\chi = \mu/4$, the D^k coefficient is $1 - \gamma\mu/2 = \bar{\eta}$. The imposed condition on γ is exactly the condition that

$$2\gamma\bar{\sigma} + \chi\bar{\sigma}(1 + 2\gamma) \leq \chi\bar{\eta}.$$

After dividing by χ , this requirement is equivalent to

$$\gamma \left[2\bar{\sigma} \left(1 + \frac{4}{\mu} \right) + \frac{\mu}{2} \right] \leq 1 - \bar{\sigma}.$$

Hence

$$\mathbf{E}[V^{k+1} \mid Z^k, \widehat{Z}^k] \leq \bar{\eta}V^k, \quad 0 < \bar{\eta} < 1. \quad (\text{C.9})$$

□

LEMMA C.14 (Expected One-Round Lyapunov Recurrence). *Under the hypotheses of Theorem C.13, let $\mathcal{F}_t$ be the history before the compressor randomness in round t , and let V_t be*

the outer-round Lyapunov quantity. Then,

$$\mathbf{E}[V_{t+1} \mid \mathcal{F}_t] \leq \bar{\eta}^{q(t)} \left(V_t + (2 + \chi)\sqrt{N} \right).$$

Proof. At the start of round t ,

$$Z_t^0 = G_t + \delta_t, \quad \widehat{Z}_t^0 = \widehat{G}_t, \quad \delta_{n,t} = -g_{n,t}.$$

Because every loss is 1-Lipschitz, $\|\delta_t\|_{2,\mathcal{H}} \leq \sqrt{N}$. The disagreement part of the Lyapunov function increases by at most $2\sqrt{N}$, while the tracking part increases by at most $\sqrt{N}$. Hence, after the local movement,

$$V_t^0 \leq V_t + (2 + \chi)\sqrt{N}.$$

After this movement, apply Theorem C.13 to each inner gossip substep. The freshness assumption makes the next compressor seed conditionally independent of the past given the current payload, so the tower property iterates the conditional contraction $q(t)$ times. $\square$

LEMMA C.15 (Expected Compressed State-Disagreement Unrolling). *Under the expected hypotheses, with $Q(s, t) = \sum_{r=s}^{t-1} q(r)$,*

$$\mathbf{E} \frac{1}{N} \sum_{n=1}^N \|G_{n,t} - \bar{G}_t\| \leq (2 + \chi)\sqrt{N} \sum_{s < t} \bar{\eta}^{Q(s,t)}.$$

Proof. Let $a_t = \mathbf{E} V_t$ and $P = (2 + \chi)\sqrt{N}$. Taking expectation in Lemma C.14 gives

$$a_{t+1} \leq \bar{\eta}^{q(t)}(a_t + P), \quad a_0 = 0.$$

The unrolling in Lemma C.11 applies to this scalar recurrence. Finally,

$$\frac{1}{N} \sum_n \|G_{n,t} - \bar{G}_t\| \leq \|P_\perp G_t\|_{2,\mathcal{H}} \leq V_t$$

for every realization, so the same bound holds after expectation. $\square$

C.4 Disagreement and Network Regret

The regret proof separates the centralized radial coin-betting term from the disagreement induced by nonidentical agent predictions. The regret assembly uses the global-loss comparison in Lemma C.27 directly. The following summation lemma is the shared tail estimate used by both potentials.

LEMMA C.16 (Square-Root Disagreement Criterion). *Let $L(t)$ be a bound on the prediction-map Lipschitz constant at time t , and define*

$$Q(s, t) = \sum_{r=s}^{t-1} q(r).$$

If, for all $t \geq 1$,

$$L(t) \sum_{s < t} \eta^{Q(s,t)} \leq \frac{C_{\text{lip}}}{\sqrt{t}},$$

then

$$2\sqrt{N} \sum_{t=1}^T L(t) \sum_{s < t} \eta^{Q(s,t)} \leq 4C_{\text{lip}} \sqrt{N} \sqrt{T}.$$

Proof. Apply the assumed pointwise bound term by term:

$$2\sqrt{N} \sum_{t=1}^T L(t) \sum_{s < t} \eta^{Q(s,t)} \leq 2C_{\text{lip}} \sqrt{N} \sum_{t=1}^T \frac{1}{\sqrt{t}}.$$

The integral comparison

$$\sum_{t=1}^T \frac{1}{\sqrt{t}} \leq 1 + \int_1^T x^{-1/2} dx \leq 2\sqrt{T}$$

gives the result. □

THEOREM C.17 (General Network Regret Template). *Let*

$$\bar{g}_t = \frac{1}{N} \sum_{n=1}^N g_{n,t}, \quad \bar{G}_t = \frac{1}{N} \sum_{n=1}^N G_{n,t}, \quad y_t = \text{Bet}_{t+1}(\bar{G}_t).$$

Suppose the centralized radial coin-betting term satisfies

$$\sum_{t=0}^{T-1} \langle \bar{g}_t, y_t - u \rangle \leq F_T^*(\|u\|) + \varepsilon$$

for every u and every integer $T \geq 1$. Suppose also that

$$\sum_{t=0}^{T-1} \frac{1}{N} \sum_{n=1}^N \|x_{n,t} - y_t\| \leq C_{\text{net}} \sqrt{T}.$$

Then, for every $u \in \mathcal{H}$ and every integer $T \geq 1$,

$$R_T^{\text{net}}(u) \leq F_T^*(\|u\|) + \varepsilon + 3C_{\text{net}} \sqrt{T}.$$

Proof. The Hilbert-space global-loss comparison in Lemma C.27 gives, round by round,

$$\frac{1}{N} \sum_{n=1}^N \bar{\ell}_t(x_{n,t}) - \bar{\ell}_t(u) \leq \langle \bar{g}_t, y_t - u \rangle + \frac{3}{N} \sum_{n=1}^N \|x_{n,t} - y_t\|.$$

Summing over t and applying the two assumed bounds gives the desired exact conjugate plus disagreement bound. □

LEMMA C.18 (Radial Prediction Lipschitz Envelope). *Let $H(G) = 0$ for $G = 0$ and $H(G) = h(\|G\|)G/\|G\|$ otherwise. Suppose that on $0 \leq r \leq R$ the scalar profile satisfies*

$$|h'(r)| \leq A, \quad |h(r)|/r \leq A \quad (r > 0).$$

Then H is A -Lipschitz on the closed ball $\{G : \|G\| \leq R\}$. Consequently $H \circ \text{clip}_R$ is also A -Lipschitz on all of $\mathcal{H}$.

Proof. Away from the origin, the derivative of the radial map has radial eigenvalue $h'(r)$ and tangential eigenvalues $h(r)/r$. The expressions therefore bound the operator norm of the derivative by A . Continuity at the origin follows from $|h(r)| \leq Ar$. Integrating the derivative along the line segment between two points in the ball gives the Lipschitz bound. The clipping map is the metric projection onto a closed Hilbert ball, hence is 1-Lipschitz, so composition with clip_R preserves the same envelope. $\square$

C.5 KT Specialization

For the KT potential, the regret proof uses the following auxiliary facts.

LEMMA C.19 (KT Square-Root Convexity). *For the KT potential and every $0 < R < t + 1$, the map*

$$z \mapsto F_t^{\text{KT}}(\sqrt{z})$$

is convex on $[0, R^2]$.

The proof follows from the standard KT excellence calculation from the Orabona-Pal coin-betting paper [94], translated to the square-root convexity formulation of Definition 6.2.

LEMMA C.20 (KT Zero Stirling Lower Bound). *For every $T \geq 1$,*

$$F_T^{\text{KT}}(0) \geq \frac{\varepsilon}{e\sqrt{\pi T}}.$$

Proof. At the origin the Gamma recurrence gives the exact two-step relation

$$F_{t+2}^{\text{KT}}(0) = F_t^{\text{KT}}(0) \frac{t+1}{t+2}.$$

The base cases $t = 1, 2$ satisfy the lower bound directly. The inequality

$$\frac{t+1}{t+2} \cdot \frac{1}{\sqrt{t}} \geq \frac{1}{\sqrt{t+2}}$$

then propagates the $1/\sqrt{t}$ envelope separately along the odd and even subsequences. $\square$

LEMMA C.21 (KT Prediction Lipschitz Envelope). *Let $B_t^{\text{KT}}(G) = \text{Bet}_{t+1}^{\text{KT}}(\text{clip}_t(G))$. For every $t \geq 1$,*

$$\|B_t^{\text{KT}}(a) - B_t^{\text{KT}}(b)\| \leq C_\star^{\text{KT}}(\varepsilon) \frac{2^t \log(t+2)}{\sqrt{t+1}} \|a - b\|,$$

where $C_\star^{\text{KT}}(\varepsilon) = (\varepsilon + 1)(1 + 2/\log 2) + 1$.

Proof. On the radius- t ball, the KT radial profile is $h_{t+1}^{\text{KT}}(r) = \beta_{t+1}^{\text{KT}}(r)F_t^{\text{KT}}(r)$. The identity $\beta_{t+1}^{\text{KT}}(r) = r/(t+1)$, the Gamma recurrence, and the elementary digamma bound $\psi(x+1) - \psi(x) \leq 1/x$ give the two scalar estimates

$$\begin{aligned} |(h_{t+1}^{\text{KT}})'(r)| &\leq C_\star^{\text{KT}}(\varepsilon) \frac{2^t \log(t+2)}{\sqrt{t+1}}, \\ |h_{t+1}^{\text{KT}}(r)|/r &\leq C_\star^{\text{KT}}(\varepsilon) \frac{2^t \log(t+2)}{\sqrt{t+1}} \end{aligned} \quad (0 < r \leq t).$$

Applying Lemma C.18 gives this envelope for the radial map on the ball, and the projection

clip_t is 1-Lipschitz. Hence the clipped prediction map has the same envelope. $\square$

LEMMA C.22 (Linear KT Gossip Schedule). *Fix a rate $0 < \eta < 1$ and endowment $\varepsilon > 0$.*

Define

$$C_{\star}^{\text{KT}}(\varepsilon) = (\varepsilon + 1) \left(1 + \frac{2}{\log 2} \right) + 1.$$

If $q(t) = \lceil ct \rceil$ and $c \geq -2 \log 2 / \log \eta$, then the KT disagreement sum is $O(\sqrt{T})$:

$$\begin{aligned} 2\sqrt{N} \sum_{t=1}^T \left(C_{\star}^{\text{KT}}(\varepsilon) \frac{2^t \log(t+2)}{\sqrt{t+1}} \right) \sum_{s < t} \eta^{Q(s,t)} \\ \leq 48 C_{\star}^{\text{KT}}(\varepsilon) \sqrt{N} \sqrt{T} \end{aligned}$$

for every $T \geq 1$.

Proof. Lemma C.21 gives a clipped prediction envelope with scalar part

$$C_{\star}^{\text{KT}}(\varepsilon) \frac{2^t \log(t+2)}{\sqrt{t+1}}.$$

For the linear schedule,

$$Q(s, t) = \sum_{r=s}^{t-1} \lceil cr \rceil \geq c(t-1) \quad (s < t).$$

Since $0 < \eta < 1$, larger exponents make $\eta^{Q(s,t)}$ smaller. The condition $c \geq -2 \log 2 / \log \eta$ ensures that the exponential decay from $\eta^{Q(s,t)}$ cancels the 2^t factor in $L^{\text{KT}}(t)$ uniformly enough that

$$C_{\star}^{\text{KT}}(\varepsilon) \frac{2^t \log(t+2)}{\sqrt{t+1}} \sum_{s < t} \eta^{Q(s,t)} \leq \frac{12 C_{\star}^{\text{KT}}(\varepsilon)}{\sqrt{t}}.$$

Applying Lemma C.16 gives the $48 C_{\star}^{\text{KT}}(\varepsilon) \sqrt{N} \sqrt{T}$ bound. $\square$

THEOREM C.23 (Expected KT Network Regret from Compressed Disagreement). *Let DECO-EF run over a real Hilbert space $\mathcal{H}$ with KT potential. Assume the expected compressed-gossip hypotheses of Theorem C.13, put $\mu = 1 - \rho$, $\chi = \mu/4$, and $\bar{\eta} = 1 - \gamma\mu/2$, and let $q(t) = \lceil ct \rceil$, where $c \geq -2 \log 2 / \log \bar{\eta}$. Define*

$$C_{\star}^{\text{KT}}(\varepsilon) = (\varepsilon + 1) \left(1 + \frac{2}{\log 2} \right) + 1,$$

$$C_{\text{net}}^{\text{KT}} = (2 + \chi) 24 C_{\star}^{\text{KT}}(\varepsilon) \sqrt{N}.$$

Then, for all $u \in \mathcal{H}$ and all integers $T \geq 1$,

$$\mathbf{E} R_T^{\text{net}}(u) \leq \Psi_T^{\text{KT}}(\|u\|) + \varepsilon + 3C_{\text{net}}^{\text{KT}} \sqrt{T}.$$

Proof. Work with the fixed γ from the statement and the corresponding compressed effective rate $\bar{\eta}$. The proof takes the global-loss route detailed in Appendix C.7: compare every local prediction to the centralized KT prediction at the network-average state, bound the centralized term by radial Hilbert-space coin betting on the averaged gradients, and use Theorem C.28 for the decentralized correction. The tail estimate in Lemma C.22 supplies the schedule bookkeeping at rate $\bar{\eta}$. The final step upper-bounds the exact KT terminal conjugate by Ψ_T^{KT} and adds the prediction-disagreement penalty. $\square$

C.6 Shifted-Exponential Specialization

The shifted-exponential potential (6.14) uses the same route. Its verification as an excellent coin-betting potential is the exponential-potential calculation of [94] with the time index shifted by one. Its betting fraction is

$$\beta_t^{\text{SE}}(x) = \tanh \left(\frac{x}{t+1} \right).$$

LEMMA C.24 (Shifted-Exponential Prediction Lipschitz Envelope). *Let $B_t^{\text{SE}}(G) = \text{Bet}_{t+1}^{\text{SE}}(\text{clip}_t(G))$. For every $t \geq 1$,*

$$\|B_t^{\text{SE}}(a) - B_t^{\text{SE}}(b)\| \leq (2\varepsilon + 1) \frac{\exp(t/2)}{\sqrt{t+1}} \|a - b\|.$$

Proof. The shifted-exponential scalar profile is

$$h_{t+1}^{\text{SE}}(r) = \tanh\left(\frac{r}{t+2}\right) \frac{\varepsilon}{\sqrt{t+1}} \exp\left(\frac{r^2}{2(t+1)}\right).$$

On $0 \leq r \leq t$, the bounds $\tanh x \leq x$, $(\tanh x)' \leq 1$, and $\exp(r^2/(2(t+1))) \leq \exp(t/2)$ give

$$|(h_{t+1}^{\text{SE}})'(r)|, \quad |h_{t+1}^{\text{SE}}(r)|/r \leq (2\varepsilon + 1) \frac{\exp(t/2)}{\sqrt{t+1}}.$$

The radial Lipschitz lemma and nonexpansiveness of clip_t give the desired constant. $\square$

LEMMA C.25 (Linear Shifted-Exponential Gossip Schedule). *Fix a rate $0 < \eta < 1$ and endowment $\varepsilon > 0$. Define*

$$C_{\star}^{\text{SE}}(\varepsilon) = 2\varepsilon + 1.$$

If $q(t) = \lceil ct \rceil$ and $c \geq -3/(2 \log \eta)$, then the shifted-exponential disagreement sum is $O(\sqrt{T})$:

$$\begin{aligned} & 2\sqrt{N} \sum_{t=1}^T \left(C_{\star}^{\text{SE}}(\varepsilon) \frac{\exp(t/2)}{\sqrt{t+1}} \right) \sum_{s < t} \eta^{Q(s,t)} \\ & \leq 4e^{3/2} C_{\star}^{\text{SE}}(\varepsilon) \sqrt{N} \sqrt{T}. \end{aligned}$$

Proof. The shifted-exponential Lipschitz lemma gives a clipped prediction envelope with scalar part

$$C_{\star}^{\text{SE}}(\varepsilon) \frac{\exp(t/2)}{\sqrt{t+1}}.$$

The same lower bound $Q(s, t) \geq c(t - 1)$ for the linear schedule gives

$$\sum_{s < t} \eta^{Q(s, t)} \leq t \exp\left(-\frac{3}{2}(t - 1)\right).$$

Thus

$$C_{\star}^{\text{SE}}(\varepsilon) \frac{\exp(t/2)}{\sqrt{t+1}} \sum_{s < t} \eta^{Q(s, t)} \leq \frac{e^{3/2} C_{\star}^{\text{SE}}(\varepsilon)}{\sqrt{t}},$$

and Lemma C.16 yields the desired constant. $\square$

THEOREM C.26 (Expected Shifted-Exponential Network Regret). *Let DECO-EF run over a real Hilbert space $\mathcal{H}$ with the shifted-exponential potential. Assume the expected compressed-gossip hypotheses of Theorem C.13, define $\mu = 1 - \rho$, $\chi = \mu/4$, and $\bar{\eta} = 1 - \gamma\mu/2$, and let $q(t) = \lceil ct \rceil$ with $c \geq -3/(2 \log \bar{\eta})$. Define*

$$C_{\star}^{\text{SE}}(\varepsilon) = 2\varepsilon + 1, \quad C_{\text{net}}^{\text{SE}} = (2 + \chi) 2e^{3/2} C_{\star}^{\text{SE}}(\varepsilon) \sqrt{N}.$$

Then, for all $u \in \mathcal{H}$ and all integers $T \geq 1$,

$$\mathbf{E} R_T^{\text{net}}(u) \leq \Psi_T^{\text{SE}}(\|u\|) + \varepsilon + 3C_{\text{net}}^{\text{SE}} \sqrt{T}.$$

Proof. Apply the generic network regret template Theorem C.17. The centralized term is controlled by Theorem C.9, and the closed-form terminal envelope follows from Lemma C.7. The decentralized term is Theorem C.29, whose constant is supplied by Lemma C.25. $\square$

C.7 Global-Loss and Prediction-Disagreement Details

LEMMA C.27 (Global-Loss Comparison Through a Centralized Prediction). *Let $\bar{\ell}_t$ be the network loss from Equation (6.2), and define*

$$\bar{g}_t = \frac{1}{N} \sum_{m=1}^N g_{m,t}.$$

Assume each $\ell_{m,t}$ is convex and 1-Lipschitz on $\mathcal{H}$, and $g_{m,t} \in \partial \ell_{m,t}(x_{m,t})$. Then for every reference point $y_t \in \mathcal{H}$ and every comparator $u \in \mathcal{H}$,

$$\frac{1}{N} \sum_{n=1}^N \bar{\ell}_t(x_{n,t}) - \bar{\ell}_t(u) \leq \langle \bar{g}_t, y_t - u \rangle + \frac{3}{N} \sum_{n=1}^N \|x_{n,t} - y_t\|.$$

Proof. Since $\bar{\ell}_t$ is an average of 1-Lipschitz functions, it is also 1-Lipschitz. Hence

$$\frac{1}{N} \sum_{n=1}^N \bar{\ell}_t(x_{n,t}) - \bar{\ell}_t(u) \leq \bar{\ell}_t(y_t) - \bar{\ell}_t(u) + \frac{1}{N} \sum_{n=1}^N \|x_{n,t} - y_t\|.$$

It remains to control $\bar{\ell}_t(y_t) - \bar{\ell}_t(u)$. For each agent m , insert and subtract $\ell_{m,t}(x_{m,t})$:

$$\begin{aligned} \ell_{m,t}(y_t) - \ell_{m,t}(u) &= \ell_{m,t}(x_{m,t}) - \ell_{m,t}(u) \\ &\quad + \ell_{m,t}(y_t) - \ell_{m,t}(x_{m,t}). \end{aligned}$$

By convexity at $x_{m,t}$,

$$\ell_{m,t}(x_{m,t}) - \ell_{m,t}(u) \leq \langle g_{m,t}, x_{m,t} - u \rangle.$$

By 1-Lipschitzness,

$$\ell_{m,t}(y_t) - \ell_{m,t}(x_{m,t}) \leq \|y_t - x_{m,t}\|.$$

Therefore

$$\ell_{m,t}(y_t) - \ell_{m,t}(u) \leq \langle g_{m,t}, x_{m,t} - u \rangle + \|x_{m,t} - y_t\|.$$

Now decompose

$$\langle g_{m,t}, x_{m,t} - u \rangle = \langle g_{m,t}, y_t - u \rangle + \langle g_{m,t}, x_{m,t} - y_t \rangle.$$

Since $\ell_{m,t}$ is 1-Lipschitz, $\|g_{m,t}\| \leq 1$. Hence, by the Cauchy Schwarz inequality,

$$\langle g_{m,t}, x_{m,t} - y_t \rangle \leq \|x_{m,t} - y_t\|.$$

Combining the last steps,

$$\ell_{m,t}(y_t) - \ell_{m,t}(u) \leq \langle g_{m,t}, y_t - u \rangle + 2\|x_{m,t} - y_t\|.$$

Averaging over m yields

$$\bar{\ell}_t(y_t) - \bar{\ell}_t(u) \leq \langle \bar{g}_t, y_t - u \rangle + \frac{2}{N} \sum_{m=1}^N \|x_{m,t} - y_t\|.$$

Substituting this into the first step gives the claim. $\square$

THEOREM C.28 (Expected Compressed KT Prediction Disagreement). *Let $DECO - EF$ run over a real Hilbert space $\mathcal{H}$ with the KT potential. Assume the expected compressed-gossip hypotheses of Theorem C.13, and let $\mu = 1 - \rho$, $\bar{\eta} = 1 - \frac{\gamma\mu}{2}$. Put $\chi = \mu/4$ and*

$$C_{\star}^{\text{KT}}(\varepsilon) = (\varepsilon + 1) \left(1 + \frac{2}{\log 2} \right) + 1,$$

$$C_{\text{net}}^{\text{KT}} = (2 + \chi) 24 C_{\star}^{\text{KT}}(\varepsilon) \sqrt{N}.$$

Let $q(t) = \lceil ct \rceil$ with

$$c \geq -\frac{2 \log 2}{\log \bar{\eta}}.$$

Then

$$\mathbf{E} \sum_{t=0}^{T-1} \frac{1}{N} \sum_{n=1}^N \|x_{n,t} - \text{Bet}_{t+1}^{\text{KT}}(\bar{G}_t)\| \leq C_{\text{net}}^{\text{KT}} \sqrt{T}.$$

Proof. Write

$$H_t(z) = \text{Bet}_{t+1}^{\text{KT}}(z), \quad B_t(z) = H_t(\text{clip}_t(z)).$$

The clipped map B_t is the prediction actually used by the agents, the clean state $G_{n,t}$ itself is not clipped before gossip. Let

$$A^{\text{KT}}(t) = C_{\star}^{\text{KT}}(\varepsilon) \frac{2^t \log(t+2)}{\sqrt{t+1}}.$$

By Lemma C.21, B_t is $A^{\text{KT}}(t)$ -Lipschitz. Moreover, by Lemma C.10,

$$\bar{G}_t = \sum_{s=0}^{t-1} (-\bar{g}_s), \quad \|\bar{g}_s\| \leq 1,$$

and therefore $\|\bar{G}_t\| \leq t$. Hence $\text{clip}_t(\bar{G}_t) = \bar{G}_t$. The $t = 0$ contribution is zero. For $t \geq 1$,

$$\begin{aligned} & \mathbf{E} \frac{1}{N} \sum_{n=1}^N \|H_t(\text{clip}_t(G_{n,t})) - H_t(\bar{G}_t)\| \\ & \leq A^{\text{KT}}(t) \mathbf{E} \frac{1}{N} \sum_{n=1}^N \|G_{n,t} - \bar{G}_t\| \\ & \leq (2 + \chi) \sqrt{N} A^{\text{KT}}(t) \sum_{s < t} \bar{\eta}^{Q(s,t)} \end{aligned}$$

by Lemma C.15. Summing over $t = 1, \dots, T-1$ and enlarging to $t \leq T$, the linear schedule

estimate Lemma C.22 gives

$$2\sqrt{N} \sum_{t=1}^T A^{\text{KT}}(t) \sum_{s < t} \bar{\eta}^{Q(s,t)} \leq 48C_{\star}^{\text{KT}}(\varepsilon)\sqrt{N}\sqrt{T}.$$

Taking half of this estimate and multiplying by the state-unrolling factor $2 + \chi$ yields exactly $C_{\text{net}}^{\text{KT}}\sqrt{T}$. $\square$

THEOREM C.29 (Expected Compressed Shifted-Exponential Prediction Disagreement). *Let DECO – EF run over a real Hilbert space $\mathcal{H}$ with the shifted-exponential potential. Assume the expected compressed-gossip hypotheses of Theorem C.13, define $\mu = 1 - \rho$, $\chi = \mu/4$, $\bar{\eta} = 1 - \gamma\mu/2$, and $C_{\star}^{\text{SE}}(\varepsilon) = 2\varepsilon + 1$. Put*

$$C_{\text{net}}^{\text{SE}} = (2 + \chi) 2e^{3/2} C_{\star}^{\text{SE}}(\varepsilon) \sqrt{N}.$$

Let $q(t) = \lceil ct \rceil$ with $c \geq -3/(2 \log \bar{\eta})$. Then

$$\begin{aligned} \mathbf{E} \sum_{t=0}^{T-1} \frac{1}{N} \sum_{n=1}^N \left\| \text{Bet}_{t+1}^{\text{SE}}(\text{clip}_t(G_{n,t})) - \text{Bet}_{t+1}^{\text{SE}}(\bar{G}_t) \right\| \\ \leq C_{\text{net}}^{\text{SE}} \sqrt{T}. \end{aligned}$$

Proof. Set

$$A^{\text{SE}}(t) = C_{\star}^{\text{SE}}(\varepsilon) \frac{\exp(t/2)}{\sqrt{t+1}}.$$

By Lemma C.24, the clipped shifted-exponential prediction map is $A^{\text{SE}}(t)$ -Lipschitz. As in the KT case, $\|\bar{G}_t\| \leq t$, so the centralized prediction is unclipped. For $t \geq 1$, the state-

disagreement unrolling gives

$$\begin{aligned}
& \mathbf{E} \frac{1}{N} \sum_n \left\| \text{Bet}_{t+1}^{\text{SE}}(\text{clip}_t(G_{n,t})) - \text{Bet}_{t+1}^{\text{SE}}(\bar{G}_t) \right\| \\
& \leq (2 + \chi) \sqrt{N} A^{\text{SE}}(t) \\
& \quad \cdot \sum_{s < t} \bar{\eta}^{Q(s,t)}.
\end{aligned}$$

The $t = 0$ contribution is zero. Summing and applying Lemma C.25 yields

$$(2 + \chi) 2e^{3/2} C_{\star}^{\text{SE}}(\varepsilon) \sqrt{N} \sqrt{T} = C_{\text{net}}^{\text{SE}} \sqrt{T}.$$

□

The expected network-regret theorem is obtained through the reusable template Theorem C.17: combine Lemma C.27, the Hilbert coin-betting theorem, the terminal conjugate envelopes from Lemma C.7, and one of the two expected prediction-disagreement theorems above.

C.8 Pathwise Network-Regret Add-On

Given the expected-regret analysis, we can also obtain worst-case, or pathwise, network-regret bounds given a stronger pathwise condition on the compressor.

DEFINITION C.30 (Pathwise Compressor Condition). *The compressor satisfies a pathwise residual bound if there is $0 \leq \alpha^{\text{path}} < 1$ such that, for all $\omega \in \Omega$ and all $v \in \mathcal{H}$,*

$$\|\mathcal{C}(\omega, v) - v\|^2 \leq \alpha^{\text{path}} \|v\|^2. \tag{C.10}$$

THEOREM C.31 (Pathwise Compressed-Message Network Regret). *Let DECO-EF run over a real Hilbert space $\mathcal{H}$ with either the KT potential (6.13) or the shifted-exponential poten-*

tial (6.14). Assume W is a valid gossip matrix for a connected communication graph, and let ρ be the graph-derived product- ℓ_2 zero-mean contraction factor satisfying $0 \leq \rho < 1$. Let the compressor be as in Definition 6.1 and additionally satisfy Definition C.30 with parameter $0 \leq \alpha^{\text{path}} < 1$. Put

$$\mu = 1 - \rho, \quad \sigma = \sqrt{\alpha^{\text{path}}}.$$

Choose

$$0 < \gamma \leq \min \left\{ 1, \frac{1 - \sigma}{2\sigma \left(1 + \frac{4}{\mu}\right) + \frac{\mu}{2}} \right\}.$$

Define $\eta = 1 - \gamma\mu/2$, and use a linear gossip schedule $q(t) = \lceil ct \rceil$, where $c \geq -2\log 2 / \log \eta$ for KT and $c \geq -3/(2\log \eta)$ for the shifted-exponential potential. Put $\chi = \mu/4$ and

$$C_{\text{sched}}^{\text{KT}} = 48 \left((\varepsilon + 1) \left(1 + \frac{2}{\log 2} \right) + 1 \right),$$

$$C_{\text{sched}}^{\text{SE}} = 4e^{3/2}(2\varepsilon + 1).$$

For the chosen potential $P \in \{\text{KT}, \text{SE}\}$, let F_T^* be its restricted terminal conjugate, with C_{sched}^P selecting the matching constant above, and set

$$C_{\text{net}}^P = (2 + \chi) \frac{C_{\text{sched}}^P}{2} \sqrt{N}, \quad C_{\text{reg}}^{P, \text{path}} = 3C_{\text{net}}^P.$$

Then, for all comparators $u \in \mathcal{H}$ and all integers $T \geq 1$,

$$R_T^{\text{net}}(u) \leq F_T^*(\|u\|) + \varepsilon + C_{\text{reg}}^{P, \text{path}} \sqrt{T}. \quad (\text{C.11})$$

The constants are independent of u and T .

The proof is a straightforward pathwise adaptation of the expected-regret analysis.