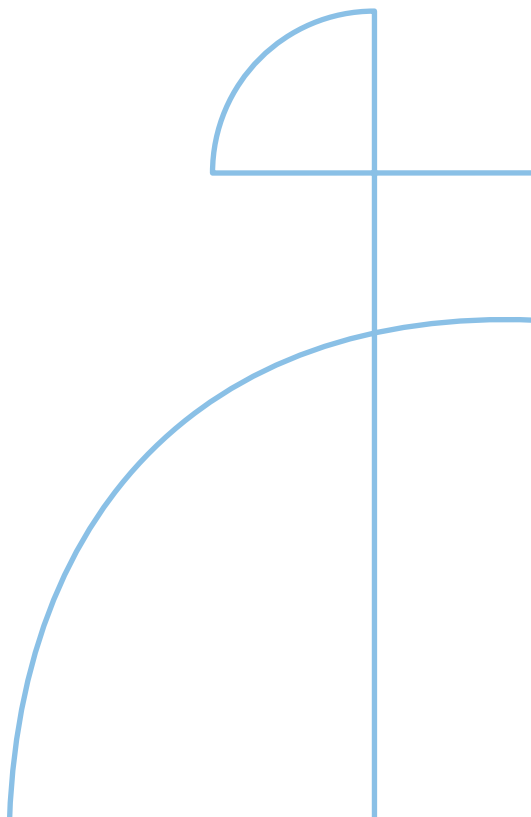
Doctoral Thesis in Electrical Engineering

Toward Efficient Federated Learning over Wireless Networks: Novel Frontiers in Resource Optimization

AFSANEH MAHMOUDI BENHANGI

KTH ROYAL INSTITUTE OF TECHNOLOGY



Toward Efficient Federated Learning over Wireless Networks: Novel Frontiers in Resource Optimization

AFSANEH MAHMOUDI BENHANGI

Academic Dissertation which, with due permission of the KTH Royal Institute of Technology, is submitted for public defence for the Degree of Doctor of Philosophy on Monday the 10th February 2025, at 1:00 p.m. in Ka-Sal C, Kistagången 16, Kista.

Doctoral Thesis in Electrical Engineering
KTH Royal Institute of Technology
Stockholm, Sweden 2025

ISBN 978-91-8106-165-9
TRITA-EECS-AVL-2025:13

Printed by: Universitetsservice US-AB, Sweden 2025

Abstract

With the rise of the Internet of Things (IoT) and 5G networks, edge computing addresses critical limitations in cloud computing's quality of service. Machine learning (ML) has become essential in processing IoT-generated data at the edge, primarily through distributed optimization algorithms that support predictive tasks. However, state-of-the-art ML models demand substantial computational and communication resources, often exceeding the capabilities of wireless devices. Moreover, training these models typically requires centralized access to datasets, but transmitting such data to the cloud introduces significant communication overhead, posing a critical challenge to resource-constrained systems. Federated Learning (FL) is a promising iterative approach that reduces communication costs through local computation on devices, where only model parameters are shared with a central server. Accordingly, every communication iteration of FL experiences costs such as computation, latency, bandwidth, and energy. Although FL enables distributed learning across multiple devices without exchanging raw data, its success is often hindered by the limitations of wireless communication overhead, including traffic congestion and device resource constraints. To address these challenges, this thesis presents cost-effective methods for making FL training more efficient in resource-constrained wireless environments.

Initially, we investigate challenges in distributed training over wireless networks, addressing background traffic and latency that impede communication iterations. We introduce the cost-aware causal FL algorithm (FedCau), which balances training performance with communication and computation costs through a novel iteration-termination method, removing the need for future FL information. A multi-objective optimization problem is formulated, integrating FL loss and iteration costs, with communication managed via the slotted-ALOHA, CSMA/CA, and OFDMA protocols. The framework is extended to include both convex and non-convex loss functions, and results are compared with established communication-efficient methods, including Lazily Aggregated Quantized Gradient (LAQ). Additionally, we develop A-LAQ (Adaptive LAQ), which conserves energy while maintaining high test accuracy by dynamically adjusting bit allocation for local model updates during FL iterations.

Next, we leverage cell-free massive multiple-input multiple-output (CFm-MIMO) networks to address the high latency in large-scale FL deployments. This architecture allows for simultaneous service to many users on the same time/frequency resources, mitigating the latency bottleneck through spatial multiplexing. Accordingly, we propose optimized uplink power allocation schemes that minimize the trade-off between energy consumption and latency, enabling more iterations under given energy and latency constraints and leading to substantial gains in FL test accuracy. In this regard, we present three approaches, beginning with a method that jointly minimizes the users' uplink energy and FL training latency. This approach optimizes the trade-off between each user's uplink latency and energy consumption, factoring in how individual transmit power impacts the energy and latency of other users to jointly reduce overall uplink energy consumption and FL training latency.

Furthermore, to address the straggler effect, we propose an adaptive mixed-resolution quantization scheme for local gradient updates, which considers high resolution only for essential entries and utilizes dynamic power control. Finally, we introduce EFCAQ, an energy-efficient FL in CFmMIMO networks, with a proposed adaptive quantization to co-optimize the straggler effect and the overall user energy consumption while minimizing the FL loss function through an adaptive number of local iterations of users.

Through extensive theoretical analysis and experimental validation, this thesis demonstrates that the proposed methods outperform state-of-the-art algorithms across various FL setups and datasets. These contributions pave the way for energy-efficient and low-latency FL systems, making them more practical for use in real-world wireless networks.

Keywords: Federated Learning, Optimization, Cell-free massive MIMO, Resource allocation, Energy, Latency.

Sammanfattning

Framväxten av sakernas Internet (IoT, Internet of Things) och 5G-nät begränsas av tjänstekvaliteten i molnet, men kantberäkningar kan adressera dessa problem. Maskininlärning (ML) kommer bli avgörande för att bearbeta IoT-genererad data vid kanten av nätet, främst genom att använda distribuerade optimeringsalgoritmer för prediktion. Dagens ML-modeller kräver dock stora beräknings- och kommunikationsresurser som ofta överstiger kapaciteten hos enskilda trådlösa enheter. Dessutom kräver träningen av dessa modeller vanligtvis centraliserad åtkomst till stora datamängder, men överföringen av denna data till molnet har betydande kommunikationskostnader, vilket är en kritisk utmaning för att driva resursbegränsade system. Federerad inlärning (FL) är en lovande iterativ ML-metod som minskar kommunikationskostnaderna genom att genomföra lokala beräkning på lokalt tillgänglig data på enheterna och endast dela modellparametrar med en central server. Varje iteration i FL har vissa kostnader när det gäller beräkningar, latens, bandbredd och energi. Även om FL möjliggör distribuerad inlärning över flera enheter utan att utbyta rådata, begränsas metoden i praktiken av den trådlösa kommunikationstekniken, t.ex. trafikstockningar i nätet och energibegränsningar i enheterna. För att adressera dessa problem presenterar denna avhandling kostnadseffektiva metoder för att göra FL-träning mer effektiv i resursbegränsade trådlösa miljöer.

Inledningsvis löser vi forskningsproblem relaterade till distribuerad inlärning över trådlösa nätverk med fokus på hur annan datatrafik och kommunikationslatensen begränsar FL-iterationerna. Vi introducerar den kostnadsmedvetna kausala FL-algoritmen FedCau som balanserar träningsprestanda mot kommunikations- och beräkningskostnader. En viktig del av lösningen är en ny termineringsmetod som tar bort det tidigare behovet av att ha information om framtida beräkningar vid termineringen. Ett flermålsoptimeringsproblem formuleras för att integrera FL-kostnader med kommunikation som genomförs med ALOHA-, CSMA/CA- eller OFDMA-protokollen. Ramverket omfattar både konvexa och icke-konvexa förlustfunktioner och resultaten jämförs med etablerade kommunikationseffektiva metoder, inklusive Lazily Aggregated Quantized Gradient (LAQ). Dessutom utvecklar vi A-LAQ (adaptiv LAQ) som sparar energi samtidigt som hög ML-noggrannhet bibehålls genom att dynamiskt justera bitallokeringen för de lokala modelluppdateringarna under FL-iterationerna.

Därefter analyserar vi hur cellfri massiv multiple-input multiple-output (CFmMIMO) teknik kan användas för att hantera den höga kommunikationslatensen som annars uppstår när storskaliga modeller tränas genom FL. Denna nya nätarkitektur består av många samarbetande basstationer vilket möjliggör att många användare kan skicka modelluppdateringar samtidigt på samma frekvenser genom rumslig multiplexing, vilket drastiskt minskar latensen. Vi föreslår nya upplänkseffektregleringsscheman som optimerar avvägningen mellan energiförbrukning och latens. Denna lösning möjliggör fler FL-iterationer under givna energi- och latensbegränsningar och leder till betydande vinster i FL-testnoggrannheten. Vi presenterar tre tillvägagångssätt varav det första är en metod som minimerar en matematisk avvägningen

mellan varje användares upplänkslatens och energiförbrukning. Metoden tar hänsyn till hur de individuella sändningseffekterna påverkar andra användares energi och latens för att gemensamt minska den totalaenergiförbrukningen och FL-träningsfördröjningen. Vårt andra bidrag är en metod för att hantera eftersläpningseffekter genom ett adaptivt kvantiseringsschema med blandad upplösning för de lokala gradientuppdateringarna. I detta schema används hög kvantiseringsupplösning endast för viktiga variabler och vi använder även dynamisk effektreglering. Slutligen introducerar vi EFCAQ som är en energieffektiv FL-metod för CFmMIMO-nätverk. EFCAQ kombinerar ett nytt adaptivt kvantiseringsschema med att samoptimera eftersläpningseffekten och användarens totala energiförbrukning så att FL-förlustfunktionen minimeras genom att använda ett adaptivt antal lokala iterationer hos varje användare.

Genom omfattande teoretisk analys och experimentell validering visar denna avhandling att de föreslagna metoderna överträffar tidigare kända algoritmer i olika FL-scenarier och för olika datauppsättningar. Våra bidrag banar väg för energieffektiva FL-system med låg latens, vilket gör dem mer praktiska för användning i verkliga trådlösa nätverk.

Nyckelord: Federerad inlärning, Optimering, Cell-fri massiv MIMO, Resursallokering, Energieffektivitet, Latens.

List of Papers

List of Papers Included in The Thesis:

Paper 1 - Machine learning over networks: co-design of distributed optimization and communications

Afsaneh Mahmoudi, Hossein S. Ghadikolaei, Carlo Fischione

IEEE 21st International Workshop on Signal Processing Advances in Wireless Communications (IEEE SPAWC), 2020.

Paper 2 - FedCau: A Proactive Stop Policy for Communication and Computation Efficient Federated Learning

Afsaneh Mahmoudi, Hossein S. Ghadikolaei, José Mairton Barros Da Silva Júnior, Carlo Fischione

IEEE Transactions on Wireless Communications, 2024.

Paper 3 - A-LAQ: Adaptive Lazily Aggregated Quantized Gradient

Afsaneh Mahmoudi, José Mairton Barros Da Silva Júnior, Hossein S. Ghadikolaei, Carlo Fischione

IEEE Globecom Workshops (GC Wkshps) on Edge Learning over 5G Mobile Networks and Beyond, 2022.

Paper 4 - Low-Latency and Energy-Efficient Federated Learning over Cell-Free Networks: A Trade-off Analysis

Afsaneh Mahmoudi, Mahmoud Zaher, Emil Björnson

Submitted to IEEE Open Journal of Communications Society, 2025.

Paper 5 - Adaptive Quantization Resolution and Power Control for Federated Learning over Cell-free Networks

Afsaneh Mahmoudi, Emil Björnson

IEEE Globecom Workshops (GC Wkshps) on Ubiquitous Network Intelligence for Next Generation Wireless Networks, 2024.

Paper 6 - Accelerating Energy-Efficient Federated Learning in Cell-Free Networks with Adaptive Quantization

Afsaneh Mahmoudi, Ming Xiao, Emil Björnson

Submitted to IEEE Transactions on Machine Learning in Communications and Networking, 2024.

List of Papers Not Included in The Thesis:

Paper A - Cost-efficient distributed optimization in machine learning over wireless networks

Afsaneh Mahmoudi, Hossein S. Ghadikolaei, Carlo Fischione

IEEE International Conference on Communications (IEEE ICC), 2020.
Papers 1 and 2 are the extensions of this paper.

Paper B - Joint Energy and Latency Optimization in Federated Learning over Cell-Free Massive MIMO Networks

Afsaneh Mahmoudi, Mahmoud Zaher, Emil Björnson

IEEE Wireless Communications and Networking Conference, 2024.
Paper 4 is an extension of this paper.

Acknowledgements

First and foremost, I would like to express my heartfelt gratitude to my supervisor, **Prof. Emil Björnson**, for his guidance, encouragement, and support throughout my Ph.D. journey. He has always been a source of great inspiration, sharing his deep scientific expertise alongside practical wisdom and thoughtful advice. Despite his busy schedule, he always made time to listen, offer guidance, and help me overcome challenges whenever I needed assistance. His dedication to fostering an environment of learning, collaboration, and excellence is truly remarkable, and his mentorship has left a lasting impact on both my academic and personal growth. It has been an honor and privilege to work under the supervision of someone whose professionalism, integrity, and genuine care for his team make him an outstanding role model. I am deeply grateful for the opportunity I have had to learn from him and for the lasting impact he has made on my academic and personal development.

I would like to express my gratitude to my co-supervisor, **Prof. Ming Xiao**, for the insightful discussions we shared and the invaluable feedback he provided throughout my Ph.D. journey. I am also thankful to my co-authors for the productive and rewarding collaborations we had during this time. My sincere appreciation goes to Prof. Mikael Johansson for dedicating his precious time and effort to thoroughly reviewing this thesis. I am deeply grateful to Assoc. Prof. Alexander Jung for graciously accepting the role of opponent in my defense. I also extend my thanks to the members of the grading committee, Assoc. Prof. Sepideh Pashami, Assoc. Prof. Le-Nam Tran, and Dr. Abdulrahman Alabbasi, as well as to Assoc. Prof. Isaac Skog for serving as the defense chair.

I extend my gratitude to all my colleagues and friends at CoS, whose companionship made this journey both encouraging and enjoyable, with special thanks to **Mahmoud Zaher** for all the collaborations and discussions we had. I am deeply thankful to my amazing friends: **Naghmeh**, **Forough Shahab**, **Sara Khosravi**, **Mohammad**, **Danial**, **Hamideh**, and **Kim Hammar**. A heartfelt thanks to **Sara Ehtesham** for over 20 years of wonderful friendship; her constant support, even from afar, has been invaluable, and she has truly been like the sister I never had. Finally, I thank all my dear friends in Stockholm and across Europe who have supported me along the way.

The most thanks and appreciation in my personal life would certainly go toward my dear husband, Mohammadreza (**Mamarz**); without his unwavering support, this journey would not have been possible to accomplish. I am also incredibly thankful for **Asal** and **Pambe**, my beloved furry companions, who brought me endless comfort and unconditional love during the challenging Ph.D. years. Last but not least, I am profoundly grateful to my family, parents and in-laws, for their boundless love and emotional support, which sustained me during this exciting yet challenging academic journey.

Afsaneh Mahmoudi,
Stockholm, January 2025

Contents

List of Figures

List of Tables

List of Acronyms

5G	Fifth-generation
6G	Sixth-generation
AdaDelta	Adaptive Delta
AdaGrad	Adaptive Gradient
A-LAQ	Adaptive Lazily Aggregated Quantized Gradient
ANN	Artificial Neural Network
AP	Access Point
AQUILA	Adaptive QUantization in devIce seLection strAtegy
BFGS	Broyden–Fletcher–Goldfarb–Shanno Method
CD	Coordinate Descent
CFmMIMO	Cell-Free Massive Multiple Input Multiple Output
CNN	Convolutional Neural Network
CoMP	Coordinated Multipoint
CSI	Channel State Information
CSMA/CA	Carrier-Sense Multiple Access with Collision Avoidance
DAdaQuant	Doubly-adaptive Quantization
DNN	Deep Neural Network
dom f	Domain of function f
EMQ	Exponent-Mantissa Quantization
FedAvg	Federated Averaging
FedLADA	Federated Local Adaptive Amended optimizer
FedOVA	Federated One-vs-all
FEEL	Federated Edge Learning
FIR	Finite Impulse Response
FL	Federated Learning
FTTQ	Federated Trained Ternary Quantization

GD	Gradient Descent
IID	Independent and Identically Distributed
IoT	Internet of Things
KKT	Karush–Kuhn–Tucker
LAQ	Lazily Aggregated Quantized Gradient
LP	Linear Programming
max-min	Maximum of Minimum
max-sum	Maximum of Summation
mMIMO	Massive Multiple Input Multiple Output
MIMO	Multiple Input Multiple Output
ML	Machine Learning
Non-IID	Non-Independent and Identically Distributed
OFDMA	Orthogonal Frequency-Division Multiplexing Access
QoS	Quality of Service
QP	Quadratic Programming
RGs	Research Gaps
ResNet	Residual Networks
RQs	Research Questions
SCA	Successive Convex Approximation
SGD	Stochastic Gradient Descent
SINR	Signal-to-Interference-plus-Noise Ratio
SNR	Signal-to-Noise Ratio
SQP	Sequential Quadratic Programming
TDD	Time-Division Duplex
TDMA	Time Division Multiple Access
UE	User Equipment

List of Notations

$\boldsymbol{W}$	Bold-font capital letter denoting a matrix
$\boldsymbol{w}$	Bold-font lower-case letter denoting a vector
$\|\cdot\|$	l_2-norm
$\lceil\cdot\rceil$	Ceiling value
$\lfloor\cdot\rfloor$	Floor value
$[w]_{i,j}$	Entry i, j of matrix $\boldsymbol{W}$
$[\boldsymbol{w}]_i$	Entry i of vector $\boldsymbol{w}$
$\mathbb{1}_x$	Indicator function
$\mathbf{1}$	Vector with all components one
$[N]$	Index set $\{1, 2, \ldots, N\}$ for a positive integer N
$\|\mathcal{A}\|$	Cardinality of the set $\mathcal{A}$
$\boldsymbol{w}^\top$	Transpose of vector $\boldsymbol{w}$
$\odot$	Element-wise (Hadamard) product
$\circ a$	Element-wise (Hadamard) power by a
$\oslash$	Element-wise (Hadamard) division
w	Italic font denoting a scalar
$\mathcal{N}_{\mathbb{C}}(a, b)$	Complex normal distribution with mean a and variance b
$\nabla_x f(.)$	Gradient of the function with respect to x
$\nabla_{xx} f(.)$	Hessian matrix of the function with respect to x

Chapter 1

Introduction

Fueled by the rise of connected wireless devices, the Internet of Things (IoT) generates vast amounts of data, driving a huge need for advanced processing solutions. Considering that IoT devices are connected to a central server in a star network, as shown in Figure 1.1, data must be transmitted to centralized servers for processing in traditional cloud computing. Then, the results must be sent back to the devices, resulting in significant pressure on the network. In addition, data transmission may require substantial bandwidth and resource costs, such as energy consumption, and the network's performance is likely to degrade as the size of the data increases [5]. To mitigate the challenges in conventional cloud computing, *edge computing* has emerged as a promising paradigm that brings computation and data storage near the devices [6], eliminating the heavy communication overhead to the cloud. Herein, machine learning (ML) is a prominent tool to analyze the data produced by edge devices [7]. Use cases include vehicular networks, smart cities, autonomous driving, surveillance cameras, intelligent healthcare, drones, security robots, and many industrial IoT cases [8–11].

The rapid growth of data volume has driven increased interest in distributed ML [12], where datasets are distributed across users rather than centralized. However, traditional distributed ML faces significant challenges, particularly the high communication overhead associated with transmitting gradient updates or even raw data to a central server. Federated Learning(FL) has emerged as a distinctive approach to address these challenges by avoiding raw data transmission between devices and servers, thus enhancing data privacy and reducing communication overhead [13, 14]. Alternatively, the devices perform local computations over their private data to obtain local parameters and then communicate those local parameters to servers over the wireless channel. However, computation and communication of such local parameters in large-scale FL require extensive resources, such as bandwidth, battery, and transmission power [15–18]. Therefore, the success of large-scale FL training greatly depends on developing communication, computation, or energy-efficient approaches. Many papers in the literature focus on developing such

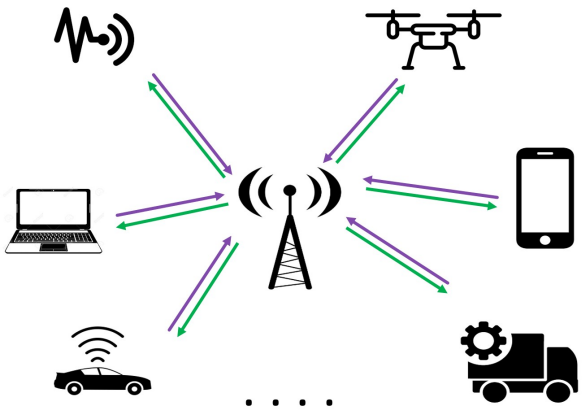


Figure 1.1: IoT devices connected to the central server in a star network. With the increasing number of IoT devices, data transmission hinders the quality of service due to the communication overhead.

efficient approaches [3,19–23] and optimal resource allocation [24–29] for large-scale FL over wireless networks. These works attempt to design cost-efficient approaches while they do not have access to the sequence of updated FL global parameters over the iterations or the required cost to perform those FL iterations. Moreover, these existing methods adapt FL to communication channels instead of developing FL methods that, by design, include communication limitations in the algorithms. Although these papers develop resource-efficient methods for FL over wireless networks, there is still a need to investigate further the cost of performing the iterative FL procedures without any need to know the future information of FL training.

In this thesis, we develop communication and computation efficient FL methods over wireless networks by introducing *causal* approaches as the methods that eliminate the need to know future training information. Moreover, we investigate and develop communication and computation efficient methods that model the general cost of performing FL over wireless networks regardless of the specific communication protocol applied for the parameter transmission. Therefore, our theoretical results are vastly applicable to many wireless communication protocols, such as slotted-ALOHA, carrier-sense multiple access with collision avoidance (CSMA/CA), orthogonal frequency-division multiplexing access (OFDMA), time division multiple access (TDMA), or any deliberate scheduling protocol. Accordingly, this work has a significant potential to be exploited and expanded by academic and industrial researchers due to the applicability of the proposed methods to a wide range of wireless communication protocols. To the best of our knowledge,

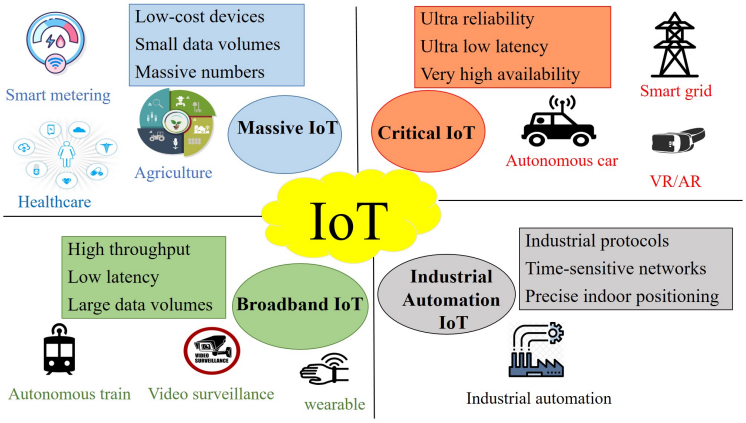


Figure 1.2: Four segments of cellular IoT with their applications: Massive IoT, Broadband IoT, Critical IoT, and Industrial Automation IoT [1].

the papers this thesis is based on are the first works considering the introduced causal approaches to develop communication and computation-efficient FL.

1.1 Edge Computing and IoT

ML has revolutionized many science and technology fields, tremendously impacting our daily lives, society, economy, and environment. The success of ML is mainly due to the availability of big datasets and large platforms that can provide vast amounts of dedicated computational and communication resources in the cloud. However, this centralized intelligence suffers from extra communication latency for inference and requires collecting all the data in a single node, which causes security and privacy vulnerabilities. A potential solution to these problems is to move the computations to the network edge, meaning that the data is processed locally in algorithms stored on a device. It helps to greatly reduce the power consumption and security vulnerability associated with processing data in the cloud.

Edge learning imposes several new challenges, such as bandwidth limitations hindering heavy coordination and computation procedures of ML algorithms; latency and message loss complicating ML coordination; privacy and security concerns blocking the sharing of datasets; restricted computational capability challenging the use of heavy training models; and the limitations of communication networks enforce inference with partial knowledge. Thus, there have been many efforts in developing edge learning techniques for training and inference tasks, which address communication complexity and privacy [30–36].

With the significant growth of edge devices connected via IoT or 5G networks,

we can categorize the connectivity requirements across various industries into four segments [1], illustrated in Figure 1.2. Massive IoT is a domain that connects many low-cost, narrow-bandwidth devices that intermittently transmit or receive minimal amounts of data. In addition, these devices may locate in challenging radio environments and operate solely on battery power. Moreover, when many devices access the wireless channel simultaneously, it overloads the channel and brings challenges due to network congestion. In this situation, IoT services can be hindered or fail due to the mismanagement of communication resources [37]. The other IoT segment is Broadband IoT, which utilizes mobile broadband technologies to deliver superior data transfer rates, low latency, extended battery life for devices, improved coverage, elevated uplink data transfer rates, and enhanced precision in device positioning compared to the Massive IoT segment.

Next, Critical IoT is characterized by its ability to provide low-latency and high-reliability data transfer. Unlike Broadband IoT, which offers low-latency service on a best-effort basis, Critical IoT can ensure data delivery within specified time constraints and level of reliability, even in the presence of a high network load. The last IoT segment is Industrial Automation IoT, which aims to enable seamless cellular connectivity integration into the wired industrial infrastructure used for advanced real-time automation. It includes capabilities for integrating 5G systems with real-time Ethernet and Time-Sensitive Networking used in industrial automation networks.

There are many challenges in cellular IoT networks, such as designing the networking and storage architecture for intelligent devices, efficient data communication protocols, proactive identification and protection of IoT devices from malicious attacks, standardization of technologies, and application interfaces [37]. Moreover, cellular IoT networks, e.g., 5G cellular networks, need to support extremely high data rates and require continuous connectivity to serve heterogeneous devices with varying quality of service requirements. Furthermore, proper channel and power allocation are crucial to support various IoT applications simultaneously.

The challenges and resource constraints mentioned above in cellular IoT networks motivate using ML techniques for resource management in these networks. For resource management problems, ML is an approach to developing low-complex mathematical models and considering resource allocations in several IoT applications of Figure 1.2. ML is a suitable solution for both the network and devices, which need to learn the context and diverse characteristics of human and machine users to achieve optimal system configurations. Moreover, intelligent devices are expected to be capable of autonomously accessing the available spectrum with the aid of learning and adaptively adjusting the transmission power to avoid interference and save energy.

Although traditional optimization and heuristics-based techniques are available; they may be computationally expensive to run on small, inexpensive, and energy-constrained devices. Furthermore, traditional game theoretical approaches may not be suitable due to the heterogeneity of the devices and the overhead due to the information exchange. Thus, ML techniques can alleviate the challenges of

traditional approaches and perform well in allocating resources for IoT devices.

1.2 Basic Definitions

In this section, we the basic definitions of optimization theory [38] we utilize in this thesis, such as convex, strongly convex, and $\bar{L}$-smooth functions. Additionally, we explain the smoothing technique, as explained in [39] and [40], to facilitate using advanced optimization methods in this thesis.

Convex Sets

Definition 1. *A set $\mathcal{C}$ is convex if the line segment between any two points in $\mathcal{C}$ lies in $\mathcal{C}$. Specifically, a set $\mathcal{C}$ is convex if for any $\boldsymbol{x_1}, \boldsymbol{x_2} \in \mathcal{C}$ and any $\theta \in [0,1]$, we have*

$$\theta \boldsymbol{x_1} + (1-\theta)\boldsymbol{x_2} \in \mathcal{C}. \tag{1.1}$$

Convex Functions

Definition 2. *Let us define **dom** f as the domain of a function $f : \mathbb{R}^d \to \mathbb{R}$. Then, a function f is convex if **dom** f is a convex set and for all $\boldsymbol{x_1}, \boldsymbol{x_2} \in$ **dom** f, and for any $\theta \in [0,1]$, the following inequality is satisfied:*

$$f(\theta \boldsymbol{x_1} + (1-\theta)\boldsymbol{x_2}) \leq \theta f(\boldsymbol{x_1}) + (1-\theta) f(\boldsymbol{x_2}). \tag{1.2}$$

Strongly Convex Functions

Definition 3. *A continuously differentiable function $f : \mathbb{R}^d \to \mathbb{R}$ with the corresponding gradient $\nabla f(\boldsymbol{x})$ is μ-strongly convex, for $\mu \geq 0$, if for all $\boldsymbol{x_1}, \boldsymbol{x_2} \in$ **dom** f, we have*

$$f(\boldsymbol{x_2}) \geq f(\boldsymbol{x_1}) + (\boldsymbol{x_2} - \boldsymbol{x_1})^\top \nabla f(\boldsymbol{x_1}) + \frac{\mu}{2}\|\boldsymbol{x_2} - \boldsymbol{x_1}\|_2^2. \tag{1.3}$$

Smooth Functions

Definition 4. *A continuously differentiable function $f : \mathbb{R}^d \to \mathbb{R}$ with the corresponding gradient $\nabla f(\boldsymbol{x})$ is $\bar{L}$-smooth, for $0 < \bar{L} < \infty$, if for all $\boldsymbol{x_1}, \boldsymbol{x_2} \in$ **dom** f, we have*

$$f(\boldsymbol{x_2}) \leq f(\boldsymbol{x_1}) + (\boldsymbol{x_2} - \boldsymbol{x_1})^\top \nabla f(\boldsymbol{x_1}) + \frac{\bar{L}}{2}\|\boldsymbol{x_2} - \boldsymbol{x_1}\|_2^2. \tag{1.4}$$

Equivalently, a continuously differentiable function $f : \mathbb{R}^d \to \mathbb{R}$ with the corresponding gradient $\nabla f(\boldsymbol{x})$ is $\bar{L}$-smooth if

$$\|\nabla f(\boldsymbol{x}_1) - \nabla f(\boldsymbol{x}_2)\|_2 \leq \bar{L}\|\boldsymbol{x}_1 - \boldsymbol{x}_2\|_2. \tag{1.5}$$

Big O Notation ($\mathcal{O}(.)$)

Definition 5. *Let f and g be functions $f, g : \mathbb{R} \to \mathbb{R}$. Then, $f(x) = \mathcal{O}(g(x))$ as $x \to \infty$ if there exists a positive real number $C \in \mathbb{R}^+$ and a real number $x_0 \in \mathbb{R}$ such that [41]*

$$|f(x)| \leq Cg(x), \quad x \geq x_0. \tag{1.6}$$

Convergence Rates for Iterative Methods

Definition 6. *Let us consider a sequence $\{a_n\}, n = 1, 2, \ldots,$ which converges to a^*, namely $\lim_{n\to\infty} a_n = a^*$. Then, the sequence of $\{a_n\}$ has the order of convergence $q \geq 1$ and rate of convergence p [42], if*

$$\lim_{n\to\infty} \frac{|a_{n+1} - a^*|}{|a_n - a^*|^q} = p. \tag{1.7}$$

Definition 6 represents the general definition of the convergence rate p and convergence order q of a sequence $\{a_n\}, n = 1, \ldots$. Accordingly, we investigate in the following the convergence terms and behaviors for different convergence orders q.

Definition 7. *Suppose that the sequence $\{a_n\}, n = 1, 2, \ldots,$ converges to a^* with order q and rate p, see Definition 6, and [42]. Thus,*

- *If $q = 1$ and $p \in (0, 1)$, the sequence $\{a_n\}, n = 1, \ldots$ **linearly** converges to a^*.*
- *If $q = 1$ and $p = 1$, $\{a_n\}, n = 1, \ldots$ **sublinearly** converges to a^*.*
- *If $q = 2$, $\{a_n\}, n = 1, \ldots$ has a **quadratic** convergence to a^*.*

Smoothing Technique

Given a function $G : \mathbb{R}^K \to \mathbb{R}$ with $\boldsymbol{x} \in \mathbf{dom}\ G$, the smoothing transformation approximates the objective function involving $|G(\boldsymbol{x})|$ with a continuously differentiable and smooth function. The smooth approximation, as detailed in [39] and [40], is expressed as:

$$\underset{\boldsymbol{x}}{\text{minimize}} \quad |G(\boldsymbol{x})| \simeq \underset{\boldsymbol{x}}{\text{minimize}} \quad \sqrt{G(\boldsymbol{x})^2 + \epsilon}, \tag{1.8}$$

where $0 < \epsilon \ll 1$ is a very small positive constant that prevents the denominator of the gradient of the smoothing function in (1.8) from becoming zero. This approximation is particularly useful for applying advanced optimization techniques to non-differentiable objective functions.

The definitions above are the basis of developing this thesis, which we mainly utilize in this thesis, particularly in Chapter 7 of the included papers.

1.3 Preliminaries of Machine Learning (ML)

In this section, we briefly introduce the ML concept in simple words. Then, we present a general challenge in ML and explain how this thesis aims to address it. Chapter 2 of this thesis provides a comprehensive description of ML.

What is ML?

ML is a branch of artificial intelligence that enables computers to learn and make decisions without being explicitly programmed. Instead of following predefined instructions, ML systems use data and algorithms to identify patterns and improve their performance over time. This transformative technology is driving innovations in diverse areas such as healthcare, finance, transportation, and everyday applications like voice assistants and recommendation systems. At its core, ML seeks to mimic human-like decision-making, making it a cornerstone of modern intelligent systems.

General Overview of FL

Federated Learning (FL) is a decentralized approach to training ML models, where data remains on local devices, and only updates (such as model gradients) are shared with a central server. This approach enhances privacy by avoiding the need to transfer raw data, making it especially valuable in sensitive domains like healthcare and finance. Additionally, FL reduces the overhead associated with data centralization, such as storage and legal concerns.

However, FL faces significant challenges. Large-scale deployments with numerous devices and high-dimensional models can lead to substantial communication overhead as transmitting updates across a distributed network becomes expensive and time-consuming. Furthermore, the training process is often limited by the latency introduced by the slowest participating user, as the central server must wait for all users to transmit their local updates before proceeding with the global aggregation. This phenomenon, known as the *straggler effect*, increases training delays and highlights the importance of mitigating its impact to optimize the overall FL performance. These challenges necessitate the development of efficient optimization methods to ensure that FL systems are both scalable and effective in real-world applications.

ML over Cellular/Cell-free Networks

ML over networks involves leveraging wireless communication systems to enable distributed learning. In this approach, devices collaborate to train models while sharing updates over the network. This approach is particularly useful in scenarios where data is generated at the edge, such as in mobile devices or IoT sensors. By integrating ML with wireless networks, we can support intelligent applications

like autonomous vehicles, smart cities, and personalized healthcare. Wireless connectivity plays a crucial role in enabling these applications by offering mobility support, flexible deployment, and scalability that wired networks cannot achieve. For example, in smart cities, wireless networks facilitate seamless communication between mobile devices, IoT sensors, and centralized systems, enabling real-time decision-making and adaptability. Similarly, personalized healthcare benefits from wireless communication through remote monitoring services, allowing healthcare professionals to access patient data in real-time without the constraints of physical connections. However, transitioning from wired to wireless communication presents some challenges, such as increased latency, vulnerability to interference, and channel variability. Overcoming these issues is critical to ensuring the reliability and efficiency of intelligent applications operating over wireless networks.

In this thesis, we explore several networking protocols to facilitate efficient communication for ML tasks:

- **Slotted-ALOHA:** A simple and decentralized protocol where devices transmit updates in predefined time slots, reducing collisions but potentially underutilizing resources.

- **CSMA/CA (Carrier Sense Multiple Access with Collision Avoidance):** A protocol that allows devices to sense the channel before transmitting, helping to minimize interference but introducing delays when the network is busy.

- **OFDMA (Orthogonal Frequency Division Multiple Access):** A highly efficient method that allocates different frequency bands to users, ensuring simultaneous transmissions and excellent resource utilization, albeit at the cost of increased complexity.

- **Cell-Free Massive MIMO (CFmMIMO):** A cutting-edge approach that employs a distributed array of antennas to provide seamless coverage, thereby enhancing reliability and throughput. Unlike OFDMA, where users are allocated distinct subcarriers within a predefined frequency band, CFmMIMO allows each user to access the entire frequency spectrum. In CFmMIMO, interference is effectively managed through advanced antenna processing techniques rather than through frequency division, as in OFDMA. This capability to manage interference spatially, coupled with the use of distributed antennas, results in significant throughput improvements compared to traditional OFDMA systems.

Each protocol has unique advantages and drawbacks, and their selection depends on the specific requirements of the ML application, such as latency, scalability, and communication overhead. By analyzing and applying these protocols, this thesis aims to advance the integration of ML into modern wireless networks.

Challenges and Bottlenecks

ML and FL face several challenges and bottlenecks that limit their efficiency and scalability, particularly in large-scale and resource-constrained environments. The primary challenge is the high communication overhead, especially during large-scale training. Transmitting model updates or gradients across a network of distributed devices consumes significant bandwidth and can cause delays, particularly with high-dimensional models and large numbers of participants. Addressing this bottleneck is a core focus of this thesis.

Another challenge is the difficulty of designing efficient FL training without knowing the future training information, such as gradients and iteration costs, which requires careful optimization to ensure consistent performance in real-world deployments.

Causal vs. Non-causal Approach

To address the challenge of designing cost-efficient FL over wireless networks, this thesis introduces a novel causal approach and compares it with the non-causal methods. The proposed causal solution focuses on optimizing the termination iteration, i.e., determining the total number of iterations required to achieve communication-computation efficient training. Considering the iterative nature of FL and aligning training dynamics with real-time constraints, the causal approach ensures that resources are utilized efficiently without compromising model performance. This contrasts with non-causal methods, which assume prior knowledge of training behavior and can lead to suboptimal resource usage in dynamic environments. Through this comparison, the thesis demonstrates the effectiveness of the causal approach in reducing overhead while maintaining efficient FL training over wireless networks.

1.4 Research Gaps (RGs)

In this section, we outline several broad and high-level Research Gaps (RGs) in the FL literature. The proposed RGs are as follows:

RG1: A key gap in the current literature is the failure to incorporate the communication and computation costs of FL training into the optimization objectives alongside the FL loss function.

RG2: Existing works typically do not consider a causal solution approach for determining the termination iteration in FL training. Instead, they often assign a pre-set number of iterations at the start of the training. However, this pre-set number may not be optimally chosen, as it does not account for resource limitations or the varying costs associated with each iteration. An alternative would be to run the algorithm until a convergence condition is satisfied, which might require unreasonably much computation and communication resources.

RG3: Furthermore, the current literature does not consider the impact of the communication protocol parameters, such as transmission probability, packet arrival probability, and the background traffic in users' queues, on the convergence of FL training.

RG4: In the CFmMIMO literature related to FL, existing studies have not explored the joint optimization of communication energy and latency. This co-optimization is critical as energy and latency exhibit opposing behavior with respect to uplink power levels. Addressing this trade-off becomes particularly important under energy- and latency-constrained scenarios. Additionally, the effect of interference from other users' power on each user's energy and latency remains unexamined.

RG5: While numerous studies in the communication-efficient FL literature have proposed adaptive quantization and compression schemes for gradient transmission, there remains significant potential for further enhancing communication efficiency, particularly in large-scale FL training scenarios. Some of the key gaps in this area that require further exploration include the following:

- The existing works mostly apply the same quantization resolution to the entire local gradient vector and do not exploit the sparsity of local gradients [43], often resulting in many near-zero elements. Many sparse near-zero elements in the local gradients require proper quantization schemes to reduce the communication overhead.
- Fixed bit-assignment approaches, such as LAQ [44], do not account for the diminishing returns that affect convergence. As a result, fixed-bit assignment is not an optimal strategy, leading to inefficient energy usage during the training process.

Accordingly, this thesis aims to address these RGs by focusing on the associated research questions, which will be introduced in the following section.

1.5 Research Objective and Questions (RQs)

The central research objective of this thesis is to optimally determine the number of global iterations required for distributed training, such as FL, without relying on future training information. This objective accounts for the impact of communication protocols and training costs, including communication latency, computation latency, and energy consumption. To achieve this goal, the thesis poses key research questions and systematically addresses them. Papers 1, 2, and 3 specifically focus on determining the optimal stopping iterations, while Papers 4, 5, and 6 explore power allocation strategies in the context of enhancing energy efficiency and reducing the training latency by mitigating the straggler effect. Together, these contributions provide a comprehensive framework for establishing optimal causal

stopping criteria for distributed learning and FL training. Accordingly, the primary high-level RQs of this thesis are as follows:

RQ1: **What are the impacts of communication protocols and background traffic on the causal termination iteration of a distributed training?**

This RQ aims to examine the impact of communication protocol parameters, such as the probability of packet arrival and transmission probability, on the causal stopping iteration of FL training, with a particular focus on the resulting communication cost during training. To address this impact, **Papers 1 and 2** present both theoretical and experimental analyses of how parameters in communication protocols, including Slotted ALOHA, CSMA/CA, and OFDMA influence communication latency. This latency arises primarily due to the queue lengths of the FL users. Additionally, background traffic in the users' queues introduces further delays in transmitting local FL updates or gradients to the server. Furthermore, our investigation considers the effects of these parameters in two scenarios: when the local models are head-of-line packets in the queue and when they are not. Based on these insights, we propose a congestion control algorithm designed to mitigate the uncontrollable effects of communication protocol parameters, thereby reducing the communication cost associated with the training.

RQ2: **How can we adapt and optimize the physical layer of a cell-free massive MIMO network to mitigate the straggler effect and enhance communication energy efficiency in FL training?**

This RQ investigates the adaptation of the physical layer in CFmMIMO networks to improve energy efficiency and mitigate the high latency caused by straggler users in FL training. In large-scale FL systems, the limited power resources of users can lead to significant communication delays, known as the straggler effect. This latency is critical as the server must wait for all users to complete their transmissions before updating the global model. Consequently, delays caused by straggler users can exceed the allocated time budget, potentially hindering the completion of the training process. Simultaneously, the limited energy resources of users present an additional challenge in FL training. Optimizing the physical layer becomes crucial to address both the straggler effect and the energy constraints. To this end, **Papers 4, 5, and 6** focus on uplink power control strategies, demonstrating how physical layer optimizations can effectively reduce communication latency and energy consumption, thereby improving the overall efficiency of FL training.

RQ3: **How can the quantization scheme be designed to enhance communication efficiency and model accuracy in FL?**

This RQ examines the influence of advanced quantization techniques on communication efficiency and model accuracy in FL. While fixed bit-assignment methods like LAQ [3] provide a baseline, adaptive quantization schemes have

demonstrated significant potential in reducing unnecessary communication overhead without compromising test accuracy compared to standard FL with the full-precision transmission. To address this RQ, **Papers 3, 5, and 6** explore various quantization strategies, including A-LAQ and other adaptive approaches, to assess their effectiveness in minimizing communication costs while maintaining model accuracy. These studies highlight the critical role of adaptive quantization in enhancing the communication efficiency of FL systems.

RQ4: **What are the impacts of computation costs on the overall performance of FL systems**?

While the primary focus of this thesis is on communication efficiency, it also addresses the computation costs incurred by FL users. These costs, including latency and energy consumption, are influenced by factors such as the size of local datasets, the number of local iterations, device processing capabilities, and the required number of processing cycles. This RQ explores how computation costs impact FL training. **Papers 2, 4, and 5** analyze the effects of these costs on training efficiency, model convergence, and resource allocation. These investigations provide valuable insights into the role of computation costs in shaping the overall performance and feasibility of FL training systems.

1.6 Thesis Contributions

In this section, we briefly summarize the contributions of each included paper in this thesis.

Paper 1 - Machine learning over networks: co-design of distributed optimization and communications

Afsaneh Mahmoudi, Hossein S. Ghadikolaei, Carlo Fischione

IEEE 21st International Workshop on Signal Processing Advances in Wireless Communications (SPAWC), 2020.

Summary

The contention level of the communication channels may significantly impact the performance of the distributed algorithms over networks. In this paper, given a total latency budget to run the ML algorithm, we show that the training performance worsens as the background traffic or the dimension of the training problem increases. We address this impact by focusing on the CSMA/CA protocol as the primary communication protocol in this paper. Inspired by the transport control

protocol [45], we consider minimizing a multi-objective function of ML loss and iteration cost functions where a dynamic choice of the background traffic arrival rate controls the queue congestion level. The main goal of solving such an optimization problem is to determine iteration-termination criteria to investigate the trade-off between achievable training performance and the overall communication cost. The other significant contribution of the paper is utilizing the *causal* approach, which is the fundamental concept of all the papers in this thesis. We recall that a method is causal if that method does not require the future information of the system to take action at the current time. On the other hand, a non-causal method needs to know all the future information to solve the problems in the system. Finally, we perform an experimental analysis of the causal mini-batch algorithm over the MNIST dataset. We conclude that ML over wireless networks may fail without the joint design of distributed optimization algorithms and communication protocols.

Contribution

The author of this thesis and the second author conceptualized the original idea for the paper. The author of this thesis performed the mathematical modeling and derivations, wrote the code and numerical experiments, and wrote the text for the article. In addition, the second and third authors provided expertise in the form of discussions on the system model and highlighting research directions. The second and third authors also proofread and assisted in structuring the text.

Paper 2 - FedCau: A Proactive Stop Policy for Communication and Computation Efficient Federated Learning

Afsaneh Mahmoudi, Hossein S. Ghadikolaei, José Mairton Barros Da Silva Júnior, Carlo Fischione

IEEE Transactions on Wireless Communications, vol. 23, no. 9, pp. 11076-11093, Sept. 2024, doi: 10.1109/TWC.2024.3378351.

Summary

Most papers in the literature aim to train the FedAvg algorithm in resource-constraint conditions and propose the best resource allocation policies before performing the training [19, 46]. These approaches rely mainly on approximating the future training information by using some lower and upper bounds of that information. This paper proposes to train the FedAvg algorithm in a causal, communication, and computation-efficient way. To this end, we utilize the well-known multi-objective optimization approach according to the scalarization procedure in [38]. Therefore, we propose FedCau to improve the FedAvg algorithm by training cost-efficiently without needing to know the future training information or any upper and lower bounds. Thus, we aim to solve an optimization problem consisting of

minimizing a multi-objective function of the FL loss function and iteration-cost function. We analyze the trade-off between the achievable test accuracy in FL training and the iteration cost defined as the overall communication-computation cost of running FL over a wireless network. In this paper, we substantially extended the idea of Paper 1 to include all the possible behaviors for an FL loss function, e.g., both convex and non-convex loss functions. We have also considered the FedAvg algorithm and performance analysis of FedCau by simulation over both MNIST and CIFAR-10 datasets. We also extended the causal approach of Paper 1 by adding a new algorithm for non-convex and non-decreasing loss functions. Moreover, we have investigated partial client participation and its effect on FedCau, deriving upper and lower bounds of the iteration-cost function over different wireless communication protocols, such as OFDMA, slotted-ALOHA, and CSMA/CA. The numerical results highlight the importance of proactively designing the stopping criteria of FL to eliminate unnecessary resource expenditure.

Contribution

The author of this thesis and the third author conceptualized the original idea for the paper. The author of this thesis performed the mathematical modeling and derivations, wrote the simulation codes and numerical experiments, and wrote the text for the article. Given the contribution of this work to the application of multiple communication protocols, considerable effort was required to ensure compatibility with existing literature and to enhance comprehensibility among the wireless communication community. Consequently, the third and fourth authors provided significant help in discussions on the system model and structuring the paper. In addition, the second author contributed to the literature review and some arguments regarding wireless communication applications.

Paper 3 - A-LAQ: Adaptive Lazily Aggregated Quantized Gradient

Afsaneh Mahmoudi, José Mairton Barros Da Silva Júnior, Hossein S. Ghadikolaei, Carlo Fischione

IEEE Globecom Workshops (GC Wkshps), 2022.

Summary

In this paper, we aim to further improve communication efficiency in LAQ [3] by using our causal methods. We extend LAQ by considering an adaptive number of bits during the FL training. The key idea in this paper is the diminishing return rule [47], which highlights the critical effect of the first iterations on the convergence of ML algorithms. According to the diminishing rule, the improvement in the loss function of the training reduces after each iteration. Therefore, allocating more

resources to the first iterations of FL training results in faster convergence and better accuracy than the random association of resources. In this paper, we consider an energy-constraint FL problem and allocate an adaptive number of bits for the training of the FL algorithm. Our numerical results show that by assigning a non-increasing number of bits against iterations, we achieve up to 11% increase in test accuracy while reducing up to 50% energy consumption.

Contribution

The author of this thesis conceptualized the original idea for the paper, performed the mathematical modeling and derivations, wrote the code and numerical experiments, and wrote the text for the article. In addition, the second and fourth authors provided expertise in the form of discussions on the system model and highlighting research directions. The second and fourth authors also proofread and assisted in structuring the text, and the third author assisted in getting the project's funding for the paper.

Paper 4 - Low-Latency and Energy-Efficient Federated Learning over Cell-Free Networks: A Trade-off Analysis

Afsaneh Mahmoudi, Mahmoud Zaher, Emil Björnson

Submitted to IEEE Open Journal of Communications Society, 2025.

Summary

Cell-free massive multiple-input multiple-output (CFmMIMO) is an emerging network architecture that supports numerous users on shared time and frequency resources, making it well-suited for large-scale federated learning (FL). While CFmMIMO improves energy efficiency via spatial multiplexing and collaborative beamforming, adapting its physical layer operations to allocate uplink transmission power among FL users is essential. This paper introduces two power allocation schemes to maximize the number of FL iterations achievable under strict energy and latency constraints. These schemes account for the impact of each user's transmit power on others' energy consumption and latency, with the goal of jointly minimizing uplink energy use and FL training latency. The first scheme minimizes a weighted sum of users' uplink energy and latency to illustrate the trade-off between these factors, while the second scheme minimizes the gap between the achievable number of global iterations under limited energy and latency budgets. Numerical results demonstrate that in constrained total energy and latency situations, our proposed approaches achieved up to a 62% increase in global iterations compared to the state-of-the-art Dinkelbach method [48] and 93% compared to the max-sum rate method [2], highlighting the superiority of our approaches for low-latency, energy-efficient, and FL training in CFmMIMO networks.

Contribution

The author of this thesis conceptualized the original idea for the paper, performed the mathematical modeling and derivations, wrote the FL simulation codes and numerical experiments, and wrote the text for the article. The second author contributed specific code for CFmMIMO simulations, while both the second and third authors participated in discussions on the system model. Additionally, the third author provided high-level revisions and feedback on the article.

Paper 5 - Adaptive Quantization Resolution and Power Control for Federated Learning over Cell-free Networks

Afsaneh Mahmoudi, Emil Björnson

IEEE Globecom Workshops (GC Wkshps), 2024.

Summary

In this paper, we co-optimize the physical layer with the FL application to address the straggler effect, introducing an adaptive mixed-resolution quantization scheme for local gradient updates that allocates high resolution only to essential entries. We then propose a dynamic uplink power control scheme to handle variable user rates and further mitigate the straggler effect. Numerical results show that our method maintains test accuracy comparable to classic FL while reducing communication overhead by at least 93 % on the CIFAR-10, CIFAR-100, and Fashion-MNIST datasets. Compared to AQUILA [49], Top-q [50], and LAQ [3] benchmarks, along with max-sum rate [2] and Dinkelbach max-min energy efficiency algorithms [48] schemes, our approach reduces communication overhead by 75 % and achieves up to 10 % higher test accuracy within a constrained latency budget.

Contribution

The author of this thesis developed the original concept for the paper, conducted the mathematical modeling and derivations, created the simulation codes, performed the numerical experiments, and wrote the article. The second author contributed by discussing the system model and providing high-level revisions and feedback on the article.

Paper 6 - Accelerating Energy-Efficient Federated Learning in Cell-Free Networks with Adaptive Quantization

Afsaneh Mahmoudi, Ming Xiao, Emil Björnson

Submitted to IEEE Transactions on Machine Learning in Communications and Networking, 2024.

Summary

This paper proposes an energy-efficient, low-latency FL framework featuring optimized uplink power allocation for seamless user-server collaboration. Our framework employs an adaptive quantization scheme, dynamically adjusting bit allocation for local gradient updates to reduce communication costs. We formulate a joint optimization problem covering FL model updates, local iterations, and power allocation, solved using sequential quadratic programming (SQP) to balance energy and latency. Additionally, users use the AdaDelta method for local FL model updates, enhancing local model convergence compared to standard SGD, and we provide a comprehensive analysis of FL convergence with AdaDelta local updates. The numerical results show that FL with AdaDelta updates significantly outperforms FL + SGD updates in terms of convergence speed. Moreover, the adaptive quantization scheme and the dynamic number of local iterations greatly reduce the total computation and communication cost while achieving comparable test accuracy to applying the maximum local iterations in standard FL. Finally, the power allocation scheme reduces both energy and latency per iteration, resulting in a higher number of global iterations in constrained latency and energy budget conditions compared to max-sum rate [2] and Dinkelbach max-min energy efficiency algorithms [48] schemes. Numerical results show that, within the same energy and latency budgets, our power allocation scheme outperforms the Dinkelbach and max-sum rate methods by increasing the test accuracy up to 7% and 19%, respectively. Moreover, for the three power allocation methods, our proposed quantization scheme outperforms AQUILA and LAQ by increasing test accuracy by up to 36% and 35%, respectively.

Contribution

The author of this thesis developed the original concept for the paper, conducted the mathematical modeling and derivations, created the simulation codes, performed the numerical experiments, and wrote the article. The second and third authors contributed to discussions, offering valuable insights on the system model and problem formulation, as well as providing contextual comments. Additionally, the third author conducted high-level revisions and offered further feedback on the article.

1.7 Thesis Organization

This thesis is organized into **seven** chapters as follows:

- **Chapter 1** serves as the introduction to this thesis, offering a comprehensive exploration of edge computing and IoT, as well as an overview of the fundamental mathematical theories utilized throughout this thesis. Additionally, it provides a broad perspective on ML over wireless networks. The chapter identifies key research gaps, outlines the research objectives and questions, and highlights the main contributions of the thesis.

- **Chapter 2** presents a detailed study of ML and its core components, including neural networks, gradient descent, stochastic gradient descent, distributed ML, and FL.
- **Chapter 3** provides an in-depth explanation of cell-free massive MIMO networks. It includes a general overview, and uplink processes for transmitting FL models, descriptions of the power control schemes we used as benchmarks, and a discussion of the potential downlink scenarios for FL training.
- **Chapter 4** outlines the optimization methods employed in this thesis, including the Bisection method, Brent's method, linear programming, and sequential quadratic programming.
- **Chapter 5** presents a comprehensive literature review of FL over wireless networks and related works. In addition, it outlines the general system model and problem formulation for this thesis. Furthermore, it details the proposed contributions, including the causal setting, innovative quantization techniques, power control strategies, and approaches for jointly minimizing energy consumption and mitigating the straggler effect.
- **Chapter 6** provides the concluding remarks of this thesis, summarizing the key findings and contributions. It also discusses potential future work and research directions that can be built upon the outcomes of this study.
- **Chapter 7** contains the papers included in this thesis, showcasing the research contributions and findings in detail.

Chapter 2

Background

Utilizing ML as the central focus of this thesis, this chapter explores its objectives, including the mathematical formulation of supervised learning problems and the methods employed to train ML models. In the following sections, we present the fundamentals of supervised ML methods, including artificial neural networks (ANN), key training techniques such as gradient descent (GD) and stochastic gradient descent (SGD), and, finally, distributed ML approaches, such as FL.

2.1 Machine Learning (ML)

ML focuses on enabling computers to learn from a set of data, called *training data*, which means teaching computers how to extract required features for prediction tasks and make inferences from that data. ML algorithms utilize mathematical models based on the available training data to make predictions or decisions. The mathematical models in ML algorithms enable the systems to adapt automatically and improve over time by accumulating new data and training over it.

The input of the learning algorithms is training data, denoted herein by $(\boldsymbol{x}, y)$, where $\boldsymbol{x} \in \mathbb{R}^d$ is the data sample vector with dimension d, and $y_i \in \mathbb{R}$ is the training label corresponding to $\boldsymbol{x}$. Let us define $\mathcal{X}$ as the set of training data samples and $\mathcal{Y}$ as the set of their corresponding labels. Then, the output of a learning algorithm is a hypothesis function $h : \mathcal{X} \to \mathcal{Y}$ from an input space $\mathcal{X}$ to an output space $\mathcal{Y}$, that aims at predicting $y \in \mathcal{Y}$ for arbitrary $\boldsymbol{x} \in \mathcal{X}$. The learning algorithm aims to find a suitable hypothesis h w.r.t. a loss function defined as $f : \mathcal{Y} \times \mathcal{Y} \to \mathcal{Y}$ that measures the error between the hypothesis $h(\boldsymbol{x})$ and the true output y [16].

Instead of optimizing over a generic family of hypothesis functions, it is common to assume that the hypothesis function $h(.)$ has a fixed form and is parameterized by a real vector parameter $\boldsymbol{w} \in \mathbb{R}^d$ as the ML parameter in the parameterized hypothesis $h(\boldsymbol{x}, \boldsymbol{w})$. Afterward, the training process aims to find the optimal value of the learning parameter $\boldsymbol{w}^*$ by minimizing the loss function $f(h(\boldsymbol{x}, \boldsymbol{w}), y)$. The

following optimization problem finds $\boldsymbol{w}^*$ as

$$\boldsymbol{w}^* \in \arg\min_{\boldsymbol{w}} f(h(\boldsymbol{x}, \boldsymbol{w}), y). \tag{2.1}$$

Generally, the training data consists of a finite set of the tuples $(\boldsymbol{x}, y)$ defined as $\mathcal{D} = \{(\boldsymbol{x}_i, y_i)\}_{i=1}^N$. Thus, considering the average of loss function over the dataset $\mathcal{D}$, we re-write optimization problem (2.1) as

$$\boldsymbol{w}^* \in \arg\min_{\boldsymbol{w}} \frac{1}{N} \sum_{i=1}^{N} f(h(\boldsymbol{x}_i, \boldsymbol{w}), y_i). \tag{2.2}$$

Generally, we try to find the hypothesis $h(.)$ that minimizes our learning/prediction error. In practice, the training error is evaluated by solving (2.2) with N data samples. The test error is assessed by comparing the hypothesis function $h(\boldsymbol{x}_j, \boldsymbol{w})$ to predict the corresponding output $y_j \in \mathcal{Y}$, where $\boldsymbol{x}_j \in \mathcal{X}$ has not attributed to the training. Specifically, classification accuracy is the ratio between the number of correct and incorrect predictions the learning model gives. Thus, it is critical to choose the proper hypothesis, and in the following section, we elaborate on a prominent example of hypothesis functions $h(.)$.

2.2 Artificial Neural Networks (ANNs)

Artificial Neural networks (ANNs) are prominent instances of the hypothesis function $h(\cdot)$ introduced in the previous section. Inspired by interconnected neurons in biological systems, ANNs *learn* from examples (as children learn to recognize cats from examples of cats) and exhibit some capability for generalization beyond the training data. ANNs have several benefits in terms of pattern recognition, learning, classification, generalization and abstraction, and interpretation of incomplete and noisy inputs.

ANNs consist of layers, neurons, activation functions, and weights. Generally, we can categorize ANNs into three types: single layer, multiple layers, and deep layers defined as deep NNs (DNNs). Figure 2.1 depicts a simple example of a single-layer ANN showing the input layer, one hidden layer, and the output layer. The hidden layer is composed of d_1 neurons, the input data is represented as $\boldsymbol{x} \in \mathbb{R}^d$, where

$$x_i := [\boldsymbol{x}]_i, \quad i = 1, \ldots, d, \tag{2.3}$$

and $\boldsymbol{W}_1 \in \mathbb{R}^{d \times d_1}$ are weights of the hidden layer. We define $\sigma(.) : \mathbb{R}^{d_1} \to \mathbb{R}^{d_1}$ as the activation function, and the operation at the hidden layer is performed as

$$\boldsymbol{r}_1 = \sigma\left(\boldsymbol{W}_1^\top \boldsymbol{r}_0 + \boldsymbol{b}_1\right), \tag{2.4}$$

where $\boldsymbol{b}_1 \in \mathbb{R}^{d_1}$ is the bias, and $\boldsymbol{r}_0 \in \mathbb{R}^d$, with each element of

$$r_0^{(i)} := [\boldsymbol{r}_0]_i = [\boldsymbol{x}]_i, \quad i = 1, \ldots, d. \tag{2.5}$$

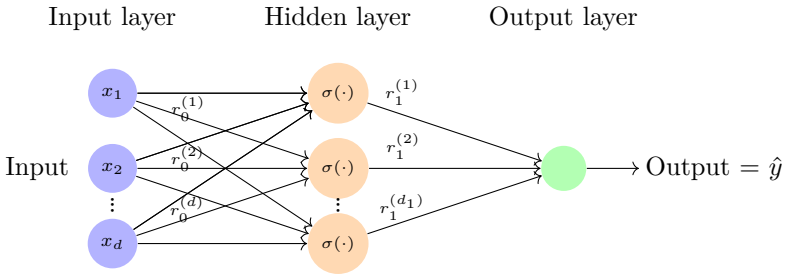


Figure 2.1: Illustration of an example of a single layer ANNs.

Similar to (2.5), we define $r_1^{(i)}$, as the element i of the vector $\boldsymbol{r}_1 \in \mathbb{R}^{d_1}$.

Finally, the output of the ANN is defined as

$$\hat{y} := h(\boldsymbol{w}, \boldsymbol{x}) = \sigma\left(\boldsymbol{W}_o^\top \boldsymbol{r}_1 + b_o\right), \tag{2.6}$$

where $\boldsymbol{W}_o \in \mathbb{R}^{d_1}$ are weights of the output layer, $b_o \in \mathbb{R}$ is the bias, and $\boldsymbol{w} := \{\boldsymbol{W}_1, \boldsymbol{b}_1, \boldsymbol{W}_o, b_o\}$ are the trainable parameters.

In recent years, the availability of massive datasets and massive computing power have enabled DNNs to demonstrate remarkable performance across a range of machine learning tasks, including computer vision, information retrieval, and language processing [51]. Different from the single layer ANNs, DNNs consist of several hidden layers $l = 1, \ldots, L_N$, where $L_N > 2$, which allows the ANNs to learn hierarchical feature abstractions of the data, with increasing abstraction the deeper you go in the network [52]. Each layer l of DNNs is composed of d_l neurons and $\boldsymbol{W_l} \in \mathbb{R}^{d_{l-1} \times d_l}$ are weights of the hidden layer l as shown in Figure 2.2. Similar to the single layer ANNs, the input of the system is $\boldsymbol{x} \in \mathbb{R}^d$, where we defined its elements in (2.3), and the activation function is $\sigma(.)$ and the operation at the hidden layer l is

$$\boldsymbol{r}_l = \sigma\left(\boldsymbol{W}_l^\top \boldsymbol{r}_{l-1} + \boldsymbol{b}_l\right), \quad l = 1, \ldots, L_N. \tag{2.7}$$

The output of the DNN layer is

$$\hat{\boldsymbol{y}} := h(\boldsymbol{w}, \boldsymbol{x}) = \sigma_o\left(\boldsymbol{W}_o^\top \boldsymbol{r}_{L_N} + \boldsymbol{b}_o\right), \tag{2.8}$$

where $\sigma_o(.)$ is the activation function of the output layer, and the trainable parameters are defined as $\boldsymbol{w} := \{\boldsymbol{W}_1, \boldsymbol{b}_1, \ldots, \boldsymbol{W}_L, \boldsymbol{b}_{L_N}, \boldsymbol{W}_o, \boldsymbol{b}_o\}$. Consequently, the goal is to obtain the parameter $\boldsymbol{w}$.

2.3 Convolutional Neural Networks (CNNs)

Convolutional Neural Networks (CNNs) are a specialized category of ANNs designed specifically for processing data with grid-like structures, such as images and

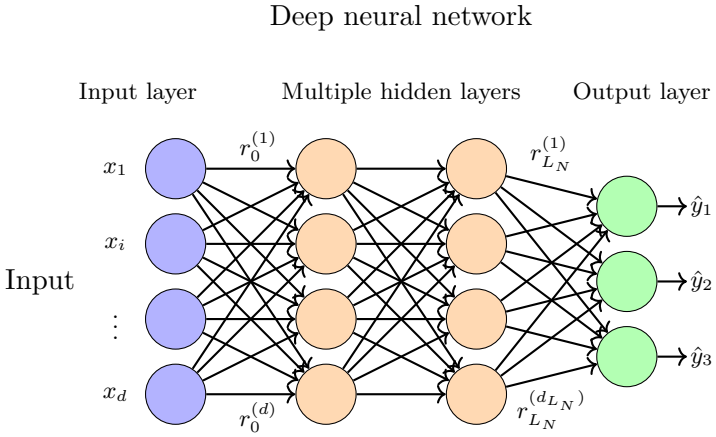


Figure 2.2: Illustration of an example of DNNs.

time-series data. CNNs excel in tasks requiring spatial feature extraction, such as image recognition, object detection, and semantic segmentation, by leveraging their unique architecture, which incorporates convolutional layers. CNNs' ability to learn hierarchical spatial features has made them a cornerstone of deep learning architectures, enabling breakthroughs in applications like medical imaging, autonomous driving, and natural language processing [53]. The combination of convolutional operations, activation functions, and pooling allows CNNs to extract features invariant to transformations, making them highly effective for a variety of tasks [54].

Figure 2.3 shows a simple example of the CNN structure. A CNN typically consists of three main types of layers: convolutional layers, pooling layers, and fully connected layers. Unlike traditional dense layers in ANNs and DNNs, the convolutional layers operate on local regions of the input data, effectively capturing spatial relationships. A convolutional layer performs the following operation:

$$\boldsymbol{r}_l = \sigma\left(\boldsymbol{W}_l * \boldsymbol{r}_{l-1} + \boldsymbol{b}_l\right), \quad l = 1, \ldots, L_N, \tag{2.9}$$

where $\boldsymbol{W}_l * \boldsymbol{r}_{l-1}$ denotes the convolution operation between $\boldsymbol{W}_l$ and $\boldsymbol{r}_{l-1}$. The input feature map is represented as $\boldsymbol{r}_{l-1} \in \mathbb{R}^{d_h \times d_w \times d_{l-1}}$ where d_h and d_w are the spatial dimensions of the feature map and d_{l-1} is the depth (number of channels) of the input feature map. The convolutional kernel, $\boldsymbol{W}_l \in \mathbb{R}^{\bar{k} \times \bar{k} \times d_w \times d_{l-1}}$ where $\bar{k}$ the spatial size of the kernel, and the bias term $\boldsymbol{b}_l \in \mathbb{R}^{d_l}$ is added to each output channel.

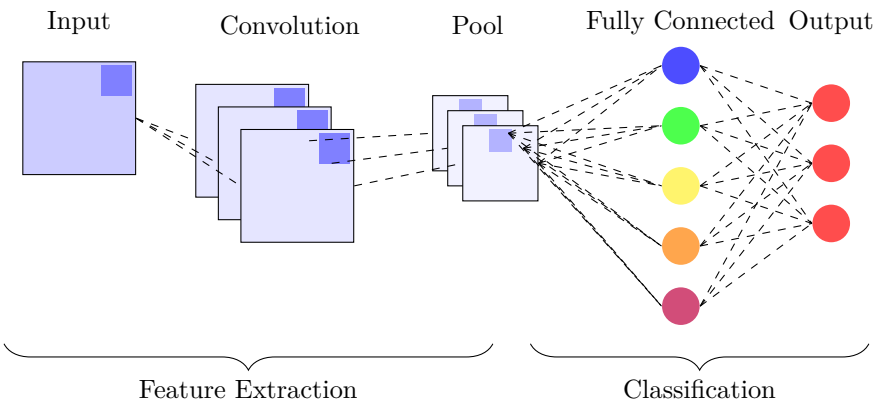


Figure 2.3: Illustration of an example of CNNs.

The pooling layers, interspersed between convolutional layers, reduce the spatial dimensions of the feature maps, improving computational efficiency and robustness to spatial transformations. Pooling can be performed using operations like max-pooling or average-pooling, where the ith element of the output is:

$$\boldsymbol{r}_{i,p} := \text{pool}\left(\boldsymbol{r}_j\right), \quad j \in \mathcal{N}_i, \tag{2.10}$$

where $\mathcal{N}_i$ represents the set of indices within the pooling region. Finally, the fully connected layers at the end of the network, similar to those in DNNs, map the extracted features into the desired output space, such as class probabilities for classification tasks.

2.4 Gradient Descent (GD)

Gradient descent (GD) is a commonly employed optimization algorithm to minimize functions, including the loss functions from ML models and ANNs. GD is the most basic approach to optimize ANNs, as every state-of-the-art ML library contains implementations of GD. The GD algorithm utilizes the training data to iteratively update the model's parameters until the cost function, which measures the model's loss function, is minimized. This iterative process continues until we obtain the minimum possible error [55].

The GD algorithm aims to minimize the parameterized loss function $f(h(\boldsymbol{x}, \boldsymbol{w}), y)$ with the model parameter $\boldsymbol{w} \in \mathbb{R}^d$ by updating the parameters in the opposite direction of the gradient of $f(h(\boldsymbol{x}, \boldsymbol{w}), y)$, denoted as $\nabla_{\boldsymbol{w}} f(h(\boldsymbol{x}, \boldsymbol{w}), y)$. We recall (2.2) which considers an optimization problem performed over a dataset of $\mathcal{D} = \{(\boldsymbol{x}_i, y_i)\}_{i=1}^N$. Then, initializing the GD algorithm by an arbitrary choice of

$\boldsymbol{w}_0$, the GD update at each iteration $k = 1, \ldots, K$ is

$$\boldsymbol{w}_k = \boldsymbol{w}_{k-1} - \frac{\alpha}{N} \sum_{i=1}^{N} \nabla_{\boldsymbol{w}} f(h(\boldsymbol{x}_i, \boldsymbol{w}_{k-1}), y_i), \quad k = 1, \ldots, K, \tag{2.11}$$

where $\alpha \in (0, 1)$ is the step size of the GD updates, and the gradient operation is done over all the samples from the dataset $\mathcal{D}$. For simplicity, we define

$$f(\boldsymbol{w}_k) := \frac{1}{N} \sum_{i=1}^{N} f(h(\boldsymbol{x}_i, \boldsymbol{w}_k), y_i), \tag{2.12}$$

which we will use in this thesis and refer to the GD method as full gradient descent.

One of the important characteristics of GD is the *descent behavior* of the sequence of a differentiable loss function $f(\boldsymbol{w})$, as $f(\boldsymbol{w}_K) < f(\boldsymbol{w}_{K-1}) < \ldots < f(\boldsymbol{w}_1)$. We can prove that for any differentiable loss function $f(\boldsymbol{w})$, the descent behavior of the GD loss function sequence is obtained by applying first-order approximation as

$$f(\boldsymbol{w}_k) = f\left(\boldsymbol{w}_{k-1} - \alpha \nabla_{\boldsymbol{w}} f(\boldsymbol{w}_{k-1})\right) \approx f(\boldsymbol{w}_{k-1}) - \alpha \nabla_{\boldsymbol{w}} f(\boldsymbol{w}_{k-1}) \nabla_{\boldsymbol{w}} f(\boldsymbol{w}_{k-1})^\top, \tag{2.13}$$

where $\nabla_{\boldsymbol{w}} f(\boldsymbol{w}_{k-1}) \nabla_{\boldsymbol{w}} f(\boldsymbol{w}_{k-1})^\top = \|\nabla_{\boldsymbol{w}} f(\boldsymbol{w}_{k-1})\|^2 > 0$, and thus (2.13) is simplified as

$$f(\boldsymbol{w}_k) = f(\boldsymbol{w}_{k-1}) - \alpha \|\nabla_{\boldsymbol{w}} f(\boldsymbol{w}_{k-1})\|^2, \tag{2.14}$$

which results in $f(\boldsymbol{w}_k) < f(\boldsymbol{w}_{k-1}), k = 1, \ldots, K$. The descent behavior of GD is a key factor in developing our papers, as we will discuss them in Chapter 2 of this thesis. Let us consider that $f(\boldsymbol{w})$ is convex and $\bar{L}$-smooth, and assuming a step size $\alpha \leq 2/\bar{L}$, then it is possible to prove that the k-th iterate of GD satisfies

$$|f(\boldsymbol{w}_k) - f(\boldsymbol{w}^*)| \leq \frac{\|\boldsymbol{w}_0 - \boldsymbol{w}^*\|^2}{2\alpha k}, \tag{2.15}$$

where $f(\boldsymbol{w}^*)$ is the value of loss function at the optimal solution $\boldsymbol{w}^*$ of problem (2.2) [38]. Thus, GD has a sublinear convergence rate. Moreover, for any differentiable $\bar{L}$-smooth and μ-strongly convex loss function $f(\boldsymbol{w})$, GD achieves a linear convergence rate [56], as

$$f(\boldsymbol{w}_k) - f(\boldsymbol{w}^*) \leq \gamma^k (f(\boldsymbol{w}_0) - f(\boldsymbol{w}^*)), \tag{2.16}$$

where $\gamma := 1 - \mu/\bar{L}$.

2.5 Coordinate Descent

Coordinate descent (CD) is an alternative optimization algorithm that minimizes a function by iteratively updating one parameter (or a subset of parameters) at

a time while keeping the others fixed. CD can be particularly effective when the objective function exhibits a separable structure or when the dimensionality of the parameter space is very high. Compared to GD, CD exploits the sparsity or block structure of optimization problems to reduce the computational complexity in each iteration [57].

The CD aims to minimize the parameterized loss function $f(h(\boldsymbol{x}, \boldsymbol{w}), y)$, where $\boldsymbol{w} = [w_1, w_2, \ldots, w_d]^\top \in \mathbb{R}^d$ represents the model parameters, by updating a single coordinate w_v (or a subset of coordinates) at each iteration k. Denoting the gradient of f with respect to the v-th coordinate as $\nabla_{w_v} f(h(\boldsymbol{x}, \boldsymbol{w}), y)$, and starting from an arbitrary initialization $\boldsymbol{w}_0$, the CD update at iteration k for coordinate v is given by:

$$w_{v,k} = w_{v,k-1} - \alpha_v \cdot \frac{1}{N} \sum_{i=1}^{N} \nabla_{w_v} f(h(\boldsymbol{x}_i, \boldsymbol{w}_{k-1}), y_i), \quad v \in \{1, 2, \ldots, d\}, \tag{2.17}$$

where $\alpha_v \in (0, 1)$ is the step size specific to the v-th coordinate, and the summation is computed over the entire dataset $\mathcal{D} = \{(\boldsymbol{x}_i, y_i)\}_{i=1}^N$.

To complete a full iteration, the algorithm sequentially (or randomly) updates each coordinate w_v, cycling through all coordinates until a pre-defined convergence criterion is met. The CD algorithm iteratively minimizes the loss function with respect to each w_v, leveraging the partial derivatives to efficiently navigate the parameter space [57].

2.6 Stochastic Gradient Descent (SGD)

The stochastic gradient descent (SGD) algorithm drastically simplifies GD by performing the iterative process over a subset of the dataset $\mathcal{D}_k^s$ [21]. As a result, the computation overhead of SGD may be substantially lower than GD with full gradient computation, particularly in large-scale ML training when the training dataset is large. Therefore, at each iteration k, instead of computing the full gradient $\nabla_{\boldsymbol{w}} f(\boldsymbol{w})$, SGD selects a mini-batch data samples $\mathcal{D}_k^s$, of size $N_k^s < N$, from the dataset and computes the corresponding gradient defined as $\boldsymbol{g}_k(\boldsymbol{w}_k, \mathcal{D}_k^s)$ as

$$\boldsymbol{g}_k(\boldsymbol{w}_k, \mathcal{D}_k^s) := \frac{1}{N_k^s} \sum_{i \in \mathcal{I}_k} \nabla_{\boldsymbol{w}} f(h(\boldsymbol{x}_i, \boldsymbol{w}_{k-1}), y_i), \quad k = 1, \ldots, K, \tag{2.18}$$

where $\mathcal{I}_k$, with size N_k^s, denotes the randomly-selected integer indexes that give the mini-batch data samples at each iteration k. Considering (2.18), each iteration of SGD follows

$$\boldsymbol{w}_k = \boldsymbol{w}_{k-1} - \alpha_k \boldsymbol{g}_k(\boldsymbol{w}_k, \mathcal{D}_k^s), \quad k = 1, \ldots, K. \tag{2.19}$$

Let us consider a fixed mini-batch size of $N_k^s = N^s \ll N$, a diminishing step size α_k, and that the full gradient is approximated by an unbiased estimate of

$$\mathbb{E}_{\mathcal{D}_k^s} \{\boldsymbol{g}_k(\boldsymbol{w}_k, \mathcal{D}_k^s)\} = \nabla_{\boldsymbol{w}} f(\boldsymbol{w}_k). \tag{2.20}$$

The convergence rate of the mini-batch SGD is $\mathcal{O}(1/\sqrt{kN^s}+1/k)$[3] under Lipschitz gradient [56]. Therefore, although SGD can be effective in terms of iteration cost and memory, it is slow to converge and unable to adapt to strong convexity.

2.6.1 Adaptive Gradient (AdaGrad) Update

AdaGrad, short for Adaptive Gradient Algorithm, is designed to improve convergence by adapting the learning rate for each model parameter individually. Unlike SGD with a fixed learning rate, AdaGrad dynamically scales the learning rate inversely based on the magnitude of gradients, ensuring that parameters with smaller gradients receive larger learning rates and vice versa. This adjustment facilitates balanced optimization progress across dimensions, making it particularly advantageous for training DNNs where gradient magnitudes can vary significantly across layers. Additionally, this adaptive mechanism inherently mimics annealing by effectively reducing the learning rate as training progresses [58, 59]. AdaGrad achieves this aim by scaling the learning rate inversely proportional to the cumulative sum of squared gradients, dynamically adjusting it to improve convergence, especially for sparse features. In contrast to SGD, AdaGrad mitigates the risk of excessively large learning rates, enabling more effective navigation of the optimization landscape, even in scenarios with sparse gradients.

The AdaGrad update process begins with an initial parameter vector, $\boldsymbol{w}_0$, and a zero-initialized cumulative gradient vector, $\bar{\boldsymbol{g}}_0 = \mathbf{0}$, then

$$\begin{aligned} \boldsymbol{w}_k &\leftarrow \boldsymbol{w}_{k-1} - \alpha \boldsymbol{\nabla}_{k-1} \oslash (\bar{\boldsymbol{g}}_k + \epsilon_a)^{\circ \frac{1}{2}}, \quad k = 1, \ldots, K, \\ \bar{\boldsymbol{g}}_k &\leftarrow \bar{\boldsymbol{g}}_{k-1} + \boldsymbol{\nabla}_{k-1} \odot \boldsymbol{\nabla}_{k-1}, \quad k = 1, \ldots, K, \end{aligned} \tag{2.21}$$

where $\boldsymbol{\nabla}_{k-1} := \boldsymbol{g}_{k-1}(\boldsymbol{w}_{k-1}, \mathcal{D}^s_{k-1})$ as defined in (2.18), $\bar{\boldsymbol{g}}_k$ is accumulated sum of squared gradients, α is the preliminary step size, and ϵ_a is a small constant to prevent division by zero.

While AdaGrad is effective for sparse data and adapts learning rates dynamically, its primary drawback is that the accumulated sum of squared gradients grows indefinitely during training. This growth causes the learning rates to shrink excessively over time, potentially stalling the optimization process for non-convex problems. To address this limitation, we present the AdaDelta in the following.

2.6.2 Adaptive Delta (AdaDelta) Update

AdaDelta (Adaptive Delta) was introduced as an extension to AdaGrad to overcome its primary limitation: the indefinite accumulation of squared gradients, which leads to excessively small learning rates during prolonged training. By employing a decaying average to regulate the accumulation of past gradients, AdaDelta ensures that learning rates remain adaptive without diminishing to impractically small values [58]. It makes AdaDelta particularly effective for non-convex optimization

problems and deep neural network training, where maintaining a reasonable learning rate is critical for convergence.

We define $E[\boldsymbol{g}^2]_k$ as the running average at each iteration k, and compute it as

$$E[\boldsymbol{g}^2]_k = \rho E[\boldsymbol{g}^2]_{k-1} + (1-\rho)\boldsymbol{\nabla}_{k-1} \odot \boldsymbol{\nabla}_{k-1}, \quad k = 1, \ldots, K \tag{2.22}$$

where $\boldsymbol{\nabla}_{k-1} := \boldsymbol{g}_{k-1}(\boldsymbol{w}_{k-1}, \mathcal{D}^s_{k-1})$ as defined in (2.18), and $\rho \in (0,1)$ is a decay constant. To refine the step size for updating $\boldsymbol{w}_k$, the AdaDelta procedure requires the element-wise square root of $E[\boldsymbol{g}^2]_k$, and it effectively becomes the RMS of the previous squared gradients up to the iteration k, as

$$\mathrm{RMS}[\boldsymbol{g}]_k = \left(E[\boldsymbol{g}^2]_k + \epsilon_a\right)^{\circ \frac{1}{2}}, \quad k = 1, \ldots, K \tag{2.23}$$

where $\epsilon_a > 0$ is a small constant to prevent division by zero. Then, the updated $\boldsymbol{w}_k$ is as follows:

$$\boldsymbol{w}_k \leftarrow \boldsymbol{w}_{k-1} - \alpha \; \boldsymbol{\nabla}_{k-1} \oslash \mathrm{RMS}[\boldsymbol{g}]_k, \quad k = 1, \ldots, K. \tag{2.24}$$

Thus far, we have provided the theories and formulations regarding ML models in one device. However, IoT devices are connected in a distributed system and must collaborate with other devices and servers. Such distributed frameworks require expanding the theoretical discussions to distributed ML models. Therefore, in the next section, we discuss distributed ML, which is the basis of this thesis.

2.7 Distributed ML

Given the dispersed characteristic of many wireless systems, distributed ML is critical in various wireless network environments. In the IoT era, where many connected devices produce large amounts of data due to the diversification of services, and new applications, it is crucial to exploit distributed ML for future wireless networks [12]. Moreover, the increasingly congested wireless channels resulting from the transmission of a large volume of data, and the crucial need for data privacy, prevent the connected devices from sharing their local data with the servers. Therefore, distributed ML is one solution to reduce data exchange and bandwidth occupation in wireless communication networks.

In a distributed ML setup, M devices in a star network are connected to a central server through a wireless communication channel. Every device $j \in \{1, 2, \ldots, M\}$ keeps a local subset $\mathcal{D}_j := \{(\boldsymbol{x}_i^j, y_i^j)\}_{i=1}^{N_j}$, where $\mathcal{D} = \cup_{j=1}^{M} \mathcal{D}_j$ denotes all the data available in the system. The data can be distributed in an IID (Independent and Identically Distributed) or non-IID manner. IID data assumes that data points are sampled independently and share the same underlying distribution across all devices, enabling stable and consistent model training. However, real-world scenarios often involve non-IID data, where data distributions vary across devices due to differences in user behavior, environments, or tasks. This heterogeneity poses

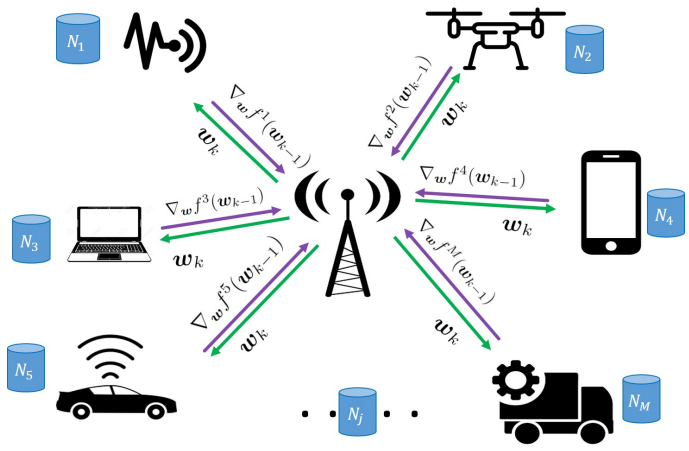


Figure 2.4: Distributed GD scheme: Each device $j \in \{1, \ldots, M\}$ computes the local gradient $\nabla_{\boldsymbol{w}} f^j(\boldsymbol{w}_{k-1})$ and transmits to the server for the new update of the global parameter $\boldsymbol{w}_k$. The server updates $\boldsymbol{w}_k$ according to (2.28) and broadcasts it to the devices for starting the new iteration k.

significant challenges, such as biased model updates or reduced convergence speed. Thereafter, each device $j \in \{1, 2, \ldots, M\}$ is associated with its local loss function defined as

$$f^j(\boldsymbol{w}) := \frac{1}{N_j} \sum_{i=1}^{N_j} f(h(\boldsymbol{x}_i^j, \boldsymbol{w}), y_i^j), \quad j = 1, \ldots, M. \tag{2.25}$$

The aim of a distributed ML algorithm is that the devices try to collaboratively minimize the finite sum of their local loss functions and obtain the optimal solution $\boldsymbol{w}^*$, defined as

$$\boldsymbol{w}^* \in \arg\min_{\boldsymbol{w}} \frac{1}{M} \sum_{j=1}^{M} f^j(\boldsymbol{w}). \tag{2.26}$$

One of the most common distributed ML algorithms is distributed GD, in which there are M devices connected to a central server in a star network through a wireless communication channel as shown in Figure 2.4. Every device $j \in \{1, 2, \ldots, M\}$ keeps its own data, $\mathcal{D}_j$ with size N_j, and performs the GD over its private data. At each iteration $k = 1, \ldots, K$ of GD, the server broadcasts the global learning parameter defined as $\boldsymbol{w}_{k-1}$ to all the devices and each device $j \in \{1, 2, \ldots, M\}$

computes its local gradient $\nabla_{\boldsymbol{w}} f^j(\boldsymbol{w}_k)$ as

$$\nabla_{\boldsymbol{w}} f^j(\boldsymbol{w}_{k-1}) = \frac{1}{N_j} \sum_{i=1}^{N_j} \nabla_{\boldsymbol{w}} f(h(\boldsymbol{x}_i^j, \boldsymbol{w}_{k-1}), y_i^j), \quad j = 1, \ldots, M, \tag{2.27}$$

and transmits it to the server through a wireless channel. When all the devices complete the transmission of their local gradient, the server updates the global parameter $\boldsymbol{w}_k$ as

$$\boldsymbol{w}_k = \boldsymbol{w}_{k-1} - \frac{\alpha}{M} \sum_{j=1}^{M} \nabla_{\boldsymbol{w}} f^j(\boldsymbol{w}_{k-1}), \tag{2.28}$$

and broadcasts it to the devices. This iterative process continues until a predefined convergence threshold ϵ is achieved as

$$\left\| \frac{1}{M} \sum_{j=1}^{M} \nabla_{\boldsymbol{w}} f^j(\boldsymbol{w}^*) \right\|_2 \leq \epsilon, \tag{2.29}$$

or the system completes the number of predefined iterations K.

2.8 Federated Learning (FL)

The recent proliferation of successful DNN applications has broadly utilized SGD variations as their optimization algorithm. As a result, numerous advancements within the field can be comprehended as modifications to the model structure, and thus the loss function, to facilitate optimization through straightforward gradient-based methods [60]. Furthermore, with the explosive amount of data produced by a vast number of IoT devices, distributed ML algorithms are widely exploited by researchers worldwide. Among these approaches, Federated Learning (FL) has emerged as a promising paradigm that preserves data privacy by avoiding data sharing between devices and servers [60, 61].

Initializing the training process with $\boldsymbol{w}_0$, FL is a distributed ML algorithm in which the server sends $\boldsymbol{w}_{k-1}$ to the devices at the beginning of each iteration $k = 1, \ldots, K$. Each device $j \in \{1, 2, \ldots, M\}$ keeps a subset $\mathcal{D}_j = \{(\boldsymbol{x}_n^j, y_n^j)\}_{n=1}^{N_j}$ of the global dataset, $\mathcal{D}$, where $\mathcal{D} = \cup_{j=1}^{M} \mathcal{D}_j$, performs one local iteration of GD [60], and computes its local parameter $\boldsymbol{w}_k^j$,

$$\boldsymbol{w}_k^j = \boldsymbol{w}_{k-1} - \frac{\alpha_k}{N_j} \sum_{n=1}^{N_j} \nabla_{\boldsymbol{w}} f(h(\boldsymbol{x}_n^j, \boldsymbol{w}_{k-1}), y_n^j), \quad k = 1, \ldots, K, \tag{2.30}$$

where α_k is the step size. Then each device transmits $\boldsymbol{w}_k^j$ to the server for updating $\boldsymbol{w}_k$, called global update, according to

$$\boldsymbol{w}_k = \sum_{j=1}^{M} \rho_j \boldsymbol{w}_k^j, \quad k = 1, \ldots, K, \tag{2.31}$$

where $\rho_j := N_j/N$.

Federated averaging (FedAvg) is a well-known example of FL, in which the computations of the local parameters are different than FL. In FedAvg, the server sends $\boldsymbol{w}_{k-1}$ to the devices at the beginning of each iteration $k = 1, \ldots, K$. Each device $j \in \{1, 2, \ldots, M\}$ keeps a subset $\mathcal{D}_j = \{(\boldsymbol{x}_n^j, y_n^j)\}_{n=1}^{N_j}$ of the global dataset, $\mathcal{D}$, where $\mathcal{D} = \cup_{j=1}^{M} \mathcal{D}_j$, and performs L of local iterations, $i = 1, \ldots, L$, of SGD [21] with data subset of $N_k^j \leq N_j$, and computes its local parameter $\boldsymbol{w}_{i,k}^j$, considering the initial point of $\boldsymbol{w}_{0,k}^j = \boldsymbol{w}_{k-1}$, [61]

$$\boldsymbol{w}_{i,k}^j = \boldsymbol{w}_{i-1,k}^j - \frac{\alpha_k}{N_k^j} \sum_{n=1}^{N_k^j} \nabla_{\boldsymbol{w}} f(h(\boldsymbol{x}_n^j, \boldsymbol{w}_{i-1,k}^j), y_n^j), \quad k = 1, \ldots, K, \tag{2.32}$$

where $\boldsymbol{w}_k^j = \boldsymbol{w}_{L,k}^j$. The update of $\boldsymbol{w}_k$ is the same as the FL global update in (2.31).

Although FL seems to have less computation complexity with one local iteration, FedAvg can reduce the iterations of communication needed for training by a factor of the number of local iterations L. This reduction in the number of communication iterations is by orders of magnitude less than FL, especially when the system tries to train DNNs on decentralized data [60].

Several FL characteristics distinguish it from other parallel optimizations. First, in a large-scale network with a large volume of generated data, the connection between the central server and a device might become slow. Slow communication is a direct consequence of this slow connection, which is crucial in latency-sensitive applications such as smart healthcare [9]. Therefore, there is a considerable need to develop communication-efficient FL methods. Moreover, due to the limitation in the computation power of devices, developing the computation or energy-efficient FL has gotten more attention in recent years. We elaborate more on this topic in the next section of the thesis.

The other characteristic of FL is that the system loses its control over devices, where disruptions in connectivity can occur. For instance, when a mobile phone is scheduled to participate in FL training and turned off or loses WiFi access, the central server experiences a loss of connection with the device. The system can only wait for or discard the lost devices because it has no control over the scheduled devices. Unfortunately, waiting for all the devices' responses is not always possible in practice. Due to this reason, it is important to select properly the devices that participate in the FL training process [61]. Finally, the data heterogeneity of devices mainly results in the local parameter's *drift*, which slows down the convergence speed of FL. Recently, the authors of [62] have addressed the challenge of user heterogeneity in FL by proposing FedGenP, a novel data augmentation method. FedGenP focuses on reducing data distribution discrepancies between users by training personalized generators at the server. These generators leverage local models from other users to synthesize latent-space data that align with areas of user disagreement. By augmenting user data with these synthetic samples, the method transfers knowledge across users while preserving privacy. This approach

effectively diminishes heterogeneity, thereby improving the global model's performance for all users. Although there are FL works in the literature when the data is non-IID, using the FedAvg algorithm on non-IID data lacks theoretical guarantee even in a convex optimization setting [60].

Thus far, we have discussed the theoretical foundations of the FL and FedAvg algorithms. In the literature, as well as in Papers 1–6 of Chapter 7 of this thesis, convergence analyses often rely on pre-defined assumptions concerning the local and global gradient vectors. In the next subsection of this chapter, we present the key assumptions utilized for the convergence analysis.

Assumptions of the gradients

Finally, we outline the key assumptions, **A1**-**A5**, on the loss functions f and f^j, as discussed in [63], which are crucial for the convergence analysis presented in the papers included in Chapter 7 of this thesis.

A1: (Unbiased Gradient). $\nabla_w f^j(\boldsymbol{w}; \boldsymbol{x}_{nj}, y_{nj})$ is an unbiased estimate of each user's true gradient $\nabla_w f^j(\boldsymbol{w})$, for any $j \in [M], n \in [|\xi|]$.

A2: (Lipschitz Gradient). The function f^j, for all $j \in [M]$ is $\bar{L}$-smooth, $0 < \bar{L} < \infty$, i.e.,

$$\left\|\nabla f^j(\boldsymbol{w}_1) - \nabla f^j(\boldsymbol{w}_2)\right\| \leq \bar{L} \left\|\boldsymbol{w}_1 - \boldsymbol{w}_2\right\|,$$

for all $\boldsymbol{w}_1, \boldsymbol{w}_2 \in \mathbb{R}^d$.

A3: (Bounded Local Variance). The function f^j, has σ_l-bounded variance as

$$\mathbb{E}_\xi \left\|[\nabla f^j(\boldsymbol{w}; \boldsymbol{x}_{nj}, y_{nj})]_i - [\nabla f^j(\boldsymbol{w})]_i\right\|^2 \leq \sigma_{l_i}^2,$$

for all $j \in [M]$ $\boldsymbol{w} \in \mathbb{R}^d$, $i \in [d]$, $n \in [|\xi|]$.

A4: (Bounded Global Variance). Considering the gradient of the global loss function $\nabla_w f(\boldsymbol{w})$, the global variance is bounded as

$$\frac{1}{M}\sum_{j=1}^{M} \left\|[\nabla f^j(\boldsymbol{w})]_i - [\nabla f(\boldsymbol{w})]_i\right\|^2 \leq \sigma_{g_i}^2,$$

for all $j \in [M]$, $\boldsymbol{w} \in \mathbb{R}^d$, $i \in [d]$.

A5: (Bounded Gradients). The function f^j has G-bounded gradients as

$$\left|[\nabla f^j(\boldsymbol{w}; \boldsymbol{x}_{nj}, y_{nj})]_i\right| \leq G,$$

for any $j \in [M], n \in [\xi]$, $\boldsymbol{w} \in \mathbb{R}^d$ and $i \in [d]$.

The bounded gradient assumption is widely used for non-convex decentralized gradient optimization. Moreover, it holds for many neural network training tasks due to the cross-entropy loss function [64].

2.9 Summary

This chapter provided an overview of ML and its objectives, focusing on the mathematical formulation of supervised learning problems and the optimization techniques used for training ML models. It explored the fundamentals of ML methods, including ANN, DNNs, and CNNs, as well as key optimization algorithms such as GD, CD, and SGD. Additionally, it examined distributed ML approaches, including FL and the FedAvg algorithm.

Chapter 3

Cell-free Massive Multiple-Input Multiple-Output Networks

The evolution of mobile networks has shifted their primary function from supporting voice calls to providing wireless access to data services over large geographical areas. Initially, their primary function was supporting voice calls; however, data transmission has since become the dominant service, with data rate (measured in bits per second) now the key indicator of network quality. As signal strength decreases rapidly with distance, mobile networks rely on a distributed network of transceivers strategically placed at elevated locations like masts and rooftops to ensure wide and reliable coverage. These transceivers are known as access points (APs), while the connected devices are called user equipment (UE) [2].

Current mobile networks operate as cellular networks, where each UE connects to the AP, providing the strongest signal and creating what's known as a cell. Despite advancements like Massive MIMO, which equips each AP with an antenna array to serve multiple UEs at the same time and frequency using directed signals, significant data rate variations remain inherent to this architecture. UEs transmit over shared time-frequency resources in the uplink, while APs use spatial processing to separate desired signals from interference. In the downlink, the APs send highly directional signals, enhancing signal-to-noise ratio (SNR) and minimizing inter-user interference, which benefits both cell-center and cell-edge UEs. Although beamforming increases SNR without increasing interference, considerable data rate disparities persist between the UEs in the coverage area, and while APs could theoretically adjust transmit power to reduce these differences, doing so would lower rates across the cell, limiting performance for center UEs.

Current cellular networks provide high peak data rates near cell centers, but substantial rate variations across each cell create inconsistent service quality. While many areas within a cell achieve acceptable rates, this is insufficient for a society where ubiquitous wireless connectivity underpins essential services such as pay-

ments, navigation, entertainment, and autonomous vehicle control. Therefore, ensuring reliable and uniform data service across entire coverage areas is essential to support these critical, connectivity-dependent applications.

The primary goal for future mobile networks should shift from increasing peak data rates to ensuring reliable rates that can be consistently delivered across the vast majority of locations within the coverage area. Since the cellular network architecture was originally designed for low-rate voice services rather than high-rate data, it is time to move beyond the cellular paradigm and adopt a network design that can meet future performance demands. Cell-free Massive Multiple-Input Multiple-Output (CFmMIMO) network is a promising candidate for achieving this goal, as it eliminates the traditional cell boundaries and provides uniform coverage by distributing APs across the entire area, ensuring more consistent and reliable data rates for UEs. Accordingly, cell-free networks offer three key advantages over traditional cellular networks. First, it reduces SNR variations, especially when compared to cellular networks with sparse AP deployments and Massive MIMO systems. Second, it enables efficient interference management through joint processing at multiple APs, a feature not typically utilized in cellular networks with equally dense AP deployments. Finally, coherent transmission in cell-free networks improves SNR by involving APs with weaker channels in the transmission process rather than relying solely on the AP with the strongest channel.

In this chapter, we focus on CFmMIMO networks. We begin by providing an introduction and overview of CFmMIMO, covering its background and general architecture. Next, we explain the concepts of block fading and coherence blocks to help readers understand the uplink process in CFmMIMO networks. We then explore potential downlink scenarios in CFmMIMO and the corresponding advantages and disadvantages of these scenarios for FL training over CFmMIMO networks.

3.1 Overview of CFmMIMO Networks

CFmMIMO incorporates concepts from several established technologies, including Cellular Massive MIMO, Coordinated Multipoint (CoMP) [65, 66], and ultra-dense networks [67]. Accordingly, CFmMIMO can be defined in two ways: First, it is the intersection of these three technologies, combining the best aspects of each into a unified network optimized for performance. It is represented by the inner region of the Venn diagram in Figure 3.1. The second definition sees CFmMIMO as an overarching concept, encompassing conventional Massive MIMO, CoMP, and ultra-dense networks as special cases. This thesis follows the first definition, emphasizing these three approaches' narrow, optimized integration.

In CFmMIMO networks, the APs cooperate to serve the UEs through joint coherent transmission and reception, leveraging physical layer concepts derived from the cellular massive MIMO domain. We consider a scenario with A_p APs arbitrarily distributed across the coverage area, where each AP is equipped with N antennas. These APs jointly serve M single-antenna UEs. The APs are interconnected via

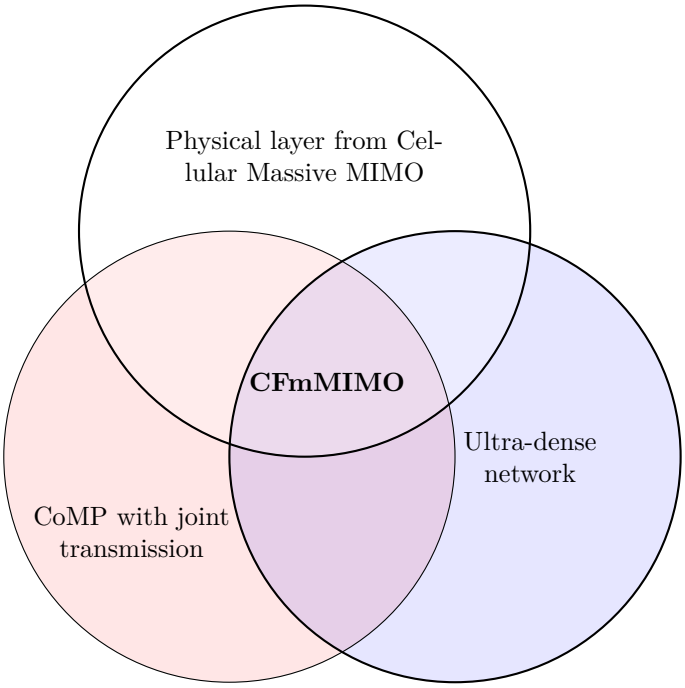


Figure 3.1: CFmMIMO is defined as the intersection of Cellular Massive MIMO's physical layer, joint transmission from CoMP literature, and the ultra-dense network deployment regime.

fronthaul links to a central server, which facilitates the necessary heavy computations and coordination between APs. The general architecture of a CFmMIMO network is depicted in Figure 3.2. With a total number of $A_p \cdot N$ AP antennas, which typically exceeds the number of UEs M [1], the cell-free network provides a large number of spatial degrees of freedom, which enables the network to effectively separate UEs in space through linear processing of the transmitted and received signals.

3.2 Block Fading and Coherence Blocks

In wireless communication, understanding propagation channels is crucial for reliable and efficient performance. Ideally, channels are linear and time-invariant, but in practice, they are often time-variant due to the movements of the transmitter, receiver, or objects in the environment. However, the channel can be approximated

[1] The term *massive* refers to $A_p \cdot N \gg M$, namely the number of AP antennas is very larger than the number of UEs.

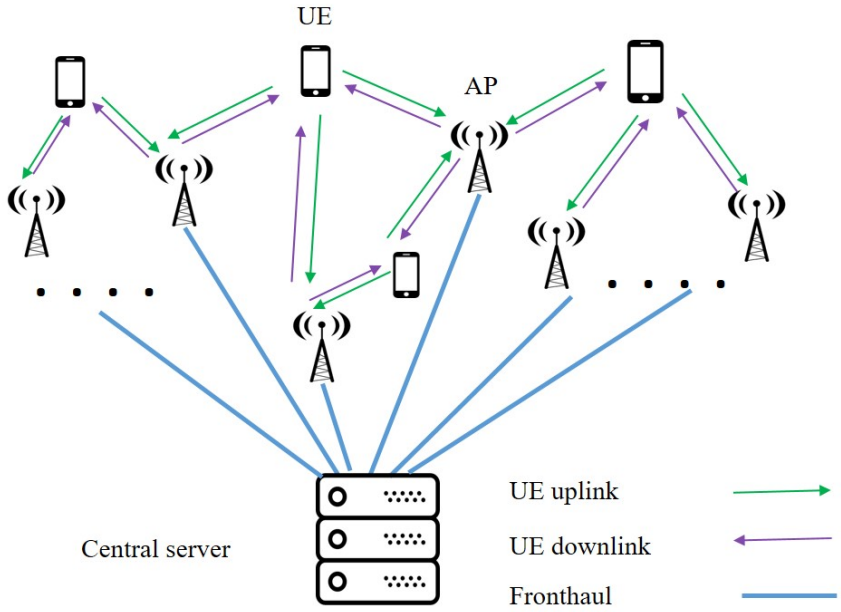


Figure 3.2: General architecture of a CFmMIMO network.

over short intervals as constant, making the system time-invariant within this period. This interval called the channel **coherence time** T_c, typically lasts a few milliseconds in mobile networks, during which the system estimates the channel properties. Within T_c, the channel can be modeled as a finite impulse response (FIR) filter, with each term representing a distinct propagation path characterized by time delay and path loss. The frequency response of the channel varies smoothly across narrow frequency bands, defined by the channel **coherence bandwidth** B_c, where the channel can be considered approximately constant.

A channel **coherence block** is a time-frequency block with a duration equal to the coherence time T_c and a frequency width equal to the coherence bandwidth B_c. Within this block, the channel remains constant and frequency-flat, allowing it to be described by a single scalar coefficient. The time-frequency resources of a communication system can be divided into multiple coherence blocks, as shown in Figure 3.3, with the blocks distributed over both time and frequency. For communication algorithm development, it is useful to analyze and optimize the system in terms of these blocks. Assuming the scalar coefficient describing the channel takes an independent random value in each coherence block, the system can be studied block by block without loss of generality. This approach is known as the **block-fading** model, where the channel experiences independent fading in each coherence block. According to the **Nyquist-Shannon** sampling theorem, a signal that fits

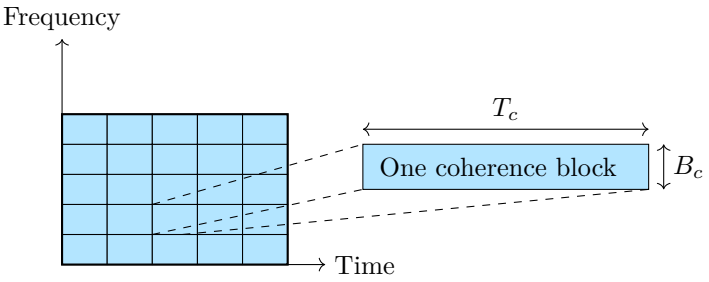


Figure 3.3: The time-frequency resources in the block-fading model: the time-frequency resources are divided into coherence blocks in which the channel is time-invariant and frequency-flat, and the channel is independent between different coherence blocks [2].

within a coherence block is uniquely described by $\tau_c := T_c B_c$ complex-valued samples. These samples represent the parameters used to convey information in a communication system and will be referred to as transmission symbols.

3.2.1 Time-Division Duplex (TDD) Protocol

The channel between AP m and UE j in a given coherence block is represented by the vector $\boldsymbol{g}_m^j \in \mathbb{C}^N$, as

$$\boldsymbol{g}_m^j := [g_{m,1}^j, \ldots, g_{m,N}^j]^\top, \quad m = 1, \ldots, A_p, \; j = 1, \ldots, M, \tag{3.1}$$

where each element $g_{m,n}^j$ denotes the complex-valued scalar channel coefficient between the UE j and the nth antenna of AP m. Under the block-fading model, the channel vector $\boldsymbol{g}_m^j$ takes an independent realization from a stationary random distribution in each coherence block. Due to channel reciprocity, this realization is the same for both the uplink and downlink. To perform coherent processing both within each AP's antennas and across cooperating APs, the APs must know the channel realizations, at least partially. The most efficient approach for channel estimation is to use a Time-Division Duplex (TDD) protocol, where each coherence block accommodates both uplink and downlink transmissions. In this setup, pilots are transmitted only in the uplink. Based on the received uplink signals, each AP estimates the channels between itself and all UEs. These estimates are then used for both uplink and downlink transmissions, leveraging channel reciprocity.

Considering the standard massive MIMO TDD protocol, the τ_c transmission symbols of a coherence block are allocated for three purposes: τ_p symbols for uplink pilots, τ_u for uplink data, and τ_d downlink data, such that $\tau_c = \tau_p + \tau_u + \tau_d$ as shown in Figure 3.4. Pilots and data are transmitted at separate times, with the protocol synchronized across APs. Uplink pilots must precede downlink data to enable precoding based on channel estimates. Uplink data may be sent before

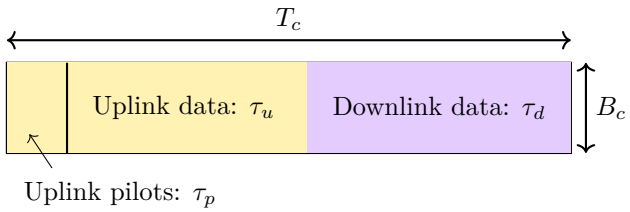


Figure 3.4: Utilizing TDD protocol where each coherence block accommodates both uplink and downlink transmissions [2].

pilots, or uplink and downlink data transmission orders can be swapped, provided all transmissions fit within a single coherence block. However, these variations have drawbacks: transmitting uplink data first increases latency, while swapping uplink and downlink data requires a guard interval for switching transmission modes.

3.3 Uplink Processing in CFmMIMO

In this subsection, we focus on the uplink scenario, where the coherence block length τ_c is divided into $\tau_p < \tau_c$ pilot channel uses, and $\tau_c - \tau_p$ channel uses for data transmission. The system operates under the standard block-fading model, where each channel remains constant within a time-frequency coherence block of τ_c channel uses and takes independent realizations across blocks [2]. The channel coefficients are modeled as independent Rayleigh fading, where $g_{m,n}^j \sim \mathcal{N}_{\mathbb{C}}(0, \beta_m^j)$, representing the channel between UE j and the nth antenna of AP m, with β_m^j denoting the large-scale fading coefficient, resulting in $\boldsymbol{g}_m^j \sim \mathcal{N}_{\mathbb{C}}(\mathbf{0}, \beta_m^j \boldsymbol{I}_N)$. The complex Gaussian distribution models the random small-scale fading caused by small-scale movements of the transmitter, receiver, or objects in the propagation environment. Specifically, the received signal results from the summation of signal copies arriving via numerous propagation paths with random phase shifts, leading to constructive and destructive superposition. This superposition can be approximated by a complex Gaussian distribution. This fading model is known as Rayleigh fading, as each $g_{m,n}^j$ follows a Rayleigh distribution. Moreover, in the block-fading model with Rayleigh fading, an independent realization is drawn from the complex Gaussian distribution in each coherence block. The variance β_m^j represents the large-scale fading, determining the average channel quality as the UE moves within a relatively small area.

3.3.1 Channel Estimation

This subsection explains the channel estimation process in CFmMIMO systems using uplink pilot transmission.

Algorithm 3.1 Pilot assignment

Inputs: M, $\beta_m^j|_{j\in[M],m\in[A_p]}$, τ_p
Initialization: Pilot index of UEs $x_1 = x_2 = \ldots = x_M = 0$
for $j = 1, \ldots, \tau_p$, **do** ▷ Pilot assignment to the first τ_p UEs
    $x_j = j$
end for
for $j = \tau_p + 1, \ldots, M$, **do**
    $q \leftarrow \arg\max_{m\in[A_p]} \beta_m^j$ ▷ The best AP for UE j
    $x_j \leftarrow \arg\min_{t\in[\tau_p]} \sum_{\substack{i=1 \\ x_i=t}}^{j-1} \beta_q^i$ ▷ Pilot with least interference at AP q
end for
return Pilot assignments $x_1, \ldots, x_M$

Pilot Transmission

Coherent transmission processing requires knowledge of the channel responses between UEs and their serving APs. Specifically, AP m needs an estimate of the channel vector $\boldsymbol{g}_m^j$ for UE j. Under the block-fading model, where channels remain constant within a coherence block and vary independently across blocks, channel estimation must be performed once per coherence block. In the TDD protocol, τ_p samples are allocated for uplink pilot signaling in each coherence block. During this phase, each UE transmits a unique pilot sequence spanning τ_p samples, and the APs use the received signals to estimate the channels. Accordingly, each UE j is assigned a τ_p-length pilot from a set of τ_p mutually orthogonal pilot sequences.

The UE pilot assignment follows Algorithm 4.1 in [2], detailed in Algorithm 3.1 and as follows: First, the τ_p UEs with indices 1 to τ_p are assigned mutually orthogonal pilots, where UE j uses pilot k for $j = 1, \ldots, \tau_p$. The remaining UEs, indexed from $\tau_p + 1$ to M, are assigned pilots sequentially. Each UE j determines the AP to which it has the strongest channel, with the AP index computed as:

$$q = \arg\max_{m\in[A_p]} \beta_m^j. \tag{3.2}$$

AP q is likely to play a key role in serving UE j, so it is beneficial to assign UE j a pilot that minimizes pilot contamination at AP q. To this aim, AP q calculates the total average channel gains of the UEs already assigned to each pilot t. The pilot with the lowest cumulative interference is then selected based on the following:

$$x_j = \arg\min_{t\in[\tau_p]} \sum_{\substack{i=1 \\ x_i=t}}^{j-1} \beta_q^i, \quad j = \tau_p + 1, \ldots, M. \tag{3.3}$$

After completing the pilot assignment, the pilot of UE j is denoted $\sqrt{\tau_p}\boldsymbol{\varphi}_j \in \mathbb{C}^{\tau_p \times 1}$, where p^u is the maximum transmit power, and $\|\boldsymbol{\varphi}_j\|^2 = 1$. The UEs send these

sequences simultaneously and then the received vector $\boldsymbol{y}^p_{m,n} \in \mathbb{C}^{\tau_p \times 1}$ at antenna n of the AP m is

$$\boldsymbol{y}^p_{m,n} = \sqrt{p_p} \sum_{j=1}^{M} g^j_{m,n} \boldsymbol{\varphi}_j + \boldsymbol{\omega}^p_{m,n}, \quad m = 1, \ldots, A_p, \tag{3.4}$$

where $p_p := \tau_p p^u$ is the pilot energy and $\boldsymbol{\omega}^p_{m,n}$ is the independent complex Gaussian noise with the noise power of σ^2. By projecting $\boldsymbol{y}^p_{m,n}$ onto the pilot $\boldsymbol{\varphi}_j$, we obtain $\hat{y}^{p,j}_{m,n} = \boldsymbol{\varphi}^H_j \boldsymbol{y}^p_{m,n}$. Then, the MMSE estimate of $g^j_{m,n}$ is obtained as [2]

$$\hat{g}^j_{m,n} = \frac{\mathbb{E}\left\{\hat{y}^{p,j}_{m,n} g^{*j}{}_{m,n}\right\}}{\mathbb{E}\left\{\left|\hat{y}^{p,j}_{m,n}\right|^2\right\}} \hat{y}^{p,j}_{m,n} = c^j_m \hat{y}^{p,j}_{m,n}, \quad m = 1, \ldots, A_p,\ j = 1, \ldots, M, \tag{3.5}$$

where

$$c^j_m := \frac{\sqrt{p_p}\beta^j_m}{p_p \sum_{j'=1}^{M} \beta^{j'}_m |\boldsymbol{\varphi}^H_{j'} \boldsymbol{\varphi}_j|^2 + \sigma^2} \quad m = 1, \ldots, A_p,\ j = 1, \ldots, M. \tag{3.6}$$

Now, considering the definitions above, we present the uplink transmission process in the following subsection.

Uplink Data Transmission

We consider that all the M UEs simultaneously send their local models to the APs in the uplink. We define q_j, $j = 1, \ldots, M$, as the independent unit-power data symbol associated with the jth UE (i.e., $\mathbb{E}\left\{|q_j|^2\right\} = 1$). UE j sends its signal using the power $p^u p_j$, where $0 \leq p_j \leq 1$ is the power control coefficient we will optimize later, and p^u is the maximum uplink power. Then, the received signal $y^{u,j}_{m,n}$ at antenna n of AP m is

$$y^u_{m,n} = \sum_{j=1}^{M} g^j_{m,n} q_j \sqrt{p^u p_j} + \omega^u_{m,n}, \quad m = 1, \ldots, A_p, \tag{3.7}$$

where $\omega^u_{m,n} \sim \mathcal{N}_{\mathbb{C}}(0, \sigma^2)$ is the additive noise at the nth antenna of AP m. After the APs receive the uplink signal, they use maximum-ratio processing. Maximum-ratio processing is an effective choice for the uplink of CFmMIMO in FL due to its low complexity and efficiency. Maximum-ratio processing is an effective choice for the uplink of CFmMIMO in FL due to its low complexity and efficiency. By coherently combining received signals using the conjugate of the estimated channel coefficients, it maximizes the SNR for each user, enhancing reliability. Additionally, it suppresses noise and leverages the distributed nature of CFmMIMO to mitigate inter-user interference, ensuring accurate aggregation of FL updates. Similar to [68, Th. 2], the achievable uplink data rate (in bit/s) at UE j is

$$R_j = B_\tau \log_2 \left(1 + \mathrm{SINR}_j(\boldsymbol{p})\right), \ j = 1, \ldots, M, \tag{3.8}$$

where $\mathrm{SINR}_j(\boldsymbol{p})$ is the signal-to-interference+noise ratio of UE j, $\boldsymbol{p} := [p_1, \ldots, p_M]^\top$, $B_\tau := B(1 - \tau_p/\tau_c)$ is the pre-log factor, B is the bandwidth (Hz), and

$$\gamma_m^j := \mathbb{E}\left\{|\hat{g}_{m,n}^j|^2\right\} = \sqrt{p_p}\beta_m^j c_m^j \quad m = 1, \ldots, A_p,\ j = 1, \ldots, M. \tag{3.9}$$

Then, for each UE $j = 1, \ldots, M$, the SINR_j is obtained as:

$$\mathrm{SINR}_j(\boldsymbol{p}) := \frac{\bar{A}_j p_j}{\bar{B}_j p_j + \sum_{j' \neq j}^{M} p_{j'} \tilde{B}_j^{j'} + I_M^j}, \tag{3.10}$$

where for each AP $m = 1, \ldots, A_p$,

$$\bar{A}_j := \left(\sum_{m=1}^{A_p} N\gamma_m^j\right)^2, \quad \bar{B}_j := \sum_{m=1}^{A_p} N\gamma_m^j \beta_m^j, \tag{3.11}$$

$$\tilde{B}_{k'}^j := \sum_{m=1}^{A_p} N\gamma_m^{k'}\beta_m^j + |\boldsymbol{\varphi}_{k'}^H \boldsymbol{\varphi}_j|^2 \left(\sum_{m=1}^{A_p} N\gamma_m^{k'} \frac{\beta_m^j}{\beta_m^{k'}}\right)^2, \tag{3.12}$$

$$I_M^j = \sum_{m=1}^{A_p} N\sigma^2 \gamma_m^j / p^u. \tag{3.13}$$

The next step involves determining the power coefficient p_j for each UE, tailored to a specific objective, such as ensuring fairness in UE rates or maximizing the total sum of the UEs' rates. To this aim, the following section delves into the topic of uplink power control, exploring strategies to achieve these goals effectively.

3.4 Uplink Power Control Schemes in CFmMIMO

In this section, we determine the power coefficients p_j by presenting two power control schemes: one that maximizes the minimum uplink rate (max-min rate) among UEs and another that maximizes the sum of UEs' uplink rates. These schemes serve as benchmarks for comparing our proposed approaches in this thesis.

3.4.1 Max-min Rate

The goal of the max-min rate is to maximize the smallest uplink rate among all UEs in the network, a concept referred to as max-min fairness. Since the uplink rate of UE j in (3.8) is an increasing function of the $\mathrm{SINR}_j(\boldsymbol{p})$, maximizing the minimum SE is equivalent to maximizing the minimum SINR across all UEs. Therefore, for the M UEs, the max-min fairness optimization problem is as follows:

$$\underset{p_1,\ldots,p_M}{\text{maximize}} \quad \underset{j\in[M]}{\text{minimize}} \quad \frac{\bar{A}_j p_j}{\bar{B}_j p_j + \sum_{j' \neq j}^{M} p_{j'} \tilde{B}_j^{j'} + I_M^j} \tag{3.14a}$$

$$\text{subject to} \quad 0 \leq p_j \leq 1, \quad j \in [M]. \tag{3.14b}$$

Algorithm 3.2 Max-min rate power allocation

Inputs: M, $\{\bar{A}_j, \bar{B}_j, \tilde{B}_j^{j'}, I_M^j\}_{j,j' \in [M]}$
Initialization: $\boldsymbol{p}$, ϵ_M, $S_{\max} = 1$, $S_{\min} = 0$, $\bar{p}_{\max} = 0$
while $S_{\max} - S_{\min} > \epsilon_M$, **do**
    Set $p_j \leftarrow \frac{p_j}{\mathrm{SINR}_j(\boldsymbol{p})}$, $j \in [M]$
    Update $\bar{p}_{\max} = \max_{j \in [M]} \; p_j$
    Set $\boldsymbol{p} \leftarrow \frac{1}{\bar{p}_{\max}} \boldsymbol{p}$
    Update $S_{\max} = \max_{j \in [M]} \; \mathrm{SINR}_j(\boldsymbol{p})$
    Update $S_{\min} = \min_{j \in [M]} \; \mathrm{SINR}_j(\boldsymbol{p})$
end while
return $\boldsymbol{p}$

To solve the max-min optimization problem in (3.14) and determine the power coefficients p_j, $j \in [M]$, we utilize the fixed-point Algorithm 7.1 from [2], outlined as Algorithm 3.2. This algorithm converges to the optimal solution of the max-min fairness problem in (3.14), ensuring that all UEs achieve equal uplink rate at the optimum, with at least one UE utilizing the maximum power coefficient of 1.

Although the max-min rate approach prioritizes UEs with the poorest channel conditions, this focus can limit the overall performance of the CFmMIMO. In large networks, where each UE primarily interferes with a small subset of neighbors, many UEs could achieve significantly higher rates without greatly affecting those with the worst conditions. In such cases, maximizing the sum of the UEs' rates may be a more effective strategy, as we will discuss in the following subsection.

3.4.2 Max-sum Rate

In this subsection, we focus on maximizing the sum rates, which quantifies the total number of bits transmitted in the network, regardless of their distribution among UEs. The sum rate maximization problem is formulated as follows:

$$\underset{p_1,\ldots,p_M}{\text{maximize}} \quad \sum_{j=1}^{M} B_\tau \log_2 \left(1 + \frac{\bar{A}_j p_j}{\bar{B}_j p_j + \sum_{j' \neq j}^{M} p_{j'} \tilde{B}_j^{j'} + I_M^j} \right) \tag{3.15a}$$

$$\text{subject to} \quad 0 \leq p_j \leq 1, \quad j \in [M]. \tag{3.15b}$$

Algorithm 3.3 shows the steps of obtaining a local optimum for optimization problem (3.15), followed by the Algorithm 7.2 in [2].

These metrics, max-min rate, and max-sum rate are well-suited for FL because they help ensure fair resource allocation among users with similar-sized FL models, optimizing communication efficiency. The max-min rate approach guarantees

Algorithm 3.3 Max-sum rate power allocation

Inputs: M, A_p, N, $\{\bar{A}_j, \bar{B}_j, \tilde{B}_j^{j'}, I_M^j\}_{j,j' \in [M]}$

Initialization: $\boldsymbol{p}$, ϵ_S, $\{u_j = d_j = \bar{p}_j = 0\}_{j \in [M]}$, $F_{O,n} = 1$, $F_{O,p} = 0$

while $F_{O,p} - F_{O,n} < \epsilon_S$, **do** ▷ While the objective improvement is less than ϵ_S

  Set $F_{O,p} \leftarrow F_{O,n}$

  $u_j \leftarrow \sqrt{\bar{A}_j p_j} \left(\bar{A}_j p_j + \bar{B}_j p_j + \sum_{j' \neq j}^{M} p_{j'} \tilde{B}_j^{j'} + I_M^j \right)^{-1}, \; j \in [M]$

  $d_j \leftarrow \left(u_j^2 \left(\bar{A}_j p_j + \bar{B}_j p_j + \sum_{j' \neq j}^{M} p_{j'} \tilde{B}_j^{j'} + I_M^j \right) - 2u_j \sqrt{\bar{A}_j p_j} + 1 \right)^{-1}, \; j \in [M]$

  $\bar{p}_j \leftarrow \bar{A}_j d_j^2 u_j^2 \left(\bar{A}_j d_j u_j^2 + \sum_{i=1}^{M} \left(\bar{B}_i + \sum_{j' \neq i}^{M} \tilde{B}_i^{j'} \right) d_i u_i^2 \right)^{-2}, \; j \in [M]$

  Set $p_j \leftarrow \min\{1, \bar{p}_j\}$

  Update $F_{O,n} \leftarrow \sum_{j=1}^{M} (d_j)^{-1}$

end while

return $\boldsymbol{p}$

that all users experience a minimum data rate, preventing any user from being a communication bottleneck. Meanwhile, the max-sum rate maximizes the total communication rate across all users, improving the overall system performance. These characteristics make these metrics particularly effective in FL scenarios where users have comparable data upload needs.

3.5 Downlink Process in CFmMIMO

Here, we investigate the possible scenarios for downlink in FL training. These scenarios are Broadcast, Unicast, and Multicast transmissions.

3.5.1 Broadcast

Broadcasting refers to the process of transmitting the same content simultaneously to multiple users or devices, often without any direct interaction with the receivers. It is commonly used in various applications, such as venue casting, media distribution, software updates, and the dissemination of breaking news [69]. Broadcasting allows the efficient delivery of information to a large audience, making it an ideal solution for scenarios where the same content needs to be shared with numerous recipients at once.

In a CFmMIMO system, broadcast transmission can be a highly effective method for distributing the global FL model to users in the downlink. This approach leverages the spatial diversity and high spectral efficiency of massive MIMO to

transmit the same information, such as broadcast messages or model updates, to all users without requiring precise channel state information (CSI). Since the global model is the same for all users, there is no need to customize the signal for individual channels, simplifying the transmission process.

Omnidirectional transmission, in particular, is used in broadcasting, where APs transmit signals uniformly in all directions without focusing on specific users. This method does not require CSI, making it particularly useful in scenarios such as emergency alerts or system initialization signals, where uniform coverage across a large area is essential. In a CFmMIMO system, the inherent spatial diversity of distributed APs ensures that users in different locations can still receive the broadcasted model with reliable performance, especially in dense deployments or when omnidirectional transmission is employed.

However, several challenges arise with broadcast transmission. Ensuring uniform signal quality across users with varying channel conditions is one key challenge. Additionally, efficient power management is needed to avoid coverage gaps, and interference must be mitigated in cases where multiple broadcasts overlap. In environments with poor AP density or severe channel fading, the lack of CSI can lead to degraded model delivery for certain users, potentially impacting the overall performance of FL. Despite these challenges, broadcast transmission remains a promising and scalable approach for distributing FL models in CFmMIMO systems, offering an efficient means of communication while reducing the need for complex signal customization.

3.5.2 Multicast

Multicasting, which involves the simultaneous delivery of common data to multiple users [70, 71], is crucial for group-oriented services such as live video streaming, virtual reality, software updates, and IoT applications. This makes the development of efficient and scalable multicast solutions a key focus in the context of 6G networks. In particular, CFmMIMO stands out as a promising backbone technology to address the challenges posed by next-generation multicast services. Its superior energy efficiency [72] aligns well with the multicast goal of minimizing unnecessary transmissions, contributing to a more sustainable communication ecosystem. Additionally, multicasting can enhance the reliability of these services by reducing the impact of network congestion and failures. In the downlink of CFmMIMO, multicasting differs from broadcasting as it aims to ensure that all intended users can reliably decode the transmitted data, which requires CSI. In contrast, broadcasting transmits the same data indiscriminately to all users without considering individual channel conditions.

In the FL downlink multicast scenario, instead of broadcasting the global FL model to all users uniformly, the system groups users based on their channel conditions or spatial locations and transmits the model to these groups via optimized multicast beams. Multicast transmission can be effectively used to send global FL models in the downlink of CFmMIMO by grouping users with similar channel char-

acteristics or spatial proximity and transmitting to these groups using optimized multicast beams. This approach improves spectral and energy efficiency compared to broadcast by focusing transmission energy on specific user groups, reducing redundancy and interference. It is useful because FL requires all users to receive the same global model, making it feasible to serve users in clusters with shared beams. However, the potential challenges include the need for accurate CSI to design multicast beams, computational complexity in user grouping and beamforming, and fairness issues, as users in weaker channel groups may experience lower quality. Additionally, the sequential nature of serving multiple groups can introduce latency, potentially impacting time-sensitive FL applications.

3.5.3 Unicast

In the downlink unicast scenario, the data is sent from the APs to each individual user separately [71]. Unlike multicast or broadcast, unicast focuses on delivering the model specifically to one user at a time, allowing for individualized transmission optimization. Unicast transmission can be used to send global FL models in the downlink to users in a CFmMIMO system by individually optimizing the signal for each user. This approach works because unicast allows for tailored beamforming and power control based on the specific channel conditions of each user, ensuring high reliability and minimal interference. It provides a highly efficient and reliable method for delivering the FL model, as each user receives the best possible signal quality. However, the potential challenges include high overhead, as each user is served individually, which can lead to increased latency and resource consumption. Additionally, unicast requires accurate and real-time CSI for each user, leading to signaling complexity and feedback overhead, and may face scalability issues in large networks with many users. transmit data when the number of users is high, which is inefficient and wasteful.

3.6 Summary

This chapter provided an overview of CFmMIMO networks and their application in training FL models. It began by introducing the background and general architecture of CFmMIMO systems. The discussion then covered the uplink process, including pilot transmission, theoretical aspects of channel estimation, and data transmission. Subsequently, state-of-the-art uplink power control schemes, which serve as benchmarks for the power control approaches proposed in this thesis, were reviewed. Finally, three downlink transmission scenarios—broadcast, multicast, and unicast—were explored, highlighting their potential applications for transmitting global FL models to the UEs in CFmMIMO networks.

Chapter 4

Optimization Methods

UNDERSTANDING optimization methods is pivotal in ML and FL, as they form the backbone for designing efficient algorithms that balance performance and resource constraints. In FL training, where users operate under strict power and energy limitations, optimization is essential to ensure efficient resource allocation and successful model training. It provides systematic approaches to achieve desired objectives while satisfying the practical challenges of distributed and energy-constrained systems. Accordingly, this chapter provides an essential foundation for the methods we use to solve the optimization problems in this thesis, such as the Bisection method, Brent method, linear programming (LP), and sequential quadratic programming (SQP).

4.1 Bisection

The Bisection method [73] is a root-finding algorithm used to locate a zero of a continuous function $g(x)$, where $x \in \mathbb{R}$, over a specified interval $[a, b]$. This method relies on the Intermediate Value Theorem [74], which states that if a continuous function changes sign over an interval, then at least one root must lie within (a, b). The Bisection method is particularly effective for monotonic functions, where the function either strictly increases or decreases, as this guarantees the existence of at most one zero within the interval.

The Bisection method operates by iteratively halving the interval $[a, b]$ to home in on the point where $g(x) = 0$. At each step, the midpoint c of the interval is calculated, and the sign of $g(c)$ is used to determine which half of the interval contains the root. The process repeats until the interval is sufficiently small, with the final midpoint c serving as an approximation of the root. The Bisection method proceeds as follows:

1. Initialization: Begin with a continuous function $g(x)$ defined over an interval $[a, b]$, where $g(a) \cdot g(b) < 0$ (indicating a sign change);

2. Compute the midpoint as $c = \frac{a+b}{2}$;
3. Evaluate the function at the midpoint, $g(c)$;
4. Update the interval based on the sign of $g(c)$, as:
   - If $g(a) \cdot g(c) < 0$, set $b \leftarrow c$,
   - If $g(c) \cdot g(b) < 0$, set $a \leftarrow c$.
5. Repeat steps 2–4 until $|b - a| < \epsilon$, where ϵ is a pre-specified tolerance level;
6. Output the zero-point: Set the root approximation as $x^* \leftarrow c$.

The Bisection method is valued for its simplicity, guaranteed convergence under the sign-change condition, and independence from derivative information, making it a robust choice for finding zero points in monotonic functions. The steps of the Bisection method are presented in Algorithm 4.1.

Drawbacks and Limitations

Despite its simplicity, the Bisection method has several limitations that reduce its effectiveness for optimization. First, its linear convergence rate is slower than that of other techniques, such as Newton's method. The Bisection method often requires many iterations to achieve precision, while Newton's method needs fewer iterations because of its quadratic convergence rate. This makes the Bisection method less efficient for problems needing quick, resource-effective solutions.

The method requires monotonic behavior, ensuring that the function has a single root within the interval and that the sign change condition holds. Multiple roots or no roots in the interval can cause the method to fail or yield incorrect results. Additionally, continuity is required; functions with discontinuities or abrupt changes can undermine its accuracy, as the midpoint evaluations and sign change assumptions may become unreliable.

The initial interval selection is critical, as a poor choice can lead to unnecessary iterations or inaccurate results. Further, the method is inherently limited to one-dimensional problems and is difficult to extend effectively to higher dimensions.

For differentiable functions, gradient-based methods are generally faster and more accurate, making the Bisection method less efficient when derivatives are available.

4.2 Brent Method

Brent's method is a hybrid root-finding algorithm that integrates the Bisection Method, the Secant Method, and Inverse Quadratic Interpolation. It combines the Bisection Method's reliability with the other methods' faster convergence. The algorithm initially attempts to use the more efficient Secant Method or Inverse

Algorithm 4.1 Bisection Method

Inputs: Function $g(.)$, interval $[a, b]$, $g(a)$, $g(b)$, ϵ, $I^{\max}$
for round $= 1, \ldots, I^{\max}$, **do** ▷ Bisection's main loop
    $c \leftarrow (a + b)/2$
    Calculate $g(c)$
    if $g(b) \cdot g(c) < 0$, **then** ▷ Update the function values
        $a \leftarrow c$
    else if $g(a) \cdot g(c) < 0$, **then**
        $b \leftarrow c$
    end if
    if $|b - a| < \epsilon$, **then** ▷ Check for the convergence
        Break and return c
    end if
end for
return $x^* \leftarrow c$

Quadratic Interpolation but reverts to the Bisection Method when necessary to maintain robustness. Developed by Richard Brent, this method builds upon an earlier algorithm by Theodorus Dekker and is sometimes referred to as the Brent-Dekker method [Chapter 5 of [75]].

4.2.1 Brent's Method for Minimization

Brent's Method for minimization is a robust and efficient optimization algorithm that combines elements of the Bisection Method, the Secant Method, and Inverse Quadratic Interpolation. Originally developed by Richard Brent [75], it was designed to provide a fast and reliable solution to unconstrained scalar optimization problems, particularly when derivatives are unavailable or difficult to compute. The method is well-suited for finding local minima of a continuous function $g(x)$ over an interval $[a, b]$, even if the function is not strictly unimodal.

Brent's Method operates by iteratively refining the search interval to locate the minimum. The algorithm utilizes the Bisection Method for robustness alongside the faster convergence techniques of the Secant Method and Inverse Quadratic Interpolation when applicable. The method follows these main steps:

1. **Initialization**: Given an initial interval $[a, b]$, the function $g(x)$ is evaluated at three points: a, b, and an interior point c, chosen to ensure that the function changes sign across the interval (i.e., a minimum must exist).

2. **Secant Method or Inverse Quadratic Interpolation**: The algorithm uses the Secant Method or Inverse Quadratic Interpolation to estimate the minimum by fitting either a secant line or quadratic approximation to the function values. It allows for faster convergence, compared to the Bisection method, in regions where the function is well-approximated by these methods.

Algorithm 4.2 Brent's Method for Minimization

Inputs: Function $g(.)$, interval $[a, b]$, $g(a)$, $g(b)$, ϵ_B, $I^{\max}$
Initialize: $c = b$, $g(c) = g(b)$
if $g(b) > g(a)$ **then** ▷ Ensure $g(b) < g(a)$
    Swap a and b
    Swap $g(a)$ and $g(b)$
end if
for round $= 1, \ldots, I^{\max}$, **do** ▷ Brent's main loop
    if $g(a) - g(b) > \epsilon_B$, **then**
        if $a \neq c$ **and** $g(a) \neq g(c)$ **and** $g(b) \neq g(c)$, **then**
            Attempt inverse quadratic interpolation of (4.2)
        else
            Use secant method of (4.1)
        end if
        if $s \notin ((3a + b)/4, b)$ **or** $(g(b) > g(a)$ **and** $s - b \geq (b - a)/2)$ **or** $(g(b) < g(a)$ **and** $s - b \geq (b - c)/2)$, **then**
            $s \leftarrow (a + b)/2$ ▷ Fallback to bisection method
        end if
        if $|s - b| < \epsilon_B$, **then** ▷ Check for the convergence
            return b
        end if
        Calculate $g(s)$
        if $g(s) < g(b)$, **then** ▷ Update the function values
            $a \leftarrow b$, $g(a) \leftarrow g(b)$
            $b \leftarrow s$, $g(b) \leftarrow g(s)$
        else
            if $g(s) < g(a)$, **then**
                $c \leftarrow a$, $g(c) \leftarrow g(a)$
                $a \leftarrow s$, $g(a) \leftarrow g(s)$
            else
                $c \leftarrow s$, $g(c) \leftarrow g(s)$
            end if
        end if
        if $|b - a| < \epsilon_B$, **then** ▷ Check for the convergence
            return b
        end if
    end if
end for
return $x^* \leftarrow b$

The Secant method, which is based on a linear approximation, considers two points a and b with the corresponding function values $g(a)$ and $g(b)$, and identifies a new potential minimum point as follows:

$$s \leftarrow b - g(b) \cdot \frac{b - a}{g(b) - g(a)}. \tag{4.1}$$

The inverse quadratic interpolation considers three points a, b, and c and the corresponding function values and then computes the point s as

$$s \leftarrow a \cdot \frac{g(b) \cdot g(c)}{(g(a) - g(b)) \cdot (g(a) - g(c))} + b \cdot \frac{g(a) \cdot g(c)}{(g(b) - g(a)) \cdot (g(b) - g(c))} + c \cdot \frac{g(a) \cdot g(b)}{(g(c) - g(a)) \cdot (g(c) - g(b))}. \quad (4.2)$$

3. **Fallback to Bisection**: If the interpolation methods fail to provide a satisfactory approximation or lead to out-of-bound points, the algorithm reverts to the more robust Bisection method, i.e. $s \leftarrow (a + b)/2$, ensuring that the search interval is always reduced and remains valid.To be more specific, the condition $s \notin ((3a + b)/4, b)$ ensures that s is sufficiently far from b. If s is too close to b, the interval reduction may be inadequate, leading to slow convergence. This condition forces s to be a meaningful step towards reducing the interval. Additionally, the conditions $s - b \geq (b - a)/2$ (when $g(b) > g(a)$) and $s - b \geq (b - c)/2$ (when $g(b) < g(a)$) ensure that the step s provides a significant reduction in the interval. If s is too close to b, it indicates that the interpolation step is not making sufficient progress towards the minimum. By enforcing these conditions, the algorithm avoids taking steps that would result in negligible progress.

4. **Convergence**: Defining ϵ_B as the pre-set tolerance level, the method terminates when the interval becomes sufficiently small, i.e., $|b-a| < \epsilon_B$, indicating that the minimum has been located within the desired precision.

The steps of Brent's method for minimization are summarized in Algorithm 4.2, indicating the main loop with maximum $I^{\max}$ rounds, the conditions for Secant or Quadratic methods, and fall-back to Bisection method. Finally, the algorithm checks for the convergence and returns the optimal solution $x^* \leftarrow b$.

Advantages and Limitations

Brent's Method is advantageous for one-dimensional optimization due to its guaranteed convergence, where the interval reliably shrinks with each iteration, ensuring that the method eventually locates a minimum, even for challenging functions. Its efficiency is enhanced by combining the robustness of the Bisection Method with the faster convergence offered by Secant and Inverse Quadratic Interpolation, making it significantly faster than Bisection alone in well-behaved regions. Additionally, Brent's Method does not rely on derivative information, which is particularly useful for optimizing non-differentiable or noisy functions. However, it has limitations, particularly in extending beyond one dimension; while it performs well in scalar optimization, adapting the method to higher dimensions is complex.

4.3 Linear Programming (LP) for Optimization

Linear programming (LP) is an optimization technique that focuses on maximizing or minimizing a linear objective function subject to constraints expressed as linear equalities and inequalities. The set of all possible solutions, known as the feasible region, forms a convex polytope—essentially a geometric shape defined by the intersection of several half-spaces, each represented by a linear inequality. Within this polytope, the objective function is a real-valued linear function. A linear programming algorithm seeks a point within the feasible region where the objective function reaches its maximum or minimum value, provided such a solution exists. The standard form of an LP problem [38] is as follows:

$$\begin{aligned} \underset{\boldsymbol{x}}{\text{minimize}} \quad & \boldsymbol{c}^\top \boldsymbol{x} && \text{(4.3a)} \\ \text{subject to} \quad & \boldsymbol{A}^\top \boldsymbol{x} \leq \boldsymbol{b} && \text{(4.3b)} \\ & \boldsymbol{x} \geq \boldsymbol{0}, && \text{(4.3c)} \end{aligned}$$

where $\boldsymbol{x} \in \mathbb{R}^n$ is the vector of decision variables, $\boldsymbol{c} \in \mathbb{R}^n$ is the cost vector, $\boldsymbol{A} \in \mathbb{R}^{n \times m}$ is the constraint matrix, $\boldsymbol{b} \in \mathbb{R}^m$ is the constraint-bound vector, and $\boldsymbol{c}^\top \boldsymbol{x}$ is the objective function.

Solution Approach for Optimization Problem (4.3)

To solve (4.3), we first convert the inequalities to equalities by introducing slack variables $\boldsymbol{s} \in \mathbb{R}^m$, $\boldsymbol{s} \geq \boldsymbol{0}$. Then, we have

$$\boldsymbol{A}^\top \boldsymbol{x} + \boldsymbol{s} = \boldsymbol{b}, \tag{4.4}$$

which transforms (4.3) into an equivalent form with equality constraints, which is necessary for most analytical approaches, such as the simplex method [76]. Next, we apply the simplex method, starting with a feasible basic solution in which some variables are set to zero (non-basic variables) while others (basic variables) satisfy the constraints in (4.4). The goal is to iteratively move from one vertex of the feasible region (a convex polytope) to another, improving the objective value $\boldsymbol{c}^\top \boldsymbol{x}$ with each step. The process begins by converting inequality constraints into equalities through the addition of slack variables, allowing all constraints to be expressed in matrix form. This transformation enables us to set up the simplex tableau—a tabular representation of the objective function and constraints. Starting from an initial feasible vertex (often where slack variables equal the constraint values and decision variables are zero), the simplex method then identifies the most promising direction to move to minimize the objective function.

The simplex method is efficient for many LP problems, though it can encounter challenges with certain degenerate or large-scale cases, where alternative methods such as the dual simplex or interior-point methods [77] might be preferable.

4.4 Sequential Quadratic Programming (SQP)

Sequential Quadratic Programming (SQP) is a widely used and highly effective optimization technique for solving constrained nonlinear optimization problems [78]. It is particularly valued for its ability to handle both equality and inequality constraints while ensuring rapid convergence under suitable conditions. The approach leverages iterative methods, solving a series of quadratic subproblems that approximate the original nonlinear problem. Due to its versatility and robustness, SQP has found applications in various fields, including engineering design, control systems, and machine learning. The main optimization problem that we solve with SQP is as follows:

$$\begin{aligned} \underset{\boldsymbol{x}}{\text{minimize}} \quad & g(\boldsymbol{x}) && \text{(4.5a)} \\ \text{subject to} \quad & h_k(\boldsymbol{x}) \leq 0, \quad k = 1, \ldots, m && \text{(4.5b)} \\ & c_j(\boldsymbol{x}) = 0, \quad j = 1, \ldots, p. && \text{(4.5c)} \end{aligned}$$

Here, $h_k(\boldsymbol{x}), k = 1, \ldots, m$ are inequality constraint functions, while $c_j(\boldsymbol{x}), j = 1, \ldots, p$ represent equality constraint functions. These functions are assumed to be at least continuously differentiable to enable the use of gradient-based approximations, a key component of the SQP method.

Since SQP is an iterative method, in the following subsection we outline the key steps in every round.

Key Steps of SQP

Considering a maximum number of $I_{\max}$ rounds, at every round i we perform the following steps:

1. **Linearize the Constraints**: We linearize the constraints around the current point $\boldsymbol{x}^i$, as

   - For inequality constraints $h_k(\boldsymbol{x}) \leq 0$, we have
   $$h_k(\boldsymbol{x}^i) + \nabla h_k(\boldsymbol{x}^i)^\top \left(\boldsymbol{x} - \boldsymbol{x}^i\right) \leq 0, \quad k = 1, \ldots, m. \tag{4.6}$$
   - For equality constraints $c_j(\boldsymbol{x}) \leq 0$, we have
   $$c_j(\boldsymbol{x}^i) + \nabla c_j(\boldsymbol{x}^i)^\top \left(\boldsymbol{x} - \boldsymbol{x}^i\right) = 0, \quad j = 1, \ldots, p. \tag{4.7}$$

2. **Hessian and Lagrangian functions**: We define $\boldsymbol{H}^i := \nabla_{xx}\mathcal{L}(\boldsymbol{x}, \boldsymbol{\lambda}, \boldsymbol{\mu})$ as the Hessian of the Lagrangian function $\mathcal{L}(\boldsymbol{x}, \boldsymbol{\lambda}, \boldsymbol{\mu})$, defined as:

   $$\mathcal{L}(\boldsymbol{x}, \boldsymbol{\lambda}, \boldsymbol{\mu}) := g(\boldsymbol{x}) + \sum_{k=1}^{m} \lambda_k h_k(\boldsymbol{x}) + \sum_{j=1}^{p} \mu_j c_j(\boldsymbol{x}), \tag{4.8}$$

   where $\boldsymbol{\lambda} := [\lambda_1, \ldots, \lambda_m]^\top$, and $\boldsymbol{\mu} := [\mu_1, \ldots, \mu_p]^\top$ are Lagrange multipliers for the equality and inequality constraints, respectively.

3. **Updating the Hessian with BFGS**: In the context of SQP for solving constrained optimization problems, the Broyden–Fletcher–Goldfarb–Shanno (BFGS) method is often used to approximate the Hessian of the Lagrangian, $\boldsymbol{H}^i$, which is essential in solving the optimization problem. Instead of computing the exact Hessian at each step, which is computationally costly, SQP with BFGS uses a quasi-Newton update to approximate $\boldsymbol{H}^i$ with $\bar{\boldsymbol{H}}^i$. The BFGS update for the Hessian approximation $\bar{\boldsymbol{H}}^i$ is

$$\bar{\boldsymbol{H}}^{i+1} = \bar{\boldsymbol{H}}^i - \frac{\bar{\boldsymbol{H}}^i \bar{\boldsymbol{s}}^i (\bar{\boldsymbol{s}}^i)^\top \bar{\boldsymbol{H}}^i}{(\bar{\boldsymbol{s}}^i)^\top \bar{\boldsymbol{H}}^i \bar{\boldsymbol{s}}^i} + \frac{\boldsymbol{z}^i (\boldsymbol{z}^i)^\top}{(\boldsymbol{z}^i)^\top \bar{\boldsymbol{s}}^i}, \tag{4.9}$$

where

$$\begin{aligned} \bar{\boldsymbol{s}}^i &= \boldsymbol{x}^{i+1} - \boldsymbol{x}^i, \\ \boldsymbol{z}^i &= \nabla_x \mathcal{L}(\boldsymbol{x}^{i+1}, \boldsymbol{\lambda}, \boldsymbol{\mu}) - \nabla_x \mathcal{L}(\boldsymbol{x}^i, \boldsymbol{\lambda}, \boldsymbol{\mu}). \end{aligned} \tag{4.10}$$

4. **Formulate and Solve the Quadratic Programming (QP) Subproblem**:

   The next step is formulating the Quadratic Programming (QP) subproblem to iteratively determine the search direction $\Delta\boldsymbol{x}^i$, for $i = 1, \ldots, I^{\max}$ rounds, that minimizes the quadratic model of the objective function $g(\boldsymbol{x})$, and a linear approximation of the constraints, as

$$\begin{aligned} \underset{\Delta\boldsymbol{x}^i}{\text{minimize}} \quad & \frac{1}{2} \Delta\boldsymbol{x}^{i\top} \bar{\boldsymbol{H}}^i \Delta\boldsymbol{x}^i + \nabla_x g(\boldsymbol{x}^i)^\top \Delta\boldsymbol{x}^i && \text{(4.11a)} \\ \text{subject to} \quad & h_k(\boldsymbol{x}^i) + \nabla h_k(\boldsymbol{x}^i)^\top \Delta\boldsymbol{x}^i \leq 0, \quad k = 1, \ldots, m && \text{(4.11b)} \\ & c_j(\boldsymbol{x}^i) + \nabla c_j(\boldsymbol{x}^i)^\top \Delta\boldsymbol{x}^i = 0, \quad j = 1, \ldots, p., && \text{(4.11c)} \end{aligned}$$

   where

$$\nabla_x g(\boldsymbol{x}^i) := \left[\frac{\partial g}{\partial x_1}, \ldots, \frac{\partial g}{\partial x_n} \right]^\top \Bigg|_{\boldsymbol{x}^i},$$

   is the gradient vector of $g(\boldsymbol{x}^i)$.

5. **Armijo Line Search and Update $\boldsymbol{x}^{i+1}$**: After solving the QP subproblem (4.11) and obtaining $\Delta\boldsymbol{x}^{i*}$, we update $\boldsymbol{x}^{i+1}$, as

$$\boldsymbol{x}^{i+1} = \boldsymbol{x}^i + \bar{\alpha}^i \Delta\boldsymbol{x}^{i*}, \tag{4.12}$$

   where $\bar{\alpha}^i > 0$ is the step length determined by a line search, ensuring a sufficient decrease in g. To determine an appropriate step length $\bar{\alpha}^i$, we apply the Armijo rule, or backtracking line search, as outlined in [78, Section 3.1]. The Armijo line search process proceeds as follows:

- **Initialization**: First, the direction $\Delta \boldsymbol{x}^i$ is computed, and the initial step length is set to $\bar{\alpha}^i = 1$.
- **Function and Gradient Evaluation**: The values of $g(\boldsymbol{x}^i + \bar{\alpha}^i \Delta \boldsymbol{x}^{i*})$ and the gradient $\nabla g(\boldsymbol{x}^i)$ are then evaluated.
- **Condition Check and Adjustment**: Within a loop, we check whether the Armijo condition:

$$g(\boldsymbol{x}^i + \bar{\alpha}^i \Delta \boldsymbol{x}^{i*}) \leq g(\boldsymbol{x}^i) + 0.1 \bar{\alpha}^i \nabla g(\boldsymbol{x}^i)^\top \Delta \boldsymbol{x}^{i*} \tag{4.13}$$

  holds. The step length $\bar{\alpha}^i$ is accepted if this condition is satisfied. Otherwise, the step length is updated according to $\bar{\alpha}^i \leftarrow 0.5 \bar{\alpha}^i$, and the loop repeats until the condition is met.

This iterative process ensures that $\bar{\alpha}^i$ achieves a sufficient decrease in the objective function $g(\boldsymbol{x})$, thereby contributing to the stability and convergence of the optimization procedure.

Karush–Kuhn–Tucker (KKT) Conditions

The Karush-Kuhn-Tucker (KKT) conditions provide necessary conditions for optimality in nonlinear optimization problems with both equality and inequality constraints [78], such as optimization problem (4.5). In the context of numerical optimization methods like SQP, the KKT conditions are used to guide each iteration towards optimality. The KKT conditions are as follows:

1. **Stationary**: The gradient of the Lagrangian with respect to $\boldsymbol{x}$ must be zero at the optimal point:

$$\nabla_x \mathcal{L}(\boldsymbol{x}^*, \boldsymbol{\lambda}, \boldsymbol{\mu}) = \nabla g(\boldsymbol{x}^*) + \sum_{k=1}^{m} \lambda_k \nabla h_k(\boldsymbol{x}^*) + \sum_{j=1}^{p} \mu_j \nabla c_j(\boldsymbol{x}^*) = 0. \tag{4.14}$$

2. **Primal Feasibility**: The solution $\boldsymbol{x}^*$ must satisfy all original constraints:

$$c_j(\boldsymbol{x}^*) = 0, \quad j = 1, \ldots, p, \qquad h_k(\boldsymbol{x}^*) = 0, \quad k = 1, \ldots, m. \tag{4.15}$$

3. **Dual Feasibility**: The Lagrange multipliers associated with the inequality constraints must be non-negative:

$$\lambda_k = 0, \quad k = 1, \ldots, m. \tag{4.16}$$

4. **Complementary Slackness**: For each inequality constraint, either the constraint is active (i.e., $h_k(\boldsymbol{x}^*) = 0$) or the corresponding Lagrange multiplier is zero:

$$\lambda_k \cdot h_k(\boldsymbol{x}^*) = 0, \quad k = 1, \ldots, m. \tag{4.17}$$

The KKT conditions are fundamental for solving constrained optimization problems, ensuring solutions are both feasible and optimal. They guide iterative methods like SQP and validate solutions by checking necessary optimality conditions, forming the basis for advanced numerical optimization techniques.

4.5 Summary

This chapter covered the foundational optimization methods employed in this thesis, including the Bisection method, Brent method, linear programming, and sequential quadratic programming. These techniques were presented to address optimization challenges in distributed and energy-constrained systems, particularly in ML and FL training, under strict resource limitations.

Chapter 5

FL over Wireless Networks

REDUCING communication and computation overhead in FL over wireless networks has been extensively explored in the literature, motivated by the growing demand for processing large-scale datasets and supporting IoT applications. These networks must operate under constraints such as limited bandwidth and energy resources, which significantly influence the efficiency of FL. In this chapter, we explore methods and frameworks designed to address these challenges, providing solutions tailored to optimize FL performance in resource-constrained wireless environments. Building on this foundation, we begin by reviewing related works and the existing literature on FL over wireless networks. This is followed by a summary of Papers 1-6, as an explanation of the general system model and problem formulation central to this thesis, along with illustrative numerical results to provide further insights.

5.1 Related Works

We begin this section with an overview of the related works on FL over wireless networks. Specifically, we discuss key papers in the literature addressing communication-efficient FL in wireless settings, highlighting the research gaps that this thesis aims to address.

5.1.1 Communication-Efficient FL over Wireless Networks

A successful FL training significantly depends on developing communication-efficient approaches, particularly in large-scale FL over wireless networks [79, 80]. In the following, we bring a comprehensive literature review of communication-efficient FL methods. Moreover, we provide a summary of related research subjects and the research gaps in Table 5.1, which we fulfill in this thesis.

Due to the extensive parameter exchange between users and the server in FL, communication overhead is among the major bottlenecks of FL [81]. One of the

Table 5.1: Summary of RGs in the FL literature we fulfill in this thesis.

Research Gaps	Papers 1-6					
	1	2	3	4	5	6
RG 1: Incorporating communication-computation cost	✓	✓	✓	✗	✗	✓
RG 2: Causal approach	✓	✓	✓	✓	✓	✓
RG 3: Impact of Communication protocols	✓	✓	✗	✗	✗	✗
RG 4: Co-optimizing energy & latency in CFmMIMO	✗	✗	✗	✓	✗	✓
RG 5: Adaptive quantization/sparsification	✗	✗	✓	✗	✓	✓

promising solutions to reduce communication overhead in FL over wireless networks is to optimally allocate resources, such as bandwidth, as in [19], [24–26], [28]. The authors in [82] have considered implementing a communication-efficient FL over wireless networks, where only the CSI of the parameter server is known, and channel capacity with an outage is considered. Then, an essential problem for mobile users is selecting appropriate local processing and communication parameters, such as transmission rates to achieve the desired balance between the overall learning performance and energy consumption.

Although FL is well-known for preserving data privacy, there are still some concerns about its reliability in privacy-preserving [83]. In [84], the authors have proposed a privacy-preserving communication-efficient framework to tackle the privacy leakage and large communication overhead in federated multi-armed bandits. In this work, the authors studied the interactions among privacy, communication, and learning performance in terms of regret. One of the main approaches to reducing the communication overhead in FL over wireless networks is user selection. The user selection has been widely investigated in the literature in various scenarios, and several trade-offs have been analyzed, such as between energy, latency, and FL accuracy [85–91].

Reducing Communication Overhead by Quantization and Sparsification

The importance of reducing communication overhead in FL over wireless networks has made researchers use communication-efficient methods. Traditional communication efficient methods, such as quantization [92, 93], data compression [21] [1], and sparsification [94] have proven effective in significantly reducing communication overhead during each iteration of FL training. These methods are particularly beneficial in scenarios where communication costs dominate the overall training

[1]Data compression refers to reducing the quantity of information exchanged in terms of bits between users, resulting in the conservation of communication resources.

process. Recent research has demonstrated the effectiveness of appropriate quantization techniques combined with error feedback in preserving the convergence of the training algorithm and maintaining its asymptotic convergence rate as reported in [50]. Sparsification is another approach to decrease the volume of information transmitted between users during each iteration of a distributed learning process [95]. A well-known implementation of sparsification is the top-q method, in which only the q most significant magnitude elements of the gradient are transmitted [50]. This approach saves communication resources and optimizes the efficiency of the distributed learning process. In the following, we elaborate more on sparsification and quantization methods exploited in FL literature.

Sparsification is one of the data compression techniques that aims to transform the high-dimensional ML models to their sparse representations by pruning some relatively unimportant elements. Therefore, these methods decrease the size of the ML model parameters exchanged among users to reduce the communication overhead [94–98]. In [96], the authors have proposed a convex optimization problem to minimize the coding length of stochastic gradients in distributed optimization. Moreover, inspired by the top-q method [50], the authors in [97] have introduced SparseFed, a novel FL method that uses global top-q update sparsification and user-level gradient clipping to mitigate model poisoning attacks. Furthermore, the authors of [99] propose Tiny Federated Edge Learning (TinyFEL), which reduces local memory usage and communication-computation burdens by early backpropagation termination and a parameter splitting mechanism, transmitting only partial model updates to significantly lower communication overhead.

Recently, gradient sparsification has received a lot of attention. Such sparsification methods only update significant gradients and accumulate insignificant gradients locally. Many existing works on gradient sparsification use a fixed degree of gradient sparsity for IID-distributed data within a data center. However, the authors of [98] have considered the adaptive degree of sparsity and non-IID local datasets and presented a fairness-aware gradient sparsification method to ensure that different users provide a similar amount of updates in the ML procedure.

Another critical point in gradient sparsification is how to preserve the accuracy after a high ratio sparsification. The authors in [94] have proposed a general gradient sparsification framework for adaptive optimizers to correct the sparse gradient update process. The authors have shown that by gradient correction, the optimizer can lead to improved convergence of the model. Furthermore, the method can optimize the model by treating accumulated insignificant gradients. Additionally, incorporating local gradients in the batch normalization layer can mitigate the effect of delayed gradients without significantly increasing communication overhead.

Quantization is another data compression technique that uses fewer communication bits to transmit the ML model parameters. In large-scale ML with explosive data production, quantization is attracting much attention because of its ability to significantly reduce the communication overhead [3], [100–103]. It has been recently shown that simple sign-based quantization, together with majority voting converges to the optimal solution (under certain assumptions) and provides a highly

communication-efficient alternative in practice [101].

The work [102] has proposed a hyper-sphere quantization, which is a general framework that can be configured to achieve a continuum of trade-offs between communication efficiency and gradient accuracy. Specifically, at the high compression ratio end, the authors have shown that hyper-sphere quantization can provide a low per-iteration communication cost of $\mathcal{O}(\log(d))$, which is favorable for FL. Another interesting quantization scheme is the method proposed in [103], which introduced a federated trained ternary quantization (FTTQ) approach for user training and inference stages. Moreover, FTTQ can compress both uploading and downloading communications, effectively reducing communication costs. The authors have demonstrated that applying FTTQ to deep learning can achieve slightly better performance on non-IID data than the canonical FL algorithms.

The lazily aggregated quantized gradient (LAQ) method [3] is the quantization method we primarily utilize and investigate throughout this thesis. Our works exploit LAQ because LAQ achieves the same linear convergence as GD while considering strongly convex loss functions. Herein, applying our causal approach on top of LAQ shows that LAQ has excellent potential to improve communication resource optimization while providing a fast convergence rate like GD. Therefore, we will explain LAQ comprehensively in the next section as an essential part of the thesis.

Adaptive Quantization and Sparsification

Recent advancements in FL have increasingly focused on reducing communication overhead through adaptive quantization methods, which balance communication efficiency and model performance. For instance, the adaptive quantized gradient approach proposed in [104] effectively reduces communication costs without compromising convergence. Similarly, AdaQuantFL [105] employs dynamic quantization levels, ensuring both efficiency and low error rates during training.

Innovative methods like [106] combine data compression with deadline-based straggler exclusion to accelerate Federated Edge Learning (FEEL), optimizing parameters to minimize overall training time. Building on this, [107] introduces a descending quantization scheme to further reduce communication loads. AQUILA [49] presents an adaptive quantization strategy that lowers communication costs while ensuring model convergence. Similarly, DAdaQuant [108] adapts quantization dynamically to varying network conditions, achieving reduced communication requirements while maintaining training speed. An additional contribution is FedAQ [109], which explores adaptive quantization for both uplink and downlink communication. This method highlights the potential of dynamic and adaptive quantization to mitigate communication bottlenecks in large-scale FL training while preserving performance and convergence.

Additionally, adaptive approaches have gained prominence for enhancing communication efficiency in FL [98,110,111]. For example, [110] introduces an adaptive quantization and sparsification scheme tailored for uplink transmissions using non-

orthogonal multiple access, achieving better efficiency in multi-user environments. Similarly, [98] proposes an online learning framework to balance the trade-off between communication and computation, leveraging gradient sparsity derived from the estimated sign of the objective function's derivative. Building on this, [111] presents an adaptive gradient compression method that dynamically adjusts the compression rate based on the unique characteristics of each user, further optimizing communication efficiency without compromising model performance. These adaptive methods highlight the potential for tailoring communication strategies to the specific needs of FL systems, ensuring scalable and efficient training in resource-constrained environments.

Joint Resource Optimization

Several studies have explored joint optimization approaches in FL to improve communication and computation efficiency, focusing on user selection, transmit power, data rate, and processing frequency. For example, [112] proposes a framework that integrates these factors to minimize training latency, demonstrating the potential of coordinated resource management.

FL's decentralized nature enables users to train local models and share updates with a central server while maintaining data privacy [13], making it a valuable paradigm for distributed ML. Recognizing its advantages, researchers have focused on enhancing FL's efficiency in communication, computation, and energy consumption [19,113,114]. For instance, [19] tackles the challenge of minimizing energy consumption under strict latency constraints, offering solutions that balance energy and time requirements. Similarly, [113] examines joint power and resource allocation within ultra-reliable low-latency vehicular networks, introducing a distributed FL-based approach to estimate queue length distributions, thus enhancing network efficiency. These works highlight the critical role of optimization techniques in advancing FL for real-world applications.

The authors in [115] propose customized local training, partial user participation, and flexible aggregation to improve communication efficiency and convergence speed in FL. Similarly, [116] introduces adaptive sparsification, reducing communication overhead while maintaining accuracy and linear convergence. Adaptive sampling is explored in [117], where dynamic training data filtering and user selection effectively reduce communication costs and improve convergence rates. In [118], a distributed approximate Newton-type algorithm enhances communication efficiency in FEEL and addresses heterogeneous data challenges using the FedOVA scheme. To address distribution divergence, the authors in [119] propose a data-sharing approach integrated with FL in wireless networks, accompanied by a joint optimization algorithm to balance model accuracy and associated costs effectively. Meanwhile, [120] mitigates model unfairness by adaptively tuning the server's update direction, ensuring equitable loss reduction across users. The authors of [121] propose an adaptive model sparsification framework for FL with error compensation, dynamically optimizing the sparsification level to balance the impact of

communicated model parameters and communication cost. Meanwhile, [122] proposes an adaptive model update and robust aggregation strategy to optimize local training. Moreover, [123] introduces the FedLADA optimizer, a momentum-based algorithm that corrects local offsets using global gradient estimates, addressing convergence challenges and improving training speed. These innovations collectively address the key challenges of communication, efficiency, and fairness in FL.

FL over CFmMIMO Networks

FL offers significant advantages in reducing data communication overhead by keeping data localized; however, the energy constraints of users pose challenges in executing high-dimensional, large-scale FL training, particularly during the transmission of local models [36]. To address this, efficient resource allocation, such as time, energy, or spatial resources, becomes essential for ensuring successful FL execution. In this context, integrating FL with CFmMIMO systems has emerged as a promising approach. CFmMIMO can offer nearly uniform quality of service (QoS) across users, facilitating the efficient handling of equally sized model updates, as explored in works such as [112, 124–130].

Several studies have tackled the challenge of reducing FL latency through joint optimization frameworks. Similarly, [124] and [125] propose joint optimization of transmit power and data rate to minimize uplink latency in CFmMIMO-supported FL systems. For instance, [124] presents a solution involving user selection, transmit power, and processing frequency to mitigate latency for users with poor communication links. Moreover, it extends this by employing the successive convex approximation (SCA) technique to optimize transmit power and data rates, addressing the straggler effect in FL training.

Additionally, optimizing local model accuracy, transmit power, data rate, and user processing frequency is considered in [112] to minimize FL latency in CFmMIMO environments. Expanding on these ideas, [126] investigates power and processing frequency allocation for supporting multiple FL groups within a CFmMIMO system, aiming to optimize the latency of each FL iteration. Furthermore, [127] formulates a joint optimization framework that minimizes the maximum communication and computation latency among all participating users.

Another approach to mitigating latency is explored in [128], which introduces a user cooperation and power control method. In this setup, users with high-quality communication links assist those with lower-quality links in transferring their parameters to the central server, with an optimization problem designed to minimize the maximum transmission time among users.

Energy-efficient FL over CFmMIMO networks has garnered significant attention from researchers aiming to enable effective FL training in energy-constrained environments. For instance, [129] addresses energy-efficiency maximization under constraints on per-user power, backhaul capacity, and throughput. The study derives a closed-form expression for spectral efficiency while accounting for the impacts of channel estimation errors and quantization distortion. Similarly, [130] proposes an

Table 5.2: Definition of mathematical parameters.

Parameter	Definition	Parameter	Definition
b	Number of bits	K	Number of global iterations
M	Number of users	$\boldsymbol{w}$	FL model
d	Size of FL model	r	Quantization grid radius
R	Uplink data rate	$\bar{L}$	Lipschitz parameter
ℓ	Latency	E	Energy
c	Iteration cost	p	Power
C	Total iteration cost	L	Number of local iterations
p_x	Transmission probability	p_r	Arrival probability
$\bar{\mathcal{L}}$	Uplink latency budget	$\mathcal{E}, \bar{\mathcal{E}}$	Uplink energy budget

energy-efficient FL framework tailored for cell-free internet-of-things networks. The authors focus on minimizing the total energy consumption of FL users by formulating and solving an optimization problem that yields an optimal central processing unit frequency allocation and a sub-optimal power allocation strategy. These contributions underscore the importance of integrating energy-efficient designs into CFmMIMO-supported FL systems to ensure sustainable FL training.

Other studies have explored various dimensions of integrating CFmMIMO with FL. For example, [131] investigates how wireless transmission errors influence the FL model's parameter updating process within CFmMIMO networks. This work highlights the importance of addressing transmission errors to ensure accurate and reliable FL training. Meanwhile, [132] focuses on decentralized precoding within CFmMIMO frameworks, proposing two novel precoding schemes designed to enhance FL's performance. These contributions demonstrate the breadth of research addressing the interplay between CFmMIMO and FL, spanning theoretical analyses and practical algorithmic developments.

The studies mentioned above have aimed to reduce FL communication overhead through adaptive quantization grids or optimal resource allocation, such as bits, power, or frequencies. However, these approaches applied uniform quantization across the entire local gradient vector without leveraging the sparsity inherent in local gradients, as highlighted in [43, 133, 134], where many elements are near zero. To address this limitation, this thesis introduces two novel quantization schemes, which are detailed later in this chapter.

In this thesis, we use two well-established quantization schemes from the literature, LAQ and AQUILA, as benchmarks to assess and compare the performance of our proposed quantization methods. The following subsections offer a brief overview of these benchmark schemes. For clarity and ease of reference, the mathematical parameters used throughout this chapter are listed in Table 5.2.

5.1.2 Lazily Aggregated Quantized Gradient (LAQ)

Lazily aggregated quantized gradients (LAQ) method [3] is a quantization framework that quantizes the gradient innovation, which is defined as the difference between the current gradient and the previous quantized gradient. Furthermore, LAQ achieves the same linear convergence as the GD in strongly convex setups and saves communication resources by using fewer transmitted bits at each communication iteration. LAQ focuses on reducing the number of user-to-server uplink communications uploads. The update process in LAQ is presented as

$$\boldsymbol{w}_k = \boldsymbol{w}_{k-1} - \alpha \tilde{\nabla}_{k-1}, \quad k = 1, \ldots, K, \tag{5.1}$$

where $\tilde{\nabla}_k$ is an approximate aggregated gradient that summarizes the parameter change at iteration k, defined as

$$\tilde{\nabla}_k := \tilde{\nabla}_{k-1} + \sum_{j=1}^{M} \delta \boldsymbol{q}_k^j, \quad j = 1, \ldots, M. \tag{5.2}$$

Let us define $\boldsymbol{q}^j(\boldsymbol{w}_k) := \text{Quant}(\nabla_{\boldsymbol{w}} f^j(\boldsymbol{w}_k); b),\ j \in \{1, \ldots, M\}$ as the quantized version of each local gradient at the iteration k, with b as number of bits. Moreover, let us denote by $\delta \boldsymbol{q}_k^j$, the gradient innovation, defined as

$$\delta \boldsymbol{q}_k^j := \boldsymbol{q}^j(\boldsymbol{w}_k) - \boldsymbol{q}^j(\boldsymbol{w}_{k-1}), \quad j = 1, \ldots, M, \tag{5.3}$$

which is the difference between two consecutive quantized gradients of $f^j(\boldsymbol{w})$. Furthermore, each local gradient is element-wise quantized by projecting to the closest point in a uniformly discretized d-dimensional grid with a radius of r_k^j, defined as

$$r_k^j := \|\nabla_{\boldsymbol{w}} f^j(\boldsymbol{w}_k) - \boldsymbol{q}^j(\boldsymbol{w}_{k-1})\|_\infty, \tag{5.4}$$

where for any vector $\boldsymbol{a} \in \mathbb{R}^d$, the ℓ_∞-norm $\|.\|_\infty : \mathbb{R}^d \to \mathbb{R}$ is defined as

$$\|\boldsymbol{a}\|_\infty := \max_{i \in \{1, \ldots, d\}} \{|[\boldsymbol{a}]_i|\}. \tag{5.5}$$

We define the quantization granularity $\tau := 1/(2^b - 1)$. Then, each element of $\boldsymbol{q}_k^j(\boldsymbol{w}_k)$ is derived as

$$[\boldsymbol{q}_k^j(\boldsymbol{w}_k)]_i = \left\lfloor \frac{\left[\nabla_{\boldsymbol{w}} f^j(\boldsymbol{w}_k)\right]_i - \left[\boldsymbol{q}^j(\boldsymbol{w}_{k-1})\right]_i + r_k^j}{2\tau r_k^j} + \frac{1}{2} \right\rfloor, \quad i = 1, \ldots, d, \tag{5.6}$$

where $[\boldsymbol{q}_k^j(\boldsymbol{w}_k)]_i \in \left\{0, 1, \ldots, 2^{b-1}\right\}, i = 1, \ldots, d$. Consequently, according to (5.3), the gradient innovation at each iteration $k = 1, \ldots, K$ is obtained as

$$\delta \boldsymbol{q}_k^j = \boldsymbol{q}^j(\boldsymbol{w}_k) - \boldsymbol{q}^j(\boldsymbol{w}_{k-1}) = 2\tau r_k^j \boldsymbol{q}_k^j(\boldsymbol{w}_k) - r_k^j \mathbf{1}, \quad j = 1, \ldots, M, \tag{5.7}$$

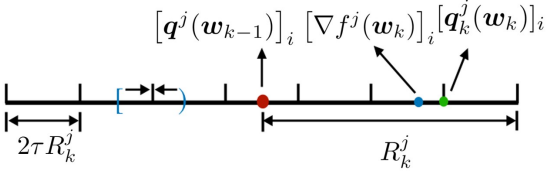


Figure 5.1: Illustration of LAQ with $b = 3$. The original value is quantized with 3 bits and $2^3 = 8$ values, each of which covers an interval of length $2\tau r_k^j$ centered at itself [3].

where $\mathbf{1} \in \mathbb{R}^d$ and $\mathbf{1} := [1, \ldots, 1]^\top$.

Let us define $\varepsilon_k^j := \nabla_{\boldsymbol{w}} f^j(\boldsymbol{w}_k) - \boldsymbol{q}^j(\boldsymbol{w}_k)$ as the local quantization error, in which the aggregated quantization error is obtained as $\varepsilon_k := \sum_{j=1}^M \varepsilon_k^j$, and the aggregated quantized gradient is defined as $\boldsymbol{q}_k := \sum_{j=1}^M \boldsymbol{q}^j(\boldsymbol{w}_k)$. Moreover, it is clear that the quantization error is not larger than half of the length of the interval that each value covers, as

$$\|\varepsilon_k^j\|_\infty \leq \tau r_k^j. \tag{5.8}$$

Figure 5.1 shows an example of quantizing one coordinate of a gradient with $b = 3$ bits. The original value is quantized with 3 bits and $2^3 = 8$ values, each of which covers an interval of length $2\tau r_k^j$ centered at itself. According to Figure 5.1, We can observe that inequality (5.8) is held about the quantization error.

In this thesis, we assume that: all the users participate in the training, each local loss function $f^j(\boldsymbol{w}_k)$ is $\bar{L}_j$-smooth, and that the loss function $f(\boldsymbol{w}_k)$ is both $\bar{L}$-smooth and μ-strongly convex. Under these assumptions, the LAQ update yields the following descent behavior:

$$f(\boldsymbol{w}_{k+1}) - f(\boldsymbol{w}_k) \leq \alpha\|\varepsilon_k\|_2^2 - \frac{\alpha}{2}\|\nabla_{\boldsymbol{w}} f(\boldsymbol{w}_k)\|_2^2 + \left(\frac{\bar{L}}{2} - \frac{1}{2\alpha}\right)\|\boldsymbol{w}_k - \boldsymbol{w}_{k+1}\|_2^2, \tag{5.9}$$

where by choosing $\alpha = 1/\bar{L}$, we can simplify the inequality (5.9) to

$$f(\boldsymbol{w}_{k+1}) - f(\boldsymbol{w}_k) \leq \alpha\|\varepsilon_k\|_2^2 - \frac{\alpha}{2}\|\nabla_{\boldsymbol{w}} f(\boldsymbol{w}_k)\|_2^2. \tag{5.10}$$

We refer the interested readers to reference [3] for all the mathematical proofs.

Recall that the communication efficiency in LAQ comes from reducing the number of bits. Specifically, from $32d$ bits to $bd + 32$ bits required to transmit at each iteration k, as bd number of bits for sending the gradient innovation $\delta\boldsymbol{q}_k^j$ and 32 bits for sending r_k^j to the server. In this thesis, we show that applying our causal approach on top of LAQ saves many communication bits with a negligible loss in test accuracy. Furthermore, we show that a dynamic choice of the number of bits can improve communication efficiency in LAQ.

5.1.3 AQUILA

AQUILA is an adaptive FL compression method that simultaneously adjusts the communication frequency and the quantization precision in a synergistic way [49]. This quantization method dynamically assigns element-wise bits to every user at each iteration, aiming to minimize the model deviation from the full-precision FL model. Defining the quantized version of the local gradient as $\boldsymbol{q}_k^j$, the global model update in AQUILA is as follows

$$\boldsymbol{w}_{k+1} = \boldsymbol{w}_k - \frac{\alpha}{M} \sum_{j=1}^{M} \boldsymbol{q}_k^j, \tag{5.11}$$

where

$$\Delta \boldsymbol{q}_k^j := \mathcal{Q}\left(\nabla_w f^j(\boldsymbol{w}_k) - \boldsymbol{q}_{k-1}^j\right), \tag{5.12}$$

$$\boldsymbol{q}_k^j := \boldsymbol{q}_{k-1}^j + \Delta \boldsymbol{q}_k^j, \tag{5.13}$$

and $\mathcal{Q} : \mathbb{R}^d \to \mathbb{R}^d$ denotes the AQUILA quantization operator. Hereafter, every element of $\Delta \boldsymbol{q}_k^j$ is obtained as

$$\left[\Delta \boldsymbol{q}_k^j\right]_i = \left\lfloor \frac{\left[\nabla_w f^j(\boldsymbol{w}_k)\right]_i - \left[\boldsymbol{q}_{k-1}^j\right]_i + r_k^j}{2\tau_k^j r_k^j} + \frac{1}{2} \right\rfloor, \quad i = 1, \ldots, d, \tag{5.14}$$

where $r_k^j := \|\nabla_w f^j(\boldsymbol{w}_k) - \boldsymbol{q}_{k-1}^j\|_\infty$ is the quantization range (radius), τ_k^j is the adaptive quantization granularity for each user $j = 1, \ldots, M$ at the iteration k, obtained as:

$$\tau_k^j := \frac{1}{2^{b_k^j} - 1}, \quad j = 1, \ldots, M, \ k = 1, \ldots, K, \tag{5.15}$$

where b_k^j is the adaptive number of element-wise bits. To obtain the optimal b_k^j, the authors in [49] have solved the following optimization problem:

$$\underset{\tau_k^j}{\text{minimize}} \quad \left(\|\nabla_w f^j(\boldsymbol{w}_k) - \boldsymbol{q}_{k-1}^j\|_2 - \|\tau_k^j r_k^j \mathbf{1}\|_2\right)^2 \tag{5.16a}$$

$$\text{subject to} \quad \tau_k^j = \frac{1}{2^{b_k^j} - 1}, \quad j = 1, \ldots, M, \ k = 1, \ldots, K. \tag{5.16b}$$

After solving the optimization problem (5.16), the optimal solutions are as follows:

$$(\tau_k^j)^* = \frac{\|\nabla_w f^j(\boldsymbol{w}_k) - \boldsymbol{q}_{k-1}^j\|_2}{r_k^j \sqrt{d}}, \tag{5.17}$$

$$(b_k^j)^* = \left\lfloor \log_2 \left(\frac{\|\nabla_w f^j(\boldsymbol{w}_k) - \boldsymbol{q}_{k-1}^j\|_2}{r_k^j \sqrt{d}} + 1 \right) \right\rfloor. \tag{5.18}$$

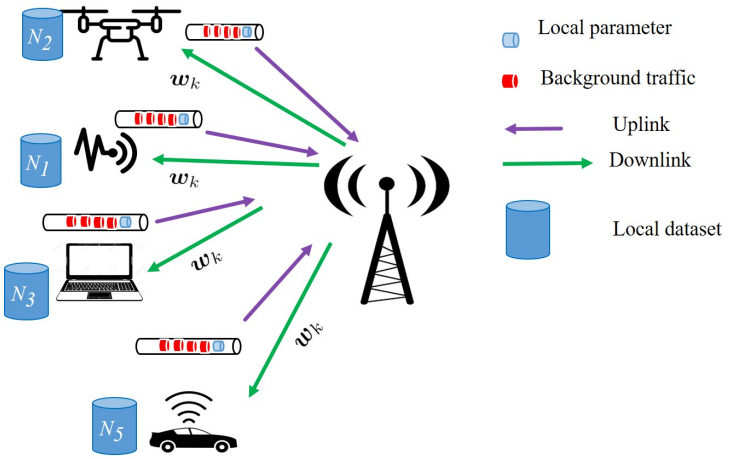


Figure 5.2: System model of FL over wireless networks. Each user performs FedAvg over its local dataset and puts the local parameter in the transmission queue with background traffic.

Similar to LAQ, the local quantization error in AQUILA is $\varepsilon_k^j := \nabla_w f^j(\boldsymbol{w}_k) - \boldsymbol{q}_{k-1}^j - \Delta \boldsymbol{q}_k^j$, and the quantization error is defined as $\varepsilon_k := \sum_{j=1}^M \varepsilon_k^j$. Similar to the assumptions in LAQ, in this thesis, we assume that all the users participate in the training, each local loss function $f^j(\boldsymbol{w}_k)$ is $\bar{L}_j$-smooth, and that the loss function $f(\boldsymbol{w}_k)$ is both $\bar{L}$-smooth and μ-strongly convex. Under these assumptions, AQUILA yields the same descent behavior as in (5.9).

5.2 Cost-efficient and Causal FL Training

In this section, we provide the system model of Papers 1-2, see Figure 5.2, and the problem formulation. Afterward, we explain the causal approach, the corresponding analytical results, and its optimality. We consider a star network consisting of M users and a server, as shown in Figure 5.2. Each user $j = 1, 2, \ldots, M$ keeps a subset $\mathcal{D}_j = \{(\boldsymbol{x}_n^j, y_n^j)\}_{n=1}^{N_j}$ of the global dataset, $\mathcal{D}$, where $\mathcal{D} = \cup_{j=1}^M \mathcal{D}_j$. Each user j computes their local parameters $\boldsymbol{w}_k^j$, while the server performs the iterations of (2.31) until a convergence criteria for $\|f(\boldsymbol{w}_k) - f(\boldsymbol{w}^*)\|$ is met [135]. We denote by K the first communication iteration at which the stopping criteria of the FedAvg algorithm is met, specifically as

$$K := \text{the first value of } k \mid \|f(\boldsymbol{w}_k) - f(\boldsymbol{w}^*)\| < \epsilon\,, \tag{5.19}$$

where $\epsilon > 0$ is the decision threshold for terminating the algorithm at communication iteration K and $f(\boldsymbol{w}^*)$ is the optimum of the loss function at $\boldsymbol{w}^*$ of (2.26).

We define $c_k > 0$, $k = 1, 2, \ldots$, as the cost of performing a complete communication iteration k of FedAvg. Accordingly, when we run FedAvg, namely an execution of (2.30) and (2.31), the training process will cost $\sum_{k=1}^{K} c_k$, which we denote as the iteration-cost function. Considering the iteration-cost function $\sum_{k=1}^{K} c_k$ we propose the following optimization problem of minimizing the loss function $f(\boldsymbol{w})$ and $\sum_{k=1}^{K} c_k$, as in [136],

$$\underset{K}{\text{minimize}} \quad \left[f(\boldsymbol{w}_K), \sum_{k=1}^{K} c_k \right] \tag{5.20a}$$

$$\text{subject to} \quad \boldsymbol{w}_k = \sum_{j=1}^{M} \rho_j \boldsymbol{w}_k^j, \quad k = 1, \ldots, K, \tag{5.20b}$$

$$\boldsymbol{w}_{0,k}^j = \boldsymbol{w}_{k-1}, \quad k = 1, \ldots, K, \tag{5.20c}$$

$$\boldsymbol{w}_{i,k}^j = \boldsymbol{w}_{i-1,k}^j - \frac{\alpha_k}{N_k^j} \sum_{n=1}^{N_k^j} \nabla_{\boldsymbol{w}} f(\boldsymbol{w}_{i-1,k}^j; \boldsymbol{x}_n^j, y_n^j), \quad k = 1, \ldots, K. \tag{5.20d}$$

Optimization problem (5.20) aims to obtain the stopping iteration K at which $f(\boldsymbol{w})$ and $\sum_{k=1}^{K} c_k$ are minimized. In the state-of-the-art literature, the threshold ϵ is independently defined before the training process. However, when we consider the communication-computation costs to solve (5.20), the threshold must be optimally designed to train the model without wasting the available limited communication-computation resources.

Note that (5.20a) represents a multi-objective function as

$$G(K) := \beta C(K) + (1 - \beta) f(\boldsymbol{w}_K), \tag{5.21}$$

where $C(K) := \sum_{k=1}^{K} c_k$, and $\beta \in (0, 1)$ is the scalarization factor of the multi-objective scalarization method [38]. Therefore, we re-write (5.20) as

$$\begin{aligned} &\underset{K}{\text{minimize}} \quad G(K) \\ &\text{subject to} \quad (5.20\text{a}) - (5.20\text{d}), \end{aligned} \tag{5.22}$$

which represents a trade-off between achievable training accuracy and the overall cost we spend for that accuracy. Accordingly, we define k^* as the iteration at which (5.22) is minimized, thus

$$\begin{aligned} k^* \quad &\in \quad \underset{K}{\arg\min} \quad G(K) \\ &\text{subject to} \quad (5.20\text{a}) - (5.20\text{d}). \end{aligned} \tag{5.23}$$

In the following lemma, we formalize that if $G(K)$ is a monotonically decreasing function of K, we can obtain k^* at which $G(K)$ is minimized.

Lemma 1. *Consider optimization problem* (5.23). *Let $G(K)$ be a non-increasing function of all $K \leq k^*$. Then, k^* indicates the index at which the sign of discrete derivative [137] of $G(K)$ changes for the first time, i.e.*

$$k^* \in \min\{K | G(K+1) - G(K) > 0\}. \tag{5.24}$$

Proof. See Appendix A-A in [136]. □

Now, the question is how to solve optimization problem (5.23). In the following, we discuss the causal and non-causal solutions for (5.23).

Proposed Causal and Non-causal Solution Approach

To solve optimization problem (5.23), in this section, we propose two solution approaches: non-causal and causal, which the detailed definitions can be found in Papers 1 and 2 of Chapter 7 of this thesis as well as in [4].

Non-causal Solution

The non-causal solution refers to the approach that requires knowing all the future information of training at the beginning. Therefore, the non-causal solution is impractical due to the need for knowledge of future iterations. However, the non-causal solutions are the optimal solutions for an optimization problem, such as (5.23), allowing us to examine the optimality of other solution approaches.

To solve optimization problem (5.23) in a non-causal setting, we can perform an exhaustive search over the discrete set of $K \in [0, +\infty)$. To this end, we should have in advance, namely at time $k = 0$, the sequence of $\{f(\boldsymbol{w}_k)\}_k$ and $\{c_k\}_k$ for all $k \in [0, +\infty)$. For analytical reasons, the non-causal setting assumes that all these values are available at the beginning of the training at $k = 0$. Although this approach is not practical, we investigate it to obtain the optimal solution k^* of optimization problem (5.23). Thus, in this thesis, the term k^* refers to the optimal solution of optimization problem (5.23), which is a non-causal solution. Moreover, we show that by considering a maximum number of the iterations for training, the optimal solution k^* of optimization problem (5.23) is finite; see the proof of Proposition 2 in [136], and in Chapter 7 of this thesis, Paper 2.

Causal Solution

Despite the non-causal solution, the system model in Figure 5.2 does not need future training information to obtain the causal solution. Herein, we denote the causal solution for (5.23) as k_c, which is an approximation of the optimal solution k^*. The following Theorem clarifies an important relation between the non-causal and causal solutions for optimization problem (5.23).

Theorem 1. *Let k^* be the solution to optimization problem (5.23), and let k_c denote the approximate solution obtained in the causal settings for optimization problem (5.23). Then, the following statements hold*

$$k_c \in \{k^*, k^*+1\}\,, \tag{5.25a}$$

$$f(\boldsymbol{w}_{k_c}) \leq f(\boldsymbol{w}_{k^*})\,, \text{ and} \tag{5.25b}$$

$$G(k_c) \geq G(k^*)\,. \tag{5.25c}$$

Proof. See Appendix A-E in [136]. □

Theorem 1 indicates that to obtain the causal solution k_c, we need to perform at most one iteration more than the optimal iteration k^*. Thus, the iteration cost $C(k_c) \leq C(k^*) + c_{k_c}$, and the training loss is less than or equal to the loss function at the optimal solution k^*. This important result highlights that the causal solution is bounded by (5.25a). Moreover, assuming that c_{k_c} is finite, we need to spend at most c_{k_c} as the extra cost to obtain k_c.

Illustrative Numerical Results

This part introduces illustrative numerical results of the causal and non-causal solutions that we have proposed. The full details are available in [4,136] as well as in Chapter 7 of this thesis, Paper 2. These results highlight the relation between k_c and k^* and the corresponding value of their loss functions as demonstrated in (5.25). We consider a star network consisting of M users and a server, where each user has a queue to keep the transmission packets, as shown in Figure 5.2. We characterize the latency of performing each FL iteration k as $c_k := \sum_{i=1}^{4} \ell_{i,k}$ [4,136], where

- $\ell_{1,k}$: communication latency in broadcasting parameters by the server;
- $\ell_{2,k}$: the computation latency in computing $\boldsymbol{w}_k^j$ for every user j;
- $\ell_{3,k}$: communication latency in sending $\boldsymbol{w}_k^j$ to the server;
- $\ell_{4,k}$: computation latency in updating parameters at the the server.

We consider solving a regression problem over a wireless network using a real-world dataset, such as MNIST (hand-written digits). We then randomly split the data samples among M users, each having $\{(\boldsymbol{x}_n^j, y_n^j)\}$, where $\boldsymbol{x}_n^j \in \mathbb{R}^{784}$ is a vectorized image at each user $j = \{1, \ldots, M\}$ with corresponding digit label y_n^j. Without loss of generality, we consider a convex loss function

$$f(\boldsymbol{w}) = \frac{1}{M} \sum_{j=1}^{M} \frac{1}{N_j} \sum_{i=1}^{N_j} \log\left(1 + e^{-\boldsymbol{w}^\top \boldsymbol{x}_n^j y_n^j}\right), \tag{5.26}$$

where we consider that each user $j = \{1, \ldots, M\}$ has the same number of samples, namely $N_j = N_l, \forall j, l \in \{1, \ldots, M\}$.

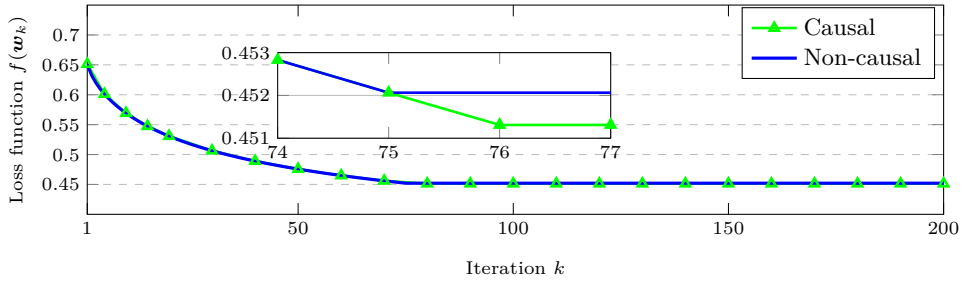
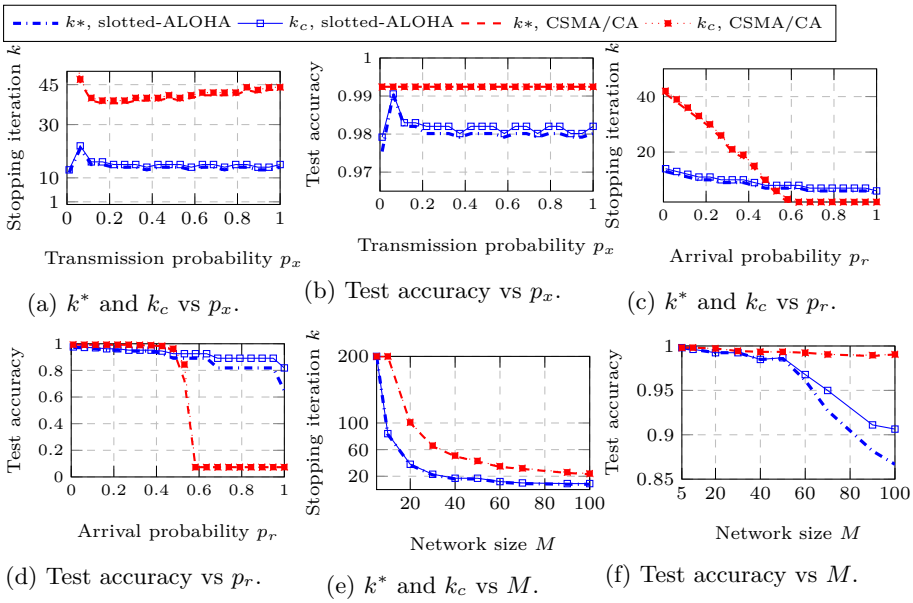


Figure 5.3: Illustration of the training loss function $f(\boldsymbol{w}_k)$ with iteration k for causal and non-causal stopping iteration, and $M = 10$, $p_r = 0.2$. We observe that the non-causal $k^* = 75$ and the causal $k_c = 76$ with $f(\boldsymbol{w}_{k_c}) < f(\boldsymbol{w}_{k^*})$, which demonstrate the results of (5.25a) and (5.25b). More details can be found in [4].

We use a network simulation to implement slotted-ALOHA with binary exponential backoff and M users in the network. We consider a capacity of one packet per slot and a slot duration of 1 μs and assume that every user generates a new background traffic packet at every time slot with probability p_r. A higher p_r value corresponds to intensive traffic, such as video streaming, while a lower p_r corresponds to lighter traffics, such as browsing. Figure 5.3 illustrates the loss function $f(\boldsymbol{w}_k)$ vs. iteration k for both causal and non-causal stopping iterations. Furthermore, we observe that $k_c = k^* + 1$ demonstrates the results we represented in (5.25a) and (5.25b). In this thesis, in addition to slotted-ALOHA, we consider more scenarios for communication protocols, such as CSMA/CA and OFDMA, and investigate a general class of loss functions, such as ANNs. For more detailed information, you can read our papers [4, 136, 138], and Papers 1 and 2 as in Chapter 7 of this thesis.

Impact of Communication Parameters

In this section, we illustrate the impacts of the communication parameters of slotted-ALOHA and CSMA/CA protocols on the performance of our causal solutions. Investigating such impacts is important for further designing communication and computation efficient FL over wireless networks which utilize these two protocols. To this end, we consider the system model shown in Figure 5.2, where the users have their packet queues and assume that the FL local parameters, shown as blue packets, are the head-of-line packets in the queues. Accordingly, we assume that the FL local parameters are treated as the priority packets in the queues and are placed in the head of the queues after users compute them. This assumption is essential to reduce the extra latency each user may face because of the background traffic packets during the FL training. We define p_x and p_r as the transmission probability and background packet arrival probability at each slotted-ALOHA or CSMA/CA protocol time slot.



(a) k^* and k_c vs p_x. (b) Test accuracy vs p_x. (c) k^* and k_c vs p_r.

(d) Test accuracy vs p_r. (e) k^* and k_c vs M. (f) Test accuracy vs M.

Figure 5.4: Illustration of the impact of communication parameters on causal and non-causal solutions.

Figure 5.4 shows the impact of communication parameters, the transmission p_x, and arrival probabilities p_r, and the number of users M on the non-causal stopping iteration k^* and the causal stopping iteration k_c with the corresponding test accuracy. Figures 5.4a-5.4b respectively characterize the stopping iterations and test accuracy of the FL, respectively, all local data size for GD iterations, and with $M = 50$ and $p_r = 0.01$. We note that the transmission probability p_x has more negative impacts when we regulate the channel by slotted-ALOHA. Figures 5.4c-5.4d represent similar results, but by swiping the background packet arrival probability p_r, when $M = 50$ and $p_x = 0.8$. The numerical results show that when facing higher arrival traffic, slotted-ALOHA performs better than CSMA/CA. Finally, we characterize the same results by varying M when $p_x = 0.8$ and $p_r = 0.01$. Figure 5.4e shows that slotted-ALOHA and CSMA/CA have similar decreasing behavior with increasing M. However, as seen in Figure 5.4f, slotted-ALOHA reduces the test accuracy in larger M.

Overall, Figure 5.4 highlights the importance of designing the communication protocols for our causal approaches to achieve a communication and computation efficient FL over wireless networks. Consequently, in Paper 2 in Chapter 7 of this thesis, we propose upper and lower bounds of iteration-cost c_k as a function of the communication parameters p_r, p_x, and M; see [136]. These bounds on c_k have a significant role in analyzing the impact of different communication protocols, such as slotted-ALOHA, CSMA/CA, or OFDMA, on the cost of each FL iteration. More-

over, the corresponding bounds demonstrate our causal methods' great potential for application in various communication protocols.

5.3 A-LAQ: Adaptive, Energy-efficient, and Causal FL

In this section, we present the system model and problem formulation of Paper 3, followed by an explanation of the causal approach, its analytical results, and optimality. The proposed Adaptive LAQ (A-LAQ) extends the LAQ method [44] by employing an adaptive number of communication bits to address energy-constrained scenarios. A-LAQ leverages the descent behavior and *diminishing return* rule [47] in FL for $\bar{L}$-smooth and convex loss functions, where accuracy improvements diminish with additional communication iterations. Initially, A-LAQ allocates a higher number of bits to minimize quantization error during the early stages of training and adaptively reduces the bits in subsequent iterations, ensuring efficient communication energy usage with minimal impact on training performance.

Main Optimization Problem

The main optimization problem of A-LAQ is as follows:

$$\begin{aligned}
\underset{\boldsymbol{w},k_0,K}{\text{minimize}} \quad & f(\boldsymbol{w}) && \text{(5.27a)}\\
\text{subject to} \quad & \boldsymbol{w}_k = \boldsymbol{w}_{k-1} - \alpha\tilde{\nabla}_{k-1}, \quad k = 1,\dots,K && \text{(5.27b)}\\
& \tilde{\nabla}_k = \tilde{\nabla}_{k-1} + \sum_{j=1}^{M} \delta\boldsymbol{q}_k^j, \quad k = 1,\dots,K && \text{(5.27c)}\\
& \delta\boldsymbol{q}_k^j = \boldsymbol{q}^j(\boldsymbol{w}_k) - \boldsymbol{q}^j(\boldsymbol{w}_{k-1}), \quad k = 1,\dots,K && \text{(5.27d)}\\
& b_k = b^{\max}\mathbb{1}_{k\le k_0} && \text{(5.27e)}\\
& \quad + b_0\mathbb{1}_{k=k_0+1} + \lceil \eta_{k-1}b_{k-1}\rceil\mathbb{1}_{k>k_0+1}\\
& b_k \ge 2, \quad k = 1,\dots,K && \text{(5.27f)}\\
& \sum_{k=1}^{K} E_k \le E, \quad k = 1,\dots,K, && \text{(5.27g)}
\end{aligned}$$

where $\boldsymbol{w}_k$ represents the global FL parameter at communication iteration k, $\rho_j, j \in [M]$ denotes the local weight, α is the step size, E_k is the communication energy spent at iteration k, E is the total communication energy budget, and $k_0 \le K$ specifies the number of initial iterations during which the communication bit allocation is set as $b_k = b^{\max}$. Here, $b^{\max}$ and b_0 are predefined bit quantities.

We propose to update the communication bits as $b_k = \lceil \eta_{k-1}b_{k-1}\rceil$ for $k = \max\{3, k_0\},\dots,K$, introducing η_{k-1} defined as:

$$\eta_{k-1} := \min\left\{\frac{\|f(\boldsymbol{w}_{k-1}) - f(\boldsymbol{w}_{k-2})\|}{\|f(\boldsymbol{w}_{k-2}) - f(\boldsymbol{w}_{k-3})\|}, 1\right\}, \tag{5.28}$$

where the rationale behind choosing η_{k-1} as in (5.62) is the diminishing return rule. Constraints (5.27b)-(5.27d) denote global LAQ update, constraints (5.27e) and (5.27f) show the adaptive b_k, and constraint (5.27g) is the total communication energy limits.

Optimization problem (5.27) addresses FL training in a communication energy-limited setup. While LAQ is a communication-efficient approach, A-LAQ demonstrates superior performance under the same resource constraints. A-LAQ assigns a high number of communication bits during the initial iterations, $1, \ldots, k_0$, followed by training with b_0 bits, where $b_0 < b$ (the bits used by LAQ), adhering to a non-increasing bit sequence as defined by (5.28). However, optimization problem (5.27) is impractical because it assumes knowledge of K and the future local gradients for $k = 1, \ldots, K$ from the start of training. As obtaining such information is unrealistic, this problem is termed *non-causal* [4]. Consequently, this thesis focuses on developing causal and practical solutions that operate without requiring future knowledge of local gradients or K.

Solution Approach

First, we obtain k_0 according to the following Proposition:

Proposition 1. *Let $f(\boldsymbol{w})$ be μ-strongly convex and $\bar{L}$-smooth. Consider $b^{\max} = 32$ bits. Thus, $k_0 = \min\{k_e, k_f\}$, where*

$$k_e := \textit{the first value of } k \textit{ such that } E_k > E - \sum_{k'=1}^{k-1} E_{k'}, \tag{5.29}$$

and

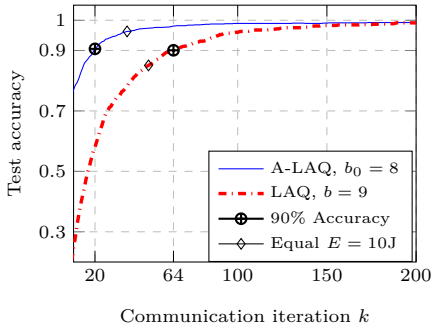
$$k_f := \textit{the first value of } k \textit{ such that} \quad k < \frac{f(\boldsymbol{w}_0) - f(\boldsymbol{w}_k)}{f(\boldsymbol{w}_{k-1}) - f(\boldsymbol{w}_k)}. \tag{5.30}$$

Proof. See Appendix A-A in [139]. □

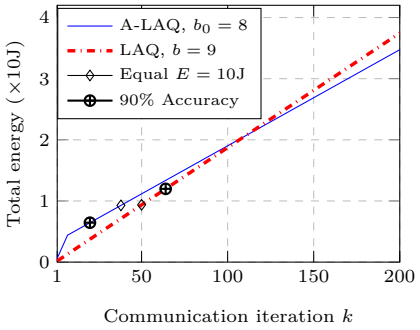
After obtaining k_0, the training will continue with an adaptive number of bits $b_k = \lceil \eta_{k-1} b_{k-1} \rceil$, where η_{k-1} is defined in (5.28). Then, based on the remaining energy budget, the causal stopping iteration K is obtained.

Illustrative Numerical Results

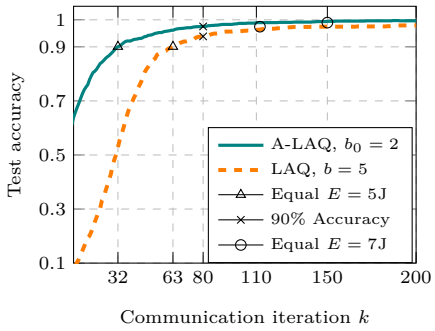
We address a convex regression problem over a wireless network using a binary dataset derived from MNIST (handwritten digits). Specifically, we retain samples of digits 0 and 1, assigning labels -1 and +1, respectively. The resulting dataset of 12,600 samples is randomly distributed across M users, where each node $j \in [M]$ holds data $\{(\boldsymbol{x}_{ij}, y_{ij})\}$, with $\boldsymbol{x}_{ij} \in \mathbb{R}^{784}$ representing the ith vectorized image and



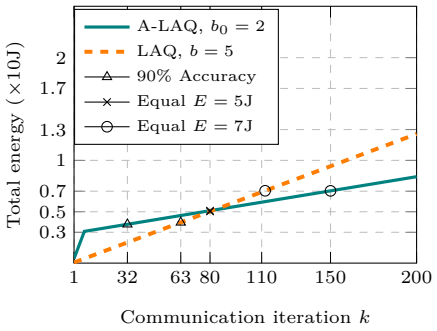
(a) Test accuracy, $M = 50$.



(b) Communication energy, $M = 50$.



(c) Test accuracy, $M = 30$.



(d) Communication energy, $M = 30$.

Figure 5.5: Comparison between A-LAQ and LAQ.

$y_{ij} \in \{-1, +1\}$ as its corresponding label. The training employs the following loss function [140]:

$$f(\boldsymbol{w}) = \sum_{j=1}^{M} \rho_j \sum_{i=1}^{|D_j|} \frac{1}{|D_j|} \log\left(1 + e^{-\boldsymbol{w}^T \boldsymbol{x}_{ij} y_{ij}}\right) + \frac{\lambda}{2}\|\boldsymbol{w}\|_2^2, \tag{5.31}$$

where $\lambda \in (0, 1)$ is a given regularization parameter and users have the equal data sample size, as $|D_j| = |D_i| = |D|/M, \forall i, j \in [M]$.

We consider an uplink OFDMA system in a single-cell network with a coverage radius of $l_c = 1$ km. The system includes L_p cellular links utilizing S_c subchannels. The power gain on each subchannel is modeled as $h_l^s = \phi/(l_d^j)^3$, where l_d^j represents the distance between a user and the master node. This follows Rayleigh fading, with ϕ exponentially distributed with a mean of 1. The noise power on each subchannel is -170 dBm/Hz, and the maximum transmit power per link is 23 dBm. The system parameters include $S_c = 64$ subchannels, a total bandwidth of 10 MHz, and a subchannel bandwidth of 150 kHz.

Table 5.3: Simulation parameters for FL over CFmMIMO.

Parameter	**Value**	**Parameter**	**Value**
Bandwidth	$B = 20\,\text{MHz}$	Noise figure	7
Area of interest (wrap around)	$1000 \times 1000\text{m}$	Pathloss exponent	$\alpha_p = 3.67$
Number of APs	$A_p = 16$	Coherence block length	$\tau_c = 200$
Number of per-AP antennas	$N = 4$	Pilot length	$\tau_p = 10$
Uplink transmit power	$p^u = 100\,\text{mW}$	Uplink noise power	$\sigma^2 = -94\,\text{dBm}$

Fig. 5.5 compares A-LAQ and LAQ performance. Figs. 5.5a and 5.5b illustrate test accuracy for $M = 50$, $b = 9$, $b_0 = 8$, $b^{\max} = 32$, and $k_0 = 7$. Black markers highlight comparisons for the same energy budget E or test accuracy. For $E = 10$J, A-LAQ achieves 96% accuracy with $K = 38$, while LAQ achieves 85% with $K = 50$. Additionally, for 90% accuracy, A-LAQ uses 50% less energy and fewer iterations than LAQ. Figs. 5.5c and 5.5d show results for $M = 30$, $b = 5$, $b_0 = 2$, $b^{\max} = 32$, and $k_0 = 7$. A-LAQ achieves the same 90% accuracy as LAQ with similar energy but fewer iterations. For $E = 5$J, both calculate the same K, but A-LAQ's accuracy is 4% higher. For $k \geq 80$, A-LAQ requires less communication energy than LAQ, with similar accuracy, showing its efficiency in high-energy scenarios.

5.4 CFmMIMO Power Control: Low-latency and Energy-Efficient FL

In this section, we propose three power control schemes designed to achieve energy-efficient and low-latency FL training in CFmMIMO networks. The focus is on optimizing uplink transmissions, where local models are sent from user devices to the APs, aiming to minimize resource consumption. The uplink poses a greater challenge compared to the downlink, as it involves transferring M local models under the constraints of limited energy from battery-powered devices. In contrast, the downlink transmission of a single global model benefits from high power and grid-connected APs. Therefore, we assume, without loss of generality, that the downlink transmission is error-free and that all users receive an identical global model $\boldsymbol{w}_k$. Additionally, the simulation parameters for this section are shown in Table 5.3.

5.4.1 Joint Energy and Latency Optimization

In this section, we consider FL training over a CFmMIMO system with A_p APs, each equipped with N antennas, which forward their received signals to the server via error-free fronthaul links, as illustrated in Figure 5.6. The objective is to jointly minimize users' uplink energy consumption and the latency of FL training. Striking a balance between uplink latency and energy is crucial, as reducing energy often requires transmitting at lower rates, leading to increased latency, and vice versa.

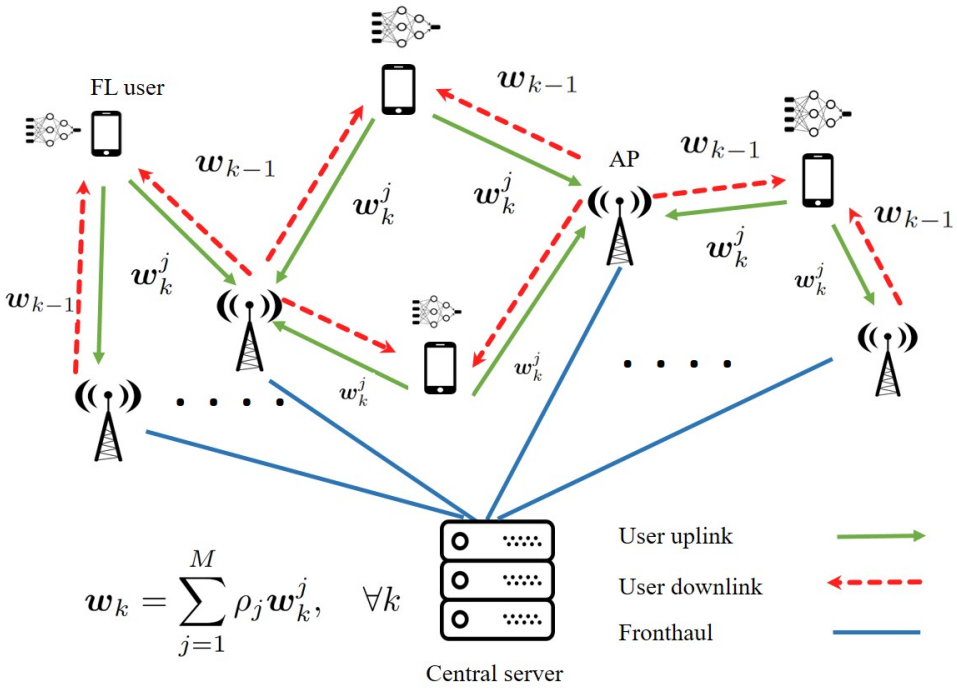


Figure 5.6: General architecture of FL over CFmMIMO.

Additionally, the optimization must consider the impact of inter-user interference on each user while addressing this trade-off.

Main Optimization Problem

Here, the goal is to allocate power coefficients $p_j, j \in [M]$ to jointly optimize users' uplink energy and latency, thereby maximizing the number of global iterations $K_{\max}$ under given energy and latency constraints.

Let $\bar{\mathcal{L}}$ and $\bar{\mathcal{E}}$ represent the total uplink latency and energy budgets for the FL training uplink phase. By optimizing the uplink power coefficients, namely $\boldsymbol{p}^* := [p_1^*, \ldots, p_M^*]^\top$, to minimize latency and energy per iteration, the number of global FL iterations K can be maximized within these constraints. Therefore, defining the optimal uplink energy $E_j^* := E_j|_{p_j=p_j^*}$, optimal uplink latency $\ell_j^* := \ell_j|_{p_j=p_j^*}$ and $\bar{\ell}^* := \max_j \ \ell_j^*$ and $\bar{E}^* := \sum_{j=1}^{M} E_j^*$, we obtain the maximum number of FL

iterations as

$$K_{\max} := \lfloor \min\{K_e,\ K_l\} \rfloor, \tag{5.32}$$
$$K_e := \frac{\bar{\mathcal{E}}}{\bar{E}^*}, \quad K_l := \frac{\bar{\mathcal{L}}}{\bar{\ell}^*}.$$

As a result, we perform an FL training with the maximum number of global FL iterations $K = K_{\max}$ based on the optimized uplink powers p_j^*.

We now present the main problem formulation of this subsection, which focuses on training an FL algorithm under uplink energy and latency constraints by optimizing the uplink power allocation for each user p_j, $j = 1, \ldots, M$. Specifically, we define the uplink latency as $\ell_j := bd/R_j$ and the uplink energy consumed by user j as:

$$E_j := p^u p_j \ell_j = p^u p_j \frac{bd}{R_j}, \quad j = 1, \ldots, M, \tag{5.33}$$

where R_j denotes the uplink data rate for user j according to the definition of uplink data rate (in bit/s) in (3.8), b represents the number of bits required to transmit each entry of $\boldsymbol{w}_k^j \in \mathbb{R}^d$, and d corresponds to the size of the local and global FL models. We propose the following optimization problem to minimize the weighted sum of uplink energy and latency across users:

$$\underset{p_1,\ldots,p_M}{\text{minimize}} \quad \sum_{j=1}^{M} (\theta_1 E_j + \theta_2 \ell_j) \tag{5.34a}$$
$$\text{subject to} \quad 0 \le p_j \le 1, \quad j = 1, \ldots, M, \tag{5.34b}$$

where $\theta_1, \theta_2 \in [0, 1]$ are scalarization weights that balance the scaling differences between energy and latency. We denote that for each user $j \in [M]$, E_j is an increasing function of p_j, while ℓ_j is a decreasing function of p_j. Thus, any linear combination of $\theta_1 E_j + \theta_2 \ell_j$ represents the trade-off between uplink energy E_j and latency ℓ_j [38] for $p_j \in [0, 1]$. Accordingly, the weighted sum in (5.34a) reflects the overall trade-off between energy and latency for the FL users at each global iteration $k \in [K]$. This trade-off captures the influence of each p_j on the energy $E_{j'}$ and latency $\ell_{j'}$ of other users $j' \neq j$, which is considered by the optimization problem (5.34). By solving (5.34), we obtain the uplink power p_j for each user j that jointly minimizes the trade-off between energy E_j and latency ℓ_j for all users.

To solve (5.34), we first present the following lemma to illustrate the structure of the objective function in (5.58a).

Lemma 2. *Let E_j and ℓ_j be the energy and latency for sending each local FL model $\boldsymbol{w}_t^j$ to the APs. We define $\nu(\boldsymbol{p}; \theta_1, \theta_2, b, d) := \sum_{j=1}^{M} \theta_1\ E_j\ +\ \theta_2\ \ell_j$ as the objective function in* (5.58a)*. When all other variables are fixed, $\nu(\boldsymbol{p}; \theta_1, \theta_2, b, d)$ is quasi-convex in the domain $0 \le p_j \le 1$, and has a unique minimum w.r.t. p_j (for*

any user $j = 1, \ldots, M$), defined as

$$p_j^* = \arg\min_{p_j} \quad \nu(\boldsymbol{p}; \theta_1, \theta_2, b, d) \tag{5.35a}$$

$$\text{subject to} \quad 0 \leq p_j \leq 1. \tag{5.35b}$$

Proof. The proof is given in Appendix A of Paper 4 in Chapter 7 of this thesis. □

Lemma 2 shows that the objective loss function $\nu(\boldsymbol{p}; \theta_1, \theta_2, b, d)$ has a unique minimum w.r.t. every p_j, $j = 1, \ldots, M$. Inspired by Lemma 2, we propose the following remark investigating the solution to optimization problem (5.34).

Remark 1. *For the quasi-convex function $\nu(\boldsymbol{p}; \theta_1, \theta_2, b, d)$ defined in Lemma 2, quasi-convexity ensures that any local minimum is also a global minimum. Since the objective function is separable across users and has a unique minimizer p_j^* for each j, the overall solution $[p_1^*, \ldots, p_M^*]$ is the global minimum of $\nu(\boldsymbol{p}; \theta_1, \theta_2, b, d)$ across all users. Thus, $\boldsymbol{p}^*$ is globally optimal.*

By utilizing Lemma 2 and Remark 1, we establish the following proposition presenting the solution to (5.34).

Proposition 2. *Consider $\nu(\boldsymbol{p}; \theta_1, \theta_2, b, d) = \sum_{j=1}^{M} \theta_1 E_j + \theta_2 \ell_j$, where $\nu(\boldsymbol{p}; \theta_1, \theta_2, b, d)$ is differentiable and has a unique minimum w.r.t. every p_j. Let $\boldsymbol{p}^* = [p_1^*, \ldots, p_M^*]^\intercal$ be the vectorized version of the solutions obtained after solving* (5.35)*. We can then obtain p_j^* as*

$$p_j^* = \left\{ p_j : \ \partial \nu_j := \frac{\partial}{\partial p_j} \nu(\boldsymbol{p}; \theta_1, \theta_2, b, d) = 0 \right\}. \tag{5.36}$$

Proposition 2 proposes an approach for solving the single-variable optimization problem (5.35) for any given θ_1 and θ_2, when the other power variables are fixed. Since their solutions are interdependent, we propose the CD method, explained in Chapter 2.5, for our proposed power allocation scheme. While the details are given in Algorithm 1 of Paper 4 in Chapter 7 of this thesis, we call this solution approach as *Approach 1* through this section.

The above solution approach applies to fixed values of θ_1 and θ_2. However, achieving the optimal power allocation scheme requires determining the best values for θ_1 and θ_2. The next subsection focuses on addressing this issue.

Jointly Optimal Power Allocation and Weight Selection

To solve (5.34) and determine the optimal $\boldsymbol{p}^*$ for maximizing $K_{\max}$, we must select weights θ_1 and θ_2 appropriately. Since $K_{\max}$ depends on the energy ($\bar{\mathcal{E}}$) and latency ($\bar{\mathcal{L}}$) budgets via (5.32), we propose an optimization problem to minimize the absolute difference between K_e and K_l, which represent the number of FL iterations supported by the energy and latency budgets, respectively. This objective ensures that $K_{\max}$, defined as the smaller of K_e and K_l, is maximized without

wasting any budget. Additionally, as discussed in Paper 4 of Chapter 7, K_e and K_l exhibit opposite behaviors when the weights, θ_1 and θ_2, vary: K_e decreases while K_l increases. By minimizing the difference between K_e and K_l, the uplink power allocation achieves a balanced trade-off, utilizing both energy and latency budgets effectively.

We focus on obtaining the weights θ_1 and θ_2 that minimize the absolute difference between K_l and K_e. We define $\theta := \theta_2/\theta_1$, for $\theta_1 \neq 0$, while noting that p_j, $j = 1, \ldots, M$, also depend on θ. We propose the following optimization problem:

$$\theta^* \in \arg\min_{\theta} G(\theta), \tag{5.37}$$

where

$$G(\theta) := \left| K_e(\theta) - K_\ell(\theta) \right|, \tag{5.38}$$

$K_e(\theta)$ and $K_\ell(\theta)$ are defined as

$$K_e(\theta) := \frac{\bar{\mathcal{E}}}{\sum_{j=1}^{M} p^u p_j(\theta)/R_j(\theta)}, \tag{5.39}$$

$$K_\ell(\theta) := \frac{\bar{\mathcal{L}}}{\max_j \frac{1}{R_j(\theta)}} = \bar{\mathcal{L}} R_j(\theta)_{\min}, \tag{5.40}$$

and $R_j(\theta)_{\min} := \min_j R_j(\theta)$. Note that we use the same p_j, K_e, and K_l as before but with a dependency on θ to be persistent in notations and avoid confusion.

To solve optimization problem (5.37) and obtain θ^*, we employ the Brent method, as explained in Chapter 4.2.1 of this thesis. We recall that the Brent method is a robust numerical optimization and root-finding algorithm combining bisection, secant, and inverse quadratic interpolation. It has the reliability of bisection and uses the potentially fast-converging secant method or inverse quadratic interpolation. The secant method, which is based on a linear approximation, considers two points θ_a, θ_b with the corresponding function values $G(\theta_a)$ and $G(\theta_b)$, and identifies a new potential minimum point as follows:

$$\theta_s \leftarrow \theta_b - G(\theta_b) \cdot \frac{\theta_b - \theta_a}{G(\theta_b) - G(\theta_a)}. \tag{5.41}$$

The inverse quadratic interpolation considers three points θ_a, θ_b, and θ_c and the corresponding function values and then computes a new potential minimum point as

$$\begin{aligned} \theta_s \leftarrow & \theta_a \cdot \frac{G(\theta_b) \cdot G(\theta_c)}{(G(\theta_a) - G(\theta_b)) \cdot (G(\theta_a) - G(\theta_c))} + \\ & \theta_b \cdot \frac{G(\theta_a) \cdot G(\theta_c)}{(G(\theta_b) - G(\theta_a)) \cdot (G(\theta_b) - G(\theta_c))} + \\ & \theta_c \cdot \frac{G(\theta_a) \cdot G(\theta_b)}{(G(\theta_c) - G(\theta_a)) \cdot (G(\theta_c) - G(\theta_b))}. \end{aligned} \tag{5.42}$$

Brent's method falls back to the more robust bisection method, i.e. $\theta_s \leftarrow (\theta_a+\theta_b)/2$, if the conditions in *Approach 2*, line 14, for secant or inverse quadratic interpolation methods are not satisfied.

After calculating θ_s, we evaluate the function $G(\theta)$ at $\theta = \theta_s$ by applying CD method with the corresponding $\theta_{1,s}$ and $\theta_{2,s}$ obtained from θ_s according to:

$$\begin{aligned} \theta_1 &:= \mathbb{1}_{\{\theta\leq 1\}} + \mathbb{1}_{\{\theta>1\}}\theta^{-1}, \\ \theta_2 &:= \mathbb{1}_{\{\theta\leq 1\}}\theta + \mathbb{1}_{\{\theta>1\}}. \end{aligned} \tag{5.43}$$

After evaluating $G(\theta_s)$, the points θ_a, θ_b, and θ_c are updated accordingly based on their function values (lines 21-30 in *Approach 2*). The algorithm checks if the difference between θ_b and θ_a is less than the tolerance ϵ_B, in which case it returns θ_b as the minimum. This process iterates until the interval is sufficiently small or the maximum number of iterations is reached, ensuring a robust and efficient convergence to the minimum. Brent's method efficiently finds the minimum by balancing fast convergence through interpolation and guaranteed progress through the bisection method.

Theorem 2. *Let us define* $K^* := K_{\max}|_{\theta=\theta^*}$*, where* $\theta^* = \theta_2^*/\theta_1^*$ *is obtained from Approach 2, and* $K_{\max}$ *is the maximum number of FL iterations as defined in* (5.32)*. Then, for any* $\theta \in (0,\infty)$*, we have* $K^* \geq K_{\max}$ *for any* $\theta \neq \theta^*$*.*

Proof. The proof is given in Appendix D of Paper 6 in Chapter 7 of this thesis. □

Theorem 2 demonstrates that solving the optimization problem (5.37) and obtaining θ^* using *Approach 2* results in achieving the maximum achievable number of FL iterations $K_{\max}$ w.r.t. θ and for given budgets $\bar{\mathcal{E}}$ and $\bar{\mathcal{L}}$.

While the details are given in Algorithm 2 of Paper 4 in Chapter 7 of this thesis, we call this solution approach as *Approach 2* through this section. Although the aforementioned approach finds the parameters θ_1^* and θ_2^* that yield the maximum $K_{\max}$, it results in high computational complexity depending on ϵ_B, $\bar{\epsilon}$, and ϵ, as stated in Theorem 2 of Paper 6. Therefore, inspired by the optimization problem (5.37), we propose a new power allocation scheme that eliminates the dependence on θ by directly minimizing the absolute difference between K_l and K_e, as stated in the next subsection.

Fast Power Allocation via Weight-Free Optimization

Here, we aim to allocate the uplink powers $p_1,\ldots,p_M$ such that it minimizes $|K_e(\boldsymbol{p}) - K_\ell(\boldsymbol{p})|$, where $\boldsymbol{p} := [p_1,\ldots,p_M]^\top$, which serves as an alternative way to maximize the number of FL iterations. Hence, we propose to solve the following optimization problem as

$$\underset{p_1,\ldots,p_M}{\text{minimize}} \quad \left| \frac{\bar{\mathcal{E}}}{\sum_{j=1}^{M} bdp^u p_j/R_j} - \frac{\bar{\mathcal{L}}}{\max_j \frac{bd}{R_j}} \right| \tag{5.44a}$$

$$\text{subject to} \quad 0 \leq p_j \leq 1, \quad j = 1,\ldots,M. \tag{5.44b}$$

The optimization problem (5.44) seeks a balance between $K_e(\boldsymbol{p})$ and $K_l(\boldsymbol{p})$, which often leads to a trade-off because improving one can worsen the other.

Then, based on the smoothing technique, we re-write (5.44) as

$$\underset{p_1,\ldots,p_M}{\text{minimize}} \quad \sqrt{\left(\frac{\bar{\mathcal{E}}}{\sum_{j=1}^{M} bdp^u p_j/R_j} - \frac{\bar{\mathcal{L}}}{\ell_{\max}}\right)^2 + \epsilon} \tag{5.45a}$$

$$\text{subject to} \quad p_j \geq 0, \quad j = 1,\ldots,M, \tag{5.45b}$$

$$p_j \leq 1, \quad j = 1,\ldots,M, \tag{5.45c}$$

where $\ell_{\max} = bd \cdot \max_j \ (R_j)^{-1}$ is an auxiliary variable to represent the maximum uplink latency, $\ell_j \leq \ell_{\max}$, we reformulate the optimization problem (5.45) to ensure each user achieves at least this rate. Accordingly, we can write (5.45) as

$$\underset{p_1,\ldots,p_M,\tilde{\ell}}{\text{minimize}} \quad \sqrt{\left(\frac{\bar{\mathcal{E}}}{\sum_{j=1}^{M} p^u p_j/R_j} - \frac{\bar{\mathcal{L}}}{\tilde{\ell}}\right)^2 + \epsilon} \tag{5.46a}$$

$$\text{subject to} \quad p_j \geq 0, \quad j = 1,\ldots,M, \tag{5.46b}$$

$$p_j \leq 1, \quad j = 1,\ldots,M, \tag{5.46c}$$

$$\text{SINR}_j \geq 2^{1/(B_\tau \tilde{\ell})} - 1, \quad j = 1,\ldots,M, \tag{5.46d}$$

where $\tilde{\ell} := \ell_{\max}/bd$, and SINR_j can be expressed as

$$\text{SINR}_j = \frac{\bar{A}_j p_j}{\bar{B}_j p_j + \sum_{j' \neq j}^{M} p_{j'} \tilde{B}_j^{j'} + I_M^j}. \tag{5.47}$$

Then, we define the constraint vector $[\boldsymbol{c}(\boldsymbol{p};\tilde{\ell})]_j = c^j(\boldsymbol{p};\tilde{\ell})$, for $j = 1,\ldots,M$ and re-write (5.46d) as

$$c^j(\boldsymbol{p};\tilde{\ell}) := \left(2^{1/B_\tau \tilde{\ell}} - 1\right)\left(\bar{B}_j p_j + \sum_{j' \neq j}^{M} p_{j'} \tilde{B}_j^{j'} + I_M^j\right) - \bar{A}_j p_j \leq 0. \tag{5.48}$$

The constraint vector $\boldsymbol{c}(\boldsymbol{p};\tilde{\ell})$ is linear in the power variables but depends on $\tilde{\ell}$ through $(2^{1/(B_\tau \tilde{\ell})} - 1)$. Accordingly, in the following subsection, we propose an algorithm to solve (5.46) with the combination of bisection over $\tilde{\ell}$ and SQP method. While the details are given in Algorithm 3 of Paper 4 in Chapter 7 of this thesis, we call this solution approach as *Approach 3* through this section.

Illustrative Numerical Results

Here, we present numerical results to evaluate the performance of the proposed algorithms compared to previous works. The simulations consider a CFmMIMO

Table 5.4: Comparison of *Approach 1* and *2* using CNN and ResNet-18, $M = 20$, $L = 8$, $\bar{\mathcal{E}} = 0.04$, $\bar{\mathcal{L}} = 10$ for CNN and $\bar{\mathcal{E}} = 0.4$, $\bar{\mathcal{L}} = 100$ for ResNet-18, CIFAR-10 with IID and non-IID distribution.

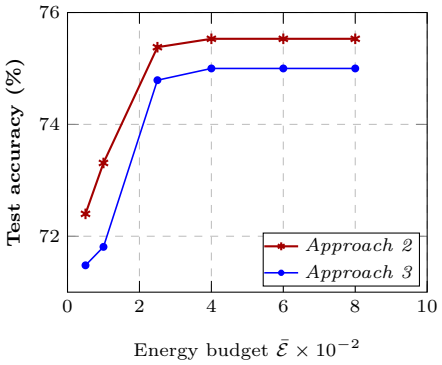
Parameters		**IID**				**Non-IID**			
		CNN		**ResNet-18**		**CNN**		**ResNet-18**	
$\theta = \theta_2/\theta_1$	b	$K_{\max}$	Accuracy (%)	$K_{\max}$	Accuracy (%)	$K_{\max}$	Accuracy (%)	$K_{\max}$	Accuracy (%)
0.1	32	12	70.33	3	88	12	68.58	3	81.01
	8	20	71.3	11	90.83	20	70.16	11	88.9
2	32	13	70.97	4	89.73	13	69.06	4	86.12
	8	52	77.76	15	91.65	52	75.02	15	90.02
1.25	32	12	70.33	4	89.73	12	68.58	4	86.12
	8	46	77.84	10	90.41	46	74.33	10	88.82
10	32	9	67.9	3	88	9	64.42	3	81.01
	8	39	75.31	10	90.41	39	74	10	88.82
$\boldsymbol{\theta^* = 1.191895}$	32	14	71.5	4	89.73	14	69.9	4	86.12
	8	56	78.6	15	91.6	56	75.55	15	90.02

network with $A_p = 16$ APs, each with $N = 4$ antennas, a central server, and $M = 20$ single-antenna FL users. Users are assigned τ_p-length pilot sequences based on the algorithm in [2], with parameters summarized in Table 5.3. Various latency budgets $\bar{\mathcal{L}}$ (seconds) and energy budgets (Joule) are considered. The CIFAR-10 dataset is distributed among users in both IID and non-IID settings, and each user trains the following CNNs locally.

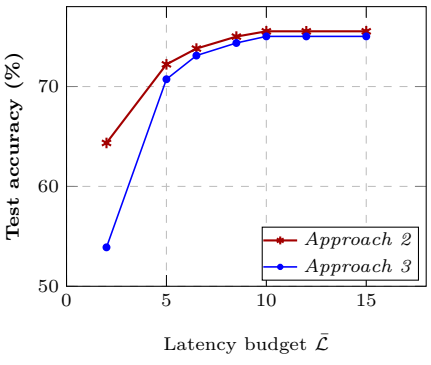
The first CNN, adapted from [141], uses three convolutional blocks (filters: 32, 64, 128), followed by max-pooling, dropout, and a dense softmax layer for classification into 10 categories, with $d = 307,498$. This architecture balances computational efficiency and accuracy, making it suitable for local FL training. ResNet-18 [142], with 18 layers and $d = 11,181,642$, features residual connections to address vanishing gradients, providing strong performance on tasks like image classification for datasets such as CIFAR-10.

We present numerical results for $K_{\max}$ and the corresponding FL test accuracy as functions of $\theta = \theta_2/\theta_1$ for *Approach 1*, with outcomes compared using the fraction θ instead of individual values. The optimal values θ^* from *Approach 2* are provided in Table 5.4 for $K = 20$ users, considering both IID and non-IID data distributions with CNN and ResNet-18 architectures. The simulations assume $b = 8$ bits for LAQ [44] and $b = 32$ for full-precision quantization.

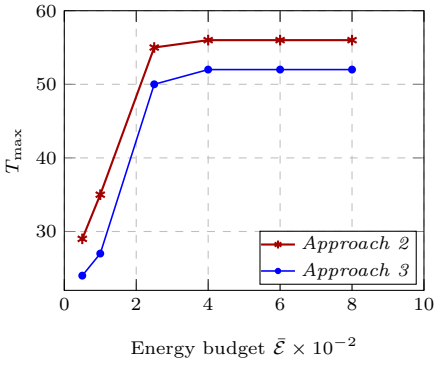
The results demonstrate that *Approach 2* effectively identifies θ^* to maximize $K_{\max}$ and improve test accuracy, achieving up to a 10% increase in accuracy by



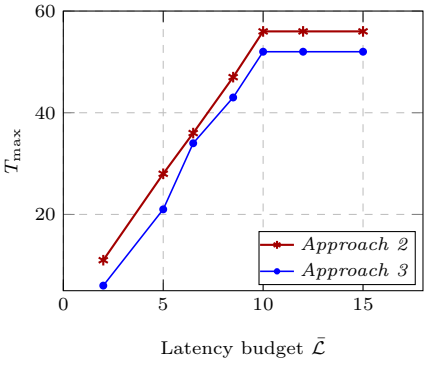
(a) Accuracy vs. $\bar{\mathcal{E}}$, $\bar{\mathcal{L}} = 10$



(b) Accuracy vs. $\bar{\mathcal{L}}$, $\bar{\mathcal{E}} = 0.04$



(c) $K_{\max}$ vs. $\bar{\mathcal{E}}$, $\bar{\mathcal{L}} = 10$



(d) $K_{\max}$ vs. $\bar{\mathcal{L}}$, $\bar{\mathcal{E}} = 0.04$

Figure 5.7: Comparison of *Approach 2* and *Approach 3*.

enabling higher $K_{\max}$ under given energy and latency budgets. Notably, for $b = 8$, the increased $K_{\max}$ compensates for quantization errors, often surpassing the test accuracy of 32-bit resolution. These findings highlight that when computation complexity is not a constraint, optimizing θ_2/θ_1 as in *Approach 2* enables higher global iterations and improved FL performance within a constrained budget.

We compare the performance of *Approach 3* with *Approach 2* for $M = 20$, $b = 8$, and a non-IID data distribution, as shown in Figure 5.7. For CNN, test accuracy and $K_{\max}$ are plotted against energy budget $\bar{\mathcal{E}}$ and latency budget $\bar{\mathcal{L}}$. While *Approach 3* achieves slightly smaller $K_{\max}$ and test accuracy than *Approach 2*, it offers significant speed advantages, being at least 11 times faster based on complexity analysis (Theorems 2 and 3 in Paper 4). The faster convergence of *Approach 3* makes it suitable for constrained computation scenarios with negligible accuracy

loss. The results show that $K_{\max}$ increases with $\bar{\mathcal{E}}$ until a threshold, after which it is constrained by $\bar{\mathcal{L}}$, resulting in constant $K_{\max}$ and test accuracy. Similarly, when $\bar{\mathcal{E}}$ is fixed, increasing $\bar{\mathcal{L}}$ improves $K_{\max}$ and accuracy up to a point, beyond which further increases yield no improvement. These observations highlight two key points: *Approach 3* offers faster implementation with comparable performance to *Approach 2*, and the knee points in the diagrams indicate budget thresholds, after which more resource utilization becomes inefficient.

Table 5.5 presents a comparison of the achievable $K_{\max}$ and test accuracy for *Approach 2* and *Approach 3* against the Dinkelbach and max-sum rate methods. The comparison is conducted under two scenarios: with and without accounting for local computation costs (e.g., energy and latency), considering non-IID local data distribution with $M = 20$ users, $L = 8$, LAQ [44] with $b = 4, 8$, and $b = 32$ for full-precision quantization, and CNN architecture with constrained latency and energy budgets.

In the first scenario, *Approach 2* and *Approach 3* outperform the benchmark methods in both achievable $K_{\max}$ and test accuracy. Using LAQ with $b = 8$, the test accuracy for all methods exceeds that with $b = 4$, as the latter's lower communication overhead allows a larger $K_{\max}$ but suffers from higher quantization error, reducing the overall convergence rate. Thus, smaller bit schemes may increase $K_{\max}$ but do not necessarily improve test accuracy. In contrast, the full-precision case achieves an $K_{\max}$ approximately five times smaller than LAQ with $b = 8$, leading to lower test accuracy. These results underscore the trade-off between quantization schemes and test accuracy, emphasizing the need to balance latency and energy efficiency in FL design.

In the second scenario, we consider the local computation energy and latency along with the uplink energy and latency to investigate the achievable $K_{\max}$ and test accuracy for the four power allocation methods. To this aim, we define $\bar{\mathcal{L}}_T$, $\bar{\mathcal{E}}_T$ as the total latency and energy budgets (uplink + computation) for the FL training. We consider the local computation energy as $E_c^j := \bar{\kappa} L C_c |\mathcal{D}_j| f_c^2$ and the computation latency as $\ell_c^j := L C_c |\mathcal{D}_j| / f_c$, where $C_c = 20000$ (cycles/sample) is the number of processing cycles to execute one sample of data, $f_c = 2$ GHz (cycles/sec) [2] as the central processing unit (computation capacity), $\bar{\kappa} = 10^{-28}$ is the effective switched capacitance that depends on the chip architecture [143]. We consider that all users contain an equal portion of local data samples as $|\mathcal{D}_j| = |\mathcal{D}|/M$, whereas for the CIFAR-10 dataset, $|\mathcal{D}| = 50,000$ samples.

Our proposed *Approach 2* and *Approach 3* outperform benchmark methods, with the smallest differences in $K_{\max}$ observed in LAQ with $b = 4$ and the largest differences in the full-precision case. The second scenario in Table 5.5 highlights the critical impact of per-iteration uplink energy and latency allocation on $K_{\max}$, even under constrained total budgets. For LAQ with $b = 8$, *Approach 2* and *Approach 3* achieve $K_{\max}$ improvements of approximately 14% and 32% over the

[2] $f_c = 2$ GHz is maximum computation capacity that, without loss of generality, we assumed that all users have the equal f_c.

Table 5.5: Comparison of *Approach 2* and *Approach 3* with benchmark methods for $M = 20$.

Methods	First scenario: without computation costs, $\bar{\mathcal{L}} = 25$, $\bar{\mathcal{E}} = 1$						Second scenario: with computation costs, $\bar{\mathcal{L}}_T = 30$, $\bar{\mathcal{E}}_T = 200$					
	LAQ, b = 4		**LAQ, b = 8**		b = 32		**LAQ, b = 4**		**LAQ, b = 8**		b = 32	
	$K_{\max}$	Accuracy (%)	$K_{\max}$	Accuracy (%)	$K_{\max}$	Accuracy	$K_{\max}$	Accuracy (%)	$K_{\max}$	Accuracy (%)	$K_{\max}$	Accuracy (%)
Approach 2	282	74.32	141	77	35	76.31	63	72.4	62	76.1	33	75.86
Approach 3	270	73	138	76.73	34	76.1	61	71.6	59	75.53	29	75.4
FL + Dinkelbach	47	66.42	23	70.9	5	52.2	60	70.5	54	75.4	18	72.3
FL + Max-sum rate	55	69.04	27	71.81	6	56.9	60	70.5	47	74.9	15	70.2

Dinkelbach and max-sum rate methods, respectively. In the full-precision case, the improvements are 62% and 93%. These significant gains in $K_{\max}$ under tight resource constraints demonstrate the effectiveness of *Approach 2* and *Approach 3*, which are particularly beneficial for achieving more accurate FL training, as higher $K_{\max}$ often correlates with improved test accuracy.

5.4.2 Adaptive Quantization and Mitigation of the Straggler Effect

In this subsection, we analyze FedAvg training with local AdaGrad updates over a CFmMIMO network, incorporating a proposed adaptive quantization scheme for local gradients and power control strategies to mitigate the straggler effect. Each user $j \in [M]$ performs L local iterations of AdaGrad updates, following the methodology in [63]. AdaGrad leverages adaptive learning rates that adjust based on gradient magnitudes, assigning larger rates to smaller gradients to ensure balanced progress across dimensions. This characteristic is particularly advantageous for training deep neural networks, as it effectively manages varying gradient scales and prevents excessively large learning rates, thus improving convergence, especially in scenarios with sparse gradients [58].

During local updates, each user performs $h = 1, \ldots, L$ local iterations using a randomly selected subset of $\xi_j \leq |\mathcal{D}_j|$ data samples. The local model, denoted as $\boldsymbol{w}_{h,k}^j \in \mathbb{R}^d$, is computed starting from the initialization $\boldsymbol{w}_{0,k}^j = \boldsymbol{w}_{k-1}$, with the initial accumulated gradient set to $\mathbf{g}_{1,k}^j = \mathbf{0}$. The updates are performed as follows:

$$\begin{aligned} \boldsymbol{w}_{h,k}^j &\leftarrow \boldsymbol{w}_{h-1,k}^j - \alpha \boldsymbol{\nabla}_{h-1,k}^j \oslash \left(\mathbf{g}_{h,k}^j + \epsilon_a\right)^{\circ \frac{1}{2}}, \ k \in [K], j \in [M], \\ \mathbf{g}_{h,k}^j &\leftarrow \mathbf{g}_{h-1,k}^j + \boldsymbol{\nabla}_{h-1,k}^j \odot \boldsymbol{\nabla}_{h-1,k}^j, \ k \in [K], j \in [M], \end{aligned} \tag{5.49}$$

where $\boldsymbol{\nabla}_{h-1,k}^j := \sum_{n \in [\xi_j]} \nabla \boldsymbol{w} f(\boldsymbol{w}_{h-1,k}^j; \boldsymbol{x}_{nj}, y_{nj})$ represents the cumulative gradient over the local data subset, $\mathbf{g}_{h,k}^j$ is the accumulated sum of squared gradients, α is the base step size, and ϵ_a is a small constant added for numerical stability to avoid division by zero. The final local model after L iterations is denoted as $\boldsymbol{w}_k^j = \boldsymbol{w}_{L,k}^j$. Once $\boldsymbol{w}_k^j$ is computed, user j evaluates the local gradient difference $\delta \boldsymbol{w}_k^j := \boldsymbol{w}_k^j - \boldsymbol{w}_{k-1}$. This gradient is quantized using the operation $\delta \boldsymbol{q}_k^j := \text{Quant}(\delta \boldsymbol{w}_k^j)$ and transmitted to the central server via a CFmMIMO system. The APs relay the received signals to the server over error-free fronthaul links, as depicted in Figure 5.8.

Using the aggregated local gradients $\delta \boldsymbol{q}_k^j$ from all M users, the central server updates the global model as follows:

$$\boldsymbol{w}_k = \boldsymbol{w}_{k-1} + \sum_{j=1}^{M} \rho_j \ \delta \boldsymbol{q}_k^j, \ k \ \in \ [K]. \tag{5.50}$$

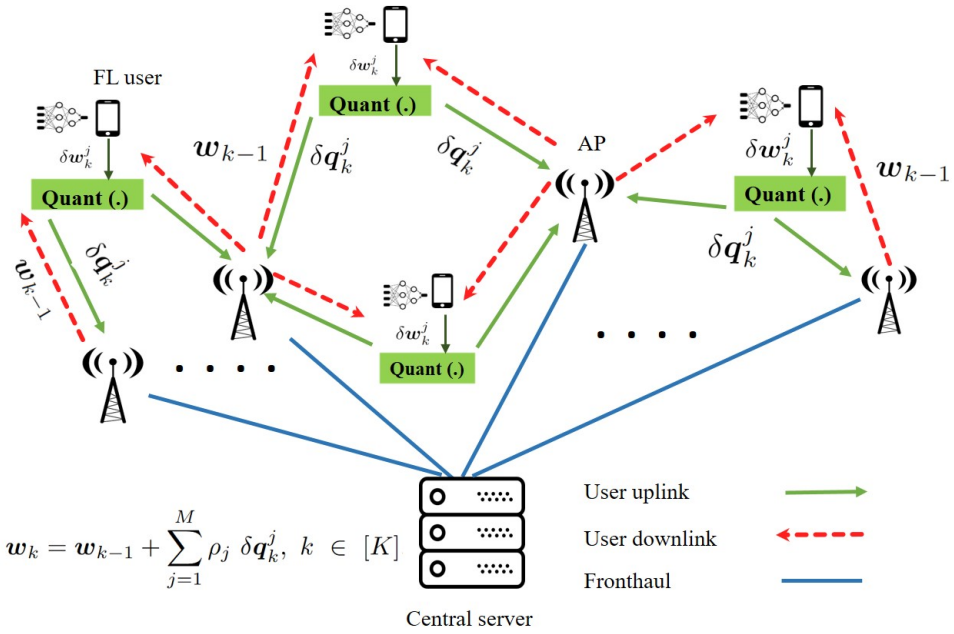


Figure 5.8: General architecture of FL over CFmMIMO with local model quantization.

Afterward, the central server sends the global model $\boldsymbol{w}_k$ to the APs via the fronthaul links. Next, the APs jointly send $\boldsymbol{w}_k$ to the users in the downlink. Finally, each user $j \in [M]$ computes its new local model and begins the next FL iteration.

Adaptive Mixed-Resolution Quantization Scheme

We propose a novel quantization scheme for the uplink transmission of $\delta \boldsymbol{w}_k^j$, $j \in [M], k \in [K]$. This approach leverages the inherent sparsity observed in local gradients [43,133,134], where many elements are near-zero (referred to as low-resolution elements). Our method introduces an element-wise quantization process that categorizes elements into two groups: high-resolution and low-resolution. For high-resolution elements, user j applies uniform quantization using $b_j \geq 2$ bits, determined by a magnitude ratio threshold λ_j. Conversely, the remaining low-resolution elements are assigned a single bit (0 or 1) to capture their sign. The adaptiveness of this scheme stems from the varying number of high-resolution elements for each user j, dictated by the values in $\delta \boldsymbol{w}_k^j$. For elements $i = 1, \ldots, d$, we define:

$$\bar{d}_k^j := \text{Number of elements } i \text{ such that } \frac{|[\delta \boldsymbol{w}_k^j]_i|}{\|\delta \boldsymbol{w}_k^j\|_\infty} \geq \lambda_j, \tag{5.51}$$

Where $\|\delta \boldsymbol{w}_k^j\|_\infty := \max_{i\in[d]} |[\delta \boldsymbol{w}_k^j]_i|$ denotes the infinity norm of the vector $\delta \boldsymbol{w}_k^j$. We introduce a bit allocation vector $\boldsymbol{b}_k^j \in \mathbb{R}^{d\times 1}$, which specifies the number of bits assigned to each element $i = 1, \ldots, d$ of $\delta \boldsymbol{w}_k^j$, for every user $j = 1, \ldots, M$, and iteration $k = 1, \ldots, K$. The bit assignment is defined as follows:

$$[\boldsymbol{b}_k^j]_i = \begin{cases} 1, & \text{if } \frac{|[\boldsymbol{w}_k^j]_i|}{\|\delta \boldsymbol{w}_k^j\|_\infty} < \lambda_j, \text{ and } [\delta \boldsymbol{w}_k^j]_i > 0, \\ 0, & \text{if } \frac{|[\delta \boldsymbol{w}_k^j]_i|}{\|\delta \boldsymbol{w}_k^j\|_\infty} < \lambda_j, \text{ and } [\delta \boldsymbol{w}_k^j]_i \le 0, \\ Q(|[\delta \boldsymbol{w}_k^j]_i|), & \text{if } \frac{|[\delta \boldsymbol{w}_k^j]_i|}{\|\delta \boldsymbol{w}_k^j\|_\infty} \ge \lambda_j, \end{cases} \tag{5.52}$$

where $Q(\cdot)$ denotes the uniform quantization of the elements satisfying (5.51) with b_j bits (including one bit for the sign) and the grid radius of $r_k^j := \|\delta \boldsymbol{w}_k^j\|_\infty - \delta w_{q,k}^j$, with $\delta w_{q,k}^j$ as the absolute value of the smallest element that satisfy (5.51). The bits 0 and 1 represent the signs of the elements that do not satisfy (5.51), with 0 indicating a negative sign and 1 indicating a positive sign. Defining s_k^j as

$$s_k^j := \bar{d}_k^j / d, \tag{5.53}$$

the total number of quantized bits for user j at iteration k is obtained as

$$b_k^j := d(b_j s_k^j + 1 - s_k^j) + 32, \tag{5.54}$$

where 32 bits are used to send r_k^j. Note that all elements that do not satisfy (5.51) have absolute values smaller than $\delta w_{q,k}^j$. Consequently, the central server receives $\delta \boldsymbol{q}_k^j$, where the elements $i \in [d]$ are given by:

$$[\delta \boldsymbol{q}_k^j]_i = \begin{cases} +\delta q_{q,k}^j/2, & \text{if } [\boldsymbol{b}_k^j]_i = 1, \\ -\delta q_{q,k}^j/2, & \text{if } [\boldsymbol{b}_k^j]_i = 0, \\ [\delta \boldsymbol{q}_k^j]_i, & \text{otherwise}, \end{cases} \tag{5.55}$$

where $\delta q_{q,k}^j$ is the quantized value of $\delta w_{q,k}^j$ with b_j bits. Since we apply one-bit uniform quantization with a radius grid of $|\delta q_{q,k}^j|$ for any element i that does not satisfy (5.51), the quantized values for these elements are $\delta q_{q,k}^j$ divided by 2.

Lemma 3. *Let $\delta \boldsymbol{q}_k^j$ be the quantized local gradient vector corresponding to $\delta \boldsymbol{w}_k^j$ of user j at FL iteration t. Let $\boldsymbol{\varepsilon}_k^j := \delta \boldsymbol{w}_k^j - \delta \boldsymbol{q}_k^j$ be the quantization error vector after quantizing $\delta \boldsymbol{w}_k^j$. For any $j \in [M], k \in [K]$, we obtain $\|\boldsymbol{\varepsilon}_k^j\|_\infty \le \bar{\gamma}_j \ \|\delta \boldsymbol{w}_k^j\|_\infty$, where*

$$\bar{\gamma}_j := \max\left\{ \frac{\lambda_j}{2} + \frac{1-\lambda_j}{4(2^{b_j}-1)}, \ \frac{1-\lambda_j}{2(2^{b_j}-1)} \right\}. \tag{5.56}$$

Proof. The proof is given in Appendix A of Paper 5. □

Lemma 3 investigates the error in our mixed-resolution quantization method, showing that the quantization error decreases as the norm of the local gradient decreases. Building on this lemma, the following proposition demonstrates the convergence of FL with AdaGrad local updates.

Proposition 3. *Let $f^j(\boldsymbol{w})$ be the local loss function of user $j \in [M]$ in the FL training with the local AdaGrad updates in (5.49), $\delta \boldsymbol{q}_k^j = \delta \boldsymbol{w}_k^j - \varepsilon_k^j$ is the quantized local vector with the quantization error defined in (5.56), and define $c_{\max} := \max_j c_j$. Suppose $f^j(\boldsymbol{w})$ and $f^j(\boldsymbol{w})$ are $\bar{L}$-smooth, for all $j \in [M]$, with G-bounded gradients and σ_l-bounded (local) variance and the (global) variance σ_g is bounded, similar to Assumptions 1-3 in [63]. Let $\Delta_f := f(\boldsymbol{w}_0) - f(\boldsymbol{w}^*)$, $\bar{\sigma}^2 := 6L\sigma_g^2 + \sigma_l^2$, $\alpha \leq \min\{(16L\bar{L})^{-1}, (2L\sqrt{\bar{L}})^{-1}\}$, and the conditions I-II in [63] hold. It then holds that*

$$\min_{k \in [K]} \mathbb{E}_k \left\|\nabla f(\boldsymbol{w}_k)\right\|^2 \leq \mathcal{O}\left(\frac{1 + \alpha LG\sqrt{K}}{\alpha LK} \cdot \Phi\right), \tag{5.57}$$

$$\Phi := 2LK\bar{\sigma}^2 + 4L^2 + Mc_{\max}^2(2L + K\bar{\sigma}^2) + \alpha^{-1}\Delta_f.$$

Proof. The proof steps are similar to the proof of [Th 1 in [63]] with the same assumptions. We substitute Δ_k^j with $\delta \boldsymbol{q}_k^j$ containing the quantization errors, and assume $\nu_k = 0, \tau = 1$. □

Optimization Problem for Power Control

We present the proposed optimization problem that mitigates the straggler effect when using the proposed quantization scheme. Our goal is to obtain the uplink powers p_k^j, $\forall j, k$, that minimize the latency of the slowest user:

$$\underset{p_k^1, \ldots, p_k^M}{\text{minimize}} \quad \max_{j \in [M]} \quad \ell_k^j \tag{5.58a}$$

$$\text{subject to} \quad 0 \leq p_k^j \leq 1, \quad \forall j \in [M], \ \forall k \in [K], \tag{5.58b}$$

where ℓ_k^j is the uplink latency of each user j at iteration k, as

$$\ell_k^j := \frac{b_k^j}{R_k^j} = \frac{d\left(b_j s_k^j + 1 - s_k^j\right) + 32}{B_\tau \log_2\left(1 + \text{SINR}_k^j\right)}. \tag{5.59}$$

We rewrite the min-max optimization problem in (5.58) as

$$\underset{p_k^1, \ldots, p_k^M}{\text{maximize}} \quad \min_{j \in [M]} \quad \frac{B_\tau \log_2(1 + \text{SINR}_k^j)}{b_k^j} \tag{5.60a}$$

$$\text{subject to} \quad 0 \leq p_k^j \leq 1, \quad \forall j \in [M], \ \forall k \in [K]. \tag{5.60b}$$

We can write (5.60) in epigraph form as

$$\underset{p_k^1,\ldots,p_k^M,\bar{\eta}_k}{\text{maximize}} \quad \bar{\eta}_k \tag{5.61a}$$

$$\text{subject to} \quad 0 \leq p_k^j \leq 1, \quad \forall j \in [M], \ \forall k \in [K], \tag{5.61b}$$

$$\left(\bar{A}_j - \theta_k^j \bar{B}_j\right) p_k^j - \theta_k^j \sum_{j' \neq j}^{M} p_k^{j'} \tilde{B}_j^{j'} \geq \theta_k^j I_M^j. \tag{5.61c}$$

by letting the new optimization variable $\bar{\eta}_k$ be a lower bound on the rate-per-bit ratio at FL iteration k:

$$\bar{\eta}_k \leq \min_{j \in [M]} \quad \frac{B_\tau \log_2(1 + \text{SINR}_k^j)}{b_k^j}. \tag{5.62}$$

We obtained (5.61c) by expressing (5.62) as $\text{SINR}_k^j \geq 2^{\bar{\eta}_k b_k^j / B_\tau} - 1, \ \forall \ j, \ k$ and then defining $\theta_k^j := 2^{\bar{\eta}_k b_k^j / B_\tau} - 1 \geq 0$.

To solve (5.61), we note that the constraint (5.61c) is linear in the power variables but depends on $\bar{\eta}_k$ through θ_k^j. Thus, at each iteration k, we propose solving (5.61) using a combination of the bisection method over $\bar{\eta}_k$ and LP for p_k^j, $j = 1, \ldots, M$. This approach finds the global optimum to (5.61) to any desired accuracy $\epsilon_B > 0$. In the following, we present some selective numerical results to illustrate the solutions. The full numerical results can be found in Paper 5 in Chapter 7 of this thesis.

Illustrative Numerical Results

Here, we present some selective numerical results, which the simulation parameters are shown in Table 5.3. We consider that users train a CNN on local datasets, which can be either IID or non-IID. The CNN architecture is standardized across all users to ensure consistency. For CIFAR-10 and CIFAR-100, the input layer processes data with a shape of $(32, 32, 3)$, while for Fashion-MNIST, the input is pre-processed from grayscale to three channels, resulting in a shape of $(28, 28, 3)$. The architecture begins with a Conv2D layer comprising 32 filters of size $(3, 3)$ with ReLU activation, followed by a MaxPooling2D layer with a pool size of $(2, 2)$ [60]. After the convolutional and pooling operations, a Flatten layer reshapes the data into a 1D array for the subsequent Dense layers. The first Dense layer consists of 64 units with ReLU activation, while the final Dense layer includes n_2 units with softmax activation for multiclass classification, where $n_2 = 10$ for CIFAR-10 and Fashion-MNIST, and $n_2 = 100$ for CIFAR-100. This architecture is designed to effectively capture spatial features for multiclass classification tasks.

We define the average percentage of high-resolution elements as $s := 100 \cdot \mathbb{E}_{k,j} s_k^j$.

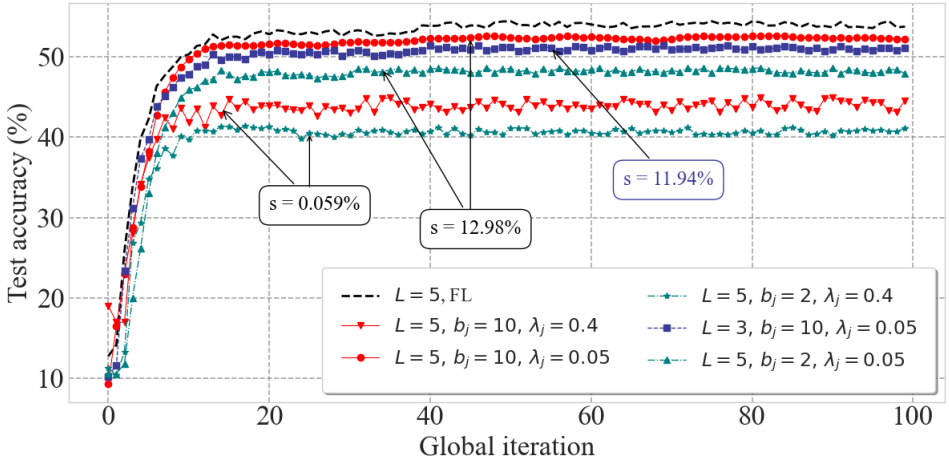


Figure 5.9: Convergence of FL with mixed-resolution quantization vs. classic FL.

We calculate the average reduction in communication overhead (in %) as

$$\bar{r} = 100 \cdot \left(1 - \frac{\mathbb{E}_{k,j}\left\{b_k^j\right\}}{b_1 d}\right), \ j = 1, \ldots, M, \tag{5.63}$$

where b_1 is the number of element-wise bits for the benchmarks (32 for classic FL, $b_1 = b_j$ for LAQ and Top-q). Considering the definition of s, we obtain $\bar{r}$ (in %) as

$$\bar{r} := 100 - \frac{100}{b_1} - s\frac{b_j - 1}{b_1}, \ j = 1, \ldots, M, \tag{5.64}$$

as the reduction in communication overhead considering the proposed mixed-resolution quantization compared to the benchmarks.

Figure 5.9 illustrates the convergence behavior of FL on the CIFAR-10 dataset with a non-IID data distribution across users, employing the proposed mixed-resolution quantization scheme. The annotation boxes indicate the average percentage of high-resolution elements. The results reveal that the proposed scheme achieves a convergence rate comparable to classical FL, maintaining similar test accuracy for $\lambda_j = 0.05$. Notably, the communication overhead is reduced by $\bar{r} = 93\%$, demonstrating that even with a small value of λ_j, substantial reductions in overhead are achievable without compromising performance.

To assess the effectiveness of the proposed mixed-resolution quantization scheme across multiple datasets, Table 5.6 summarizes the test accuracy of FL with and without the scheme under both IID and non-IID data distributions for CIFAR-10, CIFAR-100, and Fashion-MNIST. The results demonstrate that the proposed mixed-resolution quantization achieves test accuracy comparable to classic FL while reducing the communication overhead by at least $\bar{r} = 96\%$.

Table 5.6: Performance analysis of FL with mixed-resolution quantization for CIFAR-10, CIFAR-100, and Fashion-MNIST datasets with non-IID and IID data distributions, $M = 20$, $L = 5$, $b_j = 10$, $\lambda_j = 0.2$, $K = 100$.

	Non-IID			**IID**		
	Accuracy (Our approach)	s (%)	Accuracy (FL)	Accuracy (Our approach)	s (%)	Accuracy (FL)
CIFAR-10	51.86	0.8574	52.18	58.92	1.827	59.64
CIFAR-100	27.9	1.064	28.12	32.26	1.089	32.49
Fashion-MNIST	89.29	0.711	89.65	91.23	0.9993	91.73

Table 5.7: Performance comparison of our proposed method with the benchmarks, $M = 40$, $L = 5$, $b_j = 4$, $\lambda_j = 0.4$ for non-IID distributed CIFAR-10, with total latency budget of $\bar{\ell} = 3$ s, and calculated $s = 0.044\%$.

	LAQ		**Top-q**		**AQUILA**		**Mixed-resolution**	
	Accuracy (%)	$K_{\max}$	Accuracy (%)	$K_{\max}$	Accuracy (%)	$K_{\max}$	Accuracy (%)	$K_{\max}$
Our power control	25.4%	17	36.34%	38	27.77%	16	**46.13**%	27
Dinkelbach	22.23%	11	33.7%	36	26.72%	14	42.8%	24

Finally, we analyze the performance of our proposed power control to mitigate the straggler effect and compare the results with benchmarks. We compare our approach with the Dinkelbach method [48] and evaluate the communication overhead of gradient transmission using the proposed mixed-resolution method alongside AQUILA [49], LAQ [44], and Top-q sparsification [96], where only the q largest magnitude entries are transmitted. Total latency is computed as the sum of uplink latency and local gradient computation time, where the maximum computation time for users is $\ell_c = L|\mathcal{D}|a_j/(M\nu_j)$ [136], with $\nu_j = 20$ cycles/s, $a_j = 10^6$ cycles/sample, and 5×10^4 samples distributed non-IID for CIFAR-10. The maximum iterations within a training latency budget $\bar{\ell}$, denoted as $K_{\max}$, are shown in Table 5.7. Our proposed method achieves higher test accuracy than AQUILA, LAQ, and Top-q (with $q = s$), demonstrating the impact of considering the signs for low-resolution elements for training convergence. Additionally, as indicated in Table 5.7, our power control method surpasses Dinkelbach by effectively mitigating the straggler effect, yielding a higher $K_{\max}$ under latency constraints. Across communication methods, our approach reduces overhead by $\bar{r} = 75\%$ compared to AQUILA and LAQ, enabling more iterations and faster convergence under limited latency.

5.4.3 Adaptive Exponent-Mantissa Quantization and Power Control

In this subsection, we present an energy-efficient and low-latency FedAvg training with AdaDelta local updates over a CFmMIMO network and introduce an adaptive

Exponent-Mantissa Quantization (EMQ) scheme for local gradients.

The general system model of this subsection is similar to the system model in Figure 5.8. We assume that each user $j \in [M]$ performs l_k^j local iterations of SGD with AdaDelta updates [58] using a randomly chosen subset ξ of size $|\xi| \leq |\mathcal{D}_j|$. Integrating AdaDelta with FL complements the adaptive quantization scheme, which allocates more bits to higher-magnitude gradient elements and fewer bits to near-zero ones. AdaDelta's per-parameter adaptive learning rates emphasize significant gradients by scaling updates based on their historical impact. This synergy ensures critical gradient information is preserved with higher precision while efficiently compressing minor updates, addressing the communication constraints of iterative gradient transfer in FL.

In the context of AdaDelta update, we define $E[\boldsymbol{g}^2]_{l,k}^j$ as the running average at the local iteration $l \leq l_k^j$, for each user j at global iteration k, and compute it as

$$E[\boldsymbol{g}^2]_{l,k}^j = \rho E[\boldsymbol{g}^2]_{l-1,k}^j + (1-\rho)\boldsymbol{g}_{l-1,k}^j \odot \boldsymbol{g}_{l-1,k}^j, \tag{5.65}$$

where $\rho \in (0,1)$ is a decay constant and $\boldsymbol{g}_{l-1,k}^j$ is the local gradient vector, as

$$\boldsymbol{g}_{l-1,k}^j := \frac{1}{|\xi|} \sum_{n \in [|\xi|]} \nabla_w f(\boldsymbol{w}_{l-1,k}^j; \boldsymbol{x}_{nj}, y_{nj}). \tag{5.66}$$

To refine the step size for updating $\boldsymbol{w}_{l,k}^j$, the AdaDelta procedure requires the element-wise square root of $E[\boldsymbol{g}^2]_{l,k}^j$, and it effectively becomes the RMS of the previous squared gradients up to the local iteration l, as

$$\mathrm{RMS}[\boldsymbol{g}]_{l,k}^j = \left(E[\boldsymbol{g}^2]_{l,k}^j + \epsilon_a\right)^{\circ \frac{1}{2}}, \tag{5.67}$$

where $\epsilon_a > 0$ is a small constant to prevent division by zero. Considering $\boldsymbol{w}_{0,k}^j = \boldsymbol{w}_{k-1}$, we obtain

$$\boldsymbol{w}_{l,k}^j = \boldsymbol{w}_{l-1,k}^j - \alpha\, \boldsymbol{g}_{l-1,k}^j \oslash \mathrm{RMS}[\boldsymbol{g}]_{l,k}^j,\ l = 1, \ldots, l_k^j, \tag{5.68}$$

where $\alpha > 0$ is the preliminary step size and we set $\boldsymbol{w}_k^j = \boldsymbol{w}_{l,k}^j\big|_{l=l_k^j}$.

After obtaining $\boldsymbol{w}_k^j$, each user $j \in [M]$ computes $\delta \boldsymbol{w}_k^j$ defined as the difference between the current local model $\boldsymbol{w}_k^j$ and the last global model $\boldsymbol{w}_{k-1}$, as

$$\delta \boldsymbol{w}_k^j := \boldsymbol{w}_k^j - \boldsymbol{w}_{k-1}, \quad \forall k, \forall j. \tag{5.69}$$

Then, each user j quantizes each element of $\delta \boldsymbol{w}_k^j$ according to the quantization scheme we will present in the next subsection, as

$$\delta \boldsymbol{q}_k^j := \mathrm{Quant}(\delta \boldsymbol{w}_k^j), \quad \forall k, \forall j, \tag{5.70}$$

and sends $\delta \boldsymbol{q}_k^j$ to the server with b_k^j bits. Finally, the server updates the global model $\boldsymbol{w}_k$ as

$$\boldsymbol{w}_k = \boldsymbol{w}_{k-1} + \sum_{j=1}^{M} \rho_j \delta \boldsymbol{q}_k^j, \quad \forall k. \tag{5.71}$$

Next, we explain the proposed EMQ scheme to quantize $\delta \boldsymbol{w}_k^j$ and send it to the server with a limited number of bits. Then, we present the main optimization problem and the solution approach for the power control.

Exponent-Mantissa Quantization

The gradient $\delta \boldsymbol{w}_k^j$ is a high-dimensional vector with many near-zero elements that add little to the global model but significantly increase communication overhead. To address this issue, we propose EMQ, an adaptive scheme that reduces communication costs by allocating fewer bits to near-zero elements and more bits to significant ones, leveraging the inherent sparsity of local gradients [43,133]. Inspired by [144], EMQ identifies a shared exponent for all elements and assigns unique significant digits (mantissa) to each element, efficiently compressing the vector.

The quantization scheme expresses every element i of the vector $\delta \boldsymbol{w}_k^j$ as

$$[\delta \boldsymbol{w}_k^j]_i := s_{k,i}^j \cdot \bar{\delta}_{k,i}^j \cdot 10^{u_k^j}, \tag{5.72}$$

where $s_{k,i}^j \cdot \bar{\delta}_{k,i}^j$ represents the mantissa of the i^{th} element of $\delta \boldsymbol{w}_k^j$ with $s_{k,i}^j \in \{-1, +1\}$ indicating the sign of $[\delta \boldsymbol{w}_k^j]_i$ and $\bar{\delta}_{k,i}^j \in [0, 10)$. The term u_k^j denotes the exponent, which satisfies $|u_k^j| \in \mathbb{Z}^+$. To compute u_k^j, we consider the infinite norm of each vector $\delta \boldsymbol{w}_k^j$ as $\|\delta \boldsymbol{w}_k^j\|_\infty$ and compute the placement of the decimal point of $\|\delta \boldsymbol{w}_k^j\|_\infty$, as follows

$$u_k^j := \lfloor \log_{10} \|\delta \boldsymbol{w}_k^j\|_\infty \rfloor. \tag{5.73}$$

To obtain $\delta \boldsymbol{q}_k^j$, $[\boldsymbol{b}_k^j]_i$, and b_k^j, the following steps are performed:

1. Each user calculates u_k^j using (5.73), representing the shared exponent for $\delta \boldsymbol{w}_k^j$. This value is quantized using up to $b_u = 8$ bits, following the IEEE 754 binary32 format.

2. Each user assigns one bit to $s_{k,i}^j$ to represent the sign, where 0 denotes a negative value and 1 a positive value.

3. The mantissa $[\bar{\delta}_{k,i}^j] \in 0, 1, \ldots, 9$ [3] is computed and encoded adaptively. Few bits are allocated for common values like 0 and 1, while up to 5 bits are used for rarer, larger values with a prefix-free encoding starting with 11 for these cases (see Figure 5.10).

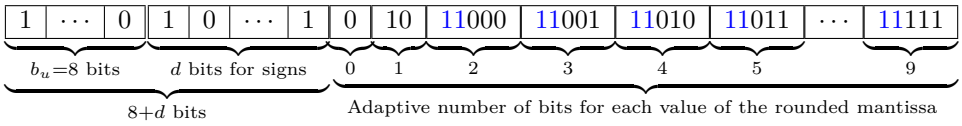


Figure 5.10: Illustration of the EMQ bit sequence assigned by each client $j \in [M]$ to $\delta \boldsymbol{q}_k^j$ at each global iteration k. Colored bits indicate the prefix used for quantizing higher-magnitude rounded mantissa $[\bar{\delta}_{k,i}^j]$.

Figure 5.10 illustrates the bit sequence used by each user j to transmit $\delta \boldsymbol{q}_k^j$ to the server. The sequence begins with $b_u + d = 8 + d$ bits to encode the shared exponent and the sign of each mantissa. The adaptive design of EMQ, leveraging a common exponent and rounded mantissa quantization, distinguishes it from the IEEE 754 binary32 single-precision floating-point standard.

We define $\boldsymbol{b}_k^j \in \mathbb{R}^d$ as the element-wise bit vector representing the mantissa values of the elements of $\delta \boldsymbol{q}_k^j$, and the scalar $b_k^j := b_u + d \cdot 1 + \sum_{i=1}^d [\boldsymbol{b}_k^j]_i$ as the total number of bits each user j transmits to the server at every FL iteration k. Therefore, each user j spends $b_k^j = b_u + d + \sum_{i=1}^d [\boldsymbol{b}_k^j]_i \leq 8 + d + 5d = 8 + 6d \approx 6d$ to send the quantized version of $\delta \boldsymbol{w}_k^j$, defined as

$$\delta \boldsymbol{q}_k^j = \text{Quant}(\delta \boldsymbol{w}_k^j) = \boldsymbol{s}_k^j \odot \bar{\boldsymbol{\delta}}_k^j \odot \mathbf{1}_k^j, \tag{5.74}$$

where $\mathbf{1}_k^j := 10^{u_k^j} \cdot \mathbf{1}_d$ represents a d-dimensional vector where every element equals $10^{u_k^j}$. The vectorized version of the rounded mantissa is given by $[\bar{\boldsymbol{\delta}}_k^j]i = [\bar{\delta} k, i^j]$, and the sign vector is $[\boldsymbol{s}_k^j]i := sk, i^j$ for $i = 1, \ldots, d$. Defining $\boldsymbol{\varepsilon}_k^j := \delta \boldsymbol{w}_k^j - \delta \boldsymbol{q}_k^j$ as the local quantization error, the aggregated quantization error is expressed as $\boldsymbol{\varepsilon} k := \sum_{j=1}^M \rho_j \boldsymbol{\varepsilon}_k^j$. It has been shown in Lemma 1 in Paper 6 that defining $\varepsilon_k^j := \bar{\boldsymbol{\varepsilon}}_k^j \cdot 10^{u_k^j - 1}$, we obtain $\|\bar{\boldsymbol{\varepsilon}}_k^j\|_\infty < 5$.

Considering the FedAvg with local AdaDelta updates and the EMQ scheme, we propose the main optimization problem of this subsection.

Main Optimization Problem

We consider the FedAvg algorithm with AdaDelta local updates integrated with our proposed EMQ scheme and formulate an optimization problem to determine the optimal communication parameters. After computing the local model $\boldsymbol{w}_k^j$, each user $j \in [M]$ calculates $\delta \boldsymbol{w}_k^j$, quantizes it to $\delta \boldsymbol{q}_k^j$ with b_k^j bits, and transmits $\delta \boldsymbol{q}_k^j$ to the central server. Let K denote the number of global iterations that we aim to obtain and define the following matrices:

[3]To facilitate using the EMQ bit sequence, we map the mantissa value of 10 to 9 without any loss of generality. The numerical results do not change by this mapping.

- $\boldsymbol{P} \in \mathbb{R}^{K\times M}$ as the matrix of power coefficients, $\boldsymbol{P} := [p_k^j]_{k,j}$,
- $\boldsymbol{w}_K$ as the final global model,
- $\boldsymbol{L} \in \mathbb{R}^{K\times M}$ as the matrix of local iteration counts, $\boldsymbol{L} := [l_k^j]_{k,j}$, and
- K as the stopping iteration (total number of global iterations).

Then, assuming $\rho_j = 1/M$, we state the following optimization problem:

$$\underset{\boldsymbol{P},\boldsymbol{w}_K,\boldsymbol{L},K}{\text{minimize}} \quad f(\boldsymbol{w}_K) \tag{5.75a}$$

$$\text{subject to} \quad \boldsymbol{w}_k = \boldsymbol{w}_{k-1} + \frac{1}{M}\sum_{j=1}^{M} \delta \boldsymbol{q}_k^j, \quad k \in [K]\,, \tag{5.75b}$$

$$l_k^j \geq 1, \quad k \in [K], j \in [M]\,, \tag{5.75c}$$

$$p_k^j \in [0,1], \quad k \in [K], j \in [M]\,, \tag{5.75d}$$

$$\sum_{k=1}^{K}\sum_{j=1}^{M} E_k^j \leq \mathcal{E}\,, \tag{5.75e}$$

$$\sum_{k=1}^{K} \ell_k^{\max} \leq \bar{\mathcal{L}}, \tag{5.75f}$$

where E_k^j denotes the uplink energy expended by user $j \in [M]$ at global iteration k. The total energy and latency budget for uplink are represented by $\mathcal{E}$ and $\bar{\mathcal{L}}$, respectively. The uplink latency of the slowest user at iteration k is $\ell_k^{\max}$, and p_k^j is the uplink power coefficient for user j. The uplink data rate R_k^j for each user is given by (3.8), and the uplink latency for each user is calculated as $\ell_k^j = b_k^j / R_k^j$ (in seconds), where

$$\ell_k^{\max} := \max_{j\in[M]} \ell_k^j, \quad k \in [K]\,, \tag{5.76}$$
$$E_k^j := p_k^j p^u \ell_k^j, \quad k \in [K],$$

and E_k^j is the uplink energy for every user $j \in [M]$.

The optimization problem in (5.75) is difficult to solve beforehand as it depends on parameters like p_k^j, l_k^j, b_k^j, and $\delta \boldsymbol{w}_k^j$ for each user $j \in [M]$ and iteration k, none of which are known prior to training. To address this, inspired by [136], we propose solving (5.75) iteratively at each step k, using only current iteration data. The goal is to compute $\boldsymbol{w}_k$, $l_k^1, \ldots, l_k^M$, and $p_k^1, \ldots, p_k^M$ to minimize $f(\boldsymbol{w}_k)$ while incorporating the proposed quantization scheme. Accordingly, we decompose (5.75) into two sub-problems for each iteration k:

1. **Sub-problem 1:** Optimize $\boldsymbol{w}_k$ and l_k^j for each user $j \in [M]$.

2. **Sub-problem 2:** Determine p_k^j and $\ell_k^{\max}$ for each user to minimize uplink energy and latency.

This separation is justified because p_k^j and $\ell_k^{\max}$ affect the total number of iterations K indirectly through energy and latency constraints rather than directly influencing $\boldsymbol{w}_k$ or l_k^j. Meanwhile, the quantization bits b_k^j—calculated locally after completing the local updates—are necessary for determining the power allocation in Sub-problem 2. Thus, the process begins with Sub-problem 1, where l_k^j is computed for each user. After performing l_k^j local iterations and obtaining the $\delta \boldsymbol{w}_k^j$, each user j calculates b_k^j, which serves as an input for Sub-problem 2, addressing power allocation and uplink latency. The two sub-problems, $\mathcal{P}1$ and $\mathcal{P}2$, are described in detail below. The first sub-problem $\mathcal{P}1$ is as follows:

$$(\mathcal{P}1) \quad \underset{\boldsymbol{w}_k, l_k^1, \ldots, l_k^M}{\text{minimize}} \quad f(\boldsymbol{w}_k) \tag{5.77a}$$

$$\text{subject to} \quad \boldsymbol{w}_k = \boldsymbol{w}_{k-1} + \frac{1}{M} \sum_{j=1}^{M} \delta \boldsymbol{q}_k^j \,, \tag{5.77b}$$

$$l_k^j \geq 1, \quad j \in [M]. \tag{5.77c}$$

To solve $\mathcal{P}1$, we perform FL local training using (5.68) and the global update in (5.71) at each iteration k. Determining l_k^j for each user $j \in [M]$ involves analyzing the behavior of quantized local gradients under the EMQ scheme, with particular attention to the local iterations l_k^j. The following proposition outlines the necessary conditions for finding the optimal l_k^j in the context of $\mathcal{P}1$ as stated in (5.77).

Proposition 4. *Let $\boldsymbol{g}_{l,k}^j$ be the gradient vector of user $j \in [M]$ at local iteration $l = 1, \ldots, l_k^j$ and at every global iteration k. We recall the AdaDelta updates according to* (5.65)-(5.68)*, and the definition of u_k^j from* (5.73)*. Let $\alpha \in (0, 1)$ be the primary step size. Then, for a sufficiently large $l_{K^{\max}}$ and any $\alpha < 1/\sqrt{10}$, there exists l_k^j such that $l_{K^{\max}} > l_k^j \geq l_{k-1}^j \geq 2$ by which we obtain $u_k^j \leq u_{k-1}^j$, for all $k \in [K^{\max}]$, and $j \in [M]$.*

Proof. The proof is given in Appendix C of Paper 6. □

Proposition 4 shows that for each user j, with local AdaDelta updates, a non-decreasing sequence of l_k^j over k results in a non-increasing sequence of u_k^j. Since the infinity norm of the quantization error ε_k^j is influenced by u_k^j, the quantization error in the EMQ scheme diminishes or even vanishes as u_k^j decreases, particularly for large K. This behavior ensures that the impact of quantization is reduced with more global iterations. Using Proposition 4, we derive the following remark to determine the optimal $l_k^1, \ldots, l_k^M$ for solving sub-problem $\mathcal{P}1$.

Remark 2. *Let us define* $\Delta \boldsymbol{w}_{l,k}^j := \bar{\Delta} \boldsymbol{w}_{l,k}^j \times 10^{u_{l,k}^j} = \boldsymbol{w}_{l,k}^j - \boldsymbol{w}_{l-1,k}^j$, *where* $u_{l,k}^j := \lfloor \log_{10} |\Delta \boldsymbol{w}_{l,k}^j|_\infty \rfloor$. *Based on Proposition 4, which demonstrates that for each user* $j \in [M]$, *a non-increasing sequence of* l_k^j *w.r.t.* k *leads to a non-increasing sequence of* u_k^j, *we obtain:*

$$l_k^j := \textit{The first } l \;\Big|\; l \geq l_{k-1}^j, \textit{and } u_{l,k}^j \leq u_{k-1}^j, \tag{5.78}$$

and obtain $u_k^j := u_{l_k^j,k}^j$.

Remark 2 outlines a method for determining l_k^j, $j \in [M]$, to solve the optimization in (5.77) for sub-problem $\mathcal{P}1$.

Next, we address the second sub-problem, which focuses on optimizing uplink power allocation while adhering to the constraints of limited uplink energy and latency budgets, as

$$(\mathcal{P}2) \quad \underset{p_k^1,\ldots,p_k^M}{\text{minimize}} \quad \theta_l \left(\max_{j \in [M]} \ell_k^j \right) + \theta_E \sum_{j=1}^{M} E_k^j \tag{5.79a}$$

$$\text{subject to} \quad 0 \leq p_k^j \leq 1, \quad j \in [M]. \tag{5.79b}$$

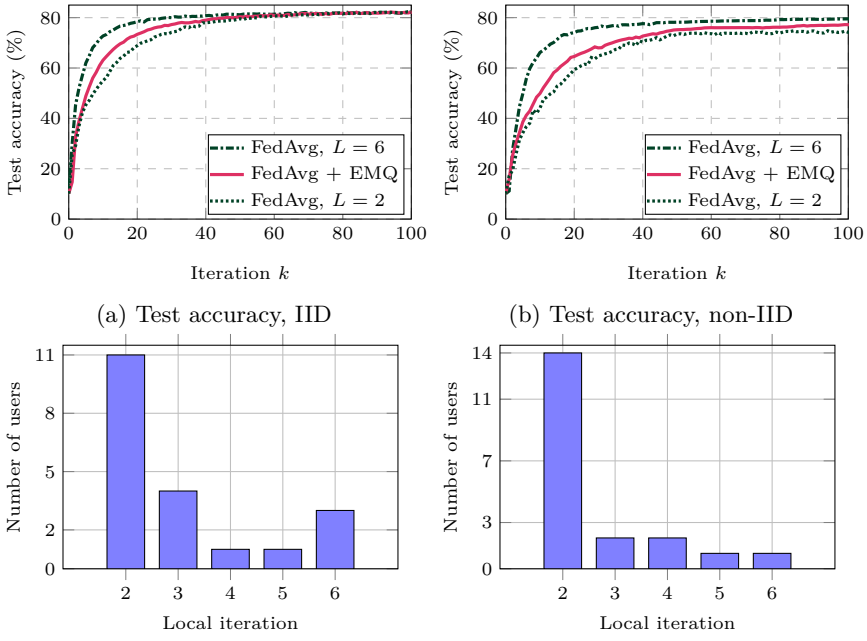
Optimization problem (5.79) captures the trade-off between uplink latency and total uplink energy at each global iteration k. The scalarization weights $0 \leq \theta_E, \theta_L \leq 1$ allow for a balance between minimizing energy consumption and controlling the straggler's latency. This trade-off is non-trivial, as achieving the lowest energy typically requires transmitting at very low rates, which increases latency while minimizing latency necessitates higher energy expenditure [145]. To solve this, we consider $\ell_k^{\max}$ as defined in (5.76), ensuring that the latency of each user $j \in [M]$ satisfies $\ell_j \leq \ell_k^{\max}$. Then, we re-write (5.79) as

$$(\mathcal{P}2) \quad \underset{p_k^1,\ldots,p_k^M,\ell_k^{\max}}{\text{minimize}} \quad \theta_l \ell_k^{\max} + \theta_E \sum_{j=1}^{M} E_k^j \tag{5.80a}$$

$$\text{subject to} \quad 0 \leq p_k^j \leq 1, \quad j \in [M], \tag{5.80b}$$

$$\ell_k^j \leq \ell_k^{\max}, \quad j \in [M]. \tag{5.80c}$$

Optimization problem (5.80) is nonlinear due to the nonlinear relationships between the decision variables p_k^j and the functions ℓ_k^j and E_k^j. Additionally, the constraint $\ell_k^j \leq \ell_k^{\max}$ introduces further nonlinearity by involving the maximum of a set of nonlinear functions. As a result, both the objective function and the feasible region become nonlinear, classifying the problem as a nonlinear optimization. To solve this, we apply the SQP approach [78], as detailed in Chapter 4.4 of this thesis.



(a) Test accuracy, IID

(b) Test accuracy, non-IID

(c) Local iterations, IID

(d) Local iterations, non-IID

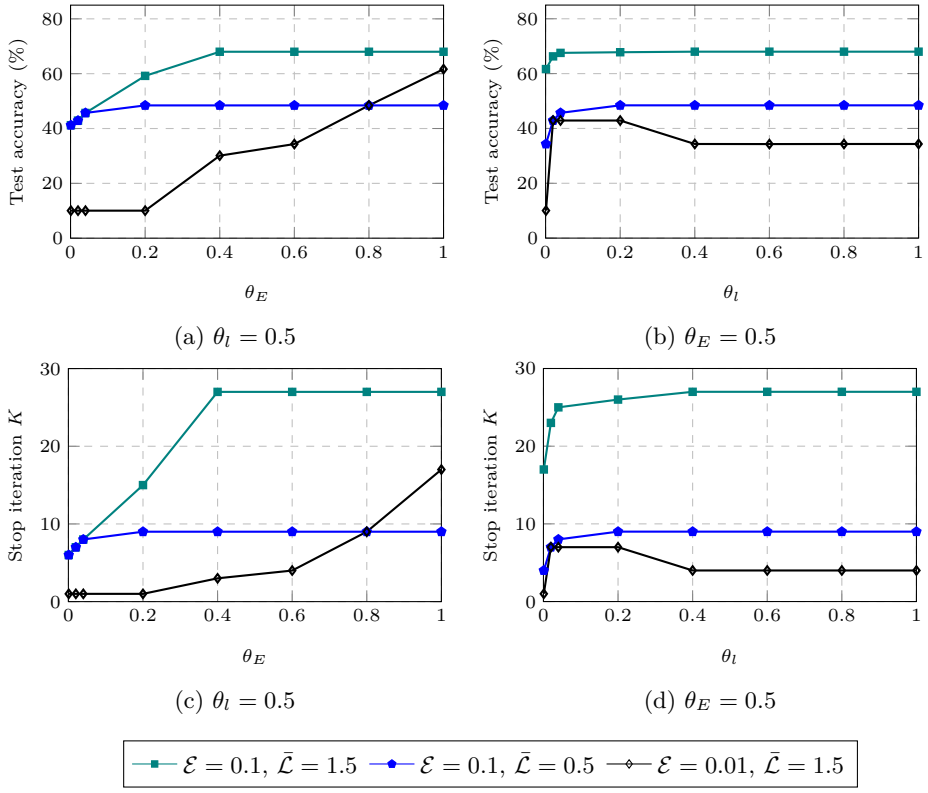
Figure 5.11: Convergence analysis of FedAvg with the EMQ scheme and adaptive local iterations compared to FedAvg with full precision,$M = 20$, $L = 2, 6$.

Illustrative Numerical Results

We consider the CIFAR-10 dataset to be distributed among the users in either an IID or non-IID manner, and each user trains the CNN [141] explained in Chapter 5.4.1 of this thesis on its local dataset. Moreover, we consider the latency budget $\bar{\mathcal{L}}$ (in second) and energy budget $\mathcal{E}$ (in Joule) in the following numerical setups.

Figure 5.11 compares the convergence of FedAvg with AdaDelta local updates and the EMQ scheme against FedAvg with full precision for $M = 20$ users under both IID and non-IID data distributions. We evaluate FedAvg with $L = 2$ and $L = 6$ local iterations, representing the minimum and maximum of the adaptive EMQ local iterations l_k^j, for $k \in [100]$.

Figures 5.11a and 5.11b illustrate the test accuracies, while Figures 5.11c and 5.11d display the local iterations at $k = 100$ for IID and non-IID cases, respectively. Notably, in FedAvg+EMQ, despite more than 55% of users having $l_k^j = 2$, the test accuracy surpasses FedAvg with $L = 2$. This demonstrates that the EMQ quantization error minimally affects training and can be offset by adaptive local iterations for a subset of users, achieving better accuracy than full precision FedAvg with $L = 2$ while significantly reducing communication overhead.



(a) $\theta_l = 0.5$ (b) $\theta_E = 0.5$

(c) $\theta_l = 0.5$ (d) $\theta_E = 0.5$

Figure 5.12: Performance analysis of FedAvg + EMQ with power allocation scheme vs. θ_E and θ_l, non-IID data distribution for $M = 20$ users.

Furthermore, FedAvg+EMQ achieves comparable test accuracy to full precision FedAvg with $L = 6$ by iteration $k = 40$ in the IID case and $k = 50$ in the non-IID case. This reduces computation resource usage by at least 49% (IID) and 56% (non-IID), highlighting the communication efficiency of the EMQ scheme.

We analyze the impact of scalarization weights θ_E and θ_l on power allocation performance under varying energy and latency budgets within the FedAvg+EMQ framework. Figure 5.12 illustrates the test accuracy and stop iteration K for non-IID data distribution across $M = 20$ users.

In Figure 5.12a, the test accuracy is plotted against θ_E for $\theta_l = 0.5$, while the corresponding stop iteration K is shown in Figure 5.12c. Results indicate that the optimal θ_E depends on the energy budget $\mathcal{E}$ and latency budget $\bar{\mathcal{L}}$. With $\theta_l = 0.5$, both test accuracy and K increase with higher θ_E, suggesting that prioritizing energy efficiency yields better results in terms of iterations and accuracy.

Figures 5.12b and 5.12d illustrate the relationship between test accuracy, stop

Table 5.8: Comparison analysis of EMQ and proposed power allocation with the benchmarks, non-IID $M = 20$, $\mathcal{E} = 1$, $\theta_E = 0.5$, $\theta_l = 1$.

Quantization scheme		**LAQ**				**AQUILA**				**EMQ**			
Latency budget $\bar{\mathcal{L}}$		1	2	2.5	3	1	2	2.5	3	1	2	2.5	3
Proposed method	**Accuracy (%)**	43.8	56.4	60.4	63.3	46.14	59.65	63.48	66.46	62.71	71.82	73.43	75.28
	K	13	26	32	39	14	27	34	41	18	36	44	52
Dinkelbach	**Accuracy (%)**	41.6	54	56.67	63.1	41.92	58.55	62.44	65.3	59.2	69.02	71.72	73.6
	K	12	24	30	38	12	25	32	39	15	29	37	43
Max-sum rate	**Accuracy (%)**	22.63	34.65	39.76	43.8	25.64	37.14	41.92	47.24	42.89	58.15	61.65	64.58
	K	4	9	11	13	5	10	12	15	7	14	17	20

Table 5.9: Comparison analysis of EMQ and proposed power allocation with the benchmarks, non-IID $M = 20$, $\bar{\mathcal{L}} = 4$, $\theta_E = 0.5$, $\theta_l = 1$.

Quantization scheme		**LAQ**				**AQUILA**				**EMQ**			
Energy budget $\mathcal{E}$		0.05	0.1	0.2	0.25	0.05	0.1	0.2	0.25	0.05	0.1	0.2	0.25
Proposed method	**Accuracy (%)**	34.65	49.7	62.6	65.58	48.62	62.44	67.09	67.09	66.89	75.1	76.1	76.1
	K	9	18	37	46	16	32	44	44	24	49	69	69
Dinkelbach	**Accuracy (%)**	10.8	22.63	34.65	39.76	21.26	27.2	37.14	41.92	30.1	42.89	58.15	61.65
	K	2	4	9	11	3	6	10	12	3	7	14	17
Max-sum rate	**Accuracy (%)**	10.8	24.9	39.76	43.8	24.33	27.2	41.92	47.24	34.33	45.68	61.65	65.23
	K	2	5	11	13	4	6	12	15	4	8	17	21

iteration K, and θ_l for $\theta_E = 0.5$. With a low energy budget ($\mathcal{E} = 0.01$), increasing θ_l leads to a decline in both K and test accuracy until they level off. This behavior arises because lower θ_l emphasizes energy minimization, enabling higher K and improved accuracy. Conversely, as θ_l grows, greater priority is given to latency minimization, resulting in a less energy-efficient power allocation. This shift reduces K and test accuracy under limited energy conditions.

Finally, we evaluate the performance of the FedAvg+EMQ framework in terms of test accuracy and stop iteration K, emphasizing the impact of our proposed power allocation strategy. Additionally, we compare the results with two benchmark power allocation methods: the max-min energy efficiency approach by Dinkelbach [48] and the max-sum rate method [2]. For quantization, we consider two alternative schemes: LAQ [44], which uses a fixed number of bits, and AQUILA [49], an adaptive bit allocation method.

Table 5.8 presents results for FedAvg with AdaDelta local updates under non-IID data distribution, with $M = 20$ users, energy budget $\mathcal{E} = 1$, $\theta_E = 0.5$, $\theta_l = 1$, and varying latency budgets. To ensure a fair comparison, the average number of

bits $\bar{b} := \mathbb{E}_{k,j} \boldsymbol{b}_k^j$ is used for LAQ, and the local iterations for LAQ and AQUILA are matched to the adaptive l_k^j in EMQ. Table 5.8 shows that across all latency budgets, our proposed power allocation method significantly outperforms the Dinkelbach and max-sum rate benchmarks, increasing test accuracy by up to 7% and 19%, respectively, due to the higher achievable K. Similarly, EMQ consistently surpasses AQUILA and LAQ in test accuracy, with gains of up to 19% and 24%, respectively, for each power allocation strategy.

Table 5.9 extends the analysis by comparing test accuracy and K under varying energy budgets ($\mathcal{E}$) with a fixed latency budget $\bar{\mathcal{L}} = 4$. The results demonstrate that our power allocation approach increases test accuracy by up to 36% and 35% compared to Dinkelbach and max-sum rate methods, respectively. Moreover, EMQ outperforms AQUILA and LAQ in test accuracy across all power allocation strategies, with improvements of up to 21% and 32%, respectively.

Overall, the combination of FedAvg+EMQ with our proposed power allocation strategy achieves the highest test accuracy across all evaluated energy and latency budgets. This result demonstrates the superiority of FedAvg+EMQ with our proposed power allocation over FedAvg with LAQ or AQUILA when paired with either Dinkelbach or max-sum rate methods.

5.5 Summary

In this chapter, we presented a comprehensive review of the literature on FL over wireless networks, focusing on communication-efficient approaches and recent advancements in adaptive quantization and sparsification methods. We then outlined our contributions to addressing research gaps in FL over wireless networks. The chapter included the system model, main problem formulation, and illustrative numerical results for each contribution. We compared our proposed methods with various benchmarks from the literature, demonstrating how our approaches outperform state-of-the-art power control and quantization techniques in reducing energy consumption and latency, particularly in large-scale FL scenarios with limited resources.

Chapter 6

Conclusions and Future Works

Explosive growth of data volumes produced by many IoT devices imposes various problems on traditional communication and computation approaches, such as cloud computing. IoT devices with a vast volume of data in a massive IoT network require extensive communication and storage resources to share their data with the servers in the cloud. Moreover, preserving data privacy, especially in critical applications, such as healthcare systems, is one of the crucial challenges in sharing sensitive data with the cloud. Recently, edge computing has gained the attention of many researchers because it brings computation and data storage to the edge near the devices. Therein, ML plays a prominent role in analyzing the data produced by the many IoT devices. Many researchers are working on ML, specifically distributed ML, due to its extensive benefits to large-scale IoT networks.

Within distributed learning, FL methods are a promising ML paradigm that preserves data privacy at edge devices and reduces communication overhead by eliminating data sharing between devices and servers. Accordingly, the devices transmit a learning parameter vector to the server utilizing wireless communication protocols, which we call FL over wireless networks. However, more efficient approaches to FL over wireless networks still need to be developed due to device communication and computation resource limitations, like bandwidth or energy.

Throughout this thesis, a systematic exploration of the key topics has been conducted. Chapter 1 served as the foundation, introducing edge computing, IoT, and ML over wireless networks while outlining the research gaps, research objectives, and contributions. Chapter 2 provided a detailed study of ML, including its core components such as neural networks, distributed ML, and FL. Subsequently, Chapter 3 explained the intricacies of CFmMIMO networks, including a general overview, uplink processes for transmitting FL models, descriptions of the power control schemes used as benchmarks, and a discussion of potential downlink scenarios for FL training. The optimization techniques central to this thesis, including the Bisection, Brent's method, LP, and SQP have been detailed in Chapter 4.

A comprehensive literature review and the proposed contributions, encompassing novel quantization techniques, power control strategies, and energy-efficient methods, have been presented in Chapter 5. The findings and future directions of this work have been summarized in this chapter, while Chapter 7 includes the included papers, offering an in-depth account of the research outcomes. The following section presents the concluding remarks of this thesis.

6.1 Concluding Remarks

The central research objective of this thesis was to optimally determine the number of global iterations required for distributed training, such as FL, without relying on future training information. This objective has accounted for the impact of communication protocols and training costs, including communication latency, computation latency, and energy consumption. To achieve this goal, the thesis has posed key research questions and systematically addressed them. Papers 1, 2, and 3 have specifically focused on determining the optimal stopping iterations, while Papers 4, 5, and 6 have explored power allocation strategies in the context of enhancing energy efficiency and reducing training latency by mitigating the straggler effect. Together, these contributions have provided a comprehensive framework for establishing optimal causal stopping criteria for distributed learning and FL training. More specifically, the following RQs have been considered in this thesis:

- **RQ1**: **What are the impacts of communication protocols and background traffic on the causal termination iteration of a distributed training?**

This thesis has addressed this RQ by analyzing the impact of communication protocol parameters on the causal stopping iteration of FL training, focusing on communication costs. Papers 1 and 2 provided theoretical and experimental insights into how parameters like packet arrival and transmission probabilities affect latency in protocols such as Slotted ALOHA, CSMA/CA, and OFDMA emphasizing the role of queue lengths and background traffic on the training performance from the iteration-cost perspective. By examining key scenarios and proposing a congestion control algorithm, this work has contributed to reducing communication delays and enhancing the efficiency of FL training.

Thereafter, we focused on developing energy-efficient and low-latency FL training over CFmMIMO networks. Accordingly, we addressed the following RQ:

- **RQ2**: **How can we adapt and optimize the physical layer of a CFmMIMO network to mitigate the straggler effect and enhance communication energy efficiency in FL training?**

This thesis has thoroughly examined the adaptation of the physical layer in CFmMIMO networks to address energy efficiency and mitigate the high latency caused

by straggler users in FL training. In large-scale FL systems, limited power resources often lead to significant communication delays, referred to as the straggler effect, as the server must wait for all users to complete their transmissions before updating the global model. These delays can exceed the allocated time budget, disrupting the training process. Additionally, the constrained energy resources of users add another layer of complexity. To tackle these challenges, Papers 4, 5, and 6 have focused on uplink power control strategies, highlighting how physical layer optimizations can effectively reduce communication latency and energy consumption. These efforts have demonstrated significant improvements in the overall efficiency and practicality of FL training in CFmMIMO networks.

Next, focusing on reducing the communication overhead of FL training, we addressed the following RQ, investigating the impact of the novel quantization schemes on FL over wireless networks.

- **RQ3**: **How can the quantization scheme be designed to enhance communication efficiency and model accuracy in FL?**

This thesis has explored the influence of advanced quantization techniques on communication efficiency and model accuracy in FL. While fixed bit-assignment methods like LAQ provided a baseline, adaptive quantization schemes have shown significant promise in reducing communication overhead without sacrificing test accuracy, compared to standard FL with the full-precision transmission. Papers 3, 5, and 6 examined various quantization strategies, including A-LAQ and other adaptive approaches, to evaluate their effectiveness in minimizing communication costs while preserving model accuracy. These studies underscore the pivotal role of adaptive quantization in improving the communication efficiency of FL systems.

Lastly, we have investigated the computation costs, such as latency or energy, associated with the users and addressed their impact on the training efficiency.

- **RQ4**: **What are the impacts of computation costs on the overall performance of FL systems**?

In addition to focusing on communication efficiency, this thesis has examined the computation costs associated with FL training. These costs, which include latency and energy consumption, are influenced by factors such as the size of local datasets, the number of local iterations, device processing capabilities, and the number of processing cycles required. Papers 2, 4, and 5 have provided a detailed analysis of how computation costs influence training efficiency, model convergence, and resource allocation. The findings highlight the critical role of managing computation costs in enhancing the practical feasibility and performance of FL systems.

6.2 Future Research Directions

Future research in the area of communication- and computation-efficient FL over wireless networks offers several promising avenues. In terms of communication efficiency, the exploration of adaptive quantization schemes has demonstrated significant improvements in resource utilization while maintaining convergence performance comparable to full-precision FL. Building upon this foundation, future work could focus on the integrated design of user/client selection mechanisms alongside adaptive communication-efficient schemes. Furthermore, rigorous mathematical analysis should be conducted to evaluate the convergence properties of such a joint design, employing comprehensive theoretical and experimental methodologies.

Future work on FL over CFmMIMO could explore the integration of downlink resource allocation with uplink optimization to enhance overall resource efficiency. While uplink resource allocation has been extensively studied in this thesis for reducing latency and improving energy efficiency, downlink resource management remains critical, particularly in scenarios where global model updates must be distributed to a large number of users, such as large-scale IoT networks. Joint uplink and downlink optimization could involve the simultaneous allocation of transmit power, bandwidth, and beamforming strategies to minimize communication latency and energy consumption. Moreover, addressing challenges such as user heterogeneity, varying channel conditions, and fairness in resource distribution will be pivotal. Advanced techniques, such as reinforcement learning or model-based optimization, could be employed to dynamically adapt resource allocation based on real-time network conditions. By jointly optimizing uplink and downlink resources, future research can ensure a more resource-efficient FL system over CFmMIMO, further advancing its applicability in real-world scenarios.

This thesis has explored the use of adaptive quantization schemes in conjunction with dynamically adjusted local iterations to reduce uplink communication overhead in FL training. However, the issue of computation overhead remains an open and critical area for further investigation. A promising direction for future research lies in the joint optimization of both communication and computation overheads. Specifically, the incorporation of dynamic local computation strategies, such as varying the number of local iterations based on system conditions, could provide a comprehensive approach to enhance overall resource efficiency. This integrated framework has the potential to address the trade-offs between communication and computation costs, paving the way for more communication- and computation-efficient FL training over wireless networks, such as CFmMIMO networks.

Finally, a potential future research direction involves applying the proposed causal approach to facilitate collaboration among APs in CFmMIMO systems for gathering environmental information. By leveraging this method, APs could dynamically and collaboratively coordinate their actions while simultaneously collecting FL parameters from devices. This approach has the potential to improve communication efficiency and optimize the overall operation of CFmMIMO networks, offering a pathway to more intelligent and resource-efficient systems.

Bibliography

[1] A. Zaidi, Y. Hussain, M. Hogan, and C. Kuhlins, "Cellular IoT evolution for industry digitalization," *Ericsson White paper*, 2019.

[2] O. T. Demir, E. Björnson, and L. Sanguinetti, "Foundations of user-centric cell-free Massive MIMO," *Foundations and Trends in Signal Processing*, vol. 14, no. 3-4, pp. 162–472, 2021.

[3] J. Sun, T. Chen, G. B. Giannakis, Q. Yang, and Z. Yang, "Lazily Aggregated Quantized Gradient innovation for communication-efficient Federated Learning," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 44, no. 4, pp. 2031–2044, 2022.

[4] A. Mahmoudi, H. S. Ghadikolaei, and C. Fischione, "Cost-efficient distributed optimization in machine learning over wireless networks," in *IEEE International Conference on Communications (ICC)*, 2020.

[5] W. Yu, F. Liang, X. He, W. G. Hatcher, C. Lu, J. Lin, and X. Yang, "A survey on the edge computing for the Internet of Things," *IEEE Access*, vol. 6, pp. 6900–6919, 2017.

[6] G. Premsankar, M. Di Francesco, and T. Taleb, "Edge computing for the Internet of Things: A case study," *IEEE Internet of Things Journal*, vol. 5, no. 2, pp. 1275–1284, 2018.

[7] W. Y. B. Lim, N. C. Luong, D. T. Hoang, Y. Jiao, Y.-C. Liang, Q. Yang, D. Niyato, and C. Miao, "Federated Learning in mobile edge networks: A comprehensive survey," *IEEE Communications Surveys & Tutorials*, vol. 22, no. 3, pp. 2031–2063, 2020.

[8] J. Bian, A. A. Arafat, H. Xiong, J. Li, L. Li, H. Chen, J. Wang, D. Dou, and Z. Guo, "Machine learning in real-time Internet of Things (iot) systems: A survey," *IEEE Internet of Things Journal*, vol. 9, no. 11, pp. 8364–8386, 2022.

[9] J. Li, Y. Meng, L. Ma, S. Du, H. Zhu, Q. Pei, and X. Shen, "A Federated Learning based privacy-preserving smart healthcare system," *IEEE Transactions on Industrial Informatics*, vol. 18, no. 3, pp. 2021–2031, 2022.

[10] L. U. Khan, I. Yaqoob, N. H. Tran, S. M. A. Kazmi, T. N. Dang, and C. S. Hong, "Edge-computing-enabled smart cities: A comprehensive survey," *IEEE Internet of Things Journal*, vol. 7, no. 10, pp. 10 200–10 232, 2020.

[11] Y. Wu, H.-N. Dai, H. Wang, Z. Xiong, and S. Guo, "A survey of intelligent network slicing management for industrial iot: Integrated approaches for smart transportation, smart energy, and smart factory," *IEEE Communications Surveys & Tutorials*, vol. 24, no. 2, pp. 1175–1211, 2022.

[12] S. Hu, X. Chen, W. Ni, E. Hossain, and X. Wang, "Distributed machine learning for wireless communication networks: Techniques, architectures, and applications," *IEEE Communications Surveys & Tutorials*, vol. 23, no. 3, pp. 1458–1493, 2021.

[13] J. Konečnỳ, H. B. McMahan, F. X. Yu, P. Richtárik, A. T. Suresh, and D. Bacon, "Federated Learning: Strategies for improving communication efficiency," *arXiv preprint arXiv:1610.05492*, 2016.

[14] D. C. Nguyen, M. Ding, P. N. Pathirana, A. Seneviratne, J. Li, and H. Vincent Poor, "Federated Learning for internet of things: A comprehensive survey," *IEEE Communications Surveys & Tutorials*, vol. 23, no. 3, pp. 1622–1658, 2021.

[15] A. Imteaj, U. Thakker, S. Wang, J. Li, and M. H. Amini, "A survey on Federated Learning for resource-constrained iot devices," *IEEE Internet of Things Journal*, vol. 9, no. 1, pp. 1–24, 2022.

[16] H. Hellström, J. M. B. da Silva Jr, M. M. Amiri, M. Chen, V. Fodor, H. V. Poor, C. Fischione *et al.*, "Wireless for Machine Learning: A Survey," *Foundations and Trends® in Signal Processing*, vol. 15, no. 4, pp. 290–399, 2022.

[17] T. Li, A. K. Sahu, A. Talwalkar, and V. Smith, "Federated Learning: Challenges, methods, and future directions," *IEEE Signal Processing Magazine*, vol. 37, no. 3, pp. 50–60, 2020.

[18] Y. Wang, Y. Xu, Q. Shi, and T.-H. Chang, "Quantized Federated Learning under transmission delay and outage constraints," *IEEE Journal on Selected Areas in Communications*, vol. 40, no. 1, pp. 323–341, 2022.

[19] Z. Yang, M. Chen, W. Saad, C. S. Hong, and M. Shikh-Bahaei, "Energy efficient Federated Learning over wireless communication networks," *IEEE Transactions on Wireless Communications*, pp. 1–1, 2020.

[20] Q. Zeng, Y. Du, K. Huang, and K. K. Leung, "Energy-efficient resource management for federated edge learning with cpu-gpu heterogeneous computing," *IEEE Transactions on Wireless Communications*, vol. 20, no. 12, pp. 7947–7962, 2021.

[21] F. Sattler, S. Wiedemann, K.-R. Muller, and W. Samek, "Robust and communication-efficient Federated Learning from non-i.i.d. data," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 31, no. 9, pp. 3400–3413, 2020.

[22] X. Mo and J. Xu, "Energy-efficient federated edge learning with joint communication and computation design," *Journal of Communications and Information Networks*, vol. 6, no. 2, pp. 110–124, 2021.

[23] Y. Mu, N. Garg, and T. Ratnarajah, "Federated Learning in massive MIMO 6g networks: Convergence analysis and communication-efficient design," *IEEE Transactions on Network Science and Engineering*, vol. 9, no. 6, pp. 4220–4234, 2022.

[24] J. Xu, H. Wang, and L. Chen, "Bandwidth allocation for multiple Federated Learning services in wireless edge networks," *IEEE Transactions on Wireless Communications*, vol. 21, no. 4, pp. 2534–2546, 2022.

[25] J. Xu and H. Wang, "Client selection and bandwidth allocation in wireless Federated Learning networks: A long-term perspective," *IEEE Transactions on Wireless Communications*, vol. 20, no. 2, pp. 1188–1200, 2021.

[26] W. Shi, S. Zhou, Z. Niu, M. Jiang, and L. Geng, "Joint device scheduling and resource allocation for latency constrained wireless Federated Learning," *IEEE Transactions on Wireless Communications*, vol. 20, no. 1, pp. 453–467, 2021.

[27] M. M. Wadu, S. Samarakoon, and M. Bennis, "Joint client scheduling and resource allocation under channel uncertainty in Federated Learning," *IEEE Transactions on Communications*, vol. 69, no. 9, pp. 5962–5974, 2021.

[28] H. Chen, S. Huang, D. Zhang, M. Xiao, M. Skoglund, and H. V. Poor, "Federated Learning over wireless iot networks with optimized communication and resources," *IEEE Internet of Things Journal*, vol. 9, no. 17, pp. 16 592–16 605, 2022.

[29] T. T. Vu, H. Q. Ngo, M. N. Dao, D. T. Ngo, E. G. Larsson, and T. Le-Ngoc, "Energy-efficient massive MIMO for Federated Learning: Transmission designs and resource allocations," *IEEE Open Journal of the Communications Society*, vol. 3, pp. 2329–2346, 2022.

[30] G. Zhu, D. Liu, Y. Du, C. You, J. Zhang, and K. Huang, "Towards an intelligent edge: Wireless communication meets machine learning," *arXiv preprint arXiv:1809.00343*, 2018.

[31] J. Park, S. Samarakoon, M. Bennis, and M. Debbah, "Wireless network intelligence at the edge," *Proceedings of the IEEE*, vol. 107, no. 11, pp. 2204–2239, 2019.

[32] T. Chen, S. Barbarossa, X. Wang, G. B. Giannakis, and Z.-L. Zhang, "Learning and management for internet-of-things: Accounting for adaptivity and scalability," *Proceedings of the IEEE*, 2019.

[33] J. Chen and X. Ran, "Deep learning with edge computing: A review," *Proceedings of the IEEE*, vol. 107, no. 8, pp. 1655–1674, 2019.

[34] X. Wang, Y. Han, V. C. M. Leung, D. Niyato, X. Yan, and X. Chen, "Convergence of edge computing and deep learning: A comprehensive survey," *IEEE Communications Surveys Tutorials*, vol. 22, no. 2, pp. 869–904, 2020.

[35] P. Kairouz, H. B. McMahan, B. Avent, A. Bellet, M. Bennis, A. N. Bhagoji, K. Bonawitz, Z. Charles, G. Cormode, R. Cummings *et al.*, "Advances and open problems in Federated Learning," *arXiv preprint arXiv:1912.04977*, 2019.

[36] M. Chen, H. V. Poor, W. Saad, and S. Cui, "Wireless communications for collaborative Federated Learning," *IEEE Communications Magazine*, vol. 58, no. 12, pp. 48–54, 2020.

[37] F. Hussain, S. A. Hassan, R. Hussain, and E. Hossain, "Machine learning for resource management in cellular and IoT networks: Potentials, current solutions, and open challenges," *IEEE communications surveys & tutorials*, vol. 22, no. 2, pp. 1251–1275, 2020.

[38] S. Boyd and L. Vandenberghe, *Convex Optimization*. USA: Cambridge University Press, 2004.

[39] X. Yang, "Smoothing approximations to nonsmooth optimization problems," *The ANZIAM Journal*, vol. 36, no. 3, pp. 274–285, 1995.

[40] A. Ben-Tal, M. Teboulle, and W. H. Yang, "A least-squares-based method for a class of nonsmooth minimization problems with applications in plasticity," *Applied Mathematics and Optimization*, vol. 24, no. 1, pp. 273–288, 1991.

[41] A. E. Ingham and A. E. Ingham, *The distribution of prime numbers*. Cambridge University Press, 1990, no. 30.

[42] M. Schatzman and M. Schatzman, *Numerical analysis: a mathematical introduction*. Oxford University Press on Demand, 2002.

[43] Y. Mao, Z. Zhao, M. Yang, L. Liang, Y. Liu, W. Ding, T. Lan, and X.-P. Zhang, "SAFARI: Sparsity-enabled Federated Learning with limited and unreliable communications," *IEEE Transactions on Mobile Computing*, vol. 23, no. 5, pp. 4819–4831, 2024.

[44] J. Sun, T. Chen, G. B. Giannakis, Q. Yang, and Z. Yang, "Lazily Aggregated Quantized Gradient innovation for communication-efficient federated learning," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 44, no. 4, pp. 2031–2044, 2022.

[45] D. P. Bertsekas, R. G. Gallager, and P. Humblet, *Data networks, second edition*. Prentice-Hall International New Jersey, 2004, vol. 2.

[46] S. Luo, X. Chen, Q. Wu, Z. Zhou, and S. Yu, "HFEL: Joint edge association and resource allocation for cost-efficient Hierarchical Federated Edge Learning," *IEEE Transactions on Wireless Communications*, vol. 19, no. 10, pp. 6535–6548, 2020.

[47] N. C. Thompson, K. Greenewald, K. Lee, and G. F. Manso, "Deep learning's diminishing returns: The cost of improvement is becoming unsustainable," *IEEE Spectrum*, vol. 58, no. 10, pp. 50–55, 2021.

[48] A. Zappone, L. Sanguinetti, G. Bacci, E. Jorswieck, and M. Debbah, "Energy-efficient power control: A look at 5G wireless technologies," *IEEE Transactions on Signal Processing*, vol. 64, pp. 1668–1683, 2016.

[49] Z. Zhao, Y. Mao, Z. Shi, Y. Liu, T. Lan, W. Ding, and X.-P. Zhang, "AQUILA: Communication efficient Federated Learning with adaptive quantization in device selection strategy," *IEEE Transactions on Mobile Computing.*, vol. 23, no. 6, pp. 7363–7376, 2024.

[50] S. U. Stich, J.-B. Cordonnier, and M. Jaggi, "Sparsified SGD with memory," in *Advances in Neural Information Processing Systems*, 2018, pp. 4447–4458.

[51] H. Song, M. Kim, D. Park, Y. Shin, and J.-G. Lee, "Learning from noisy labels with deep neural networks: A survey," *IEEE Transactions on Neural Networks and Learning Systems*, pp. 1–19, 2022.

[52] J. Verbraeken, M. Wolting, J. Katzy, J. Kloppenburg, T. Verbelen, and J. S. Rellermeyer, "A survey on distributed machine learning," *Acm computing surveys (csur)*, vol. 53, no. 2, pp. 1–33, 2020.

[53] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *nature*, vol. 521, no. 7553, pp. 436–444, 2015.

[54] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," *Advances in neural information processing systems*, vol. 25, 2012.

[55] S. Ruder, "An overview of gradient descent optimization algorithms," *arXiv preprint arXiv:1609.04747*, 2016.

[56] R. Tibshirani, *Notes on Stochastic Gradient Descent, Convex Optimization 10-725*. USA: CMU university, 2010.

[57] S. J. Wright, "Coordinate descent algorithms," *Mathematical programming*, vol. 151, no. 1, pp. 3–34, 2015.

[58] M. D. Zeiler, "Adadelta: An adaptive learning rate method," *arXiv preprint arXiv:1212.5701*, 2012.

[59] J. Duchi, E. Hazan, and Y. Singer, "Adaptive subgradient methods for online learning and stochastic optimization." *Journal of machine learning research*, vol. 12, no. 7, 2011.

[60] B. McMahan, E. Moore, D. Ramage, S. Hampson, and B. A. y Arcas, "Communication-efficient learning of deep networks from decentralized data," in *Artificial intelligence and statistics*. PMLR, 2017, pp. 1273–1282.

[61] X. Li, K. Huang, W. Yang, S. Wang, and Z. Zhang, "On the convergence of FedAvg on non-iid data," *arXiv preprint arXiv:1907.02189*, 2019.

[62] Z. Taghiyarrenani, A. Alabdallah, S. Nowaczyk, and S. Pashami, "Heterogeneous Federated Learning via personalized generative networks," *arXiv preprint arXiv:2308.13265*, 2023.

[63] S. Reddi, Z. Charles, M. Zaheer, Z. Garrett, K. Rush, J. Konečnỳ, S. Kumar, and H. B. McMahan, "Adaptive federated optimization," *arXiv preprint arXiv:2003.00295*, 2020.

[64] T. Sun, D. Li, and B. Wang, "Decentralized federated averaging," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 45, no. 4, pp. 4289–4301, 2022.

[65] S. Buzzi and C. D'Andrea, "Cell-free massive MIMO: User-centric approach," *IEEE Wireless Communications Letters*, vol. 6, no. 6, pp. 706–709, 2017.

[66] F. Riera-Palou, G. Femenias, A. G. Armada, and A. Pérez-Neira, "Clustered cell-free massive MIMO," in *2018 IEEE Globecom Workshops (GC Wkshps)*. IEEE, 2018, pp. 1–6.

[67] S. Chen, F. Qin, B. Hu, X. Li, and Z. Chen, "User-centric ultra-dense networks for 5G: Challenges, methodologies, and directions," *IEEE Wireless Communications*, vol. 23, no. 2, pp. 78–85, 2016.

[68] H. Q. Ngo, A. Ashikhmin, H. Yang, E. G. Larsson, and T. L. Marzetta, "Cell-free Massive MIMO versus small cells," *IEEE Transactions on Wireless Communications*, vol. 16, no. 3, pp. 1834–1850, 2017.

[69] A. de la Fuente, R. P. Leal, and A. G. Armada, "New technologies and trends for next generation mobile broadcasting services," *IEEE Communications Magazine*, vol. 54, no. 11, pp. 217–223, 2016.

[70] A. De La Fuente, G. Femenias, F. Riera-Palou, and G. Interdonato, "Subgroup-centric multicast Cell-Free Massive MIMO," *IEEE Open Journal of the Communications Society*, vol. 5, pp. 6872–6889, 2024.

[71] J. Li, Q. Pan, Z. Wu, P. Zhu, D. Wang, and X. You, "Spectral efficiency of unicast and multigroup multicast transmission in cell-free distributed massive mimo systems," *IEEE Transactions on Vehicular Technology*, vol. 71, no. 12, pp. 12 826–12 839, 2022.

[72] A. Papazafeiropoulos, H. Q. Ngo, P. Kourtessis, S. Chatzinotas, and J. M. Senior, "Towards optimal energy efficiency in cell-free massive MIMO systems," *IEEE Transactions on Green Communications and Networking*, vol. 5, no. 2, pp. 816–831, 2021.

[73] G. R. Wood, "The bisection method in higher dimensions," *Mathematical programming*, vol. 55, pp. 319–337, 1992.

[74] S. B. Russ, "A translation of Bolzano's paper on the intermediate value theorem," *Historia Mathematica*, vol. 7, no. 2, pp. 156–185, 1980.

[75] R. P. Brent, *Algorithms for minimization without derivatives*. Courier Corporation, 2013.

[76] J. A. Nelder and R. Mead, "A simplex method for function minimization," *The computer journal*, vol. 7, no. 4, pp. 308–313, 1965.

[77] C. Roos, T. Terlaky, and J.-P. Vial, "Interior point methods for linear optimization," 2005.

[78] J. Nocedal and S. J. Wright, *Numerical optimization*. Springer, 1999.

[79] M. Chen, D. Gündüz, K. Huang, W. Saad, M. Bennis, A. V. Feljan, and H. V. Poor, "Distributed learning in wireless networks: Recent progress and future challenges," *IEEE Journal on Selected Areas in Communications*, vol. 39, no. 12, pp. 3579–3605, 2021.

[80] Y. Sarcheshmehpour, M. Leinonen, and A. Jung, "Federated Learning from big data over networks," in *ICASSP 2021 - 2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, 2021, pp. 3055–3059.

[81] Z. Zhao, Y. Mao, Y. Liu, L. Song, Y. Ouyang, X. Chen, and W. Ding, "Towards efficient communications in Federated Learning: A contemporary survey," *arXiv preprint arXiv:2208.01200*, 2022.

[82] R. Jin, X. He, and H. Dai, "Communication efficient Federated Learning with energy awareness over wireless networks," *IEEE Transactions on Wireless Communications*, vol. 21, no. 7, pp. 5204–5219, 2022.

[83] K. Wei, J. Li, M. Ding, C. Ma, H. H. Yang, F. Farokhi, S. Jin, T. Q. S. Quek, and H. Vincent Poor, "Federated Learning with differential privacy: Algorithms and performance analysis," *IEEE Transactions on Information Forensics and Security*, vol. 15, pp. 3454–3469, 2020.

[84] T. Li and L. Song, "Privacy-preserving communication-efficient Federated multi-armed bandits," *IEEE Journal on Selected Areas in Communications*, vol. 40, no. 3, pp. 773–787, 2022.

[85] S. Liu, G. Yu, R. Yin, J. Yuan, L. Shen, and C. Liu, "Joint model pruning and device selection for communication-efficient federated edge learning," *IEEE Transactions on Communications*, vol. 70, no. 1, pp. 231–244, 2022.

[86] M. Zhang, G. Zhu, S. Wang, J. Jiang, Q. Liao, C. Zhong, and S. Cui, "Communication-efficient Federated edge learning via optimal probabilistic device scheduling," *IEEE Transactions on Wireless Communications*, vol. 21, no. 10, pp. 8536–8551, 2022.

[87] J. Shu, W. Zhang, Y. Zhou, Z. Cheng, and L. T. Yang, "Flas: Computation and communication efficient Federated Learning via adaptive sampling," *IEEE Transactions on Network Science and Engineering*, vol. 9, no. 4, pp. 2003–2014, 2022.

[88] Y. Deng, F. Lyu, J. Ren, H. Wu, Y. Zhou, Y. Zhang, and X. Shen, "Auction: Automated and quality-aware client selection framework for efficient Federated Learning," *IEEE Transactions on Parallel and Distributed Systems*, vol. 33, no. 8, pp. 1996–2009, 2022.

[89] W. Xu, B. Liang, G. Boudreau, and H. Sokun, "Clipper: Online joint client sampling and power allocation for wireless Federated Learning," *ACM Transactions on Modeling and Performance Evaluation of Computing Systems*, 2024.

[90] Z. Chen, W. Yi, H. Shin, A. Nallanathan, and G. Y. Li, "Efficient wireless Federated Learning with partial model aggregation," *IEEE Transactions on Communications*, vol. 72, no. 10, pp. 6271–6286, 2024.

[91] X. Hu, Z. Chen, C. Feng, G. Min, T. Q. S. Quek, and H. H. Yang, "Sparsified random partial model update for personalized Federated Learning," *IEEE Transactions on Mobile Computing*, pp. 1–17, 2024.

[92] M. M. Amiri, D. Gunduz, S. R. Kulkarni, and H. V. Poor, "Federated Learning with quantized global model updates," *arXiv preprint arXiv:2006.10672*, 2020.

[93] 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.

[94] S. Li, Q. Qi, J. Wang, H. Sun, Y. Li, and F. R. Yu, "GGS: General gradient sparsification for Federated Learning in edge computing," in *ICC 2020 - 2020 IEEE International Conference on Communications (ICC)*, 2020.

[95] K. Yuan, Q. Ling, and Z. Tian, "Communication-efficient decentralized event monitoring in wireless sensor networks," *IEEE Transactions on Parallel and Distributed Systems*, vol. 26, no. 8, pp. 2198–2207, 2014.

[96] J. Wangni, J. Wang, J. Liu, and T. Zhang, "Gradient sparsification for communication-efficient distributed optimization," *Advances in Neural Information Processing Systems*, vol. 31, 2018.

[97] A. Panda, S. Mahloujifar, A. Nitin Bhagoji, S. Chakraborty, and P. Mittal, "Sparsefed: Mitigating model poisoning attacks in Federated Learning with sparsification," in *Proceedings of The 25th International Conference on Artificial Intelligence and Statistics*. PMLR, 2022, pp. 7587–7624.

[98] P. Han, S. Wang, and K. K. Leung, "Adaptive gradient sparsification for efficient Federated Learning: An online learning approach," in *2020 IEEE 40th International Conference on Distributed Computing Systems (ICDCS)*, 2020, pp. 300–310.

[99] Q. Chen, H. Cheng, Y. Liang, G. Zhu, M. Li, and H. Jiang, "Tinyfel: Communication, computation, and memory efficient tiny Federated Edge Learning via model sparse update," *IEEE Internet of Things Journal*, pp. 1–1, 2024.

[100] F. Sattler, A. Marban, R. Rischke, and W. Samek, "CFD: Communication-efficient federated distillation via soft-label quantization and delta coding," *IEEE Transactions on Network Science and Engineering*, 2022.

[101] J. Bernstein, Y.-X. Wang, K. Azizzadenesheli, and A. Anandkumar, "signSGD: Compressed optimisation for non-convex problems," in *Proceedings of the 35th International Conference on Machine Learning*, 2018, pp. 560–569.

[102] X. Dai, X. Yan, K. Zhou, H. Yang, K. K. Ng, J. Cheng, and Y. Fan, "Hypersphere quantization: Communication-efficient SGD for Federated Learning," *arXiv preprint arXiv:1911.04655*, 2019.

[103] J. Xu, W. Du, Y. Jin, W. He, and R. Cheng, "Ternary compression for communication-efficient Federated Learning," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 33, no. 3, pp. 1162–1176, 2022.

[104] Y. Mao, Z. Zhao, G. Yan, Y. Liu, T. Lan, L. Song, and W. Ding, "Communication-efficient federated learning with adaptive quantization," *ACM Transactions on Intelligent Systems and Technology (TIST)*, vol. 13, no. 4, pp. 1–26, 2022.

[105] D. Jhunjhunwala, A. Gadhikar, G. Joshi, and Y. C. Eldar, "Adaptive quantization of model updates for communication-efficient federated learning," in *ICASSP 2021-2021 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2021, pp. 3110–3114.

[106] M. Zhang, Y. Li, D. Liu, R. Jin, G. Zhu, C. Zhong, and T. Q. Quek, "Joint compression and deadline optimization for wireless federated learning," *IEEE Transactions on Mobile Computing*, 2023.

[107] L. Qu, S. Song, and C.-Y. Tsui, "Feddq: Communication-efficient Federated Learning with descending quantization," in *GLOBECOM 2022 - 2022 IEEE Global Communications Conference*, 2022, pp. 281–286.

[108] R. Hönig, Y. Zhao, and R. Mullins, "Dadaquant: Doubly-adaptive quantization for communication-efficient federated learning," in *International Conference on Machine Learning*. PMLR, 2022, pp. 8852–8866.

[109] L. Qu, S. Song, and C.-Y. Tsui, "Fedaq: Communication-efficient federated edge learning via joint uplink and downlink adaptive quantization," *arXiv preprint arXiv:2406.18156*, 2024.

[110] H. Sun, X. Ma, and R. Q. Hu, "Adaptive Federated Learning with gradient compression in uplink NOMA," *IEEE Transactions on Vehicular Technology*, vol. 69, no. 12, pp. 16 325–16 329, 2020.

[111] W. Yang, Y. Yang, X. Dang, H. Jiang, Y. Zhang, and W. Xiang, "A novel adaptive gradient compression approach for communication-efficient Federated Learning," in *2021 China Automation Congress (CAC)*, 2021, pp. 674–678.

[112] T. T. Vu, D. T. Ngo, N. H. Tran, H. Q. Ngo, M. N. Dao, and R. H. Middleton, "Cell-free massive MIMO for wireless Federated Learning," *IEEE Transactions on Wireless Communications*, vol. 19, no. 10, pp. 6377–6392, 2020.

[113] S. Samarakoon, M. Bennis, W. Saad, and M. Debbah, "Distributed Federated Learning for ultra-reliable low-latency vehicular communications," *IEEE Transactions on Communications*, vol. 68, no. 2, pp. 1146–1159, 2020.

[114] M. Farooq, T. T. Vu, H. Q. Ngo, and L.-N. Tran, "Massive MIMO for serving Federated Learning and non-Federated Learning users," *IEEE Transactions on Wireless Communications*, vol. 23, no. 1, pp. 247–262, 2024.

[115] S. Liu, J. Yu, X. Deng, and S. Wan, "FedCPF: An efficient-communication Federated Learning approach for vehicular edge computing in 6g communication networks," *IEEE Transactions on Intelligent Transportation Systems*, vol. 23, no. 2, pp. 1616–1629, 2022.

[116] B. Wang, J. Fang, H. Li, and B. Zeng, "Communication-efficient Federated Learning: A variance-reduced stochastic approach with adaptive sparsification," *IEEE Transactions on Signal Processing*, 2023.

[117] J. Shu, W. Zhang, Y. Zhou, Z. Cheng, and L. T. Yang, "FLAS: Computation and communication efficient Federated Learning via adaptive sampling," *IEEE Transactions on Network Science and Engineering*, vol. 9, no. 4, pp. 2003–2014, 2022.

[118] Y. Liu, Y. Zhu, and J. J. Yu, "Resource-constrained Federated Edge Learning with heterogeneous data: Formulation and analysis," *IEEE Transactions on Network Science and Engineering*, vol. 9, no. 5, pp. 3166–3178, 2022.

[119] Z. Zhao, C. Feng, W. Hong, J. Jiang, C. Jia, T. Q. S. Quek, and M. Peng, "Federated Learning with non-IID data in wireless networks," *IEEE Transactions on Wireless Communications*, vol. 21, no. 3, pp. 1927–1942, 2022.

[120] S. M. Hamidi and E.-H. Yang, "AdaFed: Fair Federated Learning via adaptive common descent direction," *arXiv preprint arXiv:2401.04993*, 2024.

[121] S. Vaishnav, S. Khirirat, and S. Magnússon, "Communication-adaptive gradient sparsification for Federated Learning with error compensation," *IEEE Internet of Things Journal*, pp. 1–1, 2024.

[122] M. P. Uddin, Y. Xiang, B. Cai, X. Lu, J. Yearwood, and L. Gao, "ARFL: Adaptive and robust Federated Learning," *IEEE Transactions on Mobile Computing*, 2023.

[123] Y. Sun, L. Shen, H. Sun, L. Ding, and D. Tao, "Efficient Federated Learning via local adaptive amended optimizer with linear speedup," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2023.

[124] J. Zhang, J. Zhang, D. W. K. Ng, and B. Ai, "Federated Learning-based cell-free massive MIMO system for privacy-preserving," *IEEE Transactions on Wireless Communications*, 2022.

[125] T. T. Vu, D. T. Ngo, H. Q. Ngo, M. N. Dao, N. H. Tran, and R. H. Middleton, "Straggler effect mitigation for Federated Learning in cell-free Massive MIMO," in *2021 IEEE ICC*, pp. 1–6.

[126] T. T. Vu, H. Q. Ngo, T. L. Marzetta, and M. Matthaiou, "How does cell-free Massive MIMO support multiple Federated Learning groups?" in *2021 IEEE 22nd International Workshop on Signal Processing Advances in Wireless Communications (SPAWC)*, 2021, pp. 401–405.

[127] G. Wang, C. Zhao, Q. Qi, R. Han, L. Bai, and J. Choi, "Efficient Federated Learning via joint communication and computation optimization," *IEEE Transactions on Vehicular Technology*, 2024.

[128] H. Bao, K. Xiong, R. Zhang, P. Fan, D. Niyato, and K. B. Letaief, "User cooperation with power control for Federated Learning in cfmmimo networks," in *2023 IEEE International Conference on Communications Workshops (ICC Workshops)*. IEEE, 2023, pp. 122–127.

[129] M. Bashar, K. Cumanan, A. G. Burr, H. Q. Ngo, E. G. Larsson, and P. Xiao, "Energy efficiency of the cell-free Massive MIMO uplink with optimal uniform quantization," *IEEE Transactions on Green Communications and Networking*, vol. 3, no. 4, pp. 971–987, 2019.

[130] T. Zhao, X. Chen, Q. Sun, and J. Zhang, "Energy-efficient Federated Learning over cell-free IoT networks: Modeling and optimization," *IEEE Internet of Things Journal*, pp. 1–1, 2023.

[131] F. Qin, S. Xu, C. Li, Y. Xu, and L. Yang, "Theoretical analysis of Federated Learning supported by cell-free massive mimo networks with enhanced power allocation," *IEEE Wireless Communications Letters*, 2024.

[132] D. Wang, M. Tao, X. Zeng, and J. Liang, "Federated Learning for precoding design in cell-free massive MIMO systems," *IEEE Open Journal of the Communications Society*, 2023.

[133] M. M. Amiri and D. Gündüz, "Machine learning at the wireless edge: Distributed stochastic gradient descent Over-the-Air," *IEEE Transactions on Signal Processing*, vol. 68, pp. 2155–2169, 2020.

[134] Y. Oh, Y.-S. Jeon, M. Chen, and W. Saad, "Vector quantized compressed sensing for communication-efficient Federated Learning," in *2022 IEEE Globecom Workshops*, pp. 365–370.

[135] D. P. Bertsekas and J. N. Tsitsiklis, *Parallel and distributed computation: numerical methods*. Prentice hall Englewood Cliffs, NJ, 1989, vol. 23.

[136] A. Mahmoudi, H. S. Ghadikolaei, J. M. Barros Da Silva, and C. Fischione, "Fedcau: A proactive stop policy for communication and computation efficient federated learning," *IEEE Transactions on Wireless Communications*, vol. 23, no. 9, pp. 11 076–11 093, 2024.

[137] H.-G. Weigand, "A discrete approach to the concept of derivative," *ZDM – Mathematics Education*, vol. 46, no. 4, pp. 603–619, 2014.

[138] A. Mahmoudi, H. S. Ghadikolaei, and C. Fischione, "Machine learning over networks: Co-design of distributed optimization and communications," in *IEEE International Workshop on Signal Processing Advances in Wireless Communications (SPAWC)*, 2020.

[139] A. Mahmoudi, J. M. B. D. S. Júnior, H. S. Ghadikolaei, and C. Fischione, "A-LAQ: Adaptive lazily aggregated quantized gradient," in *2022 IEEE Globecom Workshops (GC Wkshps)*, 2022, pp. 1828–1833.

[140] K. Koh, S.-J. Kim, and S. Boyd, "An interior-point method for large-scale ℓ_1-regularized logistic regression," *Journal of Machine Learning Research*, vol. 8, no. Jul, pp. 1519–1555, 2007.

[141] H. Sifaou and G. Y. Li, "Over-the-air Federated Learning over scalable cell-free massive MIMO," *IEEE Transactions on Wireless Communications*, vol. 23, no. 5, pp. 4214–4227, 2024.

[142] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2016.

[143] Z. Yang, M. Chen, W. Saad, C. S. Hong, and M. Shikh-Bahaei, "Energy efficient Federated Learning over wireless communication networks," *IEEE Transactions on Wireless Communications*, vol. 20, no. 3, pp. 1935–1949, 2020.

[144] B. Widrow and I. Kollár, *Quantization noise: roundoff error in digital computation, signal processing, control, and communications.* Cambridge University Press, 2008.

[145] A. Mahmoudi, M. Zaher, and E. Björnson, "Joint energy and latency optimization in Federated Learning over cell-free massive MIMO networks," in *2024 IEEE Wireless Communications and Networking Conference (WCNC)*, 2024, pp. 1–6.

Chapter 7

Appended Papers