UNIVERSITY OF CALIFORNIA

Los Angeles

Personalized and Efficient Distributed Machine Learning

A dissertation submitted in partial satisfaction

of the requirements for the degree

Doctor of Philosophy in Electrical and Computer Engineering

by

Kaan Ozkara

2025

ABSTRACT OF THE DISSERTATION

Personalized and Efficient Distributed Machine Learning

by

Kaan Ozkara
Doctor of Philosophy in Electrical and Computer Engineering
University of California, Los Angeles, 2025
Professor Suhas Diggavi, Chair

Modern machine learning faces two simultaneous explosions of scale: i) vast amount of data generated at the edge, ii) number of model parameters have scaled up to billions, particularly for large language models (LLMs). Firstly, data is now generated from an ecosystem of devices whose statistical characteristics and hardware capabilities differ widely, demanding a personalized distributed learning that enables collaboration of heterogeneous clients and is tailored according to specific needs. Secondly, model capacity has grown to billions of parameters, stretching the limits of memory bandwidth and computational capabilities, which necessitates efficient training methodologies while preserving numerical stability. The goal of this thesis is to advance both fronts. For personalization, we develop a statistical framework that enables a fundamental understanding and characterization of client heterogeneity both in supervised and unsupervised learning settings. Our methodologies allow collaboration of clients that may possess data from distinct distributions, and have different resource constraints. For large scale training of LLMs, we propose novel meta-optimizers that speeds up convergence, and lower precision training strategies that enable training with lower memory and faster throughput. The contributions of this thesis can be summarized as follows:

- We introduce QuPeD, Quantized Personalization via Distillation, where we introduce a relaxed quantization objective coupled with knowledge distillation thats lets each client learn a model compressed to its own precision and architectural constraints while enabling collaboration. We develop an alternating proximal gradient algorithm that outperforms prior personalized FL baselines across diverse vision and language tasks. The convergence of our method reveals the coupling between heterogeneity and resource constraints.

- We generalize the hierarchical Bayes framework to personalized FL. We unify disparate personalization techniques—regularization, model interpolation, clustering—under one statistical lens, which also yields AdaPeD, an information geometry regularized algorithm. Extending the framework, we provide user level differential privacy guarantees without sacrificing personalized performance. Furthermore, we utilize a similar framework for unsupervised learning; viewing client models through a hierarchical prior, we design adaptive algorithms for personalized dimensionality reduction and diffusion based generation. Our analysis reveals the provable utility obtained from collaboration for generative models, and experiments confirm these gains. For the personalized diffusion generative models we further propose an architectural personalization, using a shared backbone and individual client identities to steer the shared backbone.

- For LLM training, we develop MADA, which parameterizes a spectrum of adaptive optimizer rules (Adam, AMSGrad, etc.), and employ hyper gradient descent to select the best optimizer interpolation adaptively online. MADA consistently outperforms popular optimizers on pre training/fine tuning and vision tasks, and our theory shows how optimizer interpolation tightens convergence bounds.

- We propose using stochastic rounding (SR) for Adam optimizer updates to enable a full BF16 training recipe. Theoretically, we show SR results in implicit regularization of the loss function and faster convergence compared to deterministic rounding approaches.

Empirically, our recipe yields higher throughput and lower memory while obtaining better validation perplexity compared to state-of-the-art mixed precision training.

The dissertation of Kaan Ozkara is approved.

Lieven Vandenberghe

Quanquan Gu

Lin Yang

Suhas Diggavi, Committee Chair

University of California, Los Angeles

2025

To my family

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ACKNOWLEDGMENTS

Reading an "Acknowledgment" section of a thesis always felt intriguing to me. It is fascinating to peek through people's lives through these couple of pages. After reading an Acknowledgments section, I am often times left with an amazement on the impact of so many people in one's life. Now I am on the other side, I know the words will not be enough, but I will try my best to let you, dear reader, peek through my life and see how lucky I am to be surrounded by such unique and brilliant people.

Some describe the PhD as a solitary journey that demands complete self-reliance. I see it instead as a partnership whose quality also depends on the advisor's guidance. I am deeply grateful to Prof. Suhas Diggavi for creating the supportive, nurturing environment that allowed me to grow, both technically and personally, over the past five and a half years. From our first meeting, I was inspired by his curiosity and unique approach to research. His reverence for science has shaped my own outlook and helped me mature as a scholar. I am thankful that his intellectual influence will accompany me throughout my life.

I have been fortunate to collaborate with many people who have shaped my growth as a researcher. I am especially grateful to Dr. Deepesh Data, whose guidance during the most challenging early stages kept me on track. His calm demeanor, positiveness, and meticulous attention to detail helped me through critical transition. I am forever thankful to Dr. Can Karakus, one of the most influential persons in my PhD journey. An exceptional engineer, researcher, mentor; and above all, a steadfast friend, his support has been invaluable.

I would like to thank Prof. Lieven Vandenberghe, Prof. Quanquan Gu, and Prof. Lin Yang, for taking the time to serve in my committee, and for their valuable feedback. Prof. Lieven Vandenberghe's courses have introduced me to optimization, I would like to thank him for being a great teacher.

Los Angeles became a true home thanks to my remarkable labmates. I am grateful to Bruce, Navjot, Antonious, Yağız, Ruida, Rida, Dhaivat, as well as to our sister-labmates

Mine, Osama, Merve, Kaan, Xinlin, and Chaorui. Their friendship filled UCLA with warmth, turning work days into lasting memories. I will especially miss the Thanksgiving dinners generously hosted by Professors Christina Fragouli and Suhas Diggavi, they captured the spirit of family I was lucky to experience during my doctoral journey.

I owe special thanks to Berkin for being such a source of joy and laughter even in the darkest days. I want to thank Taylan, he was such a great friend that I would not even mind driving in LA rush hours. I would like to thank Mine, whose contagious positiveness always lifted my spirits. I am deeply grateful to my lifelong friends Ege, Cenk, Alper, Yilmaz, Berke. Even with thousands of miles between us, our friendship has never wavered. Your support has been an anchor throughout this journey.

I am profoundly grateful to my friends Arda, Gökalp, and Mustafa. Our group chat has been one of the highlights of my PhD. Our late night gaming sessions have been a refuge from the stress of life and PhD. Arda's kindness and honesty continually inspire me to be a better person. Gökalp's boundless energy and optimism remind me to persevere through the inevitable lows. Mustafa's heartiness and friendship has assured me that someone will always listen and stand by me, even on the loneliest days. I will cherish their support for the rest of my life.

I would like to thank my family Dilek, Gülay, Ali, Hakan, Sancak, and Kamile for their unwavering support and unconditional love. I have been very fortunate to feel your presence beside me at every step. I owe special thanks to my parents, Gülay and Ali, who have always placed my well-being ahead of their own. Your selflessness is a debt I can never repay.

Finally, I extend my deepest gratitude to my partner Dilek, whose presence has added color and joy to every moment of my life. Our walks, travels, conversations, meals, and love have provided the encouragement that made this thesis possible. In spirit, you are its co-author, and your influence resonates in these lines. I am looking forward to chapters of our lives that we will author together.

VITA

2019	B.Sc. in Electrical and Electronics Engineering, Bilkent University, Ankara, Turkey.
2019	Joined PhD program at the Department of Electrical and Computer Engineering at University of California, Los Angeles.
2019	M.Sc in Electrical and Computer Engineering at University of California, Los Angeles.
2022	Applied Scientist Intern at Amazon Web Services.
2023	Advanced to Ph.D. Candidacy.
2023	Applied Scientist Intern at Amazon Web Services AI, Santa Clara, CA.
2024	Amazon PhD Fellow.
2024	Applied Scientist Intern at Amazon Web Services AI, Santa Clara, CA.

PUBLICATIONS

Kaan Ozkara, Navjot Singh, Deepesh Data, Suhas Diggavi. "QuPeD : Quantized Personalization via Distillation with Applications to Federated Learning", Advances in Neural Information Processing Systems (NeurIPS), 2021.

Kaan Ozkara*, Antonious Girgis*, Deepesh Data, Suhas Diggavi. "A Statistical Framework for Personalized Federated Learning and Estimation : Theory, Algorithms, and Privacy". International Conference on Learning Representations (ICLR) , 2023.

Kaan Ozkara, Can Karakus, Parameswaran Raman, Mingyi Hong, Shoham Sabach, Branislav Kveton, Volkan Cevher. "MADA : Meta-Adaptive Optimizers through Hypergradient Descent". International Conference on Machine Learning (ICML) , 2024.

Kaan Ozkara, Tao Yu, Youngsuk Park. "Stochastic Rounding for LLM Training : Theory and Practice". International Conference on Artificial Intelligence and Statistics (AISTATS), 2025.

Kaan Ozkara, Bruce Huang, Ruida Zhou, Suhas Diggavi. "ADEPT : Hierarchical Bayes Approach to Personalized Federated Unsupervised Learning". International Conference on Artificial Intelligence and Statistics (AISTATS), 2025.

Kaan Ozkara, Bruce Huang, Suhas Diggavi. "Personalized PCA for Federated Heterogeneous Data". IEEE International Symposium on Information Theory (ISIT), 2023.

---

* Equal Contribution.

CHAPTER 1

Introduction

The last decade has witnessed an unprecedented double-explosion of scale in machine learning. On one axis, data generation has migrated to the edge: billions of phones, wearables, and IoT sensors stream privacy-sensitive data whose statistical signatures vary as widely as the hardware that produces them. On the orthogonal axis, model capacity has ballooned to the billion-parameter regime, driven most visibly by large language models (LLMs). Taken together, these two forces render the classical paradigm of centralized, one-size-fits-all training both economically and statistically untenable.

On the first front, data generation at the edge introduced novel challenges. Edge devices collect data that are non-i.i.d. across clients and often scarce within any single device. Training a single global model therefore risks underfitting rare sub-populations and overfitting dominant ones, while purely local models squander the collective data abundance strength of the system. The remedy is personalized federated learning (FL), in which clients collaborate to jointly learn models that adapt to their unique data distributions. Furthermore, heterogeneity is not merely data driven. Clients range from micro-controllers with kilobytes of memory to data-center GPUs. A practical solution must match model size, precision, and update frequency to each client's budget while still allowing meaningful collaboration.

On the other hand, due to advances in hardware/accelerators and model architectures such as transformer blocks, we are now able to scale our deep learning models to billions of parameters. Training these large models strains the hardware and valuable resources such as training time. We thus need algorithms that converge faster, communicate less, and operate

in low precision—all without compromising numerical stability and downstream accuracy.

1.1 Contributions and Outline

In Chapters 2-4, we delve into the problem of heterogeneity in FL and propose personalization as a solution. While the traditional FL pertains using a single global model for all clients, we show its shortcomings and how carefully designed and statistically motivated optimization criteria allows design of personalized algorithms that is suitable under data heterogeneity.

In particular, in Chapter 2, we introduce QuPeD [131] where we augment the traditional FL distributed optimization criterion with two additional terms: (i) a knowledge distillation penalty between global and personalized model logits that allows collaboration of models with different architectures and precision, (ii) a soft quantization penalty term that enables quantization aware training. We examine the overall convergence behavior and discover a coupling between data heterogeneity and clients' quantization aggression. Specifically, clients that are relevantly more out of distribution can tolerate the quantization less and vice versa. Lastly, we empirically compare our model to other personalized and non-personalized FL algorithms, our algorithm outperforms all the competing methods.

In Chapter 3, we introduce a statistical framework that is based on a hierarchical Bayes view of clients' data generation [126]. Our framework is based on maximum a posteriori estimation of clients' model parameters given a prior model which induces different regularizers in optimization criteria and dictates collaboration. By adjusting the prior model assumption, our framework obtains diverse set of personalization algorithms that were explored in the literature, as well as it is able to propose new algorithms with adaptation. We utilize our framework both in personalized mean estimation and learning, under differential privacy and communication efficiency constraints. We derive theoretical performance guarantees indicating the utility rooted from collaboration. Empirically, both for estimation and learning problems we show the success of our method under various prior assumption such as based

on Gaussian, GMM, information geometric population distributions.

Afterwards, in Chapter 4, we extend personalized FL framework to unsupervised learning tasks such as dimensionality reduction (personalized PCA and Autoencoders) and personalized diffusion generative models [129]. Furthermore, we introduce a notion of hyper-priors that helps with empirical stability and prevents ill-posedness. We introduce `ADEPT` as a family of algorithms. We prove convergence results that draws a connection between heterogeneity and adaptivity, such as on Stiefel manifold for personalized PCA. We show how collaboration may help with better generation quality.

In Chapter 5, we propose a particular architectural partitioning for personalized generative models. We assume the model is made up from two parts: a lower dimensional-easy to learn personalized part and a larger backbone that is shared among clients. We connect this architecture learning Gaussian Mixture Models and derive error bound for learning mixing weights that provide insights for the personalized diffusion model training. Our proposed architecture excels at new client fine-tuning tasks.

Chapters 6.6 and 7.6 are related to the second front of training LLMs. In Chapter 6.6 we develop MADA, which parameterizes a spectrum of adaptive optimizer rules (Adam, AMSGrad, etc.) to construct an optimizer space, and employ hyper gradient descent to select the best optimizer interpolation adaptively online. By adaptively searching the optimizer space, MADA consistently outperforms popular optimizers on pre training/fine tuning and vision tasks, and theoretically we show that optimizer interpolation may result in faster convergence.

In Chapter 7.6, we propose using stochastic rounding (SR) for Adam optimizer updates to enable a full BF16 training recipe. Theoretically, we show SR results in implicit regularization of the loss function and faster convergence compared to deterministic rounding approaches. Empirically, our recipe yields higher throughput and lower memory while obtaining better validation perplexity compared to state-of-the-art mixed precision training.

CHAPTER 2

Quantized Personalization via Distillation

Traditionally, federated learning (FL) aims to train a single global model while collaboratively using multiple clients and a server. Two natural challenges that FL algorithms face are heterogeneity in data across clients and collaboration of clients with *diverse resources*. In this work, we introduce a *quantized* and *personalized* FL algorithm QuPeD that facilitates collective (personalized model compression) training via *knowledge distillation* (KD) among clients who have access to heterogeneous data and resources. For personalization, we allow clients to learn *compressed personalized models* with different quantization parameters and model dimensions/structures. Towards this, first we propose an algorithm for learning quantized models through a relaxed optimization problem, where quantization values are also optimized over. When each client participating in the (federated) learning process has different requirements for the compressed model (both in model dimension and precision), we formulate a compressed personalization framework by introducing knowledge distillation loss for local client objectives collaborating through a global model. We develop an alternating proximal gradient update for solving this compressed personalization problem, and analyze its convergence properties. Numerically, we validate that QuPeD outperforms competing personalized FL methods, FedAvg, and local training of clients in various heterogeneous settings.

2.1 Introduction

Federated Learning (FL) is a learning procedure where the aim is to utilize vast amount of data residing in numerous (in millions) edge devices (clients) to train machine learning models without collecting clients' data [116]. Formally, if there are n clients and f_i denotes the local loss function at client i, then traditional FL learns a single global model by minimizing

$$\arg\min_{\mathbf{w}\in\mathbb{R}^d} \left(f(\mathbf{w}) := \frac{1}{n}\sum_{i=1}^{n} f_i(\mathbf{w}) \right). \tag{2.1}$$

It has been realized lately that a *single* model may not provide good performance to all the clients in settings where data is distributed *heterogeneously*. This leads to the need for personalized learning, where each client wants to learn its own model [42, 48]. Since a locally learned client model may not generalize well due to insufficient data, in personalized FL process, clients maintain personalized models locally and utilize other clients' data via a global model. *Resource diversity* among clients, which is inherent to FL as the participating edge devices may vary widely in terms of resources, is often overlooked in personalized FL literature. This resource diversity may necessitate clients to learn personalized models with *different precision* as well as *different dimension/architecture*. Systematically studying both these resource heterogeneity together with data heterogeneity in personalized FL is the primary objective of this paper.

In this work, we propose a model compression framework[1] for personalized FL via knowledge distillation (KD) [69] that addresses both data and resource heterogeneity in a unified manner. Our framework allows collaboration among clients with different resource requirements both in terms of *precision* as well as *model dimension/structure*, for learning personalized quantized models (PQMs). Motivated by FL, where edge devices are resource constrained when actively used (*e.g.* when several applications are actively running on a

[1]Model compression (MC) allows inference time deployment of a compressed model. Though MC is a generic term comprising different methods, we will focus on its quantization (number of bits per model parameter) aspect.

battery powered smartphone) and available for training when not in use (*e.g.*, while charging and on wi-fi), we do training in full precision for learning compressed models to be deployed for *inference time*. For efficient model compression, we learn the quantization parameters for each client by including quantization levels in the optimization problem itself. First, we investigate our approach in a *centralized* setup, by formulating a relaxed optimization problem and minimizing it through alternating proximal gradient steps, inspired by [18]. To extend this to FL for learning PQMs with *different* dimensions/architectures, we employ our centralized algorithm locally at clients and introduce KD loss for collaboration of personalized and global models. Although there exist empirical works where KD is used in personalized FL [96], we formalize it as an optimization problem, solve it using alternating proximal updates, and analyze its convergence.

Contributions. Our contributions can be summarized as follows:

- In the centralized case, we propose a novel relaxed optimization problem that enables optimization over quantization values (centers) as well as model parameters. We use alternating proximal updates to minimize the objective and analyze its convergence properties.

- More importantly, our work is the first to formulate a personalized FL optimization problem where clients may have different model dimensions and precision requirements for their personalized models. Our proposed scheme combines alternating proximal updates with knowledge distillation.

- For optimizing a non-convex objective, in the centralized setup, we recover the standard convergence rate of $\mathcal{O}(1/T)$ (despite optimizing over quantization centers), and for federated setting, we recover the standard convergence rate of $\mathcal{O}(1/\sqrt{T})$ (despite learning PQMs with different precisions/dimensions). In the federated setting, our convergence bound has an error term that depends on multiplication of two terms averaged over clients: one characterizing client's local model smoothness and the other

data heterogeneity with respect to overall data distribution.[2]

- We perform image and text classification experiments on multiple datasets in various resource and data heterogeneity settings, and compare performance of QuPeD against Per-FedAvg [48], pFedMe [42], FedAvg [116], and local training of clients. We observe that QuPeD in full precision outperforms all these methods on all the datasets that we considered for our experiments. Further, our results show that even with quantization, QuPeD outperforms the other methods in *full precision* on multiple settings and datasets, demonstrating the effectiveness of our scheme for FL settings.

Our work should not be confused with works in distributed/federated learning, where models/gradients are compressed for *communication efficiency* [12, 83]. We also achieve communication efficiency through local iterations, but the main goal of our work is personalized quantization for inference.

Related work. To the best of our knowledge, this is the first work in personalized federated learning where the aim is to learn quantized and personalized models potentially having different dimensions/structures for inference. Our work can be seen in the intersection of *personalized federated learning* and *learning quantized models*; we also employ *knowledge distillation* for collaboration. *Personalized federated learning:* Recent works adopted different approaches for learning personalized models: (i) Combine global and local models throughout the training [40,65,114]; (ii) first learn a global model and then personalize it locally [48]; (iii) consider multiple global models to collaborate among only those clients that share similar personalized models [56,114,157,185]; (iv) augment the traditional FL objective via a penalty term that enables collaboration between global and personalized models [42,64,65].

Learning quantized models: There are two kinds of approaches for training quantized networks that are of our interest. The first one approximates the hard quantization function by using a soft surrogate [38,62,110,175], while the other one iteratively projects the model

[2]An error term depending on data heterogeneity is commonly observed in personalized FL algorithms [42,48].

parameters onto the fixed set of centers [7, 71, 94, 177]. Each approach has its own limitation; see Section 2.2.1 for a discussion. While the initial focus in learning quantized networks was on achieving good empirical performance, there are some works that analyzed convergence properties [7, 97, 177], but only in the centralized case. Among these, [7] analyzed convergence for a relaxed/regularized loss function using proximal updates. *Knowledge distillation (KD):* KD [69] is a framework for transfer learning that is generally used to train a small student network using the soft labels generated by a deep teacher network. It can also be used to train two or more networks mutually by switching teacher and students in each iteration [186]. KD has been employed in FL settings as an alternative to simple aggregation which is not feasible when clients have models with different dimensions [102]. [96] used KD in personalized FL by assuming existence of a public dataset. [135] used KD in combination with quantization in a centralized case for model compression; in contrast, we do not use KD for model compression but for collaboration between personalized and global model. Unlike the above works which are empirical, our paper is the first to formalize an optimization problem for personalized FL training with KD and analyze its convergence properties. Our proposed scheme yields personalized client models with different precision/dimension through local alternating proximal updates; see Section 2.2.2 for details.

Paper organization: In Section 2.2, we formulate the optimization problem to be minimized. In Sections 2.3 and 2.4, we describe our algorithms along-with the main convergence results for the centralized and personalized settings, respectively. Section 7.5 provides extensive numerical results. Omitted proofs/details and additional experiments are provided in supplementary material.

2.2 Problem Formulation

Our goal in this paper is for clients to collaboratively learn personalized quantized models (with potentially different precision and model sizes/types). To this end, below, we first

state our final objective function that we will end up optimizing in this paper for learning personalized quantized models, and then in the rest of this section we will describe the genesis of this objective.

Recall from (2.1), in the traditional FL setting, the local loss function at client i is denoted by f_i. For personalized compressed model training, we define the following augmented loss function at client i:

$$F_i(\mathbf{x}_i, \mathbf{c}_i, \mathbf{w}) := (1-\lambda_p)\left(f_i(\mathbf{x}_i) + f_i(\widetilde{Q}_{\mathbf{c}_i}(\mathbf{x}_i))\right) + \lambda R(\mathbf{x}_i, \mathbf{c}_i) + \lambda_p\left(f_i^{KD}(\mathbf{x}_i, \mathbf{w}) + f_i^{KD}(\widetilde{Q}_{\mathbf{c}_i}(\mathbf{x}_i), \mathbf{w})\right). \tag{2.2}$$

Here, $\mathbf{w} \in \mathbb{R}^d$ denotes the global model, $\mathbf{x}_i \in \mathbb{R}^{d_i}$ denotes the personalized model of dimension d_i at client i, $\mathbf{c}_i \in \mathbb{R}^{m_i}$ denotes the model quantization centers (where m_i is the number of centers), $\widetilde{Q}_{\mathbf{c}_i}$ denotes the soft-quantization function with respect to (w.r.t.) the set of centers $\mathbf{c}_i$, $R(\mathbf{x}_i, \mathbf{c}_i)$ denotes the distance function, f_i^{KD} denotes the knowledge distillation (KD) loss [69] between the two input models on client i's dataset, λ is a design parameter for enforcing quantization (large λ forces weights to be close to respective centers), and λ_p controls the weighted average of regular loss and KD loss functions (higher λ_p can be used when client data is limited). We will formally define the undefined quantities later in this section. Consequently, our main objective becomes:

$$\min_{\mathbf{w}\in\mathbb{R}^d, \{\mathbf{x}_i\in\mathbb{R}^{d_i}, \mathbf{c}_i\in\mathbb{R}^{m_i}: i=1,\ldots,n\}} \left(F\Big(\mathbf{w}, \{\mathbf{x}_i\}, \{\mathbf{c}_i\}\Big) := \frac{1}{n}\sum_{i=1}^{n} F_i(\mathbf{x}_i, \mathbf{c}_i, \mathbf{w})\right). \tag{2.3}$$

Thus, our formulation allows different clients to have personalized models $\mathbf{x}_1, \ldots, \mathbf{x}_n$ with different dimensions $d_1, \ldots, d_n$ and architectures, different number of quantization levels $m_1, \ldots, m_n$ (larger the m_i, higher the precision), and different quantization values in those quantization levels. Note that there are two layers of personalization, first is due to data heterogeneity, which is reflected in clients learning different models $\mathbf{x}_1, \ldots, \mathbf{x}_n$, and second is due to resource diversity, which is reflected in clients learning models with different sizes, both in terms in dimension as well as precision.

In Section 2.2.1, we motivate how we came up with the first three terms in (2.2), which are in fact about a centralized setting because the function f_i and the parameters involved, i.e., $\mathbf{x}_i, \mathbf{c}_i$, are local to client i; and then, in Section 2.2.2, we motivate the use of the last two terms containing f_i^{KD} in (2.2).

2.2.1 Model Compression in the Centralized Setup

Consider a setting where an objective function $f : \mathbb{R}^{d+m} \to \mathbb{R}$ (which could be a neural network loss function) is optimized over both the quantization centers $\mathbf{c} \in \mathbb{R}^m$ and the assignment of model parameters (or weights) $\mathbf{x} \in \mathbb{R}^d$ to those centers. There are two ways to approach this problem, and we describe these approaches, their limitations, and the possible resolutions below.

Approach 1. A natural approach is to explicitly put a constraint that weights belong to the set of centers, which suggests solving the following problem: $\min_{\mathbf{x},\mathbf{c}} f(\mathbf{x}) + \delta_{\mathbf{c}}(\mathbf{x})$, where $\delta_{\mathbf{c}}$ denotes the indicator function for $\mathbf{c} \in \mathbb{R}^m$, and for any $\mathbf{c} \in \mathbb{R}^m, \mathbf{x} \in \mathbb{R}^d$, define $\delta_{\mathbf{c}}(\mathbf{x}) := 0$ if $\forall\ j \in [d]$, $x_j = c$ for some $c \in \{c_1, \ldots, c_m\}$, otherwise, define $\delta_{\mathbf{c}}(\mathbf{x}) := \infty$. However, the discontinuity of $\delta_{\mathbf{c}}(\mathbf{x})$ makes minimize this objective challenging. To mitigate this, like recent works [7, 177], we can approximate $\delta_{\mathbf{c}}(\mathbf{x})$ using a distance function $R(\mathbf{x}, \mathbf{c})$ that is continuous everywhere (*e.g.*, the ℓ_1-distance, $R(\mathbf{x}, \mathbf{c}) := \min\{\frac{1}{2}\|\mathbf{z} - \mathbf{x}\|_1 : z_i \in \{c_1, \cdots, c_m\}, \forall i\}$).[3] This suggests solving:

$$\min_{\mathbf{x},\mathbf{c}} f(\mathbf{x}) + \lambda R(\mathbf{x}, \mathbf{c}). \tag{2.4}$$

The centers are optimized to be close to the mean or median (depending on R) of the weights; however, there is no guarantee that this will help minimizing objective f. We believe that modeling the direct effect that centers have on the loss is crucial for a complete quantized training (see Appendix in [131] for empirical verification of this fact), and our second approach

[3] [7] and [177] proposed to approximate the indicator function $\delta_{\mathbf{c}}(\mathbf{x})$ using a distance function $R_{\mathbf{c}}(\mathbf{x})$, where $\mathbf{c}$ is fixed, and unlike ours, it is not a variable that the loss function is optimized over.

is based on this idea.

Approach 2. We can embed the quantization function into the loss function itself, thus solving the problem: $\min_{\mathbf{x},\mathbf{c}} \Big(h(\mathbf{x},\mathbf{c}) := f(Q_{\mathbf{c}}(\mathbf{x}))\Big)$, where for every $\mathbf{x} \in \mathbb{R}^d, \mathbf{c} \in \mathbb{R}^m$, the (hard) quantization function is defined as $Q_{\mathbf{c}}(\mathbf{x})_i := c_k$, where $k = \arg\min_{j\in[m]}\{|x_i - c_j|\}$, which maps individual weights to the closest centers. Note that $Q_{\mathbf{c}}(\mathbf{x})$ is actually a staircase function for which the derivative w.r.t. $\mathbf{x}$ is 0 almost everywhere, which discourages the use of gradient-based methods to optimize the above objective. To overcome this, similar to [62, 175], we can use a *soft* quantization function $\widetilde{Q}_{\mathbf{c}}(\mathbf{x})$ that is differentiable everywhere with derivative not necessarily 0. For e.g., element-wise *sigmoid* or *tanh* functions, used by [175] and [62], respectively.[4] This suggests the following relaxation:

$$\min_{\mathbf{x},\mathbf{c}} \Big(h(\mathbf{x},\mathbf{c}) := f(\widetilde{Q}_{\mathbf{c}}(\mathbf{x}))\Big). \tag{2.5}$$

Though we can observe the effect of centers on neural network loss in (2.5); however, the gradient w.r.t. $\mathbf{x}$ is heavily dependent on the choice of $\widetilde{Q}_{\mathbf{c}}$ and optimizing over $\mathbf{x}$ might deviate too much from optimizing the neural network loss function. For instance, in the limiting case when $\widetilde{Q}_{\mathbf{c}}(\mathbf{x}) \to Q_{\mathbf{c}}(\mathbf{x})$, gradient w.r.t. $\mathbf{x}$ is 0 almost everywhere; hence, every point becomes a first order stationary point.

Our proposed objective for model quantization. Our aim is to come up with an objective function that would not diminish the significance of both $\mathbf{x}$ and $\mathbf{c}$ in the overall procedure. To leverage the benefits of both, we combine both optimization problems (2.4) and (2.5) into one problem:

$$\min_{\mathbf{x},\mathbf{c}} \Big(F_\lambda(\mathbf{x},\mathbf{c}) := f(\mathbf{x}) + f(\widetilde{Q}_{\mathbf{c}}(\mathbf{x})) + \lambda R(\mathbf{x},\mathbf{c})\Big). \tag{2.6}$$

Here, the first term preserves the connection of $\mathbf{x}$ to neural network loss function, and the second term enables the optimization of centers w.r.t. the neural network training loss itself. As a result, we obtain an objective function that is continuous everywhere, and for which we

[4] In their setup, the quantization centers are fixed. In contrast, we are also optimizing over these centers.

can use Lipschitz tools in the convergence analysis – which previous works did not exploit. In fact, we compute the Lipschitz parameters for a specific soft quantization function $\widetilde{Q}_{\mathbf{c}}(\mathbf{x})$ based on *sigmoid* in Appendix of [131].

Remark 1. *It is important to note that with this new objective function, we are able to optimize not only over weights but also over centers. This allows us to theoretically analyze how the movements of the centers affect the convergence. As far as we know, this has not been the case in the literature of quantized neural network training. Moreover, we observe numerically that optimizing over centers improves performance of the network; see Appendix of [131].*

2.2.2 Towards Personalized Quantized Federated Learning: Knowledge Distillation

Note that the objective function defined in (2.6) can be used for learning a quantized model locally at any client. There are multiple ways to extend that objective for learning personalized quantized models (PQMs) via collaboration. For example, when all clients want to learn personalized models with the *same* dimension (but with different quantization levels), then one natural approach is to add an ℓ_2 penalty term in the objective that would prevent local models from drifting away from the global model and from simply fitting to local data. This approach, in fact, has been adopted in previous works [42, 65, 101] for learning personalized models in FL, though not quantized ones.[5] In this paper, since we allow clients to learn PQMs with potentially *different* dimensions, the above approach of adding a ℓ_2 penalty term in the objective is not feasible. Observe that, the purpose of incorporating a ℓ_2 penalty in the objective is to ensure that the personalized models do not have significantly different output class scores compared to the global model which is trained using the data generated at all

[5]We also analyze this approach in our setting (for learning PQMs but having the *same* dimension), and we call this version QuPeL; see Appendix of [131] for problem setup and convergence results of QuPeL. In Section 7.5, we demonstrate that QuPeD (for the same task but using KD as opposed to the ℓ_2 penalty) outperforms QuPeL.

clients; this does not, however, require the global model to have the same dimension as that of local models and can be satisfied by augmenting the local objective (2.6) with a certain *knowledge distillation* (KD) loss. In our setting, since clients' goal is to learn personalized models with different dimensions that may also have *different* quantization levels, we augment the local objective (2.6) with two separate KD losses: $f_i^{KD}(\mathbf{x}_i, \mathbf{w})$ and $f_i^{KD}(\widetilde{Q}_{\mathbf{c}_i}(\mathbf{x}_i), \mathbf{w})$, where the first one ensures that the behavior of $\mathbf{x}_i \in \mathbb{R}^{d_i}$ is not very different from that of $\mathbf{w} \in \mathbb{R}^d$, and the second one ensures the same for the quantized version of $\mathbf{x}_i$ and $\mathbf{w}$. Formally, we define them using KL divergence as follows: $f_i^{KD}(\mathbf{x}_i, \mathbf{w}) := D_{KL}(s_i^w(\mathbf{w}) \| s_i(\mathbf{x}_i))$ and $f_i^{KD}(\widetilde{Q}_{\mathbf{c}_i}(\mathbf{x}_i), \mathbf{w}) := D_{KL}(s_i^w(\mathbf{w}) \| s_i(\widetilde{Q}_{\mathbf{c}_i}(\mathbf{x}_i)))$, where s_i^w and s_i denote functions whose inputs are global and personalized models, respectively – and data samples implicitly – and outputs are the softmax classification probabilities of the network.

We need to train $\mathbf{x}_i$ and $\mathbf{w}$ mutually. Identifying the limitations of existing approaches for theoretical analysis (as mentioned in related work in Section 5.1), we use reverse KL updates (*i.e.*, taking gradient steps w.r.t. the first parameter in $D_{KL}(\cdot, \cdot)$) to train the teacher network $\mathbf{w}$ from the student network $\mathbf{x}_i$. This type of update can be shown to converge and also empirically outperforms [152] (see Section 7.5). We want to emphasize that though there are works [96, 102] that have used KD in FL and studied its performance (only empirically), ours is the first work that carefully formalizes it as an optimization problem (that also incorporate quantization) which is necessary to analyze convergence properties.

2.3 Centralized Model Quantization Training

In this section, we propose a centralized training scheme (Algorithm 1) for minimizing (2.6) by optimizing over $\mathbf{x} \in \mathbb{R}^d$ (the model parameters) and $\mathbf{c} \in \mathbb{R}^m$ (quantization values/centers). During training, we keep $\mathbf{x}$ full precision and learn the optimal quantization parameters $\mathbf{c}$. The learned quantization values are then used to hard-quantize the personalized models to get quantized models for deployment in a memory-constrained setting.

Algorithm 1 Centralized Model Quantization

Input: Regularization parameter λ; initialize the full precision model $\mathbf{x}^0$ and quantization centers $\mathbf{c}^0$; a penalty function enforcing quantization $R(\mathbf{x}, \mathbf{c})$; a soft quantizer $\widetilde{Q}_{\mathbf{c}}(\mathbf{x})$; and learning rates η_1, η_2.

1: **for** $t = 0$ **to** $T - 1$ **do**

2: Compute $\mathbf{g}^t = \nabla_{\mathbf{x}^t} f(\mathbf{x}^t) + \nabla_{\mathbf{x}^t} f(\widetilde{Q}_{\mathbf{c}^t}(\mathbf{x}^t))$

3: $\mathbf{x}^{t+1} = \mathrm{prox}_{\eta_1 \lambda R_{\mathbf{c}^t}}(\mathbf{x}^t - \eta_1 \mathbf{g}^t)$

4: Compute $\mathbf{h}^t = \nabla_{\mathbf{c}^t} f(\widetilde{Q}_{\mathbf{c}^t}(\mathbf{x}^{t+1}))$

5: $\mathbf{c}^{t+1} = \mathrm{prox}_{\eta_2 \lambda R_{\mathbf{x}^{t+1}}}(\mathbf{c}^t - \eta_2 \mathbf{h}^t)$

6: **end for**

Output: Quantized model $\hat{\mathbf{x}}^T = Q_{\mathbf{c}^T}(\mathbf{x}^T)$

Description of the algorithm. We optimize (2.6) through alternating proximal gradient descent. The model parameters and the quantization vector are initialized to random vectors $\mathbf{x}^0$ and $\mathbf{c}^0$. The objective in (2.6) is composed of two parts: the loss function $f(\mathbf{x}) + f(\tilde{Q}_{\mathbf{c}}(\mathbf{x}))$ and a quantization inducing term $R(\mathbf{x}, \mathbf{c})$, which we control by a regularization coefficient λ. At each t, we compute gradient $\mathbf{g}^t$ of the loss function w.r.t. $\mathbf{x}^t$ (line 2), and then take the gradient step followed by prox step for updating $\mathbf{x}^t$ to $\mathbf{x}^{t+1}$ (line 3). For the centers, we similarly take a gradient step and follow it by a prox step for updating $\mathbf{c}^t$ to $\mathbf{c}^{t+1}$ (line 4-5). These update steps ensure that we learn the model parameters and quantization vector tied together through proximal[6] mapping of the regularization function R. Finally, we quantize the full-precision model $\mathbf{x}^T$ using $Q_{\mathbf{c}^T}$ (line 7).

Assumptions. We make the following assumptions on f:

A.1 *(Finite lower bound of f):* $f(\mathbf{x}) > -\infty$, $\forall \mathbf{x} \in \mathbb{R}^d$, which implies $F_\lambda(\mathbf{x}, \mathbf{c}) > -\infty$ for any $\mathbf{x} \in \mathbb{R}^d, \mathbf{c} \in \mathbb{R}^m, \lambda \in \mathbb{R}$.

[6]As a short notation, we use $\mathrm{prox}_{\eta_1 \lambda R_{\mathbf{c}^t}}$ to denote $\mathrm{prox}_{\eta_1 \lambda R(\cdot, \mathbf{c}^t)}$, and $\mathrm{prox}_{\eta_2 \lambda R_{\mathbf{x}^{t+1}}}$ for $\mathrm{prox}_{\eta_2 \lambda R(\mathbf{x}^{t+1}, \cdot)}$.

A.2 *(Smoothness of f):* f is L-smooth, i.e., for all $\mathbf{x}, \mathbf{y} \in \mathbb{R}^d$, we have $f(\mathbf{y}) \leq f(\mathbf{x}) + \langle \nabla f(\mathbf{x}), \mathbf{y} - \mathbf{x} \rangle + \frac{L}{2}\|\mathbf{x} - \mathbf{y}\|^2$.

A.3 *(Bounded gradients of f):* $\|\nabla f(\mathbf{x})\|_2 \leq G < \infty, \forall \mathbf{x} \in \mathbb{R}^d$.

A.1 and **A.2** are standard assumptions for convergence analysis of smooth objectives; and **A.3** is commonly used for non-convex optimization, *e.g.*, for personalized FL in [48]. To make the composite function $f(\widetilde{Q}_{\mathbf{c}}(\mathbf{x}))$ smooth, we need additional assumptions on the soft quantization function $\widetilde{Q}_{\mathbf{c}}(\mathbf{x})$. Our choice of $\widetilde{Q}_{\mathbf{c}}(\mathbf{x})$ (see Appendix in [131]) naturally satisfies these assumptions.

A.4 *(Smoothness of the soft quantizer):* We assume that $\widetilde{Q}_{\mathbf{c}}(\mathbf{x})$ is l_{Q_1}-Lipschitz and L_{Q_1}-smooth w.r.t. $\mathbf{x}$, i.e., for $\mathbf{c} \in \mathbb{R}^m$: $\forall\ \mathbf{x}, \mathbf{y} \in \mathbb{R}^d$: $\|\widetilde{Q}_{\mathbf{c}}(\mathbf{x}) - \widetilde{Q}_{\mathbf{c}}(\mathbf{y})\| \leq l_{Q_1}\|\mathbf{x} - \mathbf{y}\|$ and $\|\nabla_{\mathbf{x}}\widetilde{Q}_{\mathbf{c}}(\mathbf{x}) - \nabla_{\mathbf{y}}\widetilde{Q}_{\mathbf{c}}(\mathbf{y})\| \leq L_{Q_1}\|\mathbf{x} - \mathbf{y}\|$. We also assume $\widetilde{Q}_{\mathbf{c}}(\mathbf{x})$ is l_{Q_2}-Lipschitz and L_{Q_2}-smooth w.r.t. $\mathbf{c}$, i.e., for $\mathbf{x} \in \mathbb{R}^d$: $\forall\ \mathbf{c}, \mathbf{d} \in \mathbb{R}^m$: $\|\widetilde{Q}_{\mathbf{c}}(\mathbf{x}) - \widetilde{Q}_{\mathbf{d}}(\mathbf{x})\| \leq l_{Q_2}\|\mathbf{c} - \mathbf{d}\|$ and $\|\nabla_{\mathbf{c}}\widetilde{Q}_{\mathbf{c}}(\mathbf{x}) - \nabla_{\mathbf{d}}\widetilde{Q}_{\mathbf{d}}(\mathbf{x})\| \leq L_{Q_2}\|\mathbf{c} - \mathbf{d}\|$.

A.5 *(Bound on partial gradients of soft quantizer):* There exists constants $G_{Q_1}, G_{Q_2} < \infty$ such that: $\|\nabla_{\mathbf{x}}\widetilde{Q}_{\mathbf{c}}(\mathbf{x})\|_F = \|\nabla\widetilde{Q}_{\mathbf{c}}(\mathbf{x})_{1:d,:}\|_F \leq G_{Q_1}$ and $\|\nabla_{\mathbf{c}}\widetilde{Q}_{\mathbf{c}}(\mathbf{x})\|_F = \|\nabla\widetilde{Q}_{\mathbf{c}}(\mathbf{x})_{d+1:d+m,:}\|_F \leq G_{Q_2}$, where $\mathbf{X}_{p:q,:}$ denotes sub-matrix of $\mathbf{X}$ with rows between p and q, and $\|\cdot\|_F$ is the Frobenius norm.

Convergence result. Now we state our main convergence result (proved in Appendix of [131]) for minimizing $F_\lambda(\mathbf{x}, \mathbf{c})$ in (2.6) w.r.t. $(\mathbf{x}, \mathbf{c}) \in \mathbb{R}^{d+m}$ via Algorithm 1. This provides first-order guarantees for convergence of $(\mathbf{x}, \mathbf{c})$ to a stationary point and recovers the $\mathcal{O}\left(1/T\right)$ convergence rate of [7, 18].

Theorem 1. *Consider running Algorithm 1 for T iterations for minimizing (2.6) with $\eta_1 = 1/2(L+GL_{Q_1}+G_{Q_1}Ll_{Q_1})$ and $\eta_2 = 1/2(GL_{Q_2}+G_{Q_2}Ll_{Q_2})$. For any $t \in [T]$, define $\mathbf{G}^t := [\nabla_{\mathbf{x}^{t+1}} F_\lambda\left(\mathbf{x}^{t+1}, \mathbf{c}^t\right)^T, \nabla_{\mathbf{c}^{t+1}} F_\lambda\left(\mathbf{x}^{t+1}, \mathbf{c}^{t+1}\right)^T]^T$. Then, under **A**.1-A.5 and for $L_{\min} = \min\{\frac{1}{\eta_1}, \frac{1}{\eta_2}\}$,*

$L_{\max} = \max\{\frac{1}{\eta_1}, \frac{1}{\eta_2}\}$, *we have:*

$$\frac{1}{T}\sum_{t=0}^{T-1} \|\mathbf{G}^t\|_2^2 = \mathcal{O}\Big(\frac{L_{\max}^2\left(F_\lambda\left(\mathbf{x}^0, \mathbf{c}^0\right) - F_\lambda(\mathbf{x}^T, \mathbf{c}^T)\right)}{L_{\min} T}\Big).$$

Remark 2. *In Theorem 1, we see that gradient norm decays without any constant error terms. The convergence rate depends on Lipschitz smoothness constants of f and $f(\widetilde{Q}_{\mathbf{c}}(.))$ through $L_{\max}$ and $L_{\min}$. Choosing a smoother $\widetilde{Q}_{\mathbf{c}}(.)$ would speed up convergence; however, if chosen too small, this could result in an accuracy loss when hard-quantizing the parameters at the end of the algorithm.*

Remark 3 (Number of centers and convergence)**.** *The number of quantization levels m has a direct effect on convergence through the soft quantization function $\widetilde{Q}_{\mathbf{c}}(\mathbf{x})$ and Lipschitz constants. Note that as $m \to \infty$, we have $\widetilde{Q}_{\mathbf{c}}(\mathbf{x}) \to \mathbf{x}, \forall \mathbf{x}$. In this case, $\widetilde{Q}_{\mathbf{c}}(\mathbf{x})$ is 0-smooth and 1-Lipschitz w.r.t. all parameters. As a result, we would have $L_{\min} = L$ and $L_{\max} = 2L$. Note that the ratio $L_{\max}^2/L_{\min}$ increases as the quantization becomes more aggressive, and consequently, affects the convergence.*

2.4 Personalized Quantization for FL via Knowledge Distillation

We now consider the FL setting where we aim to learn quantized and personalized models for each client with different precision and model dimensions in heterogeneous data setting. Our proposed method QuPeD (Algorithm 4), utilizes the centralized scheme of Algorithm 1 locally at each client to minimize (2.3) over $(\{\mathbf{x}_i, \mathbf{c}_i\}_{i=1}^n, \mathbf{w})$. Here, $\mathbf{x}_i, \mathbf{c}_i$, denote the model parameters and the quantization vector (centers) for client i, and $\mathbf{w}$ denotes the global model that facilitates collaboration among clients which is encouraged through the knowledge distillation (KD) loss in the local objectives (2.2).

Description of the algorithm. Since clients perform local iterations, apart from maintaining $\mathbf{x}_i^t, \mathbf{c}_i^t$ at each client $i \in [n]$, it also maintains a model $\mathbf{w}_i^t$ that helps in utilizing other clients' data via collaboration. We call $\{\mathbf{w}_i^t\}_{i=1}^n$ local copies of the global model at

clients at time t. At each iteration t server samples a subset of all indices $\mathcal{K}_t \subseteq [n]$ with $K \leq n$ elements. Client i updates $\mathbf{w}_i^t$ in between communication rounds based on its local data and synchronizes that with the server which aggregates them to update the global model. Note that the local objective in (2.2) can be split into the weighted average of loss functions $(1-\lambda_p)(f_i(\mathbf{x}_i) + f_i(\widetilde{Q}_{\mathbf{c}_i}(\mathbf{x}_i))) + \lambda_p(f_i^{KD}(\mathbf{x}_i, \mathbf{w}_i) + f_i^{KD}(\widetilde{Q}_{\mathbf{c}_i}(\mathbf{x}_i), \mathbf{w}_i))$ and the term enforcing quantization $\lambda R(\mathbf{x}_i, \mathbf{c}_i)$. At any iteration t that is not a communication round (line 3), if $i \in \mathcal{K}_t$, client i first computes the gradient $\mathbf{g}_i^t$ of the loss function w.r.t. $\mathbf{x}_i^t$ (line 4) and then takes a gradient step followed by the proximal step using R (line 5) to update from $\mathbf{x}_i^t$ to $\mathbf{x}_i^{t+1}$. Then it computes the gradient $\mathbf{h}_i^t$ of the loss function w.r.t. $\mathbf{c}_i^t$ (line 6) and updates the centers followed by the proximal step (line 7). Finally, it updates $\mathbf{w}_i^t$ to $\mathbf{w}_i^{t+1}$ by taking a gradient step of the loss function at $\mathbf{w}_i^t$ (line 8). Thus, the local training of $\mathbf{x}_i^t, \mathbf{c}_i^t$ also incorporates knowledge from other clients' data through $\mathbf{w}_i^t$. When t is divisible by τ, sampled clients upload $\{\mathbf{w}_i^t\}$ to the server (line 10) which aggregates them (line 15) and broadcasts the updated global model (line 16) to all the clients. At the end of training, clients learn their personalized models $\{\mathbf{x}_i^T\}_{i=1}^n$ and quantization centers $\{\mathbf{c}_i^T\}_{i=1}^n$. Finally, client i quantizes $\mathbf{x}_i^T$ using $Q_{\mathbf{c}_i^T}$ (line 19).

Algorithm 2 QuPeD: Quantized Personalization via Distillation

Input: Regularization parameters λ, λ_p; synchronization gap τ; for client $i \in [n]$, initialize full precision personalized models $\mathbf{x}_i^0$, quantization centers $\mathbf{c}_i^0$, local model $\mathbf{w}_i^0$, learning rates $\eta_1^{(i)}, \eta_2^{(i)}, \eta_3$; quantization enforcing penalty function $R(\mathbf{x}, \mathbf{c})$; soft quantizer $\widetilde{Q}_{\mathbf{c}}(\mathbf{x})$; number of clients to be sampled K.

for $t = 0$ **to** $T-1$ **do**
  if τ divides t **then**
    On Server do:
    Choose a subset of clients $\mathcal{K}_t \subseteq [n]$ with size K
    Broadcast $\mathbf{w}^t$ to all **Clients**
    On Clients $i \in \mathcal{K}_t$ **to** n (in parallel) **do**:
    Receive $\mathbf{w}^t$ from **Server**; set $\mathbf{w}_i^t = \mathbf{w}^t$
  end if
  On Clients $i \in \mathcal{K}_t$ **to** n (in parallel) **do**:
  Compute $\mathbf{g}_i^t := (1-\lambda_p)(\nabla_{\mathbf{x}_i^t} f_i(\mathbf{x}_i^t) + \nabla_{\mathbf{x}_i^t} f_i(\widetilde{Q}_{\mathbf{c}_i^t}(\mathbf{x}_i^t))) + \lambda_p(\nabla_{\mathbf{x}_i^t} f_i^{KD}(\mathbf{x}_i^t, \mathbf{w}_i^t) + \nabla_{\mathbf{x}_i^t} f_i^{KD}(\widetilde{Q}_{\mathbf{c}_i^t}(\mathbf{x}_i^t), \mathbf{w}_i^t))$
  $\mathbf{x}_i^{t+1} = \text{prox}_{\eta_1^{(i)} \lambda R_{\mathbf{c}_i^t}}(\mathbf{x}_i^t - \eta_1^{(i)} \mathbf{g}_i^t)$
  Compute $\mathbf{h}_i^t := (1-\lambda_p)\nabla_{\mathbf{c}_i^t} f_i(\widetilde{Q}_{\mathbf{c}_i^t}(\mathbf{x}_i^{t+1})) + \lambda_p \nabla_{\mathbf{c}_i^t} f_i^{KD}(\widetilde{Q}_{\mathbf{c}_i^t}(\mathbf{x}_i^{t+1}), \mathbf{w}_i^t))$
  $\mathbf{c}_i^{t+1} = \text{prox}_{\eta_2^{(i)} \lambda R_{\mathbf{x}_i^{t+1}}}(\mathbf{c}_i^t - \eta_2^{(i)} \mathbf{h}_i^t)$
  $\mathbf{w}_i^{t+1} = \mathbf{w}_i^t - \eta_3 \lambda_p (\nabla_{\mathbf{w}_i^t} f_i^{KD}(\mathbf{x}_i^{t+1}, \mathbf{w}_i^t) + \nabla_{\mathbf{w}_i^t} f_i^{KD}(\widetilde{Q}_{\mathbf{c}_i^{t+1}}(\mathbf{x}_i^{t+1}), \mathbf{w}_i^t))$
  if τ divides $t+1$ **then**
    Clients send $\mathbf{w}_i^t$ to **Server**
    Server receives $\{\mathbf{w}_i^t\}$; computes $\mathbf{w}^{t+1} = \frac{1}{K} \sum_{i \in \mathcal{K}_t}^n \mathbf{w}_i^t$
  end if
end for
$\hat{\mathbf{x}}_i^T = Q_{\mathbf{c}_i^T}(\mathbf{x}_i^T)$ for all $i \in [n]$

Output: Quantized personalized models $\{\hat{\mathbf{x}}_i^T\}_{i=1}^n$

Assumptions. In addition to assumptions **A**.1 - A.5 (with **A**.3 and **A**.5 modified to have client specific gradient bounds $\{G^{(i)}, G_{Q_1}^{(i)}, G_{Q_2}^{(i)}\}$ as they have different model dimensions[7]), we assume:

A.6 *(Bounded diversity):* At any $t \in \{0, \cdots, T-1\}$ and any client $i \in [n]$, the variance of the local gradient (at client i) w.r.t. the global gradient is bounded, i.e., there exists $\kappa_i < \infty$, such that for every $\{\mathbf{x}_i^{t+1} \in \mathbb{R}^d, \mathbf{c}_i^{t+1} \in \mathbb{R}^{m_i} : i \in [n]\}$ and $\mathbf{w}^t \in \mathbb{R}^d$ generated according to Algorithm 4, we have: $\|\nabla_{\mathbf{w}^t} F_i(\mathbf{x}_i^{t+1}, \mathbf{c}_i^{t+1}, \mathbf{w}^t) - \frac{1}{n}\sum_{j=1}^n \nabla_{\mathbf{w}^t} F_j(\mathbf{x}_j^{t+1}, \mathbf{c}_j^{t+1}, \mathbf{w}^t)\|^2 \leq \kappa_i$. This assumption is equivalent to the bounded diversity assumption in [42, 48]; see Appendix A.1.

A.7 *(Smoothness of f^{KD}):* We assume $f_i^{KD}(\mathbf{x}, \mathbf{w})$ is L_{D_1}-smooth w.r.t. $\mathbf{x}$, L_{D_2}-smooth w.r.t. $\mathbf{w}$ for all $i \in [n]$; as a result it is L_D-smooth w.r.t. $[\mathbf{x}, \mathbf{w}]$ where $L_D = \max\{L_{D_1}, L_{D_2}\}$. Furthermore, we assume $f_i^{KD}(\widetilde{Q}_{\mathbf{c}}(\mathbf{x}), \mathbf{w})$ is L_{DQ_1}-smooth w.r.t. $\mathbf{x}$, L_{DQ_2}-smooth w.r.t. $\mathbf{c}$ and L_{DQ_3}-smooth w.r.t.$\mathbf{w}$ for all $i \in [n]$; as a result it is L_{DQ}-smooth w.r.t. $[\mathbf{x}, \mathbf{c}, \mathbf{w}]$ where $L_{DQ} = \max\{L_{DQ_1}, L_{DQ_2}, L_{DQ_3}\}$. In Appendix in [131] we discuss how this assumption can actually be inferred from previous assumptions.

Convergence result. In Theorem 2 we present the convergence result when there is full client participation, i.e. $K = n$, in Appendix in [131] we discuss the modification in convergence result under client sampling. The following result (proved inAppendix in [131]) achieves a rate of $\mathcal{O}\left(1/\sqrt{T}\right)$ for finding a stationary point within an error that depends on the data heterogeneity, matching result in [42]:

Theorem 2. *Under assumptions **A.1-A.7**, consider running Algorithm 4 for T iterations for minimizing (2.3) with $\tau \leq \sqrt{T}$, $\eta_1^{(i)} = 1/2(\lambda_p(2+L_{D_1}+L_{DQ_1})$ $+(1-\lambda_p)(L + G^{(i)}L_{Q_1} + G_{Q_1}^{(i)}Ll_{Q_1}))$, $\eta_2^{(i)} = 1/2(\lambda_p(1+L_{DQ_2})+(1-\lambda_p)(G^{(i)}L_{Q_2}+G_{Q_2}^{(i)}Ll_{Q_2}))$, and $\eta_3 = 1/4(\lambda_p L_w\sqrt{C_L}\sqrt{T})$ where $L_w = L_{D_2} + L_{DQ_3}$.*

[7]We keep smoothness constants to be the same across clients for notational simplicity, however, our result can easily be extended to that case.

Let $\mathbf{G}_i^t := [\nabla_{\mathbf{x}_i^{t+1}} F_i(\mathbf{x}_i^{t+1}, \mathbf{c}_i^t, \mathbf{w}^t)^T, \nabla_{\mathbf{c}_i^{t+1}} F_i(\mathbf{x}_i^{t+1}, \mathbf{c}_i^{t+1}, \mathbf{w}^t)^T, \nabla_{\mathbf{w}^t} F_i(\mathbf{x}_i^{t+1}, \mathbf{c}_i^{t+1}, \mathbf{w}^t)^T]^T$. *Then*

$$\frac{1}{T}\sum_{t=0}^{T-1}\frac{1}{n}\sum_{i=1}^{n}\left\|\mathbf{G}_i^t\right\|^2 = \mathcal{O}\left(\frac{\tau^2\overline{\kappa}+\overline{\Delta}_F}{\sqrt{T}} + \tau^2\overline{\kappa}\left(\frac{C_1}{T}+\frac{C_2}{T^{\frac{3}{2}}}\right)+\overline{\kappa}\right),$$

for some constants C_1, C_2, *where* $\overline{\Delta}_F = \frac{1}{n}\sum_{i=1}^{n}(L_{\max}^{(i)})^2\left(F_i(\mathbf{x}_i^0, \mathbf{c}_i^0, \mathbf{w}_i^0) - F_i(\mathbf{x}_i^T, \mathbf{c}_i^T, \mathbf{w}_i^T)\right)$, $\overline{\kappa} = \frac{1}{n}\sum_{i=1}^{n}(L_{\max}^{(i)})^2\kappa_i$, $C_L = 1 + \frac{\frac{1}{n}\sum_{i=1}^{n}(L_{\max}^{(i)})^2}{(\min_i\{L_{\max}^{(i)}\})^2}$, *and* $L_{\max}^{(i)}$ *depends on the smoothness parameters and the gradient bound at client* i *– the expression can be found in Appendix of [131].*

Remark 4 (Resource and data heterogeneity.)**.** *Firstly, our observation from Remark 3 holds here as well. Aggressive quantization has a scaling effect on all the terms through* $L_{\max}^{(i)}$. *Now the interesting question is: how does having different model structures across clients affect the convergence rate of Theorem 2? Note that in Assumptions* ***A****.3, A.5, we assume clients have different client-specific gradient bounds; this results in client specific* $L_{\max}^{(i)}$, *and consequently* $\overline{\kappa}$, *which couples resource and data heterogeneity across clients. Here we make an important* first *observation regarding the coupled effect of data and resource heterogeneity on the convergence rate. Suppose data distributions are fixed across clients (i.e.,* κ_i*'s are fixed) and we need to choose models for each client in the federated ecosystem. Then, for a fast convergence, for the clients that have local data that is not a representative of the general distribution (large* κ_i*), it is critical to choose models with small smoothness parameter (e.g., choosing a less aggressive quantization); whereas, clients with data that is representative of the overall data distribution (small* κ_i*) can tolerate having a less smooth model.*

2.5 Experiments

In this section, we first briefly compare numerical results for our underlying model quantization scheme (Algorithm 1) in a centralized case against related works [7, 177]. For a major part of the section, we then compare QuPeD (Algorithm 4) against other personalization schemes [42, 48, 152] for data heterogeneous clients and demonstrate its effectiveness in resource heterogeneous environments. Additional experimental results on a language task

Method	ResNet-20	ResNet-32
Full Precision (FP)	92.05	92.95
ProxQuant [7] (1bit)	90.69	91.55
BinaryRelax [177] (1bit)	87.82	90.65
Algorithm1 (1bit)	91.17	92.20
Algorithm1 (2bits)	91.45	92.47

Figure 2.1 Test accuracy (in %) on CIFAR-10.

are provided in Appendix of [131]. Implementation details, and further experiments with modifications to settings in this section, are also in Appendix of [131].

Centralized Training: We compare Algorithm 1 with [7, 177] for ResNet-20 and ResNet-32 [66] models trained on CIFAR-10 [91] dataset.

While both [7, 177] are limited to binary quantization, [7] can be seen as a specific case of our centralized method where the centers do not get updated. From Table 2.1, we see that updating centers (Algorithm 1) significantly improves the performance (0.48% increase in test accuracy). Allowing quantization with 2 bits instead of 1bit for Algorithm 1 further increases the test accuracy.

Personalized Training: We consider an image classification task on FEMNIST [21] and CIFAR-10 [91] datasets. We consider two CNN architectures: (i) CNN1 (used in [116]): has 2 convolutional and 3 fully connected layers, (ii) CNN2: this is CNN1 with an additional convolutional layer with 32 filters and 5×5 kernel size. For CIFAR-10 we choose a batch size of 25. For FEMNIST, we choose variable batch sizes to have 60 iterations for all clients per epoch. We train each algorithm for 250 epochs on CIFAR-10 and 30 epochs on FEMNIST. For quantized training, as standard practice [136], we let the first and last layers of networks to be in full precision. We use last 50 epochs on CIFAR-10, and 5 epochs on FEMNIST for the fine-tuning phase.

Table 2.1 Test accuracy (in %) for CNN1 model at all clients.[7]

Method	FEMNIST	CIFAR-10
FedAvg (FP)	94.92 ± 0.04	61.40 ± 0.29
Local Training (FP)	94.86 ± 0.93	71.57 ± 0.28
Local Training (2 Bits)	93.95 ± 0.23	70.87 ± 0.15
Local Training (1 Bit)	93.00 ± 0.50	69.05 ± 0.13
QuPeD (FP)	$\mathbf{97.31} \pm 0.12$	$\mathbf{75.06} \pm 0.40$
QuPeD (2 Bits)	96.73 ± 0.27	74.58 ± 0.44
QuPeD (1 Bit)	95.15 ± 0.21	71.20 ± 0.33
QuPeL (2 Bits)	96.10 ± 0.14	73.52 ± 0.51
QuPeL (1 Bits)	94.06 ± 0.28	71.01 ± 0.32
pFedMe (FP) [42]	96.60 ± 0.37	73.66 ± 0.65
Per-FedAvg (FP) [48]	97.16 ± 0.21	74.15 ± 0.41
Federated ML (FP) [152]	96.32 ± 0.32	74.34 ± 0.30

Data Heterogeneity (DH): We consider $n = 50$ clients for CIFAR-10 and $n = 66$ for FEMNIST. To simulate data heterogeneity on CIFAR-10, similar to [116], we allow each client to have access to data samples from only 4 randomly chosen classes. Thus, each client has 1000 training samples and 200 test samples. On FEMNIST, we use a subset of 198 writers from the dataset and distribute the data so that each client has access to data samples written by 3 randomly chosen writers. The number of training samples per client varies between 203-336 and test samples per client varies between 25-40. Test samples are sampled from the same class/writer that training samples are sampled from, in parallel with previous works in heterogeneous FL.

Resource Heterogeneity (RH): To simulate resource heterogeneity for QuPeD, we consider

[7]Here QuPeD (FP) corresponds to changing alternating proximal gradient updates with SGD update on model parameters in Algorithm 4, Local Training (FP) corresponds to SGD updates without communication.

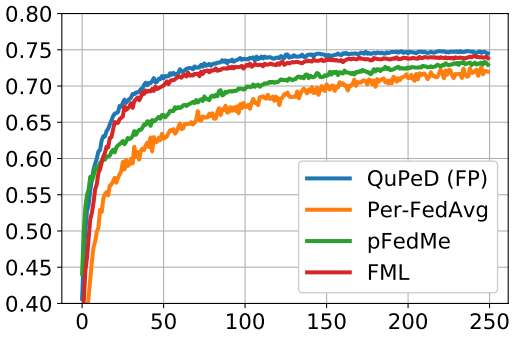


Figure 2.2 Test Acc. vs epoch (CIFAR-10)

Table 2.2 Test accuracy (in %) for CNN1 model at all clients, with client sampling.

Method	MNIST ($\frac{K}{n} = 0.1$)	FEMNIST($\frac{K}{n} = \frac{1}{3}$)
FedAvg (FP)	92.87 ± 0.05	91.30 ± 0.43
QuPeD (FP)	$\mathbf{98.17} \pm 0.32$	$\mathbf{94.93} \pm 0.25$
QuPeD (2 Bits)	98.01 ± 0.15	94.56 ± 0.18
QuPeD (1 Bit)	97.58 ± 0.23	92.52 ± 0.64
pFedMe (FP) [42]	97.79 ± 0.03	93.70 ± 0.39
Per-FedAvg (FP) [48]	95.80 ± 0.29	92.10 ± 0.22
Federated ML (FP) [152]	98.03 ± 0.31	92.73 ± 0.36

4 settings: (i) half of the clients have CNN1 in full precision (FP) and the other half CNN2 in FP, (ii) half of the clients have CNN1 in 2 bits and the other half CNN2 in FP, (iii) half of the clients have CNN1 in 2 bits and the other in FP, (iv) half of the clients have CNN1 in 2 bits and the other half CNN2 in 2 bits.

Results (DH): We compare QuPeD against FedAvg [116], local training of clients (without any collaboration), and personalized FL methods: pFedMe [42], Per-FedAvg [48], Federated Mutual Learning [152], and QuPeL (Footnote 5). For all methods, if applicable, we set $\tau = 10$ local iterations, use learning rate decay 0.99 and use weight decay of 10^{-4}; we fine tune the initial learning rate for each method independently, see Appendix of [131] for details. The results are provided in Table 2.1 with full client participation ($K = n$), plotted in Figure 2.2

Table 2.3 Test accuracy (in %) on CIFAR-10 for heterogeneous resource distribution among clients.

Resource Heterogeneity	CIFAR-10	
	Local Training	QuPeD
CNN1(FP) + CNN2(FP)	72.81 ± 0.03	75.50 ± 0.25
CNN1(2 Bits)+CNN2(FP)	72.42 ± 0.17	75.08 ± 0.18
CNN1(2 Bits)+CNN1(FP)	71.23 ± 0.08	74.84 ± 0.30
CNN1(2 Bits)+CNN2(2 Bits)	72.15 ± 0.47	74.64 ± 0.27

Table 2.4 Test accuracy (in %) on FEMNIST for heterogeneous resource distribution among clients.

Resource Heterogeneity	FEMNIST	
	Local Training	QuPeD
CNN1(FP) + CNN2(FP)	93.41 ± 0.82	97.44 ± 0.14
CNN1(2 Bits)+CNN2(FP)	92.70 ± 1.09	97.01 ± 0.05
CNN1(2 Bits)+CNN1(FP)	93.56 ± 0.38	96.96 ± 0.15
CNN1(2 Bits)+CNN2(2 Bits)	91.11 ± 0.23	96.64 ± 0.31

for CIFAR-10, and in Table 2.2 with client sampling where we state average results over 3 runs; all clients train CNN1 (see Appendix in [131] for CNN2) and quantization values are indicated in parenthesis. Thus, we only consider model personalization for data heterogeneity. In Table 2.1, we observe that full precision QuPeD consistently outperforms all other methods for both datasets. Furthermore, we observe QuPeD with 2-bit quantization is the second best performing method on CIFAR-10 (after QuPeD (FP)) and third best performing method on FEMNIST despite the loss due to quantization. Hence, QuPeD is highly effective for quantized training in data heterogeneous settings. Since QuPeD outperforms QuPeL, we can also (empirically) claim that considering KD loss to encourage collaboration is superior to ℓ_2

distance loss. Lastly, we observe from Table 2.2 that QuPeD continues to outperform other methods under client sampling.

Results (DH+RH): We now discuss personalized FL setting with both data and resource heterogeneity across clients. Note that since FedAvg, pFedMe, and Per-FedAvg cannot work in settings where clients have different model dimensions, we only provide comparisons of QuPeD with local training (no collaboration) to demonstrate its effectiveness. The results are given in Table 2.4. We observe that QuPeD (collaborative training) significantly outperforms local training in all cases (about 3.5% or higher on FEMNIST and 2.5% or higher on CIFAR-10) and works remarkably well even in cases where clients have quantized models without any significant loss in performance.

2.6 Discussion

In this work, we introduced QuPeD: an algorithm for training of quantized and personalized models in heterogeneous Federated Learning (FL) settings. Our proposed scheme tackles two of the major problems in practical FL applications: resource and data heterogeneity among participating clients. QuPeD allows clients to learn personalized models based on their local data distributions while simultaneously allowing these models to be different in architecture and weight precision. This is unlike existing personalized FL algorithms in literature which only consider the data heterogeneity aspect of FL. Our theoretical results on the convergence rates recovers previous results, which are in simpler settings, when the clients have identical resources, i.e., same model size and precision. When clients have distinct resources, our result ties convergence speed to combination of resource and data heterogeneity revealing additional insights. Our experimental results demonstrate that QuPeD delivers superior empirical performance even under aggressive quantization.

Convergence results stated in Theorems 1, 2, use bounded gradients assumptions **A**.3, A.5 for non-convex objectives considered in this paper. Incorporating recent progress on

weakening gradient assumptions in convergence analyses for FL (*e.g.,* [82, 155] and references therein) into our framework is part of future work. We remark that our current assumptions are fairly standard and applicable to our new optimization formulation to tackle data and resource heterogeneity of the clients, which is the principal focus and contribution of our work. Simulations consisting of tens of thousands (or more) of clients and lower client sampling ratios are also crucial, and are left for potential future investigations that can shed further light on our methods at massive scale.

Societal Impact: This paper considers collaborative personalization in learning, which can significantly improve learning performance. However, such personalization is only as good as the data used for training, and if not used properly could lead to information bubbles and disjointed views for each client, *e.g.,* clustering methods that put together clients with similar data statistics. We ameliorate that by not clustering similar data and models, but this is an aspect that might need more examination for social consequences. We explicitly consider diverse resources in our designs, potentially enabling clients with fewer resources (devices with less capabilities) to benefit from richer resources; potentially a positive impact in using resources more equitably.

CHAPTER 3

A Statistical View of Personalized Learning

A distinguishing characteristic of federated learning is that the (local) client data could have statistical heterogeneity. This heterogeneity has motivated the design of personalized learning, where individual (personalized) models are trained, through collaboration. There have been various personalization methods proposed in literature, with seemingly very different forms and methods ranging from use of a single global model for local regularization and model interpolation, to use of multiple global models for personalized clustering, etc. In this work, we begin with a statistical framework that unifies several different algorithms as well as suggest new algorithms. We apply our framework to personalized estimation, and connect it to the classical empirical Bayes' methodology. We develop novel private personalized estimation under this framework. We then use our statistical framework to propose new personalized learning algorithms, including `AdaPeD` based on information-geometry regularization, which numerically outperforms several known algorithms. We develop privacy for personalized learning methods with guarantees for user-level privacy and composition. We numerically evaluate the performance as well as the privacy for both the estimation and learning problems, demonstrating the advantages of our proposed methods.

3.1 Introduction

The federated learning (FL) paradigm has had huge recent success both in industry and academia [80, 116], as it enables to leverage data available in dispersed devices for learning while maintaining data privacy. Yet, it was recently realized that for some applications, due

to the statistical heterogeneity of local data, a single global learning model may perform poorly for individual clients. This motivated the need for personalized learning achieved through collaboration, and there have been a plethora of personalized models proposed in the literature as well [1, 40, 42, 48, 73, 100, 114, 131, 185]. However, the proposed approaches appear to use very different forms and methods, and there is a lack of understanding of an underlying fundamental statistical framework. Such a statistical framework could help develop theoretical bounds for performance, suggest new algorithms as well as perhaps give grounding to known methods. Our work addresses this gap.

In particular, we consider the fundamental question of how one can use collaboration to help personalized learning and estimation for users who have limited data that they want to keep private. Our proposed framework is founded on the requirement not only of personalization but also privacy, as maintaining local data privacy is what makes the federated learning framework attractive - and thus any algorithm that aims to be impactful needs to also give formal privacy guarantees. The goal of this paper is to develop a statistical framework that leads to new algorithms with provable privacy guarantees, and performance bounds. Our main contributions are (i) Development of a statistical framework for federated personalized estimation and learning (ii) Theoretical bounds and novel algorithms for private personalized estimation (iii) Design and privacy analysis of new private personalized learning algorithms; as elaborated below. Omitted proofs/details are in appendices.

• **Statistical framework:** We connect this problem to the classical empirical Bayes' method, pioneered by [77,145,160], which proposed a hierarchical statistical model [52]. This is modeled by an *unknown* population distribution $\mathbb{P}$ from which local parameters $\{\boldsymbol{\theta}_i\}$ are generated, which in turn generate the local data through the distribution $\mathbb{Q}(\boldsymbol{\theta}_i)$. Despite the large literature on this topic, especially in the context of statistical estimation, creating a framework for FL poses new challenges. In contrast to classical empirical Bayes' estimation, we introduce a distributed setting and develop a framework that allows information (communication and

privacy) constraints[1]. This framework enables us to develop statistical performance bounds as well as suggests (private) personalized federated estimation algorithms. Moreover, we develop our framework beyond estimation, for (supervised) *distributed learning*, where clients want to build *local* predictive models with limited local (labeled) samples; we develop this framework in Section 3.3, which leads to new (private) personalized learning algorithms.

• **Private personalized estimation:** Our goal is to estimate individual (local) parameters, when each user has very limited (heterogeneous) data. Such a scenario motivates federated estimation of individual parameters, *privately*. More precisely, the users observe data generated by an unknown distribution parametrized by their individual (unknown) local parameters $\boldsymbol{\theta}_i$, and want to estimate their local parameters $\boldsymbol{\theta}_i$ leveraging very limited local data; see Section 3.2 for more details. For the hierarchical statistical model, classical results have shown that one can enhance the estimate of individual parameters based on the observations of a population of samples, despite having *independently* generated parameters from an unknown population distributions. However, this has not been studied for the distributed case, with privacy and communication constraints, which we do (see Theorem 4 for the Gaussian case and Theorem 6 for the Bernoulli case, and also for mixture population models in Appendix B.5 We estimate the (parametrized) population distribution under these privacy and communication constraints and use this as an empirical prior for local estimation. The effective amplification of local samples through collaboration, in Section 3.2, gives us theoretical insight about when collaboration is most useful, under privacy and/or communication constraints. Our results suggest how to optimally balance estimates from local and population models. We also numerically evaluate these methods, including application to polling data (see Section 7.5 and Appendices) to show advantages of such collaborative estimation compared to local methods.

• **Private personalized learning:** The goal here is to obtain individual learning models capable of predicting labels with limited local data in a supervised learning setting. This

[1]The homogeneous case for distributed estimation is well-studied; see [187] and references.

is the use case for federated learning with privacy guarantees. It is intimately related to the estimation problem with distinctions including (i) to design good label predictors rather than just estimate local parameters (ii) the focus on iterative methods for optimization, requiring strong compositional privacy guarantees. Therefore, the statistical formulation for learning has a similar flavor to that in estimation, where there is a population model for local (parametrized) statistics for labeled data; see Section 3.3 for more details. We develop several algorithms, including `AdaPeD` (in Section 3.3.2), `AdaMix` (in Section 3.3.1), and `DP-AdaPeD` (in Section 3.3.3), inspired by the statistical framework. `AdaPeD` uses information divergence constraints along with adaptive weighting of local models and population models. By operating in probability (rather than Euclidean) space, using information-geometry (divergence), enables `AdaPeD` to operate with different local model sizes and architectures, giving it greater flexibility than existing methods. We integrate it with *user-level* privacy to develop `DP-AdaPeD`, with strong compositional privacy guarantees (Theorem 7). `AdaMix` is inspired by mixture population distributions, which adaptively weighs multiple global models and combines it with local data for personalization. We numerically evaluate these algorithms for synthetic and real data in Section 7.5.

Related Work. Our work can be seen in the intersection of personalized learning, estimation, and privacy. Below we give a brief description of related work; a more detailed comparison which connects our framework to other personalized algorithms is given in Appendix C.1.

Personalized FL: Recent work adopted different approaches for learning personalized models, which can be explained by our statistical framework for suitable choices of population distributions as explained in Appendix of C.1. These include, *meta-learning based methods* [1, 48, 87]; *regularization* [40, 65, 114]; *clustered FL* [56, 114, 157, 185] [115]; *using knowledge distillation* [102, 131]; *multi-task Learning* [42, 65, 157, 168, 183]; and *using common representations* [34, 44, 140, 164] and references therein. Our work enables not just a unified view of these methods, but suggests new algorithms developed in this paper, along with privacy guarantees.

After the conclusion of our work [127], we found two concurrent and independent works [27, 90] that use a hierarchical Bayes approach to construct personalized learning algorithms, and are closest to our statistical framework. [90] is based on using a MCMC method[2] to estimate a population distribution; such methods could be computationally intensive (see the discussion in [17]; [27] considers the unimodal Gaussian prior, and effectively does what the classical empirical Bayes suggests (see also Theorem 3). None of these works consider privacy, which we do both for estimation and learning algorithms (see Theorems 4, 6, Appendix B.10, and for `DP-AdaPeD` in Theorem 7). Note that MCMC methods could have detrimental privacy implications. Also, they do not include information-geometric techniques (like our `AdaPeD`) or methods inspired by mixture distributions (*e.g.,* `AdaMix`).

Privacy for Personalized Learning. There has been a lot of work in privacy for FL when the goal is to learn a *single* global model (see [60] and references therein); though there are fewer papers that address user-level privacy [55, 95, 105]. There has been more recent work on applying these ideas to learn personalized models [53, 58, 73, 75, 99]. These are for specific algorithms/models, *e.g.,* [75] focuses on the common representation model for linear regression described earlier or on item-level privacy [73, 99]. We believe that `DP-AdaPeD` proposed in this paper is among the first user-level private personalized learning algorithms with compositional guarantees, applicable to general deep learning architectures.

3.2 Personalized Estimation

We consider a client-server architecture, where there are m clients. Let $\mathbb{P}(\Gamma)$ denote a global population distribution that is parameterized by an unknown Γ and let $\boldsymbol{\theta}_1, \ldots, \boldsymbol{\theta}_m$ are sampled i.i.d. from $\mathbb{P}(\Gamma)$ and are unknown to the clients. Client i is given a dataset $X_i := (X_{i1}, \ldots, X_{in})$, where $X_{ij}, j \in [n]$ are sampled i.i.d. from some distribution $\mathbb{Q}(\boldsymbol{\theta}_i)$, parameterized by $\boldsymbol{\theta}_i \in \mathbb{R}^d$. Note that heterogeneity in clients' datasets is induced through the

[2] In our understanding their numerics seem to be restricted to a unimodal Gaussian population model.

variance in $\mathbb{P}(\Gamma)$, and if the variance of $\mathbb{P}(\Gamma)$ is zero, then all clients observe i.i.d. datasets sampled from the *same* underlying distribution.

The goal at client i for all $i \in [m]$ is to estimate $\boldsymbol{\theta}_i$ through the help of the server. We focus on one-round communication schemes, where client j applies a (potentially randomized) mechanism q on its dataset X_j and sends $q_j := q(X_j)$ to the server, who aggregates the received messages, which is denoted by $\mathsf{Agg}(q_1, \ldots, q_m)$, and broadcasts that to all clients. Based on $(X_i, \mathsf{Agg}(q_1, \ldots, q_m))$, client i outputs an estimate $\widehat{\boldsymbol{\theta}}_i$ of $\boldsymbol{\theta}_i$. We measure the performance of $\widehat{\boldsymbol{\theta}}_i$ through the Bayesian risk for mean squared error (MSE), as defined below (where $\mathbb{P}$ is the true prior distribution with associated density π, $\boldsymbol{\theta}_i \sim \mathbb{P}$ is the true local parameter, and $\widehat{\boldsymbol{\theta}}_i = \widehat{\boldsymbol{\theta}}(X_i, \mathsf{Agg}(q_1, \ldots, q_m))$ is the estimator):

$$\mathbb{E}_{\boldsymbol{\theta}_i \sim \mathbb{P}} \mathbb{E}_{\widehat{\boldsymbol{\theta}}_i, q, X_1, \ldots, X_m} \|\widehat{\boldsymbol{\theta}}_i - \boldsymbol{\theta}_i\|^2 = \int \mathbb{E}_{\widehat{\boldsymbol{\theta}}_i, q, X_1, \ldots, X_m} \|\widehat{\boldsymbol{\theta}}_i - \boldsymbol{\theta}_i\|^2 \pi(\boldsymbol{\theta}_i) d\boldsymbol{\theta}_i. \tag{3.1}$$

The above statistical framework can model many different scenarios, and we will study in detail three settings: Gaussian and Bernoulli models (Sections 3.2.1, 3.2.2 below), and Mixture model (Appendix B.5).

3.2.1 Gaussian Model

In the Gaussian setting, $\mathbb{P}(\Gamma) = \mathcal{N}(\boldsymbol{\mu}, \sigma_\theta^2 \mathbb{I}_d)$ and $\mathbb{Q}(\boldsymbol{\theta}_i) = \mathcal{N}(\boldsymbol{\theta}_i, \sigma_x^2 \mathbb{I}_d)$ for all $i \in [m]$, which implies that $\boldsymbol{\theta}_1, \ldots, \boldsymbol{\theta}_m \sim \mathcal{N}(\boldsymbol{\mu}, \sigma_\theta^2 \mathbb{I}_d)$ i.i.d. and $X_{i1}, \ldots, X_{in} \sim \mathcal{N}(\boldsymbol{\theta}_i, \sigma_x^2 \mathbb{I}_d)$ i.i.d. for $i \in [m]$. Here, $\sigma_\theta \geq 0, \sigma_x > 0$ are known, and $\boldsymbol{\mu}, \boldsymbol{\theta}_1, \ldots, \boldsymbol{\theta}_m$ are unknown. For the case of a single local sample this is identical to the classical James-Stein estimator [77]; Theorem 3 does a simple extension for multiple local samples and is actually a stepping stone for the information constrained estimation result of Theorem 4. Omitted proofs/details are provided in Appendix B.

Our proposed estimator. Since there is no distribution on $\boldsymbol{\mu}$, and given $\boldsymbol{\mu}$, we know the distribution of $\boldsymbol{\theta}_i$'s, and subsequently, of X_{ij}'s. So, we consider the maximum likelihood

estimator:

$$\widehat{\boldsymbol{\theta}}_1, \ldots, \widehat{\boldsymbol{\theta}}_m, \widehat{\boldsymbol{\mu}} := \underset{\boldsymbol{\theta}_1, \ldots, \boldsymbol{\theta}_m, \boldsymbol{\mu}}{\arg\max}\, p_{\{\boldsymbol{\theta}_i, X_i\}|\boldsymbol{\mu}}\left(\boldsymbol{\theta}_1, \ldots, \boldsymbol{\theta}_m, X_1, \ldots, X_m | \boldsymbol{\mu}\right) \tag{3.2}$$

Theorem 3. *Solving (3.2) yields the following closed form expressions for $\widehat{\boldsymbol{\mu}}$ and $\widehat{\boldsymbol{\theta}}_1, \ldots, \widehat{\boldsymbol{\theta}}_m$:*

$$\widehat{\boldsymbol{\mu}} = \frac{1}{m}\sum_{i=1}^{m} \overline{X}_i \qquad \text{and} \qquad \widehat{\boldsymbol{\theta}}_i = a\overline{X}_i + (1-a)\widehat{\boldsymbol{\mu}}, \text{ for } i \in [m], \qquad \text{where } a = \frac{\sigma_\theta^2}{\sigma_\theta^2 + \sigma_x^2/n}. \tag{3.3}$$

The above estimator achieves the MSE: $\mathbb{E}_{\boldsymbol{\theta}_i, X_1, \ldots, X_m} \|\widehat{\boldsymbol{\theta}}_i - \boldsymbol{\theta}_i\|^2 \leq \frac{d\sigma_x^2}{n}\left(\frac{1-a}{m} + a\right)$.

It follows that the mechanism q and the aggregation function Agg for the estimators in (3.3) (as described in (3.1)) are just the average functions, where client i sends $q_i = q(X_i) := \overline{X}_i = \frac{1}{n}\sum_{j=1}^{n} X_{ij}$ to the server, and the server sends $\widehat{\boldsymbol{\mu}} := \mathsf{Agg}(q_1, \ldots, q_m) = \frac{1}{m}\sum_{i=1}^{m} q_i$ back to the clients.

Remark 5 (Personalized estimate vs. local estimate)**.** *When $\sigma_\theta \to 0$, then $a \to 0$, which implies that $\widehat{\boldsymbol{\theta}}_i \to \widehat{\boldsymbol{\mu}}$ and MSE $\to d\sigma_x^2/mn$. Otherwise, when σ_θ^2 is large in comparison to σ_x^2/n or $n \to \infty$, then $a \to 1$, which implies that $\widehat{\boldsymbol{\theta}}_i \to \overline{X}_i$ and MSE $\to d\sigma_x^2/n$. These conform to the facts that (i) when there is no heterogeneity, then the global average is the best estimator, and (ii) when heterogeneity is not small, and we have a lot of local samples, then the local average is the best estimator. Observe that the multiplicative gap between the MSE of the proposed personalized estimator and the MSE of the local estimator (based on local data only, which gives an MSE of $d\sigma_x^2/n$) is given by $(\frac{1-a}{m} + a) \leq 1$ that proves the superiority of the personalized model over the local model, which is equal to $1/m$ when $\sigma_\theta = 0$ and equal to 0.01 when $m = 10^4, n = 100$ and $\sigma_x^2 = 10, \sigma_\theta^2 = 10^{-3}$, for example.*

Remark 6 (Optimality of our personalized estimator)**.** *In Appendix B.9, we show the minimax lower bound: $\inf_{\widehat{\boldsymbol{\theta}}} \sup_{\boldsymbol{\theta} \in \Theta} \mathbb{E}_{X \sim \mathcal{N}(\boldsymbol{\theta}, \sigma_x^2)} \|\widehat{\boldsymbol{\theta}}(X) - \boldsymbol{\theta}\|^2 \geq \frac{d\sigma_x^2}{n}\left(\frac{1-a}{m} + a\right)$, which exactly matches the upper bound on the MSE in Theorem 3, thus establishes the optimality our personalized estimator in (3.3).*

Privacy and communication constraints. Observe that the scheme presented above does not protect privacy of clients' data and messages from the clients to the server can be made communication-efficient. These could be achieved by employing specific mechanisms q at clients: For privacy, we can take a differentially-private q, and for communication-efficiency, we can take q to be a quantizer. Inspired by the scheme presented above, here we consider q to be a function $q : \mathbb{R}^d \to \mathcal{Y}$, that takes the average of n data points as its input, and the aggregator function Agg to be the average function. Define $\widehat{\boldsymbol{\mu}}_q := \frac{1}{m}\sum_{i=1}^m q(\overline{X}_i)$ and consider the following personalized estimator for the i-th client:

$$\widehat{\boldsymbol{\theta}}_i = a\overline{X}_i + (1-a)\widehat{\boldsymbol{\mu}}_q, \qquad \text{for some } a \in [0,1]. \tag{3.4}$$

Theorem 4. *Suppose for all $\mathbf{x} \in \mathbb{R}^d$, q satisfies $\mathbb{E}[q(\mathbf{x})] = \mathbf{x}$ and $\mathbb{E}\|q(\mathbf{x}) - \mathbf{x}\|^2 \le d\sigma_q^2$ for some finite σ_q. Then the personalized estimator in (3.4) has MSE:*

$$\mathbb{E}_{\boldsymbol{\theta}_i, q, X_1, \ldots, X_m}\|\widehat{\boldsymbol{\theta}}_i - \boldsymbol{\theta}_i\|^2 \le \frac{d\sigma_x^2}{n}\left(\frac{1-a}{m} + a\right) \qquad \text{where} \qquad a = \frac{\sigma_\theta^2 + \sigma_q^2/m-1}{\sigma_\theta^2 + \sigma_q^2/m-1 + \sigma_x^2/n}. \tag{3.5}$$

Furthermore, assuming $\boldsymbol{\mu} \in [-r, r]^d$ for some constant r (but $\boldsymbol{\mu}$ is unknown), we have:

1. *Communication efficiency: For any $k \in \mathbb{N}$, there is a q whose output can be represented using k-bits (i.e., q is a quantizer) that achieves the MSE in (3.5) with probability at least $1 - 2/mn$ and with $\sigma_q = \frac{b}{(2^k-1)}$, where $b = r + \sigma_\theta\sqrt{\log(m^2n)} + \frac{\sigma_x}{\sqrt{n}}\sqrt{\log(m^2n)}$.*

2. *Privacy: For any $\epsilon_0 \in (0,1), \delta > 0$, there is a q that is user-level (ϵ_0, δ)-locally differentially private, that achieves the MSE in (3.5) with probability at least $1 - 2/mn$ and with $\sigma_q = \frac{b}{\epsilon_0}\sqrt{8\log(2/\delta)}$, where $b = r + \sigma_\theta\sqrt{\log(m^2n)} + \frac{\sigma_x}{\sqrt{n}}\sqrt{\log(m^2n)}$.*

3.2.2 Bernoulli Model

For the Bernoulli model, $\mathbb{P}$ is supported on $[0,1]$, and $p_1, \ldots, p_m$ are sampled i.i.d. from $\mathbb{P}$, and client i is given n i.i.d. samples $X_{i1}, \ldots, X_{in} \sim \mathsf{Bern}(p_i)$. This setting has been studied in ([163, 169]) for estimating $\mathbb{P}$, whereas, our goal is to estimate individual parameter p_i

at client i using the information from other clients. In order to derive a closed form MSE result, we assume that $\mathbb{P}$ is the Beta distribution.[3] Here, $\Gamma = (\alpha, \beta), p_1, \ldots, p_m$ are unknown, and client i's goal is to estimate p_i such that the Bayesian risk $\mathbb{E}_{p_i \sim \pi} \mathbb{E}_{\widehat{p}_i, X_1, \ldots, X_m} (\widehat{p}_i - p_i)^2$ is minimized, where π denotes the density of the Beta distribution. Omitted proofs/details are provided in Appendix B.

Analogous to the Gaussian case, we can show that if α, β are known, then the posterior mean estimator has a closed form expression: $\widehat{p}_i = a\overline{X}_i + (1-a)\frac{\alpha}{\alpha+\beta}$, where $a = {}^{n}/_{\alpha+\beta+n}$ and ${}^{\alpha}/_{(\alpha+\beta)}$ is the mean of the beta distribution. When α, β are unknown, inspired by the above discussion, a natural approach would be to estimate the global mean $\mu = {}^{\alpha}/_{(\alpha+\beta)}$ and the weight $a = {}^{n}/_{(\alpha+\beta+n)}$, and use that in the above estimator. Note that, for a we need to estimate $\alpha + \beta$, which is equal to ${}^{\mu(1-\mu)}/_{\sigma^2} - 1$, where $\sigma^2 = {}^{\alpha\beta}/_{(\alpha+\beta)^2(\alpha+\beta+1)}$ is the variance of the beta distribution. Therefore, it is enough to estimate μ and σ^2 for the personalized estimators $\{\widehat{p}_i\}$. In order to make calculations simpler, instead of making one estimate of μ, σ^2 for all clients, we let each client make its own estimate of μ, σ^2 (without using their own data) as: $\widehat{\mu}_i = \frac{1}{m-1}\sum_{l \neq i} \overline{X}_l$ and $\widehat{\sigma}_i^2 = \frac{1}{m-2}\sum_{l \neq i}(\overline{X}_l - \widehat{\mu}_l)^2$,[4] and then define the local weight as $\widehat{a}_i = \frac{n}{{}^{\widehat{\mu}_i(1-\widehat{\mu}_i)}/_{\widehat{\sigma}_i^2} - 1 + n}$. Now, client i uses the following personalized estimator:

$$\widehat{p}_i = \widehat{a}_i \overline{X}_i + (1 - \widehat{a}_i)\widehat{\mu}_i. \tag{3.6}$$

Theorem 5. *With probability at least* $1 - \frac{1}{mn}$*, the MSE of the personalized estimator in* (3.6) *is given by:* $\mathbb{E}_{p_i \sim \pi} \mathbb{E}_{X_1, \ldots, X_m} (\widehat{p}_i - p_i)^2 \leq \mathbb{E}[\widehat{a}_i^2] \Big(\frac{\alpha\beta}{n(\alpha+\beta)(\alpha+\beta+1)} \Big) + \mathbb{E}[(1-\widehat{a}_i)^2] \Big(\frac{\alpha\beta}{(\alpha+\beta)^2(\alpha+\beta+1)} + \frac{3\log(4m^2n)}{m-1} \Big)$.

Remark 7. *When* $n \to \infty$*, then* $\widehat{a}_i \to 1$*, which implies that MSE tends to the MSE of the local estimator* $\overline{X}_i$*, which means if local samples are abundant, collaboration does not help much. When* $\sigma^2 = {}^{\alpha\beta}/_{(\alpha+\beta)^2(\alpha+\beta+1)} \to 0$*, i.e. there is very small heterogeneity in the system,*

[3] Beta distribution has a density $\mathsf{Beta}(\alpha, \beta) = \frac{1}{B(\alpha,\beta)} x^{\alpha-1}(1-x)^{\beta-1}$ is defined for $\alpha, \beta > 0$ and $x \in [0, 1]$, where $B(\alpha, \beta)$ is a normalizing constant. Its mean is $\frac{\alpha}{\alpha+\beta}$ and the variance is $\frac{\alpha\beta}{(\alpha+\beta)^2(\alpha+\beta+1)}$.

[4] Upon receiving $\{\overline{X}_i\}$ from all clients, the server can compute $\{\widehat{\mu}_i, \widehat{\sigma}_i^2\}$ and sends $(\widehat{\mu}_i, \widehat{\sigma}_i^2)$ to the i-th client.

then $\widehat{a}_i \to 0$, which implies that MSE tends to the error due to moment estimation (the last term in the MSE in Theorem 5).

Privacy constraints. For any privacy parameter $\epsilon_0 > 0$ and input $x \in [0,1]$, define $q^{\mathsf{priv}} : [0,1] \to \mathbb{R}$:

$$q^{\mathsf{priv}}(x) = \begin{cases} \frac{-1}{e^{\epsilon_0}-1} & \text{w.p. } \frac{e^{\epsilon_0}}{e^{\epsilon_0}+1} - x\frac{e^{\epsilon_0}-1}{e^{\epsilon_0}+1}, \\ \frac{e^{\epsilon_0}}{e^{\epsilon_0}-1} & \text{w.p. } \frac{1}{e^{\epsilon_0}+1} + x\frac{e^{\epsilon_0}-1}{e^{\epsilon_0}+1}. \end{cases} \tag{3.7}$$

The mechanism q^{priv} is unbiased and satisfies user-level ϵ_0-LDP. Thus, the ith client sends $q^{\mathsf{priv}}(\overline{X}_i)$ to the server, which computes $\widehat{\mu}_i^{\mathsf{priv}} = \frac{1}{m-1}\sum_{l\neq i} q^{\mathsf{priv}}(\overline{X}_l)$ and the variance $\widehat{\sigma}_i^{2(\mathsf{priv})} = \frac{1}{m-2}\sum_{l\neq i}(q^{\mathsf{priv}}(\overline{X}_l)) - \widehat{\mu}_l^{\mathsf{priv}})^2$ and sends $(\widehat{\mu}_i^{\mathsf{priv}}, \widehat{\sigma}_i^{2(\mathsf{priv})})$ to client i. Upon receiving this, client i defines $\widehat{a}_i^{\mathsf{priv}} = \frac{n}{\widehat{\mu}_i^{\mathsf{priv}}(1-\widehat{\mu}_i^{\mathsf{priv}})/\widehat{\sigma}_i^{2(\mathsf{priv})}+n}$ and uses $\widehat{p}_i^{\mathsf{priv}} = \widehat{a}_i^{\mathsf{priv}}\overline{X}_i + (1-\widehat{a}_i^{\mathsf{priv}})\widehat{\mu}^{\mathsf{priv}}$ to estimate p_i.

Theorem 6. *With probability at least $1 - \frac{1}{mn}$, the MSE of the personalized estimator $\widehat{p}_i^{\mathsf{priv}}$ defined above is given by: $\mathbb{E}_{p_i\sim\pi}\mathbb{E}_{q^{\mathsf{priv}},X_1,\ldots,X_m}(\widehat{p}_i^{\mathsf{priv}} - p_i)^2 \leq \mathbb{E}[(\widehat{a}_i^{\mathsf{priv}})^2]\Big(\frac{\alpha\beta}{n(\alpha+\beta)(\alpha+\beta+1)}\Big) + \mathbb{E}[(1-\widehat{a}_i^{\mathsf{priv}})^2]\Big(\frac{\alpha\beta}{(\alpha+\beta)^2(\alpha+\beta+1)} + \frac{(e^{\epsilon_0}+1)^2\log(4m^2n)}{3(e^{\epsilon_0}-1)^2(m-1)}\Big)$.*

3.3 Personalized Learning

Consider a client-server architecture with m clients. There is an unknown global population distribution $\mathbb{P}(\Gamma)$[5] over $\mathbb{R}^d$ from which m i.i.d. local parameters $\boldsymbol{\theta}_1, \ldots, \boldsymbol{\theta}_m \in \mathbb{R}^d$ are sampled. Each client $i \in [m]$ is provided with a dataset consisting of n data points $\{(X_{i1}, Y_{i1}), \ldots, (X_{in}, Y_{in})\}$, where Y_{ij}'s are generated from $(X_{ij}, \boldsymbol{\theta}_i)$ using some distribution $p_{\boldsymbol{\theta}_i}(Y_{ij}|X_{ij})$. Let $Y_i := (Y_{i1}, \ldots, Y_{in})$ and $X_i := (X_{i1}, \ldots, X_{in})$ for $i \in [m]$. The underlying statistical model for our setting is given by

$$p_{\{\boldsymbol{\theta}_i,Y_i\}|\{X_i\}}(\boldsymbol{\theta}_1, \ldots, \boldsymbol{\theta}_m, Y_1, \ldots, Y_m|X_1, \ldots, X_m) = \prod_{i=1}^{m} p(\boldsymbol{\theta}_i) \prod_{i=1}^{m}\prod_{j=1}^{n} p_{\boldsymbol{\theta}_i}(Y_{ij}|X_{ij}). \tag{3.8}$$

[5]For simplicity we will consider this unknown population distribution $\mathbb{P}$ to be parametrized by unknown (arbitrary) parameters Γ.

Note that if we minimize the negative log likelihood of (4.1), we would get the optimal parameters:

$$\widehat{\boldsymbol{\theta}}_1, \ldots, \widehat{\boldsymbol{\theta}}_m := \arg\min_{\boldsymbol{\theta}_1, \ldots, \boldsymbol{\theta}_m} \sum_{i=1}^{m} \sum_{j=1}^{n} -\log(p_{\boldsymbol{\theta}_i}(Y_{ij}|X_{ij})) + \sum_{i=1}^{m} -\log(p(\boldsymbol{\theta}_i)). \tag{3.9}$$

Here, $f_i(\boldsymbol{\theta}_i) := \sum_{j=1}^{n} -\log(p_{\boldsymbol{\theta}_i}(Y_{ij}|X_{ij}))$ denotes the loss function at the i-th client, which only depends on the local data, and $R(\{\boldsymbol{\theta}_i\}) := \sum_{i=1}^{m} -\log(p(\boldsymbol{\theta}_i))$ is the regularizer that depends on the (unknown) global population distribution $\mathbb{P}$ (parametrized by unknown Γ). Note that when clients have little data and we have large number of clients, i.e., $n \ll m$ – the setting of federated learning, clients may not be able to learn good personalized models from their local data alone (if they do, it would lead to large loss). In order to learn better personalized models, clients may utilize other clients' data through collaboration, and the above regularizer (and estimates of the unknown prior distribution $\mathbb{P}$, through estimating its parameters Γ) dictates how the collaboration might be utilized. The above-described statistical framework (3.9) can model many different scenarios, as detailed below:

1. When $\mathbb{P}(\Gamma) \equiv \mathrm{GM}(\{p_l\}_{l=1}^k, \{\boldsymbol{\mu}_l\}_{l=1}^k, \{\sigma_{\theta,l}^2\}_{l=1}^k)$ is a Gaussian mixture, for $\Gamma = \Big\{\Big(\{p_l\}_{l=1}^k, \{\boldsymbol{\mu}_l\}_{l=1}^k, \{\sigma_{\theta,l}\}_{l=1}^k\Big) : p_l \geq 0, \sum_{l=1}^k p_l = 1, \sigma_{\theta,l} \geq 0, \boldsymbol{\mu}_l \in \mathbb{R}^d\Big\}$, then,

$$R(\{\theta_i\}) = -\sum_{i=1}^{m} \log\Big(\sum_{l=1}^{k} p_l \exp(-\frac{\|\mu_l - \theta_i\|_2^2}{2\sigma_{\theta,l}^2})/((2\pi\sigma_{\theta,l})^{d/2})\Big).$$

   Here, the client models $\boldsymbol{\theta}_1, \ldots, \boldsymbol{\theta}_m$ are drawn i.i.d. from $\mathbb{P}(\Gamma)$, where $\boldsymbol{\theta}_i \sim \mathcal{N}(\boldsymbol{\mu}_l, \sigma_{\theta,l}^2 \mathbb{I}_d)$ with prob. p_l, for $l = 1, \ldots, k$. For $k = 1$, $R(\{\boldsymbol{\theta}_i\}) = \frac{md}{2}\log(2\pi\sigma_\theta^2) + \sum_{i=1}^m \frac{\|\boldsymbol{\mu}-\boldsymbol{\theta}_i\|_2^2}{2\sigma_\theta^2}$. Here, unknown $\boldsymbol{\mu}$ can be connected to the global model and $\boldsymbol{\theta}_i$'s as local models, and the alternating iterative optimization optimizes over both. This justifies the use of ℓ_2 regularizer in earlier personalized learning works [42, 64, 65, 100, 131].

2. When $\mathbb{P}(\Gamma) \equiv \mathsf{Laplace}(\boldsymbol{\mu}, b)$, for $\Gamma = \{\boldsymbol{\mu}, b > 0\}$, then $R(\{\boldsymbol{\theta}_i\}) = m\log(2b) + \sum_{i=1}^m \frac{\|\boldsymbol{\theta}_i - \boldsymbol{\mu}\|_1}{b}$.

3. When $p_{\boldsymbol{\theta}_i}(Y_{ij}|X_{ij})$ is according to $\mathcal{N}(\boldsymbol{\theta}_i, \sigma_x^2)$, then $f_i(\boldsymbol{\theta}_i)$ is the quadratic loss as in linear regression. When $p_{\boldsymbol{\theta}_i}(Y_{ij}|X_{ij}) = \sigma(\langle\boldsymbol{\theta}_i, X_{ij}\rangle)^{Y_{ij}}(1-\sigma(\langle\boldsymbol{\theta}_i, X_{ij}\rangle))^{(1-Y_{ij})}$, where $\sigma(z) = 1/1+e^{-z}$ for any $z \in \mathbb{R}$, then $f_i(\boldsymbol{\theta}_i)$ is the cross-entropy (or logistic) loss as in logistic regression.

3.3.1 AdaMix: Adaptive Personalization with Gaussian Mixture Prior

Now we write the full objective function for the Gaussian mixture prior model for a generic local loss function $f_i(\boldsymbol{\theta}_i)$ at client i (the case of linear/logistic regression with (single) Gaussian prior and solving using alternating gradient descent is discussed in Appendices.):

$$\underset{\{\boldsymbol{\theta}_i\},\{\boldsymbol{\mu}_l\},\{p_l\},\{\sigma_{\theta,l}\}}{\arg\min} \sum_{i=1}^{m} F_i^{\text{gm}}(\boldsymbol{\theta}_i) := \sum_{i=1}^{m} \left(f_i(\boldsymbol{\theta}_i) - \log(\sum_{l=1}^{k} p_l \exp(-\frac{\|\boldsymbol{\mu}_l - \boldsymbol{\theta}_i\|_2^2}{2\sigma_{\theta,l}^2})/((2\pi\sigma_{\theta,l})^{d/2})) \right) \tag{3.10}$$

A common example of $f_i(\boldsymbol{\theta}_i)$ is a generic neural network loss function with multi-class softmax output layer and cross entropy loss, i.e., $f_i(\boldsymbol{\theta}_i) := \sum_{j=1}^{n} -\log(p_{\boldsymbol{\theta}_i}(Y_{ij}|X_{ij}))$, where $p_{\theta_i}(Y_{ij}|X_{ij}) = \sigma(\langle \boldsymbol{\theta}_i, X_{ij} \rangle)^{Y_{ij}}(1 - \sigma(\langle \boldsymbol{\theta}_i, X_{ij} \rangle))^{(1-Y_{ij})}$, where $\sigma(z) = 1/1+e^{-z}$ for any $z \in \mathbb{R}$. To solve (3.10), we can either use an alternating gradient descent approach, or we can use a clustering based approach where the server runs a (soft) clustering algorithm on received personalized models. We adopt the second approach here (described in Algorithm 3) as it provides an interesting point of view and can be combined with DP clustering algorithms. Here clients receive the global parameters from the server and do a local iteration on the personalized model (multiple local iterations can be introduced as in FedAvg [116]), later the clients send the personalized models. Receiving the personalized models, server initiates GMM algorithm that outputs global parameters [6]

[6] A discrete mixture model can be proposed as a special case of GM with 0 variance. With this we can recover a similar algorithm as in [115]. Further details are presented in Appendix.

Algorithm 3 Personalized Learning with Gaussian Mixture Prior (`AdaMix`)

Input: Number of iterations T, local datasets (X_i, Y_i) for $i \in [m]$, learning rate η.

Initialize $\boldsymbol{\theta}_1^{(0)}, \ldots, \boldsymbol{\theta}_m^{(0)}$ and $\mathbb{P}^{(0)}, \boldsymbol{\mu}_1^{(0)}, \ldots, \boldsymbol{\mu}_k^{(0)}, \sigma_{\theta,1}^{(0)}, \ldots, \sigma_{\theta,k}^{(0)}$.

for $t = 1$ **to** T **do**

  On Clients:

  for $i = 1$ **to** m: **do**

    Receive $\mathbb{P}^{(t-1)}, \boldsymbol{\mu}_1^{(t-1)}, \ldots, \boldsymbol{\mu}_k^{(t-1)}$, and $\sigma_{\theta,1}^{(t-1)}, \ldots, \sigma_{\theta,k}^{(t-1)}$ from the server

    Update the personalized parameters:

$$\boldsymbol{\theta}_i^{(t)} \leftarrow \boldsymbol{\theta}_i^{(t-1)} - \eta \nabla_{\boldsymbol{\theta}_i^{(t-1)}} F_i^{\mathrm{gm}}(\boldsymbol{\theta}_i^{(t-1)})$$

    Send $\boldsymbol{\theta}_i^{(t)}$ to the server

  end for

  At the Server:

  Receive $\boldsymbol{\theta}_1^{(t)}, \ldots, \boldsymbol{\theta}_m^{(t)}$ from the clients

  Update the global parameters:

$$\mathbb{P}^{(t)}, \boldsymbol{\mu}_1^{(t)}, \ldots, \boldsymbol{\mu}_k^{(t)}, \sigma_{\theta,1}^{(t)}, \ldots, \sigma_{\theta,k}^{(t)} \leftarrow \mathsf{GMM}\Big(\boldsymbol{\theta}_1^{(t)}, \ldots, \boldsymbol{\theta}_m^{(t)}, k\Big)$$

  Broadcast $\mathbb{P}^{(t)}, \{\boldsymbol{\mu}_i^{(t)}\}_{i=1}^k, \{\sigma_{\theta,i}^{(t)}\}_{i=1}^k$ to all clients

end for

Output: Personalized models $\boldsymbol{\theta}_1^T, \ldots, \boldsymbol{\theta}_m^T$.

3.3.2 `AdaPeD`: Adaptive Personalization via Distillation

It has been empirically observed that the knowledge distillation (KD) regularizer (between local and global models) results in better performance than the ℓ_2 regularizer [131]. In fact, using our framework, we can define, for the first time, a certain prior distribution that gives the KD regularizer (see Appendix B). We use the following loss function at the i-th client:

$$f_i(\boldsymbol{\theta}_i) + \frac{1}{2}\log(2\psi) + \frac{f_i^{\mathsf{KD}}(\boldsymbol{\theta}_i, \boldsymbol{\mu})}{2\psi}, \tag{3.11}$$

where $\boldsymbol{\mu}$ denotes the global model, $\boldsymbol{\theta}_i$ denotes the personalized model at client i, and ψ can be viewed as controlling heterogeneity. The goal for each client is to minimize its local loss function, so individual components cannot be too large. For the second term, this implies that ψ cannot be unbounded. For the third term, if $f_i^{\mathsf{KD}}(\boldsymbol{\theta}_i, \boldsymbol{\mu})$ is large, then ψ will also increase (implying that the local parameters are too deviated from the global parameter), hence, it is better to emphasize local training loss to make the first term small. If $f_i^{\mathsf{KD}}(\boldsymbol{\theta}_i, \boldsymbol{\mu})$ is small, then ψ will also decrease (implying that the local parameters are close to the global parameter), so it is better to collaborate and learn better personalized models. Such adaptive weighting quantifies the uncertainty in population distribution during training, balances the learning accordingly, and improves the empirical performance over non-adaptive methods, e.g., [131].

To optimize (3.11), we propose an alternating minimization approach, which we call `AdaPeD`; see Algorithm 4. Besides the personalized model $\boldsymbol{\theta}_i^t$, each client i keeps local copies of the global model $\boldsymbol{\mu}_i^t$ and of the dissimilarity term ψ_i^t, and at synchronization times, server aggregates them to obtain global versions of these $\boldsymbol{\mu}^t, \psi^t$. In this way, the local training of $\boldsymbol{\theta}_i^t$ also incorporates knowledge from other clients' data through $\boldsymbol{\mu}_i^t$. In the end, clients have learned their personalized models $\{\boldsymbol{\theta}_i^T\}_{i=1}^m$.

Algorithm 4 Adaptive Personalization via Distillation (`AdaPeD`)

Parameters: local variances $\{\psi_i^0\}$, personalized models $\{\boldsymbol{\theta}_i^0\}$, local copies of the global model $\{\boldsymbol{\mu}_i^0\}$, learning rates η_1, η_2, η_3, synchronization gap τ

for $t = 0$ **to** $T-1$ **do**
  if τ divides t **then**
    On Server do:
    Choose a subset $\mathcal{K}^t \subseteq [n]$ of K clients
    Broadcast $\boldsymbol{\mu}^t$ and ψ^t
    On Clients $i \in \mathcal{K}^t$ (in parallel) **do**:
    Receive $\boldsymbol{\mu}^t, \psi^t$; set $\boldsymbol{\mu}_i^t = \boldsymbol{\mu}^t$, $\psi_i^t = \psi^t$
  end if
  On Clients $i \in \mathcal{K}^t$ (in parallel) **do**:
  Compute $\mathbf{g}_i^t := \nabla_{\boldsymbol{\theta}_i^t} f_i(\boldsymbol{\theta}_i^t) + \frac{\nabla_{\boldsymbol{\theta}_i^t} f_i^{\mathsf{KD}}(\boldsymbol{\theta}_i^t, \boldsymbol{\mu}_i^t)}{2\psi_i^t}$
  Update: $\boldsymbol{\theta}_i^{t+1} = \boldsymbol{\theta}_i^t - \eta_1 \mathbf{g}_i^t$
  Compute $\mathbf{h}_i^t := {}^{\nabla_{\boldsymbol{\mu}_i^t} f_i^{\mathsf{KD}}(\boldsymbol{\theta}_i^{t+1}, \boldsymbol{\mu}_i^t)}/_{2\psi_i^t}$
  Update: $\boldsymbol{\mu}_i^{t+1} = \boldsymbol{\mu}_i^t - \eta_2 \mathbf{h}_i^t$
  Compute $k_i^t := \frac{1}{2\psi_i^t} - {}^{f_i^{\mathsf{KD}}(\boldsymbol{\theta}_i^{t+1}, \boldsymbol{\mu}_i^{t+1})}/_{2(\psi_i^t)^2}$
  Update: $\psi_i^{t+1} = \psi_i^t - \eta_3 k_i^t$
  if τ divides $t+1$ **then**
    Clients send $\boldsymbol{\mu}_i^t$ and ψ_i^t to **Server**
    Server receives $\{\boldsymbol{\mu}_i^t\}_{i \in \mathcal{K}^t}$ and $\{\psi_i^t\}_{i \in \mathcal{K}^t}$
    Server computes $\boldsymbol{\mu}^{t+1} = \frac{1}{K} \sum_{i \in \mathcal{K}^t} \boldsymbol{\mu}_i^t$ and $\psi^{t+1} = \frac{1}{K} \sum_{i \in \mathcal{K}^t} \psi_i^t$
  end if
end for

Output: Personalized models $(\boldsymbol{\theta}_i^T)_{i=1}^m$

3.3.3 `DP-AdaPeD`: Differentially Private Adaptive Personalization via Distill.

Note that client i communicates $\boldsymbol{\mu}_i^t, \psi_i^t$ (which are updated by accessing the dataset for computing the gradients $\mathbf{h}_i^t, k_i^t$) to the server. So, to privatize $\boldsymbol{\mu}_i^t, \psi_i^t$, client i adds appropriate noise to $\mathbf{h}_i^k, k_i^t$. In order to obtain `DP-AdaPeD`, we replace lines 13 and 15, respectively, by the update rules:

$$\boldsymbol{\mu}_i^{t+1} = \boldsymbol{\mu}_i^t - \eta_2 \left(\frac{\mathbf{h}_i^t}{\max\{\|\mathbf{h}_i^t\|/C_1, 1\}} + \boldsymbol{\nu}_1 \right) \quad \text{and} \quad \psi_i^{t+1} = \psi_i^t - \eta_3 \left(\frac{k_i^t}{\max\{|k_i^t|/C_2, 1\}} + \nu_2 \right),$$

where $\boldsymbol{\nu}_1 \sim \mathcal{N}(0, \sigma_{q_1}^2 \mathbb{I}_d)$ and $\nu_2 \sim \mathcal{N}(0, \sigma_{q_2}^2)$, for some $\sigma_{q_1}, \sigma_{q_2} > 0$ that depend on the desired privacy level and C_1, C_2, which are some predefined constants.

The theorem below (proved in Appendix B) states the Rényi Differential Privacy (RDP) guarantees.

Theorem 7. *After T iterations, `DP-AdaPeD` satisfies $(\alpha, \epsilon(\alpha))$-RDP for $\alpha > 1$, where $\epsilon(\alpha) = \left(\frac{K}{m}\right)^2 6\frac{T}{\tau}\alpha \left(\frac{C_1^2}{K\sigma_{q_1}^2} + \frac{C_2^2}{K\sigma_{q_2}^2} \right)$, where $\frac{K}{m}$ denotes the sampling ratio of clients at each global iteration.*

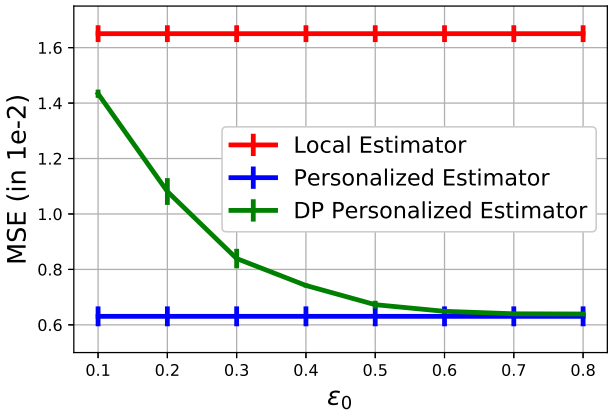
We bound the RDP, as it gives better privacy composition than using the strong composition [122]. We can also convert our results to user-level (ϵ, δ)-DP by using the standard conversion from RDP to (ϵ, δ)-DP [22]. See Appendix B for background on privacy.

3.4 Experiments

Personalized Estimation. We run one experiment for Bernoulli setting with real political data and the other for Gaussian setting with synthetic data. The latter one is differentially private.

• **Political tendencies on county level.** One natural application of Bernoulli setting is modeling bipartisan elections [163]. We did a case study by using US presidential elections on county level between 2000-2020, with $m = 3112$ counties in our dataset. For each county the

Figure 3.1 Private Estimation (Sec. 3.2.1)



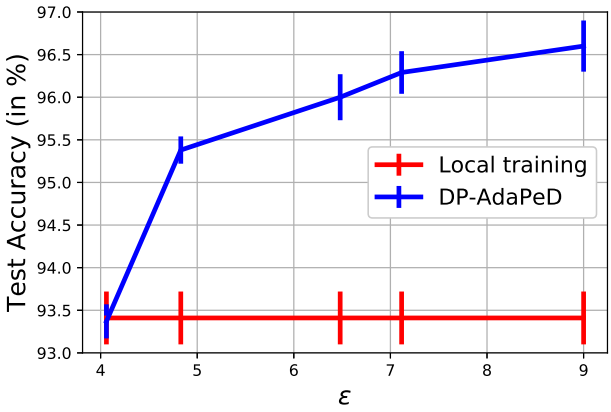
goal is to determine the political tendency parameter p_i. Given 6 election data we did 6-fold cross validation, with 5 elections for training and 1 election for test data. Local estimator takes an average of 5 training samples and personalized estimator is the posterior mean. To simulate a Bernoulli setting we set the data equal to 1 if Republican party won the election and 0 otherwise. We observe the personalized estimator provides MSE (averaged over 6 runs) gain of $10.7 \pm 1.9\%$ against local estimator.

• **DP personalized estimation.** To measure the performance tradeoff of the DP mechanism described in Section 3.2.1, we create a synthetic experiment for Gaussian setting. We let $m = 10000, n = 15$ and $\sigma_\theta = 0.1, \sigma_x = 0.5$, and create a dataset at each client as described in Gaussian setting. Applying the DP mechanism we obtain the following result in Figure 3.1. Here, as expected, when privacy is low (ϵ_0 is high) the private personalized estimator recovers the regular personalized estimator. For higher privacy the private estimator's performance starts to become worse than the non-private estimator.

In Fig. 3.1, we plot MSE vs. ϵ_0 for personalized estimation. In Fig. 3.2, we plot Test Accuracy vs. ϵ on FEMNIST with client sampling 0.33, for `DP-AdaPeD` with unsampled client iterations. Since local training is private, both plots remain constant against ϵ.

Personalized Learning. First we describe the experiment setting and then the results.

Figure 3.2 Private Learning (Sec. 3.3.3)



• **Experiment setting.** We consider image classification on MNIST, FEMNIST [21], CIFAR-10, CIFAR-100 (experimental details for CIFAR-100 is given in Appendix); and train a CNN, similar to the one considered in [116], that has 2 convolutional and 3 fully connected layers. We set $m = 66$ for FEMNIST and $m = 50$ for MNIST, CIFAR-10, CIFAR-100. For FEMNIST, we use a subset of 198 writers so that each client has access to data from 3 authors, which results in a natural type of data heterogeneity due to writing styles of authors. On MNIST, CIFAR-10 we introduce pathological heterogeneity by letting each client sample data from 3 and 4 randomly selected classes only, respectively. We set $\tau = 10$ and vary the batch size so that each epoch consists of 60 iterations. On MNIST we train for 50 epochs, on CIFAR-10 for 250 epochs, on FEMNIST for 40 and 80 epochs, for 0.33 and 0.15 client sampling ratio, respectively. We discuss further details in Appendix F.3.

• **Results.** In Table 3.1 we compare `AdaPeD` against FedAvg [116], FedAvg+ [78] and various personalized FL algorithms: pFedMe [42], Per-FedAvg [48], QuPeD [131] without model compression, and Federated ML [152]. We report further results in Appendix. We observe `AdaPeD` consistently outperforms other methods. It can be seen that methods that use knowledge distillation perform better; on top of this, `AdaPeD` enables us adjust the dependence on collaboration according to the compatibility of global and local decisions/scores. For instance, we set σ_θ^2 to a certain value initially, and observe it progressively decrease, which

Table 3.1 Test accuracy (in %) for CNN model. The CIFAR-10, MNIST, and FEMNIST columns have client sampling ratios $\frac{K}{n}$ of 0.2, 0.1, and 0.15, respectively.

Method	CIFAR-10	CIFAR-100	FEMNIST
FedAvg	60.86 ± 0.94	30.48 ± 0.33	92.18 ± 0.13
FedAvg+fine tuning [78]	63.12 ± 0.31	39.98 ± 0.26	94.12 ± 0.26
`AdaPeD` (Ours)	$\mathbf{72.49} \pm 0.42$	$\mathbf{53.11} \pm 0.34$	$\mathbf{96.55} \pm 0.32$
pFedMe [42]	69.53 ± 0.16	43.65 ± 0.18	94.95 ± 0.55
Per-FedAvg [48]	59.95 ± 0.79	34.78 ± 0.41	93.51 ± 0.31
QuPeD (FP) [131]	71.61 ± 0.70	51.94 ± 0.21	95.99 ± 0.08
Federated ML [152]	71.09 ± 0.67	50.42 ± 0.26	95.12 ± 0.18

Table 3.2 (`DP-AdaPeD`) Test Accuracy (in %) vs. ϵ on MNIST without client sampling.

Method	$\epsilon = 3.35$	$\epsilon = 13.16$	$\epsilon = 27.30$
DP-FedAvg	11.73 ± 0.85	29.91 ± 1.28	55.79 ± 0.29
`DP-AdaPeD` (Ours)	93.32 ± 1.18	98.51 ± 0.90	99.01 ± 0.65

implies clients start to rely on the collaboration more and more. Interestingly, this is not always the case: for `DP-AdaPeD`, we first observe a decrease in σ_θ^2 and later it increases. This suggests: while there is not much accumulated noise, clients prefer to collaborate, and as the noise accumulation on the global model increases due to DP noise, clients prefer not to collaborate. This is exactly the type of autonomous behavior we aimed with adaptive regularization.

• **`DP-AdaPeD`.** In Figure 3.2 and Table 3.2, we observe performance of `DP-AdaPeD` under different ϵ values. `DP-AdaPeD` outperforms DP-FedAvg because personalized models do not need to be privatized by DP mechanism, whereas the global model needs to be. Our experiments provide user-level privacy (more stringent, but appropriate in FL), as opposed to the item-level privacy.

Table 3.3 Mean squared error of our `AdaMix` algorithm and the local training for linear regression.

Method	$n = 10$	$n = 20$	$n = 30$
Local Training	39.93 ± 0.13	30.02 ± 0.08	19.97 ± 0.07
`AdaMix`	10.42 ± 0.15	3.12 ± 0.04	2.55 ± 0.04

• `DP-AdaPeD` **with unsampled client iterations.** When we let unsampled clients to do local iterations (free in terms of privacy cost and a realistic scenario in cross-silo settings) described in Appendix, we can increase `DP-AdaPeD`'s performance under more aggressive privacy constants ϵ. For instance, for FEMNIST with 1/3 client sampling we obtain the result reported in Figure 3.2.

• `AdaMix`**.** We consider linear regression on synthetic data, with $m = 1000$ clients and each client has $n \in \{10, 20, 30\}$ local samples. Each local model $\boldsymbol{\theta}_i \in \mathbb{R}^d$ is drawn from a mixture of two Gaussian distributions $\mathcal{N}(\boldsymbol{\mu}, \Sigma)$ and $\mathcal{N}(-\boldsymbol{\mu}, \Sigma)$, where $\Sigma = 0.001 \times \mathbb{I}_d$ and $d = 50$. Each client sample (X_{ij}, Y_{ij}) is distributed as $X_{ij} \sim \mathcal{N}(0, \mathbb{I}_d)$ and $Y_{ij} = \langle X_{ij}, \boldsymbol{\theta}_i \rangle + w_{ij}$, where $w_{ij} \sim \mathcal{N}(0, 0.1)$. Table 3.3 demonstrates the superior performance of `AdaMix` against the local estimator.

3.5 Conclusion

We proposed a statistical framework leading to new personalized federated estimation and learning algorithms (e.g., AdaMix, AdaPeD); we also incorporated privacy (and communication) constraints into our algorithms and analyzed them. Open questions include information theoretic lower bounds and its comparison to proposed methods; examination of how far the proposed alternating minimization methods (such as in AdaMix, AdaPeD) are from global optima.

CHAPTER 4

Personalized Federated Unsupervised Learning

Statistical heterogeneity of clients' local data is an important characteristic in federated learning, motivating personalized algorithms tailored to local data statistics. Though there has been a plethora of algorithms proposed for personalized supervised learning, discovering the structure of local data through personalized unsupervised learning is less explored. We initiate a systematic study of such personalized unsupervised learning by developing algorithms based on optimization criteria inspired by a hierarchical Bayesian statistical framework. We develop adaptive algorithms that discover the balance between using limited local data and collaborative information. We do this in the context of two unsupervised learning tasks: personalized dimensionality reduction (ADEPT-PCA and ADEPT-AE) and personalized diffusion models (ADEPT-DGM). We develop convergence analyses for our adaptive algorithms which illustrate the dependence on problem parameters (*e.g.,* heterogeneity, local sample size). We also develop a theoretical framework for personalized diffusion models, which shows the benefits of collaboration even under heterogeneity. We finally evaluate our proposed algorithms using synthetic and real data, demonstrating the effective sample amplification for personalized tasks, induced through collaboration, despite data heterogeneity.

4.1 Introduction

One of the goals of unsupervised learning is to discover the underlying structure in data and use this for tasks such as dimensionality reduction and generating new samples from the data distribution. We might want to perform this task on the local data of a client, *e.g.,* data

collected from personal sensors or other devices; such data could have heterogeneous statistics across clients. The desired *personalized* task should be tailored to particular distributions of the local data, and hence to discover this structure one might need a significant amount of local data. There might be insufficient local samples for the task, motivating collaboration between clients. Moreover, as argued in the federated learning (FL) paradigm, we would like to leverage data across clients without data sharing [80, 116]. In this paper, we initiate a systematic study of personalized federated unsupervised learning, where clients collaborate to discover personalized structure in their local data.

There has been a plethora of personalized learning models proposed in the literature, mostly for *supervised* federated learning [42, 48, 114, 126, 131]. These methods were motivated by the statistical heterogeneity of local data, causing a single "global" model to perform poorly on local data. The different personalized federated supervised learning algorithms were unified in [126] using an empirical/hierarchical Bayes statistical model [47] [1], which also suggested new *supervised* learning algorithms. However, there has been much less work on personalized federated unsupervised learning. We will build on the statistical approach studied in [126] for supervised learning, applying it to personalized unsupervised learning. This leads to new federated algorithms for personalized dimensionality reduction and personalized diffusion-based generative models for heterogeneous data. Compared to the application of the hierarchical Bayes model to a supervised setting, in our work, there are several differences. For generative models the likelihood function is not available, hence, we utilize ELBO to derive the optimization criteria. Moreover, for all settings, we introduce hyper-prior which is necessary for stable training and meaningful theoretical analysis (e.g. for convergence). Empirical performance is not sensitive to hyper-prior (a small stability parameter) as we show in the ablation studies, yet without the hyper-prior, there is a risk of converging to trivial solutions.

[1] Not to be confused with Bayesian inference/deep learning, our statistical model is being used to model the data, and consequently, to develop personalized adaptive optimization solved using first-order methods.

A question is how to use local data and learn from others despite heterogeneity. The hierarchical Bayes framework [47] suggests using an estimated population distribution effectively as a prior. However, the challenge is to make each client efficiently combine local data and collaborative information for the unsupervised learning task. We do this by *simultaneously* learning a global model (a proxy for the population model) and a local model by adaptively estimating the discrepancy between the global and local information. In Section 4.2, we define this through a loss function, making it amenable to a distributed gradient descent approach. Our main contributions are as follows.

Unsupervised collaboration learning criteria: Section 4.2 develops and uses the hierarchical Bayes framework for personalized dimensionality reduction and diffusion (generative) models. We develop an *Adaptive Distributed Emperical-Bayes based Personalized Training* (`ADEPT`) criterion which embeds the balance between local data and collaboration for these tasks (see below). As far as we know, these are the first explicit criteria for such personalized federated unsupervised learning tasks.

Personalized dimensionality reduction: Section 4.3 develops adaptive personalized algorithms for linear (PCA) (`ADEPT-PCA`) and non-linear (auto-encoders) (`ADEPT-AE`) for dimensionality reduction. We also demonstrate its convergence in Theorems 8 and 9. In Remark 9, we see that these allow us to theoretically examine the impact of heterogeneity, number of local samples, and number of clients, on optimization performance. We believe these are the first adaptive algorithms for personalized dimensionality reduction and their convergence analyses. Finally, in Section 4.5, we evaluate `ADEPT-PCA` and `ADEPT-AE` and show the benefits of adaptive collaboration. For example, table 4.2 shows effective amplification of as much as $20\times$ in local sample complexity through collaboration.

Personalized diffusion models: Section 4.4 develops an adaptive personalized diffusion generative model (`ADEPT-DGM`) to generate novel samples for local data statistics. We believe that `ADEPT-DGM` is the first algorithm for federated personalized diffusion. We develop a theory for such personalized federated diffusion models through our statistical framework

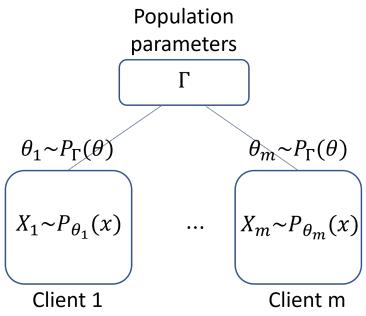


Figure 4.1 Hierarchical Bayesian model of data distribution

and use this to demonstrate, in Theorem 10, conditions when collaboration can improve performance, despite statistical heterogeneity. Finally in Section 4.5, we evaluate `ADEPT-DGM`, demonstrating the value of adaptive collaboration despite heterogeneity, as well as significant performance benefits for "worst" clients.

The common goal of dimensionality reduction and generation tasks is due to the aim of modeling the local data generation process $p(x)$. Diffusion models model the generation process to sample new data from the underlying distribution; whereas, in dimensionality reduction, the goal is to discover a low dimensional latent space, for the underlying distribution. We provide related work in appendix C.1.

4.2 Problem Formulation

We present a hierarchical Bayesian framework for personalized unsupervised learning, using it to develop optimization criteria for federated personalized dimensionality reduction and personalized diffusion models. Notation preliminaries can be found in appendix C.2.

4.2.1 Hierarchical Bayes Model for personalized learning

The *statistical model* based on hierarchical Bayes, suitable for federated unsupervised learning, is illustrated in Figure 4.1. There are m clients. Each client i is associated with a parameter $\boldsymbol{\theta}_i$ and has a local dataset $\boldsymbol{X}_i$ consisting of n data points $(\boldsymbol{x}_{i1}, \ldots, \boldsymbol{x}_{in})$ i.i.d. from distribution $p(\boldsymbol{x}|\boldsymbol{\theta}_i)$. The parameters obey a population distribution (a.k.a. prior distribution in the Bayesian model) $p(\boldsymbol{\theta}|\Gamma)$ parameterized by Γ. We have a (carefully designed) hyper prior distribution π over Γ, i.e., $\Gamma \sim \pi$ to prevent ill-posedness. This statistical model defines the joint distribution

$$\begin{aligned} &p_{\Gamma,\{\boldsymbol{\theta}_i\},\{\boldsymbol{X}_i\}}(\Gamma, \boldsymbol{\theta}_1, \ldots, \boldsymbol{\theta}_m, \boldsymbol{X}_1, \ldots, \boldsymbol{X}_m) \\ &= \pi(\Gamma) \prod_{i=1}^{m} p(\boldsymbol{\theta}_i|\Gamma) \prod_{i=1}^{m} \prod_{j=1}^{n} p(\boldsymbol{x}_{ij}|\boldsymbol{\theta}_i). \end{aligned} \tag{4.1}$$

We learn the distribution parameters $\Gamma, \{\boldsymbol{\theta}_i\}$ from data $\{\boldsymbol{X}_i\}$ by maximizing the joint distribution (a.k.a. maximum a posteriori), by minimizing `ADEPT` loss function:

$$f(\{\boldsymbol{\theta}_i\}, \Gamma; \{\boldsymbol{X}_i\}) = \frac{1}{m} \sum_{i=1}^{m} f_i(\boldsymbol{\theta}_i; \boldsymbol{X}_i) + R(\{\boldsymbol{\theta}_i\}, \Gamma),$$

where $f_i(\boldsymbol{\theta}_i; \boldsymbol{X}_i) := -\sum_{j=1}^{n} \log(p(\boldsymbol{x}_{ij}|\boldsymbol{\theta}_i))$ is the local loss function for client i reflecting the likelihood of the local dataset, and $R(\{\boldsymbol{\theta}_i\}, \Gamma) = -\frac{1}{m} \log \pi(\Gamma) - \frac{1}{m} \sum_{i=1}^{m} \log p(\boldsymbol{\theta}_i|\Gamma)$ is a regularization allowing the collaboration among clients. When the likelihood function is not easy to optimize, we leverage surrogates such as evidence lower bound (ELBO) instead.

In this work, we focus on a Gaussian population distribution over an d_θ-dimensional normed metric space $(\Theta, \|\cdot\|)$[2]. Specifically, $\Gamma = (\boldsymbol{\mu}, \sigma)$, the parameters $\boldsymbol{\theta}, \boldsymbol{\mu} \in \Theta$ and $p(\boldsymbol{\theta}|\Gamma) = \frac{1}{(2\pi\sigma^2)^{d_\theta/2}} \exp(-\frac{\|\boldsymbol{\theta}-\boldsymbol{\mu}\|^2}{2\sigma^2})$, where d is the dimensionality of $\boldsymbol{\theta}$ and $\boldsymbol{\mu}$ has the same dimension as $\boldsymbol{\theta}$. We assume an improper (non-informative) hyper prior π over $\Gamma = (\boldsymbol{\mu}, \sigma)$, as $\boldsymbol{\mu} \sim \mathcal{N}(0, \infty \cdot \boldsymbol{I}_{d_\theta})$ and σ^2 follows an inverse gamma distribution parameterized by a

[2]Note that this general framework can be applied to any general (parametric) population distribution.

hyper-parameter ξ, i.e., $\pi(\boldsymbol{\mu}, \sigma) \propto \exp(\frac{m\xi}{\sigma^2})$ [3]. We thus have the regularization

$$R(\{\boldsymbol{\theta}_i\}, \Gamma) = -\frac{1}{m}\log \pi(\Gamma) - \frac{1}{m}\sum_{i=1}^{m} \log p(\boldsymbol{\theta}_i|\Gamma)$$
$$= \frac{1}{m}\sum_{i=1}^{m} \frac{2\xi + \|\boldsymbol{\mu} - \boldsymbol{\theta}_i\|^2}{2\sigma^2} + d_\theta \log \sigma. \tag{4.2}$$

The Gaussian population distribution is a natural way of capturing relationships between clients' models by modeling the heterogeneity via covariance. Such a population distribution leads to an ℓ_2 regularization, and hence is a natural one to focus on for unsupervised methods. Hence, given the Gaussian population distribution, heterogeneity can be modeled via variance parameters, which enables adaptive optimization problems.

4.2.2 Personalized Dimensionality Reduction

Linear Dimensionality Reduction: The linear dimensionality reduction is equivalent to PCA. We extend a personalized PCA formulation that was previously studied in [128] by introducing adaptivity i.e. optimizing the loss over σ. In this setting, the dataset from client i is $\boldsymbol{X}_i \in \mathbb{R}^{d\times n}$, which contains n samples of d dimensional vectors. Let us denote $\mathbf{S}_i = \frac{1}{n}\boldsymbol{X}_i\boldsymbol{X}_i^\top$ as the sample covariance matrix of client i. For notational consistency with canonical PCA notations, we set the parameters $\boldsymbol{\theta}_i = \boldsymbol{U}_i \in \mathbb{R}^{d\times r}$. Similar to [128], we adopt the probabilistic view of PCA [165],

$$\boldsymbol{x}_{ij} = \boldsymbol{U}_i\boldsymbol{z}_{ij} + \boldsymbol{\epsilon}_{ij}, \tag{4.3}$$

where $\boldsymbol{z}_{i1}, \ldots, \boldsymbol{z}_{in} \overset{i.i.d.}{\sim} \mathcal{N}(\mathbf{0}, \boldsymbol{I}_r)$ and $\boldsymbol{\epsilon}_{i1}, \ldots, \boldsymbol{\epsilon}_{in} \overset{i.i.d.}{\sim} \mathcal{N}(\mathbf{0}, \sigma_\epsilon^2\boldsymbol{I}_d)$. This results in the likelihood function $p(\boldsymbol{X}_i|\boldsymbol{\theta} = \boldsymbol{U}_i) = \mathcal{N}(\mathbf{0}, \boldsymbol{U}_i\boldsymbol{U}_i^\top + \sigma_\epsilon^2\boldsymbol{I})$. Recall that the prior parameter is $\Gamma = (\boldsymbol{\mu}, \sigma)$, here for notational consistency we use $\boldsymbol{U} = \boldsymbol{\mu}$. The underlying metric space $(\Theta, \|\cdot\|)$ containing the parameters $\boldsymbol{U}$ and $\boldsymbol{U}$ in the Steifel manifold where $St(d, r) := \{\boldsymbol{U} \in \mathbb{R}^{d\times r} | \boldsymbol{U}^\top\boldsymbol{U} = \boldsymbol{I}\}$. The metric is $d(\boldsymbol{U}, \boldsymbol{U}) = \|P_{\mathcal{T}_{\boldsymbol{U}}}(\boldsymbol{U})\|$ where $P_{\mathcal{T}_{\boldsymbol{U}}}(\boldsymbol{U}) = \boldsymbol{U} - \frac{1}{2}\boldsymbol{U}(\boldsymbol{U}^\top\boldsymbol{U} + \boldsymbol{U}^\top\boldsymbol{U})$ is the projection

[3] Inverse-Gamma distribution is conjugate prior distribution for the variance term.

of $\boldsymbol{U}$ onto the tangent space at $\boldsymbol{U}$. Because computing the (geodesic) distance on $St(d,r)$ is hard, the defined metric first projects a matrix to the tangent space of another matrix and then computes the Frobenius norm. The personalized unsupervised learning is then minimizing the `ADEPT-PCA` loss function, f^{pca} over $\{\boldsymbol{U}_i\}, \boldsymbol{U}, \sigma$:

$$\frac{1}{m}\left(\sum_{i=1}^{m}\frac{n}{2}(\log(|\boldsymbol{W}_i|)+\text{tr}(\boldsymbol{W}_i^{-1}\mathbf{S}_i))+\frac{2\xi+d^2(\boldsymbol{U},\boldsymbol{U}_i)}{2\sigma^2}\right) + d_\theta \log\sigma \tag{4.4}$$

s.t.$\boldsymbol{U}^\top\boldsymbol{U}=\boldsymbol{I}, \boldsymbol{U}_i^\top\boldsymbol{U}_i=\boldsymbol{I}\forall i; \boldsymbol{W}_i=(\boldsymbol{U}_i\boldsymbol{U}_i^\top+\sigma_\epsilon^2\boldsymbol{I})$.

Non-linear dimensionality reduction: While PCA is a good starting point for dimensionality reduction, it cannot capture non-linear relations between the latent variable and the observed space. Hence, one can extend (4.3) to model non-linearity as follows,

$$X_{ij} = \psi_{\boldsymbol{\theta}_i^{\text{d}}}(\boldsymbol{z}_{ij}) + \boldsymbol{\epsilon}_{ij}, \tag{4.5}$$

where $\boldsymbol{\theta}_i^{\text{d}}$ parameterizes a non-linear decoding map. We can parameterize the encoder structure such that $\boldsymbol{z}_{ij} = g_{\boldsymbol{\theta}_i^{\text{e}}}(X_{ij})$. Using Gaussian distribution we get the `ADEPT-AE` criterion:

$$\underset{\{\boldsymbol{\theta}_i\},\boldsymbol{\mu},\sigma}{\arg\min} f^{\text{ae}}(\{\boldsymbol{\theta}_i\},\boldsymbol{\mu},\sigma) = \frac{1}{m}\sum_{i=1}^{m}\left(\|\boldsymbol{X}_i - \psi_{\boldsymbol{\theta}_i^{\text{d}}}(g_{\boldsymbol{\theta}_i^{\text{e}}}(\boldsymbol{X}_i))\|_F^2 + \frac{2\xi + \|\boldsymbol{\mu}-\boldsymbol{\theta}_i\|^2}{2\sigma^2}\right) + d_\theta \log\sigma \tag{4.6}$$

where $\boldsymbol{\mu}$ is the global model and $\{\boldsymbol{\theta}_i\}$ are the personalized AEs through concatenation of $\boldsymbol{\theta}_i^{\text{e}}$, $\boldsymbol{\theta}_i^{\text{d}}$.

4.2.3 Personalized Generation through Diffusion Models

The denoising diffusion model has attracted attention recently due to its capability of generating high-quality images and the theoretical foundation of the stochastic differential equations [144, 158]. The mathematical model of a diffusion model is the following stochastic

differential equation [159]

$$d\boldsymbol{x}_t = d\boldsymbol{w}_t \quad \text{(variance exploding process)}, \tag{4.7}$$

where $\boldsymbol{w}_t$ is standard d-dimensional Brownian motion. For time $t \in [0, T]$, its time-reversed process is

$$d\boldsymbol{x}_t^{\leftarrow} = \nabla \ln p_{\boldsymbol{x}_{T-t}}(\boldsymbol{x}_t^{\leftarrow})dt + d\boldsymbol{w}_t^{\leftarrow}, \tag{4.8}$$

where $p_{\boldsymbol{x}_t}$ is the probability density function of $\boldsymbol{x}_t$, $\boldsymbol{w}_t^{\leftarrow}$ is a standard Wiener process. It can be shown that $X_t^{\leftarrow}$ follows the same distribution as $\boldsymbol{x}_{T-t}$. More strongly, suppose $X_0^{\leftarrow}$ has the same distribution as $\boldsymbol{x}_T$, and then the two processes $\{\boldsymbol{x}_{T-t}\}_{t\in[0,T]}$, $\{\boldsymbol{x}_t^{\leftarrow}\}_{t\in[0,T]}$ have the same distribution.

Suppose that the score function $\nabla \ln p_{\boldsymbol{x}_t}(\boldsymbol{x})$ can be represented by a neural network $\boldsymbol{\phi}(\boldsymbol{x}; \boldsymbol{\theta}, t) \in \mathbb{R}^d$ for some parameter $\boldsymbol{\theta}$. The data generation is then integration over the time-revised process (4.8) with the drift function $\boldsymbol{\phi}(\boldsymbol{x}; \boldsymbol{\theta}, t)$ and the starting distribution $\boldsymbol{x}_0^{\leftarrow} \sim \mathcal{N}(\mathbf{0}, \sigma_0^2 \boldsymbol{I}_d)$. The generated distribution $p(\boldsymbol{x}|\boldsymbol{\theta})$ is then implicitly defined without a closed form. [89, Eq (3)] shows that

$$\begin{aligned} &\ln p(\boldsymbol{x}|\boldsymbol{\theta}) \geq ELBO_{\boldsymbol{\theta}}(\boldsymbol{x}) \\ &= -\frac{1}{2}\int_{t=0}^{T} \mathbb{E}\|\boldsymbol{\phi}(\boldsymbol{x}_t; \boldsymbol{\theta}_j, t) - \nabla_{\boldsymbol{x}_t} \ln p_{\boldsymbol{x}_t|\boldsymbol{x}_0}(\boldsymbol{x}_t|\boldsymbol{x})\|^2 dt + C \end{aligned} \tag{4.9}$$

where $\boldsymbol{x}_t \sim \mathcal{N}(\boldsymbol{x}, \boldsymbol{I}_d)$ and C is some constant factor independent of $\boldsymbol{\theta}$. Accordingly, we use ELBO as a surrogate function for the negative log-likelihood; thus, the personalized loss function is $f_i(\boldsymbol{\theta}_i; \boldsymbol{X}_i) = -ELBO_{\boldsymbol{\theta}_i}(\boldsymbol{X}_i) = -\sum_{j=1}^{n} ELBO_{\boldsymbol{\theta}_i}(\boldsymbol{x}_{ij})$. The personalized data generation is then minimizing the `ADEPT-DGM` loss function

$$\begin{aligned} &\min_{\{\boldsymbol{\theta}_i\}, \boldsymbol{\mu}, \sigma^2} f^{\mathrm{df}}(\{\boldsymbol{\theta}_i\}, \boldsymbol{\mu}, \sigma; \{\boldsymbol{X}_i\}) \\ &:= \frac{1}{m}\sum_{i=1}^{m}\left(-ELBO_{\boldsymbol{\theta}_i}(\boldsymbol{X}_i) + \frac{2\xi + \|\boldsymbol{\mu} - \boldsymbol{\theta}_i\|^2}{2\sigma^2}\right) + d_\theta \log \sigma. \end{aligned} \tag{4.10}$$

4.3 Personalized Federated Dimensionality Reduction

We present our algorithms and convergence results on personalized dimensionality reduction.

4.3.1 Personalized Adaptive PCA: ADEPT-PCA

We introduce ADEPT-PCA (algorithm 15 deferred to Appendix C.3) to train adaptive personalized PCA. To show the convergence of the algorithm we need the following standard assumption and naturally occurring lower bound on σ.

Assumption 1. *For each client i, the operator and Frobenius norms of $\mathbf{S}_i$ are bounded by*

$$\|\mathbf{S}_i\|_F \leq G_{i,F} \quad \textit{and} \quad \|\mathbf{S}_i\|_{op} \leq G_{i,op},$$

and $G_{max,F} := \max_{i\in[m]} G_{i,F}$,$G_{max,op} := \max_{i\in[m]} G_{i,op}$. The assumption implies the Lipschitz smoothness properties of the loss function w.r.t. personalized PCs $\{\boldsymbol{U}_i\}$.

Lemma 1 (A lower bound on σ_t)**.** *Given any $\omega \in (0,1)$. Let the learning rate $\eta_3 \leq (1-\omega)\frac{2\xi}{d_\theta^2}$ and the initialization $\sigma_0 \geq \omega\sqrt{\frac{2\xi}{d_\theta}}$. Then, for all $t \in [T]$, we have $\sigma_t \geq \omega\sqrt{\frac{2\xi}{d_\theta}}$.*

See Appendix C.3.3.1 for the proof. We will fix some $\omega \in (0,1)$ for the rest of the paper and initialize σ_0 accordingly so that we can utilize the lower bound in Lemma 1. The bound is due to ξ and is necessary to guarantee that the loss does not explode due to vanishing σ. Let us now define $\mathbf{g}_{i,t}^U = P_{\mathcal{T}_{\boldsymbol{U}_{i,t-1}}}(\nabla_{\boldsymbol{U}_{i,t-1}} f_i^{\text{pca}}(\boldsymbol{U}_{i,t-1}, \boldsymbol{U}_{t-1}, \sigma_{t-1}))$, $\mathbf{g}_t^V = \frac{1}{m}\sum_{i=1}^m P_{\mathcal{T}_{\boldsymbol{U}_{t-1}}}(\nabla_{\boldsymbol{U}_{t-1}} f_i^{\text{pca}}(\boldsymbol{U}_{i,t}, \boldsymbol{U}_{t-1}, \sigma_{t-1}))$, and $g_t^\sigma = \frac{\partial}{\partial \sigma_{t-1}} f^{\text{pca}}(\{\boldsymbol{U}_{i,t-1}\}_i, \boldsymbol{U}_{t-1}, \sigma_{t-1})$. Then, algorithm 15 has the following convergence bound for finding a first-order stationary point.

Theorem 8 (Convergence of ADEPT-PCA algorithm 15)**.** *Let $G_t = \left(\frac{1}{m}\sum_{i=1}^m \left\|\mathbf{g}_{i,t}^U\right\|^2\right) + \left\|\mathbf{g}_t^V\right\|^2 + (g_t^\sigma)^2$. By choosing $\eta_1 = \min\{\frac{1}{3C_{\eta_1}}, 1\}$, $\eta_2 = \min\{\frac{1}{3C_{\eta_2}}, 1\}$, and $\eta_3 = \min\left\{\frac{\eta_1}{3(L_U^{(\sigma)})^2}, \frac{\eta_2}{3(L_V^{(\sigma)})^2}, \frac{1}{L_\sigma}\right\}$, we have*

$$\frac{1}{T}\sum_{t=1}^T G_t \leq \frac{3\Delta_T^{\text{pca}}}{T\min\{\eta_1,\eta_2,\eta_3\}},$$

where T is number of total iterations, $\Delta_T^{\text{pca}} = f^{\text{pca}}(\{\boldsymbol{U}_{i,0}\}_i, \boldsymbol{U}_0, \sigma_0) - f^{\text{pca}}(\{\boldsymbol{U}_{i,T}\}_i, \boldsymbol{U}_T, \sigma_T)$, and η_1, η_2, η_3 are the learning rates for updating $\boldsymbol{U}_i$, $\boldsymbol{U}$, and σ respectively. The constants are defined such that $f^{pca}(\cdot)$ isL_σ-smooth w.r.t. σ and $\mathbf{g}_{i,t}^U, \mathbf{g}_t^V$ are $L_U^{(\sigma)}, L_V^{(\sigma)}$ continuous w.r.t. σ respectively. C_{η_1}, C_{η_2} are defined in Appendix C.3 and depend on smoothness w.r.t. $\boldsymbol{U}, \boldsymbol{U}$.

We provide a detailed comment on the factors impacting the convergence rate in Remark 9.

Remark 8 (The Polar Retraction)**.** *In algorithm 15, we use the polar retraction to map the updated* $\boldsymbol{U}$ *and* $\boldsymbol{U}_i$ *back to the Steifel manifold. We define the polar retraction in appendix C.3 in detail.*

Proof outline of Theorem 8. We show that a lower bound on σ_t is obtained with proper initialization of σ_0 and we can further derive the Lipschitz constants of the loss function (4.4). The sufficient decrease of the loss function w.r.t. $\boldsymbol{U}_i$, $\boldsymbol{U}$, and σ (Lemma 17) are derived individually using non-expansiveness of polar retraction (lemma 11) and Lipschitz type inequality (lemma 12). After the sufficient decrease, we show that carefully choosing the learning rates results in an upper bound on the squared norm of gradients. The detailed proof is in Appendix C.3. Technical challenges in this proof are to control the error due to projections, utilize the lower bound on σ to avoid non-smoothness, and use the Lipschitz continuity of the gradients to combine updates on PCs with the update on σ.

4.3.2 Personalized Adaptive AEs: `ADEPT-AE`

Algorithm 5 shows the alternating gradient descent training procedure for personalized adaptive AEs. At the beginning of local iterations (**line 8**) the clients receive the global model and σ[4] terms then do local updates on $\boldsymbol{\theta}_i$ (**line 10**). At the end of local iterations, the client updates the global model and variance term using its personalized model **lines 12,13** and sends them to the server where it is aggregated.

Assumption 2. *The loss function* $f_i^{(ae)}(\{\boldsymbol{\theta}_i\}, \boldsymbol{\mu}, \sigma)$ *is* $L_{\boldsymbol{\theta}}$*-smooth w.r.t. individual* $\{\boldsymbol{\theta}_i\}$*,* $L_{\boldsymbol{\mu}}$*-smooth w.r.t.* $\boldsymbol{\mu}$ *and* L_σ*-smooth w.r.t.* σ*. Note that only the first one is an assumption, second and third ones are derived from the fact that* σ *is lower bounded when initialized properly (Appendix C.4).*

[4]In the experiments we use individual variance terms for each weight that is $\boldsymbol{\sigma} \in \Theta$ which only has a constant effect on convergence (see Appendix C.4).

Algorithm 5 `ADEPT-AE` Algorithm

Input: Number of iterations T, learning rates (η_1, η_2, η_3), number of local iterations τ

Init local models $\{\boldsymbol{\theta}_{i,0}\}_{i=1}^m$, global model $\boldsymbol{\mu}_0$, and σ_0.
On server:
Broadcast $\boldsymbol{\mu}_0$, σ_0 to all clients
for $t = 1$ **to** T **do**
  On Clients:
  for $i = 1$ **to** m **do**
    if τ divides $t-1$ **then**
      Receive $\boldsymbol{\mu}_{t-1}$, σ_{t-1}
    end if
    $\boldsymbol{\theta}_{i,t}=\boldsymbol{\theta}_{i,t-1}-\eta_1\nabla_{\boldsymbol{\theta}_{i,t-1}} f_i^{\text{ae}}(\boldsymbol{\theta}_{i,t-1}, \boldsymbol{\mu}_{t-1}, \sigma_{t-1})$
    if τ divides t **then**
      $\boldsymbol{\mu}_{i,t} = \boldsymbol{\mu}_{t-1} - \eta_2\nabla_{\boldsymbol{\mu}_{t-1}} f_i^{\text{ae}}(\boldsymbol{\theta}_{i,t}, \boldsymbol{\mu}_{t-1}, \sigma_{t-1})$
      $\sigma_{i,t} = \sigma_{t-1} - \eta_3\nabla_{\sigma_{t-1}} f_i^{\text{ae}}(\boldsymbol{\theta}_{i,t}, \boldsymbol{\mu}_{t-1}, \sigma_{t-1})$
      Send $\boldsymbol{\mu}_{i,t}, \sigma_{i,t}$ to server
    else
      $\boldsymbol{\mu}_t = \boldsymbol{\mu}_{t-1}$, $\sigma_t = \sigma_{t-1}$
    end if
  end for
  At the Server:
  if τ divides t **then**
    Receive $\{\boldsymbol{\mu}_{i,t}\}_{i=1}^m$ and $\{\sigma_{i,t}\}_{i=1}^m$
    $\boldsymbol{\mu}_t = \frac{1}{m}\sum_{i=1}^m \boldsymbol{\mu}_{i,t}$, $\sigma_t = \frac{1}{m}\sum_{i=1}^m \sigma_{i,t}$
    Broadcast $\boldsymbol{\mu}_t$, σ_t to all clients
  end if
end for

Output: Personalized autoencoders $\{\boldsymbol{\theta}_{1,T}, \ldots, \boldsymbol{\theta}_{m,T}\}$.

Let $\mathbf{g}_{i,t}^{\boldsymbol{\theta}} = \nabla_{\boldsymbol{\theta}_{i,t-1}} f_i^{\text{ae}}(\boldsymbol{\theta}_{i,t-1}, \boldsymbol{\mu}_{t-1}, \sigma_{t-1})$, $\mathbf{g}_t^{\boldsymbol{\mu}} = \nabla_{\boldsymbol{\mu}_{t-1}} f^{\text{ae}}(\{\boldsymbol{\theta}_{i,t}\}_i, \boldsymbol{\mu}_{t-1}, \sigma_{t-1})$, $g_t^{\sigma} = \frac{\partial f(\{\boldsymbol{\theta}_{i,t}\}_i, \boldsymbol{\mu}_{t-1}, \sigma_{t-1})}{\partial \sigma_{t-1}}$. For algorithm 5, we obtain the following convergence bound for finding a first-order stationary point.

Theorem 9 (Convergence of `ADEPT-AE` (algorithm 5))**.** *Let us define* $G_t = \left(\frac{1}{m}\sum_{i=1}^{m} \|\mathbf{g}_{i,t}^{\boldsymbol{\theta}}\|^2\right) + \|\mathbf{g}_t^{\boldsymbol{\mu}}\|^2 + (g_t^{\sigma})^2$. *Then, by choosing* $\eta_1 = \frac{1}{L_{\boldsymbol{\theta}}}$, $\eta_2 = \frac{1}{L_\sigma + L_\sigma^{(\boldsymbol{\mu})^2}}$, *and* $\eta_3 = \min\{1, \frac{1}{L_{\boldsymbol{\mu}}}\}$, *we have* $\min_{t\in[T],\tau|t}\{G_t\}$ *is upper bounded by*

$$\frac{\max\{L_{\boldsymbol{\theta}}, L_\sigma + L_\sigma^{(\boldsymbol{\mu})^2}, L_{\boldsymbol{\mu}}, 1\}\Delta_T^{\text{ae}}}{R},$$

where $R = T/\tau$ *is the number of communication rounds,* $\Delta_T^{\text{ae}} = f^{\text{ae}}(\{\boldsymbol{\theta}_{i,0}\}_i, \boldsymbol{\mu}_0, \sigma_0) - f^{\text{ae}}(\{\boldsymbol{\theta}_{i,T}\}_i, \boldsymbol{\mu}_T, \sigma_T)$, *and the constants can be found in Lemma 18.*

Remark 9. *By examining the multiplicative constants within the bounds specified in theorems 8 and 9, we note a consistent observation:* σ *exhibits an inverse relationship with convergence speed. A higher* σ*, whether resulting from a large value of* ξ *or inherent heterogeneity in the setting, can expedite the convergence process. Essentially, a large* σ *diminishes collaboration, allowing the model to fit quickly due to a reduced effective number of samples. Conversely, a smaller* σ *promotes collaboration and may augment the effective sample count. This theoretical observation persists in the experiments as well, as can be seen from the convergence plots in Appendix F.3. Note that faster convergence does not necessarily imply a superior generalization error. Consequently, in our experiments, opting for a high value of* σ_0 *facilitates fast convergence in the initial stages, while still allowing flexibility for adjustments to yield a superior generalization error.*

Proof outline of Theorem 9. Similar to our proof for Theorem 8, we utilize the lower bound on σ_t and look for the sufficient decrease w.r.t. $\boldsymbol{\theta}_i$, $\boldsymbol{\mu}$, and σ. However, now we have to deal with the fact that we are doing τ local iterations after each communication round. Thus, we consider the two cases of whether it is a communication round or not separately and come to our Theorem in the end. The proof is in Appendix C.4.

4.4 Personalized Generation through Adaptive Diffusion Models: ADEPT-DGM

The ADEPT-DGM algorithm is given in algorithm 16 (deferred to Appendix C.5). It is based on optimizing the criterion given in (4.10). The adaptation method to discover the balance between local data and collaborative information is similar to that in ADEPT-AE algorithm 5. As an illustration of the effectiveness of personalized diffusion model learning, we analyze a simple personalized Gaussian distribution generation problem, i.e., client-i's target distribution is a Gaussian distribution $p(\boldsymbol{x}|\boldsymbol{\theta}_i) = \mathcal{N}(\boldsymbol{x}; \boldsymbol{\theta}_i, \sigma_0^2 \boldsymbol{I}_d)$. In the following, we first introduce some details of the canonical denoising diffusion model for Gaussian target distribution and analyze the improvement by collaboration in the proposed personalized algorithm. Diffusion model with Gaussian target distribution: When the desired distribution is $\mathcal{N}(\boldsymbol{x}; \boldsymbol{\theta}, \sigma_0^2 I_d)$, the diffusion process (4.7) is a Gaussian process and the drift term for the reverse-time process (4.8) is a linear function, i.e., $\nabla \ln p_{\boldsymbol{x}_{T-t}}(\boldsymbol{x}_t^{\leftarrow}) = -\frac{\boldsymbol{x}_t^{\leftarrow} - \boldsymbol{\theta}}{\sigma_0^2 + t}$. The time-reversed process is

$$d\boldsymbol{x}_t^{\leftarrow} = \nabla \ln p_{\boldsymbol{x}_{T-t}}(\boldsymbol{x}_t^{\leftarrow}) dt + d\boldsymbol{w}_t^{\leftarrow}, \boldsymbol{x}_0^{\leftarrow} \sim \mathcal{N}(\boldsymbol{\theta}, (\sigma_0^2 + T)\boldsymbol{I}_d). \tag{4.11}$$

Without the knowledge of $\boldsymbol{\theta}$, we approximate the score function by a neural network of $\boldsymbol{\phi}(\boldsymbol{x}; \hat{\boldsymbol{\theta}}, t) = -\frac{\boldsymbol{x} - \hat{\boldsymbol{\theta}}}{\sigma_0^2 + t}$ and approximate the initial distribution of the time-reversed process $\mathcal{N}(\boldsymbol{\theta}, (\sigma_0^2 + T)\boldsymbol{I}_d)$ by $\mathcal{N}(\boldsymbol{0}, (\sigma_0^2 + T)\boldsymbol{I}_d)$. The learned/approximated time-reversed process is then

$$d\boldsymbol{x}_t^{\leftarrow} = -\frac{\boldsymbol{x} - \hat{\boldsymbol{\theta}}}{\sigma_0^2 + t} dt + d\boldsymbol{w}_t^{\leftarrow}, \boldsymbol{x}_0^{\leftarrow} \sim \mathcal{N}(\boldsymbol{0}, (\sigma_0^2 + T)\boldsymbol{I}_d). \tag{4.12}$$

The following lemma characterizes the difference between the generation and target distributions.

Lemma 2. *The output distribution $p_{\boldsymbol{x}_T^{\leftarrow}|\hat{\boldsymbol{\theta}}}$ of the learned reversed-time process (4.12) satisfies,*

$$D_{KL}(p_{\boldsymbol{x}|\boldsymbol{\theta}} || p_{\boldsymbol{x}_T^{\leftarrow}|\hat{\boldsymbol{\theta}}}) = \left\| \boldsymbol{\theta} - \hat{\boldsymbol{\theta}} + \frac{\sigma_0^2}{\sigma_0^2 + T} \hat{\boldsymbol{\theta}} \right\|^2, \tag{4.13}$$

where D_{KL} is the Kullback-Leibler divergence and $p_{\boldsymbol{x}|\boldsymbol{\theta}} = \mathcal{N}(\boldsymbol{x}; \boldsymbol{\theta}_i, \sigma_0^2 \boldsymbol{I}_d)$ is the target distribution.

The lemma shows that the KL-divergence between target distribution and the learned distribution is measured by the difference between $\boldsymbol{\theta}$ and $\hat{\boldsymbol{\theta}}$ a bias term $\frac{\sigma_0^2}{\sigma_0^2+T}\hat{\boldsymbol{\theta}}$ due to the initial distribution mismatch of the time-reversed process. It is straightforward to verify that for $\boldsymbol{\phi}(\boldsymbol{x};\hat{\boldsymbol{\theta}},t) = -\frac{\boldsymbol{x}-\hat{\boldsymbol{\theta}}}{\sigma_0^2+t}$, training with dataset $\boldsymbol{X} = (\boldsymbol{x}_1,\ldots,\boldsymbol{x}_n)$ by maximizing ELBO (4.9) has a closed-form sample-mean solution, i.e., $\hat{\boldsymbol{\theta}} = \arg\max_{\boldsymbol{\theta}} ELBO_\theta(\boldsymbol{X}) = \frac{\sum_{j=1}^n \boldsymbol{x}_j}{n}$. Since the data are i.i.d. sampled from $\mathcal{N}(\boldsymbol{\theta},\sigma_0^2\boldsymbol{I}_d)$, we have $\mathbb{E}[D_{KL}(p_{\boldsymbol{x}|\boldsymbol{\theta}}||p_{\boldsymbol{x}_T^{\leftarrow}|\hat{\boldsymbol{\theta}}})] = \frac{\sigma_0^2}{n}$, when omitting the initial distribution bias by $T\to\infty$. It can be viewed as personalized training without collaboration and in the following, we show that the proposed personalized training with collaboration can improve over this. Personalized denoising diffusion model with Gaussian targets: The local dataset $\boldsymbol{X}_i$ of client-i sampled i.i.d. from a target Gaussian distribution $P(\boldsymbol{x}|\boldsymbol{\theta}_i) = \mathcal{N}(\boldsymbol{x};\boldsymbol{\theta}_i,\sigma_0^2\boldsymbol{I}_d)$, and the population distribution is also Gaussian with $P(\boldsymbol{\theta}|\Gamma_*) = \mathcal{N}(\boldsymbol{\theta};\boldsymbol{\mu}_*,\sigma_*^2\boldsymbol{I}_d)$. Note that we analyze a fixed unknown population distribution parameterized by $\Gamma_* = (\boldsymbol{\mu}_*,\sigma_*)$, though the proposed loss function and algorithm follows a Hierarchical Bayesian model as in Section 4.2.3.

Lemma 3 (Personalized estimation)**.** *For the parameterized score function $\boldsymbol{\phi}(\boldsymbol{x};\boldsymbol{\theta},t) = -\frac{\boldsymbol{x}-\boldsymbol{\theta}}{\sigma_0^2+t}$, the optimal solution to (4.10) is*

$$\hat{\boldsymbol{\mu}} = \frac{\sum_{i=1}^m\sum_{j=1}^n \boldsymbol{x}_{ij}}{mn}, \hat{\boldsymbol{\theta}}_i = \frac{n\alpha\hat{\sigma}^2}{n\alpha\hat{\sigma}^2+1}\left(\frac{\sum_{j=1}^n \boldsymbol{x}_{ij}}{n} + \hat{\boldsymbol{\mu}}\right),$$

where $\alpha = \frac{1}{\sigma_0^2} - \frac{1}{\sigma_0^2+T}$ and $\hat{\sigma}^2$ satisfies $\hat{\sigma}^2 = \frac{2\xi}{d} + s^2(\frac{n\alpha\hat{\sigma}^2}{n\alpha\hat{\sigma}^2+1})^2$ with $s^2 = \frac{\sum_{i=1}^m \left\|\hat{\boldsymbol{\mu}} - \frac{1}{n}\sum_{j=1}^n \boldsymbol{x}_{ij}\right\|^2}{md}$.

Remark 10. *We note that the global optimum model is the average of average samples of each client. The personalized optimum model is an interpolation of the local estimate and the true global model. The interpolation coefficient depends on the heterogeneity (large σ skews the result towards the local estimate and low σ skews it towards the true global model). A large amount of local samples n decreases the reliance on $\boldsymbol{\mu}$ These observations are in parallel with findings in [126] on mean estimation.*

Theorem 10 (Condition for performance improvement)**.** *Consider an asymptotic regime that $m\to\infty$ and $T\to\infty$. Compared to training without collaboration, the solution of $(\{\hat{\boldsymbol{\theta}}_i\},\hat{\boldsymbol{\mu}},\hat{\sigma}^2)$*

in Lemma 3 improves the averaged KL-divergence $\frac{1}{m}\sum_{i=1}^{m} D_{KL}(p_{\boldsymbol{x}|\boldsymbol{\theta}_i}||p_{\boldsymbol{x}_T^{\leftarrow}|\hat{\boldsymbol{\theta}}_i})$ *by a factor of* $\left(\frac{2\hat{\sigma}^2+\sigma_0^2/n-\sigma_*^2}{\hat{\sigma}^2+\sigma_0^2/n}\right)\frac{\sigma_0^2/n}{\hat{\sigma}^2+\sigma_0^2/n}\frac{\sigma_0^2}{n}$*, when* $\hat{\sigma}^2 > \frac{\sigma_*^2}{2} - \frac{\sigma_0^2}{2n}$.

Corollary 1. *Under the same setting as in Theorem 10, choosing* $\xi \geq \frac{3d\sigma_0^2}{2n}$ *guarantees improvement of collaboration for any population distribution* $\mathcal{N}(\boldsymbol{\mu}_*, \sigma_*\boldsymbol{I}_d)$.

Remark 11. *Note that if our estimate* $\hat{\sigma}^2$ *of the population variance is accurate,* i.e., $\hat{\sigma}^2 = \sigma_*^2$, *then collaboration always improves over only using local data for a personalized generation; and the collaboration gain is the largest. In this case the gain is larger when the number of local samples is small. However, if the estimate is inaccurate, one could be better off without collaboration. However, by setting the hyperparameter* $\xi \geq \frac{3d\sigma_0^2}{2n}$, *we can ensure that our estimate* $\hat{\sigma}^2 \geq \frac{1}{2}(\sigma_*^2 - \sigma_0^2/n)$, *ensuring that collaboration is useful.*

4.5 Experiments

4.5.1 Experimental Setting

In our experiments, we compare our adaptive personalized unsupervised algorithms with global training (FedAvg, FedAvg+fine-tuning), local training (training individual models without collaboration), and competitive baselines in terms of testing performance under different heterogeneous scenarios. While the supervised personalized FL methods that are based on architectural changes, such as shared representations [34], are not applicable to unsupervised personalized FL due to symmetric model architecture; optimization-based methods can be applied to unsupervised settings, without providing theoretical insights that our framework provides. From such methods, we compare to pFedMe [42] applied them in the AE problem. Experiments for `ADEPT-PCA` are in appendix C.6.1. Additional experiments, and details hyper-parameters can be found in appendix F.3.

ADEPT-AE: We do synthetic and real data experiments for AEs. We use 50 clients ($m = 50$) for all AE experiments. In the synthetic experiments, from a zero mean $\sigma_\mu = 0.1$ standard deviation Gaussian distribution, we sample weights for a single layer decoder $\mu^{d,*}$ with 5

latent dimensionality and 64 output dimensionality. Then by perturbing the weights with another zero mean Gaussian with σ^* we obtain the true personalized decoders, which are used as in (4.5) to generate 10 local samples across 50 clients. Heterogeneity among clients will depend on σ^* and is quantified in terms of signal-to-noise ratio as $20\log_{10}\frac{\sigma_\mu}{\sigma^*}$ dB. For real data experiments, we use MNIST, Fashion MNIST (F. MNIST), and CIFAR-10. To introduce heterogeneity, we distribute the samples such that each client has access to samples from a single class which simulates distinct data distributions for each client (referred to as pathological heterogeneity [116]). For MNIST and F. MNIST, each client has access to 120 training samples; and for CIFAR-10, 250 samples. In the experiments, we update σ in the first iteration instead of the last one and update the global model locally in each local iteration. Details of the models are in Appendix C.6.2.

ADEPT-DGM: In the experiments, we let the number of clients be $m = 50, 30, 6$ respectively for MNIST, Fashion MNIST (F. MNIST), CIFAR-10 due to the increased size of the models. For MNIST, we use a 6-layer U-Net from Hugging Face. For F. MNIST and CIFAR-10 we use 1.7M and 9.4M parameter models respectively. The model with size 1.7M has 3 downsampling and upsampling blocks, and the model with size 9.4M has 4 of them; each of the blocks consists of multiple attention and residual layers. More details are shown in the Appendix F.3. For the MNIST experiments, every client has access to 600 or 1200 samples from one class depending on the experiment; and for CIFAR-10 each client has 2000 samples sampled from 2 classes. We compare to FedAvg+fine-tuning since FedAvg cannot exclusively generate samples from the client's target distribution. We do the same modifications to Algorithm 16 as we did for `ADEPT-AE`. The results for F. MNIST, and more results with different numbers of samples or heterogeneity levels are deferred to Appendix C.6.2.

4.5.2 Results

ADEPT-AE: The results are in terms of the percentage of total energy captured per sample, which is $100(1 - \|x - \hat{x}\|^2/\|x\|^2)$ for each sample x. The results are averaged over 3 runs

Table 4.1 Total energy captured % averaged over samples and clients in the synthetic experiments, where heterogeneity(Het.) is the std of noise and SNR.

Heterogeneity	Method	Energy captured %
0.05	Baseline	88.3 ± 0.5
(6dB)	`ADEPT-AE`	$\mathbf{87.3} \pm 1.1$
	Global Training	81.3 ± 0.1
	Local Training	83.2 ± 1.9
0.025	Baseline	95.4 ± 0.8
(12dB)	`ADEPT-AE`	$\mathbf{95.9} \pm 0.1$
	Global Training	94.3 ± 0.2
	Local Training	83.2 ± 1.9
0.01	Baseline	98.6 ± 0.1
(20dB)	`ADEPT-AE`	$\mathbf{98.7} \pm 0.1$
	Global Training	98.4 ± 0.4
	Local Training	83.2 ± 1.9

and reported together with the standard deviation. Results on synthetic data are shown in Table 4.1, the baseline denotes that each client trains a personalized AE whose decoder part is the true data generating decoder. Our method outperforms local and global training and even the competitive baseline when heterogeneity is smaller. The result is similar to Figure C.1, showing our method outperforms both local and global training in the regimes in which they are strong alternatives. For results on real datasets, the competitive baseline in Table 4.2 is evaluated when 10 clients maintain all the training data from their corresponding classes ($n = 5000$ per client). Remarkably, our method ($n = 250$ per client) matches the baseline on CIFAR-10 (for high latent dimensions), indicating that adaptive personalized collaboration results in $20\times$ effective sample size. For other datasets, our method consistently outperforms FedAvg and local training by an important margin regardless of latent dimensionality. Our method reduces reconstruction error by as much as $\sim 35\%$ and $\sim 25\%$ compared to local training and FedAvg respectively. In MNIST and F. MNIST experiments, `ADEPT-AE` also outperforms all the other methods except the strong baseline we implemented.

Comparison to fine-tuning of regularization parameter: Many of the previous works propose personalization through regularization as outlined in Appendix C.1. Here we compared to pFedMe as one instance, and showed that our method with adaptivity can outperform pFedMe with offline fine-tuning. **ADEPT-DGM:** For diffusion models, we use KID [15] metric to quantitatively measure the quality of generated dataset (see Table 4.3), instead of the commonly used FID [68]; since FID is biased, it tends be unreliable when the generated datasets are small [15] as in a personalized federated setting. At the end of the training, each client generates 200 samples using the model with the lowest validation loss and compares it to the local test dataset to compute the metrics. Our method consistently results in generated samples with better quality and improves upon other methods by $5\% - 22\%$. Moreover, in fig. 4.2 we depict the resulting KID values of the clients. We see that our method brings equity, that is, the worst-performing client is much better compared to the worst clients of other methods, and the performance variance of the clients is lower. We

Table 4.2 Total energy captured % averaged over samples and clients in the real dataset experiments

Dataset	Method	Latent dimensionality	
		Low	*High*
MNIST	Baseline	78.9 ± 0.5	85.7 ± 0.4
	`ADEPT-AE`	$\mathbf{70.8} \pm 0.5$	$\mathbf{77.7} \pm 0.1$
	FedAvg	66.2 ± 0.9	75.9 ± 0.7
	Local Training	67.0 ± 0.8	69.1 ± 1.1
	pFedMe	70.5 ± 0.5	76.5 ± 0.8
F. MNIST	Baseline	88.5 ± 0.2	91.3 ± 0.1
	`ADEPT-AE`	$\mathbf{83.9} \pm 0.2$	$\mathbf{85.6} \pm 0.2$
	FedAvg	81.2 ± 0.2	84.9 ± 0.2
	Local Training	76.9 ± 2.0	77.1 ± 0.5
	pFedMe	83.5 ± 0.6	85.2 ± 0.9
CIFAR-10	Baseline	88.7 ± 0.5	93.3 ± 0.1
	`ADEPT-AE`	$\mathbf{88.4} \pm 0.5$	$\mathbf{93.3} \pm 0.2$
	FedAvg	87.4 ± 0.2	91.2 ± 0.1
	Local Training	87.7 ± 0.2	92.2 ± 0.2
	pFedMe	87.7 ± 0.3	92.4 ± 0.3

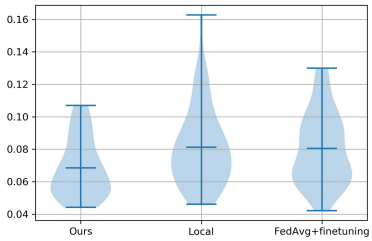


Figure 4.2 Violin plot of KID values of clients.

also illustrate randomly chosen sample images in fig. C.4 in appendix C.6.3 and estimate the amount of noise using [74]. Compared to `ADEPT-DGM`, we observe missing features and inconsistent hallucinations in images obtained using FedAvg+fine-tuning. On the other hand, local training outputs images with significantly more background noise in images (e.g. 1st from the last row and 2nd from the first row), and there is a 1.5× increase in noise level in terms of noise standard deviation ($\sigma = 0.032$ for local training vs $\sigma = 0.024$ for adaptive personalized method). Additional results in appendix F.3: In the Appendix, we provide

Table 4.3 Diffusion model generation quality for generating MNIST samples using U-Net model.

Method	MNIST	CIFAR-10
Baseline	0.062 ± 0.003	–
`ADEPT-DGM`	$\mathbf{0.069} \pm 0.002$	$\mathbf{0.058} \pm 0.003$
FedAvg+fine-tuning	0.090 ± 0.009	0.071 ± 0.006
Local Training	0.083 ± 0.003	0.064 ± 0.004

convergence plots under different heterogeneity for `ADEPT-AE` and `ADEPT-PCA` that validate our theoretical findings. Moreover, we provide additional results for `ADEPT-DGM` in different settings alongside with generated images. We also do ablation studies for sensitivity to hyper-prior ξ, which is low.

4.6 Conclusion

We developed, `ADEPT`, a hierarchical Bayes framework for personalized federated unsupervised learning; leading to new criteria for linear (`ADEPT-PCA`), non-linear (`ADEPT-AE`) dimensionality reduction, and personalized federated diffusion models (`ADEPT-DGM`). Each of our algorithms included adaptation for the heterogeneity during training which resulted in novel theoretical interpretations and superior empirical performance. Open questions include extensions with information constraints such as communication and privacy.

CHAPTER 5

Conditional Personalization for Federated Diffusion Generative Models

Recent advances in diffusion models have revolutionized generative AI, but their sheer size makes on-device personalization—and thus effective federated learning (FL)—infeasible. We propose Shared-backbone Personal Identity Representation Embeddings (`SPIRE`), a framework that casts per-client diffusion based generation as conditional generation in FL. `SPIRE` factorizes the network into (i) a high-capacity global backbone that learns a population-level score function and (ii) lightweight, learnable client embeddings that encode local data statistics. This separation enables parameter-efficient fine-tuning that touches $< 0.01\%$ of weights. We provide the first theoretical bridge between conditional diffusion training and maximum-likelihood estimation in Gaussian-mixture models. For a two-component mixture we prove that gradient descent on the DDPM with respect to mixing weights loss recovers the optimal mixing weights and enjoys dimension-free error bounds. Our analysis also hints at how client embeddings act as biases that steer a shared score network toward personalized distributions. Empirically, `SPIRE` matches or surpasses strong baselines during collaborative pre-training, and vastly outperforms them when adapting to unseen/new clients—reducing Kernel Inception Distance while updating only hundreds of parameters. `SPIRE` further mitigates catastrophic forgetting and remains robust across fine-tuning learning-rate and epoch choices.

5.1 Introduction

Modern diffusion models have achieved striking success in image, audio and video synthesis, yet their ever-growing scale puts them beyond the reach of the low-power devices that generate most of today's data. *Federated learning* (FL) offers a remedy by training a single global model across millions of clients without centralizing raw data. Unfortunately, a one-size-fits-all generator rarely serves individual users well: camera characteristics, usage patterns and personal preferences all induce *client-specific* distributions that differ markedly from the global average. These discrepancies are most pronounced for *new clients*, who arrive after the global model has been trained and possess only a handful of local samples. A key question is, how can we enable a large diffusion model with the flexibility to personalize quickly—even when users' data are scarce and heterogeneous?

Our key insight. Many real-world data sets share a *high-dimensional structure* that is common across clients (e.g., the geometry of natural images) plus a *low-dimensional* client-specific component (e.g., class priors, color statistics or individual identities). We formalize this intuition through a simple heterogeneity model in which shared parameters ϕ captures the complex structure, whereas each client controls lightweight parameters γ_j. For diffusion models we operate the split by (i) training a *shared backbone* that learns a population score function and (ii) injecting a *conditioning embedding* that acts as a signature of client statistics. Personalization thus reduces to jointly learning ϕ from all users and individually learning γ_j—a few hundred parameters instead of millions—making it feasible to adapt on-device. Our key contributions can be summarized as follows.

- We introduce `SPIRE`, a framework that treats per-client diffusion modeling as a conditional generation problem in FL. `SPIRE` decouples global and local learning through an embedding-based conditioning mechanism, enabling parameter-efficient fine-tuning for new clients. who have not participated in the federated pretraining.
- We provide the *first theoretical analysis* connecting conditional diffusion training to maximum

likelihood estimation in Gaussian-mixture models (GMMs). In the two-component case we show that gradient descent on the DDPM objective with respect to mixing weights (which corresponds to low dimensional client specific parameters γ_j) finds the global maximum and derive dimension-free error bounds (Theorem 11). The GMM also provides a theoretical insight into conditioning mechanism in our architecture, including bias to steer personalization.

- We develop a practical FL algorithm that trains backbone and embeddings jointly, requires *no additional communication* during personalization, and is robust to catastrophic forgetting. Our experiments on MNIST, CIFAR-10 and CelebA [106] demonstrate that `SPIRE` matches or exceeds strong baselines during collaborative pre-training and *vastly outperforms* them when adapting to new clients while updating fewer than 0.01% of the parameters.

Below we relate to closest works, but we provide more detailed related work in Appendix D.1.

Personalized FL. There has been very limited work on personalized federated diffusion models, despite the vast amount of work for personalized FL, particularly for supervised learning scenarios. The previous personalized FL works have focused on adjusting the optimization criteria [42, 65, 126], meta learning based approaches [1, 48, 87], clustering based approaches [114], knowledge distillation based approaches [102, 131] and so on. Conceptually, even though exclusively supervised learning is considered, the most related to our work is shared representation approach in [34], where the authors vertically partition neural networks as global feature extractors and local heads. In contrast, our approach is a horizontal partitioning, where high dimensional backbone is trained with light weight conditioning information based on client's data statistics. In terms of the task, [129] also looks at personalized diffusion models, through a lens of statistical hierarchical Bayesian framework. Their method causes a regularization in the loss function, whereas, ours is based on a conditional architecture. Furthermore their method requires two separate models which is less suitable for new clients.

Conditional diffusion models. Conditioning is one of the key properties to add additional information while training diffusion models [41]. The most common way of conditioning

is to embed the information to a vector space and use in intermediate layers as bias (or modulate the activations). Our work explains this use in a principled way and connects it to a theoretical view (see Section 7.4), for the first time, as far as we are aware. We utilize the embeddings to estimate a function that extracts the client data statistics. Furthermore, our work utilizes conditioning mechanism as a way to do PEFT, we are not aware of any other works that consider conditioning for FL and new clients.

Diffusion models for GMMs. The *two-component symmetric Gaussian-mixture model (GMM)* has long served as a testbed for analyzing EM-algorithm convergence (see, e.g., [37, 92]). [151] shows that diffusion models (DDPM loss) can be learned through gradient descent to estimate GMM score function when mixing weights, covariances are known. [32], further shows diffusion models can be used to efficiently learn general GMMs, through piecewise polynomial function approximations. In Section 7.4, our method extends the former; in a distributed and heterogeneous setup, we use gradient descent to learn the mixing weights of the GMMs and prove dimension-free bounds.

5.2 Problem setup: Personalized federated diffusion generative models

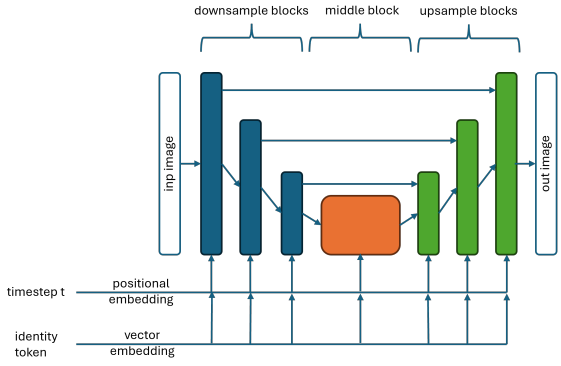


Figure 5.1 Overall diffusion model architecture.Each client holds an identity token which is mapped to a vector embedding that is trained throughout with the backbone. During the inference, a client can use its identity token to do a personalized generation.

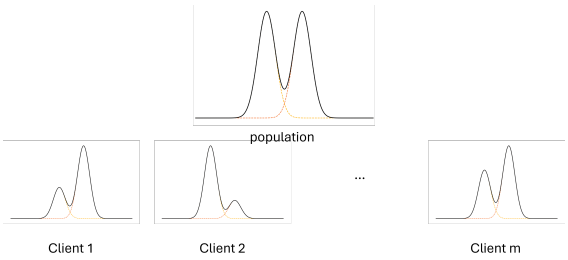


Figure 5.2 In GMM heterogeneity model, all clients share the same means but the mixing weights are individual.

In this section, we first introduce a heterogeneity model of clients' data that is partitioned into two: a complex common structure and a lower dimensional individual structure that is specific to clients. A concrete example is a mixture model where high dimensional cluster centers are common for every client but mixing weights are individual (e.g. see Figure5.2). Later, we will explain (conditional) diffusion models for personalization and the overall federated learning setup that we consider. Lastly, we will explain how the formulation is especially useful for new clients.

5.2.1 Problem setup and background

We consider a conditional personalization approach for learning individual diffusion models tailored to local data from clients. Let us denote $q^{(j)}(x)$ as the data distribution of a particular client j. We assume $q^{(j)}(x)$ is implicitly conditioned on two types of variables, i.e. $q^{(j)}(x) = q^{(j)}(x|\phi, \gamma_j)$ where $\phi \in \Phi$ is a high dimensional shared parameter, and $\gamma_j \in \Gamma$ is a lower dimensional parameter that is specific to a client j. Note that through the Bayes rule, the pdf implies a mapping from data distribution of a client and the shared parameters to γ_j, there is a (random/data dependent) mapping such that $\gamma_j = g(x, \phi), x \sim q^{(j)}$. To motivate, we give a theoretical and an empirical example.

Example 1. *Consider data generated by a two component GMM with mean, mixing weights, and covariance parameters* $(\pm\mu, \{w_j\}_{j=1}^m, \Sigma)$*, each client shares the high dimensional mean*

parameter μ but individually possesses different mixing weights w_j of the components. In this example ϕ, γ_j corresponds to μ, w_j respectively.

Example 2. *Consider a non-parametric score function that is approximated through a neural network with parameters $\boldsymbol{\theta}_j$, for a client j. Potentially, $\boldsymbol{\theta}_j$ can be partitioned into high dimensional $\boldsymbol{\theta}_{bb}$ that is learned collaboratively and low dimensional $\boldsymbol{e}_j$ that is learned locally. The goal is to use $\boldsymbol{\theta}_{bb}, \boldsymbol{e}_j$ as proxies for ϕ, γ_j respectively. Ideally, $\boldsymbol{\theta}_{bb}$ should describe a general structure in generation task, and $\boldsymbol{e}_j$ should add personalization according to input data statistics.*

The goal of the personalized diffusion models is to learn $q^{(j)}(x)$ that is the individual conditional data distribution of a client given ϕ, γ_j. In federated learning, every client holds a small dataset $\{X_i^{(j)}\}_{i=1}^n \sim q^{(j)}$ (assuming same n for simplicity), yet the globally data is from m participating clients, where m is assumed to be larger tan n. This abundance enables collaborative learning of the complex ϕ, while each client estimates its low-dimensional γ_j locally. New clients, absent from pre-training, can fine-tune only γ_j with few samples, achieving sample-efficient personalization.

Diffusion Models. Conventionally, diffusion models are composed of several parts. First, a *forward process* that mathematically describes the gradual transformation from a true sample to noise. A corresponding *backward process* is necessary to formalize how the score function can be used to denoise a noisy sample to obtain a data point from the true distribution. Lastly, since the true score function is unknown one needs to formulate an optimization problem (i.e. *score matching*) to estimate it to be used in the backward process. Due to space limitations we define the original diffusion model in Appendix D.2. Here we introduce the personalized diffusion models. In the personalized FL setup, we would like to generate samples from clients' individual probability distributions. We first start with the forward process. The forward diffusion process is defined as

$$\mathrm{d}X_t^{(j)} = -\delta_t X_t^{(j)}\,\mathrm{d}t + \sqrt{2\delta_t}\,\mathrm{d}W_t, \quad X_0^{(j)} \sim q_0^{(j)}(x). \tag{5.1}$$

here $q_0^{(j)}(x)$ denotes the true underlying distribution that generates client js data, similarly $q_t^{(j)}(x)$ will be used to denote a latent distribution, δ_t is a drift coefficient, commonly assumed a constant e.g. 1. W_t is the standard Brownian motion. $X_t^{(j)} \sim q_t^{(j)}$ is a latent variable. Equivalently, it can be shown [151] for some $\beta_t : \lim_{t\to\infty} \beta_t = 1, \beta_0 = 0$,

$$X_t^{(j)} = \sqrt{1-\beta_t^2} X_0^{(j)} + \beta_t Z_t, \text{with } X_0^{(j)} \sim q_0^{(j)} \text{ and } Z_t \sim \mathcal{N}(0, \mathrm{I}). \tag{5.2}$$

Corresponding to the forward process, is the backward SDE process, which is defined to formalize recovering a data sample from true distribution, given a corrupted sample and score function,

$$\mathrm{d}\overleftarrow{X}_t^{(j)} = \left[\delta_{T-t}\overleftarrow{X}_t^{(j)} + 2\delta_{T-t}\nabla \log q_{T-t}^{(j)}(\overleftarrow{X}_t^{(j)})\right]\mathrm{d}t + \sqrt{2\delta_{T-t}}\,\mathrm{d}W_t, \quad \overleftarrow{X}_0^{(j)} \sim q_T^{(j)}(\cdot). \tag{5.3}$$

In the backward process, we use $\overleftarrow{X}_0^{(j)}$ to denote a sample from pure noise $q_T^{(J)}$. $\nabla \log q_{T-t}^{(J)}(\overleftarrow{X}_t^{(j)})$ is the true score function at time $T-t$ (or at time t in terms of the forward process). Of course the true score function is unknown, hence we need to estimate it from the data using a fixed pure noise distribution. To that end, one constructs the score matching optimization criterion,

$$\min_\theta \int_\tau^T \mathbb{E}_{X_t^{(j)} \sim q_t^{(j)}(\cdot)} \left[\left\| s_{\theta_j}(x_t, t) - \nabla \log q_t^{(j)}(X_0^{(j)}) \right\|^2\right] \mathrm{d}t, \tag{5.4}$$

where θ_j is a neural network that is assumed to parameterize the score function. Equivalent to score matching is the DDPM view, which yields [31],

$$\min_\theta \int_\tau^T \mathbb{E}_{X_0^{(j)} \sim q_0^{(j)}(\cdot), Z_t} \left[\left\| s_{\theta_j}(X_t^{(j)}, t) + \frac{Z_t}{\beta_t} \right\|^2\right] \mathrm{d}t. \tag{5.5}$$

5.3 SPIRE: Personalized FL using conditional diffusion models

In our work, we consider using client's data (through learning its embedding) as the conditioning information to the model. This results in an horizontal split of the model. During pretraining, $\boldsymbol{\theta}_{bb}, \{\boldsymbol{e}_j\}_j$ are trained jointly, hence, the model implicitly learns to estimate the function $\boldsymbol{e}_j = g^{(j)}(x^{(j)}, \boldsymbol{\theta}_{bb})$. One can see that finding the embedding in the diffusion model training

corresponds to extracting data statistics, in other words, $g^{(j)}(x, \theta_{bb}) = \arg\min_{e_j} L_{ddpm}(q^{(j)}, \theta_j)$, where L_{ddpm} is the score matching loss in (5.6) and $x^{(j)} \sim q^{(j)}(x)$.

Form of $\boldsymbol{\theta}_j$. Our heterogeneity modeling yields separability of $\boldsymbol{\theta}_j$ into two parts. A more complicated (higher dimensional) backbone part $\boldsymbol{\theta}_{bb}$ that is shared among estimated scores of different clients, and a lower dimensional embedding of clients' data statistics, $\boldsymbol{e}_j$, that is individual to each client. We can write $\boldsymbol{\theta}_j = [\boldsymbol{\theta}_{bb}, \boldsymbol{e}_j]$. During training, overall personalized federated DDPM objective consists of finding m different models (with shared backbone and individual identity embedding in our case), $\min_{\{\theta_j\}_{j=1}^m} \frac{1}{m} \sum_{j=1}^m \int_\tau^T \mathbb{E}_{X_0 \sim q_0^{(j)}(\cdot), Z_t} [\| s_{\theta_j}(X_t^{(j)}, t) + \frac{Z_t}{\beta_t} \|^2] \, \mathrm{d}t$. Empirical version of the DDPM loss can be defined through averaging over local dataset.

$$\min_{\{\theta_j\}_{j=1}^m} \frac{1}{m} \sum_{j=1}^m L_{ddpm}^{(j)}(\{X_i^{(j)}\}_{i=1}^n, \theta_j) := \frac{1}{n} \sum_{j=1}^n \int_\tau^T \mathbb{E}_{Z_t} \left[\left\| s_{\theta_j}((X_i^{(j)})_t, t) + \frac{Z_t}{\beta_t} \right\|^2 \right] \mathrm{d}t. \tag{5.6}$$

In practice instead of taking expectation over timesteps, we sample randomly. A client, then, can plug in the estimated score (s_{θ_j}) into the backward process (5.3) to generate personalized samples.

5.3.1 Federated training setup.

During training, each client maintains a personalized embedding (through a identity token, e.g., client id number) which acts as a key for the backbone model. The embeddings are kept in client and trained locally, and backbone model is shared and trained by essentially FedAvg. Algorithm6 summarizes the key steps.

Personalization to new clients. Personalization for new clients is one of the key challenges in federated learning. It is especially, pronounced for generative models due to scale of the models. A new client joining the federated ecosystem, later than pre-training stage, can utilize the pre-trained $\boldsymbol{\theta}_{bb}$ and train only its own embedding parameters. In a way, this defines a natural way of parameter efficient fine-tuning (PEFT) for personalization, where we are estimating $q^{(new)}$ through $(\boldsymbol{\theta}_{bb}, \boldsymbol{e}_{new})$. See Algorithm 17 in Appendix D.2, for

new client fine-tuning details.

Algorithm 6 Conditional collaborative personalized pre-training for `SPIRE`

Input: Number of iterations K, learning rate (η), batch size b, number of local iterations τ

1: **Initialize** backbone model $\boldsymbol{\theta}_{bb}$, and client embeddings $\{\boldsymbol{e}_j\}_{j=1}^m$, set $\boldsymbol{\theta}_j = [\boldsymbol{\theta}_0, \boldsymbol{e}_i]$.
2: **for** $k = 1$ **to** K **do**
3: **for** $i = 1$ **to** m **do**
4: **if** τ divides k **then**: Receive $\boldsymbol{\theta}_{bb,k}$ from server and set $\boldsymbol{\theta}_{j,k} = [\boldsymbol{\theta}_{bb,k}, \boldsymbol{e}_{j,k}]$
5: Sample $(\{X_i\}_{i=1}^b, \{t_i\}_{i=1}^b)$, use (D.2) to obtain noisy samples $\{X_{t_i}\}_{i=1}^b$
6: Take gradient step on local DDPM loss $\boldsymbol{\theta}_{j,k+1} = \boldsymbol{\theta}_{j,k} - \nabla_{\theta_j} L_{ddpm}^{(j)}(\{X_{t_i}\}_{i=1}^b)$
7: **if** τ divides $t+1$ **then:** extract updated $\theta_{j,bb,k+1}$ from $\theta_{j,k+1}$ and send to server
8: **end for**
9: On server aggregate $\theta_{bb,k} = \frac{1}{m}\sum_{j=1}^m \theta_{j,bb,k}$, and broadcast to clients
10: **end for**

Algorithm 6 Line 1 initializes the shared backbone parameters $\boldsymbol{\theta}_{\text{bb}}$ and the per-client embeddings $\{\mathbf{e}_j\}$. Whenever the synchronization test $\tau \mid k$ succeeds, each client receives the latest backbone (line 5). During each local step the client samples a mini-batch and timesteps, and corrupts the data with the forward process (line 6), and performs a DDPM gradient update on the full parameter vector (line 7). At the next synchronization point, sends the updated backbone slice to the server (line 8), and keeps $\mathbf{e}_{j,k+1}$ locally. The server averages the received slices (line 10) to produce $\boldsymbol{\theta}_{\text{bb},k+1}$, which is broadcast at the start of the next round.

5.4 Theoretical justification: connection to learning mixing weights of GMMs

Recent work has revisited the same model with *fixed* mixing weights to probe the theoretical foundations of diffusion models and score matching [151]. In our setting we change the emphasis to a distributed setup. The mean vector $\boldsymbol{\mu} \in \mathbb{R}^d$ is shared across all clients, while each client j retains its own mixing weight $w_j \in (0, 1)$. This maps directly onto our heterogeneity framework, where a global parameter (the shared mean) co-exists with lightweight, client-specific parameters (the scalar weights). Conceptually, it reduces to a one-layer version of our personalized architecture. Detailed proofs of the results in this section is deferred to Appendix D.2.

5.4.1 Overview

We focus on a symmetric GMM with two equal components *in the population level* and unknown mixing weights *in client level* as in Figure 5.2. In the personalization setup our model consists of shared means $\pm\mu \in \mathbb{R}^d$, and individual mixing weights of the components $\{(w_j, 1 - w_j)\}_{j=1}^m$. In particular, the true data distribution for a particular client j is,

$$q^{(j)} = w_j \mathcal{N}(\mu, \mathrm{I}) + (1 - w_j)\mathcal{N}(-\mu, \mathrm{I}) \tag{5.7}$$

where $q^{(j)}$ is the GMM distribution, I is the identity matrix of size $d \times d$, $\mu \in \mathbb{R}^d$ is the true mean parameter, $w_j \in \mathbb{R}$ is the mixing weight for the positive component for client j. This particular mixture distribution is studied in literature, mostly when mixing weights are known and equal, i.e. 1/2 [151]. Below, we consider the setup for a particular client and omit the subscript in w_j.

Noisy samples. Utilizing the same OU forward process as in [151] and above in (D.1), it can be shown (see Appendix D.2) for $X_t \sim q_t$ we have, $X_t = \exp(-t)X_0 + \sqrt{1 - \exp(-2t)}Z_t$, with $X_0 \sim q_0$ and $Z_t \sim \mathcal{N}(0, \mathrm{I})$. In the analysis, we will fix a particu-

lar timestep t. Accordingly, we will focus on the equivalent DDPM objective:

$$\min_{s_t} \mathbb{E}_{X_0, Z_t} \left[\left\| s_t(X_t) + \frac{Z_t}{\sqrt{1 - e^{-2t}}} \right\|^2 \right]. \tag{5.8}$$

Recall, in our case of two component mixture we have, $q_0 = w\mathcal{N}(\mu, \mathrm{I}) + (1-w)\mathcal{N}(-\mu, \mathrm{I})$. Accordingly, q_t can be easily obtained to be (see Appendix D.2)

$$q_t = w\mathcal{N}(\mu_t, \mathrm{I}) + (1-w)\mathcal{N}(-\mu_t, \mathrm{I}), \text{ where } \mu_t := \mu \exp(-t).$$

The score function has a closed form solution as $\nabla_x \log q_t(x) = r_1(x)\mu_t + r_2(x)(-\mu_t) - x$ where $r_1(x) = \frac{w \exp(-\|x-\mu_t\|^2/2)}{w \exp(-\|x-\mu_t\|^2/2) + (1-w)\exp(-\|x+\mu_t\|^2/2)}$, and $r_2(x) = 1 - r_1(x)$ are conditional probability of being assigned to a cluster, i.e. 'beliefs'. Accordingly we can obtain the closed form of the score function as outlined in Lemma 4 (see Appendix D.2 for the proof).

Lemma 4. *The score function at diffusion time step t for q_t is defined as follows ,*

$$\nabla_x \log q_t(x) = \tanh\left(\mu_t^\top x - \frac{1}{2}\log(w/(1-w))\right)\mu_t - x. \tag{5.9}$$

Connection to conditional architecture. The expression in Lemma 4 can be viewed as a *single-layer neural network with a residual (identity) connection.* The shared mean vector $\boldsymbol{\mu}_t$ serves as the layer's weight vector; the log-odds term $\frac{1}{2}\log\left(\frac{w}{1-w}\right)$ is an explicit bias that shifts the pre-activation $\boldsymbol{\mu}_t^\top x$. After applying the non-linearity $\tanh(\cdot)$, the network outputs a vector aligned with $\boldsymbol{\mu}_t$, which is then combined with the skip path $-x$. This construction is exactly the one-layer special case of the conditional architecture introduced in the previous section: the conditioning signal is carried solely by the client-specific mixing weight w, while the weights $\boldsymbol{\mu}_t$ remain global. Hence the lemma illustrates, in its simplest form, how general conditional diffusion models encode per-client information through mixing-weight-induced biases. Training a conditional model can therefore be seen as *finding the appropriate mixing weights* that encode each condition, leaving the heavy shared parameters untouched.

Consequently, we consider (5.9) as the network architecture ($s_t(X_t)$ parameterized by μ_t, w) to estimate the score function. In the personalized setup, we are interested in a

fine-tuning stage where the means are known and each client learns the individual mixing weight. As a result the problem of interest can be rewritten as,

$$\min_{w} \mathbb{E}_{X_0 \sim q^{(j)}(x), Z_t} \left[\left\| \tanh\left(\mu_t^\top X_t - \frac{1}{2}\log((1-w)/w)\right)\mu_t - X_t + \frac{Z_t}{\sqrt{1-e^{-2t}}} \right\|^2 \right]. \tag{5.10}$$

Our goal is to examine the MSE at the convergence point; to that end, we relate the first order convergence condition with EM updates [92], which are provided in Appendix D.2.

5.4.2 Theoretical results

Here, first, we discuss the resulting bounds in the two component GMM problem, for learning the mixing weights. Then we look at score estimation error in a two step procedure where the mean is estimated globally, and then each client learns the mixing weight locally.

Theorem 11. *Fixing a μ_t, MSE of estimating w by a gradient method on the DDPM loss results in,*

$$\mathbb{E}_{X_0}(w - \hat{w})^2 \leq \frac{w(1-w)}{n} + \frac{d}{4\|\mu_t\|^2 n}$$

Proof Outline. We start by taking the derivative of (5.10) with respect to w at a particular timepoint t. Let $u_t = \mu_t^\top X_t - \frac{1}{2}\log(w/(1-w))$, $f(X_t) = \tanh(u) - X_t + \frac{Z_t}{\sqrt{1-e^{-2t}}}$, by the chain rule, $\frac{\mathrm{d}L}{\mathrm{d}w} = \frac{1}{w(1-w)}\mathbb{E}\left[\tanh'(u_t)(\|\mu_t\|^2 \tanh(u_t) + \mu_t^\top X_t)\right]$, where L is the shorthand notation for local DDPM loss. Hence, at the critical point we have the following condition being satisfied,$\mathbb{E}[\tanh(u_t)] = -\frac{\mu_t^\top \mathbb{E}[X_t]}{\|\mu_t\|^2}$. Now looking at the EM algorithm and in particular M-step, we see that $w^+ = \mathbb{E}[r_1(X_t)] = \mathbb{E}[\frac{1}{2}(1 + \tanh(u_t))]$, hence, plugging the convergence point of score loss into EM algorithm, $w^+ = \frac{1}{2}(1 - \frac{\mu_t^\top \mathbb{E}[X_t]}{\|\mu_t\|^2})$, in other words EM algorithm is also converged, equivalently, $\mathbb{E}[X_t] = \mu_t(2w-1)$ which is exactly the first order moment matching condition. In other words, convergence of score estimation loss corresponds EM algorithm convergence that satisfies the first order moment equality, since the means and covariances are known this corresponds to the global maximum of the likelihood. At the convergence

point we can measure the mean squared error of estimating the mixing weight from samples. In particular, let us define $\hat{w} = \frac{1}{2}(1 - \frac{\mu_t^\top \hat{X}_t}{\|\mu_t\|^2})$ as the sample mixing weight, where $\hat{X}_t$ is the sample mean at time t. Then the L_2 error can be found to be $\mathbb{E}(w - \hat{w})^2 \leq \frac{w(1-w)}{n} + \frac{d}{4\|\mu_t\|^2 n}$.

Remark 1. The bound on mixing-weight error is dimension-free because both its numerator and denominator scale with d, unlike mean estimation, whose difficulty grows in high dimensions. Thus clients with few samples can reliably learn their own weights. The bound has two components: (i) sampling term—the variance of cluster assignments; it shrinks when one mixture component dominates, (ii) separation term—inversely proportional to $\|\mu_t\|^2$; larger mean separation improves accuracy, while small separation can be offset by more data. Crucially, the bound depends only on the norm $\|\mu_t\|$—not on how accurately the mean is estimated—because correct assignment, not mean precision, drives weight estimation. Overall generation quality will, however, reflect errors in both $\hat{\mu}$ and$\hat{w}$. With the mixing-weight bound in hand, we next assess score error: we measure the mean-squared gap between the true score and that produced by the converged estimators, where the global backbone learns $\hat{\mu}_t$ via DDPM gradient descent [151] and each client refines its own $\hat{w}$ using our personalized procedure.

Theorem 12. *Let $L_{est}(\hat{\mu}, \hat{w})$ be the score matching loss between the true score function and estimated score function where the $\hat{\mu}_t$ is estimated globally using the procedure in [151], and mixing weights are estimated individually by doing gradient steps in the personalized DDPM loss. Assuming B is a constant such that $\|\mu_t\|^2 \leq B$. Then, the loss has the following error,*

$$L_{est}(\hat{\mu}, \hat{w}) = \mathcal{O}\left(\frac{d^6 B^8}{mn(1 - e^{-2t})^2}\right) + \mathcal{O}\left(\frac{1}{n}\right) \tag{5.11}$$

Proof Outline. We start by utilizing Lipschitz continuity of tanh and log functions, and some algebraic manipulations yield, $L_{est}(\hat{\mu}, \hat{w}) \leq 4B(d + (4w(1-w)+1)B)\mathbb{E}\|\mu - \hat{\mu}\|^2 + \frac{1}{2w_{min}^2(1-w_{min})^2}\frac{w(1-w)}{n}$. Converting the high probability bound in [151] to an expectation bound we have, $\mathbb{E}\|\mu_t - \hat{\mu}_t\|^2 \leq C\frac{d^5 B^6}{mn\beta_t^2}$, for constant $C > 0$. Combining with earlier bound concludes the proof outline.

Remark 2. Theorem 12 shows that score-estimation error is governed by errors in both the mean and the mixing weight. Except when the client count reaches $\Theta(d^6B^8)$, the mean-estimation term dominates. Because mixing-weight error is dimension-free and small even with limited data, a client can accurately personalize once a good global mean is pretrained. This shows the benefit of our split.

5.5 Experiments

Baselines. As diffusion models require very large U-Net models to be trained, personalized FL methods with two different models, i.e. separate individual and global models, are not feasible. Hence, we compare our method to single model methods. One such baseline is the simple *FedAvg+fine-tuning* for personalization, though it i s a simple approach it is accepted as one of the competitive personalization methods. Another baseline is *meta-learning* approaches, specifically [48], where the global model is minimizing a meta-learning based loss that allows for better personalization. Lastly, inspired by [34], we introduce a *shared representation* approach, as a competing solution to compare with, where downsample and upsample blocks are trained globally and middle block is trained individually. The goal of this such partition is to enable global learning of feature extraction (downsample blocks) and feature generation (upsample blocks) similar to global feature extraction in [34]; however, in classifier models [34] trains individual classifier heads which is not directly applicable to U-Net models. Inspired by that, we enable individual learning of middle block to introduce personalization in the latent space. We provide more details on implementation details of baselines in Appendix F.3.

Datasets. We randomly generate a synthetic dataset (using the true underlying score function) for synthetic experiments to estimate score function (mean and mixing weight) of GMM. For real datasets, we use MNIST, CIFAR-10/100, and CelebA (celebrity faces) with increasing complexity. The images have 28x28, 32x32, and 64x64 resolutions respectively.

For pretraining on MNIST, there are 30 clients and each client has access to 400 samples from a randomly selected class; for fine-tuning on new clients, there are 30 clients each client has access to 400 samples from 3 different classes which simulates another layer of challenge, namely the new clients have samples from a different mixture dataset compared to the pretraining clients. On CIFAR-10, there are 18 clients each with 2750 samples from a randomly selected class. The new clients have samples from two randomly selected classes, but the amount of data at each client drops to 275 to simulate a data scarcity setting. On CelebA dataset there are 24 clients each with data coming from 5 different individuals, the total amount of data per client is only around 25. This setting is very challenging and acts as a stress test for our method. The new clients have data from different individuals compared to pre-training data.

Models. We use a conditional U-Net architecture where client id's, that are converted to vector embeddings, are used as the conditioning information. We use 14 layer, 6M parameter architecture for MNIST experiments, and 18 layer 13M parameter for CIFAR-10, CelebA experiments. More details can be found in Appendix. We use AdamW [109] optimizer and provide more details on hyper-parameters in Appendix F.3.

Table 5.1 Pre-training performance for participating clients, and fine-tuning performance for newly joined client performance KID↓ values of respective methods on respective datasets.

Method	Pre-training			New client fine-tuning		
	MNIST	CIFAR-10	CelebA	MNIST	CIFAR-10	CelebA
FedAvg+fine-tuning	0.015	0.034	0.051	0.028	0.101	0.075
SPIRE	**0.010**	0.034	**0.046**	**0.012**	**0.038**	**0.061**
Shared Representation	**0.010**	**0.031**	0.056	0.017	0.048	0.094
Meta learning	0.061	0.073	0.081	0.079	0.093	0.120

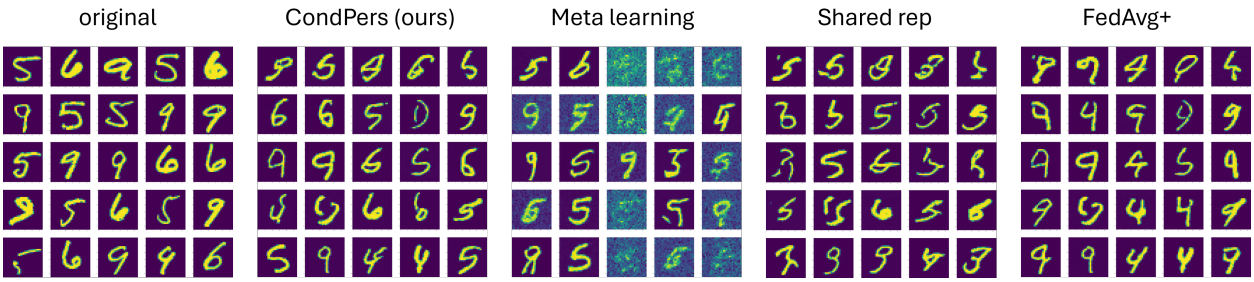


Figure 5.3 Grid of original MNIST images and generated images by competing methods for particular client, who has sampled data from classes '5,6,9; using the same random seed. Our method results in less hallucinations, more diversity while preserving structure.

5.5.1 Numerical Results

Collaborative pre-training. The main image quality metric we use is Kernel Inception Distance (KID) [15], as FID [68] is not usable for smaller datasets due to biasedness. Table 5.1 reports KID after the collaborative pre-training phase. On MNIST and CelebA our `SPIRE` achieves the lowest KID, improving over baselines. Particularly, the gap is significant on CelebA dataset. On CIFAR-10 `SPIRE` matches *FedAvg+FT* while the *Shared-Rep.* variant edges out both methods by a small margin. We attribute the CIFAR-10 result to the fact that all clients observe only a single class during pre-training—global feature extractors alone already generalize reasonably well in that setting. Across all three data sets the meta-learning baseline struggles, confirming the practical difficulty of applying first-order MAML–style algorithms to large diffusion models.

Generalization to new clients. The advantage of `SPIRE` becomes far more pronounced once entirely new clients—holding data from unseen class mixtures or unseen identities—join the federation (Table 5.1,b). After less than 10 epochs of local fine-tuning `SPIRE` yields large gains over the next best method Shared representation. Neither *FedAvg+FT* nor meta-learning can close the gap; in fact, both are outperformed by `SPIRE` on every data set. These results support our central claim: *embedding the client identifier as a learnable conditioning signal equips a* single *diffusion model with enough capacity to adapt to novel*

client distributions.

Qualitative analysis. Figures 5.3 and D.1 (in Appendix F.3) visualize samples drawn from models trained on CelebA and MNIST, respectively, for a particular client. For MNIST, the client has access to images labeled as '5','6', and '9'. For the most of the images `SPIRE` is able to generate 5,6,9 images with occasional hallucination from other classes. MAML approach fails to generate quality images. Shared representation approach often misses the structure of the numbers, and FedAvg+FT does not generate diverse images implying an overfitting to the data. For CelebA , all methods struggle with the coloring of images due to the challenge in pretraining task. Images generated by `SPIRE` exhibit noticeably sharper facial details. Competing methods over-fit, producing more washed-out or saturated outputs. The faces generated by our method are more diverse and have better visual features e.g. shadowing.

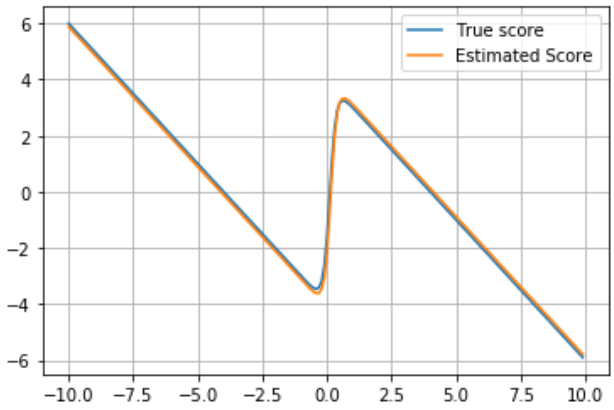


Figure 5.4 Estimated score function.

Synthetic experiments for GMMs. For synthetic experiments, we are verifying correctness of our found score function form in (5.9). Given the objective in (5.10), we use Adam algorithm to update both the mean and mixing weights. The results in the Figure indicate convergence of training and success of estimating the score through the DDPM objective. See Appendix for experimental details.

Conditional personalization is robust to catastrophic forgetting. A major

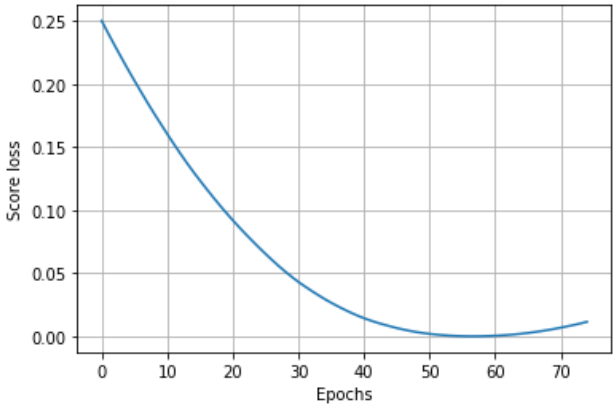


Figure 5.5 Score loss convergence.

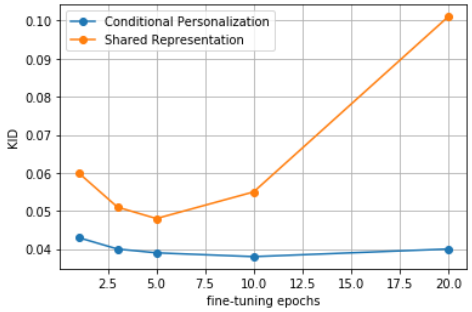


Figure 5.6 The number of local epochs need to be carefully fine-tuned for new clients using shared representation approach due to overfitting and underfitting risk, on the other hand conditional personalization performs better and is more robust with respect to number of local epochs..

advantage of our method, in new client fine-tuning, compared to competing methods is its robustness to overfitting, i.e. catastrophic forgetting. Since our method relies on updating only a small part of the model, previously learned information through the backbone has no risk of getting lost during the fine-tuning stage. This prevents an oversensitivity towards number of fine-tuning epochs or learning rate; which are mainly responsible for a potential forgetting. To display this notion, we compare to shared representation approach and illustrate the resulting KID values for new clients under different finetuning epochs Figure 5.6.

Overall. In the experiments we showed the efficacy of our proposed way of personalization. `SPIRE` is especially superior to other methods for new client fine-tuning which is a very crucial

problem in FL and, in large scale ML. Due to the natural PEFT implied by our method, it is robust to choices such as learning rate and number of iterations. It does not carry risks such as catastrophic forgetting.

5.6 Conclusion

We introduced SPIRE, the first framework that treats personalized diffusion in federated learning as a *conditional generation* problem. By decomposing the model into a global backbone and lightweight client embeddings, SPIRE enables parameter-efficient, communication-free adaptation on devices that hold only a handful of samples. Our theory establishes a link between gradient descent on the DDPM objective and Gaussian-mixture models, yielding dimension-free error bounds for the client-specific parameters. Empirically, SPIRE consistently matches or surpasses strong baselines during collaborative pre-training and delivers superior performance for new-client personalization, resulting quantitatively and qualitatively better generation. Future directions include (i) extending the embedding-based conditioning to multi-modal and text-to-image diffusion, (ii) integrating differential-privacy guarantees, and (iii) exploring hierarchical embeddings that capture groups of clients. Main limitation of our work is that it requires changing pre-trainig recipe to be done in conditioned on clients, hence, another future work that we would like to explore is integrating SPIRE into large pre-trained diffusion models such as StableDiffusion [147]. We hope our results spur further research on principled, efficient personalization for large generative models in federated settings. We discuss broader impact of our work in Appendix D.1.

CHAPTER 6

Meta-Adaptive Optimizers through Hyper-Gradient Descent

Following the introduction of Adam, several novel adaptive optimizers for deep learning have been proposed. These optimizers typically excel in some tasks but may not outperform Adam uniformly across all tasks. In this work, we introduce Meta-Adaptive Optimizers (`MADA`), a unified optimizer framework that can generalize several known optimizers and dynamically learn the most suitable one during training. The key idea in `MADA` is to parameterize the space of optimizers and dynamically search through it using hyper-gradient descent during training. We empirically compare `MADA` to other popular optimizers on vision and language tasks, and find that `MADA` consistently outperforms Adam and other popular optimizers, and is robust against sub-optimally tuned hyper-parameters. `MADA` achieves a greater validation performance improvement over Adam compared to other popular optimizers during GPT-2 training and fine-tuning. We also propose AVGrad, a modification of AMSGrad that replaces the maximum operator with averaging, which is more suitable for hyper-gradient optimization. Finally, we provide a convergence analysis to show that parameterized interpolations of optimizers can improve their error bounds (up to constants), hinting at an advantage for meta-optimizers.

6.1 Introduction

The choice of an optimization algorithm plays a critical role in determining the downstream performance of a machine learning model. Adaptive moment optimizers are the most preferred

class of optimizers employed in most learning tasks such as training Large Language Models (LLMs) [19, 166] and Diffusion Models [146]. In particular, Adam [88] is still the "go-to" optimizer in LLM training, despite the emergence of many other optimizers since then.

Recently proposed optimizers [33, 50, 103, 174, 179] report improved performance compared to Adam in specific tasks. However, it is unclear if their strong performance generalizes across a wide range of tasks as Adam's does [149]. It is also unclear if a single optimizer can be uniformly the best across all learning tasks and training regimes, such as different batch sizes, hyper-parameters, and datasets.

In this work, we introduce the concept of a *parameterized optimizer*, which can be viewed as a unified parameterization of a collection of given optimizers. The parameters of a parameterized optimizer define a convex polytope (e.g. hypercube), whose vertices may correspond to individual base optimizers, while the interior represents new optimizers formed by interpolating between them. To make this concept operational, we propose the meta-adaptive optimizer (`MADA`), which combines the parameterized optimizer with hyper-gradient descent [13] to learn the optimizer coefficients. `MADA` dynamically adjusts the parameterized optimizer coefficients *during* training, and effectively adapts the optimizer choice to the learning task. While `MADA` bears some connections to optimizer search methods, such as [33], it does not solely output a final optimizer state. It dynamically adapts the optimizer to the current neighborhood of the loss landscape on-the-fly, removing the need for an offline optimizer search stage, or an outer hyper-parameter selection loop.

Contributions. We make the following contributions:

- We introduce the concept of a parameterized optimizer, which takes a collection of existing optimizers, and unifies them into a single optimizer, combining the individual update rules through learnable coefficients.
- We propose `MADA`, a meta-optimization framework that combines parameterized optimizers with hyper-gradient descent to learn a specific optimizer instance during

training.

- We find that not all optimizers are suitable to use in a hyper-gradient optimization framework. Specifically, among popular optimizers, using AMSGrad [143] results in poor performance within `MADA` due to its use of the maximum operator. Motivated by this, we propose a modification of it called AVGrad, which replaces the maximum operator on the second-order moments with time-averaging, and leads to better performance when used as part of `MADA`. We also analyze its convergence properties (see Appendix E.2). We show that `MADA` converges to AVGrad in a simple example where it is known to perform better than Adam, providing evidence for the effectiveness of `MADA` in adapting the optimizer to the task (see Appendix F.3).
- To demonstrate that optimizer interpolations can improve convergence bounds in an analytically tractable scenario, we theoretically analyze the convergence behavior for interpolations between two optimizers, namely AVGrad and Adam. Our analysis shows that the interpolated optimizer improves the convergence bounds of the base optimizers up to constant factors.
- We develop a specific parameterized optimizer, which interpolates between Adam [88], AVGrad, Yogi [181], Adan [174], and Lion [33]. On language tasks we compare `MADA`, which is based on this parameterized optimizer, against Adam, Lion, Adan, and HyperAdam[1], and show that `MADA` consistently outperforms all baselines. We also illustrate the robustness of `MADA` to initial hyper-parameters and analyze the evolution of hyper-parameters during training. On vision tasks we compare `MADA` to Adam, SGD with momentum, and HyperAdam, and observe consistent performance improvement.

Related work. Motivated by the high cost of training large language models, a large number of novel optimizers have been proposed in recent years to speed up the training, increase generalization performance or train more resource-efficient models. As mentioned

[1]Throughout this paper we will refer to the version of Adam that uses hyper-gradients to tune β_1 and β_2 parameters, as in [25], as HyperAdam.

before, Adam [88] is the most commonly employed optimizer, and succeeding methods, in general, try to improve upon it under different scenarios. [143, 181] propose AMSGrad and YOGI, respectively, to fix Adam's potentially non-decreasing effective learning rate. [43, 174] introduce Adan and Nadam, respectively, to replace heavy-ball momentum in Adam with Nesterov momentum. [178, 179] propose LARS and LAMB to improve the performance of Adam in large-batch regime. [67] proposes AdamP to avoid premature decay of scale-invariant weights. AdaBound [111] and AdaBelief [188] try to estimate a more stable second-order moment term. [50] proposes sharpness-aware minimization (SAM) and [29] proposes Padam to increase the generalization performance of the trained models. [104] stabilizes the training by reducing gradient variance. The work in [33] is similar in spirit to our work, in that it introduces a method to symbolically search a space of optimizers; however unlike our work they consider an offline search method, whereas our method learns the optimizer *during* actual model training, and uses hyper-gradients.

A separate line of work proposed learned optimizers to delegate the optimization task to neural networks (fully connected or LSTMs) [4, 118, 119]. Instead of directly using first order information as in optimizers, these methods treat gradient information, alongside other features such as training loss, validation loss [4] and so on, as inputs to a neural network which outputs the update to be applied on a particular weight. Learned optimizers require resource-intensive offline training (e.g. thousands of TPU-months in [118]) since the updates are handled through a separate neural network that needs to be trained before deployment. Integration of a separate neural network to the optimization process introduces additional complexity, which `MADA` avoids by requiring only a few parameters to be learned on-the-fly.

Another related line of work is on gradient-based hyper-parameter optimization [5, 13, 25, 51, 113]; our work applies a similar idea in a new setting, namely optimizer search and adaptation. Finally, we note that our work more broadly relates to a long line of research on AutoML and meta-learning [6, 70, 142, 172].

6.2 Parameterized Optimizers

We focus on minimizing a loss function $F : \mathbb{R}^d \to \mathbb{R}$, that is, optimization problems of the following form

$$\min_{x \in \mathbb{R}^d} F(x).$$

Throughout the paper, we denote by $f : \mathbb{R}^d \to \mathbb{R}$ a random function computed on a minibatch sampled from the underlying data distribution. Therefore, for any $x \in \mathbb{R}^d$, we have $\mathbb{E}[f(x)] = F(x)$. Moreover, we assume F is differentiable and that $\mathbb{E}[\nabla f(x)] = \nabla F(x)$ for all $x \in \mathbb{R}^d$. We use f_t to denote the random function evaluated with input mini-batch sampled at t.

As discussed in the introduction, various optimizers were proposed in the literature, each excelling in different scenarios. The main goal in this work is to exploit the relationships between given optimizers towards the development of a meta-optimizer that automatically adapts the optimizer choice to the learning task. Informally, a parameterized optimizer can be described as the convex hull of a set of optimizers, when mapped to a Euclidean space under a certain parameterization. This section is devoted to explaining this notion in more detail, before Section 6.3 discusses how to perform learning in this Euclidean space.

In order to build a parameterized optimizer, we begin with a collection of existing optimizers. For the sake of concreteness, we illustrate the idea through the following four optimizers: Adam, AMSGrad, Adan and Yogi. A careful inspection reveals that these four optimizers can be described in one update rule that involves three iteratively generated sequences: model parameters, first-order moments, and second-order moments, which will be denoted throughout using the vectors x, m, and v, respectively. Each optimizer only differs in the way it updates these sequences. More precisely, starting with any $x_0 \in \mathbb{R}^d$ and setting $v_0 = m_0 = 0$, the generic update rule is given by

$$x_t = x_{t-1} - \alpha_t \frac{m_t}{\sqrt{v_t} + \epsilon}, \tag{6.1}$$

Table 6.1 A unified framework to express adaptive moment optimizers.

Method	First-Order Moment	Second-Order Moment
Adam [88]	$m_t = \beta_1 m_{t-1} + (1-\beta_1) g_t$	$v_t = \beta_2 v_{t-1} + (1-\beta_2) g_t^2$
AMSGrad [143]	$m_t = \beta_1 m_{t-1} + (1-\beta_1) g_t$	$\bar{v}_t = \beta_2 \bar{v}_{t-1} + (1-\beta_2) g_t^2$ $v_t = \max\{v_{t-1}, \bar{v}_t\}$
Adan [174]	$\bar{m}_t = \beta_1 \bar{m}_{t-1} + (1-\beta_1) g_t$ $n_t = \beta_3 n_{t-1} + (1-\beta_3)(g_t - g_{t-1})$ $m_t = \bar{m}_t + \beta_3 n_t$	$\hat{g}_t = g_t + \beta_3 (g_t - g_{t-1})$ $v_t = \beta_2 v_{t-1} + (1-\beta_2) \hat{g}_t^2$
Yogi [181]	$m_t = \beta_1 m_{t-1} + (1-\beta_1) g_t$	$\hat{g}_t = v_{t-1} + g_t^2 \cdot \text{sign}(g_t^2 - v_{t-1})$ $v_t = \beta_2 v_{t-1} + (1-\beta_2) \hat{g}_t$

where $\alpha_t > 0$ is the learning rate and $\epsilon > 0$ is given. table 6.1 shows how each optimizer defines its first- and second-moment iterates within this formulation, where we define $g_t := \nabla f_t(x_{t-1})$. For simplicity of the exposition, we omit the bias-correction terms for all optimizers.

To be able to parameterize the design space of these four optimizers, we introduce real coefficients restricted between 0 and 1 that interpolate terms arising from different optimizers. Particular choices of these coefficients (typically at extreme values of 0 and 1) recover the underlying optimizers, but they express a new optimizer for their intermediate values.

As an example, observe that the first-order moment update rule of Adan already subsumes those of Adam, AMSGrad, and Yogi (where $\{\bar{m}_t\}$ and $\{n_t\}$ are new sequences introduced by Adan):

$$\begin{aligned} \bar{m}_t &= \beta_1 \bar{m}_{t-1} + (1-\beta_1) g_t \\ n_t &= \beta_3 n_{t-1} + (1-\beta_3)(g_t - g_{t-1}) \\ m_t &= \bar{m}_t + \beta_3 n_t, \end{aligned} \tag{6.2}$$

where taking $\beta_3 = 0$ recovers the update rules of the other three optimizers. With respect to the second-order moments, Adan similarly covers Adam when $\beta_3 = 0$, since we get that

$\hat{g}_t = g_t$. In order to incorporate the second-order moments of AMSGrad and Yogi in the same parameterization, we introduce two new coefficients $c, \rho \in [0, 1]$, and unify the second-moment computation as follows:

$$\begin{aligned}
\hat{g}_t &= g_t + \beta_3(g_t - g_{t-1}) \\
\tilde{g}_t^2 &= c\hat{g}_t^2 + (1-c)(v_{t-1} + \hat{g}_t^2 \cdot \text{sign}(\hat{g}_t^2 - v_{t-1})) \\
\tilde{v}_t &= \beta_2 \tilde{v}_{t-1} + (1-\beta_2)\tilde{g}_t^2 \\
v_t^{(max)} &= \max\{v_{t-1}^{(max)}, \tilde{v}_t\} \\
v_t &= \rho\tilde{v}_t + (1-\rho)v_t^{(max)}.
\end{aligned} \tag{6.3}$$

It is easy to check that, for instance, the second-order moments of Adam can be recovered when $\beta_3 = 0$, and $c = \rho = 1$; those of AMSGrad are recovered when $\beta_3 = \rho = 0$ and $c = 1$; Adan can be recovered when $c = \rho = 1$; and Yogi can be recovered with $\beta_3 = c = 0$ and $\rho = 1$.

To summarize, in this example, for the four optimizers Adam, AMSGrad, Adan and Yogi, our parameterized optimizer is given by the three updating rules: (6.1), (6.2) and (6.3). Following the same line of arguments one can generate parameterized optimizers for other collections of given optimizers as well. However, unless we know a priori how to set the corresponding coefficients, the parameterized optimizer is not readily usable in practice. In the next section, we develop our meta-optimizer which uses a parameterized optimizer and learns the coefficients in an online fashion.

6.3 Meta-Adaptive Optimizers

In the previous section, we described how one can parameterize the design space of a given collection of optimizers. Here, we define `MADA` as a meta-optimization framework that dynamically learns the coefficients of a parameterized optimizer during training. In this work, we use hyper-gradient descent [13, 25] to learn these coefficients, which avoids an expensive

hyper-parameter optimization loop that involves multiple training runs. However, we note that in principle other techniques can also be combined with parameterized optimizers to learn the coefficients.

Learning interpolation coefficients. Hyper-gradient descent views the hyper-parameters as trainable parameters of the loss function, and thus differentiates the loss with respect to them and updates them through gradient steps. [25] re-purposes PyTorch `autograd` machinery [132] to automatically compute gradients with respect to optimization hyper-parameters such as learning rate and Adam β_1 and β_2 parameters. Since the update rule of a parameterized optimizer is differentiable with respect to its learnable coefficients, we can apply the same approach to update them using hyper-gradient descent steps.

Now, we describe our meta-optimizer `MADA` in detail. We will denote a parameterized optimizer by $\mathcal{O}_q$, where $q \in \mathcal{D}$ denotes the vector of coefficients that defines the optimizer, and $\mathcal{D}$ represents the domain of the vector q. In the case of the example from Section 6.2, we have that $q = (\beta_1, \beta_2, \beta_3, \rho, c)$ and $\mathcal{O}_q$ is given by (6.1), (6.2), and (6.3). Note that $\mathcal{O}_q$ also encapsulates non-learnable state parameters (such as first-order and second-order moments) and other hyper-parameters such as the learning rate, weight decay, and stability parameter. The domain $\mathcal{D}$ represents the set of values that the vector q is allowed to take in the parameterization. In the example from Section 6.2, the domain is the unit hypercube where each element is in the range $[0, 1]$. We denote by $\Pi_{\mathcal{D}}$ the orthogonal projection onto the set $\mathcal{D}$. We provide a pseudo code to illustrate this (see Algorithm 7).

Hyper-gradient computation. Before concluding this section, we briefly illustrate hyper-gradient computation[2]. In order to compute the hyper-gradient of the loss with respect to a particular optimizer coefficient, we treat the updated model weights as a function of the coefficient. For instance, considering again the example from Section 6.2, we will show how to compute the gradient of the function f_t with respect to the coefficient ρ using the

[2]Further details on hyper-gradients can be found in [13, 25].

Algorithm 7 Pseudocode for a generic `MADA`

Input: A parameterized optimizer $\mathcal{O}_q$, where $q \in \mathcal{D}$, a hyper-learning rate α, number of total iterations T.

Init.: x_0 and q_0.

1: **for** t=1 to T **do**

2: Sample f_t.

3: Update the model parameters:

$$x_t = \mathcal{O}_{q_{t-1}}(x_{t-1}).$$

4: Update the optimizer coefficients:

$$q_t = \Pi_{\mathcal{D}} \left[q_{t-1} - \alpha \nabla_q f_t(x_{t-1}) \right].$$

5: **end for**

Output: Model wights x_T.

parameterized optimizer as given in (6.3). Using the chain rule, we obtain

$$\begin{aligned} \frac{\partial f_{t+1}(x_t)}{\partial \rho} &= \frac{\partial f_{t+1}(x_t)}{\partial x_t} \cdot \frac{\partial x_t}{\partial v_t} \cdot \frac{\partial v_t}{\partial \rho} \\ &= \frac{\partial f_{t+1}(x_t)}{\partial x_t} \cdot \frac{\partial (x_{t-1} - \alpha_t \frac{m_t}{\sqrt{v_t}+\epsilon})}{\partial v_t} \cdot \frac{\partial v_t}{\partial \rho} \\ &= \frac{\partial f_{t+1}(x_t)}{\partial x_t} \cdot \frac{\alpha_t m_t}{2\sqrt{v_t}(\sqrt{v_t}+\epsilon)^2} \cdot \left(\tilde{v}_t - v_t^{(max)} \right), \end{aligned} \tag{6.4}$$

where the first term $\partial f_{t+1}(x_t)/\partial x_t$ is readily computed using standard back-propagation[3]. Note that the latter two objects of (6.4) are not explicitly constructed in practice; instead `autograd` simply *continues* back-propagating the already-computed parameter gradients into optimizer coefficients, while handling the necessary book-keeping for hyper-gradient

[3]In (6.4), the first term is a row vector, the second term is a Jacobian matrix, and the third term is a column vector, and thus the $\cdot$ operator refers to a matrix multiplication.

computation.

6.4 On Convergence of Interpolated Optimizers

Since `MADA` uses hyper-gradients to update the optimizer, a prerequisite for a base optimizer to work well within `MADA` is that its update rules must allow efficient flow of hyper-gradients to the parameterization coefficients. In particular, in our experiments with `MADA` we have found that including AMSGrad among the base optimizers has an adverse effect on the end performance of the trained model. We conjecture that this is caused by the maximum operator in the second-moment term. Specifically, note that back-propagation of gradients through $\max(a, b)$ corresponds to a simple routing operation to the larger one of a and b, where the smaller one is passed 0 gradients. In AMSGrad, $v_t = \max\{v_{t-1}, \bar{v}_t\}$ which means that for most steps the first term will be greater, causing insufficient hyper-gradient updates on β_2 parameter through $\bar{v}_t$ path[4]. To remedy this, we introduce AVGrad (formally defined in next section), which replaces the maximum operator with time-averaging of second moments, and results in better hyper-gradient flow and validations loss.

6.4.1 AVGrad and its interpolation with Adam

Following the notation in table 6.1, we define AVGrad as an optimizer where the first-order moments and second-order moments are defined by

$$\begin{aligned} m_t &= \beta_1 m_{t-1} + (1-\beta_1) g_t \\ \bar{v}_t &= \beta_2 \bar{v}_{t-1} + (1-\beta_2) g_t^2 \\ \tilde{v}_t &= \frac{1}{t}(\bar{v}_t + (t-1)\tilde{v}_{t-1}), \\ v_t &= \tilde{v}_t, \end{aligned} \tag{6.5}$$

[4]To be accurate, considering the example parameterization (6.3), $\tilde{v}_t$ term provides another path for hyper-gradients on β_2. However, in practice we observed that when AMSGrad is used, ρ parameter tends to vanish, diminishing the effect of this path as well.

where $\tilde{v}_t$ is the running average of past $\bar{v}_t$'s. We provide the convergence analysis of AVGrad in Appendix E.2 in theorems 21 and 22

In what follows, we focus on the interpolation between Adam and AVGrad, with the interpolation coefficient ρ_t. Specifically, we replace last line in (6.5) with $v_t := \rho_t \bar{v}_t + (1-\rho_t)\tilde{v}_t$ where ρ_t interpolates between second-order moments of Adam and AVGrad.

Soundness of AVGrad. We start with a proposition showing that AVGrad is a valid alternative to AMSGrad in mitigating the sub-optimal convergence issue in Adam [143]. Specifically, when ρ_t decays with $1/t$ (*i.e.*, converges to AVGrad), the interpolated optimizer fixes the non-decreasing learning rate problem in Adam, similar to AMSGrad.

Proposition 1. *The following inequality is a sufficient condition for non-increasing effective learning rate:*

$$\rho_t \leq \frac{1}{t(1-\beta_2)+1}.$$

In Appendix F.3, we use `MADA` to solve the convex problem given in [143], an example where Adam fails. We found that `MADA` quickly converges to AVGrad, and thus to the optimum, by adapting $\rho_t \to 0$, even when we initialize it from Adam ($\rho_0 = 1$).

6.4.2 Convergence of the interpolated optimizer

Due to the complexity of our overall parameterization, deriving theoretical guarantees for `MADA` at its full scope is challenging. Therefore, in this section, we reduce the scope by examining convergence properties of interpolations between two optimizers, namely, AVGrad and Adam. In this setting, we will show that the parameterized optimizer allows the design of novel interpolated optimizers which improve the convergence rate of either optimizer up to constant factors, demonstrating the value of the parameterized optimizer formulation. We defer the proof and other technical details to Appendix E.1.

We make the following standard assumptions: F is bounded from below, is L-smooth, and with stochastic gradients that are bounded almost surely.

[39] proposed a unified analysis for the convergence analysis of Adagrad and Adam in the non-convex setting. Since AVGrad can be seen as a version of Adagrad, the proof technique of [39] can be adapted to AVGrad with some modifications. The modification includes the introduction of ρ parameter which interpolates between the second-order moment and the moment average, and results in an additional degree of freedom. Subsequently, ρ can be chosen to skew the updates towards the advantageous term under different scenarios.

Here we state the convergence result of the interpolated optimizer without momentum, that is when $\beta_1 = 0$ (see precise statement in Appendix E.1). The extension to the case with momentum can be done in the same way we extend Theorem 21 to Theorem 22, similar to [39].

Theorem 13 (Convergence of interpolation of AVGrad and Adam without momentum)**.** *Under the assumptions above and $\alpha_t = \frac{\alpha}{\sqrt{t}}$ for some $\alpha > 0$, and for $\rho_t = \frac{\rho}{t}$ for a constant ρ:*

$$G_T \leq E(T)\left[\frac{C_1}{T} + \frac{C_2 \ln\left(E(T)\right)}{T} + C_3\left[\ln\left(\frac{\rho}{\beta_2}\right)\right]_+\right],$$

where, $G_T = \frac{1}{T}\sum_{t=1}^{T} \|\nabla F(x_t)\|^2$, $E(T) = \frac{\sqrt{\rho+(1-\rho)T}}{\sqrt{1-\beta_2}}$, and C_1, C_2, C_3 are constants independent of T, ρ, β_2.

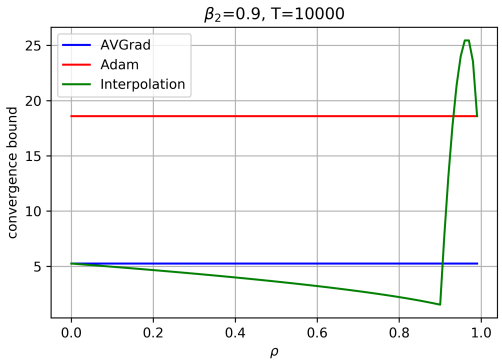


Figure 6.1 Value of the bound in Theorem 13 for a representative case where $\beta_2 = 0.9, T = 10,000$.

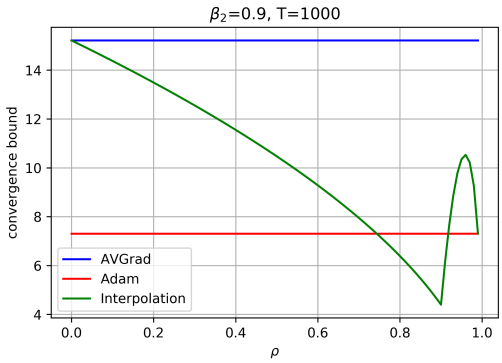


Figure 6.2 Value of the bound in Theorem 13 for a representative case where $\beta_2 = 0.9, T = 1,000$.

Remark. Aligned with the condition in Proposition 1, the contribution from v_t in Theorem 13 is multiplied with $1/t$ in order to avoid divergent terms in the bound. Observe

that when $\rho = 0$ or $\rho = 1$ we recover the convergence rates for AVGrad (see Theorem 21 in Appendix E.2) and Adam [39] respectively. We note that for the right-hand side to remain bounded, one would need either $0 \leq \rho \leq \beta_2$ (all terms vanish as in AVGrad), or $\rho = 1$ (a constant term remains as in Adam). To gain more insights, in figs. 6.1 and 6.2, we plot the value of the bound in Theorem 13 with respect to ρ for two representative cases . The first case with larger T represents a more favorable setting for AVGrad as all the terms vanish with T. The second is more favorable for Adam since the vanishing terms vanish faster than AVGrad. In both cases, the interpolated optimizer provides the best convergence bound when $\rho = \beta_2$, which suggests an advantage for the interpolated optimizers.

6.5 Experiments

In our experiments, we aim to answer how the generalization and downstream performance of `MADA` compares against fixed optimizers, and how robust `MADA` is to poor hyper-parameter initializations compared to fixed optimizers. We focus on the pre-training of models from scratch. We also try to gain insights into the behavior of `MADA` by monitoring the evolution of the optimizer coefficients. Additional details of our experimental setup, and results can be found in Appendix F.3.

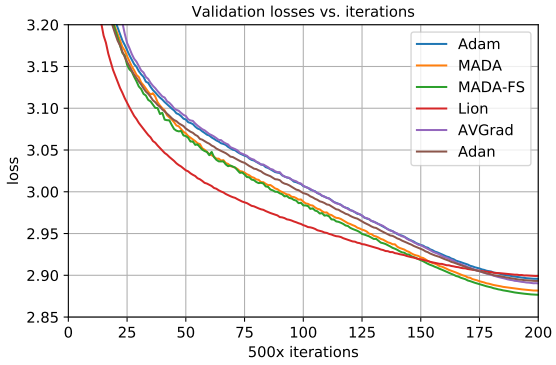


Figure 6.3 Validation losses of competing methods on OpenWebText for GPT-2 (125M) model using the same random seed.

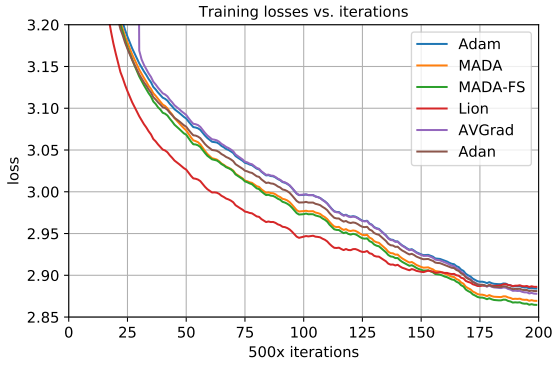


Figure 6.4 Training losses of competing methods on OpenWebText for GPT-2 (125M) model using same random seed, smoothed for convenience.

Table 6.2 Validation loss of MADA on OpenWebText vs other adaptive optimizer baselines.

Method	Validation Loss
Adam	2.8956
Adan	2.8896
HyperAdam	2.8950
Lion	2.8892
AVGrad	2.8895
MADA	**2.8806**
MADA-FS	**2.8766**
Poor initialization	
MADA$^-$	2.8921
Adam$^-$	2.9157

6.5.1 A concrete parameterized optimizer

We use a concrete parameterization that is similar to the example presented in Section 6.2, with two changes. First, we replace AMSGrad with AVGrad, and second, we include Lion among the optimizers that we interpolate. Hence, second-order moment in (6.3) becomes $v_t = \rho\bar{v}_t + (1-\rho)\tilde{v}_t$ as in the interpolation in Section 6.4. Moreover, the update term becomes $\gamma\frac{m_t}{\sqrt{v_t}+\epsilon} + (1-\gamma)\text{sign}(u_t)$ where u_t is the moment term from Lion [33]. We refer the reader to Appendix F.3 for a complete description of the parameterization. Lion was omitted in the example in Section 2 for the sake of simplicity, since it integrates less naturally with the rest of the optimizers.

.5

Table 6.3 Validation perplexities of competing methods on OpenWebText, Wikitext and Lambada datasets.

Method	OpenWebText	Wikitext	Lambada
Adam	18.0940	63.8544	77.3314
Adan	17.9863	63.5518	74.6970
HyperAdam	18.0843	61.7717	**72.6803**
Lion	17.9792	61.8661	75.3158
AVGrad	17.9840	64.2620	75.1317
MADA	**17.8249**	61.2513	74.2480
MADA-FS	**17.7544**	59.4086	**73.5623**
Poor initialization			
MADA$^-$	18.0317	**57.1613**	75.3550
Adam$^-$	18.4624	72.9017	79.1217

6.5.2 Experimental setting

Data and models. In recent years, auto-regressive models have been widely used for benchmarking and algorithm evaluation [103, 138]. Motivated by this, we evaluate MADA on the causal language modeling task with GPT-2, over two datasets: Shakespeare [84], and OpenWebText [61]. On the Shakespeare dataset, we train a 6 layer transformer with context size of 256 (11M parameters) and fine-tune GPT-2 (XL) with 48 layers (1.5B parameters). On OpenWebText, we train GPT-2 (small) with 12 layers, 1024 context size (125M parameters) and GPT-2 (medium) with 24 layers, 1024 context size (335M parameters). [5]

Baselines and learned optimizers. On OpenWebText, we compare MADA to Adam [88], HyperAdam [25] (Adam with β_1 and β_2 parameters learned through hyper-gradients), Adan [174], Lion [33], and AVGrad. For all the methods we used decoupled weight decay [109],

[5]We use nanoGPT (https://github.com/karpathy/nanoGPT) code base for the implementation.

i.e. the AdamW variant of Adam is used. We keep the learning hyper-parameters such as learning rate schedule and weight decay identical, except for Lion where we choose a lower initial learning rate as suggested in [103]. For a fair comparison, for all optimizers, we use the same codebase based on [25]. For OpenWebText experiments, we use established parameters for Adam ($\beta_1 = 0.9, \beta_2 = 0.95, \epsilon = 10^{-6}$) and also use these values as the initial parameters for `MADA`, HyperAdam, and AVGrad; for other methods we use the parameters suggested in respective papers; and measure the validation loss. For Shakespeare experiments, we compare `MADA` to Adam and HyperAdam; the relatively small model size in this experiment allows us to sweep the grid of initial β parameters and compare the final training loss across many different hyper-parameter choices. In some experiments, we also evaluate the final state of `MADA` statically, *i.e.*, we take the final optimizer learned by `MADA` and use it as the fixed optimizer from the beginning of training, which we refer to as `MADA`-FS.

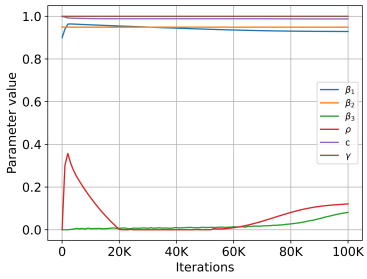


Figure 6.5 Parameter evolution for $\beta_{1,0} = 0.9, \beta_{2,0} = 0.95, \beta_{3,0} = 0$.

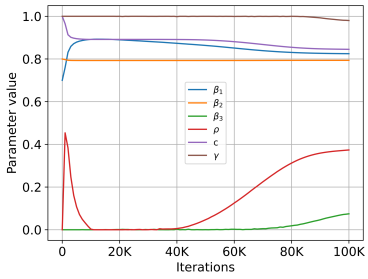


Figure 6.6 Parameter evolution for $\beta_{1,0} = 0.7, \beta_{2,0} = 0.8, \beta_{3,0} = 0$.

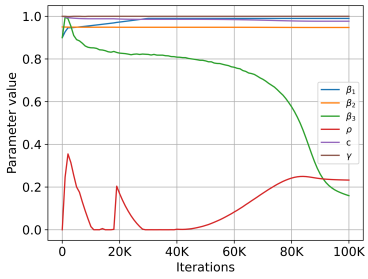


Figure 6.7 Parameter evolution for $\beta_{1,0} = 0.9, \beta_{2,0} = 0.95, \beta_{3,0} = 0.9$.

Tuning hyper-learning rates. While hyper-learning rates are additional hyper-parameters to be tuned; the tasks are less sensitive to hyper-learning rates compared to other hyper-parameters such as learning rate, most likely because each hyper-learning rate controls the update of a single parameter. As a result, hyper-learning rates can be fine-tuned easily and can be transferred across similar tasks. In general, for the hyper-learning rates of β_1, β_2, since the gradients are relatively large, we search in a set of smaller values: $\{10^{-3}, 5 \times 10^{-4}, 10^{-4}\}$. For the other parameters we search in the set $\{10^{-1}, 5 \times 10^{-2}, 10^{-2}\}$.

6.5.3 Results

In this section, we first present empirical results of training GPT-2 models on OpenWebText, and provide a discussion of generalization performance and robustness of MADA. We also show how the interpolation coefficients evolve during training. Next, we provide a visualization of the optimal training loss given various initializations of β_1, β_2 values using the smaller Shakespeare dataset. We present additional results in Appendix F.3 including a toy problem to show that MADA discovers optimal solutions that Adam cannot find, and comparing MADA and Adam for GPT-2 (medium).

GPT-2 on OpenWebText. When we initialize MADA from AVGrad and Adan with $\beta_{1,0} = 0.9, \beta_{2,0} = 0.95, \beta_{3,0} = 0.9, \rho_0 = 0$ (where subscript 0 denotes the time index) from Table 6.2 we observe that MADA consistently outperforms Adam and other recently proposed optimizers, including Lion, Adan, and HyperAdam. In particular, MADA-FS outperforms Adam with established parameters by 0.019 in validation loss which is a significant improvement for this task. In fig. 6.3 and fig. 6.4 we see that MADA is able to converge to a lower loss than baseline methods (both in terms of validation and training loss). Note that MADA is able to converge faster than AVGrad and Adam with the same learning rate and schedule, which may be attributed to theoretical result theorem 13 on interpolated optimizers.

Perplexity on benchmark datasets. To measure the generalization of MADA to other datasets, we compute the validation perplexity of the trained models on three datasets as shown in Table 6.3. We see that on OpenWebText MADA outperforms Adam and HyperAdam by 0.34 and 0.33 points); while on Wikitext [117], MADA outperforms these two baselines by 4.45 and 2.37 respectively. On Lambada [137] it is the second-best-performing method with a small gap behind HyperAdam (0.88).

MADA-FS versus MADA. We find that MADA-FS performs slightly better than MADA for OpenWebText (0.004 in validation loss and 0.07, 1.85, 0.68 points on validation perplexity on OpenWebText, Wikitext and Lambada). More details are in tables 6.2 and 6.3. On the

other hand, on Shakespeare we observe that, the dynamically-evolving optimizer performs generally better than using the fixed final optimizer. We conjecture that the difference in the number of iterations, the model size or dataset may explain this phenomenon.

Learning interpolation coefficients is crucial. We also see that HyperAdam performance is very close to Adam (2.8950 vs. 2.8956), which suggests that the main performance improvement of `MADA` does not simply originate from the tuning of the β_1, β_2 parameters, but rather from the adaptation of the interpolation coefficients in the optimizer space.

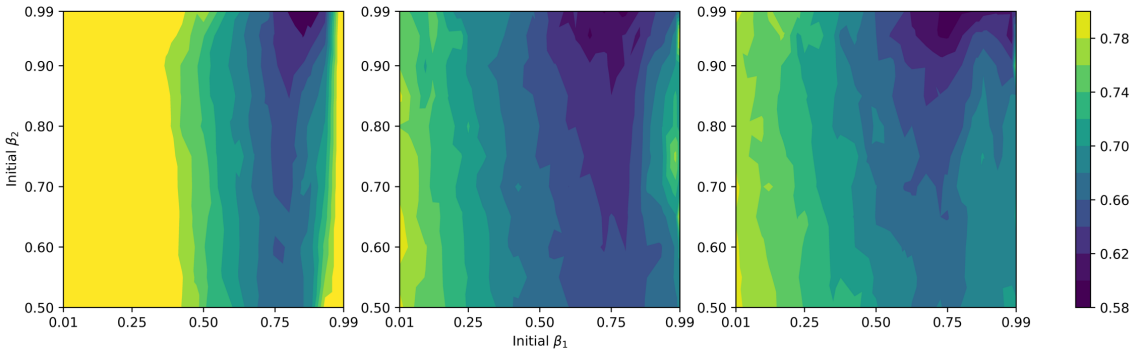


Figure 6.8 Final training loss of Adam, `MADA`, and HyperAdam vs. initial β values on Shakespeare dataset. `MADA` yields better performance for wider choice of initial β values, illustrating its robustness.

Robustness to initial hyper-parameters. Additionally, we compare `MADA` to Adam when we initialize from the suboptimal hyperparameters $\beta_{1,0} = 0.7, \beta_{2,0} = 0.8$ (we call these optimizers `MADA`$^-$ and Adam$^-$ respectively). Notably, we observe that even suboptimally initialized `MADA`$^-$ is able to outperform Adam with the established initial parameters, as well as HyperAdam. This particular example indicates the robustness of `MADA`. We provide more examples with poor initial hyper-parameter choices on Shakespeare dataset in the next subsections.

Evolving hyper-parameters. We monitor the evolution of the interpolation coefficients during training of GPT-2 (125M) model on OpenWebText for various choices of initialization of β_1, β_2 (Figures 6.5, 6.6, and 6.7).

First, we observe that sub-optimal initialization of hyper-parameters (as in Figure 6.6)

results in a more significant update of the coefficients. This suggests that default optimizer state with $\beta_{1,0} = 0.9, \beta_{2,0} = 0.95$ is close to a local minimum. A closer inspection yields insight into how MADA accelerates training. First, in Figures 6.5, 6.6 we see that β_1 increases at the beginning which facilitates taking larger steps, while towards the end it decreases to give more emphasis on individual gradients. Second, we observe that the interpolation factor of AVGrad, ρ, follows a three-phase pattern. This suggests that MADA puts more importance on individual normalization term in the beginning and towards the end, and it puts more emphasis on the less noisy averaging term during the intermediate stages.

Another interesting behavior we observe is when we initialize β_3 (which governs the weight of Adan) from a higher value such as 0.9 (Figure 6.7), we see β_1 reaches 0.99 and stays constant, in contrast with Figures 6.5, 6.6. Remarkably, MADA initialized with $\beta_3 = 0.9$ automatically trains β_1 to be 0.99; i.e. MADA, when initialized from combination of Adan and AVGrad, automatically finds the β_1 that is suggested by the authors of [174]. Note that $\beta_1, \beta_2, \beta_3$ in this work correspond to $1 - \beta_1, 1 - \beta_3, 1 - \beta_2$ in [174].

Shakespeare dataset. On Shakespeare dataset we compare MADA to Adam and HyperAdam. We show that it results in significant increase in the performance across many initial β_1, β_2 values when training 11M parameter model. Particularly, in Figure 6.8, we see that MADA results in better performance for the vast majority of initial β_1 and β_2 parameters, illustrating robustness.

Fine-tuning GPT-2 XL on Shakespeare dataset. We also fine-tune the pre-trained GPT-2 XL (1.5B parameters) model on Shakespeare, for approximately 2 epochs, using MADA, Adam, and HyperAdam methods. Table 6.4 shows the resulting training loss values, where MADA results in a large improvement. Notably, HyperAdam achieves no gain over Adam.

Vision tasks. We also validated the performance of MADA on two computer vision models: 5-layer CNN and ResNet-9 on CIFAR-10 dataset. We observe that MADA shows consistent improvement over our baselines (see test accuracy results below in Table 6.5).

Table 6.4 Training losses after 2 epochs of fine-tuning on Shakespeare dataset.

Method	Training loss
MADA	**0.255**
Adam	0.276
HyperAdam	0.278

Table 6.5 Test accuracy of competing methods for CNN and ResNet models.

Method	5-layer CNN	ResNet-9
MADA	**66.12 ± 0.14**	**93.79 ± 0.11**
Adam	65.84 ± 0.11	93.73 ± 0.10
HyperAdam	65.80 ± 0.21	93.69 ± 0.06
Momentum SGD	65.97 ± 0.94	92.60 ± 0.10

6.6 Conclusion

In this paper we propose an approach to unify a set of optimizers into a parameterized formulation. We further show we can use MADA to dynamically learn the most suitable optimizer for a particular task during training using hyper-gradient descent. We employ MADA to train language models and observe that it consistently outperforms Adam and other fixed optimizers both in terms of best validation performance, and in terms of robustness to initial hyper-parameters.

Limitations. The main limitation of our framework is the additional computational requirement and memory usage of the parameterized optimizer due to the maintained optimizer states. The additional computation can be broken down into two components (per-parameter computational steps and computation of hyper-gradients); both can be shown to be negligible with respect to the computational requirements of overall training (see Appendix E.4, where we also provide the resulting memory use and iteration speed numbers). The additional memory usage arises from the fact that we need to maintain a larger optimizer state per parameter. To mitigate, one can employ sharded data parallelism techniques [141], which shards the optimizer state across a large number of data-parallel devices. Another approach is to analyze the contributions of the constituent optimizers to the learning performance, and prune the ones that have little contribution.

Future work. Theoretically, we have shown the convergence of interpolation of AVGrad and Adam; as a future work we would like to construct a generic theoretical framework that can characterize the convergence of optimizer interpolations more generally. Empirically, we aim to gain a deeper understanding of the dynamics of the optimizer learning process in practice.

CHAPTER 7

Stochastic Rounding for LLM Training: Theory and Practice

As the parameters of Large Language Models (LLMs) have scaled to hundreds of billions, the demand for efficient training methods—balancing faster computation and reduced memory usage without sacrificing accuracy—has become more critical than ever. In recent years, various mixed precision strategies, which involve different precision levels for optimization components, have been proposed to increase training speed with minimal accuracy degradation. However, these strategies often require manual adjustments and lack theoretical justification. In this work, we leverage stochastic rounding (SR) to address numerical errors of training with low-precision representation. We provide theoretical analyses of implicit regularization and convergence under the Adam optimizer when SR is utilized. With the insights from these analyses, we extend previous BF16 + SR strategy to be used in distributed settings, enhancing the stability and performance for large scale training. Empirical results from pre-training models with up to 6.7B parameters, for the first time, demonstrate that our BF16 with SR strategy outperforms (BF16, FP32) mixed precision strategies, achieving better validation perplexity, up to 1.54× higher throughput, and 30% less memory usage.

7.1 Introduction

With the increased scale of LLMs, mixed precision (MP) training strategies [120] has become the *de facto* training strategy. Compared to FP32 training, MP vastly increases the throughput

without degrading the accuracy of the model by delegating some or most operations to 16-bit depending on the strategy used. On the other hand, vanilla BF16 training, which is even more efficient and faster than MP as every tensor and operations are in BF16 (with nearest rounding behavior for FP operations), has been shown to cause significant degradation in accuracy (see fig. 7.1 and [139, 182]).

Previous efforts [139, 182] have proposed changing the default nearest rounding (NR) behavior of some operators to stochastic rounding (SR) [36], an unbiased estimator for quantization. Utilizing SR in the model update step of BF16 training improves the trained model performance compared to vanilla BF16 approach. Recent accelerators such as AWS Trainium instances enable SR for all downcasting operations (e.g. in GEMMs) in hardware level [11, 49, 125]. However, there is still a clear performance gap between BF16+SR and MP for LLM training (for instance a 7% degradation in validation perplexity for BERT-base [182], and 2% degradation in validation loss for a 420M parameter decoder model [139]). These works have used the same hyper-parameters for MP and SR training for a direct comparison.

	BF16+SR	(BF16,FP32) MP	BF16
Accuracy	↑*	↑	↓
Throughput	↑	↑	↑
Memory eff.	↑	↓	↑
Robustness	↑*	↓	↓

Table 7.1 * denotes first shown in this work. Performance summary of competing methods. Our BF16+SR strategy shows better perplexity, faster throughput, less memory usage and more robustness towards high learning rates compared to the state of the art mixed precision methods.

In this work, our primary goal is to understand the practical and theoretical properties of stochastic rounding (SR) in training of LLMs. We hypothesize that as stochastic rounding provides an unbiased estimator, it leads to improved performance in training with less error accumulated. However, stochastic and nearest rounding's interactions with the widely

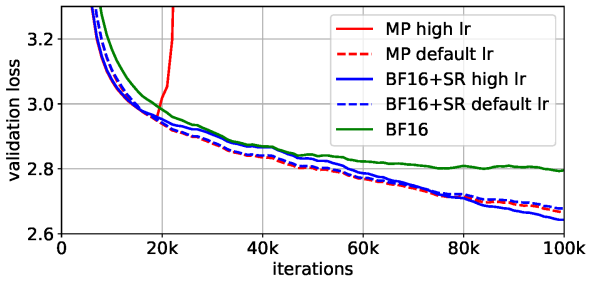


Figure 7.1 Validation losses of competing methods while pre-training GPT-2 (350M). When employed with a relatively high learning rate, training with SR outperforms mixed precision strategy. If the same high learning rate is used for the mixed precision training we observe divergent training, which indicates robustness of SR to higher learning rates.

adopted Adam optimizer [88] have not been thoroughly examined in the literature. This lack of understanding presents an opportunity to explore how different rounding methods affect convergence and overall training dynamics. By investigating these questions, we aim to provide valuable insights, both practically–by improving training efficiency and accuracy–and theoretically–by advancing the knowledge of optimizer behavior under varying rounding conditions.

We reveal that, SR shows a lower quantization error compared to NR in the convergence bound. Empirically, we show that BF16+SR strategy can outperform state of the art (BF16, FP32) MP training in terms of accuracy for training billion scale LLMs (up to 7B in our experiments), while retaining all the advantages of BF16 training. Our detailed contributions are as follows:

- Theoretically, we analyze convergence properties of SR & NR in Adam in a non-convex setting (applied to the update step), and conclude that the additional convergence error due to SR is strictly smaller than error when NR is used. With appropriate choice of hyper-parameters (e.g. high learning rate) quantization error due to SR can be subsumed by Adam's original convergence bound.
- We demonstrate that training with SR implicitly regularizes the loss function by a

quantization error penalty term. By examining training with small learning rates, we find that the penalty term may dominate and result in stagnation, which highlights the necessity of using higher learning rates for effective SR training.

- We propose a BF16 AdamW optimizer [109], with SR applied to the model update step. This optimizer for the first time empirically outperforms mixed precision training, with higher throughput and less memory usage in training billion parameter language models.

Outline. In section 7.2, first, we introduce the problem and necessary backgrounds on rounding options. Then, we motivate the use of stochastic rounding and examine the effect of learning rate. Next, in section 7.3, we introduce the BF16-AdamW-SR algorithm utilizing stochastic rounding. In section 7.4, we analyze the implicit regularization effect of SR, and compare the convergence behavior of AdamW under stochastic, and nearest rounding. Lastly, in section 7.5, we present our experimental results showcasing superior efficiency and efficacy of training with BF16+SR compared to mixed precision (BF16, FP32) and (nearest rounding) BF16 training schemes. Proofs of results, additional experiments, and details can be found in appendices.

7.1.1 Related works

Stochastic rounding. SR is an alternative rounding approach to the canonical nearest rounding (NR) where high precision number is quantized to the nearest low precision value. While NR has the best worst error guarantees, it is biased. SR on the other hand, is an unbiased quantizer, as the probability of quantization is inversely proportional to the distance to low precision values, which makes it more suitable for training ML models to reduce accumulating errors across iterations [63]. In particular, [35] shows that SR has better probabilistic error bounds when summing small numbers, as in deep learning applications, compared to nearest rounding. [97, 173] analyze error terms due to SR in gradient descent

and show that for simple machine learning tasks (e.g. regression) SR gives competitive performance. [139, 181] apply SR for training ≤1B scale LLMs and argue that there is a performance discrepancy compared to mixed precision training or FP32 training.

Efficient LLM training. Traditional machine learning training algorithms generally rely on 32-bit representations (FP32) to avoid perturbing the training performance. Recently, many works have been proposed to lower the training costs without accuracy drop using various data types, such as FP16 [120], BF16 [81], FP8 [162, 171] and so on. [120] realized that FP16 operations can be utilized by keeping a FP32 master copy for accumulating the updates without accuracy degradation. [182] showed that Kahan summation [79] could be utilized to account for numerical errors of BF16 computations. [180] used multiple-component floating-point to reduce errors during BF16 computations. Contrary to previous work, our method is able to match the performance of mixed precision training without introducing any auxiliary variables.

7.2 Motivation and Background

In this work, we are interested in minimizing a non-convex loss function $F : \mathbb{R}^d \to \mathbb{R}$, and in optimization problems of the form

$$\min_{x \in \mathbb{R}^d} F(x).$$

We denote by $f : \mathbb{R}^d \to \mathbb{R}$ a random function computed on a mini-batch sampled from the underlying data distribution. Therefore, for any model parameters $x \in \mathbb{R}^d$, we have $\mathbb{E}[f(x)] = F(x)$. We assume F is differentiable and that $\mathbb{E}[\nabla f(x)] = \nabla F(x)$ for all $x \in \mathbb{R}^d$. We use f_t or $f(x_t)$ to denote the random function evaluated with input batch sampled at time t.

7.2.1 Stochastic and nearest rounding

Here we define stochastic rounding and examine why it is more preferred to nearest rounding for training. Nearest (deterministic) rounding can be formalized as,

$$Q(x) := \text{sign}(x) \cdot \Delta_x \cdot \left\lfloor \frac{x}{\Delta_x} + \frac{1}{2} \right\rfloor, \tag{7.1}$$

where x is input with a higher precision compared to output (e.g. 32-bit and 16-bit representations), $\Delta_x = \lceil x \rceil - \lfloor x \rfloor$ is resolution, i.e. the distance between quantization levels, around x. Note, for non-uniform data types such as floating point representations the resolution depends on the magnitude of the input, which is denoted by the subscript Δ_x. Nearest rounding is the default rounding mode for (IEEE) FP operations (such as summation/subtraction and so on); typically, a higher precision buffer is used and then the result is nearest rounded. If the operands are in different precision, the lower precision is upcast and standard FP operations are followed afterwards. Next, we define the stochastic rounding operation,

$$Q_{SR}(x) := \begin{cases} \lceil x \rceil, & \text{w.p. } \frac{x - \lfloor x \rfloor}{\Delta_x} \\ \lfloor x \rfloor, & \text{w.p. } \frac{\lceil x \rceil - x}{\Delta_x} \end{cases}. \tag{7.2}$$

The crucial property of SR is, contrary to nearest rounding, unbiasedness, $\mathbb{E}[Q_{SR}(x)] = x$. Unbiasedness property prevents stagnation (or imprecision), in expectation, particularly when two numbers of different magnitude is summed in the update step. As an example, consider the gradient step for the quantized model in the form $x_{t+1} = x_t + u$ where u is the update that depends on learning rate and gradient information. When $u \ll x_t$, we have $Q(x_t + u) = x_t$; hence, stagnation occurs. However, with SR we have $\mathbb{E}[Q_{SR}(x_t + u)] = x_t + u$.

7.2.2 Effect of learning rate under SR

zamirai2021revisitingbfloat16training,rae2022scalinglanguagemodelsmethods have attempted at BF16 training with SR but found out there is a performance gap with mixed precision

training. Here, we present toy examples where this gap becomes more pronounced with a small learning rate, suggesting the necessity of using a higher learning rate for SR training. We model SR as a random walk and see that with small updates SR can take a long time to converge compared to higher precision training.

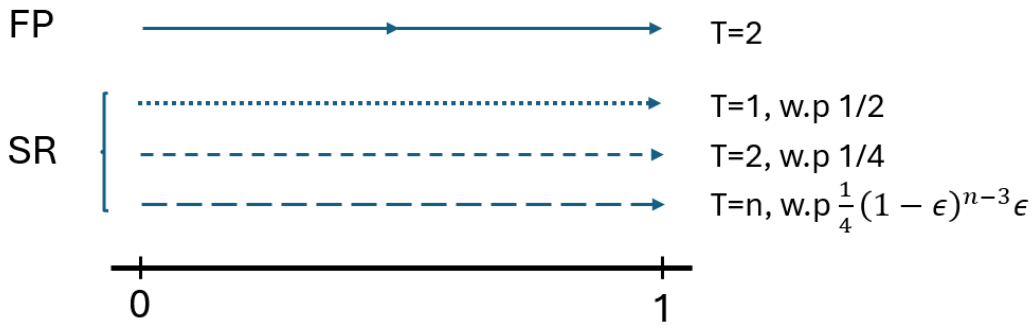


Figure 7.2 Depiction of updates when full precision and quantized stochastic rounding updates are used in the toy example.

Example. Let $x_0 = 0$, we are interested in expected number of iterations $\mathbb{E}[T]$ where $T : x_T = 1$ assuming 0 and 1 are consecutive quantization levels. Let the updates be $u_t = 1/2$ for $t = 0, 1$ and $u_t = \epsilon$ for $t > 2$. Note, with accurate full precision updates $x_2 = 1$, the next level is achieved in 2 steps. Whereas, for stochastic updates $T \leq 2$ w.p. $\frac{3}{4}$, and $T = n, n \geq 3$ w.p. $\frac{1}{4}(1-\epsilon)^{n-3}\epsilon$ (as seen in fig. 7.2). Consequently, we have $\mathbb{E}[T] > 2$ for $\epsilon < \frac{1}{2}$; and $\mathbb{E}[T] \to \infty$ for decaying updates $\epsilon \to 0$.

Example. We also present a toy simulation example with a linear function. Consider a linear loss function,

$$f(x) = \begin{cases} -(x-2)+1 & \text{for } x < 3, \\ 0 & \text{otherwise,} \end{cases}$$

iterated over with Adam optimizer using a decaying learning rate with minimum learning rate 1×10^{-5}. When we initialize from $x_0 = 2$, FP32 updates converge in 4600 steps; whereas the BF16 stochastic rounding updates converge in average in 7700 steps (plots provided in appendix F.3). And when the minimum learning rate is 5×10^{-5} SR converges in ∼4600

steps.

The toy examples indicate that SR can still result in stagnation. This stagnation can be prevented when the updates are larger. To this end, in the next sections we will conclude, both theoretically and empirically, choosing a higher learning rate for SR training is critical for a competitive performance. A caveat is, in practice, Adam is known to diverge with higher learning rates; we find that SR alleviates this in section 7.2.3 and section 7.5.

7.2.3 Robustness of Adam with SR

The examples so far implied that a higher learning rate is necessary for SR to be successfully employed. In practice, a higher learning rate results in unstable training of LLMs, even divergence. Such unstable training is associated with time-domain correlation of gradients in training with Adam optimizer. In particular, [123] shows that as time-domain correlation increases, a lower learning rate is required for training. In section 7.5, we observe that training with SR is more robust towards higher learning rates compared to BF16 nearest rounding training and (BF16-FP32) mixed precision training. We conjecture that additive noise while training with SR enables a learning rate-robust training by decorrelating the gradients. To formalize, we have the following proposition that demonstrates the decorrelation effect of SR in the simple toy setting in [123],

Proposition 2. *Assume that gradient estimated at dimension* $i \in \{1, \dots, d\}$ *and* $t \in [1, \dots, T]$ *is composed of two Bernoulli parts* $g[i,t] = g^{(1)}[i] + g^{(2)}[i,t]$*, where* $g^{(1)}[i] \sim \rho \times Bernoulli(\frac{1}{2})$ *and* $g^{(2)}[i,t] \sim Bernoulli(\frac{1}{2})$ *are independent. For simplicity, we model SR perturbation as an additional independent term* $\xi[i,t] \sim \Delta \times Bernoulli(\frac{1}{2})$*, where* Δ *denotes aggressiveness of quantization (e.g. resolution); and* $g_{SR}[i,t] = g^{(1)}[i] + g^{(2)}[i,t] + \xi[i,t]$*. Then correlation among gradients across time-domain while training with SR is* $\text{cor}_{SR} = \frac{\rho^2}{1+\rho^2+\Delta^2}$*, and without SR is* $\text{cor} = \frac{\rho^2}{1+\rho^2}$*.*

Clearly, $\text{cor} > \text{cor}_{SR}$ and Δ reduces the correlation, hence, increases stability. Note, in

practice, $\xi[i,t]$ is dependent on the gradients but there is no clear correlation. Our result holds as long as ξ is not negatively correlated with gradients.

7.3 Adam Optimizer with SR

Algorithm 8 AdamW with Stochastic Rounding and Shared Randomness

Require: Learning rate α_t, beta parameters (β_1, β_2), weight decay λ, reduced gradient $\nabla f(x_t)$ at iteration t, number of devices M, r_t^m random states for each device, $r_t^{(opt)}$ random state for the optimizer.

1: $c_{1,0} \leftarrow 1$, $c_{2,0} \leftarrow 1$
2: **for** $t = 1$ to T **do**
3: **for** $m = 1$ to M **do**
4: $g_t \leftarrow \nabla f(x_t)$ ## reduced gradient
5: $m_{t+1} \leftarrow \beta_1 \cdot m_t + (1 - \beta_1) \cdot g_t$
6: $v_{t+1} \leftarrow \beta_2 \cdot v_t + (1 - \beta_2) \cdot g_t^2$
7: $\hat{m}_{t+1} \leftarrow \frac{m_{t+1}}{1-(\beta_1)^t}$
8: $\hat{v}_{t+1} \leftarrow \sqrt{\frac{v_{t+1}}{1-(\beta_2)^t}}$
9: $r \leftarrow r_t^{(opt)}$ ## set the same random state across ranks for optimizer
10: $x_{t+1} \leftarrow Q_{SR}\left(x_t - \left(\alpha_t \cdot \frac{\hat{m}_{t+1}}{\sqrt{\hat{v}_{t+1}+\epsilon}} + \alpha_t \cdot \lambda \cdot x_t\right)\right)$ ## $r_t^{(opt)}$ is updated during SR
11: $r \leftarrow r_t^m$ ## get the original random state
12: **end for**
13: **end for**
14: **return** x_T

In this section, we present the AdamW algorithm with SR applied to the update step and shared randomness. algorithm 8 assumes a distributed setup that consists of M model replicas. In our experiments, every tensor in algorithm 8 is in BF16 data type. At the start of every iteration, the devices receive reduced (aggregated) mini-batch gradients (line 4); Using

the gradients, the optimizer states of Adam is updated (line 5,6). Before the update step, (line 9) we fork the random state such that every rank has the same randomness; otherwise, model replicas drift from each other, since same input in line 10 could be quantized to different model weights due to SR. As models diverge from each other, the computed/aggregated gradients deviates from its expectations, which will disrupt the training.

The model update (line 10) with SR is done by *dithering* [150], which temporarily upcasts to FP32 and adds with a uniform FP32 noise to the mantissa bits, and finally shifting out the fractional part; this implementation is standard and practically has nearly no overhead on throughput (also simulates the hardware implementation e.g. [11]). Since the update does not coincide with the forward pass, where the memory usage is the highest, temporary FP32 tensors do not affect the peak memory occupied during training. Consequently, BF16 training and BF16+SR training report similar throughput and memory usage. Note that mixed precision training typically uses FP32 optimizer states (lines 5,6) storage & FP32 gradient communication (line 5), and keeps the master weights in FP32 (line 10). In contrast, our method adopts BF16 for all these variables.

For our theoretical result in section 7.4, we consider operations in lines 4,5,6 to be computed in full precision for a simple analysis. We ablate the precision of these operations in appendix F.3, where the performance difference with them in high precision vs. low precision is virtually non-existent. We further provide a proof sketch for the case where, for instance line 6 is also in low precision in appendix F.2.

7.4 Theoretical Analysis

In this section, we start with implicit regularization effect of SR when combined with gradient descent. We utilize gradient descent for simplicity, general insights would translate to other first order optimization methods as well. Then, we move onto convergence properties of Adam when SR is used for parameter updates.

7.4.1 Implicit regularization under SR

Gradient descent is assumed to discretize a continuous function $x(t)$ which is defined by the ODE $\dot{x}(t) = l(x(t)) := -\nabla F(x(t))$ (also called gradient flow). The reason for discretization is because gradient updates $x_{t+1} = x_t - \alpha \nabla F(x(t))$ are not computed continuously but only at discrete points. This results in modified gradient flow: $\dot{x}(t) = \tilde{l}(x(t))$, where $\tilde{l}(x(t)) := l(x(t)) + \alpha l_1(x(t)) + \alpha^2 l_2(x(t)) + \ldots$, here l_i are error terms due to discrete approximation of gradient descent of the continuous function [10, 156]. Note that as $\alpha \to 0$ we have continuous updates, i.e. $\tilde{l}(x(t)) \to l(x(t))$. Stochastic rounding (SR) introduces an additional error term such that the first order update is $\hat{l}(w) := l(w) + \xi_\alpha(w)$. Hence, the modified flow becomes $\tilde{l}(x(t)) := \hat{l}(w) + \alpha \hat{l}_1(x(t)) + \alpha^2 \hat{l}_2(x(t)) + \mathcal{O}(\alpha^3)$ where $\hat{l}_i$ absorbs the perturbed higher order terms due to SR and $\xi_\alpha(x)$ defined as a Bernoulli random variable:

Lemma 5 (Effective gradient perturbation)**.**

$$\xi_\alpha(x) = \begin{cases} \frac{1}{\alpha}\Big(\lceil x - \alpha\nabla F(x)\rceil - (x - \alpha\nabla F(x))\Big) \\ \qquad w.p.\ \frac{x-\alpha\nabla F(x) - \lfloor x-\alpha\nabla F(x)\rfloor}{\Delta_x}, \\ \frac{1}{\alpha}\Big(\lfloor x - \alpha\nabla F(x)\rfloor - (x - \alpha\nabla F(x))\Big) \\ \qquad w.p.\ \frac{\lceil x-\alpha\nabla F(x)\rceil - (x-\alpha\nabla F(x))}{\Delta_x}, \end{cases}$$

where $\Delta_x = \lceil x - \alpha\nabla F(x)\rceil - \lfloor x - \alpha\nabla F(x)\rfloor$ *is the difference between quantization levels. Note that* $\mathbb{E}[\xi_\alpha(x)] = 0$ *and variance is* $\mathbb{E}[\|\xi_\alpha(x)\|^2] = \frac{1}{\alpha^2}\Big(\lceil (x - \alpha\nabla F(x)\rceil - (x - \alpha\nabla F(x))\Big)^\top \Big(\lfloor (x - \alpha\nabla F(x)\rfloor - (x - \alpha\nabla F(x)\Big)$.

Now we present the implicit loss function that is minimized when gradient descent with perturbed updates due to SR is employed.

Theorem 14. *Let* $F(x)$ *be the loss function to be minimized,* α *be a constant learning rate,* $\xi_\alpha(w_t)$ *be random vector quantization error while doing updates with SR. Then during gradient*

descent with SR on the loss function, implicitly, the following expected modified loss, $F_{SR}(x)$, is being optimized:

$$F_{SR}(x) := F(x) + \frac{\alpha}{4}\|\nabla F(x)\|^2 + \frac{\alpha}{4}\mathbb{E}[\|\xi_\alpha(x)\|^2]$$

Furthermore, for $\alpha \to 0, \alpha > 0$ assuming that terms with first order dependence on α vanishes, we have that the second term vanished and from the third term we are left with $\frac{1}{4}\sum_{i=1}^{d} \Delta_{x_i}|\nabla F(x)_i|$.

theorem 14 suggests that doing SR updates implicitly regularizes the loss function to minimize the error due to quantization via SR. This implies the model is optimized in a quantization aware manner. Under no quantization error case, e.g. when $\Delta_x \to 0$, the third term vanishes and we obtain the original implicit loss function in [10]. Additionally, when $\alpha \to 0$ the quantization still has regularization effect that forces the gradients to be small. The strength of the effect depends on Δ_x. Note that for a large Δ_x the quantization regularization may dominate and cause stagnation. Here, for simplicity we analyzed the case with gradient descent; the analysis could be extended to Adam optimizer following for instance [24]. As shown in appendix F.1, with this modified flow, there is a $\mathcal{O}(\alpha^3)$ error between the true solution of gradient flow and discrete update (similar to [10]); For any biased rounding scheme, such as nearest rounding, there will be a $\mathcal{O}(\alpha)$ error difference between the true updates and modified updates at every iteration; consequently, at every iteration SR benefits from lower error in expectation (see appendix F.1 for details).

7.4.2 Convergence analysis of Adam with SR

Our analysis is based on extending the Adam analysis in [39] with SR. We consider the case with no momentum, i.e. $\beta_1 = 0$, the extension to the case with momentum can be done easily as in [39], effect of momentum is simply a multiplicative slow down term on all terms. For the analysis we make the following standard assumptions.

Assumptions. (i) F is bounded below by F^*, (ii) bound on stochastic gradients, that is

$\forall x \in \mathbb{R}^d$, $\|\nabla f(x)\|_\infty \leq R$ a.s. , (iii) F is smooth, i.e., $\forall x, y \in \mathbb{R}^d$, $\|\nabla F(x) - \nabla F(y)\|_2 \leq L\|x - y\|_2$.

Theorem 15 (No momentum, $\beta_1 = 0$)**.** *Assuming access to full precision gradients, and learning rate* $\alpha_t = \alpha\sqrt{\frac{1-\beta_2^t}{1-\beta_2}}$ *with* $\alpha > 0$*, when SR is used for the update step to obtain quantized weights, we have*

$$G_T \leq \frac{A}{T} + \frac{2Rd}{T}\left(\frac{2R}{\sqrt{1-\beta_2}} + \frac{\alpha L}{1-\beta_2}\right)\left(T\ln\left(\frac{1}{\beta_2}\right)\right) + \underbrace{\frac{Rd\Delta L}{T\sqrt{1-\beta_2}}\left(\sqrt{T\ln\left(1+\frac{R^2}{\epsilon(1-\beta_2)}\right)} + T\sqrt{\ln\left(\frac{1}{\beta_2}\right)}\right)}_{\textit{quantization error}}.$$

Here $\Delta = \max_i \Delta_{x_i}$, $G_T = \frac{1}{T}\sum_{t=1}^{T}\mathbb{E}\|\nabla F(x_t)\|^2$ *and* A *is a constant that depends on* d, R, β_2, α.

Note that the first two terms are in common with analysis in [39] where the second term, Adam's (non-vanishing) gap, is $2Rd\left(\frac{2R}{\sqrt{1-\beta_2}} + \frac{\alpha L}{1-\beta_2}\right)\ln\left(\frac{1}{\beta_2}\right)$. And the third term, (non-vanishing) quantization error, is due to error introduced by SR as $\frac{Rd\Delta L}{\sqrt{1-\beta_2}}\sqrt{\ln\left(\frac{1}{\beta_2}\right)}$. Now we examine when the quantization error (third term) becomes negligible or dominant over Adam's gap (second term).

Corollary 2 (Comparison to full precision Adam)**.** *Under* $\Delta \ll \frac{\alpha}{\sqrt{1-\beta_2}}\sqrt{ln\left(\frac{1}{\beta_2}\right)}$ *or* $\Delta \ll \frac{R}{L}\sqrt{ln\left(\frac{1}{\beta_2}\right)}$*, 'quantization error' in theorem 15 becomes much smaller than non-vanishing gap in Adam.*

The effect of quantization error can be mitigated through hyper-parameters only and not solely dependent on the uncontrolled problem parameters (e.g. L, R, d). Note that with increased α the quantization error tends to be negligible, this is in parallel with our empirical observation that with high learning rate SR outperforms mixed precision training. Another observation we make is that error due to SR can dominate when $\beta_2 \to 1$ and

$\alpha = \mathcal{O}(1/\sqrt{T})$. Although, in practice $\beta_2 < 1$ is used ($\beta_2 = 0.95$ commonly for GPT models), in theory $\beta_2 \to 1, \alpha = \mathcal{O}(1/\sqrt{T})$ diminishes the non-vanishing term in Adam error bound while resulting slow down for vanishing terms [39], such cases will include a non-vanishing quantization error when SR is employed. Note that our theoretical observations is also in parallel with practical observations on performance drop with quantized training when $\beta_2 \approx 1$ in [180].

7.4.3 Convergence analysis of Adam with NR

Here, we present the convergence analysis when NR is applied in the update step instead of SR. Note this corresponds to the mixed precision (MP) training where the update is in lower precision with NR but other parts (e.g. optimizer states) are in high precision.

Theorem 16. *Under the same setting in theorem 15, Adam with NR gives the following convergence bound*

$$
\begin{aligned}
G_T \leq \frac{A}{T} &+ \frac{2Rd}{T}\left(\frac{2R}{\sqrt{1-\beta_2}} + \frac{\alpha L}{1-\beta_2}\right)\left(T\ln\left(\frac{1}{\beta_2}\right)\right) \\
&+ \frac{2Rd\Delta L}{T\sqrt{1-\beta_2}}\left(\sqrt{T\ln\left(1+\frac{R^2}{\epsilon(1-\beta_2)}\right)} + T\sqrt{\ln\left(\frac{1}{\beta_2}\right)}\right) \\
&+ \frac{\sqrt{1-\beta_2}d(R\Delta + L\Delta^2)}{\alpha}.
\end{aligned}
$$

Note that nearest rounding results in a larger error bound due to (i) multiplicative term of $\times 2$ in the second line in theorem 16 and (ii) additional error term in third line (both colored in red), indicating favorable convergence behaviour for SR training. The distinction is caused by the biasedness of nearest rounding, while $L\Delta^2$ is due to the variance (squared error), other error terms are the results of bias of the error term. This result emphasizes the detrimental effect of the accumulating bias resulting from nearest rounding. Focusing on the new non-vanishing term of nearest rounding, we observe that a large α can make the term negligible; however, in that case Adam's convergence errors start to hurt (second term in the

Table 7.2 Validation perplexity of competing methods for GPT models.

Method	GPT(350M)	GPT(770M)	GPT(1.3B)	GPT(2.7B)	GPT(6.7B)
BF16+SR(ours)	**14.07**	**12.63**	**10.46**	**10.22**	**10.05**
(BF16,FP32)MP	14.45	12.83	10.48	10.31	10.11
BF16	16.38	15.28	11.81	11.67	11.64

bound). Therefore, getting rid of the new error term without perturbing Adam's original terms is not plausible. For nearest rounding, with a decaying learning rate, e.g. $\alpha = 1/\sqrt{T}$, the error explodes unless $\beta_2 \to 1$.

Remark. The most related results are Theorem 1 from [72] and Theorem 1,2 from [98]. Theorem 1 [72] uses nearest rounding for weight quantization in Adam optimizer without momentum term (similar to our theorem 15) in a convex setting. To prevent error accumulations due to quantization bias, hou2018analysis assumes that the diameter of weight space is bounded. As a result, even as $\Delta \to 0$, their bound includes a non-vanishing term that depends on diameter of the weight space which is not the case in our work. [98] uses SR with gradient descent in convex setting and derives non-vanishing term in convergence bound similar to ours. However, since the vanilla stochastic gradient descent does not include any non-vanishing terms, there are no cases in their result where the quantization error can be subsumed by the original full precision bound such as ours. Another related result is [26] Theorem 3.2, where the authors analyze the convergence of weight quantized Adam algorithm. Different than us, they assume ℓ_2 bounds for gradient boundedness, which results non-vanishing term to implicitly depend on $d\sqrt{d}$ instead of d.

7.5 Experiments

We showcase the empirical success of our SR training paradigm by comparing to state-of-the-art mixed precision strategies. As far as we are aware, our results are the first to outperform

mixed precision training while using full BF16[1] training with SR *without additional overhead and auxiliary tensors*. We first describe our experiment setting and then move on with the results.

7.5.1 Settings

Baselines. We compare with two widely-adopted mixed precision strategies:

- O1-level: the native PyTorch's automatic mixed precision (`torch.amp`), which patches functions to internally carry out Tensor Core-friendly operations in BF16, and operations that benefit from additional precision in FP32. Because casts occur in functions, model weights remain FP32;
- O2-level: "almost BF16" mixed precision proposed in [120], which maintains a high precision FP32 copy of the model as master weights and casts model weights (except batchnorm) and data to BF16 for forward & backward passes. It uses FP32 for optimizer states, optimization and collective operations (e.g. all-reduce & gradient accumulation) in distributed settings.

O1-level mixed precision has more operations in FP32, hence, is slower in terms of throughput but can result in lower perplexity compared to O2-level mixed precision. We further augment our experiments with a full BF16 lightweight training strategy with *rounding-to-the-nearest* (as it is the default rounding mode for FP operations), where every tensor and operations, including optimizer states, are in BF16. Note, unless otherwise stated BF16 denotes BF16+default nearest rounding mode. Our SR strategy described in section 7.3 still keeps every tensor in BF16 but modifies the final update step (line 10 in algorithm 8) in optimization to be done with SR. For the sake of simplicity none of the strategies utilize ZeRO (optimizer sharding) [141] or fused AdamW implementations.

Models. We conduct experiments using GPT-2 medium and large models, with 350M

[1]except cross-entropy calculation for GPT-Neo, which is done in FP32 by default in code package.

and 770M parameters respectively [138]; and GPT-Neo [16] with 1.2B, 2.7B, 6.7B parameters. For GPT-2 experiments we follow nanoGPT [85] and for GPT-Neo we use NeMo repository [2].

Hyper-parameters. For each training strategy we use the same hyper-parameters except for the learning rate, which is individually tuned for each method. In particular, every competing method is trained with AdamW optimizer [109] with default GPT training parameters $\beta_1 = 0.9, \beta_2 = 0.95, \epsilon = 1e{-}8$, and weight decay 0.1 (except GPT-2 350M model where we used 0.2 for stability); our method, algorithm 8, also uses the same AdamW parameters. We defer more details (e.g. warmup schedule) to appendix F.3.

Training setup. For GPT-2 (350M, 770M) we train, on 49B and 46B tokens for 100k iterations. For GPT-Neo (1.3B), we train on 40B tokens; for GPT-Neo (2.7B, 6.7B) we train on 20B tokens each for 20k iterations. More information on distributed training setup (e.g. global-micro batch sizes) can be found in appendix F.3.

Learning rate. We individually tune the learning rate of each training strategy. We observe that our SR method works better with a learning rate that is $2-4\times$ of the default learning rate recommended in O1 and O2 mixed precision strategy. Such a learning rate may cause unstable or divergent training mixed precision strategy (table 7.3), and full BF16 training with rounding-to-the-nearest. The default learning rates for GPT-2 and GPT-Neo models are from [16, 103] respectively. We tune the maximum learning rate and leave the minimum learning rate as suggested by default settings similar to [130], see appendix F.3 for more details.

7.5.2 Results

Our main comparison is to (BF16, FP32) O1-level mixed precision training (`torch.amp`) as it achieves better (lower) perplexity compared to O2-level, and is natively supported by PyTorch. For throughput we also compare to O2-level, since it is considerably faster than

[2] https://github.com/NVIDIA/NeMo

Table 7.3 Default, tuned for mixed precision, tuned for SR learning rates ($\times 1e{-}4$); and whether mixed precision training **converges with SR's learning rate**. Note SR converges in all cases.

Model size	Default	MP	SR	MP converges
350M	3	3	7	×
770M	2	2	7	×
1.3B	2	4	4	✓
2.7B	1.6	4	5	✓
6.7B	1.2	3.6	5.5	×

Table 7.4 Throughput (sequences/second) of SR compared to O1 level mixed precision training (`torch.amp`).

Method	GPT(350M)	GPT(770M)	GPT(1.3B)	GPT(2.7B)	GPT(6.7B)
BF16+SR (ours)	501	254	125	84	43
(BF16,FP32) MP	469	224	105	57	28
Gain	1.07×	1.14×	1.19×	1.47×	1.54×

O1-level as most of the operations are in BF16.

Validation perplexity. In terms of validation perplexity, we compare our BF16+SR strategy to BF16 and mixed precision training in table 7.2. We observe our proposed method slightly outperforms mixed precision strategy, and significantly outperforms vanilla BF16 training. Although, our strategy does not use FP32 tensors it can converge to a better minimum thanks to being able to using a higher learning rate. In particular, we observe up to **2.6%** improvement in perplexity (GPT-2 (350M) compared to MP); for GPT-Neo the validation perplexity decreases are more conservative (for instance, **0.6%** on 6.7B) as the mixed precision models are already capable of achieving low perplexity. The difference in the perplexity gap would be higher if we trained for the same wall-clock time or with higher batch size to equate the memory usage.

Table 7.5 Memory consumption (GB) per node; SR compared to O1 level mixed precision training (`torch.amp`).

Method	GPT(350M)	GPT(770M)	GPT(1.3B)	GPT(2.7B)	GPT(6.7B)
BF16+SR (ours)	186	208	184	171	184
(BF16,FP32) MP	234	298	206	197	232
Tensor parallelism	1	1	8	8	8
Memory savings	21%	30%	11%	13%	21%

Table 7.6 Throughput of SR compared to O2 level mixed precision training evaluated in Megatron-LM.

Method \GPT-Neo size	1.3B	2.7B	6.7B
BF16+SR (ours)	135	75	32
(BF16,FP32) MP	129	71	31
Gain	1.05×	1.06×	1.03×

Throughput. In table 7.4, we compare to O1-level mixed precision strategy and observe that our method is considerably faster. The relative speed gain gets larger as the model size increases due to increasing number of model parameters in FP32 operations, data conversions and communication for the mixed precision strategy. For instance, we see a multiplicative gain of **1.54**× for 6.7B model. Additionally, we compare the throughput to O2-level (almost BF16 training) in table 7.6 using Megatron-LM repository [154] since it has more efficient training implementations than NeMo. We observe that our BF16+SR strategy is only marginally faster, as almost all operations for O2-level is done in BF16 and only accumulations are in FP32.

Memory. As our BF16+SR training strategy uses only BF16 tensors without any auxiliary tensors, it is the most efficient strategy in terms of memory as can be seen in table 7.5. Note that GPT-Neo experiments use tensor parallelism which shards the model across GPUs resulting in a smaller memory difference compared to TP=1. table 7.5 indicates

a **30**% reduction for GPT-2 (770M) and **21**% reduction for GPT-Neo (6.7B); the memory save is so significant that the training could support considerably larger models, batch sizes, or lower gradient accumulation steps.

On shared randomness. Throughout our experiments we share the randomness by keeping a designated random state for update steps as in algorithm 8. The absence of shared randomness results in suboptimal performance due to model drift across different devices. When a large learning rate is employed without shared randomness, the SR strategy is prone to divergence. Conversely, with smaller learning rates, while divergence is mitigated, the final validation loss experiences a reduction in accuracy. The influence of shared randomness is also contingent upon the distributed computing setting. In particular, when sharding is utilized (e.g., via TP), its effect is more nuanced, as the majority of the model remains shared across devices, with the exception of components such as normalization layers and, in some cases, embedding layers.

Overall, we observe that BF16+SR strategy is faster, more accurate, and memory efficient compared to the state-of-the-art O1 and O2 mixed precision methods. Another advantage is, it is easy to implement and does not require substantial changes to the model or training process. To summarize our findings we report the results for GPT-Neo (2.7B) in table 7.7. We defer the training loss curves, ablation studies and more details on experiments to appendix F.3.

Table 7.7 Overall comparison of all metrics for GPT-Neo (2.7B).

Metric	MP	SR
Validation ppl	10.31	**10.22**
Throughput (o1)	57 seq/sec	**84 seq/sec**
Throughput (o2)	71 seq/sec	**75 seq/sec**
Memory	1572GB	**1371GB**

Downstream tasks. In this work, our main focus is optimization and pre-training of LLMs. We pre-train models up-to 50B tokens, as a result we do not expect the models to do very well in downstream tasks. However, for the sake of a downstream training comparison,

here, we report zero shot evaluation experiments on common reasoning tasks using the 2.7B model. The results in table 7.8 indicates that BF16+SR is able to match the mixed precision downstream performance.

Table 7.8 Comparison of mean zero-shot accuracies for various datasets under BF16+SR and (BF16, FP32) MP settings.

Dataset / Accuracy	MP	SR
hellaswag (acc)	30.21%	30.40%
hellaswag (acc norm)	33.71%	34.42%
arc-easy (acc)	48.19%	47.43%
arc-easy (acc norm)	41.84%	42.21%
arc-challenge (acc)	21.84%	22.27%
arc-challenge (acc norm)	25.26%	26.71%
openbookqa (acc)	17.67%	19.20%
openbookqa (acc norm)	29.60%	29.66%
piqa (acc)	37.10%	36.50%

Note that both methods do not perform great in these tasks, as typically to excel in these tasks very long duration of pre-training (usually with number of tokens larger than 250B) or fine-tuning from an existent heavily pre-trained model is used. In the paper our main focus was pre-training/optimization with SR, where models were pre-trained for up to 50B tokens from scratch; hence, we are more interested in the perplexity/loss values similar to [139, 180, 182] does for quantized LLM training. Nonetheless, we conclude SR matches MP in downstream tasks in our experiments.

7.6 Conclusion

We demonstrate for the first time that low precision BF16 combined with SR training is a highly competitive approach for large-scale LLM training. Our theoretical analysis on Adam shows that, SR does not hinder convergence—its error can be absorbed within Adam's natural tolerance, particularly when an appropriately large learning rate is applied. Updates with SR effectively lead to implicit quantization-aware training. These theoretical insights are in parallel to our empirical results, where BF16+SR is able to outperform mixed-precision strategies (BF16, FP32). Overall, SR training is lightweight, straightforward to implement, and offers significant advantages, including higher throughput, reduced memory usage, lower communication/network overhead, and improved validation perplexity. We note that our method generalizes in a straightforward manner to even lower precision such as FP8/4, which we leave as future works due to limited hardware support.

CHAPTER 8

Conclusion and Future Direction

This dissertation set out to address two orthogonal but increasingly intertwined challenges in contemporary machine learning: *personalization at the edge* and *efficient optimization at scale*. The former is driven by the explosion of heterogeneous, privacy-sensitive data on resource-limited devices; the latter by the ever-growing appetite for billion-parameter models whose training footprints dwarf the capacity of even state-of-the-art accelerators. In response, this thesis has contributed a research program that advances personalized federated learning (FL) and principled, hardware-aware optimization for large language models (LLMs).

1. **Personalization under Heterogeneity.** We demonstrated, through the QuPeD framework (Chapter 2), that coupling knowledge-distillation penalties with quantization-aware training yields a single, tunable objective capable of reconciling architectural and numerical asymmetries across clients. Empirically, QuPeD exceeded prior approaches on both accuracy and communication budget while illuminating a fundamental trade-off between data heterogeneity and quantization aggressiveness.

   Building on this, Chapter 3 developed a *hierarchical Bayesian* lens that unifies a broad class of personalized FL algorithms. By casting collaboration as MAP estimation under adjustable priors, we derived new regularizers, proved utility guarantees—both with and without differential privacy—and instantiated practical algorithms that adapt automatically to mixture-of-Gaussians and information-geometric population priors.

   Chapter 4 extended personalization to unsupervised settings. The `ADEPT` family introduced hyper-priors and adaptive penalties for tasks such as personalized PCA, autoencoders,

and diffusion generative modeling. Theory linked the strength of adaptation to the degree of heterogeneity, while experiments confirmed superior generation and dimensionality reduction quality.

2. **Adaptive and Efficient Optimization for LLMs.** Addressing the second axis of scale, Chapter 6.6 proposed `MADA`, a hyper-gradient procedure that *learns to learn*. By parameterizing a continuum of adaptive optimizers and performing online interpolation, `MADA` consistently outpaced canonical algorithms (Adam, AMSGrad, *etc.*) on both vision and language workloads, with theoretical insights on the effect of interpolating optimizers on convergence.

   Chapter 7.6 aimed to improve the precision bottleneck by combining Adam with *stochastic rounding*. We showed SR acts as an implicit regularizer, stabilizing low-precision updates and provably accelerating convergence compared to deterministic rounding. The resulting **full-BF16** training recipe reduced memory footprint, increased throughput, and improved validation perplexity on large-scale language tasks.

In this thesis, several unifying principles emerged:

- **Personalization and efficiency.** Effective personalization and efficiency are not separate pursuits; statistical assumptions (priors, distillation penalties) must be thought with systems constraints (quantization, low-precision arithmetic) for maximal benefit.
- **Adaptivity.** Whether via adaptive penalties in `ADEPT` or hyper-gradient interpolation in `MADA`, adaptivity proved central to coping with challenging optimization tasks.
- **Regularization.** From Bayesian priors to stochastic rounding, regularization both stabilize training and furnish generalization guarantees, bridging theory and practice.

8.1 Concluding Remarks

Personalization and efficiency have emerged as parallel prerequisites for the next generation of AI. While this thesis tackles them along two research tracks—personalized learning at the edge and optimization for large models—it underscores their equal urgency and hints at shared underlying principles such as adaptivity, principled regularization, and systems–aware algorithm design. Taken together, these findings hint at a future where personalized models run locally and lean training keeps costs low—opening AI to everyone without sacrificing privacy or sustainability.

APPENDIX A

Appendix of Chapter 2

A.1 Preliminaries

A.1.1 Notation

- Given a composite function $g(\mathbf{x}, \mathbf{y})$ we will denote $\nabla g(\mathbf{x}, \mathbf{y})$ or $\nabla_{(\mathbf{x},\mathbf{y})} g(\mathbf{x}, \mathbf{y})$ as the gradient; $\nabla_{\mathbf{x}} g(\mathbf{x}, \mathbf{y})$ and $\nabla_{\mathbf{y}} g(\mathbf{x}, \mathbf{y})$ as the partial gradients with respect to $\mathbf{x}$ and $\mathbf{y}$.
- For a vector $\mathbf{u}$, $\|\mathbf{u}\|$ denotes the ℓ_2-norm $\|\mathbf{u}\|_2$. For a matrix $\mathbf{A}$, $\|\mathbf{A}\|_F$ denotes the Frobenius norm.
- Unless otherwise stated, for a given vector $\mathbf{x}$, x_i denotes the $i'th$ element in vector $\mathbf{x}$; and $\mathbf{x}_i$ denotes that the vector belongs to client i. Furthermore, $\mathbf{x}_i^t$ denotes a vector that belongs to client i at time t.

A.1.2 Equivalence of Assumption A.6 to Assumptions in Related Work

In particular, diversity assumption (Assumption 5) in [48] is as follows:

$$\frac{1}{n}\sum_{i=1}^{n}\left\|\nabla_{\mathbf{w}} f_i(\mathbf{w}) - \nabla_{\mathbf{w}} f(\mathbf{w})\right\|^2 \leq B,$$

where f_i is local function, B is a constant and $\nabla_{\mathbf{w}} f(\mathbf{w}) = \frac{1}{n}\sum_{j=1}^{n} \nabla_{\mathbf{w}} f_i(\mathbf{w})$. Now we will show the equivalence to our stated assumption **A**.6. Let us define,

$$x_i(\mathbf{w}^t) := \arg\min_{\mathbf{x}\in\mathbb{R}^d} \Big\{ \left\langle \mathbf{x} - \mathbf{x}_i^t, \nabla f_i\left(\mathbf{x}_i^t\right)\right\rangle + \left\langle \mathbf{x} - \mathbf{x}_i^t, \nabla_{\mathbf{x}_i^t} f_i(\widetilde{Q}_{\mathbf{c}_i^t}(\mathbf{x}_i^t))\right\rangle + \left\langle \mathbf{x} - \mathbf{x}_i^t, \lambda_p(\mathbf{x}_i^t - \mathbf{w}^t)\right\rangle + \frac{1}{2\eta_1}\left\|\mathbf{x} - \mathbf{x}_i^t\right\|_2^2 + \lambda R(\mathbf{x}, \mathbf{c}_i^t)\Big\}$$

$$c_i(\mathbf{w}^t) := \arg\min_{\mathbf{c}\in\mathbb{R}^m} \left\{ \left\langle \mathbf{c} - \mathbf{c}_i^t, \nabla_{\mathbf{c}_i^t} f_i(\widetilde{Q}_{\mathbf{c}_i^t}(x_i(\mathbf{w}^t))) \right\rangle + \frac{1}{2\eta_2} \left\| \mathbf{c} - \mathbf{c}_i^t \right\|_2^2 + \lambda R(x_i(\mathbf{w}^t), \mathbf{c}) \right\}$$

Then we can define,

$$\psi_i(x_i(\mathbf{w}^t), c_i(\mathbf{w}^t), \mathbf{w}^t) := F_i(\mathbf{x}_i^{t+1}, \mathbf{c}_i^{t+1}, \mathbf{w}^t)$$

as a result, we can further define $g_i(\mathbf{w}^t) := \psi_i(x_i(\mathbf{w}^t), c_i(\mathbf{w}^t), \mathbf{w}^t)$. Therefore, our assumption **A.6** is equivalent to stating the following assumption: At any $t \in [T]$ and any client $i \in [n]$, the variance of the local gradient (at client i) w.r.t. the global gradient is bounded, i.e., there exists $\kappa_i < \infty$, such that for every $\mathbf{w}^t \in \mathbb{R}^d$, we have:

$$\left\| \nabla_{\mathbf{w}^t} g_i(\mathbf{w}^t) - \frac{1}{n} \sum_{j=1}^{n} \nabla_{\mathbf{w}^t} g_j(\mathbf{w}^t) \right\|^2 \leq \kappa_i,$$

And we also define $\kappa := \frac{1}{n} \sum_{i=1}^{n} \kappa_i$ and then,

$$\frac{1}{n} \sum_{i=1}^{n} \left\| \nabla_{\mathbf{w}^t} g_i(\mathbf{w}^t) - \nabla_{\mathbf{w}^t} g(\mathbf{w}^t) \right\|^2 \leq \kappa,$$

here $\nabla_{\mathbf{w}^t} g(\mathbf{w}^t) = \frac{1}{n} \sum_{j=1}^{n} \nabla_{\mathbf{w}^t} g_j(\mathbf{w}^t)$. Hence, our assumption is equivalent to assumptions that are found in aforementioned works.

A.1.3 Alternating Proximal Steps

We define the following functions: $f : \mathbb{R}^d \to \mathbb{R}, \widetilde{Q} : \mathbb{R}^{d+m} \to \mathbb{R}^d$, and we also define $h(\mathbf{x}, \mathbf{c}) = f(\widetilde{Q}_{\mathbf{c}}(\mathbf{x})), h : \mathbb{R}^{d+m} \to \mathbb{R}$ where $\mathbf{x} \in \mathbb{R}^d$ and $\mathbf{c} \in \mathbb{R}^m$. Note that here $\widetilde{Q}_{\mathbf{c}}(\mathbf{x})$ denotes $\widetilde{Q}(\mathbf{x}, \mathbf{c})$. Throughout our paper we will use $\widetilde{Q}_{\mathbf{c}}(\mathbf{x})$ to denote $\widetilde{Q}(\mathbf{x}, \mathbf{c})$, in other words both $\mathbf{c}$ and $\mathbf{x}$ are inputs to the function $\widetilde{Q}_{\mathbf{c}}(\mathbf{x})$. We propose an alternating proximal gradient algorithm. Our updates are as follows:

$$\begin{aligned} \mathbf{x}^{t+1} &= \text{prox}_{\eta_1 \lambda R(\cdot, \mathbf{c}^t)}(\mathbf{x}^t - \eta_1 \nabla f(\mathbf{x}^t) - \eta_1 \nabla_{\mathbf{x}^t} f(\widetilde{Q}_{\mathbf{c}^t}(\mathbf{x}^t))) \\ \mathbf{c}^{t+1} &= \text{prox}_{\eta_2 \lambda R(\mathbf{x}^{t+1}, \cdot)}(\mathbf{c}^t - \eta_2 \nabla_{\mathbf{c}^t} f(\widetilde{Q}_{\mathbf{c}^t}(\mathbf{x}^{t+1}))) \end{aligned} \tag{A.1}$$

For simplicity we assume the functions in the objective function are differentiable, however, our analysis could also be done using subdifferentials.

Our method is inspired by [18] where the authors introduce an alternating proximal minimization algorithm to solve a broad class of non-convex problems as an alternative to coordinate descent methods. In this work we construct another optimization problem that can be used as a surrogate in learning quantized networks where both model parameters and quantization levels are subject to optimization. In particular, [18] considers a general objective function of the form $F(\mathbf{x}, \mathbf{y}) = f(\mathbf{x}) + g(\mathbf{y}) + \lambda H(\mathbf{x}, \mathbf{y})$, whereas, our objective function is tailored for learning quantized networks: $F_\lambda(\mathbf{x}, \mathbf{c}) = f(\mathbf{x}) + f(\widetilde{Q}_\mathbf{c}(\mathbf{x})) + \lambda R(\mathbf{x}, \mathbf{c})$. Furthermore, they consider updates where the proximal mappings are with respect to functions f, g, whereas in our case the proximal mappings are with respect to the distance function $R(\mathbf{x}, \mathbf{c})$ to capture the soft projection.

A.1.4 A Soft Quantization Function

In this section we give an example of the soft quantization function that can be used in previous sections. In particular, we can define the following soft quantization function: $\widetilde{Q}_\mathbf{c}(\mathbf{x}) : \mathbb{R}^{d+m} \to \mathbb{R}^d$ and $\widetilde{Q}_\mathbf{c}(\mathbf{x})_i := \sum_{j=2}^{m} (c_j - c_{j-1}) \sigma(P(x_i - \frac{c_j + c_{j-1}}{2})) + c_1$ where σ denotes the sigmoid function and P is a parameter controlling how closely $\widetilde{Q}_\mathbf{c}(\mathbf{x})$ approximates $Q_\mathbf{c}(\mathbf{x})$. Note that as $P \to \infty$, $\widetilde{Q}_\mathbf{c}(\mathbf{x}) \to Q_\mathbf{c}(\mathbf{x})$. This function can be seen as a simplification of the function that was used in [175].

Assumption. For all $j \in [m]$, c_j is in a compact set. In other words, there exists a finite $c_{\max}$ such that $|c_j| \leq c_{\max}$ for all $j \in [m]$.

In addition, we assume that the centers are sorted, i.e., $c_1 < \cdots < c_m$. Now, we state several useful facts.

Fact 1. $\widetilde{Q}_\mathbf{c}(\mathbf{x})$ *is continuously and infinitely differentiable everywhere.*

Fact 2. $\sigma(\mathbf{x})$ *is a Lipschitz continuous function.*

Fact 3. *Sum of Lipschitz continuous functions is also Lipschitz continuous.*

Fact 4. *Product of bounded and Lipschitz continuous functions is also Lipschitz continuous.*

Fact 5. *Let $g : \mathbb{R}^n \to \mathbb{R}^m$. Then, the coordinate-wise Lipschitz continuity implies overall Lipschitz continuity. In other words, let g_i be the i'th output then if g_i is Lipschitz continuous for all i, then g is also Lipschitz continuous.*

In our convergence analysis, we require that $\widetilde{Q}_{\mathbf{c}}(\mathbf{x})$ is Lipschitz continuous as well as smooth with respect to both $\mathbf{x}$ and $\mathbf{c}$.

Claim 1. *$\widetilde{Q}_{\mathbf{c}}(\mathbf{x})$ is l_{Q_1}-Lipschitz continuous and L_{Q_1}-smooth with respect to $\mathbf{x}$.*

Proof. First we prove Lipschitz continuity. Note,

$$\frac{\partial \widetilde{Q}_{\mathbf{c}}(\mathbf{x})_i}{\partial x_j} = \begin{cases} 0, \text{ if } i \neq j \\ P\sum_{j=2}^{m}(c_j - c_{j-1})\sigma(P(x_i - \frac{c_j + c_{j-1}}{2}))(1 - \sigma(P(x_i - \frac{c_j + c_{j-1}}{2}))), \text{ if } i = j \end{cases} \tag{A.2}$$

As a result, $\|\frac{\partial \widetilde{Q}_{\mathbf{c}}(\mathbf{x})_i}{\partial x_j}\| \leq \frac{P}{4}(c_m - c_1) \leq \frac{P}{2}c_{max}$. The norm of the gradient of $\widetilde{Q}_{\mathbf{c}}(\mathbf{x})_i$ with respect to x is bounded which implies there exists $l_{Q_1}^{(i)}$ such that $\|\widetilde{Q}_{\mathbf{c}}(\mathbf{x})_i - \widetilde{Q}_{\mathbf{c}}(\mathbf{x}')_i\| \leq l_{Q_1}^{(i)}\|\mathbf{x} - \mathbf{x}'\|$; using Fact 5 and the fact that i was arbitrary, there exists l_{Q_1} such that $\|\widetilde{Q}_{\mathbf{c}}(\mathbf{x}) - \widetilde{Q}_{\mathbf{c}}(\mathbf{x}')\| \leq l_{Q_1}\|\mathbf{x} - \mathbf{x}'\|$. In other words, $\widetilde{Q}_{\mathbf{c}}(\mathbf{x})$ is Lipschitz continuous.

For smoothness note that, $\nabla_{\mathbf{x}}\widetilde{Q}_{\mathbf{c}}(\mathbf{x}) = \nabla\widetilde{Q}_{\mathbf{c}}(\mathbf{x})_{1:d,:}$. Now we focus on an arbitrary term of $\nabla_{\mathbf{x}}\widetilde{Q}_{\mathbf{c}}(\mathbf{x})_{j,i}$. From (A.2) we know that this term is 0 if $i \neq j$, and a weighted sum of product of sigmoid functions if $i = j$. Then, using the Facts 1-4 the function $\nabla_{\mathbf{x}}\widetilde{Q}_{\mathbf{c}}(\mathbf{x})_{j,i}$ is Lipschitz continuous. Since i, j were arbitrarily chosen, $\nabla_{\mathbf{x}}\widetilde{Q}_{\mathbf{c}}(\mathbf{x})_{j,i}$ is Lipschitz continuous for all i, j. Then, by Fact 5, $\nabla_{\mathbf{x}}\widetilde{Q}_{\mathbf{c}}(\mathbf{x})$ is Lipschitz continuous, which implies that $\widetilde{Q}_{\mathbf{c}}(\mathbf{x})$ is L_{Q_1}-smooth for some coefficient $L_{Q_1} < \infty$. □

Claim 2. *$\widetilde{Q}_{\mathbf{c}}(\mathbf{x})$ is l_{Q_2}-Lipschitz continuous and L_{Q_2}-smooth with respect to $\mathbf{c}$.*

Proof. For Lipschitz continuity we have,

$$\begin{aligned}\frac{\partial \widetilde{Q}_{\mathbf{c}}(\mathbf{x})_i}{\partial c_j} &= \sigma(P(x_i - \frac{c_j + c_{j-1}}{2})) - \sigma(P(x_i - \frac{c_j + c_{j+1}}{2})) \\ &+ (c_j - c_{j+1})\frac{P}{2}\sigma(P(x_i - \frac{c_j + c_{j+1}}{2}))(1 - \sigma(P(x_i - \frac{c_j + c_{j+1}}{2}))) \\ &- (c_j - c_{j-1})\frac{P}{2}\sigma(P(x_i - \frac{c_j + c_{j-1}}{2}))(1 - \sigma(P(x_i - \frac{c_j + c_{j-1}}{2})))\end{aligned}$$

As a result, $\|\frac{\partial \widetilde{Q}_{\mathbf{c}}(\mathbf{x})_i}{\partial c_j}\| \leq 2 + c_{max}\frac{P}{2}$. Similar to Claim 1 using the facts that i is arbitrary and the Fact 5, we find there exists l_{Q_2} such that $\|\widetilde{Q}_{\mathbf{c}}(\mathbf{x}) - \widetilde{Q}_{\mathbf{d}}(\mathbf{x})\| \leq l_{Q_1}\|\mathbf{c} - \mathbf{d}\|$. In other words, $\widetilde{Q}_{\mathbf{c}}(\mathbf{x})$ is Lipschitz continuous. And for the smoothness, following the same idea from the proof of Claim 2 we find $\widetilde{Q}_{\mathbf{c}}(\mathbf{x})$ is L_{Q_2}-smooth with respect to $\mathbf{c}$. □

The example we gave in this section is simple yet provides technical necessities we require in the analysis. Other examples can also be used as long as they provide the smoothness properties that we utilize in the next sections.

A.1.5 Lipschitz Relations

In this section we will use the assumptions **A.1-5** and show useful relations for partial gradients derived from the assumptions. We have the following gradient for the composite function:

$$\nabla_{(\mathbf{x},\mathbf{c})} h(\mathbf{x}, \mathbf{c}) = \nabla_{(\mathbf{x},\mathbf{c})} f(\widetilde{Q}_{\mathbf{c}}(\mathbf{x})) = \nabla_{(\mathbf{x},\mathbf{c})} \widetilde{Q}_{\mathbf{c}}(\mathbf{x}) \nabla_{\widetilde{Q}_{\mathbf{c}}(\mathbf{x})} f(\widetilde{Q}_{\mathbf{c}}(\mathbf{x})) \tag{A.3}$$

where $dim(\nabla h(\mathbf{x}, \mathbf{c})) = (d + m) \times 1$, $dim(\nabla_{\widetilde{Q}_{\mathbf{c}}(\mathbf{x})} f(\widetilde{Q}_{\mathbf{c}}(\mathbf{x}))) = d \times 1$, $dim(\nabla \widetilde{Q}_{\mathbf{c}}(\mathbf{x})) = (d + m) \times d$. Note that the soft quantization functions of our interest are elementwise which implies $\frac{\partial \widetilde{Q}_{\mathbf{c}}(\mathbf{x})_i}{\partial x_j} = 0$ if $i \neq j$. In particular, for the gradient of the quantization function we have,

$$\nabla_{(\mathbf{x},\mathbf{c})}\widetilde{Q}_{\mathbf{c}}(\mathbf{x}) = \begin{bmatrix} \frac{\partial \widetilde{Q}_{\mathbf{c}}(\mathbf{x})_1}{\partial x_1} & 0 & \dots \\ 0 & \frac{\partial \widetilde{Q}_{\mathbf{c}}(\mathbf{x})_2}{\partial x_2} & 0 \dots \\ \vdots & \vdots & \vdots \\ \frac{\partial \widetilde{Q}_{\mathbf{c}}(\mathbf{x})_1}{\partial c_1} & \frac{\partial \widetilde{Q}_{\mathbf{c}}(\mathbf{x})_2}{\partial c_1} & \dots \\ \vdots & \vdots & \vdots \\ \frac{\partial \widetilde{Q}_{\mathbf{c}}(\mathbf{x})_1}{\partial c_m} & \frac{\partial \widetilde{Q}_{\mathbf{c}}(\mathbf{x})_2}{\partial c_m} & \dots \end{bmatrix} \tag{A.4}$$

Moreover for the composite function we have,

$$\nabla h(\mathbf{x},\mathbf{c}) = \begin{bmatrix} \frac{\partial f}{\partial \widetilde{Q}_{\mathbf{c}}(\mathbf{x})_1}\frac{\partial \widetilde{Q}_{\mathbf{c}}(\mathbf{x})_1}{\partial x_1} & +0 & +\dots \\ 0 & +\frac{\partial f}{\partial \widetilde{Q}_{\mathbf{c}}(\mathbf{x})_2}\frac{\partial \widetilde{Q}_{\mathbf{c}}(\mathbf{x})_2}{\partial x_2} & +\dots \\ \vdots & & \\ \frac{\partial f}{\partial \widetilde{Q}_{\mathbf{c}}(\mathbf{x})_1}\frac{\partial \widetilde{Q}_{\mathbf{c}}(\mathbf{x})_1}{\partial c_1} & +\frac{\partial f}{\partial \widetilde{Q}_{\mathbf{c}}(\mathbf{x})_2}\frac{\partial \widetilde{Q}_{\mathbf{c}}(\mathbf{x})_2}{\partial c_1} & +\dots \\ \vdots & & \\ \frac{\partial f}{\partial \widetilde{Q}_{\mathbf{c}}(\mathbf{x})_1}\frac{\partial \widetilde{Q}_{\mathbf{c}}(\mathbf{x})_1}{\partial c_m} & +\frac{\partial f}{\partial \widetilde{Q}_{\mathbf{c}}(\mathbf{x})_2}\frac{\partial \widetilde{Q}_{\mathbf{c}}(\mathbf{x})_2}{\partial c_m} & +\dots \end{bmatrix} \tag{A.5}$$

In (A.5) and (A.4) we use x_i, c_j to denote $(\mathbf{x})_i$ and $(\mathbf{c})_j$ (i'th, j'th element respectively) for notational simplicity. Now, we prove two claims that will be useful in the main analysis.

Claim 3.

$$\|\nabla_{\mathbf{x}} f(\widetilde{Q}_{\mathbf{c}}(\mathbf{x})) - \nabla_{\mathbf{y}} f(\widetilde{Q}_{\mathbf{c}}(\mathbf{y}))\| = \|\nabla h(\mathbf{x},\mathbf{c})_{1:d} - \nabla h(\mathbf{y},\mathbf{c})_{1:d}\| \leq (GL_{Q_1} + G_{Q_1} Ll_{Q_1})\|\mathbf{x}-\mathbf{y}\|$$

Proof.

$$\begin{aligned} \|\nabla h(\mathbf{x},\mathbf{c})_{1:d} - \nabla h(\mathbf{y},\mathbf{c})_{1:d}\| &= \|\nabla_{\mathbf{x}} f(\widetilde{Q}_{\mathbf{c}}(\mathbf{x})) - \nabla_{\mathbf{y}} f(\widetilde{Q}_{\mathbf{c}}(\mathbf{y}))\| \\ &= \|\nabla_{\widetilde{Q}_{\mathbf{c}}(\mathbf{x})} f(\widetilde{Q}_{\mathbf{c}}(\mathbf{x}))\nabla_{\mathbf{x}}\widetilde{Q}_{\mathbf{c}}(\mathbf{x}) - \nabla_{\widetilde{Q}_{\mathbf{c}}(\mathbf{y})} f(\widetilde{Q}_{\mathbf{c}}(\mathbf{y}))\nabla_{\mathbf{y}}\widetilde{Q}_{\mathbf{c}}(\mathbf{y})\| \end{aligned}$$

$$
\begin{aligned}
&= \|\nabla_{\widetilde{Q}_{\mathbf{c}(\mathbf{x})}} f(\widetilde{Q}_{\mathbf{c}}(\mathbf{x}))\nabla_{\mathbf{x}}\widetilde{Q}_{\mathbf{c}}(\mathbf{x}) - \nabla_{\widetilde{Q}_{\mathbf{c}(\mathbf{x})}} f(\widetilde{Q}_{\mathbf{c}}(\mathbf{x}))\nabla_{\mathbf{y}}\widetilde{Q}_{\mathbf{c}}(\mathbf{y}) \\
&\quad + \nabla_{\widetilde{Q}_{\mathbf{c}(\mathbf{x})}} f(\widetilde{Q}_{\mathbf{c}}(\mathbf{x}))\nabla_{\mathbf{y}}\widetilde{Q}_{\mathbf{c}}(\mathbf{y}) - \nabla_{\widetilde{Q}_{\mathbf{c}(\mathbf{y})}} f(\widetilde{Q}_{\mathbf{c}}(\mathbf{y}))\nabla_{\mathbf{y}}\widetilde{Q}_{\mathbf{c}}(\mathbf{y})\| \\
&\overset{(a)}{\leq} \|\nabla_{\widetilde{Q}_{\mathbf{c}(\mathbf{x})}} f(\widetilde{Q}_{\mathbf{c}}(\mathbf{x}))\|\|\nabla_{\mathbf{x}}\widetilde{Q}_{\mathbf{c}}(\mathbf{x}) - \nabla_{\mathbf{y}}\widetilde{Q}_{\mathbf{c}}(\mathbf{y})\|_F \\
&\quad + \|\nabla_{\mathbf{y}}\widetilde{Q}_{\mathbf{c}}(\mathbf{y})\|_F\|\nabla_{\widetilde{Q}_{\mathbf{c}(\mathbf{x})}} f(\widetilde{Q}_{\mathbf{c}}(\mathbf{x})) - \nabla_{\widetilde{Q}_{\mathbf{c}(\mathbf{y})}} f(\widetilde{Q}_{\mathbf{c}}(\mathbf{y}))\| \\
&\leq GL_{Q_1}\|\mathbf{x}-\mathbf{y}\| + G_{Q_1}L\|\widetilde{Q}_{\mathbf{c}}(\mathbf{x}) - \widetilde{Q}_{\mathbf{c}}(\mathbf{y})\| \\
&\leq GL_{Q_1}\|\mathbf{x}-\mathbf{y}\| + G_{Q_1}Ll_{Q_1}\|\mathbf{x}-\mathbf{y}\| \\
&= (GL_{Q_1} + G_{Q_1}Ll_{Q_1})\|\mathbf{x}-\mathbf{y}\|
\end{aligned}
$$

To obtain (a) we have used the fact $\|\mathbf{A}\mathbf{x}\|_2 \leq \|\mathbf{A}\|_F\|\mathbf{x}\|_2$. □

Claim 4.

$$
\begin{aligned}
\|\nabla_{\mathbf{c}} f(\widetilde{Q}_{\mathbf{c}}(\mathbf{x})) - \nabla_{\mathbf{d}} f(\widetilde{Q}_{\mathbf{d}}(\mathbf{x}))\| &= \|\nabla h(\mathbf{x},\mathbf{c})_{d+1:m} - \nabla h(\mathbf{x},\mathbf{d})_{d+1:m}\| \\
&\leq (GL_{Q_2} + G_{Q_2}Ll_{Q_2})\|\mathbf{c}-\mathbf{d}\|
\end{aligned}
$$

Proof. We can follow similar steps,

$$
\begin{aligned}
\|\nabla h(\mathbf{x},\mathbf{c})_{d+1:m} - \nabla h(\mathbf{x},\mathbf{d})_{d+1:m}\| &= \|\nabla_{\mathbf{c}} f(\widetilde{Q}_{\mathbf{c}}(\mathbf{x})) - \nabla_{\mathbf{d}} f(\widetilde{Q}_{\mathbf{d}}(\mathbf{x}))\| \\
&= \|\nabla_{\widetilde{Q}_{\mathbf{c}(\mathbf{x})}} f(\widetilde{Q}_{\mathbf{c}}(\mathbf{x}))\nabla_{\mathbf{c}}\widetilde{Q}_{\mathbf{c}}(\mathbf{x}) - \nabla_{\widetilde{Q}_{\mathbf{d}(\mathbf{y})}} f(\widetilde{Q}_{\mathbf{d}}(\mathbf{y}))\nabla_{\mathbf{d}}\widetilde{Q}_{\mathbf{d}}(\mathbf{y})\| \\
&= \|\nabla_{\widetilde{Q}_{\mathbf{c}(\mathbf{x})}} f(\widetilde{Q}_{\mathbf{c}}(\mathbf{x}))\nabla_{\mathbf{c}}\widetilde{Q}_{\mathbf{c}}(\mathbf{x}) - \nabla_{\widetilde{Q}_{\mathbf{c}(\mathbf{x})}} f(\widetilde{Q}_{\mathbf{c}}(\mathbf{x}))\nabla_{\mathbf{d}}\widetilde{Q}_{\mathbf{d}}(\mathbf{x}) \\
&\quad + \nabla_{\widetilde{Q}_{\mathbf{c}(\mathbf{x})}} f(\widetilde{Q}_{\mathbf{c}}(\mathbf{x}))\nabla_{\mathbf{d}}\widetilde{Q}_{\mathbf{d}}(\mathbf{y}) - \nabla_{\widetilde{Q}_{\mathbf{d}(\mathbf{x})}} f(\widetilde{Q}_{\mathbf{d}}(\mathbf{y}))\nabla_{\mathbf{d}}\widetilde{Q}_{\mathbf{d}}(\mathbf{y})\| \\
&\leq \|\nabla_{\widetilde{Q}_{\mathbf{c}(\mathbf{x})}} f(\widetilde{Q}_{\mathbf{c}}(\mathbf{x}))\|\|\nabla_{\mathbf{c}}\widetilde{Q}_{\mathbf{c}}(\mathbf{x}) - \nabla_{\mathbf{d}}\widetilde{Q}_{\mathbf{d}}(\mathbf{y})\|_F \\
&\quad + \|\nabla_{\mathbf{d}}\widetilde{Q}_{\mathbf{d}}(\mathbf{y})\|_F\|\nabla_{\widetilde{Q}_{\mathbf{c}(\mathbf{x})}} f(\widetilde{Q}_{\mathbf{c}}(\mathbf{x})) - \nabla_{\widetilde{Q}_{\mathbf{d}(\mathbf{y})}} f(\widetilde{Q}_{\mathbf{d}}(\mathbf{y}))\| \\
&\leq GL_{Q_2}\|\mathbf{c}-\mathbf{d}\| + G_{Q_2}L\|\widetilde{Q}_{\mathbf{c}}(\mathbf{x}) - \widetilde{Q}_{\mathbf{d}}(\mathbf{x})\| \\
&\leq GL_{Q_2}\|\mathbf{c}-\mathbf{d}\| + G_{Q_2}Ll_{Q_2}\|\mathbf{c}-\mathbf{d}\| \\
&= (GL_{Q_2} + G_{Q_2}Ll_{Q_2})\|\mathbf{c}-\mathbf{d}\|
\end{aligned}
$$

where $\nabla_{\mathbf{c}}\widetilde{Q}_{\mathbf{c}}(\mathbf{x}) = \nabla\widetilde{Q}_{\mathbf{c}}(\mathbf{x})_{(d+1:d+m,:)}$. □

A.1.6 Assumption A.7 is a Corollary of other Assumptions

In this section we discuss how Assumption **A.7** can be inferred from **A.1**-**A.5**. Here we drop client indices i for notational simplicity. Let us define $f_w(\mathbf{w})$ as the neural network loss function with model $\mathbf{w}$. First we will argue that $f^{KD}(\mathbf{w}, \mathbf{x})$ is smooth, given that $f_w(\mathbf{w})$ and $f(\mathbf{x})$ are two smooth neural network loss functions with Cross Entropy as the loss function, and that $f_w(\mathbf{w})$ and $f(\mathbf{x})$ have bounded gradients. These two standard assumptions imply the smoothness of $f^{KD}(\mathbf{w}, \mathbf{x})$ individually with respect to both input parameters.

Proposition 3. *$f^{KD}(\mathbf{w}, \mathbf{x})$ is L_{D_1}-smooth with respect to $\mathbf{x}$ and L_{D_2}-smooth with respect to $\mathbf{w}$ for some positive constants L_{D_1}, L_{D_2}.*

Proof. Note that $f(\mathbf{x}) = \frac{1}{N}\sum_{i=1}^{N} \mathbf{y}_i^T \log(\frac{1}{s(\mathbf{x};\xi_i)})$ where i denotes the index of data sample, $\mathbf{y}_i$ is the one hot encoding label vector, N is the total number of data samples, log denotes elementwise logarithm and $\frac{1}{s(\mathbf{x};\xi_i)}$ denotes elementwise inverse; softmax function $s(\mathbf{x}) : \mathbb{R}^d \to \mathbb{R}^K$, where K denotes the number of classes (similarly $s^w(\mathbf{w}) : \mathbb{R}^d \to \mathbb{R}^K$ is the function whose output is a vector of softmax probabilities and input is global model), is defined in Section 2.2, here we explicitly state that data samples ξ_i is a parameterization of s. Assuming $f(\mathbf{x})$ is smooth for any possible pair of $(\mathbf{y}_i, \xi_i)$ implies $\log(\frac{1}{s(\mathbf{x})_j})$ is C_x-smooth for some constant C_x, here we used $s(\mathbf{x})_j$ to denote j'th output of $s(\mathbf{x})$ and we omitted ξ_i since $\log(\frac{1}{s(\mathbf{x})_j})$ is smooth independent of ξ_i. Note that $s(\mathbf{x})_j : \mathbb{R}^d \to \mathbb{R}$. We have,

$$f^{KD}(\mathbf{w}, \mathbf{x}) = \frac{1}{N}\sum_{i=1}^{N}(s^w(\mathbf{w};\xi_i))^T \log(\frac{s^w(\mathbf{w};\xi_i)}{s(\mathbf{x};\xi_i)}) = \frac{1}{N}\sum_{i=1}^{N}(s^w(\mathbf{w};\xi_i))^T \log(s^w(\mathbf{w};\xi_i)) + \frac{1}{N}\sum_{i=1}^{N}(s^w(\mathbf{w};\xi_i))^T \log(\frac{1}{s(\mathbf{x};\xi_i)})$$

where the operations are elementwise as before. In this expression only the last term depends on $\mathbf{x}$ and for each i, since $s^w(\mathbf{w};\xi_i)_j \leq 1$ with $\sum_j s^w(\mathbf{w};\xi_i)_j = 1$, the expression is a weighted average of smooth functions $\log(\frac{1}{s(\mathbf{x})_j})$; as a result, $f^{KD}(\mathbf{w}, \mathbf{x})$ is L_{D_1}-smooth with respect to $\mathbf{x}$ for some constant L_{D_1}.

Now we investigate smoothness with respect to $\mathbf{w}$. First, note that we can assume for all j and $\mathbf{w}$, $s^w(\mathbf{w})_j$ is lower bounded by a positive constant $M > 0$ and upper bounded by a positive constant $P < 1$ since, by definition, output vector of the softmax function contains values between 0 and 1 (we ignore the limiting case when a logit is infinitely large, this is also implied by the smoothness assumptions on $f_w(\mathbf{w})$ and $f(\mathbf{x})$). Then, note that assuming a gradient bound on $f_w(\mathbf{w}) = \frac{1}{N}\sum_{i=1}^{N} \mathbf{y}_i^T \log(\frac{1}{s(\mathbf{w};\xi_i)})$ implies that for all j $\|\nabla \log(\frac{1}{s^w(\mathbf{w})_j})\| = \|\frac{\nabla s^w(\mathbf{w})_j}{s^w(\mathbf{w})_j}\| \leq G_w$ for some constant G_w (again the division operation is elementwise); since $0 < s^w(\mathbf{w})_j < 1$ we have $\|\nabla s^w(\mathbf{w})_j\| \leq G_w$. Moreover, similar to the first part, by assuming $f_w(\mathbf{w})$ is smooth we obtain that $\log(\frac{1}{s^w(\mathbf{w})_j})$ is smooth for all j. This implies having a bounded Hessian:

$$
\begin{aligned}
\text{for some constant } C \text{ we have } C \geq \left\|\nabla^2 \log(\frac{1}{s^w(\mathbf{w})_j})\right\|_F &= \left\|\frac{\nabla s^w(\mathbf{w})_j \nabla s^w(\mathbf{w})_j^T}{s^w(\mathbf{w})_j^2} - \frac{\nabla^2 s^w(\mathbf{w})_j}{s^w(\mathbf{w})_j}\right\|_F \\
&\overset{(a)}{\geq} \left|\left\|\frac{\nabla s^w(\mathbf{w})_j \nabla s^w(\mathbf{w})_j^T}{s^w(\mathbf{w})_j}\right\|_F \right. \\
&\quad \left. - \left\|\nabla^2 s^w(\mathbf{w})_j\right\|_F\right| \\
&= \left|\frac{G_w^2}{s^w(\mathbf{w})_j} - \left\|\nabla^2 s^w(\mathbf{w})_j\right\|_F\right| \\
&\geq \left\|\nabla^2 s^w(\mathbf{w})_j\right\|_F - \frac{G_w^2}{s^w(\mathbf{w})_j} \\
&\geq \left\|\nabla^2 s^w(\mathbf{w})_j\right\|_F - \frac{G_w^2}{M}
\end{aligned}
$$

where (a) is due to reverse triangular inequality and $0 < s^w(\mathbf{w})_j < 1$. As a result we obtain $C + \frac{G_w^2}{M} \geq \left\|\nabla^2 s^w(\mathbf{w})_j\right\|_F$ for all j, i.e., $s^w(\mathbf{w})_j$ is L_S-smooth with some constant $L_S \leq C + \frac{G_w^2}{M}$. Thus, both $s^w(\mathbf{w})_j$ and $\log(s^w(\mathbf{w})_j)$ are smooth functions. Note both $s^w(\mathbf{w})_j$ and $\log(s^w(\mathbf{w})_j)$ are bounded functions. Consequently, the first summation term in the definition of $f^{KD}(\mathbf{w}, \mathbf{x})$ consists of the sum of product of bounded and smooth functions and the second term consists of sum of smooth functions multiplied with positive constants (as $\log(\frac{1}{s(\mathbf{x};\xi_i)})$ does not depend

on $\mathbf{w}$). Using Fact 3 and Fact 4 we conclude there exists a constant L_{D_2} such that $f^{KD}(\mathbf{w}, \mathbf{x})$ is L_{D_2}-smooth w.r.t $\mathbf{w}$. □

Proposition 4. *$f^{KD}(\mathbf{w}, \widetilde{Q}_{\mathbf{c}}(\mathbf{x}))$ is L_{DQ_1}-smooth with respect to $\mathbf{x}$, L_{DQ_2}-smooth with respect to $\mathbf{c}$, and L_{DQ_3}-smooth with respect to $\mathbf{w}$ for some constants $L_{DQ_1}, L_{DQ_2}, L_{DQ_3}$.*

Proof. The proof is exactly the same as in Proposition 3, instead of using smoothness of $f(\mathbf{x})$, using smoothness of $f(\widetilde{Q}_{\mathbf{c}}(\mathbf{x}))$ with respect to $\mathbf{x}, \mathbf{c}$ from Claims 3 and 4 gives the result. □

A.2 Proof of Theorem 1

This proof consists of two parts. First we show the sufficient decrease property by sequentially using Lipschitz properties for each update step in Algorithm 1. For each variable $\mathbf{x}$ and $\mathbf{c}$ we find the decrease inequalities and then combine them to obtain an overall sufficient decrease. Then we bound the norm of the gradient using optimality conditions of the proximal updates in Algorithm 1. Using sufficient decrease and bound on the gradient we arrive at the result. We leave some of the derivations and proof of the claims to Appendix A.2.3.

Alternating updates. Remember that for the Algorithm 1 we have the following alternating updates:

$$\mathbf{x}^{t+1} = \text{prox}_{\eta_1 \lambda R_{\mathbf{c}^t}}(\mathbf{x}^t - \eta_1 \nabla f(\mathbf{x}^t) - \eta_1 \nabla_{\mathbf{x}^t} f(\widetilde{Q}_{\mathbf{c}^t}(\mathbf{x}^t)))$$
$$\mathbf{c}^{t+1} = \text{prox}_{\eta_2 \lambda R_{\mathbf{x}^{t+1}}}(\mathbf{c}^t - \eta_2 \nabla_{\mathbf{c}^t} f(\widetilde{Q}_{\mathbf{c}^t}(\mathbf{x}^{t+1})))$$

These translate to following optimization problems for $\mathbf{x}$ and $\mathbf{c}$ respectively (see end of the section for derivation):

$$\mathbf{x}^{t+1} = \arg\min_{\mathbf{x} \in \mathbb{R}^d} \left\{ \left\langle \mathbf{x} - \mathbf{x}^t, \nabla_{\mathbf{x}^t} f\left(\mathbf{x}^t\right) \right\rangle + \left\langle \mathbf{x} - \mathbf{x}^t, \nabla_{\mathbf{x}^t} f(\widetilde{Q}_{\mathbf{c}^t}(\mathbf{x}^t)) \right\rangle + \frac{1}{2\eta_1} \left\| \mathbf{x} - \mathbf{x}^t \right\|_2^2 + \lambda R(\mathbf{x}, \mathbf{c}^t) \right\} \tag{A.6}$$

$$\mathbf{c}^{t+1} = \arg\min_{\mathbf{c} \in \mathbb{R}^m} \left\{ \left\langle \mathbf{c} - \mathbf{c}^t, \nabla_{\mathbf{c}^t} f(\widetilde{Q}_{\mathbf{c}^t}(\mathbf{x}^{t+1})) \right\rangle + \frac{1}{2\eta_2} \left\| \mathbf{c} - \mathbf{c}^t \right\|_2^2 + \lambda R(\mathbf{x}^{t+1}, \mathbf{c}) \right\} \tag{A.7}$$

A.2.1 Sufficient Decrease

This section is divided into two, first we will show sufficient decrease property with respect to $\mathbf{x}$, then we will show sufficient decrease property with respect to $\mathbf{c}$.

A.2.1.1 Sufficient Decrease Due to x

Claim 5. *$f(\mathbf{x}) + f(\widetilde{Q}_{\mathbf{c}}(\mathbf{x}))$ is $(L + GL_{Q_1} + G_{Q_1}LL_{Q_1})$-smooth with respect to $\mathbf{x}$.*

Using Claim 5 we have,

$$\begin{aligned} F_\lambda(\mathbf{x}^{t+1}, \mathbf{c}^t) + (\frac{1}{2\eta_1} - \frac{L + GL_{Q_1} + G_{Q_1}LL_{Q_1}}{2})\|\mathbf{x}^{t+1} - \mathbf{x}^t\|^2 & \\ = f(\mathbf{x}^{t+1}) + f(\widetilde{Q}_{\mathbf{c}^t}(\mathbf{x}^{t+1})) + \lambda R(\mathbf{x}^{t+1}, \mathbf{c}^t) & \quad \text{(A.8)} \\ + (\frac{1}{2\eta_1} - \frac{L + GL_{Q_1} + G_{Q_1}LL_{Q_1}}{2})\|\mathbf{x}^{t+1} - \mathbf{x}^t\|^2 & \\ \leq f(\mathbf{x}^t) + f(\widetilde{Q}_{\mathbf{c}^t}(\mathbf{x}^t)) + \lambda R(\mathbf{x}^{t+1}, \mathbf{c}^t) + \left\langle \nabla f(\mathbf{x}^t), \mathbf{x}^{t+1} - \mathbf{x}^t \right\rangle & \\ + \left\langle \nabla_{\mathbf{x}^t} f(\widetilde{Q}_{\mathbf{c}^t}(\mathbf{x}^t)), \mathbf{x}^{t+1} - \mathbf{x}^t \right\rangle + \frac{1}{2\eta_1}\|\mathbf{x}^{t+1} - \mathbf{x}^t\|^2 & \quad \text{(A.9)} \end{aligned}$$

Claim 6. *Let*

$$\begin{aligned} A(\mathbf{x}^{t+1}) &:= \lambda R(\mathbf{x}^{t+1}, \mathbf{c}^t) + \left\langle \nabla f(\mathbf{x}^t), \mathbf{x}^{t+1} - \mathbf{x}^t \right\rangle + \left\langle \nabla_{\mathbf{x}^t} f(\widetilde{Q}_{\mathbf{c}^t}(\mathbf{x}^t)), \mathbf{x}^{t+1} - \mathbf{x}^t \right\rangle \\ &\quad + \frac{1}{2\eta_1}\|\mathbf{x}^{t+1} - \mathbf{x}^t\|^2 \\ A(\mathbf{x}^t) &:= \lambda R(\mathbf{x}^t, \mathbf{c}^t). \end{aligned}$$

Then $A(\mathbf{x}^{t+1}) \leq A(\mathbf{x}^t)$.

Now we use Claim 6 and get,

$$\begin{aligned} f\left(\mathbf{x}^t\right) + \left\langle \mathbf{x}^{t+1} - \mathbf{x}^t, \nabla f\left(\mathbf{x}^t\right) \right\rangle + \frac{1}{2\eta_1}\left\|\mathbf{x}^{t+1} - \mathbf{x}^t\right\|_2^2 + \lambda R\left(\mathbf{x}^{t+1}, \mathbf{c}^t\right) + f(\widetilde{Q}_{\mathbf{c}^t}(\mathbf{x}^t)) & \\ + \left\langle \nabla_{\mathbf{x}^t} f(\widetilde{Q}_{\mathbf{c}^t}(\mathbf{x}^t)), \mathbf{x}^{t+1} - \mathbf{x}^t \right\rangle \leq f\left(\mathbf{x}^t\right) + f(\widetilde{Q}_{\mathbf{c}^t}(\mathbf{x}^t)) + \lambda R\left(\mathbf{x}^t, \mathbf{c}^t\right) & \\ = F_\lambda\left(\mathbf{x}^t, \mathbf{c}^t\right). & \end{aligned}$$

Using (A.9) we have,

$$F_\lambda(\mathbf{x}^{t+1}, \mathbf{c}^t) + (\frac{1}{2\eta_1} - \frac{L + GL_{Q_1} + G_{Q_1}LL_{Q_1}}{2})\|\mathbf{x}^{t+1} - \mathbf{x}^t\|^2 \leq F_\lambda\left(\mathbf{x}^t, \mathbf{c}^t\right).$$

Now, we choose $\eta_1 = \frac{1}{2(L+GL_{Q_1}+G_{Q_1}Ll_{Q_1})}$ and obtain the decrease property for $\mathbf{x}$:

$$F_\lambda\left(\mathbf{x}^{t+1}, \mathbf{c}^t\right) + \frac{L + GL_{Q_1} + G_{Q_1}Ll_{Q_1}}{2}\|\mathbf{x}^{t+1} - \mathbf{x}^t\|^2 \leq F_\lambda\left(\mathbf{x}^t, \mathbf{c}^t\right). \tag{A.10}$$

A.2.1.2 Sufficient Decrease Due to c

From Claim 4 we have $f(\widetilde{Q}_{\mathbf{c}}(\mathbf{x}))$ is $(GL_{Q_2} + G_{Q_2}LL_{Q_2})$-smooth with respect to $\mathbf{c}$. Using Claim 4,

$$\begin{aligned}
&F_\lambda(\mathbf{x}^{t+1}, \mathbf{c}^{t+1}) + (\frac{1}{2\eta_2} - \frac{GL_{Q_2} + G_{Q_2}LL_{Q_2}}{2})\|\mathbf{c}^{t+1} - \mathbf{c}^t\|^2 \\
&= f(\mathbf{x}^{t+1}) + f(\widetilde{Q}_{\mathbf{c}^{t+1}}(\mathbf{x}^{t+1})) + \lambda R(\mathbf{x}^{t+1}, \mathbf{c}^{t+1}) + (\frac{1}{2\eta_2} - \frac{GL_{Q_2} + G_{Q_2}LL_{Q_2}}{2})\|\mathbf{c}^{t+1} - \mathbf{c}^t\|^2 \\
&\leq f(\mathbf{x}^{t+1}) + f(\widetilde{Q}_{\mathbf{c}^t}(\mathbf{x}^{t+1})) + \lambda R(\mathbf{x}^{t+1}, \mathbf{c}^{t+1}) + \left\langle \nabla_{\mathbf{c}^t} f(\widetilde{Q}_{\mathbf{c}^t}(\mathbf{x}^{t+1})), \mathbf{c}^{t+1} - \mathbf{c}^t \right\rangle \\
&\qquad + \frac{1}{2\eta_2}\|\mathbf{c}^{t+1} - \mathbf{c}^t\|^2
\end{aligned} \tag{A.11}$$

Now we state the counterpart of Claim 6 for $\mathbf{c}$.

Claim 7. *Let*

$$\begin{aligned}
B(\mathbf{c}^{t+1}) &:= \lambda R(\mathbf{x}^{t+1}, \mathbf{c}^{t+1}) + \left\langle \nabla_{\mathbf{c}^t} f(\widetilde{Q}_{\mathbf{c}^t}(\mathbf{x}^{t+1})), \mathbf{c}^{t+1} - \mathbf{c}^t \right\rangle + \frac{1}{2\eta_1}\|\mathbf{c}^{t+1} - \mathbf{c}^t\|^2 \\
B(\mathbf{c}^t) &:= \lambda R(\mathbf{x}^{t+1}, \mathbf{c}^t).
\end{aligned}$$

Then $B(\mathbf{c}^{t+1}) \leq B(\mathbf{c}^t)$.

Now using Claim 7,

$$\begin{aligned}
f(\mathbf{x}^{t+1}) + \eta_2\left\|\mathbf{c}^{t+1} - \mathbf{c}^t\right\|_2^2 + \lambda R(\mathbf{x}^{t+1}, \mathbf{c}^{t+1}) + f(\widetilde{Q}_{\mathbf{c}^t}(\mathbf{x}^{t+1})) + \left\langle \mathbf{c}^{t+1} - \mathbf{c}^t, \nabla_{\mathbf{c}^t} f(\widetilde{Q}_{\mathbf{c}^t}(\mathbf{x}^{t+1})) \right\rangle \\
\leq f\left(\mathbf{x}^{t+1}\right) + f(\widetilde{Q}_{\mathbf{c}^t}(\mathbf{x}^{t+1})) + \lambda R\left(\mathbf{x}^{t+1}, \mathbf{c}^t\right) = F_\lambda\left(\mathbf{x}^{t+1}, \mathbf{c}^t\right)
\end{aligned}$$

Setting $\eta_2 = \frac{1}{2(GL_{Q_2}+G_{Q_2}Ll_{Q_2})}$ and using the bound in (A.11), we obtain the sufficient decrease for $\mathbf{c}$:

$$F_\lambda\left(\mathbf{x}^{t+1}, \mathbf{c}^{t+1}\right) + \frac{GL_{Q_2} + G_{Q_2}Ll_{Q_2}}{2}\|\mathbf{c}^{t+1} - \mathbf{c}^t\|^2 \leq F_\lambda\left(\mathbf{x}^{t+1}, \mathbf{c}^t\right) \tag{A.12}$$

A.2.1.3 Overall Decrease

Summing the bounds in (A.10) and (A.12), we have the overall decrease property:

$$F_\lambda(\mathbf{x}^{t+1}, \mathbf{c}^{t+1}) + \frac{L + GL_{Q_1} + G_{Q_1}Ll_{Q_1}}{2}\|\mathbf{x}^{t+1} - \mathbf{x}^t\|^2 + \frac{GL_{Q_2} + G_{Q_2}Ll_{Q_2}}{2}\|\mathbf{c}^{t+1} - \mathbf{c}^t\|^2 \leq F_\lambda\left(\mathbf{x}^t, \mathbf{c}^t\right) \tag{A.13}$$

Let us define $L_{\min} = \min\{L + GL_{Q_1} + G_{Q_1}Ll_{Q_1}, GL_{Q_2} + G_{Q_2}Ll_{Q_2}\}$, and $\mathbf{z}^t = (\mathbf{x}^t, \mathbf{c}^t)$. Then from (A.13):

$$F_\lambda\left(\mathbf{z}^{t+1}\right) + \frac{L_{\min}}{2}(\|\mathbf{z}^{t+1} - \mathbf{z}^t\|^2) = F_\lambda\left(\mathbf{x}^{t+1}, \mathbf{c}^{t+1}\right) + \frac{L_{\min}}{2}(\|\mathbf{x}^{t+1} - \mathbf{x}^t\|^2 + \|\mathbf{c}^{t+1} - \mathbf{c}^t\|^2) \leq F_\lambda\left(\mathbf{x}^t, \mathbf{c}^t\right) = F_\lambda(\mathbf{z}^t)$$

Telescoping the above bound for $t = 0, \ldots, T-1$, and dividing by T:

$$\frac{1}{T}\sum_{t=0}^{T-1}(\left\|\mathbf{z}^{t+1} - \mathbf{z}^t\right\|_2^2) \leq \frac{2\left(F_\lambda\left(\mathbf{z}^0\right) - F_\lambda\left(\mathbf{z}^T\right)\right)}{L_{\min}T} \tag{A.14}$$

A.2.2 Bound on the Gradient

We now find the first order stationarity guarantee. Taking the derivative of (A.6) with respect to $\mathbf{x}$ at $\mathbf{x} = \mathbf{x}^{t+1}$ and setting it to 0 gives us the first order optimality condition:

$$\nabla f(\mathbf{x}^t) + \nabla_{\mathbf{x}^t} f(\widetilde{Q}_{\mathbf{c}^t}(\mathbf{x}^t)) + \frac{1}{\eta_1}\left(\mathbf{x}^{t+1} - \mathbf{x}^t\right) + \lambda\nabla_{\mathbf{x}^{t+1}} R\left(\mathbf{x}^{t+1}, \mathbf{c}^t\right) = 0 \tag{A.15}$$

Combining the above equality and Claim 5:

$$\left\|\nabla_{\mathbf{x}^{t+1}} F_\lambda(\mathbf{x}^{t+1}, \mathbf{c}^t)\right\|_2 = \left\|\nabla f(\mathbf{x}^{t+1}) + \nabla_{\mathbf{x}^{t+1}} f(\widetilde{Q}_{\mathbf{c}^t}(\mathbf{x}^{t+1})) + \lambda\nabla_{\mathbf{x}^{t+1}} R(\mathbf{x}^{t+1}, \mathbf{c}^t)\right\|_2$$

$$
\begin{aligned}
&\overset{(a)}{=} \left\| \frac{1}{\eta} \left(\mathbf{x}^t - \mathbf{x}^{t+1} \right) + \nabla f(\mathbf{x}^{t+1}) - \nabla f(\mathbf{x}^t) + \nabla_{\mathbf{x}^{t+1}} f(\widetilde{Q}_{\mathbf{c}^t}(\mathbf{x}^{t+1})) \right. \\
&\qquad\qquad \left. - \nabla_{\mathbf{x}^t} f(\widetilde{Q}_{\mathbf{c}^t}(\mathbf{x}^t)) \right\|_2 \\
&\leq (\frac{1}{\eta_1} + L + GL_{Q_1} + G_{Q_1} L l_{Q_1}) \left\| \mathbf{x}^{t+1} - \mathbf{x}^t \right\|_2 \\
&\overset{(b)}{=} 3(L + GL_{Q_1} + G_{Q_1} L l_{Q_1}) \left\| \mathbf{x}^{t+1} - \mathbf{x}^t \right\|_2 \\
&\leq 3(L + GL_{Q_1} + G_{Q_1} L l_{Q_1}) \left\| \mathbf{z}_{t+1} - \mathbf{z}_t \right\|_2
\end{aligned}
$$

where (a) is from (A.15) and (b) is because we chose $\eta_1 = \frac{1}{2(L+GL_{Q_1}+G_{Q_1}Ll_{Q_1})}$. First order optimality condition in (A.7) for $\mathbf{c}^{t+1}$ gives:

$$
\nabla_{\mathbf{c}^{t+1}} f(\widetilde{Q}_{\mathbf{c}^{t+1}}(\mathbf{x}^{t+1})) + \frac{1}{\eta_2}(\mathbf{c}^{t+1} - \mathbf{c}^t) + \lambda \nabla_{\mathbf{c}^{t+1}} R(\mathbf{x}^{t+1}, \mathbf{c}^{t+1}) = 0
$$

Combining the above equality and Claim 4:

$$
\begin{aligned}
\left\| \nabla_{\mathbf{c}^{t+1}} F_\lambda(\mathbf{x}^{t+1}, \mathbf{c}^{t+1}) \right\|_2 &= \left\| \nabla_{\mathbf{c}^{t+1}} f(\widetilde{Q}_{\mathbf{c}^{t+1}}(\mathbf{x}^{t+1})) + \lambda \nabla_{\mathbf{c}^{t+1}} R(\mathbf{x}^{t+1}, \mathbf{c}^{t+1}) \right\|_2 \\
&= \left\| \frac{1}{\eta_2} \left(\mathbf{c}^t - \mathbf{c}^{t+1} \right) + \nabla_{\mathbf{c}^{t+1}} f(\widetilde{Q}_{\mathbf{c}^{t+1}}(\mathbf{x}^{t+1})) - \nabla_{\mathbf{c}^t} f(\widetilde{Q}_{\mathbf{c}^t}(\mathbf{x}^{t+1})) \right\|_2 \\
&\leq (\frac{1}{\eta_2} + GL_{Q_2} + G_{Q_2} L l_{Q_2}) \left\| \mathbf{c}^{t+1} - \mathbf{c}^t \right\|_2 \\
&\overset{(a)}{=} 3(GL_{Q_2} + G_{Q_2} L l_{Q_2}) \left\| \mathbf{c}^{t+1} - \mathbf{c}^t \right\|_2 \\
&\leq 3(GL_{Q_2} + G_{Q_2} L l_{Q_2}) \left\| \mathbf{z}^{t+1} - \mathbf{z}^t \right\|_2
\end{aligned}
$$

where (a) is because we set $\eta_2 = \frac{1}{2(GL_{Q_2}+G_{Q_2}Ll_{Q_2})}$. Then:

$$
\begin{aligned}
\left\| [\nabla_{\mathbf{x}^{t+1}} F_\lambda(\mathbf{x}^{t+1}, \mathbf{c}^t)^T, \nabla_{\mathbf{c}^{t+1}} F_\lambda(\mathbf{x}^{t+1}, \mathbf{c}^{t+1})^T]^T \right\|_2^2 &= \left\| \nabla_{\mathbf{x}} F_\lambda \left(\mathbf{x}^{t+1}, \mathbf{c}^t \right) \right\|_2^2 \\
&\quad + \left\| \nabla_{\mathbf{c}} F_\lambda \left(\mathbf{x}^{t+1}, \mathbf{c}^{t+1} \right) \right\|_2^2 \\
&\leq 3^2 (G(L_{Q_1} + L_{Q_2}) + L(1 + G_{Q_1} l_{Q_1} + G_{Q_2} l_{Q_2}))^2 \|\mathbf{z}^{t+1} - \mathbf{z}^t\|^2
\end{aligned}
$$

Letting $L_{\max} = \max\{L + GL_{Q_1} + G_{Q_1} L l_{Q_1}, GL_{Q_2} + G_{Q_2} L l_{Q_2}\}$ we have:

$$\left\|[\nabla_{\mathbf{x}^{t+1}} F_\lambda(\mathbf{x}^{t+1}, \mathbf{c}^t)^T, \nabla_{\mathbf{c}^{t+1}} F_\lambda(\mathbf{x}^{t+1}, \mathbf{c}^{t+1})^T]^T\right\|_2^2 \leq 9L_{\max}^2 \left\|\mathbf{z}^{t+1} - \mathbf{z}^t\right\|_2^2$$

Summing over all time points and dividing by T:

$$\begin{aligned}\frac{1}{T}\sum_{t=0}^{T-1}\left\|[\nabla_{\mathbf{x}^{t+1}} F_\lambda(\mathbf{x}^{t+1}, \mathbf{c}^t)^T, \nabla_{\mathbf{c}^{t+1}} F_\lambda(\mathbf{x}^{t+1}, \mathbf{c}^{t+1})^T]^T\right\|_2^2 &\leq \frac{1}{T}\sum_{t=0}^{T-1} 9L_{\max}^2(\left\|\mathbf{z}^{t+1} - \mathbf{z}^t\right\|_2) \\ &\leq \frac{18L_{\max}^2\left(F_\lambda\left(\mathbf{z}^0\right) - F_\lambda(\mathbf{z}^T)\right)}{L_{\min}T},\end{aligned}$$

where in the last inequality we use (A.14). This concludes the proof of Theorem 1.

A.2.3 Omitted Details

First we derive the optimization problems that the alternating updates correspond to. Remember we had the following alternating updates:

$$\begin{aligned}\mathbf{x}^{t+1} &= \text{prox}_{\eta_1 \lambda R_{\mathbf{c}^t}}(\mathbf{x}^t - \eta_1 \nabla f(\mathbf{x}^t) - \eta_1 \nabla_{\mathbf{x}^t} f(\widetilde{Q}_{\mathbf{c}^t}(\mathbf{x}^t))) \\ \mathbf{c}^{t+1} &= \text{prox}_{\eta_2 \lambda R_{\mathbf{x}^{t+1}}}(\mathbf{c}^t - \eta_2 \nabla_{\mathbf{c}^t} f(\widetilde{Q}_{\mathbf{c}^t}(\mathbf{x}^{t+1})))\end{aligned}$$

For $\mathbf{x}^{t+1}$, from the definition of proximal mapping we have:

$$\begin{aligned}\mathbf{x}^{t+1} &= \arg\min_{\mathbf{x}\in\mathbb{R}^d}\left\{\frac{1}{2\eta_1}\left\|\mathbf{x} - \mathbf{x}^t + \eta_1 \nabla_{\mathbf{x}^t} f\left(\mathbf{x}^t\right) + \eta_1 \nabla_{\mathbf{x}^t} f(\widetilde{Q}_{\mathbf{c}^t}(\mathbf{x}^t))\right\|_2^2 + \lambda R(\mathbf{x}, \mathbf{c}^t)\right\} \\ &= \arg\min_{\mathbf{x}\in\mathbb{R}^d}\Bigg\{\left\langle \mathbf{x} - \mathbf{x}^t, \nabla_{\mathbf{x}^t} f\left(\mathbf{x}^t\right)\right\rangle + \left\langle \mathbf{x} - \mathbf{x}^t, \nabla_{\mathbf{x}^t} f(\widetilde{Q}_{\mathbf{c}^t}(\mathbf{x}^t))\right\rangle + \frac{1}{2\eta_1}\left\|\mathbf{x} - \mathbf{x}^t\right\|_2^2 \\ &\qquad + \frac{\eta_1}{2}\|\nabla_{\mathbf{x}^t} f(\mathbf{x}^t) + \nabla_{\mathbf{x}^t} f(\widetilde{Q}_{\mathbf{c}^t}(\mathbf{x}^t))\|^2 + \lambda R(\mathbf{x}, \mathbf{c}^t)\Bigg\} \\ &= \arg\min_{\mathbf{x}\in\mathbb{R}^d}\Bigg\{\left\langle \mathbf{x} - \mathbf{x}^t, \nabla_{\mathbf{x}^t} f\left(\mathbf{x}^t\right)\right\rangle + \left\langle \mathbf{x} - \mathbf{x}^t, \nabla_{\mathbf{x}^t} f(\widetilde{Q}_{\mathbf{c}^t}(\mathbf{x}^t))\right\rangle + \frac{1}{2\eta_1}\left\|\mathbf{x} - \mathbf{x}^t\right\|_2^2 \\ &\qquad + \lambda R(\mathbf{x}, \mathbf{c}^t)\Bigg\} \qquad \text{(A.16)}\end{aligned}$$

Note, in the third equality we remove the terms that do not depend on $\mathbf{x}$. Similarly, for $\mathbf{c}^{t+1}$ we have:

$$\begin{aligned}
\mathbf{c}^{t+1} &= \arg\min_{\mathbf{c}\in\mathbb{R}^m} \left\{ \frac{1}{2\eta_2} \left\| \mathbf{c} - \mathbf{c}^t + \eta_2 \nabla_{\mathbf{c}^t} f(\widetilde{Q}_{\mathbf{c}^t}(\mathbf{x}^{t+1})) \right\|_2^2 + \lambda R(\mathbf{x}^{t+1}, \mathbf{c}) \right\} \\
&= \arg\min_{\mathbf{c}\in\mathbb{R}^m} \Bigg\{ \left\langle \mathbf{c} - \mathbf{c}^t, \nabla_{\mathbf{c}^t} f(\widetilde{Q}_{\mathbf{c}^t}(\mathbf{x}^{t+1})) \right\rangle + \frac{1}{2\eta_2} \left\| \mathbf{c} - \mathbf{c}^t \right\|_2^2 + \frac{\eta_2}{2} \|\nabla_{\mathbf{c}^t} f(\widetilde{Q}_{\mathbf{c}^t}(\mathbf{x}^{t+1}))\|^2 \\
&\qquad + \lambda R(\mathbf{x}^{t+1}, \mathbf{c}) \Bigg\} \\
&= \arg\min_{\mathbf{c}\in\mathbb{R}^m} \left\{ \left\langle \mathbf{c} - \mathbf{c}^t, \nabla_{\mathbf{c}^t} f(\widetilde{Q}_{\mathbf{c}^t}(\mathbf{x}^{t+1})) \right\rangle + \frac{1}{2\eta_2} \left\| \mathbf{c} - \mathbf{c}^t \right\|_2^2 + \lambda R(\mathbf{x}^{t+1}, \mathbf{c}) \right\} \qquad \text{(A.17)}
\end{aligned}$$

Minimization problems in (A.16) and (A.17) are the main problems to characterize the update rules and we use them in multiple places throughout the section.

A.2.3.1 Proof of the Claims

Claim (Restating Claim 5)**.** *$f(\mathbf{x}) + f(\widetilde{Q}_{\mathbf{c}}(\mathbf{x}))$ is $(L + GL_{Q_1} + G_{Q_1}LL_{Q_1})$-smooth with respect to $\mathbf{x}$.*

Proof. From our assumptions, we have f is L-smooth. And from Claim 3 we have $f(\widetilde{Q}_{\mathbf{c}}(\mathbf{x}))$ is $(GL_{Q_1} + G_{Q_1}LL_{Q_1})$-smooth. Using the fact that if two functions g_1 and g_2 are L_1 and L_2 smooth respectively, then $g_1 + g_2$ is $(L_1 + L_2)$-smooth concludes the proof. □

Claim (Restating Claim 6)**.** *Let*

$$\begin{aligned}
A(\mathbf{x}^{t+1}) &:= \lambda R(\mathbf{x}^{t+1}, \mathbf{c}^t) + \left\langle \nabla f(\mathbf{x}^t), \mathbf{x}^{t+1} - \mathbf{x}^t \right\rangle + \left\langle \nabla_{\mathbf{x}^t} f(\widetilde{Q}_{\mathbf{c}^t}(\mathbf{x}^t)), \mathbf{x}^{t+1} - \mathbf{x}^t \right\rangle \\
&\qquad + \frac{1}{2\eta_1} \|\mathbf{x}^{t+1} - \mathbf{x}^t\|^2 \\
A(\mathbf{x}^t) &:= \lambda R(\mathbf{x}^t, \mathbf{c}^t).
\end{aligned}$$

Then $A(\mathbf{x}^{t+1}) \leq A(\mathbf{x}^t)$.

Proof. Let $A(\mathbf{x})$ denote the expression inside the arg min in (A.16) and we know that (A.16) is minimized when $\mathbf{x} = \mathbf{x}^{t+1}$. So we have $A(\mathbf{x}^{t+1}) \leq A(\mathbf{x}^t)$. This proves the claim. □

Claim (Restating Claim 7)**.** *Let*

$$B(\mathbf{c}^{t+1}) := \lambda R(\mathbf{x}^{t+1}, \mathbf{c}^{t+1}) + \left\langle \nabla_{\mathbf{c}^t} f(\widetilde{Q}_{\mathbf{c}^t}(\mathbf{x}^{t+1})), \mathbf{c}^{t+1} - \mathbf{c}^t \right\rangle + \frac{1}{2\eta_1}\|\mathbf{c}^{t+1} - \mathbf{c}^t\|^2$$

$$B(\mathbf{c}^t) := \lambda R(\mathbf{x}^{t+1}, \mathbf{c}^t).$$

Then $B(\mathbf{c}^{t+1}) \leq B(\mathbf{c}^t)$.

Proof. Let $B(\mathbf{c})$ denote the expression inside the arg min in (A.17) and we know that (A.17) is minimized when $\mathbf{c} = \mathbf{c}^{t+1}$. So we have $B(\mathbf{c}^{t+1}) \leq B(\mathbf{c}^t)$. This proves the claim. □

A.2.4 Proof of Theorem 2

In this part, different than Section 2.3, we have an additional update due to local iterations. The key is to integrate the local iterations into our alternating update scheme. To do this, we utilize Assumptions **A.6** and **A.7**. This proof consists of two parts. First, we show the sufficient decrease property by sequentially using and combining Lipschitz properties for each update step in Algorithm 4. Then, we bound the norm of the gradient using optimality conditions of the proximal updates in Algorithm 4. Then, by combining the sufficient decrease results and bounds on partial gradients we will derive our result. We defer proofs of the claims and some derivation details to the end of this section. In this analysis we take $\mathbf{w}^t = \frac{1}{n}\sum_{i=1}^n \mathbf{w}_i^t$, so that $\mathbf{w}^t$ is defined for every time point.

Alternating updates. Let us first restate the alternating updates for $\mathbf{x}_i$ and $\mathbf{c}_i$:

$$\mathbf{x}_i^{t+1} = \text{prox}_{\eta_1 \lambda R_{\mathbf{c}_i^t}}\Bigg(\mathbf{x}_i^t - (1-\lambda_p)\eta_1 \nabla f_i(\mathbf{x}_i^t) - (1-\lambda_p)\eta_1 \nabla_{\mathbf{x}_i^t} f_i(\widetilde{Q}_{\mathbf{c}_i^t}(\mathbf{x}_i^t)) \\ - \eta_1 \lambda_p \nabla_{\mathbf{x}_i^t} f_i^{KD}(\mathbf{x}_i^t, \mathbf{w}_i^t) - \eta_1 \lambda_p \nabla_{\mathbf{x}_i^t} f_i^{KD}(\widetilde{Q}_{\mathbf{c}_i^t}(\mathbf{x}_i^t), \mathbf{w}_i^t)\Bigg)$$

$$\mathbf{c}_i^{t+1} = \text{prox}_{\eta_2 \lambda R_{\mathbf{x}_i^{t+1}}}\left(\mathbf{c}_i^t - (1-\lambda_p)\eta_2 \nabla_{\mathbf{c}_i^t} f_i(\widetilde{Q}_{\mathbf{c}_i^t}(\mathbf{x}_i^{t+1})) - \eta_2 \lambda_p \nabla_{\mathbf{c}_i^t} f_i^{KD}(\widetilde{Q}_{\mathbf{c}_i^t}(\mathbf{x}_i^{t+1}), \mathbf{w}_i^t)\right)$$

The alternating updates are equivalent to solving the following two optimization problems.

$$\mathbf{x}_i^{t+1} = \arg\min_{\mathbf{x}\in\mathbb{R}^d}\Bigg\{(1-\lambda_p)\left\langle \mathbf{x} - \mathbf{x}_i^t, \nabla f_i\left(\mathbf{x}_i^t\right)\right\rangle + (1-\lambda_p)\left\langle \mathbf{x} - \mathbf{x}_i^t, \nabla_{\mathbf{x}_i^t} f_i(\widetilde{Q}_{\mathbf{c}_i^t}(\mathbf{x}_i^t))\right\rangle$$

$$+\left\langle \mathbf{x}-\mathbf{x}_i^t, \lambda_p \nabla_{\mathbf{x}_i^t} f_i^{KD}(\mathbf{x}_i^t, \mathbf{w}_i^t)\right\rangle + \left\langle \mathbf{x}-\mathbf{x}_i^t, \lambda_p \nabla_{\mathbf{x}_i^t} f_i^{KD}(\widetilde{Q}_{\mathbf{c}_i^t}(\mathbf{x}_i^t), \mathbf{w}_i^t)\right\rangle$$
$$+\frac{1}{2\eta_1}\left\|\mathbf{x}-\mathbf{x}_i^t\right\|_2^2 + \lambda R(\mathbf{x}, \mathbf{c}_i^t)\Bigg\} \tag{A.18}$$

$$\mathbf{c}_i^{t+1} = \arg\min_{\mathbf{c}\in\mathbb{R}^m}\Bigg\{ \left\langle \mathbf{c}-\mathbf{c}_i^t, (1-\lambda_p)\nabla_{\mathbf{c}_i^t} f_i(\widetilde{Q}_{\mathbf{c}_i^t}(\mathbf{x}_i^{t+1}))\right\rangle + \left\langle \mathbf{c}-\mathbf{c}_i^t, \lambda_p \nabla_{\mathbf{c}_i^t} f_i^{KD}(\widetilde{Q}_{\mathbf{c}_i^t}(\mathbf{x}_i^{t+1}), \mathbf{w}_i^t)\right\rangle$$
$$+\frac{1}{2\eta_2}\left\|\mathbf{c}-\mathbf{c}_i^t\right\|_2^2 + \lambda R(\mathbf{x}_i^{t+1}, \mathbf{c})\Bigg\} \tag{A.19}$$

Note that the update on $\mathbf{w}^t$ from Algorithm 4 can be written as:

$$\mathbf{w}^{t+1} = \mathbf{w}^t - \eta_3 \mathbf{g}^t, \quad \text{where} \quad \mathbf{g}^t = \frac{1}{n}\sum_{i=1}^n \nabla_{\mathbf{w}_i^t} F_i(\mathbf{x}_i^{t+1}, \mathbf{c}_i^{t+1}, \mathbf{w}_i^t).$$

In the convergence analysis we require smoothness of the local functions F_i w.r.t. the global parameter $\mathbf{w}$. Recall the definition of $F_i(\mathbf{x}_i, \mathbf{c}_i, \mathbf{w}) = (1-\lambda_p)\left(f_i(\mathbf{x}_i) + f_i(\widetilde{Q}_{\mathbf{c}_i}(\mathbf{x}_i))\right) + \lambda R(\mathbf{x}_i, \mathbf{c}_i) + \lambda_p\left(f_i^{KD}(\mathbf{x}_i, \mathbf{w}) + f_i^{KD}(\widetilde{Q}_{\mathbf{c}_i}(\mathbf{x}_i), \mathbf{w})\right)$ from (2.2). It follows that from Assumption **A**.7 that F_i is $(\lambda_p(L_{D_2} + L_{DQ_3}))$-smooth with respect to $\mathbf{w}$: Now let us move on with the proof.

A.2.5 Sufficient Decrease

We will divide this part into three and obtain sufficient decrease properties for each variable: $\mathbf{x}_i, \mathbf{c}_i, \mathbf{w}$.

A.2.5.1 Sufficient Decrease Due to $\mathbf{x}_i$

We begin with a useful claim.

Claim 8. $(1-\lambda_p)(f_i(\mathbf{x}) + f_i(\widetilde{Q}_{\mathbf{c}}(\mathbf{x}))) + \lambda_p(f_i^{KD}(\mathbf{x}, \mathbf{w}) + f_i^{KD}(\widetilde{Q}_{\mathbf{c}}(\mathbf{x}), \mathbf{w}))$ *is* $(\lambda_p(L_{D_1} + L_{DQ_1}) + (1-\lambda_p)(L + G^{(i)}L_{Q_1} + G_{Q_1}^{(i)} L l_{Q_1}))$*-smooth with respect to* $\mathbf{x}$.

Proof. From our assumptions, we have f_i is L-smooth, $f_i^{KD}(\mathbf{x}, \mathbf{w})$ is L_{D_1}-smooth and $f_i^{KD}(\widetilde{Q}_{\mathbf{c}}(\mathbf{x}), \mathbf{w})$ is L_{DQ_1}-smooth with respect to $\mathbf{x}$. And applying the Claim 3 to each

client separately gives that $f_i(\widetilde{Q}_{\mathbf{c}}(\mathbf{x}))$ is $(G^{(i)}L_{Q_1} + G^{(i)}_{Q_1}Ll_{Q_1})$-smooth. Using the fact that if two functions g_1 and g_2 (defined over the same space) are L_1 and L_2-smooth respectively, then $g_1 + g_2$ is $(L_1 + L_2)$-smooth, and the fact that αg_1 is αL_1-smooth for a given constant α concludes the proof. □

From Claim 8 we have:

$$
\begin{aligned}
&F_i(\mathbf{x}_i^{t+1}, \mathbf{c}_i^t, \mathbf{w}^t) + (\frac{1}{2\eta_1} - \frac{\lambda_p(L_{D_1} + L_{DQ_1}) + (1-\lambda_p)(L + G^{(i)}L_{Q_1} + G^{(i)}_{Q_1}Ll_{Q_1})}{2})\|\mathbf{x}_i^{t+1} - \mathbf{x}_i^t\|^2 \\
&= (1-\lambda_p)\left(f_i(\mathbf{x}_i^{t+1}) + f_i(\widetilde{Q}_{\mathbf{c}_i^t}(\mathbf{x}_i^{t+1}))\right) + \lambda_p\left(f_i^{KD}(\mathbf{x}_i^{t+1}, \mathbf{w}^t) + f_i^{KD}(\widetilde{Q}_{\mathbf{c}_i^t}(\mathbf{x}_i^{t+1}), \mathbf{w}^t)\right) \\
&\quad + (\frac{1}{2\eta_1} - \frac{\lambda_p(L_{D_1} + L_{DQ_1}) + (1-\lambda_p)(L + G^{(i)}L_{Q_1} + G^{(i)}_{Q_1}Ll_{Q_1})}{2})\|\mathbf{x}_i^{t+1} - \mathbf{x}_i^t\|^2 \\
&\quad + \lambda R(\mathbf{x}_i^{t+1}, \mathbf{c}_i^t) \\
&\leq (1-\lambda_p)\left(f_i(\mathbf{x}_i^t) + f_i(\widetilde{Q}_{\mathbf{c}_i^t}(\mathbf{x}_i^t))\right) + \lambda_p\left(f_i^{KD}(\mathbf{x}_i^t, \mathbf{w}^t) + f_i^{KD}(\widetilde{Q}_{\mathbf{c}_i^t}(\mathbf{x}_i^t), \mathbf{w}^t)\right) \\
&\quad + (1-\lambda_p)\left\langle \nabla f_i(\mathbf{x}_i^t), \mathbf{x}_i^{t+1} - \mathbf{x}_i^t \right\rangle + (1-\lambda_p)\left\langle \nabla_{\mathbf{x}_i^t} f_i(\widetilde{Q}_{\mathbf{c}_i^t}(\mathbf{x}_i^t)), \mathbf{x}_i^{t+1} - \mathbf{x}_i^t \right\rangle \\
&\quad + \lambda_p\left\langle \nabla_{\mathbf{x}_i^t} f_i^{KD}(\mathbf{x}_i^t, \mathbf{w}^t), \mathbf{x}_i^{t+1} - \mathbf{x}_i^t \right\rangle + \lambda_p\left\langle \nabla_{\mathbf{x}_i^t} f_i^{KD}(\widetilde{Q}_{\mathbf{c}_i^t}(\mathbf{x}_i^t), \mathbf{w}^t), \mathbf{x}_i^{t+1} - \mathbf{x}_i^t \right\rangle \\
&\quad + \lambda R(\mathbf{x}_i^{t+1}, \mathbf{c}_i^t) + \frac{1}{2\eta_1}\|\mathbf{x}_i^{t+1} - \mathbf{x}_i^t\|^2 \\
&= (1-\lambda_p)\left(f_i(\mathbf{x}_i^t) + f_i(\widetilde{Q}_{\mathbf{c}_i^t}(\mathbf{x}_i^t))\right) + \lambda_p\left(f_i^{KD}(\mathbf{x}_i^t, \mathbf{w}^t) + f_i^{KD}(\widetilde{Q}_{\mathbf{c}_i^t}(\mathbf{x}_i^t), \mathbf{w}^t)\right) \\
&\quad + (1-\lambda_p)\left\langle \nabla f_i(\mathbf{x}_i^t), \mathbf{x}_i^{t+1} - \mathbf{x}_i^t \right\rangle + (1-\lambda_p)\left\langle \nabla_{\mathbf{x}_i^t} f_i(\widetilde{Q}_{\mathbf{c}_i^t}(\mathbf{x}_i^t)), \mathbf{x}_i^{t+1} - \mathbf{x}_i^t \right\rangle \\
&\quad + \lambda_p\left\langle \nabla_{\mathbf{x}_i^t} f_i^{KD}(\mathbf{x}_i^t, \mathbf{w}^t) - \nabla_{\mathbf{x}_i^t} f_i^{KD}(\mathbf{x}_i^t, \mathbf{w}_i^t), \mathbf{x}_i^{t+1} - \mathbf{x}_i^t \right\rangle \\
&\quad + \lambda_p\left\langle \nabla_{\mathbf{x}_i^t} f_i^{KD}(\widetilde{Q}_{\mathbf{c}_i^t}(\mathbf{x}_i^t), \mathbf{w}^t) - \nabla_{\mathbf{x}_i^t} f_i^{KD}(\widetilde{Q}_{\mathbf{c}_i^t}(\mathbf{x}_i^t), \mathbf{w}_i^t), \mathbf{x}_i^{t+1} - \mathbf{x}_i^t \right\rangle \\
&\quad + \lambda_p\left\langle \nabla_{\mathbf{x}_i^t} f_i^{KD}(\mathbf{x}_i^t, \mathbf{w}_i^t), \mathbf{x}_i^{t+1} - \mathbf{x}_i^t \right\rangle + \lambda_p\left\langle \nabla_{\mathbf{x}_i^t} f_i^{KD}((\widetilde{Q}_{\mathbf{c}_i^t}(\mathbf{x}_i^t), \mathbf{w}_i^t), \mathbf{x}_i^{t+1} - \mathbf{x}_i^t \right\rangle \\
&\quad + \frac{1}{2\eta_1}\|\mathbf{x}_i^{t+1} - \mathbf{x}_i^t\|^2 + \lambda R(\mathbf{x}_i^{t+1}, \mathbf{c}_i^t) \\
&\leq (1-\lambda_p)\left(f_i(\mathbf{x}_i^t) + f_i(\widetilde{Q}_{\mathbf{c}_i^t}(\mathbf{x}_i^t))\right) + \lambda_p\left(f_i^{KD}(\mathbf{x}_i^t, \mathbf{w}^t) + f_i^{KD}(\widetilde{Q}_{\mathbf{c}_i^t}(\mathbf{x}_i^t), \mathbf{w}^t)\right)
\end{aligned}
$$

$$
\begin{aligned}
&+ (1-\lambda_p)\left\langle \nabla f_i(\mathbf{x}_i^t), \mathbf{x}_i^{t+1} - \mathbf{x}_i^t \right\rangle + (1-\lambda_p)\left\langle \nabla_{\mathbf{x}_i^t} f_i(\widetilde{Q}_{\mathbf{c}_i^t}(\mathbf{x}_i^t)), \mathbf{x}_i^{t+1} - \mathbf{x}_i^t \right\rangle \\
&+ \lambda_p \left\langle \nabla_{\mathbf{x}_i^t} f_i^{KD}(\mathbf{x}_i^t, \mathbf{w}_i^t), \mathbf{x}_i^{t+1} - \mathbf{x}_i^t \right\rangle + \lambda_p \left\langle \nabla_{\mathbf{x}_i^t} f_i^{KD}((\widetilde{Q}_{\mathbf{c}_i^t}(\mathbf{x}_i^t), \mathbf{w}_i^t), \mathbf{x}_i^{t+1} - \mathbf{x}_i^t \right\rangle \\
&+ \frac{\lambda_p}{2}\|\nabla_{\mathbf{x}_i^t} f_i^{KD}(\widetilde{Q}_{\mathbf{c}_i^t}(\mathbf{x}_i^t), \mathbf{w}^t) - \nabla_{\mathbf{x}_i^t} f_i^{KD}(\widetilde{Q}_{\mathbf{c}_i^t}(\mathbf{x}_i^t), \mathbf{w}_i^t)\|^2 + \lambda_p \|\mathbf{x}_i^{t+1} - \mathbf{x}_i^t\|^2 \\
&+ \frac{\lambda_p}{2}\|\nabla_{\mathbf{x}_i^t} f_i^{KD}(\mathbf{x}_i^t, \mathbf{w}^t) - \nabla_{\mathbf{x}_i^t} f_i^{KD}(\mathbf{x}_i^t, \mathbf{w}_i^t)\|^2 + \frac{1}{2\eta_1}\|\mathbf{x}_i^{t+1} - \mathbf{x}_i^t\|^2 + \lambda R(\mathbf{x}_i^{t+1}, \mathbf{c}_i^t). \quad \text{(A.20)}
\end{aligned}
$$

In the last inequality we used:

$$
\begin{aligned}
&\left\langle \lambda_p(\nabla_{\mathbf{x}_i^t} f_i^{KD}(\mathbf{x}_i^t, \mathbf{w}^t) - \nabla_{\mathbf{x}_i^t} f_i^{KD}(\mathbf{x}_i^t, \mathbf{w}_i^t)), \mathbf{x}_i^{t+1} - \mathbf{x}_i^t \right\rangle \\
&\qquad = \left\langle \sqrt{\lambda_p}(\nabla_{\mathbf{x}_i^t} f_i^{KD}(\mathbf{x}_i^t, \mathbf{w}^t) - \nabla_{\mathbf{x}_i^t} f_i^{KD}(\mathbf{x}_i^t, \mathbf{w}_i^t)), \sqrt{\lambda_p}(\mathbf{x}_i^{t+1} - \mathbf{x}_i^t) \right\rangle \\
&\qquad \le \frac{\lambda_p}{2}\|\nabla_{\mathbf{x}_i^t} f_i^{KD}(\mathbf{x}_i^t, \mathbf{w}^t) - \nabla_{\mathbf{x}_i^t} f_i^{KD}(\mathbf{x}_i^t, \mathbf{w}_i^t)\|^2 + \frac{\lambda_p}{2}\|\mathbf{x}_i^{t+1} - \mathbf{x}_i^t\|^2
\end{aligned}
$$

and similarly,

$$
\begin{aligned}
&\left\langle \lambda_p(\nabla_{\mathbf{x}_i^t} f_i^{KD}(\widetilde{Q}_{\mathbf{c}_i^t}(\mathbf{x}_i^t), \mathbf{w}^t) - \nabla_{\mathbf{x}_i^t} f_i^{KD}(\widetilde{Q}_{\mathbf{c}_i^t}(\mathbf{x}_i^t), \mathbf{w}_i^t)), \mathbf{x}_i^{t+1} - \mathbf{x}_i^t \right\rangle \\
&\qquad \le \frac{\lambda_p}{2}\|\nabla_{\mathbf{x}_i^t} f_i^{KD}(\widetilde{Q}_{\mathbf{c}_i^t}(\mathbf{x}_i^t), \mathbf{w}^t) - \nabla_{\mathbf{x}_i^t} f_i^{KD}(\widetilde{Q}_{\mathbf{c}_i^t}(\mathbf{x}_i^t), \mathbf{w}_i^t)\|^2 \\
&\qquad\quad + \frac{\lambda_p}{2}\|\mathbf{x}_i^{t+1} - \mathbf{x}_i^t\|^2
\end{aligned}
$$

Claim 9. *Let*

$$
\begin{aligned}
A(\mathbf{x}_i^{t+1}) &:= (1-\lambda_p)\left\langle \nabla f_i(\mathbf{x}_i^t), \mathbf{x}_i^{t+1} - \mathbf{x}_i^t \right\rangle + (1-\lambda_p)\left\langle \nabla_{\mathbf{x}_i^t} f_i(\widetilde{Q}_{\mathbf{c}_i^t}(\mathbf{x}_i^t)), \mathbf{x}_i^{t+1} - \mathbf{x}_i^t \right\rangle \\
&\quad + \left\langle \lambda_p(\nabla_{\mathbf{x}_i^t} f_i^{KD}(\mathbf{x}_i^t, \mathbf{w}_i^t)), \mathbf{x}_i^{t+1} - \mathbf{x}_i^t \right\rangle + \left\langle \lambda_p(\nabla_{\mathbf{x}_i^t} f_i^{KD}(\widetilde{Q}_{\mathbf{c}_i^t}(\mathbf{x}_i^t), \mathbf{w}_i^t)), \mathbf{x}_i^{t+1} - \mathbf{x}_i^t \right\rangle \\
&\quad + \lambda R(\mathbf{x}_i^{t+1}, \mathbf{c}_i^t) + \frac{1}{2\eta_1}\|\mathbf{x}_i^{t+1} - \mathbf{x}_i^t\|^2 \\
A(\mathbf{x}_i^t) &:= \lambda R(\mathbf{x}_i^t, \mathbf{c}_i^t).
\end{aligned}
$$

Then $A(\mathbf{x}_i^{t+1}) \le A(\mathbf{x}_i^t)$.

Proof. Let $A(\mathbf{x})$ denote the expression inside the arg min in (A.18) and we know that (A.18) is minimized when $\mathbf{x} = \mathbf{x}_i^{t+1}$. So we have $A(\mathbf{x}_i^{t+1}) \le A(\mathbf{x}_i^t)$. This proves the claim. □

Using the inequality from Claim 9 in (A.20) gives

$$
\begin{aligned}
&F_i(\mathbf{x}_i^{t+1}, \mathbf{c}_i^t, \mathbf{w}^t) + \left(\frac{1}{2\eta_1} - \frac{\lambda_p(L_{D_1} + L_{DQ_1}) + (1-\lambda_p)(L + G^{(i)}L_{Q_1} + G_{Q_1}^{(i)} L l_{Q_1})}{2}\right)\|\mathbf{x}_i^{t+1} - \mathbf{x}_i^t\|^2 \\
&\overset{(a)}{\leq} (1-\lambda_p)\left(f_i(\mathbf{x}_i^t) + f_i(\widetilde{Q}_{\mathbf{c}_i^t}(\mathbf{x}_i^t))\right) + \lambda_p\left(f_i^{KD}(\mathbf{x}_i^t, \mathbf{w}^t) + f_i^{KD}(\widetilde{Q}_{\mathbf{c}_i^t}(\mathbf{x}_i^t), \mathbf{w}^t)\right) + A(\mathbf{x}_i^{t+1}) \\
&\quad + \frac{\lambda_p}{2}\|\nabla_{\mathbf{x}_i^t} f_i^{KD}(\mathbf{x}_i^t, \mathbf{w}^t) - \nabla_{\mathbf{x}_i^t} f_i^{KD}(\mathbf{x}_i^t, \mathbf{w}_i^t)\|^2 + \frac{\lambda_p}{2}\|\nabla_{\mathbf{x}_i^t} f_i^{KD}(\widetilde{Q}_{\mathbf{c}_i^t}(\mathbf{x}_i^t), \mathbf{w}^t) \\
&\quad - \nabla_{\mathbf{x}_i^t} f_i^{KD}(\widetilde{Q}_{\mathbf{c}_i^t}(\mathbf{x}_i^t), \mathbf{w}_i^t)\|^2 + \lambda_p\|\mathbf{x}_i^{t+1} - \mathbf{x}_i^t\|^2 \\
&\overset{(b)}{\leq} (1-\lambda_p)\left(f_i(\mathbf{x}_i^t) + f_i(\widetilde{Q}_{\mathbf{c}_i^t}(\mathbf{x}_i^t))\right) + \lambda_p\left(f_i^{KD}(\mathbf{x}_i^t, \mathbf{w}^t) + f_i^{KD}(\widetilde{Q}_{\mathbf{c}_i^t}(\mathbf{x}_i^t), \mathbf{w}^t)\right) + \lambda R(\mathbf{x}_i^t, \mathbf{c}_i^t) \\
&\quad + \frac{\lambda_p}{2}\|\nabla f_i^{KD}(\mathbf{x}_i^t, \mathbf{w}^t) - \nabla f_i^{KD}(\mathbf{x}_i^t, \mathbf{w}_i^t)\|^2 + \frac{\lambda_p}{2}\|\nabla f_i^{KD}(\widetilde{Q}_{\mathbf{c}_i^t}(\mathbf{x}_i^t), \mathbf{w}^t) \\
&\quad - \nabla f_i^{KD}(\widetilde{Q}_{\mathbf{c}_i^t}(\mathbf{x}_i^t), \mathbf{w}_i^t)\|^2 + \lambda_p\|\mathbf{x}_i^{t+1} - \mathbf{x}_i^t\|^2 \\
&\overset{(c)}{\leq} (1-\lambda_p)\left(f_i(\mathbf{x}_i^t) + f_i(\widetilde{Q}_{\mathbf{c}_i^t}(\mathbf{x}_i^t))\right) + \lambda_p\left(f_i^{KD}(\mathbf{x}_i^t, \mathbf{w}^t) + f_i^{KD}(\widetilde{Q}_{\mathbf{c}_i^t}(\mathbf{x}_i^t), \mathbf{w}^t)\right) \\
&\quad + \lambda R(\mathbf{x}_i^t, \mathbf{c}_i^t) + \frac{\lambda_p(L_D^2 + L_{DQ}^2)}{2}\|\mathbf{w}^t - \mathbf{w}_i^t\|^2 + \lambda_p\|\mathbf{x}_i^{t+1} - \mathbf{x}_i^t\|^2 \\
&= F_i(\mathbf{x}_i^t, \mathbf{c}_i^t, \mathbf{w}^t) + \frac{\lambda_p(L_D^2 + L_{DQ}^2)}{2}\|\mathbf{w}_i^t - \mathbf{w}^t\|^2 + \lambda_p\|\mathbf{x}_i^{t+1} - \mathbf{x}_i^t\|^2. \qquad \text{(A.21)}
\end{aligned}
$$

To obtain (a), we substituted the value of $A(\mathbf{x}_i^{t+1})$ from Claim 9 into (A.20). In (b), we used $A(\mathbf{x}_i^{t+1}) \leq \lambda R(\mathbf{x}_i^t, \mathbf{c}_i^t)$, $\|\nabla_{\mathbf{x}_i^t} f_i^{KD}(\mathbf{x}_i^t, \mathbf{w}^t) - \nabla_{\mathbf{x}_i^t} f_i^{KD}(\mathbf{x}_i^t, \mathbf{w}_i^t)\|^2 \leq \|\nabla f_i^{KD}(\mathbf{x}_i^t, \mathbf{w}^t) - \nabla f_i^{KD}(\mathbf{x}_i^t, \mathbf{w}_i^t)\|^2$ and $\|\nabla_{\mathbf{x}_i^t} f_i^{KD}(\widetilde{Q}_{\mathbf{c}_i^t}(\mathbf{x}_i^t)), \mathbf{w}^t) - \nabla_{\mathbf{x}_i^t} f_i^{KD}(\widetilde{Q}_{\mathbf{c}_i^t}(\mathbf{x}_i^t)), \mathbf{w}_i^t)\|^2 \leq \|\nabla f_i^{KD}(\mathbf{x}_i^t, \mathbf{w}^t) - \nabla f_i^{KD}(\mathbf{x}_i^t, \mathbf{w}_i^t)\|^2$. And in (c) we used the assumption that $f_i^{KD}(\mathbf{x}, \mathbf{w})$ is L_D-smooth and $f_i^{KD}(\widetilde{Q}_{\mathbf{c}}(\mathbf{x}), \mathbf{w})$ is L_{DQ}-smooth.

Substituting $\eta_1 = \frac{1}{2(\lambda_p(2+L_{D_1}+L_{DQ_1})+(1-\lambda_p)(L+G^{(i)}L_{Q_1}+G_{Q_1}^{(i)}Ll_{Q_1}))}$ in (A.21) gives:

$$
\begin{aligned}
&F_i(\mathbf{x}_i^{t+1}, \mathbf{c}_i^t, \mathbf{w}^t) + \left(\frac{\lambda_p(2 + L_{D_1} + L_{DQ_1}) + (1-\lambda_p)(L + G^{(i)}L_{Q_1} + G_{Q_1}^{(i)} L l_{Q_1})}{2}\right)\|\mathbf{x}_i^{t+1} - \mathbf{x}_i^t\|^2 \\
&\qquad \leq F_i(\mathbf{x}_i^t, \mathbf{c}_i^t, \mathbf{w}^t) + \frac{\lambda_p(L_D^2 + L_{DQ}^2)}{2}\|\mathbf{w}_i^t - \mathbf{w}^t\|^2. \qquad \text{(A.22)}
\end{aligned}
$$

A.2.5.2 Sufficient Decrease Due to $\mathbf{c}_i$

In parallel with Claim 8, we have following smoothness result for $\mathbf{c}$:

Claim 10. $(1-\lambda_p)f_i(\widetilde{Q}_{\mathbf{c}}(\mathbf{x}))+\lambda_p f_i^{KD}(\widetilde{Q}_{\mathbf{c}}(\mathbf{x}),\mathbf{w}))$ *is* $(\lambda_p L_{DQ_2}+(1-\lambda_p)(G^{(i)}L_{Q_2}+G^{(i)}_{Q_2}Ll_{Q_2}))$-*smooth with respect to* $\mathbf{c}$.

From Claim 10 we have:

$$
\begin{aligned}
&F_i(\mathbf{x}_i^{t+1},\mathbf{c}_i^{t+1},\mathbf{w}^t)+(\frac{1}{2\eta_2}-\frac{\lambda_p L_{DQ_2}+(1-\lambda_p)(G^{(i)}L_{Q_2}+G^{(i)}_{Q_2}Ll_{Q_2})}{2})\|\mathbf{c}_i^{t+1}-\mathbf{c}_i^t\|^2\\
&\quad=(1-\lambda_p)\left(f_i(\mathbf{x}_i^{t+1})+f_i(\widetilde{Q}_{\mathbf{c}_i^{t+1}}(\mathbf{x}_i^{t+1}))\right)\\
&\qquad+\lambda R(\mathbf{x}_i^{t+1},\mathbf{c}_i^{t+1})+\lambda_p\left(f_i^{KD}(\mathbf{x}_i^{t+1},\mathbf{w}^t)+f_i^{KD}(\widetilde{Q}_{\mathbf{c}_i^{t+1}}(\mathbf{x}_i^{t+1})),\mathbf{w}^t)\right)\\
&\qquad+(\frac{1}{2\eta_2}-\frac{\lambda_p L_{DQ_2}+(1-\lambda_p)(G^{(i)}L_{Q_2}+G^{(i)}_{Q_2}Ll_{Q_2})}{2})\|\mathbf{c}_i^{t+1}-\mathbf{c}_i^t\|^2\\
&\quad\le(1-\lambda_p)\left(f_i(\mathbf{x}_i^{t+1})+f_i(\widetilde{Q}_{\mathbf{c}_i^{t}}(\mathbf{x}_i^{t+1}))\right)+\lambda R(\mathbf{x}_i^{t+1},\mathbf{c}_i^{t+1})\\
&\qquad+\lambda_p\left(f_i^{KD}(\mathbf{x}_i^{t+1},\mathbf{w}^t)+f_i^{KD}(\widetilde{Q}_{\mathbf{c}_i^{t}}(\mathbf{x}_i^{t+1}),\mathbf{w}^t)\right)+\frac{1}{2\eta_2}\|\mathbf{c}_i^{t+1}-\mathbf{c}_i^t\|^2\\
&\qquad+(1-\lambda_p)\left\langle\nabla_{\mathbf{c}_i^t}f_i(\widetilde{Q}_{\mathbf{c}_i^t}(\mathbf{x}_i^{t+1})),\mathbf{c}_i^{t+1}-\mathbf{c}_i^t\right\rangle+\lambda_p\left\langle\nabla_{\mathbf{c}_i^t}f_i^{KD}(\widetilde{Q}_{\mathbf{c}_i^t}(\mathbf{x}_i^{t+1}),\mathbf{w}^t),\mathbf{c}_i^{t+1}-\mathbf{c}_i^t\right\rangle\\
&\quad=(1-\lambda_p)\left(f_i(\mathbf{x}_i^{t+1})+f_i(\widetilde{Q}_{\mathbf{c}_i^{t}}(\mathbf{x}_i^{t+1}))\right)+\lambda R(\mathbf{x}_i^{t+1},\mathbf{c}_i^{t+1})\\
&\qquad+\lambda_p\left(f_i^{KD}(\mathbf{x}_i^{t+1},\mathbf{w}^t)+f_i^{KD}(\widetilde{Q}_{\mathbf{c}_i^{t}}(\mathbf{x}_i^{t+1}),\mathbf{w}^t)\right)+\frac{1}{2\eta_2}\|\mathbf{c}_i^{t+1}-\mathbf{c}_i^t\|^2\\
&\qquad+(1-\lambda_p)\left\langle\nabla_{\mathbf{c}_i^t}f_i(\widetilde{Q}_{\mathbf{c}_i^t}(\mathbf{x}_i^{t+1})),\mathbf{c}_i^{t+1}-\mathbf{c}_i^t\right\rangle+\lambda_p\left\langle\nabla_{\mathbf{c}_i^t}f_i^{KD}(\widetilde{Q}_{\mathbf{c}_i^t}(\mathbf{x}_i^{t+1}),\mathbf{w}_i^t),\mathbf{c}_i^{t+1}-\mathbf{c}_i^t\right\rangle\\
&\qquad+\lambda_p\left\langle\nabla_{\mathbf{c}_i^t}f_i^{KD}(\widetilde{Q}_{\mathbf{c}_i^t}(\mathbf{x}_i^{t+1}),\mathbf{w}^t)-\nabla_{\mathbf{c}_i^t}f_i^{KD}(\widetilde{Q}_{\mathbf{c}_i^t}(\mathbf{x}_i^{t+1}),\mathbf{w}_i^t),\mathbf{c}_i^{t+1}-\mathbf{c}_i^t\right\rangle\\
&\quad\le(1-\lambda_p)\left(f_i(\mathbf{x}_i^{t+1})+f_i(\widetilde{Q}_{\mathbf{c}_i^{t}}(\mathbf{x}_i^{t+1}))\right)+\lambda R(\mathbf{x}_i^{t+1},\mathbf{c}_i^{t+1})\\
&\qquad+\lambda_p\left(f_i^{KD}(\mathbf{x}_i^{t+1},\mathbf{w}^t)+f_i^{KD}(\widetilde{Q}_{\mathbf{c}_i^{t}}(\mathbf{x}_i^{t+1}),\mathbf{w}^t)\right)+\frac{1}{2\eta_2}\|\mathbf{c}_i^{t+1}-\mathbf{c}_i^t\|^2\\
&\qquad+(1-\lambda_p)\left\langle\nabla_{\mathbf{c}_i^t}f_i(\widetilde{Q}_{\mathbf{c}_i^t}(\mathbf{x}_i^{t+1})),\mathbf{c}_i^{t+1}-\mathbf{c}_i^t\right\rangle+\lambda_p\left\langle\nabla_{\mathbf{c}_i^t}f_i^{KD}(\widetilde{Q}_{\mathbf{c}_i^t}(\mathbf{x}_i^{t+1}),\mathbf{w}_i^t),\mathbf{c}_i^{t+1}-\mathbf{c}_i^t\right\rangle\\
&\qquad+\frac{\lambda_p}{2}\|\nabla_{\mathbf{c}_i^t}f_i^{KD}(\widetilde{Q}_{\mathbf{c}_i^t}(\mathbf{x}_i^{t+1}),\mathbf{w}^t)-\nabla_{\mathbf{c}_i^t}f_i^{KD}(\widetilde{Q}_{\mathbf{c}_i^t}(\mathbf{x}_i^{t+1}),\mathbf{w}_i^t)\|^2+\frac{\lambda_p}{2}\|\mathbf{c}_i^{t+1}-\mathbf{c}_i^t\|^2. \quad \text{(A.23)}
\end{aligned}
$$

in the last inequality we used:

$$
\begin{aligned}
&\left\langle\lambda_p(\nabla_{\mathbf{c}_i^t}f_i^{KD}(\widetilde{Q}_{\mathbf{c}_i^t}(\mathbf{x}_i^{t+1}),\mathbf{w}^t)-\nabla_{\mathbf{c}_i^t}f_i^{KD}(\widetilde{Q}_{\mathbf{c}_i^t}(\mathbf{x}_i^{t+1}),\mathbf{w}_i^t)),\mathbf{c}_i^{t+1}-\mathbf{c}_i^t\right\rangle\\
&\qquad\qquad\le\frac{\lambda_p}{2}\|\nabla_{\mathbf{c}_i^t}f_i^{KD}(\widetilde{Q}_{\mathbf{c}_i^t}(\mathbf{x}_i^{t+1}),\mathbf{w}^t)-\nabla_{\mathbf{c}_i^t}f_i^{KD}(\widetilde{Q}_{\mathbf{c}_i^t}(\mathbf{x}_i^{t+1}),\mathbf{w}_i^t)\|^2
\end{aligned}
$$

$$+ \frac{\lambda_p}{2}\|\mathbf{c}_i^{t+1} - \mathbf{c}_i^t\|^2$$

Claim 11. *Let*

$$\begin{aligned} B(\mathbf{c}_i^{t+1}) &:= \lambda R(\mathbf{x}_i^{t+1}, \mathbf{c}_i^{t+1}) + (1-\lambda_p)\left\langle \nabla_{\mathbf{c}_i^t} f_i(\widetilde{Q}_{\mathbf{c}_i^t}(\mathbf{x}_i^{t+1})), \mathbf{c}_i^{t+1} - \mathbf{c}_i^t \right\rangle \\ &\quad + \lambda_p \left\langle \nabla_{\mathbf{c}_i^t} f_i^{KD}(\widetilde{Q}_{\mathbf{c}_i^t}(\mathbf{x}_i^{t+1}), \mathbf{w}_i^t), \mathbf{c}_i^{t+1} - \mathbf{c}_i^t \right\rangle + \frac{1}{2\eta_2}\|\mathbf{c}_i^{t+1} - \mathbf{c}_i^t\|^2 \\ B(\mathbf{c}_i^t) &:= \lambda R(\mathbf{x}_i^{t+1}, \mathbf{c}_i^t). \end{aligned}$$

Then $B(\mathbf{c}_i^{t+1}) \leq B(\mathbf{c}_i^t)$.

Proof. Let $B(\mathbf{c})$ denote the expression inside the arg min in (A.19) and we know that (A.19) is minimized when $\mathbf{c} = \mathbf{c}_i^{t+1}$. So we have $B(\mathbf{c}_i^{t+1}) \leq B(\mathbf{c}_i^t)$. This proves the claim. □

Substituting the bound from Claim 11 in (A.23),

$$\begin{aligned} &F_i(\mathbf{x}_i^{t+1}, \mathbf{c}_i^{t+1}, \mathbf{w}^t) + (\frac{1}{2\eta_2} - \frac{\lambda_p L_{DQ_2} + (1-\lambda_p)(G^{(i)}L_{Q_2} + G_{Q_2}^{(i)} L l_{Q_2})}{2})\|\mathbf{c}_i^{t+1} - \mathbf{c}_i^t\|^2 \\ &\leq (1-\lambda_p)\left(f_i(\mathbf{x}_i^{t+1}) + f_i(\widetilde{Q}_{\mathbf{c}_i^t}(\mathbf{x}_i^{t+1}))\right) + \lambda_p\left(f_i^{KD}(\mathbf{x}_i^{t+1}, \mathbf{w}^t) + f_i^{KD}(\widetilde{Q}_{\mathbf{c}_i^t}(\mathbf{x}_i^{t+1}), \mathbf{w}^t)\right) \\ &\quad + \frac{\lambda_p}{2}\|\nabla_{\mathbf{c}_i^t} f_i^{KD}(\widetilde{Q}_{\mathbf{c}_i^t}(\mathbf{x}_i^{t+1}), \mathbf{w}^t) - \nabla_{\mathbf{c}_i^t} f_i^{KD}(\widetilde{Q}_{\mathbf{c}_i^t}(\mathbf{x}_i^{t+1}), \mathbf{w}_i^t)\|^2 \\ &\quad + B(\mathbf{c}_i^{t+1}) + \frac{\lambda_p}{2}\|\mathbf{c}_i^{t+1} - \mathbf{c}_i^t\|^2 \\ &\leq (1-\lambda_p)\left(f_i(\mathbf{x}_i^{t+1}) + f_i(\widetilde{Q}_{\mathbf{c}_i^t}(\mathbf{x}_i^{t+1}))\right) + \lambda_p\left(f_i^{KD}(\mathbf{x}_i^{t+1}, \mathbf{w}^t) + f_i^{KD}(\widetilde{Q}_{\mathbf{c}_i^t}(\mathbf{x}_i^{t+1}), \mathbf{w}^t)\right) \\ &\quad + \frac{\lambda_p}{2}\|\nabla f_i^{KD}(\widetilde{Q}_{\mathbf{c}_i^t}(\mathbf{x}_i^{t+1}), \mathbf{w}^t) - \nabla f_i^{KD}(\widetilde{Q}_{\mathbf{c}_i^t}(\mathbf{x}_i^{t+1}), \mathbf{w}_i^t)\|^2 + R(\mathbf{x}_i^{t+1}, \mathbf{c}_i^t) \\ &\quad + \frac{\lambda_p}{2}\|\mathbf{c}_i^{t+1} - \mathbf{c}_i^t\|^2 \\ &\leq (1-\lambda_p)\left(f_i(\mathbf{x}_i^{t+1}) + f_i(\widetilde{Q}_{\mathbf{c}_i^t}(\mathbf{x}_i^{t+1}))\right) + \lambda_p\left(f_i^{KD}(\mathbf{x}_i^{t+1}, \mathbf{w}^t) + f_i^{KD}(\widetilde{Q}_{\mathbf{c}_i^t}(\mathbf{x}_i^{t+1}), \mathbf{w}^t)\right) \\ &\quad + \frac{\lambda_p L_{DQ}^2}{2}\|\mathbf{w}^t - \mathbf{w}_i^t\|^2 + R(\mathbf{x}_i^{t+1}, \mathbf{c}_i^t) + \frac{\lambda_p}{2}\|\mathbf{c}_i^{t+1} - \mathbf{c}_i^t\|^2 \\ &= F_i(\mathbf{x}_i^{t+1}, \mathbf{c}_i^t, \mathbf{w}^t) + \frac{\lambda_p L_{DQ}^2}{2}\|\mathbf{w}^t - \mathbf{w}_i^t\|^2 + \frac{\lambda_p}{2}\|\mathbf{c}_i^{t+1} - \mathbf{c}_i^t\|^2 \end{aligned}$$

Substituting $\eta_2 = \frac{1}{2(\lambda_p(1+L_{DQ_2})+(1-\lambda_p)(G^{(i)}L_{Q_2}+G^{(i)}_{Q_2}Ll_{Q_2}))}$ gives us:

$$F_i(\mathbf{x}_i^{t+1}, \mathbf{c}_i^{t+1}, \mathbf{w}^t) + \frac{\lambda_p(1+L_{DQ_2}) + (1-\lambda_p)(G^{(i)}L_{Q_2} + G^{(i)}_{Q_2}Ll_{Q_2})}{2}\|\mathbf{c}_i^{t+1} - \mathbf{c}_i^t\|^2 \leq F_i(\mathbf{x}_i^{t+1}, \mathbf{c}_i^t, \mathbf{w}^t) + \frac{\lambda_p L_{DQ}^2}{2}\|\mathbf{w}^t - \mathbf{w}_i^t\|^2 \tag{A.24}$$

A.2.5.3 Sufficient Decrease Due to w

Now, we use $(\lambda_p(L_{D_2} + L_{DQ_3}))$-smoothness of $F_i(\mathbf{x}, \mathbf{c}, \mathbf{w})$ with respect to $\mathbf{w}$:

$$F_i(\mathbf{x}_i^{t+1}, \mathbf{c}_i^{t+1}, \mathbf{w}^{t+1}) \leq F_i(\mathbf{x}_i^{t+1}, \mathbf{c}_i^{t+1}, \mathbf{w}^t) + \left\langle \nabla_{\mathbf{w}^t} F_i(\mathbf{x}_i^{t+1}, \mathbf{c}_i^{t+1}, \mathbf{w}^t), \mathbf{w}^{t+1} - \mathbf{w}^t \right\rangle + \frac{\lambda_p(L_{D_2} + L_{DQ_3})}{2}\|\mathbf{w}^{t+1} - \mathbf{w}^t\|^2$$

After some algebraic manipulations (see Appendix A.2.7) we have:

$$\begin{aligned} &F_i(\mathbf{x}_i^{t+1}, \mathbf{c}_i^{t+1}, \mathbf{w}^{t+1}) + (\frac{\eta_3}{2} - \lambda_p(L_{D_2} + L_{DQ_3})\eta_3^2)\left\|\nabla_{\mathbf{w}^t} F_i(\mathbf{x}_i^{t+1}, \mathbf{c}_i^{t+1}, \mathbf{w}^t)\right\|^2 \\ &\quad\leq F_i(\mathbf{x}_i^{t+1}, \mathbf{c}_i^{t+1}, \mathbf{w}^t) \\ &\qquad + (\eta_3 + 2\lambda_p(L_{D_2} + L_{DQ_3})\eta_3^2)\left\|\mathbf{g}^t - \nabla_{\mathbf{w}_i^t} F_i(\mathbf{x}_i^{t+1}, \mathbf{c}_i^{t+1}, \mathbf{w}_i^t)\right\|^2 \\ &\qquad + (\eta_3 + 2\lambda_p(L_{D_2} + L_{DQ_3})\eta_3^2)(\lambda_p(L_{D_2} + L_{DQ_3}))^2\left\|\mathbf{w}_i^t - \mathbf{w}^t\right\|^2 \end{aligned} \tag{A.25}$$

A.2.5.4 Overall Decrease

Let $L_x^{(i)}, L_c^{(i)}$ for any $i \in [n]$ and L_w are defined as follows:

$$L_x^{(i)} = (1-\lambda_p)(L + G^{(i)}L_{Q_1} + G^{(i)}_{Q_1}Ll_{Q_1}) + \lambda_p(2 + L_{D_1} + L_{DQ_1}) \tag{A.26}$$

$$L_c^{(i)} = (1-\lambda_p)(G^{(i)}L_{Q_2} + G^{(i)}_{Q_2}Ll_{Q_2}) + \lambda_p(1 + L_{DQ_2}) \tag{A.27}$$

$$L_w = L_{D_2} + L_{DQ_3}. \tag{A.28}$$

Summing (A.22), (A.24), (A.25) we get the overall decrease property:

$$F_i(\mathbf{x}_i^{t+1}, \mathbf{c}_i^{t+1}, \mathbf{w}^{t+1}) + (\frac{\eta_3}{2} - \lambda_p L_w \eta_3^2)\left\|\nabla_{\mathbf{w}^t} F_i(\mathbf{x}_i^{t+1}, \mathbf{c}_i^{t+1}, \mathbf{w}^t)\right\|^2 + \frac{L_x}{2}\|\mathbf{x}_i^{t+1} - \mathbf{x}_i^t\|^2$$
$$+ \frac{L_c}{2}\|\mathbf{c}_i^{t+1} - \mathbf{c}_i^t\|^2 \leq (\eta_3 + 2\lambda_p L_w \eta_3^2)\left\|\mathbf{g}^t - \nabla_{\mathbf{w}_i^t} F_i(\mathbf{x}_i^{t+1}, \mathbf{c}_i^{t+1}, \mathbf{w}_i^t)\right\|^2$$
$$+ (L_{DQ}^2 + \frac{L_D^2}{2} + \eta_3 \lambda_p L_w^2 + 2\lambda_p^2 L_w^3 \eta_3^2)\lambda_p \|\mathbf{w}_i^t - \mathbf{w}^t\|^2 + F_i(\mathbf{x}_i^t, \mathbf{c}_i^t, \mathbf{w}^t) \quad \text{(A.29)}$$

Let $L_{\min}^{(i)}$ for any $i \in [n]$ and $L_{\min}$ are defined as follows:

$$L_{\min}^{(i)} = \min\{L_x^{(i)}, L_c^{(i)}, (\eta_3 - 2\lambda_p L_w \eta_3^2)\} \quad \text{(A.30)}$$
$$L_{\min} = \min\{L_{\min}^{(i)} : i \in [n]\}. \quad \text{(A.31)}$$

Then,

$$F_i(\mathbf{x}_i^{t+1}, \mathbf{c}_i^{t+1}, \mathbf{w}^{t+1}) + \frac{L_{\min}}{2}\left(\left\|\nabla_{\mathbf{w}^t} F_i(\mathbf{x}_i^{t+1}, \mathbf{c}_i^{t+1}, \mathbf{w}^t)\right\|^2 + \|\mathbf{x}_i^{t+1} - \mathbf{x}_i^t\|^2 + \|\mathbf{c}_i^{t+1} - \mathbf{c}_i^t\|^2\right)$$
$$\leq (\eta_3 + 2\lambda_p L_w \eta_3^2)\left\|\mathbf{g}^t - \nabla_{\mathbf{w}_i^t} F_i(\mathbf{x}_i^{t+1}, \mathbf{c}_i^{t+1}, \mathbf{w}_i^t)\right\|^2$$
$$+ (L_{DQ}^2 + \frac{L_D^2}{2} + \eta_3 \lambda_p L_w^2 + 2\lambda_p^2 L_w^3 \eta_3^2)\lambda_p \|\mathbf{w}_i^t - \mathbf{w}^t\|^2 + F_i(\mathbf{x}_i^t, \mathbf{c}_i^t, \mathbf{w}^t) \quad \text{(A.32)}$$

We have obtained the sufficient decrease property for the alternating steps; now, we need to arrive at the first order stationarity of the gradient of general loss function. To do this we move on with bounding the gradients with respect to each type of variables.

A.2.6 Bound on the Gradient

Now, we will use the first order optimality conditions due to proximal updates and bound the partial gradients with respect to variables $\mathbf{x}$ and $\mathbf{c}$. After obtaining bounds for partial gradients we will bound the overall gradient and use our results from Section A.2.5 to arrive at the final bound.

A.2.6.1 Bound on the Gradient w.r.t. $\mathbf{x}_i$

Taking the derivative inside the minimization problem (A.18) with respect to $\mathbf{x}$ at $\mathbf{x} = \mathbf{x}_i^{t+1}$ and setting it to 0 gives the following optimality condition:

$$(1-\lambda_p)\left(\nabla_{\mathbf{x}_i^t} f_i(\mathbf{x}_i^t) + \nabla_{\mathbf{x}_i^t} f_i(\widetilde{Q}_{\mathbf{c}_i^t}(\mathbf{x}_i^t))\right) + \lambda_p\left(\nabla_{\mathbf{x}_i^t} f_i^{KD}(\mathbf{x}_i^t, \mathbf{w}_i^t) + \nabla_{\mathbf{x}_i^t} f_i^{KD}(\widetilde{Q}_{\mathbf{c}_i^t}(\mathbf{x}_i^t), \mathbf{w}_i^t)\right) + \frac{1}{\eta_1}(\mathbf{x}_i^{t+1} - \mathbf{x}_i^t) + \lambda \nabla_{\mathbf{x}_i^{t+1}} R(\mathbf{x}_i^{t+1}, \mathbf{c}_i^t) = 0 \tag{A.33}$$

Then we have,

$$\begin{aligned}
\left\|\nabla_{\mathbf{x}_i^{t+1}} F_i(\mathbf{x}_i^{t+1}, \mathbf{c}_i^t, \mathbf{w}^t)\right\| &= \Big\|(1-\lambda_p)\left(\nabla_{\mathbf{x}_i^{t+1}} f_i(\mathbf{x}_i^{t+1}) + \nabla_{\mathbf{x}_i^{t+1}} f_i(\widetilde{Q}_{\mathbf{c}_i^t}(\mathbf{x}_i^{t+1}))\right) \\
&\quad + \lambda_p\left(\nabla_{\mathbf{x}_i^{t+1}} f_i^{KD}(\mathbf{x}_i^{t+1}, \mathbf{w}^t) + \nabla_{\mathbf{x}_i^{t+1}} f_i^{KD}(\widetilde{Q}_{\mathbf{c}_i^t}(\mathbf{x}_i^{t+1}), \mathbf{w}_i^t)\right) \\
&\quad + \lambda \nabla_{\mathbf{x}_i^{t+1}} R(\mathbf{x}_i^{t+1}, \mathbf{c}_i^t)\Big\| \\
&\overset{(a)}{=} \Big\|(1-\lambda_p)\left(\nabla_{\mathbf{x}_i^{t+1}} f_i(\mathbf{x}_i^{t+1}) - \nabla_{\mathbf{x}_i^t} f_i(\mathbf{x}_i^t) + \nabla_{\mathbf{x}_i^{t+1}} f_i(\widetilde{Q}_{\mathbf{c}_i^t}(\mathbf{x}_i^{t+1}))\right. \\
&\quad \left. - \nabla_{\mathbf{x}_i^t} f_i(\widetilde{Q}_{\mathbf{c}_i^t}(\mathbf{x}_i^t))\right) - \frac{1}{\eta_1}(\mathbf{x}_i^{t+1} - \mathbf{x}_i^t) \\
&\quad + \lambda_p\left(\nabla_{\mathbf{x}_i^{t+1}} f_i^{KD}(\mathbf{x}_i^{t+1}, \mathbf{w}^t) - \nabla_{\mathbf{x}_i^t} f_i^{KD}(\mathbf{x}_i^t, \mathbf{w}_i^t)\right. \\
&\quad \left. + \nabla_{\mathbf{x}_i^{t+1}} f_i^{KD}(\widetilde{Q}_{\mathbf{c}_i^t}(\mathbf{x}_i^{t+1}), \mathbf{w}^t) - \nabla_{\mathbf{x}_i^t} f_i^{KD}(\widetilde{Q}_{\mathbf{c}_i^t}(\mathbf{x}_i^t), \mathbf{w}_i^t)\right)\Big\| \\
&\overset{(b)}{\leq} \left(\frac{1}{\eta_1} + (1-\lambda_p)(L + G^{(i)} L_{Q_1} + G_{Q_1}^{(i)} L l_{Q_1})\right)\|\mathbf{x}_i^{t+1} - \mathbf{x}_i^t\| \\
&\quad + \lambda_p\|\nabla_{\mathbf{x}^{t+1}} f_i^{KD}(\mathbf{x}_i^{t+1}, \mathbf{w}^t) - \nabla_{\mathbf{x}_i^t} f_i^{KD}(\mathbf{x}_i^t, \mathbf{w}_i^t)\| \\
&\quad + \lambda_p\|\nabla_{\mathbf{x}_i^{t+1}} f_i^{KD}(\widetilde{Q}_{\mathbf{c}_i^t}(\mathbf{x}_i^{t+1}), \mathbf{w}^t) - \nabla_{\mathbf{x}_i^t} f_i^{KD}(\widetilde{Q}_{\mathbf{c}_i^t}(\mathbf{x}_i^t), \mathbf{w}^t)\| \\
&\leq \left(\frac{1}{\eta_1} + (1-\lambda_p)(L + G^{(i)} L_{Q_1} + G_{Q_1}^{(i)} L l_{Q_1})\right)\|\mathbf{x}_i^{t+1} - \mathbf{x}_i^t\| \\
&\quad + \lambda_p\|\nabla f_i^{KD}(\mathbf{x}_i^{t+1}, \mathbf{w}^t) - \nabla f_i^{KD}(\mathbf{x}_i^t, \mathbf{w}_i^t)\|
\end{aligned}$$

$$
\begin{aligned}
&+ \lambda_p \|\nabla f_i^{KD}(\widetilde{Q}_{\mathbf{c}_i^t}(\mathbf{x}_i^{t+1}), \mathbf{w}^t) - \nabla f_i^{KD}(\widetilde{Q}_{\mathbf{c}_i^t}(\mathbf{x}_i^t), \mathbf{w}^t)\| \\
&\overset{(c)}{\leq} \left(\frac{1}{\eta_1} + (1-\lambda_p)(L + G^{(i)} L_{Q_1} + G_{Q_1}^{(i)} L l_{Q_1})\right) \|\mathbf{x}_i^{t+1} - \mathbf{x}_i^t\| \\
&+ \lambda_p (L_D + L_{DQ})(\|\mathbf{x}_i^{t+1} - \mathbf{x}_i^t\| + \|\mathbf{w}_i^t - \mathbf{w}^t\|) \\
&= \left(\frac{1}{\eta_1} + \lambda_p(L_D + L_{DQ}) + (1-\lambda_p)(L + G^{(i)} L_{Q_1} + G_{Q_1}^{(i)} L l_{Q_1})\right) \|\mathbf{x}_i^{t+1} - \mathbf{x}_i^t\| \\
&+ \lambda_p (L_D + L_{DQ}) \|\mathbf{w}_i^t - \mathbf{w}^t\|
\end{aligned}
$$

where (a) is from (A.33) by substituting the value of $\lambda \nabla_{\mathbf{x}_i^{t+1}} R(\mathbf{x}_i^{t+1}, \mathbf{c}_i^t)$, (b) is due to Claim 3 and **A**.1, and (c) is due to **A**.7. This implies:

$$
\begin{aligned}
\left\| \nabla_{\mathbf{x}_i^{t+1}} F_i(\mathbf{x}_i^{t+1}, \mathbf{c}_i^t, \mathbf{w}^t) \right\|^2 &\leq 2(\lambda_p(L_D + L_{DQ}))^2 \|\mathbf{w}_i^t - \mathbf{w}^t\|^2 \\
&+ 2\left(\frac{1}{\eta_1} + \lambda_p(L_D + L_{DQ}) + (1-\lambda_p)(L + G^{(i)} L_{Q_1} + G_{Q_1}^{(i)} L l_{Q_1})\right)^2 \|\mathbf{x}_i^{t+1} - \mathbf{x}_i^t\|^2
\end{aligned}
$$

Substituting $\eta_1 = \frac{1}{2(\lambda_p(2+L_{D_1}+L_{DQ_1})+(1-\lambda_p)(L+G^{(i)}L_{Q_1}+G_{Q_1}^{(i)}Ll_{Q_1}))}$ we have:

$$
\begin{aligned}
\left\| \nabla_{\mathbf{x}_i^{t+1}} F_i(\mathbf{x}_i^{t+1}, \mathbf{c}_i^t, \mathbf{w}^t) \right\|^2 &\leq 2(\lambda_p(L_D + L_{DQ}))^2 \|\mathbf{w}_i^t - \mathbf{w}^t\|^2 \\
&+ 2\Big(\lambda_p(4 + 2L_{D_1} + 2L_{DQ_1} + L_D + L_{DQ}) \\
&+ 3(1-\lambda_p)(L + G^{(i)} L_{Q_1} + G_{Q_1}^{(i)} L l_{Q_1})\Big)^2 \|\mathbf{x}_i^{t+1} - \mathbf{x}_i^t\|^2 \\
&= 18\Big(\frac{\lambda_p}{3}(4 + 2L_{D_1} + 2L_{DQ_1} + L_D + L_{DQ}) \\
&+ (1-\lambda_p)(L + G^{(i)} L_{Q_1} + G_{Q_1}^{(i)} L l_{Q_1})\Big)^2 \|\mathbf{x}_i^{t+1} - \mathbf{x}_i^t\|^2
\end{aligned}
$$

$$+ 2(\lambda_p(L_D + L_{DQ}))^2\|\mathbf{w}_i^t - \mathbf{w}^t\|^2 \tag{A.34}$$

A.2.6.2 Bound on the Gradient w.r.t. $\mathbf{c}_i$

Similarly, taking the derivative inside the minimization problem (A.19) with respect to $\mathbf{c}$ at $\mathbf{c} = \mathbf{c}_i^{t+1}$ and setting it to 0 gives the following optimality condition:

$$(1-\lambda_p)\nabla_{\mathbf{c}_i^t} f_i(\widetilde{Q}_{\mathbf{c}_i^t}(\mathbf{x}_i^{t+1})) + \lambda_p \nabla_{\mathbf{c}_i^t} f_i^{KD}(\widetilde{Q}_{\mathbf{c}_i^t}(\mathbf{x}_i^{t+1}), \mathbf{w}_i^t) + \frac{1}{\eta_2}(\mathbf{c}_i^{t+1} - \mathbf{c}_i^t) + \lambda \nabla_{\mathbf{c}_i^{t+1}} R(\mathbf{x}_i^{t+1}, \mathbf{c}_i^{t+1}) = 0 \tag{A.35}$$

Then we have

$$\begin{aligned}
\left\|\nabla_{\mathbf{c}_i^{t+1}} F_i(\mathbf{x}_i^{t+1}, \mathbf{c}_i^{t+1}, \mathbf{w}^t)\right\| &= \Big\|(1-\lambda_p)\nabla_{\mathbf{c}_i^{t+1}} f_i(\widetilde{Q}_{\mathbf{c}_i^{t+1}}(\mathbf{x}_i^{t+1})) \\
&\quad + \lambda_p \nabla_{\mathbf{c}_i^{t+1}} f_i^{KD}(\widetilde{Q}_{\mathbf{c}_i^{t+1}}(\mathbf{x}_i^{t+1}), \mathbf{w}^t) + \lambda \nabla_{\mathbf{c}_i^{t+1}} R(\mathbf{x}_i^{t+1}, \mathbf{c}_i^{t+1})\Big\| \\
&\overset{(a)}{=} \Big\|(1-\lambda_p)\left(\nabla_{\mathbf{c}_i^{t+1}} f_i(\widetilde{Q}_{\mathbf{c}_i^{t+1}}(\mathbf{x}_i^{t+1})) - \nabla_{\mathbf{c}_i^t} f_i(\widetilde{Q}_{\mathbf{c}_i^t}(\mathbf{x}_i^{t+1}))\right) \\
&\quad + \lambda_p\left(\nabla_{\mathbf{c}_i^{t+1}} f_i^{KD}(\widetilde{Q}_{\mathbf{c}_i^{t+1}}(\mathbf{x}_i^{t+1}), \mathbf{w}^t) - \nabla_{\mathbf{c}_i^t} f_i^{KD}(\widetilde{Q}_{\mathbf{c}_i^t}(\mathbf{x}_i^{t+1}), \mathbf{w}_i^t)\right) \\
&\quad + \frac{1}{\eta_2}(\mathbf{c}_i^t - \mathbf{c}_i^{t+1})\Big\| \\
&\leq (1-\lambda_p)\left\|\nabla_{\mathbf{c}_i^{t+1}} f_i(\widetilde{Q}_{\mathbf{c}_i^{t+1}}(\mathbf{x}_i^{t+1})) - \nabla_{\mathbf{c}_i^t} f_i(\widetilde{Q}_{\mathbf{c}_i^t}(\mathbf{x}_i^{t+1}))\right\| \\
&\quad + \lambda_p\left\|\nabla_{\mathbf{c}_i^{t+1}} f_i^{KD}(\widetilde{Q}_{\mathbf{c}_i^{t+1}}(\mathbf{x}_i^{t+1}), \mathbf{w}^t) - \nabla_{\mathbf{c}_i^t} f_i^{KD}(\widetilde{Q}_{\mathbf{c}_i^t}(\mathbf{x}_i^{t+1}), \mathbf{w}_i^t)\right\| \\
&\quad + \frac{1}{\eta_2}\|\mathbf{c}_i^t - \mathbf{c}_i^{t+1}\| \\
&\leq (1-\lambda_p)\left\|\nabla_{\mathbf{c}_i^{t+1}} f_i(\widetilde{Q}_{\mathbf{c}_i^{t+1}}(\mathbf{x}_i^{t+1})) - \nabla_{\mathbf{c}_i^t} f_i(\widetilde{Q}_{\mathbf{c}_i^t}(\mathbf{x}_i^{t+1}))\right\| \\
&\quad + \lambda_p\left\|\nabla f_i^{KD}(\widetilde{Q}_{\mathbf{c}_i^{t+1}}(\mathbf{x}_i^{t+1}), \mathbf{w}^t) - \nabla f_i^{KD}(\widetilde{Q}_{\mathbf{c}_i^t}(\mathbf{x}_i^{t+1}), \mathbf{w}_i^t)\right\| \\
&\quad + \frac{1}{\eta_2}\|\mathbf{c}_i^t - \mathbf{c}_i^{t+1}\|
\end{aligned}$$

$$\leq \left(\frac{1}{\eta_2} + \lambda_p L_{DQ} + (1-\lambda_p)(G^{(i)} L_{Q_2} + G^{(i)}_{Q_2} L l_{Q_2})\right) \|\mathbf{c}_i^{t+1} - \mathbf{c}_i^t\| + \lambda_p L_{DQ} \|\mathbf{w}^t - \mathbf{w}_i^t\|$$

where in (a) we substituted the value of $\lambda \nabla_{\mathbf{c}_i^{t+1}} R(\mathbf{x}_i^{t+1}, \mathbf{c}_i^{t+1})$ from (A.35) and the last inequality is due to Claim 4 and Assumption **A.7**. As a result we have,

$$\left\|\nabla_{\mathbf{c}_i^{t+1}} F_i(\mathbf{x}_i^{t+1}, \mathbf{c}_i^{t+1}, \mathbf{w}^t)\right\|^2 \leq 2\left(\frac{1}{\eta_2} + \lambda_p L_{DQ} + (1-\lambda_p)(G^{(i)} L_{Q_2} + G^{(i)}_{Q_2} L l_{Q_2})\right)^2 \|\mathbf{c}_i^{t+1} - \mathbf{c}_i^t\|^2 + 2(\lambda_p L_{DQ})^2 \|\mathbf{w}^t - \mathbf{w}_i^t\|^2$$

substituting $\eta_2 = \frac{1}{2(\lambda_p(1+L_{DQ_2})+(1-\lambda_p)(G^{(i)} L_{Q_2} + G^{(i)}_{Q_2} L l_{Q_2}))}$ we have:

$$\left\|\nabla_{\mathbf{c}_i^{t+1}} F_i(\mathbf{x}_i^{t+1}, \mathbf{c}_i^{t+1}, \mathbf{w}^t)\right\|^2 \leq 18\left(\frac{\lambda_p}{3}(2 + 2L_{DQ_2} + L_{DQ}) + (1-\lambda_p)(G^{(i)} L_{Q_2} + G^{(i)}_{Q_2} L l_{Q_2})\right)^2 \|\mathbf{c}_i^{t+1} - \mathbf{c}_i^t\|^2 + 2(\lambda_p L_{DQ})^2 \|\mathbf{w}^t - \mathbf{w}_i^t\|^2 \tag{A.36}$$

A.2.6.3 Overall Bound

Let $\|\mathbf{G}_i^t\|^2 = \left\|[\nabla_{\mathbf{x}_i^{t+1}} F_i(\mathbf{x}_i^{t+1}, \mathbf{c}_i^t, \mathbf{w}^t)^T, \nabla_{\mathbf{c}_i^{t+1}} F_i(\mathbf{x}_i^{t+1}, \mathbf{c}_i^{t+1}, \mathbf{w}^t)^T, \nabla_{\mathbf{w}^t} F_i(\mathbf{x}_i^{t+1}, \mathbf{c}_i^{t+1}, \mathbf{w}^t)^T]^T\right\|^2$. Then,

$$\begin{aligned}
\|\mathbf{G}_i^t\|^2 &= \left\|[\nabla_{\mathbf{x}_i^{t+1}} F_i(\mathbf{x}_i^{t+1}, \mathbf{c}_i^t, \mathbf{w}^t)^T, \nabla_{\mathbf{c}_i^{t+1}} F_i(\mathbf{x}_i^{t+1}, \mathbf{c}_i^{t+1}, \mathbf{w}^t)^T, \nabla_{\mathbf{w}^t} F_i(\mathbf{x}_i^{t+1}, \mathbf{c}_i^{t+1}, \mathbf{w}^t)^T]^T\right\|^2 \\
&= \left\|\nabla_{\mathbf{x}_i^{t+1}} F_i(\mathbf{x}_i^{t+1}, \mathbf{c}_i^t, \mathbf{w}^t)\right\|^2 + \left\|\nabla_{\mathbf{c}_i^{t+1}} F_i(\mathbf{x}_i^{t+1}, \mathbf{c}_i^{t+1}, \mathbf{w}^t)\right\|^2 + \left\|\nabla_{\mathbf{w}^t} F_i(\mathbf{x}_i^{t+1}, \mathbf{c}_i^{t+1}, \mathbf{w}^t)\right\|^2 \\
&\leq 18\left(\frac{\lambda_p}{3}(4 + 2L_{D_1} + 2L_{DQ_1} + L_D + L_{DQ}) + (1-\lambda_p)(L + G^{(i)} L_{Q_1} + G^{(i)}_{Q_1} L l_{Q_1})\right)^2 \|\mathbf{x}_i^{t+1} - \mathbf{x}_i^t\|^2
\end{aligned}$$

$$+ 18\left(\frac{\lambda_p}{3}(2+2L_{DQ_2}+L_{DQ})+(1-\lambda_p)(G^{(i)}L_{Q_2}+G^{(i)}_{Q_2}Ll_{Q_2})\right)^2\|\mathbf{c}_i^{t+1}-\mathbf{c}_i^t\|^2$$
$$+ 2(\lambda_p(L_D+2L_{DQ}))^2\|\mathbf{w}_i^t-\mathbf{w}^t\|^2+\left\|\nabla_{\mathbf{w}^t}F_i(\mathbf{x}_i^{t+1},\mathbf{c}_i^{t+1},\mathbf{w}^t)\right\|^2$$

where the last inequality is due to (A.34), (A.36) and using that $2(\lambda_p L_{DQ})^2+2(\lambda_p(L_D+L_{DQ}))^2\leq 2(\lambda_p(L_D+2L_{DQ}))^2$. Let

$$L^{(i)}_{\max}=\max\left\{\sqrt{\frac{1}{18}},\left(\frac{\lambda_p}{3}(2+2L_{DQ_2}+L_{DQ})+(1-\lambda_p)(G^{(i)}L_{Q_2}+G^{(i)}_{Q_2}Ll_{Q_2})\right),\right.$$
$$\left.\left(\frac{\lambda_p}{3}(4+2L_{D_1}+2L_{DQ_1}+L_D+L_{DQ})+(1-\lambda_p)(L+G^{(i)}L_{Q_1}+G^{(i)}_{Q_1}Ll_{Q_1})\right)\right\} \tag{A.37}$$

Then,

$$\left\|[\nabla_{\mathbf{x}_i^{t+1}}F_i(\mathbf{x}_i^{t+1},\mathbf{c}_i^t,\mathbf{w}^t)^T,\nabla_{\mathbf{c}_i^{t+1}}F_i(\mathbf{x}_i^{t+1},\mathbf{c}_i^{t+1},\mathbf{w}^t)^T,\nabla_{\mathbf{w}^t}F_i(\mathbf{x}_i^{t+1},\mathbf{c}_i^{t+1},\mathbf{w}^t)^T]^T\right\|^2$$
$$\leq 18(L^{(i)}_{\max})^2(\|\mathbf{x}_i^{t+1}-\mathbf{x}_i^t\|^2+\|\mathbf{c}_i^{t+1}-\mathbf{c}_i^t\|^2+\left\|\nabla_{\mathbf{w}^t}F_i(\mathbf{x}_i^{t+1},\mathbf{c}_i^{t+1},\mathbf{w}^t)\right\|^2)$$
$$+ 2(\lambda_p(L_D+2L_{DQ}))^2\|\mathbf{w}_i^t-\mathbf{w}^t\|^2$$
$$\overset{(a)}{\leq} 36\frac{(L^{(i)}_{\max})^2}{L_{\min}}\left[(\eta_3+2\lambda_pL_w\eta_3^2)\left\|\mathbf{g}^t-\nabla_{\mathbf{w}_i^t}F_i(\mathbf{x}_i^{t+1},\mathbf{c}_i^{t+1},\mathbf{w}_i^t)\right\|^2\right.$$
$$+ (L_{DQ}^2+\frac{L_D^2}{2}+\eta_3\lambda_pL_w^2+2\lambda_p^2L_w^3\eta_3^2)\lambda_p\|\mathbf{w}_i^t-\mathbf{w}^t\|^2$$
$$\left.+ F_i(\mathbf{x}_i^t,\mathbf{c}_i^t,\mathbf{w}^t)-F_i(\mathbf{x}_i^{t+1},\mathbf{c}_i^{t+1},\mathbf{w}^{t+1})\right]+2(\lambda_p(L_D+2L_{DQ}))^2\|\mathbf{w}_i^t-\mathbf{w}^t\|^2, \tag{A.38}$$

where in (a) we use the bound from (A.32), and $L_{\min}$ is defined in (A.31).

Now we state a useful lemma that bounds the average deviation between the local versions of the global model at all clients, and the global model itself. See Appendix A.2.7 for a proof.

Lemma 6. *Let η_3 be chosen such that $\eta_3\leq\sqrt{\frac{1}{6\tau^2(\lambda_pL_w)^2\left(1+\frac{\bar{L}^2_{\max}}{(L^{(\min)}_{\max})^2}\right)}}$ where $L^{(\min)}_{\max}=\min\{L^{(i)}_{\max}:$*

$i \in [n]\}$ *and* $\overline{L}_{\max} = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(L_{\max}^{(i)})^2}$ *(where* $L_{\max}^{(i)}$ *is defined in* (A.37)*), then we have,*

$$\frac{1}{T}\sum_{t=0}^{T-1}\frac{1}{n}\sum_{i=1}^{n}(L_{\max}^{(i)})^2\|\mathbf{w}^t - \mathbf{w}_i^t\|^2 \leq \frac{1}{T}\sum_{t=0}^{T-1}\gamma_t \leq 6\tau^2\eta_3^2\frac{1}{n}\sum_{i=1}^{n}(L_{\max}^{(i)})^2\kappa_i$$

As a corollary:

Corollary 3. *Recall,* $\mathbf{g}^t = \frac{1}{n}\sum_{i=1}^{n}\nabla_{\mathbf{w}^t}F_i(\mathbf{x}_i^{t+1}, \mathbf{c}_i^{t+1}, \mathbf{w}_i^t)$*. Then, we have:*

$$\frac{1}{T}\sum_{t=0}^{T-1}\frac{1}{n}\sum_{i=1}^{n}(L_{\max}^{(i)})^2\left\|\mathbf{g}^t - \nabla_{\mathbf{w}_i^t}F_i(\mathbf{x}_i^{t+1}, \mathbf{c}_i^{t+1}, \mathbf{w}_i^t)\right\|^2 \leq 3\frac{1}{n}\sum_{i=1}^{n}(L_{\max}^{(i)})^2\kappa_i$$
$$+ 3(\lambda_p L_w)^2\left(1 + \frac{\overline{L}_{\max}^2}{(L_{\max}^{(\min)})^2}\right)6\tau^2\eta_3^2\frac{1}{n}\sum_{i=1}^{n}(L_{\max}^{(i)})^2\kappa_i,$$

where $L_{\max}^{(i)}$ *is defined in* (A.37)*, and* $\overline{L}_{\max}, L_{\max}^{(\min)}$ *are defined in Lemma 6.*

Let $\overline{\kappa} := \frac{1}{n}\sum_{i=1}^{n}(L_{\max}^{(i)})^2\kappa_i$ and $C_L := 1 + \frac{\frac{1}{n}\sum_{i=1}^{n}(L_{\max}^{(i)})^2}{(\min_i\{L_{\max}^{(i)}\})^2}$, using Lemma 6 and Corollary 3, summing the bound in (A.38) over time and clients, dividing by T and n:

$$\begin{aligned}
\frac{1}{T}\sum_{t=0}^{T-1}\frac{1}{n}\sum_{i=1}^{n}\|\mathbf{G}_i^t\|^2 \leq \frac{36}{L_{\min}}\Bigg[&(\eta_3 + 2\lambda_p L_w\eta_3^2)\times\left(3\overline{\kappa} + 3(\lambda_p L_w)^2 C_L 6\tau^2\eta_3^2\overline{\kappa}\right)\\
&+ (L_{DQ}^2 + \frac{L_D^2}{2} + \eta_3\lambda_p L_w^2 + 2\lambda_p^2 L_w^3\eta_3^2)\lambda_p \times 6\tau^2\eta_3^2\overline{\kappa}\\
&+ \frac{1}{T}\sum_{t=0}^{T-1}\frac{1}{n}\sum_{i=1}^{n}(L_{\max}^{(i)})^2\left(F_i(\mathbf{x}_i^t, \mathbf{c}_i^t, \mathbf{w}^t) - F_i(\mathbf{x}_i^{t+1}, \mathbf{c}_i^{t+1}, \mathbf{w}^{t+1})\right)\Bigg]\\
&+ 2(\lambda_p(L_D + 2L_{DQ}))^2 \times 6\tau^2\eta_3^2\overline{\kappa}\\
= \frac{36}{L_{\min}}\Bigg[&3\tau^2\eta_3^2\overline{\kappa}(2\lambda_p L_{DQ}^2 + \lambda_p L_D^2 + \eta_3\lambda_p^2 L_w^2(2 + 6C_L) + \eta_3^2\lambda_p^3 L_w^3(4 + 12C_L))\\
&+ 3\eta_3\overline{\kappa} + 6\lambda_p L_w\eta_3^2\overline{\kappa}\Bigg] + \frac{36}{L_{\min}}\frac{1}{n}\sum_{i=1}^{n}\frac{(L_{\max}^{(i)})^2\Delta_F^{(i)}}{T}\\
&+ 12\lambda_p^2(L_D + 2L_{DQ})^2\tau^2\eta_3^2\overline{\kappa} \qquad \text{(A.39)}
\end{aligned}$$

where $\Delta_F^{(i)} = F_i(\mathbf{x}_i^0, \mathbf{c}_i^0, \mathbf{w}_i^0) - F_i(\mathbf{x}_i^T, \mathbf{c}_i^T, \mathbf{w}_i^T)$.

Choice of η_3. Note that in Lemma 6 we chose η_3 such that $\eta_3 \leq \sqrt{\frac{1}{6\tau^2\lambda_p^2 L_w^2 C_L}}$. Now, we further introduce upper bounds on η_3.

- We can choose η_3 small enough so that $L_{\min} = \eta_3 - 2\lambda_p L_w \eta_3^2$; see the definition of $L_{\min}$ in (A.31).
- We can choose η_3 small enough so that $\eta_3 - 2\lambda_p L_w \eta_3^2 \geq \frac{\eta_3}{2}$. This is equivalent to choosing $\eta_3 \leq \frac{1}{4\lambda_p L_w}$.

These two choices imply $L_{\min} \geq \frac{\eta_3}{2}$.

In the end, we have 2 critical constraints on $\eta_3, \{\eta_3 : \eta_3 \leq \sqrt{\frac{1}{6\tau^2\lambda_p^2 L_w^2 C_L}}, \eta_3 \leq \frac{1}{4\lambda_p L_w}\}$. Then, let $\{\eta_3 : \eta_3 \leq \frac{1}{4\lambda_p L_w \tau\sqrt{C_L}}\}$. Moreover, choosing $\tau \leq \sqrt{T}$ we can take $\eta_3 = \frac{1}{4\lambda_p L_w \sqrt{C_L}\sqrt{T}}$ this choice clearly satisfies the above constraints.

From (A.39) we have,

$$\begin{aligned}
\frac{1}{T}\sum_{t=0}^{T-1}\frac{1}{n}\sum_{i=1}^{n}\left\|\mathbf{G}_i^t\right\|^2 &\overset{(a)}{\leq} \frac{72}{\eta_3}\Big[3\tau^2\eta_3^2\overline{\kappa}(2\lambda_p L_{DQ}^2 + \lambda_p L_D^2 + \eta_3\lambda_p^2 L_w^2(2+6C_L) + \eta_3^2\lambda_p^3 L_w^3(4+12C_L)) \\
&\quad + 3\eta_3\overline{\kappa} + 6\lambda_p L_w\eta_3^2\overline{\kappa}\Big] + \frac{72}{\eta_3}\frac{1}{n}\sum_{i=1}^{n}\frac{(L_{\max}^{(i)})^2\Delta_F^{(i)}}{T} \\
&\quad + 12\lambda_p^2(L_D + 2L_{DQ})^2\tau^2\eta_3^2\overline{\kappa} \\
&= 72\Big[3\tau^2\eta_3\overline{\kappa}(2\lambda_p L_{DQ}^2 + \lambda_p L_D^2 + \eta_3\lambda_p^2 L_w^2(2+6C_L) + \eta_3^2\lambda_p^3 L_w^3(4+12C_L)) \\
&\quad + 3\overline{\kappa} + 6\lambda_p L_w\eta_3\overline{\kappa}\Big] + \frac{72}{\eta_3}\frac{1}{n}\sum_{i=1}^{n}\frac{(L_{\max}^{(i)})^2\Delta_F^{(i)}}{T} \\
&\quad + 12\lambda_p^2(L_D + 2L_{DQ})^2\tau^2\eta_3^2\overline{\kappa}
\end{aligned}$$

In (a) we used $L_{\min} \geq \frac{\eta_3}{2}$. Now, we plug in $\eta_3 = \frac{1}{4\lambda_p L_w\sqrt{C_L}\sqrt{T}}$ then:

$$\frac{1}{T}\sum_{t=0}^{T-1}\frac{1}{n}\sum_{i=1}^{n}\left\|\mathbf{G}_i^t\right\|^2 \leq 72\left[\frac{3}{4}\tau^2\overline{\kappa}\frac{L_D^2 + 2L_{DQ}^2}{\sqrt{C_L}L_w}\frac{1}{\sqrt{T}} + \frac{3(2+6C_L)}{16C_L}\tau^2\overline{\kappa}\frac{1}{T} + \frac{3(4+12C_L)}{64C_L^{\frac{3}{2}}}\tau^2\overline{\kappa}\frac{1}{T^{\frac{3}{2}}}\right.$$

$$
\begin{aligned}
&\quad + 3\overline{\kappa} + \frac{3}{2}\frac{\overline{\kappa}}{\sqrt{C_L}\sqrt{T}}\Bigg] + 288\lambda_p L_w\sqrt{C_L}\frac{1}{n}\sum_{i=1}^{n}\frac{(L_{\max}^{(i)})^2\Delta_F^{(i)}}{\sqrt{T}} \\
&\quad + \frac{3}{4}\tau^2\overline{\kappa}\frac{(L_D+2L_{DQ})^2}{C_L L_w^2}\frac{1}{T} \\
&= \frac{54\frac{L_D^2+2L_{DQ}^2}{\sqrt{C_L}L_w}\tau^2\overline{\kappa} + \frac{108\overline{\kappa}}{\sqrt{C_L}} + 288\sqrt{C_L}\lambda_p L_w\frac{1}{n}\sum_{i=1}^{n}(L_{\max}^{(i)})^2\Delta_F^{(i)}}{\sqrt{T}} \\
&\quad + \frac{\frac{27}{2}\frac{2+C_L}{C_L}\tau^2\overline{\kappa} + \frac{3(L_D+2L_{DQ})^2}{4C_L L_w^2}\tau^2\overline{\kappa}}{T} + \frac{\frac{27}{2}\frac{2+C_L}{C_L^{\frac{3}{2}}}\tau^2\overline{\kappa}}{T^{\frac{3}{2}}} + 216\overline{\kappa}
\end{aligned}
$$

A.2.7 Omitted Details in Proof of Theorem 2

Obtaining (A.25),

$$
\begin{aligned}
F_i(\mathbf{x}_i^{t+1},\mathbf{c}_i^{t+1},\mathbf{w}^{t+1}) &\le F_i(\mathbf{x}_i^{t+1},\mathbf{c}_i^{t+1},\mathbf{w}^t) + \left\langle \nabla_{\mathbf{w}^t}F_i(\mathbf{x}_i^{t+1},\mathbf{c}_i^{t+1},\mathbf{w}^t), \mathbf{w}^{t+1}-\mathbf{w}^t\right\rangle \\
&\quad + \frac{\lambda_p(L_{D_2}+L_{DQ_3})}{2}\|\mathbf{w}^{t+1}-\mathbf{w}^t\|^2 \\
&= F_i(\mathbf{x}_i^{t+1},\mathbf{c}_i^{t+1},\mathbf{w}^t) - \left\langle \nabla_{\mathbf{w}^t}F_i(\mathbf{x}_i^{t+1},\mathbf{c}_i^{t+1},\mathbf{w}^t), \eta_3\mathbf{g}^t\right\rangle + \frac{\lambda_p(L_{D_2}+L_{DQ_3})}{2}\|\eta_3\mathbf{g}^t\|^2 \\
&= F_i(\mathbf{x}_i^{t+1},\mathbf{c}_i^{t+1},\mathbf{w}^t) \\
&\quad - \eta_3\left\langle \nabla_{\mathbf{w}^t}F_i(\mathbf{x}_i^{t+1},\mathbf{c}_i^{t+1},\mathbf{w}^t), \mathbf{g}^t - \nabla_{\mathbf{w}^t}F_i(\mathbf{x}_i^{t+1},\mathbf{c}_i^{t+1},\mathbf{w}^t) + \nabla_{\mathbf{w}^t}F_i(\mathbf{x}_i^{t+1},\mathbf{c}_i^{t+1},\mathbf{w}^t)\right\rangle \\
&\quad + \frac{\lambda_p(L_{D_2}+L_{DQ_3})}{2}\eta_3^2\|\mathbf{g}^t - \nabla_{\mathbf{w}^t}F_i(\mathbf{x}_i^{t+1},\mathbf{c}_i^{t+1},\mathbf{w}^t) + \nabla_{\mathbf{w}^t}F_i(\mathbf{x}_i^{t+1},\mathbf{c}_i^{t+1},\mathbf{w}^t)\|^2 \\
&= F_i(\mathbf{x}_i^{t+1},\mathbf{c}_i^{t+1},\mathbf{w}^t) - \eta_3\left\|\nabla_{\mathbf{w}^t}F_i(\mathbf{x}_i^{t+1},\mathbf{c}_i^{t+1},\mathbf{w}^t)\right\|^2 \\
&\quad - \eta_3\left\langle \nabla_{\mathbf{w}^t}F_i(\mathbf{x}_i^{t+1},\mathbf{c}_i^{t+1},\mathbf{w}^t), \mathbf{g}^t - \nabla_{\mathbf{w}^t}F_i(\mathbf{x}_i^{t+1},\mathbf{c}_i^{t+1},\mathbf{w}^t)\right\rangle \\
&\quad + \frac{\lambda_p(L_{D_2}+L_{DQ_3})}{2}\eta_3^2\|\mathbf{g}^t - \nabla_{\mathbf{w}^t}F_i(\mathbf{x}_i^{t+1},\mathbf{c}_i^{t+1},\mathbf{w}^t) + \nabla_{\mathbf{w}^t}F_i(\mathbf{x}_i^{t+1},\mathbf{c}_i^{t+1},\mathbf{w}^t)\|^2 \\
&\le F_i(\mathbf{x}_i^{t+1},\mathbf{c}_i^{t+1},\mathbf{w}^t) - \eta_3\left\|\nabla_{\mathbf{w}^t}F_i(\mathbf{x}_i^{t+1},\mathbf{c}_i^{t+1},\mathbf{w}^t)\right\|^2 + \frac{\eta_3}{2}\left\|\nabla_{\mathbf{w}^t}F_i(\mathbf{x}_i^{t+1},\mathbf{c}_i^{t+1},\mathbf{w}^t)\right\|^2 \\
&\quad + \frac{\eta_3}{2}\left\|\mathbf{g}^t - \nabla_{\mathbf{w}^t}F_i(\mathbf{x}_i^{t+1},\mathbf{c}_i^{t+1},\mathbf{w}^t)\right\|^2 + \lambda_p(L_{D_2}+L_{DQ_3})\eta_3^2\left\|\mathbf{g}^t - \nabla_{\mathbf{w}^t}F_i(\mathbf{x}_i^{t+1},\mathbf{c}_i^{t+1},\mathbf{w}^t)\right\|^2
\end{aligned}
$$

$$
\begin{aligned}
&+ \lambda_p(L_{D_2} + L_{DQ_3})\eta_3^2 \left\| \nabla_{\mathbf{w}^t} F_i(\mathbf{x}_i^{t+1}, \mathbf{c}_i^{t+1}, \mathbf{w}^t) \right\|^2 \\
&= F_i(\mathbf{x}_i^{t+1}, \mathbf{c}_i^{t+1}, \mathbf{w}^t) - (\frac{\eta_3}{2} - \lambda_p(L_{D_2} + L_{DQ_3})\eta_3^2) \left\| \nabla_{\mathbf{w}^t} F_i(\mathbf{x}_i^{t+1}, \mathbf{c}_i^{t+1}, \mathbf{w}^t) \right\|^2 \\
&\quad + (\frac{\eta_3}{2} + \lambda_p(L_{D_2} + L_{DQ_3})\eta_3^2) \left\| \mathbf{g}^t - \nabla_{\mathbf{w}_i^t} F_i(\mathbf{x}_i^{t+1}, \mathbf{c}_i^{t+1}, \mathbf{w}_i^t) + \nabla_{\mathbf{w}_i^t} F_i(\mathbf{x}_i^{t+1}, \mathbf{c}_i^{t+1}, \mathbf{w}_i^t) \right. \\
&\quad \left. - \nabla_{\mathbf{w}^t} F_i(\mathbf{x}_i^{t+1}, \mathbf{c}_i^{t+1}, \mathbf{w}^t) \right\|^2 \\
&\leq F_i(\mathbf{x}_i^{t+1}, \mathbf{c}_i^{t+1}, \mathbf{w}^t) - (\frac{\eta_3}{2} - \lambda_p(L_{D_2} + L_{DQ_3})\eta_3^2) \left\| \nabla_{\mathbf{w}^t} F_i(\mathbf{x}_i^{t+1}, \mathbf{c}_i^{t+1}, \mathbf{w}^t) \right\|^2 \\
&\quad + (\eta_3 + 2\lambda_p(L_{D_2} + L_{DQ_3})\eta_3^2) \left\| \mathbf{g}^t - \nabla_{\mathbf{w}_i^t} F_i(\mathbf{x}_i^{t+1}, \mathbf{c}_i^{t+1}, \mathbf{w}_i^t) \right\|^2 \\
&\quad + (\eta_3 + 2\lambda_p(L_{D_2} + L_{DQ_3})\eta_3^2) \left\| \nabla_{\mathbf{w}_i^t} F_i(\mathbf{x}_i^{t+1}, \mathbf{c}_i^{t+1}, \mathbf{w}_i^t) - \nabla_{\mathbf{w}^t} F_i(\mathbf{x}_i^{t+1}, \mathbf{c}_i^{t+1}, \mathbf{w}^t) \right\|^2 \\
&\leq F_i(\mathbf{x}_i^{t+1}, \mathbf{c}_i^{t+1}, \mathbf{w}^t) - (\frac{\eta_3}{2} - \lambda_p(L_{D_2} + L_{DQ_3})\eta_3^2) \left\| \nabla_{\mathbf{w}^t} F_i(\mathbf{x}_i^{t+1}, \mathbf{c}_i^{t+1}, \mathbf{w}^t) \right\|^2 \\
&\quad + (\eta_3 + 2\lambda_p(L_{D_2} + L_{DQ_3})\eta_3^2) \left\| \mathbf{g}^t - \nabla_{\mathbf{w}_i^t} F_i(\mathbf{x}_i^{t+1}, \mathbf{c}_i^{t+1}, \mathbf{w}_i^t) \right\|^2 \\
&\quad + (\eta_3 + 2\lambda_p(L_{D_2} + L_{DQ_3})\eta_3^2)(\lambda_p(L_{D_2} + L_{DQ_3}))^2 \left\| \mathbf{w}_i^t - \mathbf{w}^t \right\|^2 .
\end{aligned}
$$

Rearranging the terms gives (A.25),

Proof Lemma 6. Let t_c be the latest synchronization time before t. Define $\gamma_t = \frac{1}{n}\sum_{i=1}^n (L_{\max}^{(i)})^2 \|\mathbf{w}^t - \mathbf{w}_i^t\|^2$. Then:

$$
\begin{aligned}
\gamma_t = \frac{1}{n}\sum_{i=1}^n (L_{\max}^{(i)})^2 \Bigg\| & \mathbf{w}^{t_c} - \frac{\eta_3}{n}\sum_{j=t_c}^{t}\sum_{k=1}^{n} \nabla_{\mathbf{w}_k^j} F_k(\mathbf{x}_k^{j+1}, \mathbf{c}_k^{j+1}, \mathbf{w}_k^j) \\
& - (\mathbf{w}^{t_c} - \eta_3 \sum_{j=t_c}^{t} \nabla_{\mathbf{w}_i^j} F_i(\mathbf{x}_i^{j+1}, \mathbf{c}_i^{j+1}, \mathbf{w}_i^j) \Bigg\|^2
\end{aligned}
$$

$$
\begin{aligned}
&\overset{\text{(a)}}{\leq} \tau \sum_{j=t_c}^{t} \frac{\eta_3^2}{n} \sum_{i=1}^{n} (L_{\max}^{(i)})^2 \left\| \frac{1}{n} \sum_{k=1}^{n} \nabla_{\mathbf{w}_k^j} F_k(\mathbf{x}_k^{j+1}, \mathbf{c}_k^{j+1}, \mathbf{w}_k^j) - \nabla_{\mathbf{w}_i^j} F_i(\mathbf{x}_i^{j+1}, \mathbf{c}_i^{j+1}, \mathbf{w}_i^j) \right\|^2 \qquad (\star 1) \\
&= \tau \sum_{j=t_c}^{t} \frac{\eta_3^2}{n} \sum_{i=1}^{n} (L_{\max}^{(i)})^2 \left[\left\| \frac{1}{n} \sum_{k=1}^{n} \left(\nabla_{\mathbf{w}_k^j} F_k(\mathbf{x}_k^{j+1}, \mathbf{c}_k^{j+1}, \mathbf{w}_k^j) - \nabla_{\mathbf{w}^j} F_k(\mathbf{x}_k^{j+1}, \mathbf{c}_k^{j+1}, \mathbf{w}^j) \right.\right.\right. \\
&\qquad \left.\left.\left. + \nabla_{\mathbf{w}^j} F_k(\mathbf{x}_k^{j+1}, \mathbf{c}_k^{j+1}, \mathbf{w}^j) \right) - \nabla_{\mathbf{w}^j} F_i(\mathbf{x}_i^{j+1}, \mathbf{c}_i^{j+1}, \mathbf{w}^j) + \nabla_{\mathbf{w}^j} F_i(\mathbf{x}_i^{j+1}, \mathbf{c}_i^{j+1}, \mathbf{w}^j) \right.\right. \\
&\qquad \left.\left. - \nabla_{\mathbf{w}_i^j} F_i(\mathbf{x}_i^{j+1}, \mathbf{c}_i^{j+1}, \mathbf{w}_i^j \right\|^2 \right] \\
&\leq \tau \sum_{j=t_c}^{t_c+\tau} 3\frac{\eta_3^2}{n} \sum_{i=1}^{n} (L_{\max}^{(i)})^2 \left[\left\| \frac{1}{n} \sum_{k=1}^{n} \left(\nabla_{\mathbf{w}_k^j} F_k(\mathbf{x}_k^{j+1}, \mathbf{c}_k^{j+1}, \mathbf{w}_k^j) - \nabla_{\mathbf{w}^j} F_k(\mathbf{x}_k^{j+1}, \mathbf{c}_k^{j+1}, \mathbf{w}^j) \right) \right\|^2 \right. \\
&\qquad + \left\| \frac{1}{n} \sum_{k=1}^{n} \nabla_{\mathbf{w}^j} F_k(\mathbf{x}_k^{j+1}, \mathbf{c}_k^{j+1}, \mathbf{w}^j) - \nabla_{\mathbf{w}^j} F_i(\mathbf{x}_i^{j+1}, \mathbf{c}_i^{j+1}, \mathbf{w}^j) \right\|^2 \\
&\qquad \left. + \left\| \nabla_{\mathbf{w}^j} F_i(\mathbf{x}_i^{j+1}, \mathbf{c}_i^{j+1}, \mathbf{w}^j) - \nabla_{\mathbf{w}_i^j} F_i(\mathbf{x}_i^{j+1}, \mathbf{c}_i^{j+1}, \mathbf{w}_i^j \right\|^2 \right] \\
&\leq \tau \sum_{j=t_c}^{t_c+\tau} 3\eta_3^2 \left[\frac{(\lambda_p L_w)^2 n}{n^2} \sum_{k=1}^{n} \frac{1}{n} \sum_{i=1}^{n} (L_{\max}^{(i)})^2 \|\mathbf{w}_k^j - \mathbf{w}^j\|^2 + \frac{1}{n} \sum_{i=1}^{n} (L_{\max}^{(i)})^2 \kappa_i \right. \\
&\qquad \left. + (\lambda_p L_w)^2 \frac{1}{n} \sum_{i=1}^{n} (L_{\max}^{(i)})^2 \|\mathbf{w}^j - \mathbf{w}_i^j\|^2 \right] \\
&\leq \tau \sum_{j=t_c}^{t_c+\tau} 3\eta_3^2 \left[\frac{(\lambda_p L_w)^2 n}{n^2} \frac{1}{n} \sum_{i=1}^{n} (L_{\max}^{(i)})^2 \frac{1}{(L_{\max}^{(\min)})^2} \sum_{k=1}^{n} (L_{\max}^{(k)})^2 \|\mathbf{w}_k^j - \mathbf{w}^j\|^2 \right. \\
&\qquad \left. + \frac{1}{n} \sum_{i=1}^{n} (L_{\max}^{(i)})^2 \kappa_i + (\lambda_p L_w)^2 \gamma_j \|^2 \right] \\
&= \tau \sum_{j=t_c}^{t_c+\tau} 3\eta_3^2 \left((\lambda_p L_w)^2 \frac{1}{n} \sum_{i=1}^{n} (L_{\max}^{(i)})^2 \frac{1}{(L_{\max}^{(\min)})^2} \gamma_j + \frac{1}{n} \sum_{i=1}^{n} (L_{\max}^{(i)})^2 \kappa_i + (\lambda_p L_w)^2 \gamma_j \right) \\
&= \tau \sum_{j=t_c}^{t_c+\tau} 3\eta_3^2 \left((\lambda_p L_w)^2 \frac{\overline{L}_{\max}^2}{(L_{\max}^{(\min)})^2} \gamma_j + \frac{1}{n} \sum_{i=1}^{n} (L_{\max}^{(i)})^2 \kappa_i + (\lambda_p L_w)^2 \gamma_j \right) \qquad (\star 2)
\end{aligned}
$$

in (a) we use the facts that $\| \sum_{i=1}^{K} a_i \|^2 \leq K \sum_{i=1}^{K} \|a_i\|^2$, $t \leq \tau + t_c$ and that we are summing

over non-negative terms. As a result, we have:

$$\begin{aligned}
\gamma_t &\leq \tau \sum_{j=t_c}^{t_c+\tau} 3\eta_3^2 \left((\lambda_p L_w)^2 \left(1 + \frac{\overline{L}_{\max}^2}{(L_{\max}^{(\min)})^2} \right) \gamma_j + \frac{1}{n} \sum_{i=1}^{n} (L_{\max}^{(i)})^2 \kappa_i \right) \\
\implies \sum_{t=t_c}^{t_c+\tau} \gamma_t &\leq \sum_{t=t_c}^{t_c+\tau} \sum_{j=t_c}^{t_c+\tau} 3\tau\eta_3^2 \left((\lambda_p L_w)^2 \left(1 + \frac{\overline{L}_{\max}^2}{(L_{\max}^{(\min)})^2} \right) \gamma_j + \frac{1}{n} \sum_{i=1}^{n} (L_{\max}^{(i)})^2 \kappa_i \right) \\
&= 3\tau^2\eta_3^2 (\lambda_p L_w)^2 \left(1 + \frac{\overline{L}_{\max}^2}{(L_{\max}^{(\min)})^2} \right) \sum_{j=t_c}^{t_c+\tau} \gamma_j + 3\tau^3\eta_3^2 \frac{1}{n} \sum_{i=1}^{n} (L_{\max}^{(i)})^2 \kappa_i
\end{aligned}$$

Let us choose η_3 such that $3\tau^2\eta_3^2(\lambda_p L_w)^2 \left(1 + \frac{\overline{L}_{\max}^2}{(L_{\max}^{(\min)})^2}\right) \leq \frac{1}{2} \Leftrightarrow \eta_3 \leq \sqrt{\frac{1}{6\tau^2(\lambda_p L_w)^2 \left(1 + \frac{\overline{L}_{\max}^2}{(L_{\max}^{(\min)})^2}\right)}}$, sum over all syncronization times, and divide both sides by T:

$$\begin{aligned}
\frac{1}{T} \sum_{t=0}^{T-1} \gamma_t &\leq \frac{1}{2} \sum_{j=0}^{T-1} \gamma_j + 3\tau^2\eta_3^2 \frac{1}{n} \sum_{i=1}^{n} (L_{\max}^{(i)})^2 \kappa_i \\
\implies \frac{1}{T} \sum_{t=0}^{T-1} \gamma_t &\leq 6\tau^2\eta_3^2 \frac{1}{n} \sum_{i=1}^{n} (L_{\max}^{(i)})^2 \kappa_i
\end{aligned}$$

□

Proof of Corollary 3. From (⋆1) ≤ (⋆2) in the proof of Lemma 6, we have:

$$\begin{aligned}
&\sum_{t=t_c}^{t_c+\tau} \frac{1}{n} \sum_{i=1}^{n} (L_{\max}^{(i)})^2 \left\| \mathbf{g}^t - \nabla_{\mathbf{w}_i^t} F_i(\mathbf{x}_i^{t+1}, \mathbf{c}_i^{t+1}, \mathbf{w}_i^t) \right\|^2 \\
&\qquad \leq \sum_{t=t_c}^{t_c+\tau} 3 \left((\lambda_p L_w)^2 \left(1 + \frac{\overline{L}_{\max}^2}{(L_{\max}^{(\min)})^2} \right) \gamma_t + \frac{1}{n} \sum_{i=1}^{n} (L_{\max}^{(i)})^2 \kappa_i \right)
\end{aligned}$$

Summing over all t_c and dividing by T:

$$\begin{aligned}
&\frac{1}{T} \sum_{t=0}^{T-1} \frac{1}{n} \sum_{i=1}^{n} (L_{\max}^{(i)})^2 \left\| \mathbf{g}^t - \nabla_{\mathbf{w}_i^t} F_i(\mathbf{x}_i^{t+1}, \mathbf{c}_i^{t+1}, \mathbf{w}_i^t) \right\|^2 \\
&\qquad \leq 3(\lambda_p L_w)^2 \left(1 + \frac{\overline{L}_{\max}^2}{(L_{\max}^{(\min)})^2} \right) \frac{1}{T} \sum_{t=0}^{T-1} \gamma_t + 3 \frac{1}{n} \sum_{i=1}^{n} (L_{\max}^{(i)})^2 \kappa_i \\
&\qquad \overset{\text{(a)}}{\leq} 3(\lambda_p L_w)^2 \left(1 + \frac{\overline{L}_{\max}^2}{(L_{\max}^{(\min)})^2} \right) \times 6\tau^2\eta_3^2 \frac{1}{n} \sum_{i=1}^{n} (L_{\max}^{(i)})^2 \kappa_i + 3 \frac{1}{n} \sum_{i=1}^{n} (L_{\max}^{(i)})^2 \kappa_i,
\end{aligned}$$

where (a) is from Lemma 6. □

A.2.8 Proof Outline with Client Sampling

Incorporating partial client participation and analyzing the resulting algorithm is fairly simple. Essentially only changes are in Lemma 6 and Corollary 3, as everything before that is for local updates only. Now we give a summary of what changes:

Let $\mathcal{K}_t$ denote the set of clients that participates at time t, where $|\mathcal{K}_t| = K$, i.e., K clients participate in the training process at any time. In this case, we define the average parameter $\mathbf{w}^t$ and the gradient $\mathbf{g}^t$ as the average over the respective parameters of only the active clients at time t; we also define γ_t similarly.

- **C**hange in the proof of Lemma 6: In the proof of Lemma 6, the second term on the RHS of the second inequality, with the above modification will be equal to $\left\| \frac{1}{K} \sum_{k \in \mathcal{K}_t} \nabla_{\mathbf{w}^j} F_k(\mathbf{x}_k^{j+1}, \mathbf{c}_k^{j+1}, \mathbf{w}^j) - \nabla_{\mathbf{w}^j} F_i(\mathbf{x}_i^{j+1}, \mathbf{c}_i^{j+1}, \mathbf{w}^j) \right\|^2$. Earlier, the average was over all clients from 1 to n and this term was bounded by κ_i using Assumption A.6. Now, we can use the Jensen's inequality (iteratively) and Assumption A.6 and bound this by $2\kappa_i + \frac{2}{K} \sum_{j \in \mathcal{K}_t} \kappa_j$. This change will propagate over until the end.
- **C**hange in the proof of Corollary 3: Since this is a corollary to Lemma 6, this will also see a similar change.
- **R**emaining convergence proof: Now, continuing the exact same convergence proof and using the modified bounds of Lemma 1 and Corollary 3 will give the bound of our algorithm with partial client participation.

This is the modification in the entire proof.

A.2.9 Problem Setup and Convergence Result of QuPeL

For QuPeL, we define the following augmented loss function at client i:

$$F_i(\mathbf{x}_i, \mathbf{c}_i, \mathbf{w}) := f_i(\mathbf{x}_i) + f_i(\widetilde{Q}_{\mathbf{c}_i}(\mathbf{x}_i)) + \lambda R(\mathbf{x}_i, \mathbf{c}_i) + \frac{\lambda_p}{2}(\|\mathbf{x}_i - \mathbf{w}\|^2 + \|\widetilde{Q}_{\mathbf{c}_i}(\mathbf{x}_i) - \mathbf{w}\|^2). \tag{A.40}$$

Here, $\mathbf{w} \in \mathbb{R}^d$ denotes the global model, $\mathbf{x}_i \in \mathbb{R}^d$ denotes the personalized model at client i, and $\mathbf{c}_i \in \mathbb{R}^{m_i}$ denotes the model quantization centers at client i, where m_i is the number of centers at client i, with $\log m_i$ representing the number of bits per parameter, which could be different for each client – larger the m_i, higher the precision. Note that different than QuPeD, this time all the model parameters $\mathbf{x}_i$ are of the same dimension with $\mathbf{w}$.

From (A.40) we get the following global loss function:

$$\arg\min \left(F\Big(\{\mathbf{x}_i\}, \{\mathbf{c}_i\}, \mathbf{w}\Big) := \frac{1}{n} \sum_{i=1}^{n} F_i(\mathbf{x}_i, \mathbf{c}_i, \mathbf{w}) \right), \tag{A.41}$$

where minimization is taken over $\mathbf{x}_1, \ldots, \mathbf{x}_n \in \mathbb{R}^d$, $\mathbf{c}_i \in \mathbb{R}^{m_i}$ for $i \in [n]$, and $\mathbf{w} \in \mathbb{R}^d$.

We present our proposed algorithm QuPeL for minimizing (A.41) in Algorithm 9.

A.2.10 Convergence Analysis

First we state our convergence result for QuPeL,

Theorem 17. *Consider running Algorithm 9 for T iterations with $\tau \leq \sqrt{T}$, and with learning rates $\eta_1 = \frac{1}{2(3\lambda_p + \lambda_p L_{Q_1} + L + G L_{Q_1} + G_{Q_1} L l_{Q_1})}$, $\eta_2 = \frac{1}{2(\lambda_p + \lambda_p L_{Q_2} + G L_{Q_2} + G_{Q_2} L L_{Q_2})}$, and $\eta_3 = 1/8\lambda_p\sqrt{T}$. For any $t \in [T]$, define*

$$\mathbf{G}_i^t := [\nabla_{\mathbf{x}_i^{t+1}} F_i(\mathbf{x}_i^{t+1}, \mathbf{c}_i^t, \mathbf{w}^t)^T, \nabla_{\mathbf{c}_i^{t+1}} F_i(\mathbf{x}_i^{t+1}, \mathbf{c}_i^{t+1}, \mathbf{w}^t)^T, \nabla_{\mathbf{w}^t} F_i(\mathbf{x}_i^{t+1}, \mathbf{c}_i^{t+1}, \mathbf{w}^t)^T]^T.$$

*Then, under assumptions **A**.1-A.6, we have:*

$$\frac{1}{T} \sum_{t=0}^{T-1} \frac{1}{n} \sum_{i=1}^{n} \left\| \mathbf{G}_i^t \right\|^2 = \mathcal{O}\left(\frac{\Lambda}{\sqrt{T}} + \frac{L_{\max}^2 \tau^2 \kappa + \tau^2 \kappa^2}{T} + \frac{L_{\max}^2 \tau^2 \kappa}{T^{\frac{3}{2}}} + L_{\max}^2 \kappa \right),$$

where $L_{\max} = \max\{\sqrt{\frac{1}{18}}, \frac{2\lambda_p}{3} + \frac{2\lambda_p L_{Q_2}}{3} + G L_{Q_2} + G_{Q_2} L L_{Q_2}, \frac{7}{3}\lambda_p + \frac{2\lambda_p L_{Q_1}}{3} + \frac{\lambda_p G_{Q_1} l_{Q_1}}{3} + L + G l_{Q_1} + G_{Q_1} L l_{Q_1}\}$, and $\Lambda = L_{\max}^2 \tau \kappa (1 + G_{Q_1}^2 + G_{Q_2}^2) + L_{\max}^2 + L_{\max}^2 \lambda_p \Delta_F$ with $\Delta_F = \frac{1}{n} \sum_{i=1}^{n} \big(F_i(\mathbf{x}_i^0, \mathbf{c}_i^0, \mathbf{w}_i^0) - F_i(\mathbf{x}_i^T, \mathbf{c}_i^T, \mathbf{w}_i^T) \big)$.

The rest of this section is devoted to proving Theorem 17.

The analysis is similar to the one in Appendix A.2.4, however, this time we can assume same gradient bounds for each client since all the clients have the same model structure. As a result the analysis requires slightly less algebraic manipulations. Naturally, we don't need **A.7** in this analysis. We defer proofs of the claims and some derivation details to the end of section. We take $\mathbf{w}^t = \frac{1}{n}\sum_{i=1}^n \mathbf{w}_i^t$, so that $\mathbf{w}^t$ is defined for every time point.

Alternating updates. Let us first restate the alternating updates for $\mathbf{x}_i$ and $\mathbf{c}_i$:

$$\begin{aligned}
\mathbf{x}_i^{t+1} &= \operatorname{prox}_{\eta_1\lambda R_{\mathbf{c}_i^t}}(\mathbf{x}_i^t - \eta_1\nabla f_i(\mathbf{x}_i^t) - \eta_1\nabla_{\mathbf{x}_i^t} f_i(\widetilde{Q}_{\mathbf{c}_i^t}(\mathbf{x}_i^t)) - \eta_1\lambda_p(\mathbf{x}_i^t - \mathbf{w}_i^t) \\
&\quad + \nabla_{\mathbf{x}_i^t}\widetilde{Q}_{\mathbf{c}_i^t}(\mathbf{x}_i^t)(\widetilde{Q}_{\mathbf{c}_i^t}(\mathbf{x}_i^t) - \mathbf{w}_i^t)) \\
\mathbf{c}_i^{t+1} &= \operatorname{prox}_{\eta_2\lambda R_{\mathbf{x}_i^{t+1}}}(\mathbf{c}_i^t - \eta_2\nabla_{\mathbf{c}_i^t} f_i(\widetilde{Q}_{\mathbf{c}_i^t}(\mathbf{x}_i^t)) - \eta_2\lambda_p\nabla_{\mathbf{c}_i^{t+1}}\widetilde{Q}_{\mathbf{c}_i^t}(\mathbf{x}_i^{t+1})(\widetilde{Q}_{\mathbf{c}_i^t}(\mathbf{x}_i^{t+1}) - \mathbf{w}_i^t))
\end{aligned}$$

The alternating updates are equivalent to solving the following two optimization problems.

$$\begin{aligned}
\mathbf{x}_i^{t+1} = \arg\min_{\mathbf{x}\in\mathbb{R}^d} \Big\{ &\left\langle \mathbf{x} - \mathbf{x}_i^t, \nabla f_i\left(\mathbf{x}_i^t\right)\right\rangle + \left\langle \mathbf{x} - \mathbf{x}_i^t, \nabla_{\mathbf{x}_i^t} f_i(\widetilde{Q}_{\mathbf{c}_i^t}(\mathbf{x}_i^t))\right\rangle + \left\langle \mathbf{x} - \mathbf{x}_i^t, \lambda_p(\mathbf{x}_i^t - \mathbf{w}_i^t)\right\rangle \\
&+ \left\langle \mathbf{x} - \mathbf{x}_i^t, \lambda_p\nabla_{\mathbf{x}_i^t}\widetilde{Q}_{\mathbf{c}_i^t}(\mathbf{x}_i^t)(\widetilde{Q}_{\mathbf{c}_i^t}(\mathbf{x}_i^t) - \mathbf{w}_i^t)\right\rangle + \frac{1}{2\eta_1}\left\|\mathbf{x} - \mathbf{x}_i^t\right\|_2^2 + \lambda R(\mathbf{x}, \mathbf{c}_i^t)\Big\}
\end{aligned} \tag{A.42}$$

$$\begin{aligned}
\mathbf{c}_i^{t+1} = \arg\min_{\mathbf{c}\in\mathbb{R}^m} \Big\{ &\left\langle \mathbf{c} - \mathbf{c}_i^t, \nabla_{\mathbf{c}_i^t} f_i(\widetilde{Q}_{\mathbf{c}_i^t}(\mathbf{x}_i^{t+1}))\right\rangle + \left\langle \mathbf{c} - \mathbf{c}_i^t, \lambda_p\nabla_{\mathbf{c}_i^{t+1}}\widetilde{Q}_{\mathbf{c}_i^t}(\mathbf{x}_i^{t+1})(\widetilde{Q}_{\mathbf{c}_i^t}(\mathbf{x}_i^{t+1}) - \mathbf{w}_i^t)\right\rangle \\
&+ \frac{1}{2\eta_2}\left\|\mathbf{c} - \mathbf{c}_i^t\right\|_2^2 + \lambda R(\mathbf{x}_i^{t+1}, \mathbf{c})\Big\}
\end{aligned} \tag{A.43}$$

Note that the update on $\mathbf{w}^t$ from Algorithm 9 can be written as:

$$\mathbf{w}^{t+1} = \mathbf{w}^t - \eta_3\mathbf{g}^t, \qquad \text{where} \qquad \mathbf{g}^t = \frac{1}{n}\sum_{i=1}^n \nabla_{\mathbf{w}_i^t} F_i(\mathbf{x}_i^{t+1}, \mathbf{c}_i^{t+1}, \mathbf{w}_i^t).$$

In the convergence analysis we will require smoothness of the local functions F_i w.r.t. the global parameter $\mathbf{w}$. Recall the definition of $F_i(\mathbf{x}_i, \mathbf{c}_i, \mathbf{w}) = f_i(\mathbf{x}_i) + f_i(\widetilde{Q}_{\mathbf{c}_i}(\mathbf{x}_i)) + \lambda R(\mathbf{x}_i, \mathbf{c}_i) + \frac{\lambda_p}{2}(\|\mathbf{x}_i - \mathbf{w}\|^2 + \|\widetilde{Q}_{\mathbf{c}_i}(\mathbf{x}_i) - \mathbf{w}\|^2)$ from (A.40). It follows that F_i is $2\lambda_p$-smooth with respect to $\mathbf{w}$:

$$\left\|\nabla_{\mathbf{w}} F_i(\mathbf{x}, \mathbf{c}, \mathbf{w}) - \nabla_{\mathbf{w}'} F_i(\mathbf{x}, \mathbf{c}, \mathbf{w}')\right\| = \|\lambda_p(2\mathbf{w} - \mathbf{x} - \widetilde{Q}_{\mathbf{c}}(\mathbf{x})) - \lambda_p(2\mathbf{w}' - \mathbf{x} - \widetilde{Q}_{\mathbf{c}}(\mathbf{x}))\|$$

$$\leq 2\lambda_p\|\mathbf{w}-\mathbf{w}'\|, \quad \forall \mathbf{w},\mathbf{w}' \in \mathbb{R}^d. \tag{A.44}$$

Now let us move on with the proof.

A.2.11 Sufficient Decrease

We will divide this part into three and obtain sufficient decrease properties for each variable: $\mathbf{x}_i, \mathbf{c}_i, \mathbf{w}$.

A.2.11.1 Sufficient Decrease Due to $\mathbf{x}_i$

We begin with a useful claim.

Claim 12. *$f_i(\mathbf{x}) + f_i(\widetilde{Q}_{\mathbf{c}}(\mathbf{x})) + \frac{\lambda_p}{2}(\|\mathbf{x}-\mathbf{w}\|^2 + \|\widetilde{Q}_{\mathbf{c}}(\mathbf{x})-\mathbf{w}\|^2)$ is $(\lambda_p + \lambda_p L_{Q_1} + L + GL_{Q_1} + G_{Q_1}Ll_{Q_1})$-smooth with respect to $\mathbf{x}$.*

From Claim 12 we have:

$$\begin{aligned}
&F_i(\mathbf{x}_i^{t+1}, \mathbf{c}_i^t, \mathbf{w}^t) + (\frac{1}{2\eta_1} - \frac{\lambda_p + \lambda_p L_{Q_1} + L + GL_{Q_1} + G_{Q_1}Ll_{Q_1}}{2})\|\mathbf{x}_i^{t+1}-\mathbf{x}_i^t\|^2 \\
&= f_i(\mathbf{x}_i^{t+1}) + f_i(\widetilde{Q}_{\mathbf{c}_i^t}(\mathbf{x}_i^{t+1})) + \lambda R(\mathbf{x}_i^{t+1}, \mathbf{c}_i^t) + \frac{\lambda_p}{2}\|\mathbf{x}_i^{t+1}-\mathbf{w}^t\|^2 \\
&\quad + \frac{\lambda_p}{2}\|\widetilde{Q}_{\mathbf{c}_i^t}(\mathbf{x}_i^{t+1})-\mathbf{w}^t\|^2 + (\frac{1}{2\eta_1} - \frac{\lambda_p + \lambda_p L_{Q_1} + L + GL_{Q_1} + G_{Q_1}Ll_{Q_1}}{2})\|\mathbf{x}_i^{t+1}-\mathbf{x}_i^t\|^2 \\
&\leq f_i(\mathbf{x}_i^t) + f_i(\widetilde{Q}_{\mathbf{c}_i^t}(\mathbf{x}_i^t)) + \frac{\lambda_p}{2}\|\mathbf{x}_i^t-\mathbf{w}^t\|^2 + \frac{\lambda_p}{2}\|\widetilde{Q}_{\mathbf{c}_i^t}(\mathbf{x}_i^t)-\mathbf{w}^t\|^2 + \lambda R(\mathbf{x}_i^{t+1}, \mathbf{c}_i^t) \\
&\quad + \langle \nabla f_i(\mathbf{x}_i^t), \mathbf{x}_i^{t+1}-\mathbf{x}_i^t\rangle + \left\langle \nabla_{\mathbf{x}_i^t} f_i(\widetilde{Q}_{\mathbf{c}_i^t}(\mathbf{x}_i^t)), \mathbf{x}_i^{t+1}-\mathbf{x}_i^t\right\rangle + \langle \lambda_p(\mathbf{x}_i^t-\mathbf{w}_i^t), \mathbf{x}_i^{t+1}-\mathbf{x}_i^t\rangle \\
&\quad + \left\langle \lambda_p \nabla_{\mathbf{x}_i^t}\widetilde{Q}_{\mathbf{c}_i^t}(\mathbf{x}_i^t)(\widetilde{Q}_{\mathbf{c}_i^t}(\mathbf{x}_i^t)-\mathbf{w}_i^t), \mathbf{x}_i^{t+1}-\mathbf{x}_i^t\right\rangle + \langle \lambda_p(\mathbf{w}_i^t-\mathbf{w}^t), \mathbf{x}_i^{t+1}-\mathbf{x}_i^t\rangle \\
&\quad + \left\langle \lambda_p \nabla_{\mathbf{x}_i^t}\widetilde{Q}_{\mathbf{c}_i^t}(\mathbf{x}_i^t)(\mathbf{w}_i^t-\mathbf{w}^t), \mathbf{x}_i^{t+1}-\mathbf{x}_i^t\right\rangle + \frac{1}{2\eta_1}\|\mathbf{x}_i^{t+1}-\mathbf{x}_i^t\|^2
\end{aligned} \tag{A.45}$$

Claim 13. *Let*

$$A(\mathbf{x}_i^{t+1}) := \lambda R(\mathbf{x}_i^{t+1}, \mathbf{c}_i^t) + \langle \nabla f_i(\mathbf{x}_i^t), \mathbf{x}_i^{t+1}-\mathbf{x}_i^t\rangle + \left\langle \nabla_{\mathbf{x}_i^t} f_i(\widetilde{Q}_{\mathbf{c}_i^t}(\mathbf{x}_i^t)), \mathbf{x}_i^{t+1}-\mathbf{x}_i^t\right\rangle$$

$$+\left\langle \lambda_p \nabla_{\mathbf{x}_i^t} \widetilde{Q}_{\mathbf{c}_i^t}(\mathbf{x}_i^t)(\widetilde{Q}_{\mathbf{c}_i^t}(\mathbf{x}_i^t) - \mathbf{w}_i^t), \mathbf{x}_i^{t+1} - \mathbf{x}_i^t \right\rangle + \left\langle \lambda_p(\mathbf{x}_i^t - \mathbf{w}_i^t), \mathbf{x}_i^{t+1} - \mathbf{x}_i^t \right\rangle$$
$$+ \frac{1}{2\eta_1}\|\mathbf{x}_i^{t+1} - \mathbf{x}_i^t\|^2$$

$A(\mathbf{x}_i^t) := \lambda R(\mathbf{x}_i^t, \mathbf{c}_i^t)$.

Then $A(\mathbf{x}_i^{t+1}) \leq A(\mathbf{x}_i^t)$.

Using the inequality from Claim 13 in (A.45) gives

$$\begin{aligned}
F_i(\mathbf{x}_i^{t+1}, \mathbf{c}_i^t, \mathbf{w}^t) + (\frac{1}{2\eta_1} - \frac{\lambda_p + \lambda_p L_{Q_1} + L + GL_{Q_1} + G_{Q_1} Ll_{Q_1}}{2})\|\mathbf{x}_i^{t+1} - \mathbf{x}_i^t\|^2 \\
&\overset{(a)}{\leq} f_i(\mathbf{x}_i^t) + f_i(\widetilde{Q}_{\mathbf{c}_i^t}(\mathbf{x}_i^t)) + \frac{\lambda_p}{2}\|\mathbf{x}_i^t - \mathbf{w}^t\|^2 + \frac{\lambda_p}{2}\|\widetilde{Q}_{\mathbf{c}_i^t}(\mathbf{x}_i^t) - \mathbf{w}^t\|^2 + A(\mathbf{x}_i^{t+1}) \\
&\quad + \left\langle \lambda_p \nabla_{\mathbf{x}_i^t} \widetilde{Q}_{\mathbf{c}_i^t}(\mathbf{x}_i^t)(\mathbf{w}_i^t - \mathbf{w}^t), \mathbf{x}_i^{t+1} - \mathbf{x}_i^t \right\rangle + \left\langle \lambda_p(\mathbf{w}_i^t - \mathbf{w}^t), \mathbf{x}_i^{t+1} - \mathbf{x}_i^t \right\rangle \\
&\overset{(b)}{\leq} f_i(\mathbf{x}_i^t) + f_i(\widetilde{Q}_{\mathbf{c}_i^t}(\mathbf{x}_i^t)) + \frac{\lambda_p}{2}\|\mathbf{x}_i^t - \mathbf{w}^t\|^2 + \frac{\lambda_p}{2}\|\widetilde{Q}_{\mathbf{c}_i^t}(\mathbf{x}_i^t) - \mathbf{w}^t\|^2 + \lambda R(\mathbf{x}_i^t, \mathbf{c}_i^t) \\
&\quad + \frac{\lambda_p(1 + G_{Q_1}^2)}{2}\|\mathbf{w}_i^t - \mathbf{w}^t\|^2 + \lambda_p\|\mathbf{x}_i^{t+1} - \mathbf{x}_i^t\|^2 \\
&= F_i(\mathbf{x}_i^t, \mathbf{c}_i^t, \mathbf{w}^t) + \frac{\lambda_p(1 + G_{Q_1}^2)}{2}\|\mathbf{w}_i^t - \mathbf{w}^t\|^2 + \lambda_p\|\mathbf{x}_i^{t+1} - \mathbf{x}_i^t\|^2 \qquad \text{(A.46)}
\end{aligned}$$

To obtain (a), we substituted the value of $A(\mathbf{x}_i^{t+1})$ from Claim 13 into (A.45). In (b), we used $A(\mathbf{x}_i^{t+1}) \leq \lambda R(\mathbf{x}_i^t, \mathbf{c}_i^t)$ and $\langle \lambda_p(\mathbf{w}_i^t - \mathbf{w}^t), \mathbf{x}_i^{t+1} - \mathbf{x}_i^t \rangle = \langle \sqrt{\lambda_p}(\mathbf{w}_i^t - \mathbf{w}^t), \sqrt{\lambda_p}(\mathbf{x}_i^{t+1} - \mathbf{x}_i^t) \rangle \leq \frac{\lambda_p}{2}\|\mathbf{w}_i^t - \mathbf{w}^t\|^2 + \frac{\lambda_p}{2}\|\mathbf{x}_i^{t+1} - \mathbf{x}_i^t\|^2$, and

$$\begin{aligned}
\left\langle \lambda_p \nabla_{\mathbf{x}_i^t} \widetilde{Q}_{\mathbf{c}_i^t}(\mathbf{x}_i^t)(\mathbf{w}_i^t - \mathbf{w}^t), \mathbf{x}_i^{t+1} - \mathbf{x}_i^t \right\rangle &\leq \frac{\lambda_p}{2}\left\|\widetilde{Q}_{\mathbf{c}_i^t}(\mathbf{x}_i^t)(\mathbf{w}_i^t - \mathbf{w}^t)\right\|_2^2 + \frac{\lambda_p}{2}\|\mathbf{x}_i^{t+1} - \mathbf{x}_i^t\|^2 \\
&\leq \frac{\lambda_p}{2}\left\|\widetilde{Q}_{\mathbf{c}_i^t}(\mathbf{x}_i^t)\right\|_F^2 \|\mathbf{w}_i^t - \mathbf{w}^t\|_2^2 + \frac{\lambda_p}{2}\|\mathbf{x}_i^{t+1} - \mathbf{x}_i^t\|^2 \\
&\leq \frac{\lambda_p G_{Q_1}^2}{2}\|\mathbf{w}_i^t - \mathbf{w}^t\|^2 + \frac{\lambda_p}{2}\|\mathbf{x}_i^{t+1} - \mathbf{x}_i^t\|^2,
\end{aligned}$$

where the second inequality follows from $\|A\mathbf{x}\|_2 \leq \|A\|_F\|\mathbf{x}\|_2$, which holds for any matrix A and vector $\mathbf{x}$ (dimension compatible); and the third inequality uses Assumption **A**.5.

Substituting $\eta_1 = \frac{1}{2(3\lambda_p + \lambda_p L_{Q_1} + L + GL_{Q_1} + G_{Q_1} Ll_{Q_1})}$ in (A.46) gives:

$$F_i(\mathbf{x}_i^{t+1}, \mathbf{c}_i^t, \mathbf{w}^t) + (\frac{3\lambda_p + \lambda_p L_{Q_1} + L + GL_{Q_1} + G_{Q_1} Ll_{Q_1}}{2})\|\mathbf{x}_i^{t+1} - \mathbf{x}_i^t\|^2 \leq F_i(\mathbf{x}_i^t, \mathbf{c}_i^t, \mathbf{w}^t)$$

$$+\frac{\lambda_p(1+G_{Q_1}^2)}{2}\|\mathbf{w}_i^t-\mathbf{w}^t\|^2 \tag{A.47}$$

A.2.11.2 Sufficient Decrease Due to $\mathbf{c}_i$

In parallel with Claim 12, we have following smoothness result for $\mathbf{c}$:

Claim 14. *$f_i(\widetilde{Q}_{\mathbf{c}}(\mathbf{x}))+\frac{\lambda_p}{2}(\|\widetilde{Q}_{\mathbf{c}}(\mathbf{x})-\mathbf{w}\|^2)$ is $(\lambda_p L_{Q_2}+GL_{Q_2}+G_{Q_2}Ll_{Q_2})$-smooth with respect to $\mathbf{c}$.*

From Claim 14 we have:

$$\begin{aligned}
&F_i(\mathbf{x}_i^{t+1},\mathbf{c}_i^{t+1},\mathbf{w}^t)+(\frac{1}{2\eta_2}-\frac{\lambda_p L_{Q_2}+GL_{Q_2}+G_{Q_2}Ll_{Q_2}}{2})\|\mathbf{c}_i^{t+1}-\mathbf{c}_i^t\|^2\\
&\quad=f_i(\mathbf{x}_i^{t+1})+f_i(\widetilde{Q}_{\mathbf{c}_i^{t+1}}(\mathbf{x}_i^{t+1}))+\lambda R(\mathbf{x}_i^{t+1},\mathbf{c}_i^{t+1})+\frac{\lambda_p}{2}\|\widetilde{Q}_{\mathbf{c}_i^t}(\mathbf{x}_i^{t+1})-\mathbf{w}^t\|^2\\
&\qquad+\frac{\lambda_p}{2}\|\mathbf{x}_i^{t+1}-\mathbf{w}^t\|^2+(\frac{1}{2\eta_2}-\frac{\lambda_p L_{Q_2}+GL_{Q_2}+G_{Q_2}Ll_{Q_2}}{2})\|\mathbf{c}_i^{t+1}-\mathbf{c}_i^t\|^2\\
&\quad\le f_i(\mathbf{x}_i^{t+1})+f_i(\widetilde{Q}_{\mathbf{c}_i^t}(\mathbf{x}_i^{t+1}))+\lambda R(\mathbf{x}_i^{t+1},\mathbf{c}_i^{t+1})+\frac{\lambda_p}{2}\|\widetilde{Q}_{\mathbf{c}_i^t}(\mathbf{x}_i^{t+1})-\mathbf{w}^t\|^2\\
&\qquad+\frac{\lambda_p}{2}\|\mathbf{x}_i^{t+1}-\mathbf{w}^t\|^2+\left\langle\nabla_{\mathbf{c}_i^t}f_i(\widetilde{Q}_{\mathbf{c}_i^t}(\mathbf{x}_i^{t+1})),\mathbf{c}_i^{t+1}-\mathbf{c}_i^t\right\rangle+\frac{1}{2\eta_2}\|\mathbf{c}_i^{t+1}-\mathbf{c}_i^t\|^2\\
&\qquad+\left\langle\lambda_p\nabla_{\mathbf{c}_i^t}\widetilde{Q}_{\mathbf{c}_i^t}(\mathbf{x}_i^{t+1})(\widetilde{Q}_{\mathbf{c}_i^t}(\mathbf{x}_i^{t+1})-\mathbf{w}_i^t),\mathbf{c}_i^{t+1}-\mathbf{c}_i^t\right\rangle \qquad\text{(A.48)}\\
&\qquad+\left\langle\lambda_p\nabla_{\mathbf{c}_i^t}\widetilde{Q}_{\mathbf{c}_i^t}(\mathbf{x}_i^{t+1})(\mathbf{w}_i^t-\mathbf{w}^t),\mathbf{c}_i^{t+1}-\mathbf{c}_i^t\right\rangle \qquad\text{(A.49)}
\end{aligned}$$

Claim 15. *Let*

$$\begin{aligned}
B(\mathbf{c}_i^{t+1}):=\lambda R(\mathbf{x}_i^{t+1},\mathbf{c}_i^{t+1})&+\left\langle\nabla_{\mathbf{c}_i^t}f_i(\widetilde{Q}_{\mathbf{c}_i^t}(\mathbf{x}_i^{t+1})),\mathbf{c}_i^{t+1}-\mathbf{c}_i^t\right\rangle+\frac{1}{2\eta_2}\|\mathbf{c}_i^{t+1}-\mathbf{c}_i^t\|^2\\
&+\left\langle\lambda_p\nabla_{\mathbf{c}_i^{t+1}}\widetilde{Q}_{\mathbf{c}_i^t}(\mathbf{x}_i^{t+1})(\widetilde{Q}_{\mathbf{c}_i^t}(\mathbf{x}_i^{t+1})-\mathbf{w}_i^t),\mathbf{c}_i^{t+1}-\mathbf{c}_i^t\right\rangle\\
B(\mathbf{c}_i^t):=\lambda R(\mathbf{x}_i^{t+1},\mathbf{c}_i^t).&
\end{aligned}$$

Then $B(\mathbf{c}_i^{t+1})\le B(\mathbf{c}_i^t)$.

Substituting the bound from Claim 15 in (A.49),

$$
\begin{aligned}
&F_i(\mathbf{x}_i^{t+1}, \mathbf{c}_i^{t+1}, \mathbf{w}^t) + (\frac{1}{2\eta_2} - \frac{\lambda_p L_{Q_2} + G L_{Q_2} + G_{Q_2} L L_{Q_2}}{2}) \|\mathbf{c}_i^{t+1} - \mathbf{c}_i^t\|^2 \\
&\quad \leq f_i(\mathbf{x}_i^{t+1}) + f_i(\widetilde{Q}_{\mathbf{c}_i^t}(\mathbf{x}_i^{t+1})) + \lambda R(\mathbf{x}_i^{t+1}, \mathbf{c}_i^t) + \frac{\lambda_p}{2} \|\mathbf{x}_i^{t+1} - \mathbf{w}^t\|^2 \\
&\qquad + \frac{\lambda_p}{2} \|\widetilde{Q}_{\mathbf{c}_i^t}(\mathbf{x}_i^{t+1}) - \mathbf{w}^t\|^2 + \left\langle \lambda_p \nabla_{\mathbf{c}_i^t} \widetilde{Q}_{\mathbf{c}_i^t}(\mathbf{x}_i^{t+1})(\mathbf{w}_i^t - \mathbf{w}^t), \mathbf{c}_i^{t+1} - \mathbf{c}_i^t \right\rangle \\
&\quad = F_i(\mathbf{x}_i^{t+1}, \mathbf{c}_i^t, \mathbf{w}^t) + \frac{\lambda_p}{2} \|\mathbf{c}_i^{t+1} - \mathbf{c}_i^t\|^2 + \frac{\lambda_p G_{Q_2}^2}{2} \|\mathbf{w}_i^t - \mathbf{w}^t\|^2
\end{aligned}
$$

Substituting $\eta_2 = \frac{1}{2(\lambda_p + \lambda_p L_{Q_2} + G L_{Q_2} + G_{Q_2} L L_{Q_2})}$ gives us:

$$
\begin{aligned}
F_i(\mathbf{x}_i^{t+1}, \mathbf{c}_i^{t+1}, \mathbf{w}^t) + \frac{\lambda_p + \lambda_p L_{Q_2} + G L_{Q_2} + G_{Q_2} L L_{Q_2}}{2} \|\mathbf{c}_i^{t+1} - \mathbf{c}_i^t\|^2 \leq F_i(\mathbf{x}_i^{t+1}, \mathbf{c}_i^t, \mathbf{w}^t) \\
+ \frac{\lambda_p G_{Q_2}^2}{2} \|\mathbf{w}_i^t - \mathbf{w}^t\|^2.
\end{aligned} \tag{A.50}
$$

A.2.11.3 Sufficient Decrease Due to w

Now, we use $2\lambda_p$-smoothness of $F_i(\mathbf{x}, \mathbf{c}, \mathbf{w})$ with respect to $\mathbf{w}$:

$$
\begin{aligned}
F_i(\mathbf{x}_i^{t+1}, \mathbf{c}_i^{t+1}, \mathbf{w}^{t+1}) \leq F_i(\mathbf{x}_i^{t+1}, \mathbf{c}_i^{t+1}, \mathbf{w}^t) + \left\langle \nabla_{\mathbf{w}^t} F_i(\mathbf{x}_i^{t+1}, \mathbf{c}_i^{t+1}, \mathbf{w}^t), \mathbf{w}^{t+1} - \mathbf{w}^t \right\rangle \\
+ \lambda_p \|\mathbf{w}^{t+1} - \mathbf{w}^t\|^2
\end{aligned}
$$

Then we have,

$$
\begin{aligned}
F_i(\mathbf{x}_i^{t+1}, \mathbf{c}_i^{t+1}, \mathbf{w}^{t+1}) &\leq F_i(\mathbf{x}_i^{t+1}, \mathbf{c}_i^{t+1}, \mathbf{w}^t) + \left\langle \nabla_{\mathbf{w}^t} F_i(\mathbf{x}_i^{t+1}, \mathbf{c}_i^{t+1}, \mathbf{w}^t), \mathbf{w}^{t+1} - \mathbf{w}^t \right\rangle \\
&\quad + \lambda_p \|\mathbf{w}^{t+1} - \mathbf{w}^t\|^2 \\
&= F_i(\mathbf{x}_i^{t+1}, \mathbf{c}_i^{t+1}, \mathbf{w}^t) + \left\langle \nabla_{\mathbf{w}^t} F_i(\mathbf{x}_i^{t+1}, \mathbf{c}_i^{t+1}, \mathbf{w}^t), \eta_3 \mathbf{g}^t \right\rangle + \lambda_p \|\eta_3 \mathbf{g}^t\|^2 \\
&= F_i(\mathbf{x}_i^{t+1}, \mathbf{c}_i^{t+1}, \mathbf{w}^t) - \eta_3 \left\langle \nabla_{\mathbf{w}^t} F_i(\mathbf{x}_i^{t+1}, \mathbf{c}_i^{t+1}, \mathbf{w}^t), \mathbf{g}^t - \nabla_{\mathbf{w}^t} F_i(\mathbf{x}_i^{t+1}, \mathbf{c}_i^{t+1}, \mathbf{w}^t) \right. \\
&\quad \left. + \nabla_{\mathbf{w}^t} F_i(\mathbf{x}_i^{t+1}, \mathbf{c}_i^{t+1}, \mathbf{w}^t) \right\rangle \\
&\quad + \lambda_p 2\eta_3^2 \|\mathbf{g}^t - \nabla_{\mathbf{w}^t} F_i(\mathbf{x}_i^{t+1}, \mathbf{c}_i^{t+1}, \mathbf{w}^t) + \nabla_{\mathbf{w}^t} F_i(\mathbf{x}_i^{t+1}, \mathbf{c}_i^{t+1}, \mathbf{w}^t)\|^2 \\
&= F_i(\mathbf{x}_i^{t+1}, \mathbf{c}_i^{t+1}, \mathbf{w}^t) - \eta_3 \left\| \nabla_{\mathbf{w}^t} F_i(\mathbf{x}_i^{t+1}, \mathbf{c}_i^{t+1}, \mathbf{w}^t) \right\|^2
\end{aligned}
$$

$$
\begin{aligned}
&- \eta_3 \left\langle \nabla_{\mathbf{w}^t} F_i(\mathbf{x}_i^{t+1}, \mathbf{c}_i^{t+1}, \mathbf{w}^t), \mathbf{g}^t - \nabla_{\mathbf{w}^t} F_i(\mathbf{x}_i^{t+1}, \mathbf{c}_i^{t+1}, \mathbf{w}^t) \right\rangle \\
&+ \lambda_p \eta_3^2 \|\mathbf{g}^t - \nabla_{\mathbf{w}^t} F_i(\mathbf{x}_i^{t+1}, \mathbf{c}_i^{t+1}, \mathbf{w}^t) + \nabla_{\mathbf{w}^t} F_i(\mathbf{x}_i^{t+1}, \mathbf{c}_i^{t+1}, \mathbf{w}^t)\|^2 \\
\leq\ & F_i(\mathbf{x}_i^{t+1}, \mathbf{c}_i^{t+1}, \mathbf{w}^t) - \eta_3 \left\| \nabla_{\mathbf{w}^t} F_i(\mathbf{x}_i^{t+1}, \mathbf{c}_i^{t+1}, \mathbf{w}^t) \right\|^2 \\
&+ \frac{\eta_3}{2} \left\| \nabla_{\mathbf{w}^t} F_i(\mathbf{x}_i^{t+1}, \mathbf{c}_i^{t+1}, \mathbf{w}^t) \right\|^2 + \frac{\eta_3}{2} \left\| \mathbf{g}^t - \nabla_{\mathbf{w}^t} F_i(\mathbf{x}_i^{t+1}, \mathbf{c}_i^{t+1}, \mathbf{w}^t) \right\|^2 \\
&+ 2\lambda_p \eta_3^2 \left(\left\| \mathbf{g}^t - \nabla_{\mathbf{w}^t} F_i(\mathbf{x}_i^{t+1}, \mathbf{c}_i^{t+1}, \mathbf{w}^t) \right\|^2 + \left\| \nabla_{\mathbf{w}^t} F_i(\mathbf{x}_i^{t+1}, \mathbf{c}_i^{t+1}, \mathbf{w}^t) \right\|^2 \right) \\
=\ & F_i(\mathbf{x}_i^{t+1}, \mathbf{c}_i^{t+1}, \mathbf{w}^t) - (\frac{\eta_3}{2} - 2\lambda_p \eta_3^2) \left\| \nabla_{\mathbf{w}^t} F_i(\mathbf{x}_i^{t+1}, \mathbf{c}_i^{t+1}, \mathbf{w}^t) \right\|^2 \\
&+ (\frac{\eta_3}{2} + 2\lambda_p \eta_3^2) \left\| \mathbf{g}^t - \nabla_{\mathbf{w}_i^t} F_i(\mathbf{x}_i^{t+1}, \mathbf{c}_i^{t+1}, \mathbf{w}_i^t) + \nabla_{\mathbf{w}_i^t} F_i(\mathbf{x}_i^{t+1}, \mathbf{c}_i^{t+1}, \mathbf{w}_i^t) \right. \\
&\left. - \nabla_{\mathbf{w}^t} F_i(\mathbf{x}_i^{t+1}, \mathbf{c}_i^{t+1}, \mathbf{w}^t) \right\|^2 \\
\leq\ & F_i(\mathbf{x}_i^{t+1}, \mathbf{c}_i^{t+1}, \mathbf{w}^t) - (\frac{\eta_3}{2} - 2\lambda_p \eta_3^2) \left\| \nabla_{\mathbf{w}^t} F_i(\mathbf{x}_i^{t+1}, \mathbf{c}_i^{t+1}, \mathbf{w}^t) \right\|^2 \\
&+ (\eta_3 + 4\lambda_p \eta_3^2) \left\| \mathbf{g}^t - \nabla_{\mathbf{w}_i^t} F_i(\mathbf{x}_i^{t+1}, \mathbf{c}_i^{t+1}, \mathbf{w}_i^t) \right\|^2 \\
&+ (\eta_3 + 4\lambda_p \eta_3^2) \left\| \nabla_{\mathbf{w}_i^t} F_i(\mathbf{x}_i^{t+1}, \mathbf{c}_i^{t+1}, \mathbf{w}_i^t) - \nabla_{\mathbf{w}^t} F_i(\mathbf{x}_i^{t+1}, \mathbf{c}_i^{t+1}, \mathbf{w}^t) \right\|^2 \\
\leq\ & F_i(\mathbf{x}_i^{t+1}, \mathbf{c}_i^{t+1}, \mathbf{w}^t) - (\frac{\eta_3}{2} - 2\lambda_p \eta_3^2) \left\| \nabla_{\mathbf{w}^t} F_i(\mathbf{x}_i^{t+1}, \mathbf{c}_i^{t+1}, \mathbf{w}^t) \right\|^2 \\
&+ (\eta_3 + 4\lambda_p \eta_3^2) \left\| \mathbf{g}^t - \nabla_{\mathbf{w}_i^t} F_i(\mathbf{x}_i^{t+1}, \mathbf{c}_i^{t+1}, \mathbf{w}_i^t) \right\|^2 \\
&+ (\eta_3 + 4\lambda_p \eta_3^2) 4\lambda_p^2 \left\| \mathbf{w}_i^t - \mathbf{w}^t \right\|^2
\end{aligned}
$$

Rearranging the terms, we have:

$$F_i(\mathbf{x}_i^{t+1}, \mathbf{c}_i^{t+1}, \mathbf{w}^{t+1}) + (\frac{\eta_3}{2} - 2\lambda_p\eta_3^2)\left\|\nabla_{\mathbf{w}^t} F_i(\mathbf{x}_i^{t+1}, \mathbf{c}_i^{t+1}, \mathbf{w}^t)\right\|^2$$
$$\leq (\eta_3 + 4\lambda_p\eta_3^2)\left\|\mathbf{g}^t - \nabla_{\mathbf{w}_i^t} F_i(\mathbf{x}_i^{t+1}, \mathbf{c}_i^{t+1}, \mathbf{w}_i^t)\right\|^2 + (\eta_3 + 4\lambda_p\eta_3^2)4\lambda_p^2\left\|\mathbf{w}_i^t - \mathbf{w}^t\right\|^2$$
$$+ F_i(\mathbf{x}_i^{t+1}, \mathbf{c}_i^{t+1}, \mathbf{w}^t) \tag{A.51}$$

A.2.11.4 Overall Decrease

Define

$$L_x = 3\lambda_p + \lambda_p L_{Q_1} + L + GL_{Q_1} + G_{Q_1} Ll_{Q_1} \tag{A.52}$$

$$L_c = \lambda_p + \lambda_p L_{Q_2} + GL_{Q_2} + G_{Q_2} Ll_{Q_2}. \tag{A.53}$$

Summing (A.47), (A.50), (A.51) we get the overall decrease property:

$$F_i(\mathbf{x}_i^{t+1}, \mathbf{c}_i^{t+1}, \mathbf{w}^{t+1}) + (\frac{\eta_3}{2} - 2\lambda_p\eta_3^2)\left\|\nabla_{\mathbf{w}^t} F_i(\mathbf{x}_i^{t+1}, \mathbf{c}_i^{t+1}, \mathbf{w}^t)\right\|^2 + \frac{L_x}{2}\|\mathbf{x}_i^{t+1} - \mathbf{x}_i^t\|^2$$
$$+ \frac{L_c}{2}\|\mathbf{c}_i^{t+1} - \mathbf{c}_i^t\|^2 \leq (\eta_3 + 4\lambda_p\eta_3^2)\left\|\mathbf{g}^t - \nabla_{\mathbf{w}_i^t} F_i(\mathbf{x}_i^{t+1}, \mathbf{c}_i^{t+1}, \mathbf{w}_i^t)\right\|^2$$
$$+ (1 + G_{Q_1}^2 + G_{Q_2}^2 + 4\eta_3\lambda_p + 16\lambda_p^2\eta_3^2)\lambda_p\|\mathbf{w}_i^t - \mathbf{w}^t\|^2 + F_i(\mathbf{x}_i^t, \mathbf{c}_i^t, \mathbf{w}^t) \tag{A.54}$$

Let

$$L_{\min} = \min\{L_x, L_c, (\eta_3 - 4\lambda_p\eta_3^2)\}. \tag{A.55}$$

Then,

$$F_i(\mathbf{x}_i^{t+1}, \mathbf{c}_i^{t+1}, \mathbf{w}^{t+1}) + \frac{L_{\min}}{2}\left(\left\|\nabla_{\mathbf{w}^t} F_i(\mathbf{x}_i^{t+1}, \mathbf{c}_i^{t+1}, \mathbf{w}^t)\right\|^2 + \|\mathbf{x}_i^{t+1} - \mathbf{x}_i^t\|^2 + \|\mathbf{c}_i^{t+1} - \mathbf{c}_i^t\|^2\right)$$
$$\leq (\eta_3 + 4\lambda_p\eta_3^2)\left\|\mathbf{g}^t - \nabla_{\mathbf{w}_i^t} F_i(\mathbf{x}_i^{t+1}, \mathbf{c}_i^{t+1}, \mathbf{w}_i^t)\right\|^2 + F_i(\mathbf{x}_i^t, \mathbf{c}_i^t, \mathbf{w}^t)$$
$$+ (1 + G_{Q_1}^2 + G_{Q_2}^2 + 4\eta_3\lambda_p + 16\lambda_p^2\eta_3^2)\lambda_p\|\mathbf{w}_i^t - \mathbf{w}^t\|^2 \tag{A.56}$$

A.2.12 Bound on the Gradient

Now, we will use the first order optimality conditions due to proximal updates and bound the partial gradients with respect to variables $\mathbf{x}$ and $\mathbf{c}$. After obtaining bounds for partial gradients we will bound the overall gradient and use our results from Section A.2.11 to arrive at the final bound.

A.2.12.1 Bound on the Gradient w.r.t. $\mathbf{x}_i$

Taking the derivative inside the minimization problem (A.42) with respect to $\mathbf{x}$ at $\mathbf{x} = \mathbf{x}_i^{t+1}$ and setting it to 0 gives the following optimality condition:

$$\nabla f_i(\mathbf{x}_i^t) + \nabla_{\mathbf{x}_i^t} f_i(\widetilde{Q}_{\mathbf{c}_i^t}(\mathbf{x}_i^t)) + \lambda_p(\mathbf{x}_i^t - \mathbf{w}_i^t) + \lambda_p \nabla_{\mathbf{x}_i^t} \widetilde{Q}_{\mathbf{c}_i^t}(\mathbf{x}_i^t)(\widetilde{Q}_{\mathbf{c}_i^t}(\mathbf{x}_i^t) - \mathbf{w}_i^t) + \frac{1}{\eta_1}(\mathbf{x}_i^{t+1} - \mathbf{x}_i^t) + \lambda \nabla_{\mathbf{x}_i^{t+1}} R(\mathbf{x}_i^{t+1}, \mathbf{c}_i^t) = 0 \quad \text{(A.57)}$$

Then we have,

$$\begin{aligned}
\left\| \nabla_{\mathbf{x}_i^{t+1}} F_i(\mathbf{x}_i^{t+1}, \mathbf{c}_i^t, \mathbf{w}^t) \right\| &= \Big\| \nabla f_i(\mathbf{x}_i^{t+1}) + \nabla_{\mathbf{x}_i^{t+1}} f_i(\widetilde{Q}_{\mathbf{c}_i^t}(\mathbf{x}_i^{t+1})) + \lambda_p(\mathbf{x}_i^{t+1} - \mathbf{w}^t) \\
&\quad + \lambda_p \nabla_{\mathbf{x}_i^{t+1}} \widetilde{Q}_{\mathbf{c}_i^t}(\mathbf{x}_i^{t+1})(\widetilde{Q}_{\mathbf{c}_i^t}(\mathbf{x}_i^{t+1}) - \mathbf{w}^t) + \lambda \nabla_{\mathbf{x}_i^{t+1}} R(\mathbf{x}_i^{t+1}, \mathbf{c}_i^t) \Big\| \\
&\overset{(a)}{=} \Big\| \nabla f_i(\mathbf{x}_i^{t+1}) - \nabla f_i(\mathbf{x}_i^t) + \nabla_{\mathbf{x}_i^{t+1}} f_i(\widetilde{Q}_{\mathbf{c}_i^t}(\mathbf{x}_i^{t+1})) - \nabla_{\mathbf{x}_i^t} f_i(\widetilde{Q}_{\mathbf{c}_i^t}(\mathbf{x}_i^t)) \\
&\quad + \lambda_p \nabla_{\mathbf{x}_i^{t+1}} \widetilde{Q}_{\mathbf{c}_i^t}(\mathbf{x}_i^{t+1})(\widetilde{Q}_{\mathbf{c}_i^t}(\mathbf{x}_i^{t+1}) - \mathbf{w}^t) \\
&\quad - \lambda_p \nabla_{\mathbf{x}_i^t} \widetilde{Q}_{\mathbf{c}_i^t}(\mathbf{x}_i^t)(\widetilde{Q}_{\mathbf{c}_i^t}(\mathbf{x}_i^t) - \mathbf{w}_i^t) \\
&\quad + (\lambda_p - \frac{1}{\eta_1})(\mathbf{x}_i^{t+1} - \mathbf{x}_i^t) + \lambda_p(\mathbf{w}_i^t - \mathbf{w}^t) \Big\| \\
&\leq (\frac{1}{\eta_1} + \lambda_p + \lambda_p G_{Q_1} l_{Q_1} + L + G L_{Q_1} + G_{Q_1} L l_{Q_1}) \|\mathbf{x}_i^{t+1} - \mathbf{x}_i^t\| \\
&\quad + \lambda_p(1 + G_{Q_1}) \|\mathbf{w}_i^t - \mathbf{w}^t\|
\end{aligned}$$

where (a) is from (A.57) and the last inequality is due to Lipschitz continuous gradients and triangle inequality. This implies:

$$\left\|\nabla_{\mathbf{x}_i^{t+1}} F_i(\mathbf{x}_i^{t+1}, \mathbf{c}_i^t, \mathbf{w}^t)\right\|^2 \leq 2(\frac{1}{\eta_1} + \lambda_p + \lambda_p G_{Q_1} l_{Q_1} + L + GL_{Q_1} + G_{Q_1} Ll_{Q_1})^2 \|\mathbf{x}_i^{t+1} - \mathbf{x}_i^t\|^2 + 2\lambda_p^2 (1 + G_{Q_1})^2 \|\mathbf{w}_i^t - \mathbf{w}^t\|^2$$

Substituting $\eta_1 = \frac{1}{2(3\lambda_p + \lambda_p L_{Q_1} + L + GL_{Q_1} + G_{Q_1} Ll_{Q_1})}$ we have:

$$\begin{aligned}\left\|\nabla_{\mathbf{x}_i^{t+1}} F_i(\mathbf{x}_i^{t+1}, \mathbf{c}_i^t, \mathbf{w}^t)\right\|^2 &\leq 2(7\lambda_p + 2\lambda_p L_{Q_1} + \lambda_p G_{Q_1} l_{Q_1} \\ &\quad + 3(L + GL_{Q_1} + G_{Q_1} Ll_{Q_1}))^2 \|\mathbf{x}_i^{t+1} - \mathbf{x}_i^t\|^2 \\ &\quad + 2\lambda_p^2 (1 + G_{Q_1})^2 \|\mathbf{w}_i^t - \mathbf{w}^t\|^2 \\ &= 18(\frac{7}{3}\lambda_p + \frac{2\lambda_p L_{Q_1}}{3} + \frac{\lambda_p G_{Q_1} l_{Q_1}}{3} + L + Gl_{Q_1} \\ &\quad + G_{Q_1} Ll_{Q_1})^2 \|\mathbf{x}_i^{t+1} - \mathbf{x}_i^t\|^2 \\ &\quad + 2\lambda_p^2 (1 + G_{Q_1})^2 \|\mathbf{w}_i^t - \mathbf{w}^t\|^2 \end{aligned} \tag{A.58}$$

A.2.12.2 Bound on the Gradient w.r.t. $\mathbf{c}_i$

Similarly. taking the derivative inside the minimization problem (A.43) with respect to $\mathbf{c}$ at $\mathbf{c}_i^{t+1}$ and setting it to 0 gives the following optimality condition:

$$\nabla_{\mathbf{c}_i^t} f_i(\widetilde{Q}_{\mathbf{c}_i^t}(\mathbf{x}_i^{t+1})) + \frac{1}{\eta_2}(\mathbf{c}_i^{t+1} - \mathbf{c}_i^t) + \lambda_p \nabla_{\mathbf{c}_i^t} \widetilde{Q}_{\mathbf{c}_i^t}(\mathbf{x}_i^{t+1})(\widetilde{Q}_{\mathbf{c}_i^t}(\mathbf{x}_i^{t+1}) - \mathbf{w}_i^t) + \lambda \nabla_{\mathbf{c}_i^{t+1}} R(\mathbf{x}_i^{t+1}, \mathbf{c}_i^{t+1}) = 0 \tag{A.59}$$

Then we have

$$\left\|\nabla_{\mathbf{c}_i^{t+1}} F_i(\mathbf{x}_i^{t+1}, \mathbf{c}_i^{t+1}, \mathbf{w}^t)\right\| = \left\|\nabla_{\mathbf{c}_i^{t+1}} f_i(\widetilde{Q}_{\mathbf{c}_i^{t+1}}(\mathbf{x}_i^{t+1})) + \lambda_p \nabla_{\mathbf{c}_i^{t+1}} \widetilde{Q}_{\mathbf{c}_i^{t+1}}(\mathbf{x}_i^{t+1})(\widetilde{Q}_{\mathbf{c}_i^{t+1}}(\mathbf{x}_i^{t+1}) - \mathbf{w}_i^t)\right.$$

$$
\begin{aligned}
&\quad + \lambda \nabla_{\mathbf{c}_i^{t+1}} R(\mathbf{x}_i^{t+1}, \mathbf{c}_i^{t+1}) \Big\| \\
&= \Big\| \nabla_{\mathbf{c}_i^{t+1}} f_i(\widetilde{Q}_{\mathbf{c}_i^{t+1}}(\mathbf{x}_i^{t+1})) - \nabla_{\mathbf{c}_i^t} f_i(\widetilde{Q}_{\mathbf{c}_i^t}(\mathbf{x}_i^{t+1})) + \frac{1}{\eta_2}(\mathbf{c}_i^t - \mathbf{c}_i^{t+1}) \\
&\quad + \lambda_p \nabla_{\mathbf{c}_i^{t+1}} \widetilde{Q}_{\mathbf{c}_i^{t+1}}(\mathbf{x}_i^{t+1})(\widetilde{Q}_{\mathbf{c}_i^{t+1}}(\mathbf{x}_i^{t+1}) - \mathbf{w}_i^t) \\
&\quad - \lambda_p \nabla_{\mathbf{c}_i^t} \widetilde{Q}_{\mathbf{c}_i^t}(\mathbf{x}_i^{t+1})(\widetilde{Q}_{\mathbf{c}_i^t}(\mathbf{x}_i^{t+1}) - \mathbf{w}^t) \Big\| \\
&\leq (\frac{1}{\eta_2} + \lambda_p G_{Q_2} l_{Q_2} + G L_{Q_2} + G_{Q_2} L l_{Q_2}) \|\mathbf{c}_i^{t+1} - \mathbf{c}_i^t\| \\
&\quad + \lambda_p G_{Q_2} \|\mathbf{w}_i^t - \mathbf{w}^t\|
\end{aligned}
$$

the first equality is due to (A.59) and the last inequality is due to Lipschitz continuous gradient and triangle inequality. As a result we have,

$$
\begin{aligned}
\Big\| \nabla_{\mathbf{c}_i^{t+1}} F_i(\mathbf{x}_i^{t+1}, \mathbf{c}_i^{t+1}, \mathbf{w}^t) \Big\|^2 &\leq 2(\frac{1}{\eta_2} + \lambda_p G_{Q_2} l_{Q_2} + G L_{Q_2} + G_{Q_2} L l_{Q_2})^2 \|\mathbf{c}_i^{t+1} - \mathbf{c}_i^t\|^2 \\
&\quad + 2\lambda_p^2 G_{Q_2}^2 \|\mathbf{w}_i^t - \mathbf{w}^t\|^2 \qquad \text{(A.60)}
\end{aligned}
$$

substituting $\eta_2 = \frac{1}{2(\lambda_p + \lambda_p L_{Q_2} + G L_{Q_2} + G_{Q_2} L L_{Q_2})}$ we have:

$$
\begin{aligned}
\Big\| \nabla_{\mathbf{c}_i^{t+1}} F_i(\mathbf{x}_i^{t+1}, \mathbf{c}_i^{t+1}, \mathbf{w}^t) \Big\|^2 &\leq 18(\frac{2\lambda_p}{3} + \frac{2\lambda_p L_{Q_2}}{3} + G L_{Q_2} + G_{Q_2} L L_{Q_2})^2 \|\mathbf{c}_i^{t+1} - \mathbf{c}_i^t\|^2 \\
&\quad + 2\lambda_p^2 G_{Q_2}^2 \|\mathbf{w}_i^t - \mathbf{w}^t\|^2
\end{aligned}
$$

A.2.12.3 Overall Bound

Then, let us write $\|\mathbf{G}_i^t\|^2$:

$$
\begin{aligned}
\|\mathbf{G}_i^t\|^2 &= \Big\| [\nabla_{\mathbf{x}_i^{t+1}} F_i(\mathbf{x}_i^{t+1}, \mathbf{c}_i^t, \mathbf{w}^t)^T, \nabla_{\mathbf{c}_i^{t+1}} F_i(\mathbf{x}_i^{t+1}, \mathbf{c}_i^{t+1}, \mathbf{w}^t)^T, \nabla_{\mathbf{w}^t} F_i(\mathbf{x}_i^{t+1}, \mathbf{c}_i^{t+1}, \mathbf{w}^t)^T]^T \Big\|^2 \\
&= \Big\| \nabla_{\mathbf{x}_i^{t+1}} F_i(\mathbf{x}_i^{t+1}, \mathbf{c}_i^t, \mathbf{w}^t) \Big\|^2 + \Big\| \nabla_{\mathbf{c}_i^{t+1}} F_i(\mathbf{x}_i^{t+1}, \mathbf{c}_i^{t+1}, \mathbf{w}^t) \Big\|^2 + \Big\| \nabla_{\mathbf{w}^t} F_i(\mathbf{x}_i^{t+1}, \mathbf{c}_i^{t+1}, \mathbf{w}^t) \Big\|^2
\end{aligned}
$$

$$\leq 18(\frac{7}{3}\lambda_p + \frac{2\lambda_p L_{Q_1}}{3} + \frac{\lambda_p G_{Q_1} l_{Q_1}}{3} + L + Gl_{Q_1} + G_{Q_1} L l_{Q_1})^2 \|\mathbf{x}_i^{t+1} - \mathbf{x}_i^t\|^2$$
$$+ 18(\frac{2\lambda_p}{3} + \frac{2\lambda_p L_{Q_2}}{3} + GL_{Q_2} + G_{Q_2} L L_{Q_2})^2 \|\mathbf{c}_i^{t+1} - \mathbf{c}_i^t\|^2 + \left\|\nabla_{\mathbf{w}^t} F_i(\mathbf{x}_i^{t+1}, \mathbf{c}_i^{t+1}, \mathbf{w}^t)\right\|^2$$
$$+ 2\lambda_p^2((1+G_{Q_1})^2 + G_{Q_2}^2)\|\mathbf{w}_i^t - \mathbf{w}^t\|^2$$

where the last inequality is due to (A.58) and (A.60). Let

$$L_{\max} = \max\left\{\sqrt{\frac{1}{18}}, \frac{2\lambda_p}{3} + \frac{2\lambda_p L_{Q_2}}{3} + GL_{Q_2} + G_{Q_2} L L_{Q_2}, \right.$$
$$\left. \frac{7}{3}\lambda_p + \frac{2\lambda_p L_{Q_1}}{3} + \frac{\lambda_p G_{Q_1} l_{Q_1}}{3} + L + Gl_{Q_1} + G_{Q_1} L l_{Q_1}\right\}. \tag{A.61}$$

Then,

$$\left\|[\nabla_{\mathbf{x}_i^{t+1}} F_i(\mathbf{x}_i^{t+1}, \mathbf{c}_i^t, \mathbf{w}^t)^T, \nabla_{\mathbf{c}_i^{t+1}} F_i(\mathbf{x}_i^{t+1}, \mathbf{c}_i^{t+1}, \mathbf{w}^t)^T, \nabla_{\mathbf{w}^t} F_i(\mathbf{x}_i^{t+1}, \mathbf{c}_i^{t+1}, \mathbf{w}^t)^T]^T\right\|^2$$
$$\leq 18L_{\max}^2(\|\mathbf{x}_i^{t+1} - \mathbf{x}_i^t\|^2 + \|\mathbf{c}_i^{t+1} - \mathbf{c}_i^t\|^2 + \left\|\nabla_{\mathbf{w}^t} F_i(\mathbf{x}_i^{t+1}, \mathbf{c}_i^{t+1}, \mathbf{w}^t)\right\|^2)$$
$$+ 2\lambda_p^2((1+G_{Q_1})^2 + G_{Q_2}^2)\|\mathbf{w}_i^t - \mathbf{w}^t\|^2$$
$$\overset{(a)}{\leq} 36\frac{L_{\max}^2}{L_{\min}}\left[(\eta_3 + 4\lambda_p\eta_3^2)\left\|\mathbf{g}^t - \nabla_{\mathbf{w}_i^t} F_i(\mathbf{x}_i^{t+1}, \mathbf{c}_i^{t+1}, \mathbf{w}_i^t)\right\|^2\right.$$
$$+ (1 + G_{Q_1}^2 + G_{Q_2}^2 + 4\eta_3\lambda_p + 16\lambda_p^2\eta_3^2)\lambda_p\|\mathbf{w}_i^t - \mathbf{w}^t\|^2$$
$$\left. + F_i(\mathbf{x}_i^t, \mathbf{c}_i^t, \mathbf{w}^t) - F_i(\mathbf{x}_i^{t+1}, \mathbf{c}_i^{t+1}, \mathbf{w}^{t+1})\right] + 2\lambda_p^2((1+G_{Q_1})^2 + G_{Q_2}^2)\|\mathbf{w}_i^t - \mathbf{w}^t\|^2. \tag{A.62}$$

In (a) we use the bound from (A.56), and $L_{\min}$ is defined in (A.55).

Now we state a useful lemma that enables us to relate local version of the global model, $\mathbf{w}_i$, to global model itself, $\mathbf{w}$.

Lemma 7. *Let η_3 be chosen such that $\eta_3 \leq \sqrt{\frac{1}{48\tau^2\lambda_p^2}}$, then we have,*

$$\frac{1}{T}\sum_{t=0}^{T-1}\frac{1}{n}\sum_{i=1}^{n}\|\mathbf{w}^t - \mathbf{w}_i^t\|^2 \leq 6\tau^2\eta_3^2\kappa.$$

As a corollary:

Corollary 4. *Recall,* $\mathbf{g}^t = \frac{1}{n}\sum_{i=1}^n \nabla_{\mathbf{w}^t} F_i(\mathbf{x}_i^{t+1}, \mathbf{c}_i^{t+1}, \mathbf{w}_i^t)$. *Then, we have:*

$$\frac{1}{T}\sum_{t=0}^{T-1}\frac{1}{n}\sum_{i=1}^{n}\left\|\mathbf{g}^t - \nabla_{\mathbf{w}_i^t} F_i(\mathbf{x}_i^{t+1}, \mathbf{c}_i^{t+1}, \mathbf{w}_i^t)\right\|^2 \leq 144\lambda_p^2\tau^2\eta_3^2\kappa + 3\kappa.$$

See end of the section for the proofs. Using Lemma 7 and Corollary 4, summing the bound in (A.62) over time and clients, dividing by T and n:

$$\begin{aligned}
\frac{1}{T}\sum_{t=0}^{T-1}\frac{1}{n}\sum_{i=1}^{n}\|\mathbf{G}_i^t\|^2 \leq 36\frac{L_{\max}^2}{L_{\min}}\Bigg[&(\eta_3 + 4\lambda_p\eta_3^2)(144\lambda_p^2\tau^2\eta_3^2\kappa + 3\kappa) \\
&+ (1 + G_{Q_1}^2 + G_{Q_2}^2 + 4\eta_3\lambda_p + 16\lambda_p^2\eta_3^2)\lambda_p 6\tau^2\eta_3^2\kappa \\
&+ \frac{\sum_{i=1}^n \left(F_i(\mathbf{x}_i^0, \mathbf{c}_i^0, \mathbf{w}_i^0) - F_i(\mathbf{x}_i^T, \mathbf{c}_i^T, \mathbf{w}_i^T)\right)}{nT}\Bigg] \\
&+ 2\lambda_p^2((1 + G_{Q_1})^2 + G_{Q_2}^2)6\tau^2\eta_3^2\kappa \\
= 36\frac{L_{\max}^2}{L_{\min}}\Bigg[&6\tau^2\eta_3^2\kappa(\lambda_p(1 + G_{Q_1}^2 + G_{Q_2}^2) + 28\eta_3\lambda_p^2 + 112\eta_3^2\lambda_p^3) \\
&+ 3\eta_3\kappa + 12\lambda_p\eta_3^2\kappa + \frac{\Delta_F}{T}\Bigg] + 12\lambda_p^2((1 + G_{Q_1})^2 + G_{Q_2}^2)\tau^2\eta_3^2\kappa \quad \text{(A.63)}
\end{aligned}$$

where $\Delta_F = \frac{\sum_{i=1}^n \left(F_i(\mathbf{x}_i^0, \mathbf{c}_i^0, \mathbf{w}_i^0) - F_i(\mathbf{x}_i^T, \mathbf{c}_i^T, \mathbf{w}_i^T)\right)}{n}$.

Choice of η_3. Assuming $\tau \leq \sqrt{T}$ we can take $\eta_3 = \frac{1}{4\lambda_p\sqrt{T}}$, details of this choice is discussed in the end of this section, and after some algebra we have the end result:

$$\begin{aligned}
\frac{1}{T}\sum_{t=0}^{T-1}\frac{1}{n}\sum_{i=1}^{n}\|\mathbf{G}_i^t\|^2 \leq &\frac{54L_{\max}^2\tau\kappa(1 + G_{Q_1}^2 + G_{Q_2}^2) + 54L_{\max}^2 + 576L_{\max}^2\lambda_p\Delta_F}{\sqrt{T}} \\
&+ \frac{1512L_{\max}^2\tau^2\kappa + \frac{3}{16}\tau^2\kappa^2((1 + G_{Q_1}^2) + G_{Q_2}^2)}{T} \\
&+ \frac{6048L_{\max}^2\tau^2\kappa}{T^{\frac{3}{2}}} + 216L_{\max}^2\kappa
\end{aligned}$$

This concludes the proof.

A.2.13 Proof of Lemma 7 and Corollary 4

Let us restate and prove Lemma 7,

Lemma (Restating Lemma 7)**.**

$$\frac{1}{T}\sum_{t=0}^{T-1}\frac{1}{n}\sum_{i=1}^{n}\|\mathbf{w}^t-\mathbf{w}_i^t\|^2\leq 6\tau^2\eta_3^2\kappa \tag{A.64}$$

Proof. Let t_c be the latest synchronization time before t. Define $\gamma_t=\frac{1}{n}\sum_{i=1}^{n}\|\mathbf{w}^t-\mathbf{w}_i^t\|^2$. Then:

$$\begin{aligned}
\gamma_t &= \frac{1}{n}\sum_{i=1}^{n}\left\|\mathbf{w}^{tc}-\frac{\eta_3}{n}\sum_{j=t_c}^{t}\sum_{k=1}^{n}\nabla_{\mathbf{w}_k^j}F_k(\mathbf{x}_k^{j+1},\mathbf{c}_k^{j+1},\mathbf{w}_k^j-(\mathbf{w}^{tc}-\eta_3\sum_{j=t_c}^{t}\nabla_{\mathbf{w}_i^j}F_i(\mathbf{x}_i^{j+1},\mathbf{c}_i^{j+1},\mathbf{w}_i^j)\right\|^2\\
&\overset{(a)}{\leq}\tau\sum_{j=t_c}^{t}\frac{\eta_3^2}{n}\sum_{i=1}^{n}\left\|\frac{1}{n}\sum_{k=1}^{n}\nabla_{\mathbf{w}_k^j}F_k(\mathbf{x}_k^{j+1},\mathbf{c}_k^{j+1},\mathbf{w}_k^j-\nabla_{\mathbf{w}_i^j}F_i(\mathbf{x}_i^{j+1},\mathbf{c}_i^{j+1},\mathbf{w}_i^j\right\|^2 \qquad (\star 1)\\
&=\tau\sum_{j=t_c}^{t}\frac{\eta_3^2}{n}\sum_{i=1}^{n}\left[\left\|\frac{1}{n}\sum_{k=1}^{n}\Big(\nabla_{\mathbf{w}_k^j}F_k(\mathbf{x}_k^{j+1},\mathbf{c}_k^{j+1},\mathbf{w}_k^j)-\nabla_{\mathbf{w}^j}F_k(\mathbf{x}_k^{j+1},\mathbf{c}_k^{j+1},\mathbf{w}^j)\right.\right.\\
&\qquad\left.+\nabla_{\mathbf{w}^j}F_k(\mathbf{x}_k^{j+1},\mathbf{c}_k^{j+1},\mathbf{w}^j)\Big)-\nabla_{\mathbf{w}^j}F_i(\mathbf{x}_k^{j+1},\mathbf{c}_k^{j+1},\mathbf{w}^j)\right.\\
&\qquad\left.+\nabla_{\mathbf{w}^j}F_i(\mathbf{x}_k^{j+1},\mathbf{c}_k^{j+1},\mathbf{w}^j)-\nabla_{\mathbf{w}_i^j}F_i(\mathbf{x}_k^{j+1},\mathbf{c}_k^{j+1},\mathbf{w}_i^j\right\|^2\Big]\\
&\leq\tau\sum_{j=t_c}^{t_c+\tau}3\frac{\eta_3^2}{n}\sum_{i=1}^{n}\left[\left\|\frac{1}{n}\sum_{k=1}^{n}\Big(\nabla_{\mathbf{w}_k^j}F_k(\mathbf{x}_k^{j+1},\mathbf{c}_k^{j+1},\mathbf{w}_k^j)-\nabla_{\mathbf{w}^j}F_k(\mathbf{x}_k^{j+1},\mathbf{c}_k^{j+1},\mathbf{w}^j)\Big)\right\|^2\right.\\
&\qquad+\left\|\frac{1}{n}\sum_{k=1}^{n}\nabla_{\mathbf{w}^j}F_k(\mathbf{x}_k^{j+1},\mathbf{c}_k^{j+1},\mathbf{w}^j)-\nabla_{\mathbf{w}^j}F_i(\mathbf{x}_i^{j+1},\mathbf{c}_i^{j+1},\mathbf{w}^j)\right\|^2\\
&\qquad\left.+\left\|\nabla_{\mathbf{w}^j}F_i(\mathbf{x}_i^{j+1},\mathbf{c}_i^{j+1},\mathbf{w}^j)-\nabla_{\mathbf{w}_i^j}F_i(\mathbf{x}_i^{j+1},\mathbf{c}_i^{j+1},\mathbf{w}_i^j\right\|^2\right]\\
&\leq\tau\sum_{j=t_c}^{t_c+\tau}3\frac{\eta_3^2}{n}\sum_{i=1}^{n}\left[\frac{4\lambda_p^2 n}{n^2}\sum_{k=1}^{n}\|\mathbf{w}_k^j-\mathbf{w}^j\|^2+\kappa_i+4\lambda_p^2\|\mathbf{w}^j-\mathbf{w}_i^j\|^2\right]\\
&=\tau\sum_{j=t_c}^{t_c+\tau}3\eta_3^2(4\lambda_p^2\gamma_j+\kappa+4\lambda_p^2\gamma_j) \qquad (\star 2)
\end{aligned}$$

in (a) we use the facts that $\| \sum_{i=1}^{K} a_i \|^2 \leq K \sum_{i=1}^{K} \|a_i\|^2$, $t \leq \tau + t_c$ and that we are summing over non-negative terms. As a result, we have:

$$\gamma_t \leq \tau \sum_{j=t_c}^{t_c+\tau} 3\eta_3^2(8\lambda_p^2\gamma_j + \kappa)$$

$$\implies \sum_{t=t_c}^{t_c+\tau} \gamma_t \leq \sum_{t=t_c}^{t_c+\tau} \sum_{j=t_c}^{t_c+\tau} 3\tau\eta_3^2(8\lambda_p^2\gamma_j + \kappa) = 24\tau^2\eta_3^2\lambda_p^2 \sum_{j=t_c}^{t_c+\tau} \gamma_j + 3\tau^3\eta_3^2\kappa$$

Let us choose η_3 such that $24\tau^2\eta_3^2\lambda_p^2 \leq \frac{1}{2} \Leftrightarrow \eta_3 \leq \sqrt{\frac{1}{48\tau^2\lambda_p^2}}$, sum over all syncronization times, and divide both sides by T:

$$\frac{1}{T}\sum_{t=0}^{T-1} \gamma_t \leq \frac{1}{2}\sum_{j=0}^{T-1} \gamma_j + 3\tau^2\eta_3^2\kappa$$

$$\implies \frac{1}{T}\sum_{t=0}^{T-1} \gamma_t \leq 6\tau^2\eta_3^2\kappa$$

□

Let us restate and prove Corollary 4.

Corollary (Restating Corollary 4)**.** *Recall,* $\mathbf{g}^t = \frac{1}{n}\sum_{i=1}^{n} \nabla_{\mathbf{w}^t} F_i(\mathbf{x}_i^{t+1}, \mathbf{c}_i^{t+1}, \mathbf{w}_i^t)$. *Then, we have:*

$$\frac{1}{T}\sum_{t=0}^{T-1} \frac{1}{n}\sum_{i=1}^{n} \left\| \mathbf{g}^t - \nabla_{\mathbf{w}_i^t} F_i(\mathbf{x}_i^{t+1}, \mathbf{c}_i^{t+1}, \mathbf{w}_i^t) \right\|^2 \leq 36\lambda_p^2\tau^2\eta_3^2\kappa + 3\kappa$$

Proof. From ($\star$1) $\leq$ Lemma $\star$2 in the proof of (7), we have:

$$\sum_{t=t_c}^{t_c+\tau} \frac{1}{n}\sum_{i=1}^{n} \left\| \mathbf{g}^t - \nabla_{\mathbf{w}_i^t} F_i(\mathbf{x}_i^{t+1}, \mathbf{c}_i^{t+1}, \mathbf{w}_i^t) \right\|^2 \leq \sum_{j=t_c}^{t_c+\tau} 3(8\lambda_p^2\gamma_j + \kappa)$$

Summing over all t_c and dividing by T:

$$\frac{1}{T}\sum_{t=0}^{T-1} \frac{1}{n}\sum_{i=1}^{n} \left\| \mathbf{g}^t - \nabla_{\mathbf{w}_i^t} F_i(\mathbf{x}_i^{t+1}, \mathbf{c}_i^{t+1}, \mathbf{w}_i^t) \right\|^2 \leq 24\lambda_p^2 \frac{1}{T}\sum_{t=0}^{T-1} \gamma_t + 3\kappa$$

$$\overset{(a)}{\leq} 144\lambda_p^2\tau^2\eta_3^2\kappa + 3\kappa$$

where (a) is from (7).

A.2.14 Choice of η_3

Note that, in Lemma 7 we chose η_3 such that $\eta_3 \leq \frac{1}{\sqrt{48}\lambda_p\tau}$. Now, we further introduce upper bounds on η_3.

- We can choose η_3 small enough so that $L_{\min} = \eta_3 - 4\lambda_p\eta_3^2$; see (A.55) for the definition of $L_{\min}$.
- We can choose η_3 small enough so that $\eta_3 - 4\lambda_p\eta_3^2 \geq \frac{\eta_3}{2}$. This is equivalent to choosing $\eta_3 \leq \frac{1}{8\lambda_p}$.

These two choices implies $L_{\min} \geq \frac{\eta_3}{2}$.

In the end, we have 2 critical constraints on $\eta_3, \{\eta_3 : \eta_3 \leq \frac{1}{\sqrt{48}\lambda_p\tau}, \eta_3 \leq \frac{1}{8\lambda_p}\}$. Then, let $\{\eta_3 : \eta_3 \leq \frac{1}{8\lambda_p\tau}\}$. Moreover, assuming $\tau \leq \sqrt{T}$ we can take $\eta_3 = \frac{1}{8\lambda_p\sqrt{T}}$ this choice clearly satisfies the above constraints.

From (A.63) we have,

$$\begin{aligned}
\frac{1}{T}\sum_{t=0}^{T-1}\frac{1}{n}\sum_{i=1}^{n}\left\|\mathbf{G}_i^t\right\|^2 &\leq 36\frac{L_{\max}^2}{L_{\min}}\Bigg[6\tau^2\eta_3^2\kappa(\lambda_p(1+G_{Q_1}^2+G_{Q_2}^2)+28\eta_3\lambda_p^2+112\eta_3^2\lambda_p^3) \\
&\qquad + 3\eta_3\kappa + 12\lambda_p\eta_3^2\kappa + \frac{\Delta_F}{T}\Bigg] + 12\lambda_p^2((1+G_{Q_1})^2+G_{Q_2}^2)\tau^2\eta_3^2\kappa \\
&\leq 72\frac{L_{\max}^2}{\eta_3}\Bigg[6\tau^2\eta_3^2\kappa(\lambda_p(1+G_{Q_1}^2+G_{Q_2}^2)+28\eta_3\lambda_p^2+112\eta_3^2\lambda_p^3) \\
&\qquad + 3\eta_3\kappa + 12\lambda_p\eta_3^2\kappa + \frac{\Delta_F}{T}\Bigg] + 12\lambda_p^2((1+G_{Q_1})^2+G_{Q_2}^2)\tau^2\eta_3^2\kappa \\
&= 72L_{\max}^2\Bigg[6\tau^2\eta_3\kappa(\lambda_p(1+G_{Q_1}^2+G_{Q_2}^2)+28\eta_3\lambda_p^2+112\eta_3^2\lambda_p^3)
\end{aligned}$$

$$+ 3\kappa + 12\lambda_p\eta_3\kappa + \frac{\Delta_F}{\eta_3 T}\Bigg] + 12\lambda_p^2((1+G_{Q_1})^2 + G_{Q_2}^2)\tau^2\eta_3^2\kappa$$

Now, we plug in $\eta_3 = \dfrac{1}{8\lambda_p\sqrt{T}}$:

$$\begin{aligned}
&= 72L_{\max}^2\Bigg[\frac{3}{4}\tau^2\kappa\frac{(1+G_{Q_1}^2+G_{Q_2}^2)}{\sqrt{T}} + 21\tau^2\kappa\frac{1}{T} + 84\tau^2\kappa\frac{1}{T^{\frac{3}{2}}} \\
&\qquad + 3\kappa + \frac{3}{4}\frac{1}{\sqrt{T}} + \frac{8\lambda_p\Delta_F}{\sqrt{T}}\Bigg] + \frac{3}{16}((1+G_{Q_1}^2)+G_{Q_2}^2)\tau^2\kappa^2\frac{1}{T} \\
&= \frac{54L_{\max}^2\tau\kappa(1+G_{Q_1}^2+G_{Q_2}^2) + 54L_{\max}^2 + 576L_{\max}^2\lambda_p\Delta_F}{\sqrt{T}} \\
&\qquad + \frac{1512L_{\max}^2\tau^2\kappa + \frac{3}{16}\tau^2\kappa^2((1+G_{Q_1}^2)+G_{Q_2}^2)}{T} \\
&\qquad + \frac{6048L_{\max}^2\tau^2\kappa}{T^{\frac{3}{2}}} + 216L_{\max}^2\kappa
\end{aligned}$$

where $L_{\max}$ is defined in (A.61).

This completes the proof of Theorem 17. □

A.2.15 Additional Details and Results for Experiments

In this section, we first discuss the implementation details for the `prox` steps for Algorithm 1 and Algorithm 4 in Section A.2.16. Section A.2.17 discusses implementation details for the algorithms along with hyperparameters which was omitted in Section 7.5 of the main paper due to space constraints. Section A.2.18 provides additional experiments for comparison of QuPeD (Algorithm 4) with other personalized learning schemes with additional types of heterogeneity and datasets. Section A.2.19 discusses the hardware resources on which the algorithms were implemented and the training times.

A.2.16 Proximal Updates

For the implementation of Algorithm 1,4, we consider ℓ_1-loss for the distance function $R(\mathbf{x}, \mathbf{c})$. In other words, $R(\mathbf{x}, \mathbf{c}) = \min\{\frac{1}{2}\|\mathbf{z} - \mathbf{x}\|_1 : z_i \in \{c_1, \cdots, c_m\}, \forall i\}$. For simplicity, we define

$\mathcal{C} = \{\mathbf{z} : z_i \in \{c_1, \cdots, c_m\}, \forall i\}$. For the first type of update (update of $\mathbf{x}$) we have:

$$\begin{aligned}
\text{prox}_{\eta_1 \lambda R(\cdot, \mathbf{c})}(\mathbf{y}) &= \arg\min_{\mathbf{x} \in \mathbb{R}^d} \left\{ \frac{1}{2\eta_1} \|\mathbf{x} - \mathbf{y}\|_2^2 + \lambda R(\mathbf{x}, \mathbf{c}) \right\} \\
&= \arg\min_{\mathbf{x} \in \mathbb{R}^d} \left\{ \frac{1}{2\eta_1} \|\mathbf{x} - \mathbf{y}\|_2^2 + \frac{\lambda}{2} \min_{\mathbf{z} \in \mathcal{C}} \|\mathbf{z} - \mathbf{x}\|_1 \right\} \\
&= \arg\min_{\mathbf{x} \in \mathbb{R}^d} \min_{\mathbf{z} \in \mathcal{C}} \left\{ \frac{1}{2\eta_1} \|\mathbf{x} - \mathbf{y}\|_2^2 + \frac{\lambda}{2} \|\mathbf{z} - \mathbf{x}\|_1 \right\}
\end{aligned} \tag{A.65}$$

This corresponds to solving:

$$\min_{\mathbf{z} \in \mathcal{C}} \min_{\mathbf{x} \in \mathbb{R}^d} \left\{ \frac{1}{\eta_1} \|\mathbf{x} - \mathbf{y}\|_2^2 + \lambda \|\mathbf{z} - \mathbf{x}\|_1 \right\}$$

Since both ℓ_1 and squared ℓ_2 norms are decomposable; if we fix $\mathbf{z}$, for the inner problem we have the following solution to soft thresholding:

$$x^\star(\mathbf{z})_i = \begin{cases} y_i - \frac{\lambda \eta_1}{2}, & \text{if } y_i - \frac{\lambda \eta_1}{2} > z_i \\ y_i + \frac{\lambda \eta_1}{2}, & \text{if } y_i + \frac{\lambda \eta_1}{2} < z_i \\ z_i, & \text{otherwise} \end{cases} \tag{A.66}$$

As a result we have:

$$\min_{\mathbf{z} \in \mathcal{C}} \left\{ \frac{1}{\eta_1} \|x^\star(\mathbf{z}) - \mathbf{y}\|_2^2 + \lambda \|\mathbf{z} - x^\star(\mathbf{z})\|_1 \right\}$$

This problem is separable, in other words we have:

$$\mathbf{z}_i^\star = \arg\min_{z_i \in \{c_1, \cdots, c_m\}} \left\{ \frac{1}{\eta_1} (x^\star(\mathbf{z})_i - y_i)^2 + \lambda |z_i - x^\star(\mathbf{z})_i| \right\} \ \forall i$$

Substituting $x^\star(\mathbf{z})_i$ and solving for z_i gives us:

$$\mathbf{z}_i^\star = \arg\min_{z_i \in \{c_1, \cdots, c_m\}} \{|z_i - y_i|\} \ \forall i$$

Or equivalently we have,

$$\mathbf{z}^\star = \arg\min_{\mathbf{z}\in\mathcal{C}} \|\mathbf{z}-\mathbf{y}\|_1 = Q_{\mathbf{c}}(\mathbf{y}) \tag{A.67}$$

As a result, $\text{prox}_{\eta_1\lambda R(\cdot,\mathbf{c})}(\cdot)$ becomes the soft thresholding operator:

$$\text{prox}_{\eta_1\lambda R(\cdot,\mathbf{c})}(\mathbf{y})_i = \begin{cases} y_i - \frac{\lambda\eta_1}{2}, & \text{if } y_i \geq Q_{\mathbf{c}}(\mathbf{y})_i + \frac{\lambda\eta_1}{2} \\ y_i + \frac{\lambda\eta_1}{2}, & \text{if } y_i \leq Q_{\mathbf{c}}(\mathbf{y})_i - \frac{\lambda\eta_1}{2} \\ Q_{\mathbf{c}}(\mathbf{y})_i, & \text{otherwise} \end{cases} \tag{A.68}$$

And for the second type of update we have $\text{prox}_{\eta_2\lambda R(\mathbf{x},\cdot)}(\cdot)$ becomes:

$$\begin{aligned} \text{prox}_{\eta_2\lambda R(\mathbf{x},\cdot)}(\boldsymbol{\mu}) &= \arg\min_{\mathbf{c}\in\mathbb{R}^m} \left\{ \frac{1}{2\eta_2}\|\mathbf{c}-\boldsymbol{\mu}\|_2^2 + \lambda R(\mathbf{x},\mathbf{c}) \right\} \\ &= \arg\min_{\mathbf{c}\in\mathbb{R}^m} \left\{ \frac{1}{2\eta_2}\|\mathbf{c}-\boldsymbol{\mu}\|_2^2 + \frac{\lambda}{2}\min_{\mathbf{z}\in C}\|\mathbf{z}-\mathbf{x}\|_1 \right\} \\ &= \arg\min_{\mathbf{c}\in\mathbb{R}^m} \left\{ \frac{1}{2\eta_2}\|\mathbf{c}-\boldsymbol{\mu}\|_2^2 + \frac{\lambda}{2}\|Q_{\mathbf{c}}(\mathbf{x})-\mathbf{x}\|_1 \right\} \end{aligned} \tag{A.69}$$

Then,

$$\begin{aligned} \text{prox}_{\eta_2\lambda R(\mathbf{x},\cdot)}(\boldsymbol{\mu})_j &= \arg\min_{\mathbf{c}_j\in\mathbb{R}^m} \left\{ \frac{1}{2\eta_2}(c_j-\mu_j)^2 + \frac{\lambda}{2}\sum_{i=1}^{d}|Q_{\mathbf{c}}(\mathbf{x})_i - x_i| \right\} \\ &= \arg\min_{\mathbf{c}_j\in\mathbb{R}^m} \left\{ \frac{1}{2\eta_2}(c_j-\mu_j)^2 + \frac{\lambda}{2}\sum_{i=1}^{d}\mathbb{1}(Q_{\mathbf{c}}(\mathbf{x})_i = c_j)|c_j - x_i| \right\} \end{aligned}$$

We remark that the second term of the optimization problem is hard to solve; in particular we need to know the assignments of x_i to c_j. In the algorithm, at each time point t, we are given the previous epoch's assignments. We can utilize that and approximate the optimization problem by assuming $\mathbf{c}^{t+1}$ will be in a neighborhood of $\mathbf{c}^t$. We can take the gradient of $R(\mathbf{x}^{t+1},\mathbf{c})$ at $\mathbf{c}=\mathbf{c}^t$ while finding the optimal point. This is also equivalent to optimizing the first order Taylor approximation around $\mathbf{c}=\mathbf{c}^t$. As a result we have the following optimization problem:

$$\text{prox}_{\eta_2\lambda R(\mathbf{x}^{t+1},\cdot)}(\boldsymbol{\mu})_j \approx \arg\min_{\mathbf{c}_j\in\mathbb{R}^m} \left\{ \frac{1}{2\eta_2}(c_j-\mu_j)^2 + \frac{\lambda}{2}\sum_{i=1}^{d}\mathbb{1}(Q_{\mathbf{c}^t}(\mathbf{x}^{t+1})_i = c_j^t)|c_j^t - x_i^{t+1}| \right.$$

$$+(c_j - c_j^t)\frac{\lambda}{2}\sum_{i=1}^{d}\mathbb{1}(Q_{\mathbf{c}^t}(\mathbf{x}^{t+1})_i = c_j^t)\frac{\partial|c_j^t - x_i^{t+1}|}{\partial c_j^t}\Bigg\}$$

In our implementation, we take $\frac{\partial|c_j^t - x_i^{t+1}|}{\partial c_j^t}$ as 1 if $c_j^t > x_i^{t+1}$, -1 if $c_j^t < x_i^{t+1}$ and 0 otherwise. Now taking the derivative with respect to c_j and setting it to 0 gives us:

$$\begin{aligned}\text{prox}_{\eta_2\lambda R(\mathbf{x}^{t+1},\cdot)}(\boldsymbol{\mu})_j \approx \mu_j - \frac{\lambda\eta_2}{2}(&\sum_{i=1}^{d}\mathbb{1}(Q_{\mathbf{c}^t}(\mathbf{x}^{t+1})_i = c_j^t)\mathbb{1}(x_i^{t+1} > c_j^t)\\ &-\sum_{i=1}^{d}\mathbb{1}(Q_{\mathbf{c}^t}(\mathbf{x}^{t+1})_i = c_j^t)\mathbb{1}(x_i^{t+1} < c_j^t))\end{aligned}$$

Proximal map pulls the updated centers toward the median of the weights that are assigned to them.

Using $P \to \infty$. In the experiments we observed that using $P \to \infty$, i.e. using hard quantization function produces good results and also simplifies the implementation. The implications of $P \to \infty$ are as follows:

- We take $\nabla_{\mathbf{x}} f(\widetilde{Q}_{\mathbf{c}}(\mathbf{x})) = 0$ and $\nabla_{\mathbf{x}} f^{KD}(\widetilde{Q}_{\mathbf{c}}(\mathbf{x}), \mathbf{w}) = 0$.
- We take $\nabla_{\mathbf{c}} f(\widetilde{Q}_{\mathbf{c}}(\mathbf{x})) = \nabla_{\mathbf{c}} f(Q_{\mathbf{c}}(\mathbf{x})) = \begin{bmatrix}\sum_{i=1}^{d}\frac{\partial f(Q_{\mathbf{c}}(\mathbf{x}))}{\partial Q_{\mathbf{c}}(\mathbf{x})_i}\mathbb{1}(Q_{\mathbf{c}}(\mathbf{x})_i = c_1)\\ \vdots \\ \sum_{i=1}^{d}\frac{\partial f(Q_{\mathbf{c}}(\mathbf{x}))}{\partial Q_{\mathbf{c}}(\mathbf{x})_i}\mathbb{1}(Q_{\mathbf{c}}(\mathbf{x})_i = c_m)\end{bmatrix}$ and

$$\nabla_{\mathbf{c}} f^{KD}(\widetilde{Q}_{\mathbf{c}}(\mathbf{x}), \mathbf{w}) = \nabla_{\mathbf{c}} f^{KD}(Q_{\mathbf{c}}(\mathbf{x}), \mathbf{w}) = \begin{bmatrix}\sum_{i=1}^{d}\frac{\partial f^{KD}(Q_{\mathbf{c}}(\mathbf{x}),\mathbf{w})}{\partial Q_{\mathbf{c}}(\mathbf{x})_i}\mathbb{1}(Q_{\mathbf{c}}(\mathbf{x})_i = c_1)\\ \vdots \\ \sum_{i=1}^{d}\frac{\partial f^{KD}(Q_{\mathbf{c}}(\mathbf{x}),\mathbf{w})}{\partial Q_{\mathbf{c}}(\mathbf{x})_i}\mathbb{1}(Q_{\mathbf{c}}(\mathbf{x})_i = c_m)\end{bmatrix}.$$

A.2.17 Implementation Details and Hyperparameters

In this section we discuss the implementation details and hyperparameters used for the algorithms considered in Section 7.5 of our main paper.

Fine tuning. In both centralized and federated settings we employ a fine tuning procedure similar to [7]. At the end of the regular training procedure, model weights are hard-quantized.

After the hard-quantization, during the fine tuning epochs we let the unquantized parts of the network to continue training (e.g. batch normalization layers) and different from [7] we also continue to train quantization levels.

A.2.17.1 Centralized Setting

For centralized training, we use CIFAR-10 dataset and train a ResNet [66] model following [7] and [177]. We employ ADAM with learning rate 0.01 and no weight decay. We choose $\lambda(t) = 10^{-4}t$. For the implementation of ResNet models we used a toolbox[1]. In Table 2.1 we reported the results from [177] directly and implemented ProxQuant using their published code[2]. We use a learning schedule for η_2, particularly, we start with $\eta_2 = 10^{-4}$ and multiply it with 0.1 at epochs 80 and 140.

A.2.17.2 Federated Setting

For each of the methods we tuned the local step learning rate separately on the set $\{0.2, 0.15, 0.125, 0.1, 0.075, 0.05\}$. We observed that except for the two cases, for all other cases, 0.1 was the best choice for the learning rate in terms of accuracy: The two exceptions are the local training methods on FEMNIST and Per-FedAvg on CIFAR-10, for which, respectively, 0.075 and 0.125 were the best choices for the learning rate.

- QuPeD[3]: For CNN1 we choose $\lambda_p = 0.25$, $\lambda(t) = 10^{-6}t$ for 2Bits and $\lambda = 5 \times 10^{-7}t\frac{1}{0.99^t}$ for 1Bit training on CIFAR-10. On FEMNIST [4] and MNIST we choose $\lambda(t) = 5 \times 10^{-6}t$ for 2Bits and $\lambda = 10^{-6}t\frac{1}{0.99^t}$ for 1Bit training. For CNN2 we use $\lambda_p = 0.15$. Global model has the same learning schedule as the personalized models. Furthermore, we use $\eta_2 = 10^{-4}$.

[1] https://github.com/akamaster/pytorch_resnet_cifar10

[2] https://github.com/allenbai01/ProxQuant

[3] For federated experiments we have used Pytorch's Distributed package.

[4] We use https://github.com/tao-shen/FEMNIST_pytorch to import FEMNIST dataset.

QuPeL: We used $\lambda_p = 0.2$, $\eta_3 = 0.5$ (same as pFedMe [42]) and took λ values from QuPeD.

- Per-FedAvg [48] and pFedMe [42]:To implement Per-FedAvg, we used the same learning rate as mentioned in Section 7.5, schedule for main learning rate and $\alpha = 0.001$ for CNN1 and $\alpha = 2.5 \times 10^{-3}$ for CNN2 (we tuned in the interval $[8 \times 10^{-4}, 5 \times 10^{-3}]$), for the auxiliary learning rate. For pFedMe we used the same learning rate schedule for main learning rate, $K = 5$ for the number of local iterations; and we used $\lambda = 0.5$, $\eta = 0.2$ for CNN1 and $\lambda = 0.2$, $\eta = 0.15$ for CNN2 (we tuned in the interval $[0.1, 1]$ for both parameters).
- Federated Mutual Learning [152]: Since authors do not discuss the hyperparameters in the paper, we used $\alpha = \beta = 0.25$ for CNN1 and $\alpha = \beta = 0.15$ for CNN2, similar to our use of λ_p in QuPeD. Global model has the same learning schedule as the personalized models.

For QuPeD and Federated ML we used CNN1 as the global model in all settings. For the other methods where global and personalized models cannot be different we used the same structure as personalized models.

A.2.18 Additional Results for Federated Setting

In this section we provide additional experimental results for comparison of QuPeD with other pearsonalized learning schems from literature.

Comparison on another CNN architecture (CNN2). We first report experimental results on CIFAR-10 for CNN2 in Table A.1 (with the same setting we have for Table 2.1). This is a deeper architecture than CNN1, as described in Section 7.5 in the main paper.

For the results in Table A.1, it can be seen that the comments made for Table 2.1 in the main paper directly hold as QuPeD is able to outperform other schemes by a significant margin. This demonstrates that QuPeD also works for a deeper neural network (than CNN1 considered in the main paper).

Table A.1 Test accuracy (in %) for CNN2 model at all clients, CIFAR-10.

Method	Test Accuracy in %
FedAvg (FP)	62.49 ± 0.42
Local Training (FP)	73.86 ± 0.22
Local Training (2 Bits)	73.24 ± 0.14
Local Training (1 Bit)	70.23 ± 0.10
QuPeD (FP)	$\mathbf{76.39} \pm 0.36$
QuPeD (2 Bits)	75.32 ± 0.18
QuPeD (1 Bit)	72.01 ± 0.31
PFedMe (FP) [42]	74.70 ± 0.10
Per-FedAvg(FP) [48]	74.60 ± 0.48
Federated Mutual Learning(FP) [152]	75.74 ± 0.56

Another Type of Data Heterogeneity. We report results for another data heterogeneity setting where each client has access to data samples from random 3 classes on CIFAR-10. Sampling data from 3 random classes per client is a more challenging setting compared 4 classes per client considered in Section 7.5. In Table A.2 we see that FedAvg's performance further decreased due to increased heterogeneity. Moreover, most of the other personalized FL methods are outperformed by local training whereas QuPeD still performs better than local training, and other personalized FL methods. We observe that QuPeD with 2 Bits aggressive quantization outperforms all the other competing methods except Federated ML [152] (for which it shows a similar accuracy). Moreover, QuPeD (1Bit) is able to outperform Per-FedAvg.

Importance of updating the centers. In our proposed schemes: Algorithm 4, we optimize over both the quantization levels and the model parameters. We compare performance of our proposed scheme with the case when we only optimize over model parameters and not quantization levels in Table A.3. As seen from the results in the table, having the center

Table A.2 Test accuracy (in %) for CNN1 model at all clients, with 3 classes accessed per client on CIFAR-10.

Method	Test Accuracy (in %)
FedAvg (FP)	59.23 ± 0.25
Local Training (FP)	78.03 ± 0.59
Local Training (2 Bits)	77.47 ± 0.64
Local Training (1 Bit)	75.89 ± 0.66
QuPeD (FP)	$\mathbf{80.30} \pm 0.60$
QuPeD (2 Bits)	79.31 ± 0.74
QuPeD (1 Bit)	77.23 ± 0.58
QuPeL (2 Bits)	77.87 ± 0.53
QuPeL (1 Bits)	74.46 ± 0.73
pFedMe (FP) [42]	78.22 ± 0.91
Per-FedAvg (FP) [48]	75.08 ± 0.39
Federated ML (FP) [152]	79.44 ± 0.82

updates in the optimization problem is critical, particularly, for the 1Bit quantization case for which we observe an increase in the performance by 4%.

Results on MNIST. We now provide additional results on MNIST dataset to compared QuPeD with other competing schemes. We consider 50 clients in total, where each client samples data from 3 or 4 random classes and uses CNN1. We train for a total of 50 epochs, for quantized training we allocate the last 7 epochs for finetuning.

QuPeD (FP) outperforms all methods except Per-FedAvg on MNIST when clients sample data from 4 random classes. The difference is almost negligible (0.04%). As we can observe in Table A.4 with the increased heterogeneity QuPeD starts to outperform Per-FedAvg by a 0.20% margin. Moreover, we observe QuPeD with 2Bit quantization also outperforms

Table A.3 Test accuracy (in %) comparison between the cases with and without center updates for CNN1 model at all clients, 4 classes accessed per client on FEMNIST.

Method	Test Accuracy (in %)
QuPeD (FP)	**97.31** $\pm$ 0.12
QuPeD (2 Bits)	96.73 $\pm$ 0.27
QuPeD (1 Bit)	95.15 $\pm$ 0.21
QuPeD (2 Bits) no center updates	96.48 $\pm$ 0.10
QuPeD (1 Bit) no center updates	91.17 $\pm$ 0.58

Per-FedAvg.

Text classification task on AG News Dataset. To show that our method can also be applied for tasks different than vision tasks. text classification problem using the AG News dataset (available at https://pytorch.org/text/stable/datasets.html). We used half of the dataset to make the training procedure more challenging. We used EmbeddingBag structure available at https://pytorch.org/tutorials/beginner/text_sentiment_ngrams_tutorial.html and distributed the data such that each of the 42 clients has access to samples from 3 out of 4 classes. The results we obtained are provided in Table A.5

These results demonstrate the effectiveness of QuPeD on text data in comparison with local training.

A.2.19 Computational Resources

For our experiments, we used a server which has 6 Nvidia RTX2080Ti GPU's and Intel Xeon Gold 6230 CPU @ 2.10GHz CPU's. The longest epoch time is 125 seconds with QuPeD (2Bit) training on CIFAR-10. Our code uses a maximum of 9GB memory per GPU.

Table A.4 Test accuracy (in %) for CNN1 model at all clients, on MNIST.

Method	3 classes per client	4 classes per client
FedAvg (FP)	98.64 ± 0.10	98.65 ± 0.09
Local Training (FP)	98.79 ± 0.03	98.66 ± 0.15
Local Training (2 Bits)	98.53 ± 0.07	98.37 ± 0.11
Local Training (1 Bit)	98.41 ± 0.02	97.95 ± 0.20
QuPeD (FP)	$\mathbf{99.05} \pm 0.10$	98.89 ± 0.11
QuPeD (2 Bits)	98.96 ± 0.13	98.67 ± 0.18
QuPeD (1 Bit)	98.57 ± 0.08	98.25 ± 0.16
QuPeL (2 Bits)	98.95 ± 0.12	98.61 ± 0.19
QuPeL (1 Bits)	98.33 ± 0.14	98.11 ± 0.26
pFedMe (FP) [42]	98.98 ± 0.05	98.82 ± 0.15
Per-FedAvg (FP) [48]	98.82 ± 0.05	$\mathbf{98.93} \pm 0.09$
Federated ML (FP) [152]	99.00 ± 0.06	98.84 ± 0.13

Table A.5 Test accuracy (in %) for Embedding Bag model at all clients, on AG News.

Method	
FedAvg (FP)	83.04 ± 0.60
Local training (FP)	84.20 ± 1.53
Local training (2 Bits)	82.68 ± 0.34
Local training (1 Bit)	82.12 ± 2.17
QuPeD (FP)	$\mathbf{85.06} \pm 1.07$
QuPeD (2 Bits)	83.62 ± 0.50
QuPeD (1 Bit)	82.72 ± 1.20

Algorithm 9 QuPeL: Quantized Personalization

Input: Regularization parameters λ, λ_p; synchronization gap τ; for each client $i \in [n]$, initialize full precision personalized model $\mathbf{x}_i^0$, quantization centers $\mathbf{c}_i^0$, and local model $\mathbf{w}_i^0$; a penalty function enforcing quantization $R(\mathbf{x}, \mathbf{c})$; a soft quantizer $\widetilde{Q}_{\mathbf{c}}(\mathbf{x})$; and learning rates η_1, η_2, η_3.

for $t = 0$ **to** $T-1$ **do**
  On Clients $i = 1$ **to** n (in parallel) **do**:
  if τ does not divide t **then**
    Compute $\mathbf{g}_i^t := \nabla_{\mathbf{x}_i^t} f_i(\mathbf{x}_i^t) + \nabla_{\mathbf{x}_i^t} f_i(\widetilde{Q}_{\mathbf{c}_i^t}(\mathbf{x}_i^t)) + \lambda_p(\mathbf{x}_i^t - \mathbf{w}_i^t) + \lambda_p \nabla_{\mathbf{x}_i^t} \widetilde{Q}_{\mathbf{c}_i^t}(\mathbf{x}_i^t)(\widetilde{Q}_{\mathbf{c}_i^t}(\mathbf{x}_i^t) - \mathbf{w}_i^t)$
    $\mathbf{x}_i^{t+1} = \mathrm{prox}_{\eta_1 \lambda R_{\mathbf{c}_i^t}}(\mathbf{x}_i^t - \eta_1 \mathbf{g}_i^t)$
    Compute $\mathbf{h}_i^t := \nabla_{\mathbf{c}_i^t} f_i(\widetilde{Q}_{\mathbf{c}_i^t}(\mathbf{x}_i^{t+1})) + \lambda_p \nabla_{\mathbf{c}_i^{t+1}} \widetilde{Q}_{\mathbf{c}_i^t}(\mathbf{x}_i^{t+1})(\widetilde{Q}_{\mathbf{c}_i^t}(\mathbf{x}_i^{t+1}) - \mathbf{w}_i^t)$
    $\mathbf{c}_i^{t+1} = \mathrm{prox}_{\eta_2 \lambda R_{\mathbf{x}_i^{t+1}}}(\mathbf{c}_i^t - \eta_2 \mathbf{h}_i^t)$
    $\mathbf{w}_i^{t+1} = \mathbf{w}_i^t - \eta_3 \lambda_p((\mathbf{w}_i^t - \mathbf{x}_i^{t+1}) + (\mathbf{w}_i^t - \widetilde{Q}_{\mathbf{c}_i^{t+1}}(\mathbf{x}_i^{t+1}))$
  else
    Send $\mathbf{w}_i^t$ to **Server**
    Receive $\mathbf{w}^t$ from **Server** and set $\mathbf{w}_i^{t+1} = \mathbf{w}^t$
  end if
  On Server do:
  if τ divides t **then**
    Receive $\{\mathbf{w}_i^t\}_{i=1}^n$ and compute $\mathbf{w}^t := \frac{1}{n} \sum_{i=1}^n \mathbf{w}_i^t$
    Broadcast $\mathbf{w}^t$ to all **Clients**
  end if
end for
$\hat{\mathbf{x}}_i^T = Q_{\mathbf{c}_i^T}(\mathbf{x}_i^T)$ for all $i \in [n]$

Output: Quantized personalized models $\hat{\mathbf{x}}_i^T$ for $i \in [n]$

APPENDIX B

Appendix of Chapter 3

B.1 Preliminaries

We give standard privacy definitions that we use in Section B.2, some existing results on RDP to DP conversion and RDP composition in Section B.2.1, and user-level differential privacy in Section B.2.2.

B.2 Privacy Definitions

In this subsection, we define different privacy notions that we will use in this paper: local differential privacy (LDP), central different privacy (DP), and Renyi differential privacy (RDP), and their user-level counterparts.

Definition 1 (Local Differential Privacy - LDP [86]). *For $\epsilon_0 \geq 0$, a randomized mechanism $\mathcal{R} : \mathcal{X} \to \mathcal{Y}$ is said to be ϵ_0-local differentially private (in short, ϵ_0-LDP), if for every pair of inputs $d, d' \in \mathcal{X}$, we have*

$$\Pr[\mathcal{R}(d) \in \mathcal{S}] \leq e^{\epsilon_0} \Pr[\mathcal{R}(d') \in \mathcal{S}], \qquad \forall \mathcal{S} \subset \mathcal{Y}. \tag{B.1}$$

Let $\mathcal{D} = \{x_1, \ldots, x_n\}$ denote a dataset comprising n points from $\mathcal{X}$. We say that two datasets $\mathcal{D} = \{x_1, \ldots, x_n\}$ and $\mathcal{D}' = \{x'_1, \ldots, x'_n\}$ are neighboring (and denoted by $\mathcal{D} \sim \mathcal{D}'$) if they differ in one data point, i.e., there exists an $i \in [n]$ such that $x_i \neq x'_i$ and for every $j \in [n], j \neq i$, we have $x_j = x'_j$.

Definition 2 (Central Differential Privacy - DP [45, 46]). *For $\epsilon, \delta \geq 0$, a randomized mechanism $\mathcal{M} : \mathcal{X}^n \to \mathcal{Y}$ is said to be (ϵ, δ)-differentially private (in short, (ϵ, δ)-DP), if for all neighboring datasets $\mathcal{D} \sim \mathcal{D}' \in \mathcal{X}^n$ and every subset $\mathcal{S} \subseteq \mathcal{Y}$, we have*

$$\Pr\left[\mathcal{M}(\mathcal{D}) \in \mathcal{S}\right] \leq e^{\epsilon_0} \Pr\left[\mathcal{M}(\mathcal{D}') \in \mathcal{S}\right] + \delta. \tag{B.2}$$

If $\delta = 0$, then the privacy is referred to as pure DP.

Definition 3 ($(\lambda, \epsilon(\lambda))$-RDP (Renyi Differential Privacy) [121]). *A randomized mechanism $\mathcal{M} : \mathcal{X}^n \to \mathcal{Y}$ is said to have $\epsilon(\lambda)$-Renyi differential privacy of order $\lambda \in (1, \infty)$ (in short, $(\lambda, \epsilon(\lambda))$-RDP), if for any neighboring datasets $\mathcal{D} \sim \mathcal{D}' \in \mathcal{X}^n$, the Renyi divergence between $\mathcal{M}(\mathcal{D})$ and $\mathcal{M}(\mathcal{D}')$ is upper-bounded by $\epsilon(\lambda)$, i.e.,*

$$\begin{aligned} D_\lambda(\mathcal{M}(\mathcal{D})||\mathcal{M}(\mathcal{D}')) &= \frac{1}{\lambda - 1} \log \left(\mathbb{E}_{\theta \sim \mathcal{M}(\mathcal{D}')} \left[\left(\frac{\mathcal{M}(\mathcal{D})(\theta)}{\mathcal{M}(\mathcal{D}')(\theta)} \right)^\lambda \right] \right) \\ &\leq \epsilon(\lambda), \end{aligned}$$

where $\mathcal{M}(\mathcal{D})(\theta)$ denotes the probability that $\mathcal{M}$ on input $\mathcal{D}$ generates the output θ. For convenience, instead of $\epsilon(\lambda)$ being an upper bound, we define it as $\epsilon(\lambda) = \sup_{\mathcal{D} \sim \mathcal{D}'} D_\lambda(\mathcal{M}(\mathcal{D})||\mathcal{M}(\mathcal{D}'))$.

B.2.1 RDP to DP Conversion and RDP Composition

As mentioned after Theorem 7, we can convert the RDP guarantees of `DP-AdaPeD` to its DP guarantees using existing conversion results from literature. To the best of our knowledge, the following gives the best conversion.

Lemma 8 (From RDP to DP [8, 22]). *Suppose for any $\lambda > 1$, a mechanism $\mathcal{M}$ is $(\lambda, \epsilon(\lambda))$-RDP. Then, the mechanism $\mathcal{M}$ is (ϵ, δ)-DP, where ϵ, δ are define below:*

For a given $\delta \in (0, 1)$:

$$\epsilon = \min_\lambda \epsilon(\lambda) + \frac{\log(1/\delta) + (\lambda - 1)\log(1 - 1/\lambda) - \log(\lambda)}{\lambda - 1}$$

For a given $\epsilon > 0$:

$$\delta = \min_\lambda \frac{\exp\left((\lambda - 1)\left(\epsilon(\lambda) - \epsilon\right)\right)}{\lambda - 1} \left(1 - \frac{1}{\lambda}\right)^\lambda.$$

The main strength of RDP in comparison to other privacy notions comes from composition. The following result states that if we adaptively compose two RDP mechanisms with the same order, their privacy parameters add up in the resulting mechanism.

Lemma 9 (Adaptive composition of RDP [121, Proposition 1]). *For any $\lambda > 1$, let $\mathcal{M}_1 : \mathcal{X} \to \mathcal{Y}_1$ be a $(\lambda, \epsilon_1(\lambda))$-RDP mechanism and $\mathcal{M}_2 : \mathcal{Y}_1 \times \mathcal{X} \to \mathcal{Y}$ be a $(\lambda, \epsilon_2(\lambda))$-RDP mechanism. Then, the mechanism defined by $(\mathcal{M}_1, \mathcal{M}_2)$ satisfies $(\lambda, \epsilon_1(\lambda) + \epsilon_2(\lambda))$-RDP.*

B.2.2 User-level Differential Privacy [95]

Consider a set of m users, each having a local dataset of n samples. Let $\mathcal{D}_i = \{x_{i1}, \ldots, x_{in}\}$ denote the local dataset at the i-th user for $i \in [m]$, where $x_{ij} \in \mathcal{X}$ and $\mathcal{X} \subset \mathbb{R}^d$. We define $\mathcal{D} = (\mathcal{D}_1, \ldots, \mathcal{D}_m) \in (\mathcal{X}^n)^m$ as the entire dataset.

We have already defined DP, LDP, and RDP in Section B.2 w.r.t. the item-level privacy. Here, we extend those definition w.r.t. the user-level privacy. In order to do that, we need a generic neighborhood relation between datasets: We say that two datasets $\mathcal{D}$, $\mathcal{D}'$ are neighboring with respect to distance metric dis if we have $\mathsf{dis}(\mathcal{D}, \mathcal{D}') \leq 1$.

Item-level DP/RDP vs. User-level DP/RDP. By choosing $\mathsf{dis}(\mathcal{D}, \mathcal{D}') = \sum_{i=1}^{m} \sum_{j=1}^{n} \mathbb{1}\{x_{ij} \neq x'_{ij}\}$, we recover the standard definition of the DP/RDP from Definitions 2, 3, which we call *item-level* DP/RDP. In the item-level DP/RDP, two datasets $\mathcal{D}$, $\mathcal{D}'$ are neighboring if they differ in a single item. On the other hand, by choosing $\mathsf{dis}(\mathcal{D}, \mathcal{D}') = \sum_{i=1}^{m} \mathbb{1}\{\mathcal{D}_i \neq \mathcal{D}'_i\}$, we call it *user-level* DP/RDP, where two datasets $\mathcal{D}, \mathcal{D}' \in (\mathcal{X}^n)^m$ are neighboring when they differ in a local dataset of any single user. Observe that when each user has a single item ($n = 1$), then both item-level and user-level privacy are equivalent.

User-level Local Differential Privacy (LDP). When we have a single user (i.e., $m = 1$ and $\mathcal{D} = \mathcal{X}^n$), by choosing $\mathsf{dis}\,(\mathcal{D}, \mathcal{D}') = \mathbb{1}\{\mathcal{D} \neq \mathcal{D}'\}$ for $\mathcal{D}, \mathcal{D}' \in \mathcal{X}^n$, we call it *user-level LDP*. In this case each user privatize her own local dataset using a private mechanism.

We can define user-level LDP/DP/RDP analogously to their item-level counterparts using

the neighborhood relation dis defined above.

B.3 Personalized Estimation – Gaussian Model

B.3.1 Proof of Theorem 3

We will derive the optimal estimator and prove the MSE for one dimensional case, i.e., for $d = 1$; the final result can be obtained by applying these to each of the d coordinates separately.

The posterior estimators of the local means $\theta_1, \ldots, \theta_m$ and the global mean μ are obtained by solving the following optimization problem:

$$\begin{aligned}
\hat{\theta}_1, \ldots, \hat{\theta}_m, \hat{\mu} &= \underset{\theta_1,\ldots,\theta_m,\mu}{\arg\max}\, p_{\mathbf{X}|\theta}\left(X_1, \ldots, X_m|\theta_1, \ldots, \theta_m\right) p_{\theta|\mu}(\theta_1, \ldots, \theta_m|\mu) \\
&= \underset{\theta_1,\ldots,\theta_m,\mu}{\arg\min} -\log\left(p_{\mathbf{X}|\theta}\left(X_1, \ldots, X_m|\theta_1, \ldots, \theta_m\right)\right) - \log\left(p_{\theta|\mu}(\theta_1, \ldots, \theta_m|\mu)\right) \\
&= \underset{\theta_1,\ldots,\theta_m,\mu}{\arg\min}\, C + \sum_{i=1}^{m}\sum_{j=1}^{n} \frac{\left(X_i^j - \theta_i\right)^2}{\sigma_x^2} + \sum_{i=1}^{m} \frac{\left(\theta_i - \mu\right)^2}{\sigma_\theta^2},
\end{aligned}$$

where the second equality is obtained from the fact that the log function is a monotonic function, and C is a constant independent of the variables $\theta = (\theta_1, \ldots, \theta_m)$. Observe that the objective function $F(\theta, \mu) = \sum_{i=1}^{m}\sum_{j=1}^{n} \frac{\left(X_i^j - \theta_i\right)^2}{\sigma_x^2} + \sum_{i=1}^{m} \frac{(\theta_i - \mu)^2}{\sigma_\theta^2}$ is jointly convex in (θ, μ). Thus, the optimal is obtained by setting the derivative to zero as it is an unbounded optimization problem.

$$\begin{aligned}
\left.\frac{\partial F}{\partial \theta_i}\right|_{\mu=\hat{\mu},\theta_i=\hat{\theta}_i} &= \frac{\sum_{j=1}^{n} 2(\hat{\theta}_i - X_i^j)}{\sigma_x^2} + \frac{2(\hat{\theta}_i - \hat{\mu})}{\sigma_\theta^2} = 0, \qquad \forall i \in [m] \\
\left.\frac{\partial F}{\partial \mu}\right|_{\mu=\hat{\mu},\theta_i=\hat{\theta}_i} &= \frac{\sum_{i=1}^{m} 2(\hat{\mu} - \hat{\theta}_i)}{\sigma_\theta^2} = 0.
\end{aligned}$$

By solving these $m+1$ equations in $m+1$ unknowns, we get:

$$\hat{\theta}_i = \alpha \left(\frac{1}{n} \sum_{j=1}^{n} X_i^j \right) + (1-\alpha) \left(\frac{1}{mn} \sum_{i=1}^{m} \sum_{j=1}^{n} X_i^j \right), \tag{B.3}$$

where $\alpha = \frac{\sigma_\theta^2}{\sigma_\theta^2 + \frac{\sigma_x^2}{n}}$. By letting $\overline{X}_i = \frac{1}{n} \sum_{j=1}^{n} X_i^j$ for all $i \in [m]$, we can write $\hat{\theta}_i = \alpha \overline{X}_i + (1-\alpha) \frac{1}{m} \sum_{i=1}^{m} \overline{X}_i$.

Observe that $\mathbb{E}\left[\hat{\theta}_i | \theta\right] = \alpha\theta_i + \frac{1-\alpha}{m} \sum_{l=1}^{m} \theta_l$, where $\theta = (\theta_1, \ldots, \theta_m)$. Thus, the estimator (B.3) is an unbiased estimate of $\{\theta_i\}$. Substituting the $\hat{\theta}_i$ in the MSE, we get that

$$\begin{aligned}
\mathbb{E}_{X_1,\ldots,X_m}\left[\left(\hat{\theta}_i - \theta_i\right)^2\right] &= \mathbb{E}_\theta\left[\mathbb{E}_{X_1,\ldots,X_m}\left[\left(\hat{\theta}_i - \theta_i\right)^2 | \theta\right]\right] \\
&= \mathbb{E}_\theta\left[\mathbb{E}_{X_1,\ldots,X_m}\left[\left(\hat{\theta}_i - \mathbb{E}\left[\hat{\theta}_i|\theta\right] + \mathbb{E}\left[\hat{\theta}_i|\theta\right] - \theta_i\right)^2 | \theta\right]\right] \\
&= \mathbb{E}_\theta\left[\mathbb{E}_{X_1,\ldots,X_m}\left[\left(\hat{\theta}_i - \mathbb{E}\left[\hat{\theta}_i|\theta\right]\right)^2 | \theta\right]\right] \qquad \text{(B.4)} \\
&\quad + \mathbb{E}_\theta\left[\mathbb{E}_{X_1,\ldots,X_m}\left[\left(\mathbb{E}\left[\hat{\theta}_i|\theta\right] - \theta_i\right)^2 | \theta\right]\right] \qquad \text{(B.5)}
\end{aligned}$$

Claim 16.

$$\mathbb{E}_\theta\left[\mathbb{E}_{X_1,\ldots,X_m}\left[\left(\hat{\theta}_i - \mathbb{E}\left[\hat{\theta}_i|\theta\right]\right)^2 | \theta\right]\right] = \alpha^2 \frac{\sigma_x^2}{n} + (1-\alpha)^2 \frac{\sigma_x^2}{mn} + 2\alpha(1-\alpha)\frac{\sigma_x^2}{mn}$$

$$\mathbb{E}_\theta\left[\mathbb{E}_{X_1,\ldots,X_m}\left[\left(\mathbb{E}\left[\hat{\theta}_i|\theta\right] - \theta_i\right)^2 | \theta\right]\right] = (1-\alpha)^2 \mathbb{E}_\theta\left[\left(\frac{1}{m}\sum_{k=1}^{m}\theta_k - \theta_i\right)^2\right] \leq (1-\alpha)^2 \frac{\sigma_\theta^2 (m-1)}{m}$$

Proof. For the first equation:

$$\begin{aligned}
&\mathbb{E}_\theta\left[\mathbb{E}_{X_1,\ldots,X_m}\left[\left(\hat{\theta}_i - \mathbb{E}\left[\hat{\theta}_i|\theta\right]\right)^2 | \theta\right]\right] \\
&= \mathbb{E}_\theta\left[\mathbb{E}_{X_1,\ldots,X_m}\left[\left(\alpha(\overline{X}_i - \theta_i) + (1-\alpha)\frac{1}{m}\sum_{k=1}^{m}(\overline{X}_k - \theta_k)\right)^2 \mid \theta\right]\right] \\
&= \alpha^2 \mathbb{E}\left[\mathbb{E}\left[(\overline{X}_i - \theta_i)^2 \mid \theta\right]\right] + (1-\alpha)^2 \mathbb{E}\left[\mathbb{E}\left[\left(\frac{1}{m}\sum_{k=1}^{m}(\overline{X}_k - \theta_k)\right)^2 \mid \theta\right]\right] \\
&\qquad + 2\alpha(1-\alpha)\mathbb{E}\left[\mathbb{E}\left[\frac{1}{m}\sum_{k=1}^{m}(\overline{X}_i - \theta_i)(\overline{X}_k - \theta_k) \mid \theta\right]\right]
\end{aligned}$$

$$= \alpha^2 \frac{\sigma_x^2}{n} + (1-\alpha)^2 \frac{\sigma_x^2}{mn} + 2\alpha(1-\alpha)\frac{\sigma_x^2}{mn}$$

For the second equation, first note that $\mathbb{E}\left[\hat{\theta}_i|\theta\right] - \theta_i = \alpha\theta_i + \frac{1-\alpha}{m}\sum_{k=1}^m \theta_k - \theta_i = (1-\alpha)\left(\frac{1}{m}\sum_{k=1}^m \theta_k - \theta_i\right)$:

$$\begin{aligned}
\mathbb{E}_\theta\left[\mathbb{E}_{X_1,\dots,X_m}\left[\left(\mathbb{E}\left[\hat{\theta}_i|\theta\right] - \theta_i\right)^2 |\theta\right]\right] &= (1-\alpha)^2\mathbb{E}\left[\left(\frac{1}{m}\sum_{k=1}^m \theta_k - \theta_i\right)^2\right] \\
&= \frac{(1-\alpha)^2}{m^2}\mathbb{E}\left[\left(\sum_{k\neq i}(\theta_k - \theta_i)\right)^2\right] \\
&= \frac{(1-\alpha)^2}{m^2}\left[\sum_{k\neq i}\mathbb{E}(\theta_k-\theta_i)^2 + \sum_{k\neq i, l\neq i, k\neq l}\mathbb{E}(\theta_k-\theta_i)(\theta_l-\theta_i)\right] \\
&\leq \frac{(1-\alpha)^2}{m^2}\left[\sum_{k\neq i}[\mathbb{E}(\theta_k-\mu)^2 + \mathbb{E}(\theta_i-\mu)^2] + \sum_{k\neq i, l\neq i, k\neq l}\mathbb{E}(\theta_k-\theta_i)(\theta_l-\theta_i)\right] \\
&= \frac{(1-\alpha)^2}{m^2}\left[2(m-1)\sigma_\theta^2 + \sum_{k\neq i, l\neq i, k\neq l}\mathbb{E}(\theta_k-\theta_i)(\theta_l-\theta_i)\right] \\
&= \frac{(1-\alpha)^2}{m^2}\left[2(m-1)\sigma_\theta^2 + \sum_{k\neq i, l\neq i, k\neq l}\mathbb{E}(\mu-\theta_i)^2\right] \quad \text{(Since } \mathbb{E}[\theta_k] = \mu \text{ for all } k \in [m]\text{)} \\
&= \frac{(1-\alpha)^2}{m^2}\left[2(m-1)\sigma_\theta^2 + (m-1)(m-2)\sigma_\theta^2\right] \\
&= (1-\alpha)^2\frac{\sigma_\theta^2(m-1)}{m}
\end{aligned}$$

This concludes the proof of Claim 16. □

Substituting the result of Claim 16 into (B.5), we get

$$\begin{aligned}
\mathbb{E}_{X_1,\dots,X_m}\left[\left(\hat{\theta}_i - \theta_i\right)^2\right] &\leq \alpha^2\frac{\sigma_x^2}{n} + (1-\alpha)^2\frac{\sigma_x^2}{mn} + 2\alpha(1-\alpha)\frac{\sigma_x^2}{mn} + (1-\alpha)^2\frac{\sigma_\theta^2(m-1)}{m} \quad \text{(B.6)} \\
&\overset{\text{(a)}}{=} \frac{\sigma_x^2}{n}\left(\alpha^2 + \frac{(1-\alpha)^2 + 2\alpha(1-\alpha)}{m} + \alpha(1-\alpha)\frac{m-1}{m}\right) \\
&= \frac{\sigma_x^2}{n}\left(\alpha + \frac{1-\alpha}{m}\right),
\end{aligned}$$

where in (a) we used $\alpha = \frac{\sigma_\theta^2}{\sigma_\theta^2 + \frac{\sigma_x^2}{n}}$ for the last term to write $(1-\alpha)^2\frac{\sigma_\theta^2(m-1)}{m} = \frac{\sigma_x^2}{n}\alpha(1-\alpha)\frac{m-1}{m}$.

Observe that the estimator in (B.3) is a weighted summation between two estimators: the local estimator $\overline{X}_i = \frac{1}{n}\sum_{j=1}^{n} X_i^j$, and the global estimator $\hat{\mu} = \frac{1}{m}\sum_{i=1}^{m} \overline{X}_i$. Thus, the MSE in (a) consists of four terms: 1) The variance of the local estimator ($\frac{\sigma_x^2}{n}$). 2) The variance of the global estimator ($\frac{\sigma_x^2}{nm}$). 3) The correlation between the local estimator and the global estimator ($\frac{\sigma_x^2}{nm}$). 4) The bias term $\mathbb{E}_\theta\left[\mathbb{E}_{X_1,\ldots,X_m}\left[\left(\mathbb{E}\left[\hat{\theta}_i|\theta\right]-\theta_i\right)^2|\theta\right]\right]$. This completes the proof of Theorem 3.

B.3.2 Proof of Theorem 4, Equation (3.5)

Similar to the proof of Theorem 3, here also we will derive the optimal estimator and prove the MSE for the one dimensional case, and the final result can be obtained by applying these to each of the d coordinates separately.

Let $\theta = (\theta_1, \ldots, \theta_m)$ denote the personalized models vector. For given a constraint function q, we set the personalized model as follows:

$$\hat{\theta}_i = \alpha\left(\frac{1}{n}\sum_{j=1}^{n} X_i^j\right) + (1-\alpha)\left(\frac{1}{m}\sum_{i=1}^{m} q(\overline{X}_i)\right) \qquad \forall i \in [m], \tag{B.7}$$

where $\overline{X}_i = \frac{1}{n}\sum_{j=1}^{n} X_i^j$. From the second condition on the function q, we get that

$$\mathbb{E}\left[\hat{\theta}_i|\theta\right] = \alpha\theta_i + \frac{1-\alpha}{m}\sum_{l=1}^{m}\theta_l, \tag{B.8}$$

Thus, by following similar steps as the proof of Theorem 3, we get that:

$$\begin{aligned}
\mathbb{E}\left[\left(\hat{\theta}_i-\theta_i\right)^2\right] &= \mathbb{E}\left[\mathbb{E}\left[\left(\hat{\theta}_i-\theta_i\right)^2|\theta\right]\right] \\
&= \mathbb{E}\left[\mathbb{E}\left[\left(\hat{\theta}_i-\mathbb{E}\left[\hat{\theta}_i|\theta\right]+\mathbb{E}\left[\hat{\theta}_i|\theta\right]-\theta_i\right)^2|\theta\right]\right] \\
&= \mathbb{E}\left[\mathbb{E}\left[\left(\hat{\theta}_i-\mathbb{E}\left[\hat{\theta}_i|\theta\right]\right)^2|\theta\right]\right] + \mathbb{E}\left[\mathbb{E}\left[\left(\mathbb{E}\left[\hat{\theta}_i|\theta\right]-\theta_i\right)^2|\theta\right]\right] \\
&\overset{(a)}{=} \alpha^2\frac{\sigma_x^2}{n} + (1-\alpha)^2\mathbb{E}\left[\left(\frac{1}{m}\sum_{l=1}^{m} q(\overline{X}_l)-\theta_l\right)^2|\theta\right] \\
&\quad + 2\alpha(1-\alpha)\mathbb{E}\left[\left(\overline{X}_i-\theta_i\right)\left(\frac{1}{m}\sum_{l=1}^{m} q(\overline{X}_l)-\theta_l\right)|\theta\right]
\end{aligned} \tag{B.9}$$

$$
\begin{aligned}
&+ (1-\alpha)^2 \mathbb{E}\left[\left(\frac{1}{m}\sum_{k=1}^{m}\theta_k - \theta_i\right)^2\right] \\
&\overset{(b)}{=} \alpha^2\frac{\sigma_x^2}{n} + \frac{(1-\alpha)^2\left(\frac{\sigma_x^2}{n}+\sigma_q^2\right)}{m} + \frac{2\alpha(1-\alpha)\sigma_x^2}{mn} \qquad \text{(B.10)} \\
&+ (1-\alpha)^2 \mathbb{E}\left[\left(\frac{1}{m}\sum_{k=1}^{m}\theta_k - \theta_i\right)^2\right] \\
&\leq \alpha^2\frac{\sigma_x^2}{n} + \frac{(1-\alpha)^2\left(\frac{\sigma_x^2}{n}+\sigma_q^2\right)}{m} + 2\alpha(1-\alpha)\frac{\sigma_x^2}{mn} + (1-\alpha)^2\frac{\sigma_\theta^2(m-1)}{m} \\
&\overset{(c)}{=} \frac{\sigma_x^2}{n}\left(\alpha^2 + \frac{(1-\alpha)^2+2\alpha(1-\alpha)}{m} + \alpha(1-\alpha)\frac{m-1}{m}\right) \\
&= \frac{\sigma_x^2}{n}\left(\alpha + \frac{1-\alpha}{m}\right), \qquad \text{(B.11)}
\end{aligned}
$$

where step (a) follows by substituting the expectation of the personalized model from (B.8). Step (b) follows from the first and third conditions of the function q. Step (c) follows by choosing $\alpha = \frac{\sigma_\theta^2+\frac{\sigma_q^2}{m-1}}{\sigma_\theta^2+\frac{\sigma_q^2}{m-1}+\frac{\sigma_x^2}{n}}$. This derives the result stated in (3.5) in Theorem 4.

B.3.2.1 Proof of Theorem 4, Part 1

The proof consists of two steps. First, we use the concentration property of the Gaussian distribution to show that the local sample means $\{\overline{X}_i\}$ are bounded within a small range with high probability. Second, we apply an unbiased stochastic quantizer on the projected sample mean.

The local samples $X_i^1, \ldots, X_i^n$ are drawn i.i.d. from a Gaussian distribution with mean θ_i and variance σ_x^2, and hence, we have that $\overline{X}_i \sim \mathcal{N}(\theta_i, \frac{\sigma_x^2}{n})$. Thus, from the concentration property of the Gaussian distribution, we get that $\Pr[|\overline{X}_i - \theta_i| > c_1] \leq \exp\left(-\frac{nc_1^2}{\sigma_x^2}\right)$ for all $i \in [m]$. Similarly, the models $\theta_1, \ldots, \theta_m$ are drawn i.i.d. from a Gaussian distribution with mean $\mu \in [-r, r]$ and variance σ_θ^2, hence,, we get $\Pr[|\theta_i - \mu| > c_2] \leq \exp\left(-\frac{c_2^2}{\sigma_\theta^2}\right)$ for all $i \in [m]$. Let $\mathcal{E} = \left\{\overline{X}_i \in [-a, a] : \forall i \in [m]\right\}$, where $a = r + c_1 + c_2$. Thus, from the union bound, we get

that $\Pr[\mathcal{E}] > 1 - m(e^{-\frac{nc_1^2}{\sigma_x^2}} + e^{-\frac{c_2^2}{\sigma_\theta^2}})$. By setting $c_1 = \sqrt{\frac{\sigma_x^2}{n}\log(m^2n)}$ and $c_2 = \sqrt{\sigma_\theta^2\log(m^2n)}$, we get that $a = r + \frac{\sigma_x}{\sqrt{n}}\sqrt{\log(m^2n)} + \sigma_\theta\sqrt{\log(m^2n)}$, and $\Pr[\mathcal{E}] = 1 - \frac{2}{mn}$.

Let $q_k : [-a, a] \to \mathcal{Y}_k$ be a quantization function with k-bits, where $\mathcal{Y}_k$ is a discrete set of cardinality $|\mathbb{Y}_k| = 2^k$. For given $x \in [-a, a]$, the output of the function q_k is given by:

$$q_k(x) = \frac{2a}{2^k - 1}\left(\lfloor\tilde{x}\rfloor + \mathsf{Bern}\left(\tilde{x} - \lfloor\tilde{x}\rfloor\right)\right) - a, \tag{B.12}$$

where $\mathsf{Bern}(p)$ is a Bernoulli random variable with bias p, and $\tilde{x} = \frac{2^k-1}{2a}(x + a) \in [0, 2^k - 1]$. Observe that the output of the function q_k requires only k-bits for transmission. Furthermore, the function q_k satisfies the following conditions:

$$\mathbb{E}\left[q_k(x)\right] = x, \tag{B.13}$$

$$\sigma_{q_k}^2 = \mathbb{E}\left[(q_k(x) - x)^2\right] \leq \frac{a^2}{(2^k - 1)^2}. \tag{B.14}$$

Let each client applies the function q_k on the projected local mean $\tilde{X}_i = \mathsf{Proj}_{[-a,a]}\left[\overline{X}_i\right]$ and sends the output to the server for all $i \in [m]$. Conditioned on the event $\mathcal{E}$, i.e., $\overline{X}_i \in [-a, a] \quad \forall i \in [m]$, and using (B.11), we get that

$$MSE = \mathbb{E}_{\theta,\mathbf{X}}\left[\left(\hat{\theta}_i - \theta_i\right)^2\right] \leq \frac{\sigma_x^2}{n}\left(\frac{1-\alpha}{m} + \alpha\right), \tag{B.15}$$

where $\alpha = \frac{\sigma_\theta^2 + \frac{a^2}{(2^k-1)^2(m-1)}}{\sigma_\theta^2 + \frac{a^2}{(2^k-1)^2(m-1)} + \frac{\sigma_x^2}{n}}$ and $a = r + \frac{\sigma_x}{\sqrt{n}}\sqrt{\log(m^2n)} + \sigma_\theta\sqrt{\log(m^2n)}$. Note that the event $\mathcal{E}$ happens with probability at least $1 - \frac{2}{mn}$.

B.3.2.2 Proof of Theorem 4, Part 2

We define the (random) mechanism $q_p : [-a, a] \to \mathcal{R}$ that takes an input $x \in [-a, a]$ and generates a user-level (ϵ_0, δ)-LDP output $y \in \mathbb{R}$, where $y = q_p(x)$ is given by:

$$q_p(x) = x + \nu, \tag{B.16}$$

where $\nu \sim \mathcal{N}(0, \sigma_{\epsilon_0}^2)$ is a Gaussian noise. By setting $\sigma_{\epsilon_0}^2 = \frac{8a^2\log(2/\delta)}{\epsilon_0^2}$, we get that the output of the function $q_p(x)$ is (ϵ_0, δ)-LDP from [46]. Furthermore, the function q_p satisfies the

following conditions:

$$\mathbb{E}\left[q_p(x)\right] = x, \tag{B.17}$$

$$\sigma_{q_p}^2 = \mathbb{E}\left[(q_p(x) - x)^2\right] \leq \frac{8a^2 \log(2/\delta)}{\epsilon_0^2}. \tag{B.18}$$

Similar to the proof of Theorem 4, Part 1, let each client applies the function q_p on the projected local mean $\tilde{X}_i = \mathsf{Proj}_{[-a,a]}\left[\overline{X}_i\right]$ and sends the output to the server for all $i \in [m]$. Conditioned on the event $\mathcal{E}$, i.e., $\overline{X}_i \in [-a, a] \quad \forall i \in [m]$, and using (B.11), we get that

$$\text{MSE} = \mathbb{E}_{\theta,\mathbf{X}}\left[\left(\hat{\theta}_i - \theta_i\right)^2\right] \leq \frac{\sigma_x^2}{n}\left(\frac{1-\alpha}{m} + \alpha\right), \tag{B.19}$$

where $\alpha = \frac{\sigma_\theta^2 + \frac{8a^2\log(2/\delta)}{\epsilon_0^2(m-1)}}{\sigma_\theta^2 + \frac{8a^2\log(2/\delta)}{\epsilon_0^2(m-1)} + \frac{\sigma_x^2}{n}}$ and $a = r + \frac{\sigma_x}{\sqrt{n}}\sqrt{\log(m^2 n)} + \sigma_\theta\sqrt{\log(m^2 n)}$. Note that the event $\mathcal{E}$ happens with probability at least $1 - \frac{2}{mn}$.

Remark 12 (Privacy with communication efficiency)**.** *Note that our private estimation algorithm for the Gaussian case adds Gaussian noise (which is a real number) but that can also be made communication-efficient by alternatively adding a discrete Gaussian noise [22].*

B.3.3 Lower Bound

Here we discuss the lower bound using Fisher information technique similar to [9]. In particular we use a Bayesian version of Cramer-Rao lower bound and van Trees inequality [57]. Let us denote $f(X|\theta)$ as the data generating conditional density function and $\pi(\theta)$ as the prior distribution that generates θ. Let us denote $\mathbb{E}_\theta$ as the expectation with respect to the randomness of θ and $\mathbb{E}$ as the expectation with respect to randomness of X and θ. First we define two types of Fisher information:

$$I_X(\theta) = \mathbb{E}_\theta \nabla_\theta \log(f(X|\theta)) \nabla_\theta \log(f(X|\theta))^T$$

$$I(\pi) = \mathbb{E} \nabla_\theta \log(\pi(\theta)) \nabla_\theta \log(\pi(\theta))^T$$

namely Fisher information of estimating θ from samples X and Fisher information of prior π. Here the logarithm is elementwise. For van Trees inequality we need the following regularity conditions:

- $f(X|\cdot)$ and $\pi(\cdot)$ are absolutely continuous and $\pi(\cdot)$ vanishes at the end points of Θ.
- $\mathbb{E}_\theta \nabla_\theta \log(f(X|\theta)) = 0$
- We also assume both density functions are continuously differentiable.

These assumptions are satisfied for the Gaussian setting for any finite mean μ, they are satisfied for Bernoulli setting as long as parameters α and β are larger than 1. Assuming local samples X are generated i.i.d with $f(x|\theta)$, the van Trees inequality for one dimension is as follows:

$$\mathbb{E}(\widehat{\theta}(X) - \theta)^2 \geq \frac{1}{n\mathbb{E}I_x(\theta) + I(\pi)}$$

where $I_X(\theta) = \mathbb{E}_\theta \log(f(X|\theta))'^2$ and $I(\pi) = \mathbb{E}\log(\pi(\theta))'^2$. Assuming $\theta \in \mathbb{R}^d$ and each dimension is independent from each other, by [57] we have:

$$\mathbb{E}\|\widehat{\theta}(X) - \theta\|^2 \geq \frac{d^2}{n\mathbb{E}\mathrm{Tr}(I_x(\theta)) + \mathrm{Tr}(I(\pi))} \tag{B.20}$$

Note, the lower bound on the average risk directly translates as a lower bound on $\sup_{\theta\in\Theta} \mathbb{E}_X\|\widehat{\theta}(X) - \theta\|^2$. Before our proof we have a useful fact:

Fact 6. *Given some random variable $X \sim N(Y, \sigma_y^2)$ where $Y \sim N(Z, \sigma_z^2)$ we have $X \sim N(z, \sigma_z^2 + \sigma_y^2)$.*

Proof. We will give the proof in one dimension, however, it can easily be extended to multidimensional case where each dimension is independent. For all $t \in \mathbb{R}$ we have,

$$\mathbb{E}_X[\exp(itX)] = \mathbb{E}_Y\mathbb{E}_X[\exp(itX)|Y] = \mathbb{E}_Y[\exp(itY - \frac{\sigma_x^2 t^2}{2})]$$

$$
\begin{aligned}
&= \exp(-\frac{\sigma_x^2 t^2}{2})\mathbb{E}_Y[\exp(itY)] \\
&= \exp(-\frac{\sigma_x^2 t^2}{2})\exp(itz - \frac{\sigma_y^2 t^2}{2}) \\
&= \exp(itz - \frac{(\sigma_x^2+\sigma_y^2)t^2}{2})
\end{aligned}
$$

where the last line is the characteristic function of a Gaussian with mean z and variance $\sigma_x^2 + \sigma_y^2$. □

Gaussian case with perfect knowledge of prior. In this setting we know that $\theta_i \sim N(\mu\mathbf{1}, \sigma_\theta^2 \mathrm{I}_d)$, hence, $I(\pi) = \frac{1}{\sigma_\theta^2}\mathrm{I}_d$, similarly $I_X(\theta) = \frac{1}{\sigma_x^2}\mathrm{I}_d$. Then,

$$
\sup_{\theta_i} \mathbb{E}\|\widehat{\theta}_i(X) - \theta_i\|^2 \geq \frac{d^2}{n\mathbb{E}\frac{d}{\sigma_x^2} + \frac{d}{\sigma_\theta^2}} = \frac{d\sigma_\theta^2\sigma_x^2}{n\sigma_\theta^2 + \sigma_x^2} \tag{B.21}
$$

Gaussian case with estimated population mean. In this setting instead of a true prior we have a prior whose mean is the average of all data spread across clients, i.e., we assume $\theta_i \sim N(\widehat{\mu}, \sigma_\theta^2 \mathrm{I}_d)$ where $\widehat{\mu} = \frac{1}{mn}\sum_{i,j}^{m,n} X_i^j$. We additionally know that there is a Markov relation such that $X_i^j|\theta_j \sim N(\theta_j, \sigma_x^2 \mathrm{I}_d)$ and $\theta_j \sim N(\mu, \sigma_\theta^2 \mathrm{I}_d)$. While the true prior is parameterized with mean μ, θ_i in this form is not parameterized by μ but by $\widehat{\mu}$ which itself has randomness due X_i^j. However, using Fact 6 twice we can write $\theta_i \sim N(\mu, (\sigma_\theta^2 + \frac{\sigma_\theta^2}{m} + \frac{\sigma_x^2}{mn})\mathrm{I}_d)$. Then using the van Trees inequality similar to the lower bound in perfect case we can obtain:

$$
\sup_{\theta_i \in \Theta} \mathbb{E}_X\|\widehat{\theta}_i(X) - \theta_i\|^2 \geq d\frac{\sigma_\theta^2\sigma_x^2 + \frac{\sigma_x^4}{mn}}{n\sigma_\theta^2 + \sigma_x^2} \tag{B.22}
$$

B.4 Personalized Estimation – Bernoulli Model

B.4.1 When α, β are Known

Analogous to the Gaussian case, we can show that if α, β are known, then the posterior mean estimator has a closed form expression: $\widehat{p}_i = a\overline{X}_i + (1-a)\frac{\alpha}{\alpha+\beta}$ (where $a = {}^{n}/_{\alpha+\beta+n}$) and

achieves the MSE: $\mathbb{E}_{p_i\sim\pi}\mathbb{E}_{\widehat{p}_i,X_1,\ldots,X_m}(\widehat{p}_i - p_i)^2 \leq \frac{\alpha\beta}{n(\alpha+\beta)(\alpha+\beta+1)}\frac{n}{\alpha+\beta+n}$. We show this below.

For a client i, let $\pi(p_i)$ be distributed as Beta(α, β). In this setting, we model that each client generates local samples according to Bern(p_i). Consequently, each client has a Binomial distribution regarding the sum of local data samples. Estimating Bernoulli parameter p_i is related to Binomial distribution Bin(n, p_i) (the sum of data samples) Z_i since it is the sufficient statistic of Bernoulli distribution. The distribution for Binomial variable Z_i given p_i is $P(Z_i = z_i|p_i) = \binom{n}{z_i}p_i^{z_i}(1-p_i)^{n-z_i}$. It is a known fact that for any prior, the Bayesian MSE risk minimizer is the posterior mean $\mathbb{E}\left[p_i|Z_i = z_i\right]$.

When $p_i \sim$ Beta(α, β), we have posterior

$$\begin{aligned}
f(p_i|Z_i = z_i) &= \frac{P(z_i|p_i)}{P(z_i)}\pi(p_i) \\
&= \frac{\binom{n}{z_i}p_i^{z_i}(1-p_i)^{n-z_i}}{P(z_i)}\frac{p_i^{\alpha-1}(1-p_i)^{\beta-1}}{B(\alpha,\beta)} \\
&= \frac{\binom{n}{z_i}}{P(z_i)}\frac{B(\alpha+z_i,\beta+n-z_i)}{B(\alpha,\beta)}\frac{p_i^{\alpha+z_i-1}(1-p_i)^{\beta+n-z_i-1}}{B(\alpha+z_i,\beta+n-z_i)},
\end{aligned}$$

where $B(\alpha,\beta) = \frac{\Gamma(\alpha)\Gamma(\beta)}{\Gamma(\alpha+\beta)}$, and

$$\begin{aligned}
P(z_i) &= \int P(z_i|p_i)\pi(p_i)dp_i \\
&= \int \binom{n}{z_i}p_i^{z_i}(1-p_i)^{n-z_i}\frac{p_i^{\alpha-1}(1-p_i)^{\beta-1}}{B(\alpha,\beta)}dp_i \\
&= \binom{n}{z_i}\frac{B(z_i+\alpha, n-z_i+\beta)}{B(\alpha,\beta)}\underbrace{\int \frac{p_i^{\alpha+z_i-1}(1-p_i)^{\beta+n-z_i-1}}{B(\alpha+z_i,\beta+n-z_i)}dp_i}_{\text{integral of a Beta distribution}} \\
&= \binom{n}{z_i}\frac{B(z_i+\alpha, n-z_i+\beta)}{B(\alpha,\beta)}
\end{aligned}$$

Thus, we get that the posterior distribution $f(p_i|Z_i = z_i) = \frac{p_i^{\alpha+z_i-1}(1-p_i)^{\beta+n-z_i-1}}{B(\alpha+z_i,\beta+n-z_i)}$ is a beta distribution Beta$(z_i+\alpha, n-z_i+\beta)$. As a result, the posterior mean is given by:

$$\widehat{p}_i = \frac{\alpha+Z_i}{\alpha+\beta+n} = a\left(\frac{Z_i}{n}\right) + (1-a)\left(\frac{\alpha}{\alpha+\beta}\right), \tag{B.23}$$

where $a = \frac{n}{\alpha+\beta+n}$. Observe that $\mathbb{E}_{p_i\sim\mathsf{Beta}(\alpha,\beta)}[p_i] = \frac{\alpha}{\alpha+\beta}$, i.e., the estimator is a weighted summation between the local estimator $\frac{z_i}{n}$ and the global estimator $\mu = \frac{\alpha}{\alpha+\beta}$.
We have $R_{p_i}(\widehat{p}_i) = \mathbb{E}_\pi\mathbb{E}(\widehat{p}_i - p_i)^2$. The MSE of the posterior mean is given by:

$$\begin{aligned}
\text{MSE} &= \mathbb{E}[(\hat{p}_i - p_i)^2] \\
&= \mathbb{E}\left[\left(a\left(\frac{z_i}{n} - p_i\right) + (1-a)(\mu - p_i)\right)^2\right] \\
&= a^2\mathbb{E}\left[\left(\frac{z_i}{n} - p_i\right)^2\right] + (1-a)^2\mathbb{E}\left[(\mu - p_i)^2\right] \\
&= a^2\mathbb{E}_{p_i\sim\pi(p_i)}\left[\frac{p_i(1-p_i)}{n}\right] + (1-a)^2\frac{\alpha\beta}{(\alpha+\beta)^2(\alpha+\beta+1)} \\
&= a^2\frac{\alpha\beta}{n(\alpha+\beta)(\alpha+\beta+1)} + (1-a)^2\frac{\alpha\beta}{(\alpha+\beta)^2(\alpha+\beta+1)} \\
&= \frac{\alpha\beta}{n(\alpha+\beta)(\alpha+\beta+1)}\left(\frac{n}{\alpha+\beta+n}\right).
\end{aligned}$$

The last equality is obtained by setting $a = \frac{n}{\alpha+\beta+n}$.

Remark 13. *Note that $\overline{X}_i := \frac{Z_i}{n}$ is the estimator based only on the local data and $\alpha/(\alpha+\beta)$ is the true global mean, and $\widehat{p}_i = a\overline{X}_i + (1-a)\frac{\alpha}{\alpha+\beta}$, where $a = n/\alpha+\beta+n$ (see (B.23)) is the estimator based on all the data. Observe that when $n \to \infty$, then $a \to 1$, which implies that $\widehat{p}_i \to \overline{X}_i$. Otherwise, when $\alpha+\beta$ is large (i.e., the variance of the beta distribution is small), then $a \to 0$, which implies that $\widehat{p}_i \to \alpha/(\alpha+\beta)$. Both these conclusions conform to the conventional wisdom as mentioned in the Gaussian case. It can be shown that the local estimate $\overline{X}_i$ achieves the Bayesian risk of $\mathbb{E}_{p_i\sim Beta(\alpha,\beta)}\mathbb{E}_{X_i}[(\overline{X}_i - p_i)^2] = \mathbb{E}_{p_i\sim Beta(\alpha,\beta)}(p_i(1-p_i))/n = \alpha\beta/n(\alpha+\beta)(\alpha+\beta+1)$, which implies that the personalized estimation with perfect prior always outperforms the local estimate with a multiplicative gain $a = n/(n+\alpha+\beta) \leq 1$.*

B.4.2 When α, β are Unknown: Proof of Theorem 5

The personalized model of the ith client with unknown parameters α, β is given by:

$$\hat{p}_i = \overline{a}_i\overline{X}_i + (1-\overline{a}_i)\left(\hat{\mu}_i\right), \tag{B.24}$$

where $\overline{a}_i = \frac{n}{\frac{\hat{\mu}_i(1-\hat{\mu}_i)}{\hat{\sigma}_i^2}+n}$, the empirical mean $\hat{\mu}_i = \frac{1}{m-1}\sum_{l\neq i}\overline{X}_l$, and the empirical variance $\hat{\sigma}_i^2 = \frac{1}{m-2}\sum_{l\neq i}(\overline{X}_l - \hat{\mu}_i)^2$. From [163, Lemma 1], with probability $1-\frac{1}{m^2n}$, we get that

$$|\mu - \hat{\mu}_i| \leq \sqrt{\frac{3\log(4m^2n)}{m-1}}$$
$$|\sigma^2 - \hat{\sigma}_i^2| \leq \sqrt{\frac{3\log(4m^2n)}{m-1}},$$

where $\mu = \frac{\alpha}{\alpha+\beta}$, $\sigma^2 = \frac{\alpha\beta}{(\alpha+\beta)^2(\alpha+\beta+1)}$ are the true mean and variance of the beta distribution, respectively. Let $c = \sqrt{\frac{3\log(4m^2n)}{m-1}}$. Conditioned on the event $\mathcal{E} = \{|\mu - \hat{\mu}_i| \leq c, |\sigma^2 - \hat{\sigma}_i^2| \leq c : \forall i \in [m]\}$ that happens with probability at least $1-\frac{1}{mn}$, we get that:

$$\begin{aligned}
\mathbb{E}\left[(\hat{p}_i - p_i)^2 \,|Z_{-i}\right] &= \overline{a}^2\mathbb{E}\left[\left(\frac{Z_i}{n} - p_i\right)^2\right] + (1-\overline{a})^2\mathbb{E}\left[(\hat{\mu}_i - p_i)^2\,|Z_{-i}\right] \\
&= \overline{a}^2\left(\frac{\alpha\beta}{n(\alpha+\beta)(\alpha+\beta+1)}\right) + (1-\overline{a})^2\left(\mathbb{E}\left[(\mu-p_i)^2\right] + (\mu-\hat{\mu}_i)^2\right) \\
&= \overline{a}^2\left(\frac{\alpha\beta}{n(\alpha+\beta)(\alpha+\beta+1)}\right) + (1-\overline{a})^2\left(\frac{\alpha\beta}{(\alpha+\beta)^2(\alpha+\beta+1)} + (\mu-\hat{\mu}_i)^2\right) \\
&\leq \overline{a}^2\left(\frac{\alpha\beta}{n(\alpha+\beta)(\alpha+\beta+1)}\right) + (1-\overline{a})^2\left(\frac{\alpha\beta}{(\alpha+\beta)^2(\alpha+\beta+1)} + c^2\right),
\end{aligned}$$

where the expectation is with respect to $z_i \sim \mathsf{Binom}(p_i, n)$ and $p_i \sim \mathsf{Beta}(\alpha,\beta)$ and $Z_{-i} = \{z_1,\ldots,z_{i-1},z_{i+1},\ldots,z_m\}$ denotes the entire dataset except the ith client data (z_i). By taking the expectation with respect to the datasets Z_{-i}, we get that the MSE is bounded by:

$$\text{MSE} \leq \mathbb{E}\left[\overline{a}^2\right]\left(\frac{\alpha\beta}{n(\alpha+\beta)(\alpha+\beta+1)}\right) + \mathbb{E}\left[(1-\overline{a})^2\right]\left(\frac{\alpha\beta}{(\alpha+\beta)^2(\alpha+\beta+1)} + \frac{3\log(4m^2n)}{m-1}\right),$$

with probability at least $1-\frac{1}{mn}$. This completes the proof of Theorem 5.

B.4.3 With Privacy Constraints: Proof of Theorem 6

First, we prove some properties of the private mechanism q_p. Observe that for any two inputs $x, x' \in [0,1]$, we have that:

$$\frac{\Pr[q_p(x)=y]}{\Pr[q_p(x')=y]} = \frac{\frac{e^{\epsilon_0}}{e^{\epsilon_0}+1} - x\frac{e^{\epsilon_0}-1}{e^{\epsilon_0}+1}}{\frac{e^{\epsilon_0}}{e^{\epsilon_0}+1} - x'\frac{e^{\epsilon_0}-1}{e^{\epsilon_0}+1}} \leq e^{\epsilon_0}, \tag{B.25}$$

for $y = \frac{-1}{e^{\epsilon_0}-1}$. Similarly, we can prove (B.25) for the output $y = \frac{e^{\epsilon_0}}{e^{\epsilon_0}-1}$. Thus, the mechanism q_p is user-level ϵ_0-LDP. Furthermore, for given $x \in [0,1]$, we have that

$$\mathbb{E}\left[q_p(x)\right] = x. \tag{B.26}$$

Thus, the output of the mechanism q_p is an unbiased estimate of the input x. From the Hoeffding's inequality for bounded random variables, we get that:

$$\begin{aligned} \Pr[|\hat{\mu}_i^{(p)} - \mu| > t] &\leq 2\exp\left(\frac{-3(e^{\epsilon_0}-1)^2(m-1)t^2}{(e^{\epsilon_0}+1)^2}\right) \\ \Pr[|\hat{\sigma}_i^{2(p)} - \sigma_2| > t] &\leq 2\exp\left(\frac{-3(e^{\epsilon_0}-1)^2(m-1)t^2}{(e^{\epsilon_0}+1)^2}\right) \end{aligned} \tag{B.27}$$

Thus, we have that the event $\mathcal{E} = \{|\hat{\mu}_i^{(p)} - \mu| \leq c_p, |\hat{\sigma}_i^{2(p)} - \sigma^2| \leq c_p : \forall i \in [m]\}$ happens with probability at least $1 - \frac{1}{mn}$, where $c_p = \sqrt{\frac{(e^{\epsilon_0}+1)^2\log(4m^2n)}{3(e^{\epsilon_0}-1)^2(m-1)}}$. By following the same steps as the non-private estimator, we get the fact that the MSE of the private model is bounded by:

$$\begin{aligned} \text{MSE} \leq \mathbb{E}\left[\overline{a}^2\right] &\left(\frac{\alpha\beta}{n(\alpha+\beta)(\alpha+\beta+1)}\right) \\ &+ \mathbb{E}\left[(1-\overline{a})^2\right]\left(\frac{\alpha\beta}{(\alpha+\beta)^2(\alpha+\beta+1)} + \frac{(e^{\epsilon_0}+1)^2\log(4m^2n)}{3(e^{\epsilon_0}-1)^2(m-1)}\right), \end{aligned} \tag{B.28}$$

where $\overline{a}^{(p)} = \frac{n}{\frac{\hat{\mu}_i^{(p)}(1-\hat{\mu}_i^{(p)})}{\hat{\sigma}_i^{2(p)}}+n}$ and the expectation is with respect to the randomness of the private mechanism q_p and the clients data $\{z_1, \ldots, z_{i-1}, z_{i+1}, \ldots, z_m\}$. This completes the proof of Theorem 6.

Remark 14 (Privacy with communication efficiency)**.** *Note that our private estimation algorithm for the Bernoulli case is already communication-efficient as each client sends only one bit to the server.*

Remark 15 (Client sampling)**.** *For simplicity, in the theoretical analysis in Gaussian and Bernoulli models, we assume that all clients participate in the estimation process. However, a simple modification to our analysis also handles the case where only K out of m clients participate: in all our theorem statements we would have to modify to have K instead m. Note that we do client sampling for our experiments in Table 3.1.*

B.5 Personalized Estimation – Mixture Model

Consider a set of m clients, where the i-th client has a local dataset $X_i = (X_{i1}, \ldots, X_{in})$ of n samples for $i \in [m]$, where $X_{ij} \in \mathbb{R}^d$. The local samples X_i of the i-th client are drawn i.i.d. from a Gaussian distribution $\mathcal{N}(\boldsymbol{\theta}_i, \sigma_x^2 \mathbb{I}_d)$ with unknown mean $\boldsymbol{\theta}_i$ and known variance $\sigma_x^2 \mathbb{I}_d$.

In this section, we assume that the personalized models $\boldsymbol{\theta}_1, \ldots, \boldsymbol{\theta}_m$ are drawn i.i.d. from a discrete distribution $\mathbb{P} = [p_1, \ldots, p_k]$ for given k candidates $\boldsymbol{\mu}_1, \ldots, \boldsymbol{\mu}_k \in \mathbb{R}^d$. In other works, $\Pr[\boldsymbol{\theta}_i = \boldsymbol{\mu}_l] = p_l$ for $l \in [k]$ and $i \in [m]$. The goal of each client is to estimate her personalized model $\{\boldsymbol{\theta}_i\}$ that minimizes the mean square error defined as follows:

$$\text{MSE} = \mathbb{E}_{\{\boldsymbol{\theta}_i, X_i\}} \|\boldsymbol{\theta}_i - \hat{\boldsymbol{\theta}}_i\|^2, \tag{B.29}$$

where the expectation is taken with respect to the personalized models $\boldsymbol{\theta}_i$ and the local samples $\{X_{ij} \sim \mathcal{N}(\boldsymbol{\theta}_i, \sigma_x^2 \mathbb{I}_d)\}$. Furthermore, $\hat{\boldsymbol{\theta}}_i$ denotes the estimate of the personalized model $\boldsymbol{\theta}_i$ for $i \in [m]$.

First, we start with a simple case when the clients have perfect knowledge of the prior distribution, i.e., the i-th client knows the k Gaussian distributions $\mathcal{N}(\boldsymbol{\mu}_1, \sigma_{\boldsymbol{\theta}}^2), \ldots, \mathcal{N}(\boldsymbol{\mu}_k, \sigma_{\boldsymbol{\theta}}^2)$ and the prior distribution $\boldsymbol{\alpha} = [\alpha_1, \ldots, \alpha_k]$. This will serve as a stepping stone to handle the more general case when the prior distribution is unknown.

B.5.1 When the Prior Distribution is Known

In this case, the i-th client does not need the data of the other clients as she has a perfect knowledge about the prior distribution.

Theorem 18. *For given a perfect knowledge $\boldsymbol{\alpha} = [\alpha_1, \ldots, \alpha_k]$ and $\mathcal{N}(\boldsymbol{\mu}_1, \sigma_{\boldsymbol{\theta}}^2), \ldots, \mathcal{N}(\boldsymbol{\mu}_k, \sigma_{\boldsymbol{\theta}}^2)$, the optimal personalized estimator that minimizes the MSE is given by:*

$$\hat{\boldsymbol{\theta}}_i = \sum_{l=1}^{k} a_l^{(i)} \boldsymbol{\mu}_l, \tag{B.30}$$

where $\alpha_l^{(i)} = \dfrac{p_l \exp\left(-\frac{\sum_{j=1}^n \|X_{ij}-\boldsymbol{\mu}_l\|^2}{2\sigma_x^2}\right)}{\sum_{s=1}^k p_s \exp\left(-\frac{\sum_{j=1}^n \|X_{ij}-\boldsymbol{\mu}_s\|^2}{2\sigma_x^2}\right)}$ *denotes the weight associated to the prior model* $\boldsymbol{\mu}_l$ *for* $l \in [k]$.

Proof. Let $\boldsymbol{\theta}_i \sim \mathbb{P}$, where $\mathbb{P} = [p_1, \ldots, p_k]$ and $p_l = \Pr[\boldsymbol{\theta}_i = \boldsymbol{\mu}_l]$ for $l \in [k]$. The goal is to design an estimator $\hat{\boldsymbol{\theta}}_i$ that minimizes the MSE given by:

$$\text{MSE} = \mathbb{E}_{\boldsymbol{\theta}_i \sim \mathbb{P}} \mathbb{E}_{\{X_{ij} \sim \mathcal{N}(\boldsymbol{\theta}_i, \sigma_x^2)\}} \left[\|\hat{\boldsymbol{\theta}}_i - \boldsymbol{\theta}_i\|^2 \right]. \tag{B.31}$$

Let $X_i = (X_{i1}, \ldots, X_{in})$. By following the standard proof of the minimum MSE, we get that:

$$\begin{aligned}
\mathbb{E}_{\boldsymbol{\theta}_i} \mathbb{E}_{X_i} \left[\|\hat{\boldsymbol{\theta}}_i - \boldsymbol{\theta}_i\|^2 \right] &= \mathbb{E}_{X_i} \mathbb{E}_{\boldsymbol{\theta}_i | X_i} \left[\|\hat{\boldsymbol{\theta}}_i - \mathbb{E}[\boldsymbol{\theta}_i | X_i] + \mathbb{E}[\boldsymbol{\theta}_i | X_i] - \boldsymbol{\theta}_i\|^2 \Big| X_i \right] \\
&= \mathbb{E}_{X_i} \mathbb{E}_{\boldsymbol{\theta}_i | X_i} \left[\|\mathbb{E}[\boldsymbol{\theta}_i | X_i] - \boldsymbol{\theta}_i\|^2 \big| X_i \right] + \mathbb{E}_{X_i} \mathbb{E}_{\boldsymbol{\theta}_i | X_i} \left[\|\mathbb{E}[\boldsymbol{\theta}_i | X_i] - \hat{\boldsymbol{\theta}}_i\|^2 \Big| X_i \right] \\
&\geq \mathbb{E}_{X_i} \mathbb{E}_{\boldsymbol{\theta}_i | X_i} \left[\|\mathbb{E}[\boldsymbol{\theta}_i | X_i] - \boldsymbol{\theta}_i\|^2 \big| X_i \right],
\end{aligned} \tag{B.32}$$

where the last inequality is achieved with equality when $\hat{\boldsymbol{\theta}}_i = \mathbb{E}[\boldsymbol{\theta}_i | X_i]$. The distribution on $\boldsymbol{\theta}_i$ given the local dataset X_i is given by:

$$\begin{aligned}
\Pr[\boldsymbol{\theta}_i = \boldsymbol{\mu}_l | X_i] &= \frac{f(X_i | \boldsymbol{\theta}_i = \boldsymbol{\mu}_l) \Pr[\boldsymbol{\theta}_i = \boldsymbol{\mu}_l]}{f(X_i)} \\
&= \frac{f(X_i | \boldsymbol{\theta}_i = \boldsymbol{\mu}_l) \Pr[\boldsymbol{\theta}_i = \boldsymbol{\mu}_l]}{\sum_{s=1}^k f(X_i | \boldsymbol{\theta}_i = \boldsymbol{\mu}_s) \Pr[\boldsymbol{\theta}_i = \boldsymbol{\mu}_s]} \\
&= \frac{p_l \exp\left(-\frac{\sum_{j=1}^n \|X_{ij}-\boldsymbol{\mu}_l\|^2}{2\sigma_x^2}\right)}{\sum_{s=1}^k p_s \exp\left(-\frac{\sum_{j=1}^n \|X_{ij}-\boldsymbol{\mu}_s\|^2}{2\sigma_x^2}\right)} = \alpha_l^{(i)}
\end{aligned} \tag{B.33}$$

As a result, the optimal estimator is given by:

$$\hat{\boldsymbol{\theta}}_i = \mathbb{E}[\boldsymbol{\theta}_i | X_i] = \sum_{l=1}^k \alpha_l^{(i)} \boldsymbol{\mu}_l. \tag{B.34}$$

This completes the proof of Theorem 18. □

The optimal personalized estimation in (B.30) is a weighted summation over all possible candidates vectors $\boldsymbol{\mu}_1, \ldots, \boldsymbol{\mu}_k$, where the weight $\alpha_l^{(i)}$ increases if the prior p_l increases and/or

the local samples $\{X_{ij}\}$ are close to the model $\boldsymbol{\mu}_l$ for $l \in [k]$. Observe that the optimal estimator $\hat{\boldsymbol{\theta}}_i$ in Theorem 18 that minimizes the MSE is completely different from the local estimator $\left(\frac{1}{n}\sum_{j=1}^{n} X_{ij}\right)$. Furthermore, it is easy to see that the local estimator has the MSE $\left(\frac{d\sigma_x^2}{n}\right)$ which increases linearly with the data dimension d. On the other hand, the MSE of the optimal estimator in Theorem 18 is a function of the prior distribution $\mathbb{P} = [p_1, \ldots, p_k]$, the prior vectors $\boldsymbol{\mu}_1, \ldots, \boldsymbol{\mu}_k$, and the local variance σ_x^2.

B.5.2 When the Prior Distribution is Unknown

Now, we consider a more practical case when the prior distribution $\mathbb{P} = [p_1, \ldots, p_k]$ and the candidates $\boldsymbol{\mu}_1, \ldots, \boldsymbol{\mu}_k$ are unknown to the clients. In this case, the clients collaborate with each other by their local data to estimate the priors $\mathbb{P}$ and $\boldsymbol{\mu}_1, \ldots, \boldsymbol{\mu}_k$, and then, each client uses the estimated priors to design her personalized model as in (B.30).

We present Algorithm 10 based on alternating minimization. The algorithm starts by initializing the local models $\{\boldsymbol{\theta}_i^{(0)} := \frac{1}{n}\sum_{j=1}^{n} X_{ij}\}$. Then, the algorithm works in rounds alternating between estimating the priors $\mathbb{P}^{(t+1)} = [p_1^{(t+1)}, \ldots, p_k^{(t+1)}]$, $\boldsymbol{\mu}_1^{(t+1)}, \ldots, \boldsymbol{\mu}_k^{(t+1)}$ for given local models $\{\boldsymbol{\theta}_i^{(t)}\}$ and estimating the personalized models $\{\boldsymbol{\theta}_i^{(t+1)}\}$ for given global priors $\mathbb{P}^{(t+1)}$ and $\boldsymbol{\mu}_1^{(t+1)}, \ldots, \boldsymbol{\mu}_k^{(t+1)}$. Observe that for given the prior information $\mathbb{P}^{(t)}, \{\boldsymbol{\mu}_l^t\}$, each client updates her personalized model in Step 6 which is the optimal estimator for given priors according to Theorem 18. On the other hand, for given personalized models $\{\boldsymbol{\theta}_i^{(t)}\}$, we estimate the priors $\mathbb{P}^{(t)}, \{\boldsymbol{\mu}_l^t\}$ using clustering algorithm with k sets in Step 11. The algorithm Cluster takes m vectors $\boldsymbol{a}_1, \ldots, \boldsymbol{a}_m$ and an integer k as its input, and its goal is to generate a set of k cluster centers $\boldsymbol{\mu}_1, \ldots, \boldsymbol{\mu}_k$ that minimizes $\sum_{i=1}^{m} \min_{l \in k} \|\boldsymbol{a}_i - \boldsymbol{\mu}_l\|^2$. Furthermore, these clustering algorithms can also return the prior distribution $\mathbb{P}$, by setting $p_l := \frac{|\mathcal{S}_l|}{m}$, where $\mathcal{S}_l \subset \{\boldsymbol{a}_1, \ldots, \boldsymbol{a}_m\}$ denotes the set of vectors that are belongs to the l-th cluster. There are lots of algorithms that do clustering, but perhaps, Lloyd's algorithm [107] and Ahmadian [2] are the most common algorithms for k-means clustering. Our Algorithm 10 can work with any clustering algorithm.

Algorithm 10 Alternating Minimization for Personalized Estimation

Input: Number of iterations T, local datasets $(X_{i1}, \ldots, X_{in})$ for $i \in [m]$.

1: **Initialize** $\boldsymbol{\theta}_i^0 = \frac{1}{n}\sum_{j=1}^n X_{ij}$ for $i \in [m]$.

2: **for** $t = 1$ **to** T **do**

3: **O**n Clients:

4: **for** $i = 1$ **to** m: **do**

5: Receive $\mathbb{P}^{(t)}, \boldsymbol{\mu}_1^{(t)}, \ldots, \boldsymbol{\mu}_k^{(t)}$ from the server

6: Update the personalized model:

$$\boldsymbol{\theta}_i^t \leftarrow \sum_{l=1}^k \alpha_l^{(i)} \boldsymbol{\mu}_l^{(t)} \qquad \text{and} \qquad \alpha_l^{(i)} = \frac{p_l^{(t)} \exp\left(-\frac{\sum_{j=1}^n \|X_{ij} - \boldsymbol{\mu}_l^{(t)}\|^2}{2\sigma_x^2}\right)}{\sum_{s=1}^k p_s^{(t)} \exp\left(-\frac{\sum_{j=1}^n \|X_{ij} - \boldsymbol{\mu}_s^{(t)}\|^2}{2\sigma_x^2}\right)}$$

7: Send $\boldsymbol{\theta}_i^t$ to the server

8: **end for**

9: **A**t the Server:

10: Receive $\boldsymbol{\theta}_1^{(t)}, \ldots, \boldsymbol{\theta}_m^{(t)}$ from the clients

11: Update the global parameters: $\mathbb{P}^{(t)}, \boldsymbol{\mu}_1^{(t)}, \ldots, \boldsymbol{\mu}_k^{(t)} \leftarrow \mathsf{Cluster}\left(\boldsymbol{\theta}_1^{(t)}, \ldots, \boldsymbol{\theta}_m^{(t)}, k\right)$

12: Broadcast $\mathbb{P}^{(t)}, \boldsymbol{\mu}_1^{(t)}, \ldots, \boldsymbol{\mu}_k^{(t)}$ to all clients

13: **end for**

Output: Personalized models $\boldsymbol{\theta}_1^T, \ldots, \boldsymbol{\theta}_m^T$.

B.5.3 Privacy/Communication Constraints

In the personalized estimation Algorithm 10, each client shares her personalized estimator $\boldsymbol{\theta}_i^{(t)}$ to the server at each iteration which is not communication-efficient and violates the privacy. In this section we present ideas on how to design communication-efficient and/or private Algorithms for personalized estimation.

Lemma 10. *Let $\boldsymbol{\mu}_1, \dots \boldsymbol{\mu}_k \in \mathbb{R}^d$ be unknown means such that $\|\boldsymbol{\mu}_i\|_2 \leq r$ for each $i \in [k]$. Let $\boldsymbol{\theta}_1, \dots, \boldsymbol{\theta}_m \sim \mathbb{P}$, where $\mathbb{P} = [p_1, \dots, p_k]$ and $p_l = \Pr[\boldsymbol{\theta}_i = \boldsymbol{\mu}_l]$. For $i \in [m]$, let $X_{i1}, \dots, X_{in} \sim \mathcal{N}(\boldsymbol{\theta}_i, \sigma_x^2)$, i.i.d. Then, with probability at least $1 - \frac{1}{mn}$, the following bound holds for all $i \in [m]$:*

$$\left\| \frac{1}{n} \sum_{j=1}^{n} X_{ij} \right\|_2 \leq 4\sqrt{d\frac{\sigma_x^2}{n}} + 2\sqrt{\log(m^2 n)\frac{\sigma_x^2}{n}} + r. \tag{B.35}$$

Proof. Observe that the vector $(\overline{X}_i - \theta_i) = \frac{1}{n}\sum_{i=1}^{n} X_{ij} - \theta_i$ is a sub-Gaussian random vector with proxy $\frac{\sigma_x^2}{n}$. As a result, we have that:

$$\|\overline{X}_i - \theta_i\|_2 \leq 4\sqrt{d\frac{\sigma_x^2}{n}} + 2\sqrt{\log(1/\eta)\frac{\sigma_x^2}{n}}, \tag{B.36}$$

with probability at least $1 - \eta$ from [170]. Since $\mu_1, \dots, \mu_k \in \mathbb{R}^d$ are such that $\|\boldsymbol{\mu}_i\|_2 \leq r$ for each $i \in [k]$, we have:

$$\|\overline{X}_i\|_2 \leq 4\sqrt{d\frac{\sigma_x^2}{n}} + 2\sqrt{\log(1/\eta)\frac{\sigma_x^2}{n}} + r, \tag{B.37}$$

with probability $1 - \eta$ from the triangular inequality. Thus, by choosing $\eta = \frac{1}{m^2 n}$ and using the union bound, this completes the proof of Lemma 10. □

Lemma 10 shows that the average of the local samples $\{\overline{X}_i\}$ has a bounded ℓ_2 norm with high probability. Thus, we can design a communication-efficient estimation Algorithm as follows: Each client clips her personal model $\boldsymbol{\theta}_i^{(t)}$ within radius $4\sqrt{d\frac{\sigma_x^2}{n}} + 2\sqrt{\log(m^2 n)\frac{\sigma_x^2}{n}} + r$. Then, each client applies a vector-quantization scheme (e.g., [3, 14, 59]) to the clipped vector before sending it to the server.

To design a private estimation algorithm with discrete priors, each client clips her personalized estimator $\boldsymbol{\theta}_i^{(t)}$ within radius $4\sqrt{d\frac{\sigma_x^2}{n}} + 2\sqrt{\log(m^2 n)\frac{\sigma_x^2}{n}} + r$. Then, we can use a differentially private algorithm for clustering (see e.g., [161] for clustering under LDP constraints and [54] for clustering under central DP constraints.). Since, we run T iterations in Algorithm 10, we can obtain the final privacy analysis (ϵ, δ) using the strong composition theorem [46].

B.6 Personalized Learning – Linear Regression

In this section, we present the personalized linear regression problem. Consider A set of m clients, where the i-th client has a local dataset consisting of n samples $(X_{i1}, Y_{i1}), \ldots, (X_{in}, Y_{in})$, where $X_{ij} \in \mathbb{R}^d$ denotes the feature vector and $Y_{ij} \in \mathbb{R}$ denotes the corresponding response. Let $Y_i = (Y_{i1}, \ldots, Y_{i1}) \in \mathbb{R}^n$ and $X_i = (X_{i1}, \ldots, X_{in}) \in \mathbb{R}^{n \times d}$ denote the response vector and the feature matrix at the i-th client, respectively. Following the standard regression, we assume that the response vector Y_i is obtained from a linear model as follows:

$$Y_i = X_i \boldsymbol{\theta}_i + \boldsymbol{w}_i, \tag{B.38}$$

where $\boldsymbol{\theta}_i$ denotes personalized model of the i-th client and $\boldsymbol{w}_i \sim \mathcal{N}(0, \sigma_x^2 \mathbb{I}_n)$ is a noise vector. The clients' parameters $\boldsymbol{\theta}_1, \ldots, \boldsymbol{\theta}_m$ are drawn i.i.d. from a Gaussian distribution $\boldsymbol{\theta}_1, \ldots, \boldsymbol{\theta}_m \sim \mathcal{N}(\boldsymbol{\mu}, \sigma_\theta^2 \mathbb{I}_d)$, i.i.d.

Our goal is to solve the optimization problem stated in (3.9) (for the linear regression setup) and learn the optimal personalized parameters $\{\widehat{\boldsymbol{\theta}}_i\}$. The following theorem characterizes the exact form of the optimal $\{\widehat{\boldsymbol{\theta}}_i\}$ and computes their minimum mean squared error w.r.t. the true parameters $\{\boldsymbol{\theta}_i\}$.

Theorem 19. *The optimal personalized parameters at client i with known $\boldsymbol{\mu}, \sigma_\theta^2, \sigma_x^2$ is given by:*

$$\widehat{\boldsymbol{\theta}}_i = \left(\frac{\mathbb{I}}{\sigma_\theta^2} + \frac{X_i^T X_i}{\sigma_x^2}\right)^{-1} \left(\frac{X_i^T Y_i}{\sigma_x^2} + \frac{\boldsymbol{\mu}}{\sigma_\theta^2}\right). \tag{B.39}$$

The mean squared error (MSE) of the above $\widehat{\boldsymbol{\theta}}_i$ is given by:

$$\mathbb{E}_{\mathbf{w}_i,\boldsymbol{\theta}_i}\left\|\widehat{\boldsymbol{\theta}}_i-\boldsymbol{\theta}_i\right\|^2=\mathsf{Tr}\left(\left(\frac{\mathbb{I}}{\sigma_\theta^2}+\frac{X_i^TX_i}{\sigma_x^2}\right)^{-1}\right), \tag{B.40}$$

Proof. The personalized model with perfect prior is obtained by solving the optimization problem stated in (3.9), which is given below for convenience. Note that for linear regression with Gaussian prior, we have $\mathbb{P}(\Gamma)\equiv\mathcal{N}(\boldsymbol{\mu},\sigma_\theta^2\mathbb{I}_d)$ and $p_{\boldsymbol{\theta}_i}(Y_{ij}|X_{ij})$ according to $\mathcal{N}(\boldsymbol{\theta}_i,\sigma_x^2)$.

$$\begin{aligned}\widehat{\boldsymbol{\theta}}_i&=\arg\min_{\boldsymbol{\theta}_i}\sum_{j=1}^{n}-\log(p_{\boldsymbol{\theta}_i}(Y_{ij}|X_{ij}))-\log(p(\boldsymbol{\theta}_i)).\\&=\arg\min_{\boldsymbol{\theta}_i}\sum_{j=1}^{n}\frac{(Y_{ij}-X_{ij}\boldsymbol{\theta}_i)^2}{2\sigma_x^2}+\frac{\|\boldsymbol{\theta}_i-\boldsymbol{\mu}\|^2}{2\sigma_\theta^2}.\\&=\arg\min_{\boldsymbol{\theta}_i}\frac{\|Y_i-X_i\boldsymbol{\theta}_i\|^2}{2\sigma_x^2}+\frac{\|\boldsymbol{\theta}_i-\boldsymbol{\mu}\|^2}{2\sigma_\theta^2}.\end{aligned}$$

By taking the derivative with respect to $\boldsymbol{\theta}_i$, we get

$$\frac{\partial}{\partial\boldsymbol{\theta}_i}=\frac{X_i^T(X_i\boldsymbol{\theta}_i-Y_i)}{\sigma_x^2}+\frac{\boldsymbol{\theta}_i-\boldsymbol{\mu}}{\sigma_\theta^2}. \tag{B.41}$$

Equating the above partial derivative to zero, we get that the optimal personalized parameters $\widehat{\boldsymbol{\theta}}_i$ is given by:

$$\widehat{\boldsymbol{\theta}}_i=\left(\frac{\mathbb{I}}{\sigma_\theta^2}+\frac{X_i^TX_i}{\sigma_x^2}\right)^{-1}\left(\frac{X_i^TY_i}{\sigma_x^2}+\frac{\boldsymbol{\mu}}{\sigma_\theta^2}\right). \tag{B.42}$$

Taking the expectation w.r.t. w_i, we get:

$$\mathbb{E}_{\mathbf{w}_i}[\widehat{\boldsymbol{\theta}}_i]=\left(\frac{\mathbb{I}}{\sigma_\theta^2}+\frac{X_i^TX_i}{\sigma_x^2}\right)^{-1}\left(\frac{X_i^TX_i\boldsymbol{\theta}_i}{\sigma_x^2}+\frac{\boldsymbol{\mu}}{\sigma_\theta^2}\right), \tag{B.43}$$

Thus, we can bound the MSE as following:

$$\begin{aligned}\mathbb{E}_{\mathbf{w}_i,\boldsymbol{\theta}_i}\left\|\widehat{\boldsymbol{\theta}}_i-\boldsymbol{\theta}_i\right\|^2&=\mathbb{E}_{\mathbf{w}_i,\boldsymbol{\theta}_i}\left\|\widehat{\boldsymbol{\theta}}_i-\mathbb{E}_{\mathbf{w}_i}[\widehat{\boldsymbol{\theta}}_i]+\mathbb{E}_{\mathbf{w}_i}[\widehat{\boldsymbol{\theta}}_i]-\boldsymbol{\theta}_i\right\|^2\\&=\mathbb{E}_{\mathbf{w}_i,\boldsymbol{\theta}_i}\left\|\widehat{\boldsymbol{\theta}}_i-\mathbb{E}_{\mathbf{w}_i}[\widehat{\boldsymbol{\theta}}_i]\right\|^2+\mathbb{E}_{\mathbf{w}_i,\boldsymbol{\theta}_i}\left\|\mathbb{E}_{\mathbf{w}_i}[\widehat{\boldsymbol{\theta}}_i]-\boldsymbol{\theta}_i\right\|^2+2\mathbb{E}_{\mathbf{w}_i,\boldsymbol{\theta}_i}\left\langle\widehat{\boldsymbol{\theta}}_i-\mathbb{E}_{\mathbf{w}_i}[\widehat{\boldsymbol{\theta}}_i],\mathbb{E}_{\mathbf{w}_i}[\widehat{\boldsymbol{\theta}}_i]-\boldsymbol{\theta}_i\right\rangle\\&=\mathbb{E}_{\mathbf{w}_i,\boldsymbol{\theta}_i}\left\|\widehat{\boldsymbol{\theta}}_i-\mathbb{E}_{\mathbf{w}_i}[\widehat{\boldsymbol{\theta}}_i]\right\|^2+\mathbb{E}_{\mathbf{w}_i,\boldsymbol{\theta}_i}\left\|\mathbb{E}_{\mathbf{w}_i}[\widehat{\boldsymbol{\theta}}_i]-\boldsymbol{\theta}_i\right\|^2\end{aligned}$$

In the last equality, we used

$$\mathbb{E}_{\mathbf{w}_i,\boldsymbol{\theta}_i}\left\langle \widehat{\boldsymbol{\theta}}_i - \mathbb{E}_{\mathbf{w}_i}[\widehat{\boldsymbol{\theta}}_i], \mathbb{E}_{\mathbf{w}_i}[\widehat{\boldsymbol{\theta}}_i] - \boldsymbol{\theta}_i \right\rangle = \mathbb{E}_{\boldsymbol{\theta}_i}\left\langle \mathbb{E}_{\mathbf{w}_i}[\widehat{\boldsymbol{\theta}}_i] - \mathbb{E}_{\mathbf{w}_i}[\widehat{\boldsymbol{\theta}}_i], \mathbb{E}_{\mathbf{w}_i}[\widehat{\boldsymbol{\theta}}_i] - \boldsymbol{\theta}_i \right\rangle = 0,$$

where the first equality holds because $\mathbb{E}_{\mathbf{w}_i}[\widehat{\boldsymbol{\theta}}_i] - \boldsymbol{\theta}_i$ is independent of $\mathbf{w}_i$.

Letting $\boldsymbol{M} = \frac{\mathbb{I}}{\sigma_\theta^2} + \frac{X_i^T X_i}{\sigma_x^2}$, and Tr denoting the trace operation, we get

$$\begin{aligned}
\mathbb{E}_{\mathbf{w}_i,\boldsymbol{\theta}_i}\left\|\widehat{\boldsymbol{\theta}}_i - \boldsymbol{\theta}_i\right\|^2 &= \mathsf{Tr}\left(\boldsymbol{M}^{-1}\mathbb{E}_{\mathbf{w}_i}\left[\left(\frac{X_i^T\mathbf{w}_i}{\sigma_x^2}\right)\left(\frac{X_i^T\mathbf{w}_i}{\sigma_x^2}\right)^T\right]\boldsymbol{M}^{-1}\right) \\
&\qquad + \mathsf{Tr}\left(\boldsymbol{M}^{-1}\mathbb{E}_{\boldsymbol{\theta}_i}\left[\left(\frac{\boldsymbol{\theta}_i - \boldsymbol{\mu}}{\sigma_\theta^2}\right)\left(\frac{\boldsymbol{\theta}_i - \boldsymbol{\mu}}{\sigma_\theta^2}\right)^T\right]\boldsymbol{M}^{-1}\right) \\
&= \mathsf{Tr}\left(\boldsymbol{M}^{-1}\frac{X_i^T X_i}{\sigma_x^2}\boldsymbol{M}^{-1}\right) + \mathsf{Tr}\left(\boldsymbol{M}^{-1}\frac{\mathbb{I}}{\sigma_\theta^2}\boldsymbol{M}^{-1}\right) \\
&= \mathsf{Tr}\left(\boldsymbol{M}^{-1}\right).
\end{aligned}$$

This completes the proof of Theorem 19. □

Observe that the local model of the i-th client, i.e., estimating $\boldsymbol{\theta}_i$ only from the local data (Y_i, X_i), is given by:

$$\widehat{\boldsymbol{\theta}}_i^{(l)} = \left(X_i^T X_i\right)^{-1} X_i^T Y_i, \tag{B.44}$$

where we assume the matrix $X_i^T X_i$ has a full rank (otherwise, we take the pseudo inverse). This local estimate achieves the MSE given by:

$$\mathbb{E}\left\|\widehat{\boldsymbol{\theta}}_i^{(l)} - \boldsymbol{\theta}_i\right\|^2 = \mathsf{Tr}\left(\left(X_i^T X_i\right)^{-1}\right)\sigma_x^2, \tag{B.45}$$

we can prove it by following similar steps as the proof of Theorem 19. When $\sigma_\theta^2 \to \infty$, we can easily see that the local estimate (B.44) matches the personalized estimate in (B.39).

To make the regression problem more practical, we assume that the mean $\boldsymbol{\mu}$, the local variance σ_x^2, and the global variance σ_θ^2 are unknown. Hence, we estimate the personalized parameters by minimizing the negative log likelihood:

$$\widehat{\boldsymbol{\theta}}_1, \ldots, \widehat{\boldsymbol{\theta}}_m = \underset{\{\boldsymbol{\theta}_i\},\boldsymbol{\mu},\sigma_x^2,\sigma_\theta^2}{\arg\min} \sum_{i=1}^m \sum_{j=1}^n -\log\left(p_{\boldsymbol{\theta}_i}\left(Y_{ij}|X_{ij}\right)\right) + \sum_{i=1}^m -\log\left(p\left(\boldsymbol{\theta}_i\right)\right)$$

$$= \arg\min \frac{nm}{2} \log(2\pi\sigma_x^2) + \sum_{i=1}^{m}\sum_{j=1}^{n} \frac{(Y_{ij} - X_{ij}\theta_i)^2}{2\sigma_x^2} + \frac{md}{2} \log(2\pi\sigma_\theta^2) + \sum_{i=1}^{m} \frac{\|\boldsymbol{\theta}_i - \boldsymbol{\mu}\|^2}{2\sigma_\theta^2}. \tag{B.46}$$

Instead of solving the above optimization problem explicitly, we can optimize it through gradient descent (GD) and the resulting algorithm is presented in Algorithm 11. In addition to keeping the personalized models $\{\boldsymbol{\theta}_i^t\}$, each client also maintains local copies of $\{\boldsymbol{\mu}_i^t, \sigma_{\theta,i}^t, \sigma_{x,i}^t\}$ and updates all these parameters by taking appropriate gradients of the objective in (B.46) and synchronize them with the server to update the global copy of these parameters $\{\boldsymbol{\mu}^t, \sigma_\theta^t, \sigma_x^t\}$.

B.7 Personalized Learning – Logistic Regression

As described in Section 3.3, by taking $\mathbb{P}(\Gamma) \equiv \mathcal{N}(\boldsymbol{\mu}, \sigma_\theta^2 \mathbb{I}_d)$ and $p_{\theta_i}(Y_{ij}|X_{ij}) = \sigma(\langle \boldsymbol{\theta}_i, X_{ij}\rangle)^{Y_{ij}}(1 - \sigma(\langle \boldsymbol{\theta}_i, X_{ij}\rangle))^{(1-Y_{ij})}$, where $\sigma(z) = 1/1+e^{-z}$ for any $z \in \mathbb{R}$, then the overall optimization problem becomes:

$$\arg\min_{\{\boldsymbol{\theta}_i\},\boldsymbol{\mu},\sigma_\theta} \sum_{i=1}^{m}\sum_{j=1}^{n} \left[Y_{ij} \log\left(\frac{1}{1+e^{-\langle \boldsymbol{\theta}_i, X_{ij}\rangle}}\right) + (1 - Y_{ij}) \log\left(\frac{1}{1+e^{\langle \boldsymbol{\theta}_i, X_{ij}\rangle}}\right)\right] + \frac{md}{2} \log(2\pi\sigma_\theta^2) + \sum_{i=1}^{m} \frac{\|\boldsymbol{\mu} - \boldsymbol{\theta}_i\|_2^2}{2\sigma_\theta^2}. \tag{B.47}$$

When $\boldsymbol{\mu}$ and σ_θ^2 are unknown, we would like to learn them by gradient descent, as in the linear regression case. The corresponding algorithm is described in Algorithm 12.

Algorithm 12 Logistic Regression SGD

Input: Number of iterations T, local datasets (Y_i, X_i) for $i \in [m]$, learning rate η.

Initialize $\boldsymbol{\theta}_i^0$ for $i \in [m]$, $\boldsymbol{\mu}^0$, $\sigma_\theta^{2,0}$.
for $t = 1$ **to** T **do**
  On Clients:
  for $i = 1$ **to** m: **do**
    Receive $(\boldsymbol{\mu}^t, \sigma_\theta^{2,t})$ from the server and set $\boldsymbol{\mu}_i^t := \boldsymbol{\mu}^t, \sigma_{\theta,i}^{2,t} := \sigma_\theta^{2,t}$
    Update the personalized model:

$$\boldsymbol{\theta}_i^t \leftarrow \boldsymbol{\theta}_i^{t-1} - \eta \left(\sum_{j=1}^{n} \nabla_{\theta_i^{t-1}} l_{CE}^{(p)}(\boldsymbol{\theta}_i^{t-1}, (X_i^j, Y_i^j)) + \frac{\boldsymbol{\mu}_i^{t-1} - \boldsymbol{\theta}_i^{t-1}}{\sigma_{\theta,i}^{2,t-1}} \right),$$

    where $l_{CE}^{(p)}$ denotes the cross-entropy loss.
    Update local version of mean: $\boldsymbol{\mu}_i^t \leftarrow \boldsymbol{\mu}_i^{t-1} - \eta \left(\frac{\boldsymbol{\mu}_i^{t-1} - \boldsymbol{\theta}_i^{t-1}}{\sigma_{\theta,i}^{2,t-1}} \right)$
    Update global variance: $\sigma_{\theta,i}^{2,t} \leftarrow \sigma_{\theta,i}^{2,t-1} - \eta \left(\frac{d}{2\sigma_{\theta,i}^{2,t-1}} - \frac{\|\boldsymbol{\mu}_i^{t-1} - \boldsymbol{\theta}_i^{t-1}\|^2}{2(\sigma_{\theta,i}^{2,t-1})^2} \right)$
    Send $(\boldsymbol{\mu}_i^t, \sigma_{\theta,i}^{2,t})$ to the server
  end for
  At the Server:
  Receive $\{(\boldsymbol{\mu}_i^t, \sigma_{\theta,i}^{2,t})\}$ from the clients
  Aggregate mean: $\boldsymbol{\mu}^t = \frac{1}{m} \sum_{i=1}^m \boldsymbol{\mu}_i^t$
  Aggregate global variance: $\sigma_\theta^{2,t} = \frac{1}{m} \sum_{i=1}^m \sigma_{\theta,i}^{2,t}$
  Broadcast $(\boldsymbol{\mu}^t, \sigma_\theta^{2,t})$ to all clients
end for

Output: Personalized models $\boldsymbol{\theta}_1^T, \ldots, \boldsymbol{\theta}_m^T$.

B.8 Personalized Learning – Mixture Model

In this section, we present the linear regression problem as a generalization to the estimation problem with discrete priors. This model falls into the framework studied in [115] and is illustrated to show how our framework also captures it.

Consider a set of m clients, where the i-th client has a local dataset $(X_{i1}, Y_{i1}), \ldots, (X_{in}, Y_{in})$ of m samples, where $X_{ij} \in \mathbb{R}^d$ denotes the feature vector and $Y_{ij} \in \mathbb{R}$ denotes the corresponding response. Let $Y_i = (Y_{i1}, \ldots, Y_{i1}) \in \mathbb{R}^n$ and $X_i = (X_{i1}, \ldots, X_{in}) \in \mathbb{R}^{n \times d}$ denote the response vector and the feature matrix at the i-th client, respectively. Following the standard regression, we assume that the response vector Y_i is obtained from a linear model as follows:

$$Y_i = X_i \boldsymbol{\theta}_i + \mathbf{w}_i, \tag{B.48}$$

where $\boldsymbol{\theta}_i$ denotes personalized model of the i-th client and $\mathbf{w}_i \sim \mathcal{N}(0, \sigma_x^2 \mathbb{I}_n)$ is a noise vector. The clients models are drawn i.i.d. from a discrete distribution $\boldsymbol{\theta}_1, \ldots, \boldsymbol{\theta}_m \sim \mathbb{P}$, where $\mathbb{P} = [p_1, \ldots, p_k]$ such that $p_l = \Pr[\boldsymbol{\theta}_i = \boldsymbol{\mu}_l]$ for $i \in [m]$ and $l \in [k]$.

Our goal is to solve the optimization problem stated in (3.9) (for the linear regression with the above discrete prior) and learn the optimal personalized parameters $\{\widehat{\boldsymbol{\theta}}_i\}$.

We assume that the discrete distribution $\mathbb{P}$ and the prior candidates $\{\boldsymbol{\mu}_l\}_{l=1}^k$ are unknown to the clients. Inspired from Algorithm 10 for estimation with discrete priors, we obtain Algorithm 13 for learning with discrete prior. Note that this is *not* a new algorithm, and is essentially the algorithm proposed in [115] applied to linear regression. We wanted to show how our framework captures mixture model in [115] through this example.

Description of Algorithm 13. Client i initializes its personalized parameters $\boldsymbol{\theta}_i^{(0)} = (X_i^T X_i)^{-1} X_i^T Y_i$, which is the optimal as a function of the local dataset at the i-th client without any prior knowledge. In any iteration t, for a given prior information $\mathbb{P}^{(t)}$, $\{\boldsymbol{\mu}_l^{(t)}\}$, the i-th client updates the personalized model as $\boldsymbol{\theta}_i^t = \sum_{l=1}^k \alpha_l^{(i)} \boldsymbol{\mu}_l^{(t)}$, where the weights $\alpha_l^{(i)} \propto p_l^{(t)} \exp\left(-\frac{\|X_i \boldsymbol{\mu}_l^{(t)} - Y_i\|^2}{2\sigma_x^2}\right)$ and sends its current estimate of the personalized parameter

$\boldsymbol{\theta}_i^t$ to the server. Upon receiving $\boldsymbol{\theta}_1^t, \ldots, \boldsymbol{\theta}_m^t$, server will run Cluster algorithm to update the global parameters $\mathbb{P}, \boldsymbol{\mu}_1^{(t)}, \ldots, \boldsymbol{\mu}_k^{(t)}$, and broadcasts them to the clients.

B.9 Personalized Learning – AdaPeD

B.9.1 Knowledge Distillation Population Distribution

In this section we discuss what type of a population distribution can give rise to algorithms/problems that include a knowledge distillation (KD) (or KL divergence) penalty term between local and global models. From Section 3.3, Equation (3.9), consider $p_{\boldsymbol{\theta}_i}(y|x)$ as a randomized mapping from input space $\mathcal{X}$ to output class $\mathcal{Y}$, parameterized by $\boldsymbol{\theta}_i$. For simplicity, consider the case where $|\mathcal{X}|$ is finite, e.g. for MNIST it could be all possible 28×28 black and white images. Every $p_{\boldsymbol{\theta}_i}(y|x)$ corresponds to a probability matrix (parameterized by $\boldsymbol{\theta}_i$) of size $|\mathcal{Y}| \times |\mathcal{X}|$, where the (y, x)'th represents the probability of the class y (row) given the data sample x (column). Therefore, each column is a probability vector. Since we want to sample the probability matrix, it suffices to restrict our attention to any set of $|\mathcal{Y}| - 1$ rows, as the remaining row can be determined by these $|\mathcal{Y}| - 1$ rows.

Similarly, for a global parameter $\boldsymbol{\mu}$, let $p_{\boldsymbol{\mu}}(y|x)$ define a randomized mapping from $\mathcal{X}$ to $\mathcal{Y}$, parameterized by the global parameter $\boldsymbol{\mu}$. Note that for a fixed global parameter $\boldsymbol{\mu}$, the randomized map $p_{\boldsymbol{\mu}}(y|x)$ is fixed, whereas, our goal is to sample $p_{\boldsymbol{\theta}_i}(y|x)$ for $i = 1, \ldots, m$, one for each client. For simplicity of notation, define $p_{\boldsymbol{\theta}_i} := p_{\boldsymbol{\theta}_i}(y|x)$ and $p_{\boldsymbol{\mu}} := p_{\boldsymbol{\mu}}(y|x)$ to be the corresponding probability matrices, and let the distribution for sampling $p_{\boldsymbol{\theta}_i}(y|x)$ be denoted by $p_{p_{\boldsymbol{\mu}}}(p_{\boldsymbol{\theta}_i})$. Note that different mappings $p_{\boldsymbol{\theta}_i}(y|x)$ correspond to different $\boldsymbol{\theta}_i$'s, so we define $p(\boldsymbol{\theta}_i)$ (in Equation (3.9)) as $p_{p_{\boldsymbol{\mu}}}(p_{\boldsymbol{\theta}_i})$, which is the density of sampling the probability matrix $p_{\boldsymbol{\theta}_i}(y|x)$.

For the KD population distribution, we define this density $p_{p_{\boldsymbol{\mu}}}(p_{\boldsymbol{\theta}_i})$ as:

$$p_{p_{\boldsymbol{\mu}}}(p_{\boldsymbol{\theta}_i}) = c(\psi) e^{-\psi D_{\mathsf{KL}}(p_{\boldsymbol{\mu}}(y|x) \| p_{\boldsymbol{\theta}_i}(y|x))} \tag{B.49}$$

where ψ is an 'inverse variance' type of parameter, $c(\psi)$ is a normalizing function that depends on $(\psi, p_{\boldsymbol{\mu}})$, and $D_{\mathsf{KL}}(p_{\boldsymbol{\mu}}(y|x)\|p_{\boldsymbol{\theta}_i}(y|x))$
$= \sum_{x\in\mathcal{X}} p(x) \sum_{y\in\mathcal{Y}} p_{\boldsymbol{\mu}}(y|x) \log\left(\frac{p_{\boldsymbol{\mu}}(y|x)}{p_{\boldsymbol{\theta}_i}(y|x)}\right)$ is the conditional KL divergence, where $p(x)$ denotes the probability of sampling a data sample $x \in \mathcal{X}$. Now all we need is to find $c(\psi)$ given a fixed $\boldsymbol{\mu}$ (and therefore fixed $p_{\boldsymbol{\mu}}(y|x)$). Here we consider $D_{\mathsf{KL}}(p_{\boldsymbol{\mu}}\|p_{\boldsymbol{\theta}_i})$, but our analysis can be extended to $D_{\mathsf{KL}}(p_{\boldsymbol{\theta}_i}\|p_{\boldsymbol{\mu}})$ or $\|p_{\boldsymbol{\theta}_i} - p_{\boldsymbol{\mu}}\|_2$ as well.

For simplicity and to make the calculations easier, we consider a binary classification task with $\mathcal{Y} = \{0, 1\}$ and define $p_{\boldsymbol{\mu}}(x) := p_{\boldsymbol{\mu}}(y = 1|X = x)$ and $q_i(x) := p_{\boldsymbol{\theta}_i}(y = 1|X = x)$. We have:

$$D_{\mathsf{KL}}(p_{\boldsymbol{\mu}}(y|x)\|p_{\boldsymbol{\theta}_i}(y|x)) = \sum_x p(x)\Bigg(p_{\boldsymbol{\mu}}(x)(\log p_{\boldsymbol{\mu}}(x) - \log q_i(x)) \\ + (1 - p_{\boldsymbol{\mu}}(x))(\log(1 - p_{\boldsymbol{\mu}}(x)) - \log(1 - q_i(x)))\Bigg).$$

Hence, after some algebra we have,

$$p_{p_{\boldsymbol{\mu}}}(p_{\boldsymbol{\theta}_i}) = c(\psi)e^{\psi\sum_x p(x)H(p_{\boldsymbol{\mu}}(x))}e^{\psi\sum_x p(x)(p_{\boldsymbol{\mu}}(x)\log(q_i(x))+(1-p_{\boldsymbol{\mu}}(x))\log(1-q_i(x))))}$$

Then,

$$c(\psi)\prod_x\left[\int_0^1 e^{\psi p(x)H(p_{\boldsymbol{\mu}}(x))}e^{\psi p(x)(p_{\boldsymbol{\mu}}(x)\log(q_i(x))+(1-p_{\boldsymbol{\mu}}(x))\log(1-q_i(x))))}dq_i(x)\right] = 1.$$

Note that

$$\int_0^1 e^{\psi p(x)(p_{\boldsymbol{\mu}}(x)\log(q_i(x))+(1-p_{\boldsymbol{\mu}}(x))\log(1-q_i(x))))}dq_i(x) = B\left(1 + \frac{p_{\boldsymbol{\mu}}(x)}{\psi p(x)}, 1 + \frac{1 - p_{\boldsymbol{\mu}}(x)}{\psi p(x)}\right)$$

Accordingly, after some algebra, we can obtain $c(\psi) = \frac{e^{-\psi\sum_x p(x)H(p_{\boldsymbol{\mu}}(x))}}{\prod_x B\left(1+\frac{p_{\boldsymbol{\mu}}(x)}{\psi p(x)}, 1+\frac{1-p_{\boldsymbol{\mu}}(x)}{\psi p(x)}\right)}$, where H is binary Shannon entropy. Substituting this in (B.49), we get

$$p_{p_{\boldsymbol{\mu}}}(p_{\boldsymbol{\theta}_i}) = \frac{e^{-\psi\sum_x p(x)H(p_{\boldsymbol{\mu}}(x))}}{\prod_x B(1 + \frac{p_{\boldsymbol{\mu}}(x)}{\psi p(x)}, 1 + \frac{1-p_{\boldsymbol{\mu}}(x)}{\psi p(x)})}e^{-\psi D_{\mathsf{KL}}(p_{\boldsymbol{\mu}}(y|x)\|p_{\boldsymbol{\theta}_i}(y|x))}$$

which is the population distribution that can result in a KD type regularizer. Note that when we take the negative logarithm of the population distribution we obtain KL divergence loss

and an additional term that depends on ψ and $p_{\boldsymbol{\mu}}$. This is the form seen in Section 3.3.2 Equation (3.11) for AdaPeD algorithm. For numerical purpose, we take the additional term $-\log\left(\frac{e^{-\psi\sum_x p(x)H(p_{\boldsymbol{\mu}}(x))}}{\prod_x B(1+\frac{p_{\boldsymbol{\mu}}(x)}{\psi p(x)},1+\frac{1-p_{\boldsymbol{\mu}}(x)}{\psi p(x)})}\right)$ to be simple $\frac{1}{2}\log(2\psi)$. As mentioned in Section 3.3.2, this serves the purpose of regularizing ψ. This is in contrast to the objective considered in [131], which only has the KL divergence loss as the regularizer, without the additional term.

B.9.2 AdaPeD with Unsampled Client Iterations

When there is a flexibility in computational resources for doing local iterations, unsampled clients can do local training on their personalized models to speed-up convergence at no cost to privacy. This can be used in cross-silo settings, such as cross-institutional training for hospitals, where privacy is crucial and there are available computing resources most of the time. We propose the algorithm for AdaPeD with with unsampled client iterations in Algorithm 14:

Algorithm 14 Adaptive Personalization via Distillation (`AdaPeD`) with unsampled client iterations

Parameters: local variances $\{\psi_i^0\}$, personalized models $\{\boldsymbol{\theta}_i^0\}$, local copies of the global model $\{\boldsymbol{\mu}_i^0\}$, synchronization gap τ, learning rates η_1, η_2, η_3, number of sampled clients K.

for $t = 0$ **to** $T-1$ **do**
  if τ divides t **then**
    On Server do:
    Choose a subset $\mathcal{K}^t \subseteq [n]$ of K clients
    Broadcast $\boldsymbol{\mu}^t$ and ψ^t
    On Clients $i \in \mathcal{K}^t$ (in parallel) **do**:
    Receive $\boldsymbol{\mu}^t$ and ψ^t; set $\boldsymbol{\mu}_i^t = \boldsymbol{\mu}^t$, $\psi_i^t = \psi^t$
  end if
  On Clients $i \notin \mathcal{K}^t$ (in parallel) **do**:
  Compute $\mathbf{g}_i^t := \nabla_{\boldsymbol{\theta}_i^t} f_i(\boldsymbol{\theta}_i^t) + \frac{\nabla_{\boldsymbol{\theta}_i^t} f_i^{\mathsf{KD}}(\boldsymbol{\theta}_i^t, \boldsymbol{\mu}_i^{t_i'})}{2\psi_i^{t_i'}}$ where t_i' is the last time index where client i received global parameters from the server
  Update: $\boldsymbol{\theta}_i^{t+1} = \boldsymbol{\theta}_i^t - \eta_1 \mathbf{g}_i^t$
  On Clients $i \in \mathcal{K}^t$ (in parallel) **do**:
  Compute $\mathbf{g}_i^t := \nabla_{\boldsymbol{\theta}_i^t} f_i(\boldsymbol{\theta}_i^t) + \frac{\nabla_{\boldsymbol{\theta}_i^t} f_i^{\mathsf{KD}}(\boldsymbol{\theta}_i^t, \boldsymbol{\mu}_i^t)}{2\psi_i^t}$
  Update: $\boldsymbol{\theta}_i^{t+1} = \boldsymbol{\theta}_i^t - \eta_1 \mathbf{g}_i^t$
  Compute $\mathbf{h}_i^t := \frac{\nabla_{\boldsymbol{\mu}_i^t} f_i^{\mathsf{KD}}(\boldsymbol{\theta}_i^{t+1}, \boldsymbol{\mu}_i^t)}{2\psi_i^t}$
  Update: $\boldsymbol{\mu}_i^{t+1} = \boldsymbol{\mu}_i^t - \eta_2 \mathbf{h}_i^t$
  Compute $k_i^t := \frac{1}{2\psi_i^t} - \frac{f_i^{\mathsf{KD}}(\boldsymbol{\theta}_i^{t+1}, \boldsymbol{\mu}_i^{t+1})}{2(\psi_i^t)^2}$
  Update: $\psi_i^{t+1} = \psi_i^t - \eta_3 k_i^t$
  if τ divides $t+1$ **then**
    Clients send $\boldsymbol{\mu}_i^t$ and ψ_i^t to **Server**
    Server receives $\{\boldsymbol{\mu}_i^t\}_{i \in \mathcal{K}^t}$ and $\{\psi_i^t\}_{i \in \mathcal{K}^t}$
    Server computes $\boldsymbol{\mu}^{t+1} = \frac{1}{K} \sum_{i \in \mathcal{K}^t} \boldsymbol{\mu}_i^t$ and $\psi^{t+1} = \frac{1}{K} \sum_{i \in \mathcal{K}^t} \psi_i^t$
  end if
end for

Output: Personalized models $(\boldsymbol{\theta}_i^T)_{i=1}^m$

Of course, when a client is not sampled for a long period of rounds this approach can become similar to a local training; hence, it might be reasonable to put an upper limit on the successive number of local iterations for each client.

B.10 Personalized Learning – DP-AdaPeD

Proof of Theorem 7

Theorem (Restating Theorem 7)**.** *After T iterations,* `DP-AdaPeD` *satisfies $(\alpha, \epsilon(\alpha))$-RDP for $\alpha > 1$, where $\epsilon(\alpha) = \left(\frac{K}{m}\right)^2 6 \left(\frac{T}{\tau}\right) \alpha \left(\frac{C_1^2}{K\sigma_{q_1}^2} + \frac{C_2^2}{K\sigma_{q_2}^2}\right)$, where $\frac{K}{m}$ denotes the sampling ratio of the clients at each global iteration.*

Proof. In this section, we provide the privacy analysis of `DP-AdaPeD`. We first analyze the RDP of a single global round $t \in [T]$ and then, we obtain the results from the composition of the RDP over total T global rounds. Recall that privacy leakage can happen through communicating $\{\boldsymbol{\mu}_i\}$ and $\{\psi_i^t\}$ and we privatize both of these. In the following, we do the privacy analysis of privatizing $\{\boldsymbol{\mu}_i\}$ and a similar analysis could be done for $\{\psi_i^t\}$ as well.

At each synchronization round $t \in [T]$, the server updates the global model $\boldsymbol{\mu}^{t+1}$ as follows:

$$\boldsymbol{\mu}^{t+1} = \frac{1}{K} \sum_{i \in \mathcal{K}t} \boldsymbol{\mu}_i^t, \tag{B.50}$$

where $\boldsymbol{\mu}_i^t$ is the update of the global model at the i-th client that is obtained by running τ local iterations at the i-th client. At each of the local iterations, the client clips the gradient $\mathbf{h}_i^t$ with threshold C_1 and adds a zero-mean Gaussian noise vector with variance $\sigma_{q_1}^2 \mathbb{I}_d$. When neglecting the noise added at the local iterations, the norm-2 sensitivity of updating the global model $\boldsymbol{\mu}_i^{t+1}$ at the synchronization round t is bounded by:

$$\Delta\boldsymbol{\mu} = \max_{\mathcal{K}^t, \mathcal{K}'^t} \|\boldsymbol{\mu}^{t+1} - \boldsymbol{\mu}'^{t+1}\|_2^2 \leq \frac{\tau C_1^2}{K^2}, \tag{B.51}$$

where $\mathcal{K}^t, \mathcal{K}'^t \subset [m]$ are neighboring sets that differ in only one client. Additionally, $\boldsymbol{\mu}^{t+1} = \frac{1}{K}\sum_{i \in \mathcal{K}t} \boldsymbol{\mu}_i^t$ and $\boldsymbol{\mu}'^{t+1} = \frac{1}{K}\sum_{i \in \mathcal{K}'t} \boldsymbol{\mu}_i^t$. Since we add i.i.d. Gaussian noises with variance $\sigma_{q_1}^2$

at each local iteration at each client, and then, we take the average of theses vectors over K clients, it is equivalent to adding a single Gaussian vector to the aggregated vectors with variance $\frac{\tau\sigma_{q_1}^2}{K}$. Thus, from the RDP of the sub-sampled Gaussian mechanism in [122, Table 1], [20], we get that the global model $\boldsymbol{\mu}^{t+1}$ of a single global iteration of `DP-AdaPeD` is $(\alpha, \epsilon_t^{(1)}(\alpha))$-RDP, where $\epsilon_t(\alpha)$ is bounded by:

$$\epsilon_t^{(1)}(\alpha) = \left(\frac{K}{m}\right)^2 \frac{6\alpha C_1^2}{K\sigma_{q_1}^2}. \tag{B.52}$$

Similarly, we can show that the global parameter ψ^{t+1} at any synchronization round of `DP-AdaPeD` is $(\alpha, \epsilon_t^{(2)}(\alpha))$-RDP, where $\epsilon_t(\alpha)$ is bounded by:

$$\epsilon_t^{(2)}(\alpha) = \left(\frac{K}{m}\right)^2 \frac{6\alpha C_2^2}{K\sigma_{q_2}^2}. \tag{B.53}$$

Using adaptive RDP composition [121, Proposition 1], we get that each synchronization round of `DP-AdaPeD` is $(\alpha, \epsilon_t^{(1)}(\alpha) + \epsilon_t^{(2)}(\alpha))$-RDP. Thus, by running `DP-AdaPeD` over T/τ synchronization rounds and from the composition of the RDP, we get that `DP-AdaPeD` is $(\alpha, \epsilon(\alpha))$-RDP, where $\epsilon(\alpha) = \left(\frac{T}{\tau}\right)\left(\epsilon_t^{(1)}(\alpha) + \epsilon_t^{(2)}(\alpha)\right)$. This completes the proof of Theorem 7.

□

B.11 Expanded related work and connections to existing methods

In Section 5.1, we mentioned that the our framework has connections to several personalized FL methods. In this appendix we provide a few more details related to these connections.

Regularization: As noted earlier using (3.9) with the Gaussian population prior connects to the use of ℓ_2 regularizer in earlier personalized learning works [42, 64, 65, 100, 131], which also iterates between local and global model estimates. This form can be explicitly seen in Appendix B.6, where in Algorithm 11, we see that the Gaussian prior along with iterative optimization yields the regularized form seen in these methods. In these cases[1], $\mathbb{P}(\Gamma) \equiv \mathcal{N}(\boldsymbol{\mu}, \sigma_\theta^2 \mathbb{I}_d)$ for

[1]One can generalize these by including σ_θ^2 in the unknown parameters.

unknown parameters $\Gamma = \{\boldsymbol{\mu}\}$. Note that since the parameters of the population distribution are unknown, these need to be estimated during the iterative learning process. In the algorithm, 11 it is seen the $\boldsymbol{\mu}$ plays the role of the global model (and is truly so for the linear regression problem studied in Appendix B.6).

Clustered FL: If one uses a *discrete* mixture model for the population distribution then the iterative algorithm suggested by our framework connects to [56, 114, 115, 157, 185]. In particular, consider a population model with parameters in the m-dimensional probability simplex $\{\boldsymbol{\alpha} : \boldsymbol{\alpha} = [\alpha_1, \ldots, \alpha_k], \alpha_i \geq 0, \forall i, \sum_i \alpha_i = 1\}$ which describing a distribution. If there are m (unknown) discrete distributions $\{\mathcal{D}_1, \ldots, \mathcal{D}_m\}$, one can consider these as the unknown description of the population model in addition to $\boldsymbol{\alpha}$. Therefore, each local data are generated either as a mixture as in [115] or by choosing one of the unknown discrete distributions with probability $\boldsymbol{\alpha}$ dictating the probability of choosing $\mathcal{D}_i$, when hard clustering is used (*e.g.,* [114]). Each node j chooses a mixture probability $\boldsymbol{\alpha}^{(j)}$ uniformly from the m-dimensional probability simplex. In the former case, it uses this mixture probability to generate a local mixture distribution. In the latter it chooses $\mathcal{D}_i$ with probability $\boldsymbol{\alpha}_i^{(j)}$.

As mentioned earlier, not all parametrized distributions can be written as a mixture of *finite* number distributions, which is the assumption for discrete mixtures. Consider a unimodal Gaussian population distribution (as also studied in Appendix B.6). Since $\mathbb{P}(\Gamma) \equiv \mathcal{N}(\boldsymbol{\mu}, \sigma_\theta^2 \mathbb{I}_d)$, for node i, we sample $\boldsymbol{\theta}_i \sim \mathbb{P}(\Gamma)$. We see that the actual data distribution for this node is $p_{\boldsymbol{\theta}_i}(y|\boldsymbol{x}) = \mathcal{N}(\boldsymbol{\theta}_i^\top \boldsymbol{x}, \sigma^2)$. Clearly the set of such distributions $\{p_{\boldsymbol{\theta}_i}(y|\boldsymbol{x})\}$ *cannot* be written as any finite mixture as $\boldsymbol{\theta}_i \in \mathbb{R}^d$ and $p_{\boldsymbol{\theta}_i}(y|\boldsymbol{x})$ is a unimodal Gaussian distribution, with same parameter $\boldsymbol{\theta}_i$ for all data generated in node i. Essentially the generative framework of finite mixtures (as in [115]) could be restrictive as it does not capture such parametric models.

Knowledge distillation: The population distribution related to a regularizer based on Kullback-Leibler divergence (knowledge distillation) has been shown in Appendix B.9. Therefore this can be cast in terms of information geometry where the probability falls of exponen-

tially with in this geometry. Hence these connect to methods such as [96, 102, 131, 152], but the exact regularizer used does not take into account the full parametrization, and one can therefore improve upon these methods.

FL with Multi-task Learning (MTL): In this framework, a *fixed* relationship between tasks is usually assumed [157]. Therefore one can model this as a Gaussian model with *known* parameters relating the individual models. The individual models are chosen from a joint Gaussian with particular (known) covariance dictating the different models, and therefore giving the quadratic regularization used in FL-MTL [157]. In this the parameters of the Gaussian model are known and fixed.

Common representations: The works in [44, 75] use a linear model where $y \sim \mathcal{N}(\boldsymbol{x}^\top \boldsymbol{\theta}_i, \sigma^2)$ can be considered a local generative model for node i. The common representation approach assumes that $\boldsymbol{\theta}_i = \sum_{j=1}^{k} \boldsymbol{B}\boldsymbol{w}_j^{(i)}$, for some $k \ll d$, where $\boldsymbol{\theta}_i \in \mathbb{R}^d$. Therefore, one can parametrize a population by this (unknown) common basis $\boldsymbol{B}$, and under a mild assumption that the weights are bounded, we can choose a uniform measure in this bounded cube to choose $\boldsymbol{w}^{(i)}$ for each node i. The alternating optimization iteratively discovers the global common representation and the local weights as done in [44, 75] (and references therein). This common linear representation approach was generalized in [34, 44] to neural networks, where a set of parameters to obtain a common representation ("head") at each client was obtained and each local client appendd it with a "tail" combining the representation to obtain the final model. This also fits into our statistical framework, where the common representation (head) parameters are chosen from a population model (like the common subspace in the linear case) and the tail parameters are independently chosen (again as in the linear case).

Empirical and Hierarchical Bayes: As mentioned in Section 5.1, our work is inspired by empirical Bayes framework, introduced in [77, 145, 160], which is the origin of hierarchical Bayes methods; see also [52, pp. 132]. [77, 160] studied jointly estimating Gaussian individual parameters, generated by an unknown (parametrized) Gaussian population distribution. They showed a surprising result that one can enhance the estimate of individual parameters

based on the observations of a population of Gaussian random variables with *independently* generated parameters from an unknown (parametrized) Gaussian population distribution. Effectively, this methodology advocated *estimating* the unknown population distribution using the individual independent samples, and then using it effectively as an empirical prior for individual estimates.[2] This was studied for Bernoulli variables with heterogeneously generated individual parameters by [108] and the optimal error bounds for maximum likelihood estimates for population distributions were recently developed in [169]. Hierarchical Bayes, builds on empirical Bayes framework and is sometimes associated with a fully Bayes method. Our choice to use empirical Bayes framework as the foundation is also because it is more computationally feasible than a fully Bayes method. The subtle difference between the two is that empirical Bayes uses a point estimate of a (parametrized) prior, whereas, the terminology hierarchical Bayes often refers to a fully Bayes method where the (non-parametric) prior is estimated by computationally intensive methods like MCMC (see the discussion in [17]). As mentioned in Section 5.1, a contribution of our work is to connect a well studied statistical framework of empirical (hierarchical) Bayes to model heterogeneity in personalized federated learning. This statistical model yields a framework for personalized FL and leads to new algorithms and bounds especially in the local data starved regime.

B.12 Additional Details and Results for Experiments

B.12.1 Implementation Details

In this section we give further details on implementation and setting of the experiments that were used in Section 7.5.

CIFAR-100 Experiment Setting. We do additional experiments on CIFAR-100. CIFAR-100 is a dataset consisting of 100 classes and 20 superclasses. Each superclass

[2]This was shown to uniformly improve the mean-squared error averaged over the population, compared to an estimate using just the single local sample.

corresponds to a category of 5 classes (e.g. superclass flowers correspond to orchids, poppies, roses, sunflowers, tulips). To introduce heterogeneity we let each client sample data samples from 2 super classes (the classification task is still to classify among 100 classes). For classification on CIFAR-100 dataset we consider a 5-layer CNN with 2 convolutional layers of 64 filters and 5x5 filter size, following that we have 2 fully connected layers with activation sizes of 384,192 and finally an output layer of dimension 100. We set number of local epochs to be 2, batch size to be 25 per client; number of clients is 50, client participation $\frac{K}{n} = 0.2$, and number of epochs 200 (100 communication rounds). In this new dataset the classification task is more complex given the increased number of labels.

Hyperparameters. We implemented Per-FedAvg and pFedMe based on the code from GitHub,[3]. Other implementations were not available online, so we implemented ourselves. For each of the methods we tuned learning rate in the set $\{0.3, 0.2, 0.15, 0.125, 0.1, 0.075, 0.05\}$ and have a decaying learning schedule such that learning rate is multiplied by 0.99 at each epoch. We use weight decay of $1e-4$. For MNIST and FEMNIST experiments for both personalized and global models we used a 5-layer CNN, the first two layers consist of convolutional layers of filter size 5×5 with 6 and 16 filters respectively. Then we have 3 fully connected layers of dimension 256×120, 120×84, 84×10 and lastly a softmax operation. For CIFAR-10 experiments we use a similar CNN, the only difference is the first fully connected layer is of dimension 400×120.

- `AdaPeD`[4]: We fine-tuned ψ in between $0.5 - 5$ with 0.5 increments and set it to 3.5. We set $\eta_3 = 5e-2$. We manually prevent ψ becoming smaller than 0.5 so that local loss does not become dominated by the KD loss. We use $\eta_2 = 0.1$ and $\eta_1 = 0.1$. [5] When taking the derivative with respect to ψ we observed sometimes multiplying the right term (consist of KD loss function) by some constant (5 in our experiments) gives better performance.

[3] https://github.com/CharlieDinh/pFedMe

[4] For federated experiments we have used PyTorch's Data Distributed Parallel package.

[5] We use https://github.com/tao-shen/FEMNIST_pytorch to import FEMNIST dataset.

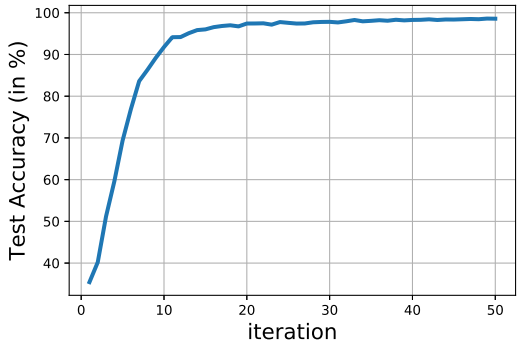


Figure B.1 `AdaPeD` Test Accuracy (in %) vs iteration on MNIST with 0.1 sampling ratio.

- Per-FedAvg [48] and pFedMe [42]: For Per-FedAvg, we used 0.075 as the learning rate and $\alpha = 0.001$. For pFedMe we used the same learning rate schedule for main learning rate, $K = 3$ for the number of local iterations; and we used $\lambda = 0.5$, $\eta = 0.2$.
- QuPeD [131]: We choose $\lambda_p = 0.25$, $\eta_1 = 0.1$ and $\eta_3 = 0.1$ as stated.
- Federated Mutual Learning [152]: Since authors do not discuss the hyperparameters in the paper, we used $\alpha = \beta = 0.25$, global model has the same learning schedule as the personalized models.

B.12.2 Additional Experiments

Convergence plots for `AdaPeD`. We put the experimental convergence plots (test accuracy vs no. of iteration) for `AdaPeD` in Figures B.1, B.2.

Personalized estimation: synthetic experiments in Bernoulli setting. For this setting, for $\mathbb{P}$ we consider three distributions that [163] considered: normal, uniform and '3-spike' which have equal weight at $1/4$, $1/2$, $3/4$. Additionally, we consider a Beta prior. We compute squared error of personalized estimators and local estimators $(\frac{Z_i}{n})$ w.r.t. the true p_i and report the average over all clients. We use $m = 10000$ clients and 14 local samples similar to [163]. Personalized estimator provides a decrease in MSE by

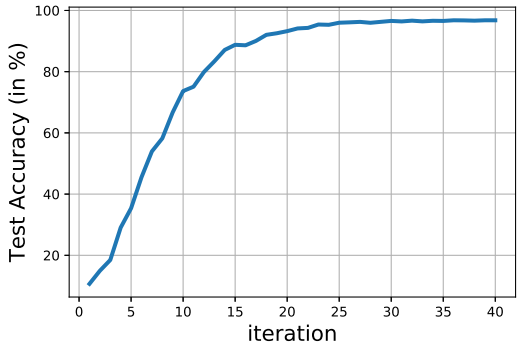


Figure B.2 `AdaPeD` Test Accuracy (in %) vs iteration on FEMNIST with 0.33 sampling ratio.

$37.1 \pm 3.9\%, 12.0 \pm 1.6\%, 24.3 \pm 2.8\%, 34.0 \pm 3.7\%$, respectively, for each aforementioned population distribution compared to their corresponding local estimators. Furthermore, as theoretically noted, less spread out prior distributions (low data heterogeneity) results in higher MSE gap between personalized and local estimators.

Linear regression. For this, we create a setting similar to [76]. We set $m = 10,000$, $n = 10$; and sample client true models according to a Gaussian centered at some randomly chosen μ with variance σ_θ^2. We randomly generate design matrices X_i and create Y_i at each client by adding a zero mean Gaussian noise with true variance σ_x^2 to $X_i\theta_i$. We set true values $\sigma_\theta^2 = 0.01, \sigma_x^2 = 0.05$ and we sample each component of μ from a Gaussian with 0 mean and 0.1 standard deviation and each component of X from a Gaussian with 0 mean and variance 0.05, both i.i.d. We measure the average MSE over all clients with and compare personalized and local methods. When $d = 50$, personalized regression has an MSE gain of $8.0 \pm 0.8\%, 14.8 \pm 1.2\%$, and when $d = 100$, $9.2 \pm 1.1\%, 12.3 \pm 2.0\%$ compared to local and FedAvg regression, respectively. Moreover, compared to personalized regression where $\mu, \sigma_\theta, \sigma_x$ are known, alternating algorithm only results in 1% and 4.7% increase in MSE respectively for $d = 50$ and $d = 100$.

Estimation Experiments. We provide more results for the estimation setting discussed in Figure 3.1. In Figure B.3 we have a setting with 1000 clients and 5 local samples and in Figure B.4 500 clients and 5 local samples per client. We observe with as the number

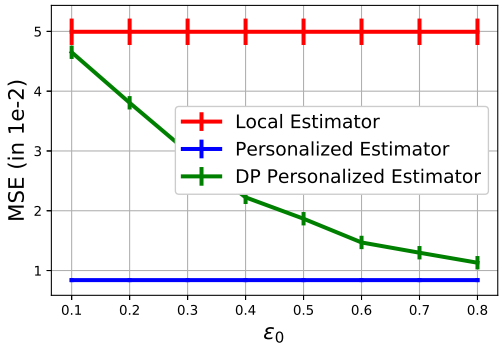


Figure B.3 Private Estimation with m=1000, n=5

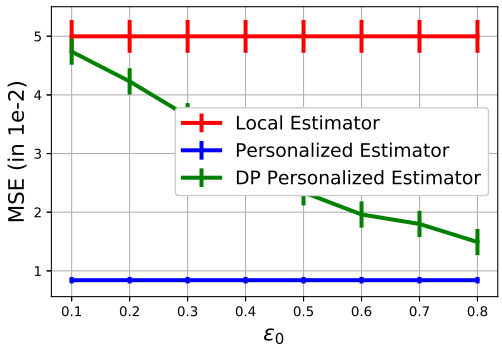


Figure B.4 Private Estimation with m=500, n=5

of clients increase DP-Personalized Estimator can converge to Personalized Estimator with less privacy budget. We also observe compared to Figure 3.1, less number of local samples increases the performance discrepancy between personalized and local estimator.

Additional Learning Experiments with Different Number of Clients. We do additional experiments with different number of clients. On FEMNIST we use the same model and same data sample per client as in Section7.5, number of clients is 30, total number of epochs is 30 and we fix the local iteration to be 40 per epoch, we do full client sampling to simulate a cross-silo environment. As seen in Table B.1, AdaPeD continues to outperform the competing methods following the trend in Section 7.5.

On CIFAR-10 we use the same model as in Section7.5, and divide the dataset to 30 clients where each client has access to data samples from 4 classes. Total number of epochs

Table B.1 Test accuracy (in %) for FEMNIST with $m = 30$ clients.

Method	FEMNIST
FedAvg	95.91 ± 0.78
FedAvg+fine tuning [78]	96.22 ± 0.57
`AdaPeD` (Ours)	$\mathbf{98.10} \pm 0.09$
pFedMe [42]	96.03 ± 0.50
Per-FedAvg [48]	96.71 ± 0.14
QuPeD (FP) [131]	97.72 ± 0.16
Federated ML [152]	96.80 ± 0.13

is 250 and we fix the local iteration to be 40 per epoch; we set $\frac{K}{n} = 0.2$ and number of local epochs to be 2. AdaPeD outperforms the competing methods in parallel to the experiments in Section7.5, as can be seen in Table B.2.

Table B.2 Test accuracy (in %) for CIFAR-10 with $m = 30$ clients.

Method	CIFAR-10
FedAvg	53.92 ± 0.94
FedAvg+fine tuning [78]	67.44 ± 1.11
`AdaPeD` (Ours)	$\mathbf{73.86} \pm 0.39$
pFedMe [42]	71.97 ± 0.09
Per-FedAvg [48]	64.09 ± 0.46
QuPeD (FP) [131]	73.21 ± 0.44
Federated ML [152]	72.53 ± 0.36

Additional Experiment Implementation Details.

We use the same strategy as in Appendix B.12.1 to tune the main learning rates. We use 1e-4 weight decay.

- `AdaPeD`: We fine-tuned ψ in between $0.5 - 5$ with 0.5 increments and set it to 4 for CIFAR-10/100 and to 3 for FEMNIST. We manually prevent ψ becoming smaller than 1 so that local loss does not become dominated by the KD loss. We use $\eta_2 = 0.075$ and $\eta_1 = 0.075$ for CIFAR-10 and CIFAR-100 and $\eta_2 = 0.1$ and $\eta_1 = 0.1$ for FEMNIST.
- Per-FedAvg [48] and pFedMe [42]: For Per-FedAvg, we used 0.1 as the learning rate and $\alpha = 0.0001$. For pFedMe we used the same learning rate schedule for main learning rate, $L = 3$ for the number of local approximation iterations; and we used $\lambda = 0.1$, $\eta = 0.1$.
- QuPeD [131]: We set $\lambda_p = 0.25$, $\eta_1 = 0.1$ for local learning rate and $\eta_2 = 0.1$ for global learning rate.
- Federated Mutual Learning [152]: Since authors do not discuss the hyperparameters in the paper, we used $\alpha = \beta = 0.25$.

Algorithm 11 Linear Regression GD

Input: Number of iterations T, local datasets (Y_i, X_i) for $i \in [m]$, learning rate η.

Initialize $\boldsymbol{\theta}_i^0$ for $i \in [m]$, $\boldsymbol{\mu}^0$, $\sigma_x^{2,0}$, $\sigma_\theta^{2,0}$.
for $t = 1$ **to** T **do**
  On Clients:
  for $i = 1$ **to** m: **do**
    Receive and set $\boldsymbol{\mu}_i^t = \boldsymbol{\mu}^t, \sigma_{\theta,i}^{2,t} = \sigma_\theta^{2,t}, \sigma_{x,i}^{2,t} = \sigma_x^{2,t}$
    Update the personalized model: $\boldsymbol{\theta}_i^t \leftarrow \boldsymbol{\theta}_i^{t-1} + \eta \left(\sum_{j=1}^n \frac{X_{ij}(Y_{ij} - X_{ij}\boldsymbol{\theta}_i^{t-1})}{\sigma_{x,i}^{2,t-1}} + \frac{\boldsymbol{\mu}_i^{t-1} - \boldsymbol{\theta}_i^{t-1}}{\sigma_{\theta,i}^{2,t-1}} \right)$
    Update local version of mean: $\boldsymbol{\mu}_i^t \leftarrow \boldsymbol{\mu}_i^{t-1} - \eta \left(\frac{\boldsymbol{\mu}_i^{t-1} - \boldsymbol{\theta}_i^{t-1}}{\sigma_{\theta,i}^{2,t-1}} \right)$
    Update local variance: $\sigma_{x,i}^{2,t} \leftarrow \sigma_{x,i}^{2,t-1} - \eta \left(\frac{n}{2\sigma_{x,i}^{2,t-1}} - \sum_{j=1}^n \frac{(Y_{ij} - X_{ij}\boldsymbol{\theta}_i^{t-1})^2}{2(\sigma_{x,i}^{2,t-1})^2} \right)$
    Update global variance: $\sigma_{\theta,i}^{2,t} \leftarrow \sigma_{\theta,_i}^{2,t-1} - \eta \left(\frac{d}{2\sigma_{\theta,i}^{2,t-1}} - \frac{\|\boldsymbol{\mu}_i^{t-1} - \boldsymbol{\theta}_i^{t-1}\|^2}{2(\sigma_{\theta,i}^{2,t-1})^2} \right)$
  end for
  At the Server:
  Aggregate mean: $\boldsymbol{\mu}^t = \frac{1}{m} \sum_{i=1}^m \boldsymbol{\mu}_i^t$
  Aggregate global variance: $\sigma_\theta^{2,t} = \frac{1}{m} \sum_{i=1}^m \sigma_{\theta,i}^{2,t}$
  Aggregate local variance: $\sigma_x^{2,t} = \frac{1}{m} \sum_{i=1}^m \sigma_{x,i}^{2,t}$
  Broadcast $\boldsymbol{\mu}^t, \sigma_\theta^{2,t}, \sigma_x^{2,t}$
end for

Output: Personalized models $\boldsymbol{\theta}_1^T, \ldots, \boldsymbol{\theta}_m^T$.

Algorithm 13 Alternating Minimization for Personalized Learning

Input: Number of iterations T, local datasets (X_i, Y_i) for $i \in [m]$.

Initialize $\boldsymbol{\theta}_i^0 = (X_i^T X_i)^{-1} X_i^T Y_i$ for $i \in [m]$ (if $X_i^T X_i$ is not full-rank, take the pseudo-inverse).
for $t = 1$ **to** T **do**
  On Clients:
  for $i = 1$ **to** m: **do**
    Receive $\mathbb{P}^{(t)}, \boldsymbol{\mu}_1^{(t)}, \ldots, \boldsymbol{\mu}_k^{(t)}$ from the server
    Update the personalized parameters and the coefficients:

$$\boldsymbol{\theta}_i^t \leftarrow \sum_{l=1}^{k} \alpha_l^{(i)} \boldsymbol{\mu}_l^{(t)} \qquad \text{and} \qquad \alpha_l^{(i)} = \frac{p_l^{(t)} \exp\left(-\frac{\|X_i \boldsymbol{\mu}_l^{(t)} - Y_i\|^2}{2\sigma_x^2}\right)}{\sum_{s=1}^{k} p_s^{(t)} \exp\left(-\frac{\|X_i \boldsymbol{\mu}_s^{(t)} - Y_i\|^2}{2\sigma_x^2}\right)}$$

    Send $\boldsymbol{\theta}_i^{(t)}$ to the server
  end for
  At the Server:
  Receive $\boldsymbol{\theta}_1^{(t)}, \ldots, \boldsymbol{\theta}_m^{(t)}$ from the clients
  Update the global parameters: $\mathbb{P}^{(t)}, \boldsymbol{\mu}_1^{(t)}, \ldots, \boldsymbol{\mu}_k^{(t)} \leftarrow \mathsf{Cluster}\left(\boldsymbol{\theta}_1^{(t)}, \ldots, \boldsymbol{\theta}_m^{(t)}, k\right)$
  Broadcast $\mathbb{P}^{(t)}, \boldsymbol{\mu}_1^{(t)}, \ldots, \boldsymbol{\mu}_k^{(t)}$ to all clients
end for

Output: Personalized models $\boldsymbol{\theta}_1^T, \ldots, \boldsymbol{\theta}_m^T$.

APPENDIX C

Appendix of Chapter 4

C.1 Related Works, Limitations, and Broader Impact

As mentioned earlier, there have been several recent works on personalized *supervised* learning, and we give a more detailed accounting of it in Appendix C.1. There has been much less attention given to personalized federated unsupervised learning. The closest work to ours on personalized dimensionality reduction is [128, 153] which studies personalized PCA algorithms. [153] has a restrictive assumption that principal components for global and local models are non-overlapping. [128] develops a criterion for personalized PCA; however, the authors assume heterogeneity of the setting is known; in contrast, `ADEPT-PCA` learns and adapts the solution accordingly. There is some literature on specific tasks such as training recommender systems [93, 134], grouping clients based on latent representations [176], generating data to improve performance of personalized supervised learning [23, 133]. However, our approach to developing personalized dimensionality reduction for heterogeneous data is distinct from theirs. We are not aware of any FL work for personalized diffusion generative models. There is work using pre-trained diffusion models and then fine-tuning them [112, 124, 148, 184]. However, these do not fit into the federated learning paradigm and require data collection from clients to obtain the pre-trained model. One way to view our approach is to *simultaneously* build such "global" models (akin to pre-trained models) and individual (personalized) models, while not sharing data.

Related Works: Some of the related works beyond those given in section 5.1 are as follows:

Personalized Federated Learning (FL) has seen recent advances with diverse approaches for learning personalized models. These approaches encompass meta-learning-based methods [1, 48, 87], regularization techniques [40, 65, 114], clustered FL [56, 114, 115, 185], knowledge distillation strategies [101, 131], multi-task learning [42, 157, 168, 183], and the utilization of common representations [34, 44, 164]. Additionally, recently there have been works on using a hierarchical Bayesian view to derive novel personalized supervised FL algorithms [28, 90, 126]. Adaptation is also considered in [28, 126]. In [28], variance estimation is performed for supervised learning to estimate the heterogeneity within the local models and the estimated variance is used to form an initialization for each local model to be trained on. However, if we apply variance estimation to our criterion, we observe an early stopping issue during the training. In contrast, ours is based on a standard gradient descent. Moreover, they do not examine convergence which we do in our unsupervised algorithms. In [126], the authors consider an adaptation for supervised learning based on KL-divergence criterion and do not have hyper prior. Moreover, a convergence analysis is not explored in their algorithm. In that case, the σ is inclined to smaller values or even vanishes during the training, and our `ADEPT` criterion resolves the issue by introducing the hyper prior. [167] investigated local features of globally trained diffusion models through FedAvg, but they do not consider personalized generation. We did not find any other works on personalized federated generation models.

Limitations: As we mentioned in the conclusion, the introduction of information constraints such as privacy is an important future work. Without formal privacy guarantees, an adversarial client can utilize a global model to generate data similar to other clients' data. Being the first such result, we have a simplified setting for the theoretical analysis of `ADEPT-DGM`; our analysis can be extended to more complex settings.

Broader Impact: Discovering the structure of local data is important in many applications. As we argued, collaboration becomes critical when there is insufficient local data for doing such analysis. Such applications occur in many domains including cyber-physical systems, autonomous systems, sensing, medical devices, etc., where one wants to discover the structure

of heterogeneous local data. These applications are becoming widespread and therefore solutions to these problems could have a broader impact in these applications, especially where one wants to do this without explicit data sharing. Our methods for personalization can benefit users with limited data and resources to discover their particular structure of local data. Not explicitly sharing data is a first step towards privacy; however, as mentioned earlier, building explicit privacy mechanisms is needed to give privacy guarantees. This is an important aspect that has been studied in federated learning, but has received less attention in personalized learning and is an important topic of future exploration. By incorporating this as well, many more applications with societal benefits could be realized.

C.2 Preliminaries

For the notations used in this paper:

- We use bold lowercase letters (such as $\boldsymbol{u}$, $\boldsymbol{v}$) to denote vectors and we use bold uppercase letters (such as $\boldsymbol{U}$, $\boldsymbol{V}$) to denote matrices.
- Given a composite function $f(\boldsymbol{u}, \boldsymbol{v})$, we denote $\nabla f(\boldsymbol{u}, \boldsymbol{v})$ or $\nabla_{(\boldsymbol{u},\boldsymbol{v})} f(\boldsymbol{u}, \boldsymbol{v})$ as the gradient; $\nabla_{\boldsymbol{u}} f(\boldsymbol{u}, \boldsymbol{v})$ and $\nabla_{\boldsymbol{v}} f(\boldsymbol{u}, \boldsymbol{v})$ as the partial gradients with respect to $\boldsymbol{u}$ and $\boldsymbol{v}$.
- For a vector $\boldsymbol{u}$, $\|\boldsymbol{u}\|$ denotes the ℓ_2-norm $\|\boldsymbol{u}\|_2$. For a matrix $\boldsymbol{U}$, $\|\boldsymbol{U}\|_2$ and $\|\boldsymbol{U}\|_{op}$ both denote the ℓ_2-norm (operator norm) of the matrix and $\|\boldsymbol{U}\|, \|\boldsymbol{U}\|_F$ denotes the Frobenius norm of the matrix.
- We use $\{\boldsymbol{U}_i\}_{i=1}^m$ or $\{\boldsymbol{U}_i\}$ (when the context is clear) to denote the collection $\{U_1, \ldots, U_m\}$. When there are multiple indices, we use $\{\boldsymbol{U}_{i,t}\}_i := \{\boldsymbol{U}_{1,t}, \ldots, \boldsymbol{U}_{m,t}\}$ to denote the collection over index i.

C.3 Proofs for Adaptive PCA: `ADEPT-PCA`

C.3.1 Proof Summary

First, we define the polar retraction used in `ADEPT-PCA` and show the non-expansiveness and Lipschitz properties under the polarization retraction (lemma 11, lemma 12). Next, we establish a lower bound on σ_t given a proper initialization σ_0 (lemma 1). With the lower bound on σ_t, we are able to derive the Lipschitz smoothness constants and gradients bounds with respect to $\boldsymbol{U}$, $\boldsymbol{U}$ (lemma 14 and lemma 15). Also, we derive the Lipschitz continuous constants of the derivative $\frac{\partial}{\partial\sigma} f_i^{\text{pca}}$ with respect to $\boldsymbol{U}$ and $\boldsymbol{U}$ (lemma 16). With the Lipschitz constants, we derive the sufficient decrease of the loss function for $\boldsymbol{U}$, $\boldsymbol{U}$, and σ (lemma 17). Finally, we combine them and get to our theorem 8.

C.3.2 Lemmas

For any point $\boldsymbol{U} \in St(d,r)$, a retraction at a point $\boldsymbol{U} \in St(d,r)$ is a map $\mathcal{R}_{\boldsymbol{U}} : \mathcal{T}_{\boldsymbol{U}} \to St(d,r)$ that induces local coordinates on the Stiefel manifold. In this work, we use polar retraction that is defined as $\mathcal{R}_{\boldsymbol{U}}(\boldsymbol{U}) = (\boldsymbol{U}+\boldsymbol{U})(\boldsymbol{I}+\boldsymbol{U}^\top\boldsymbol{U})^{-\frac{1}{2}}$. The polar retraction is a second-order retraction that approximates the exponential mapping up to second-order terms. Consequently, it possesses the following non-expansiveness property and we state the Lemmas and properties that we use throughout the proof before showing the proofs.

Lemma 11 (Non-expansiveness of polar retraction [30])**.** *Let $\boldsymbol{U} \in St(d,r)$, for any point $\boldsymbol{U} \in \mathcal{T}_{\boldsymbol{U}}$ with bounded norm, $\|\boldsymbol{U}\|_F \leq M$, there exists $C \in \mathbb{R}$ such that*

$$\|\mathcal{R}_{\boldsymbol{U}}(\boldsymbol{U}) - (\boldsymbol{U}+\boldsymbol{U})\|_F \leq C\|\boldsymbol{U}\|_F^2. \tag{C.1}$$

Lemma 12 (Lipschitz type inequality [30])**.** *Let $\boldsymbol{U}, \boldsymbol{U} \in St(d,r)$. If a function ψ is L-Lipschitz smooth in $\mathbb{R}^{d\times r}$, the following inequality holds:*

$$|\psi(\boldsymbol{U}) - (\psi(\boldsymbol{U}) + \langle P_{\mathcal{T}_{\boldsymbol{U}}}(\nabla\psi(\boldsymbol{U})), \boldsymbol{U}-\boldsymbol{U}\rangle)| \leq \frac{L_g}{2}\|\boldsymbol{U}-\boldsymbol{U}\|_F^2$$

where $L_g = L + G$ with $G := \max_{\boldsymbol{U}\in St(d,r)} \|\nabla\psi(\boldsymbol{U})\|_2$.

Assumption 3. *For each client i, the operator and Frobenius norms of $\mathbf{S}_i$ are bounded by*

$$\|\mathbf{S}_i\|_F \leq G_{i,F} \quad \textit{and} \quad \|\mathbf{S}_i\|_{op} \leq G_{i,op},$$

and we define $G_{max,F} := \max_{i\in[m]} G_{i,F}$ and $G_{max,op} := \max_{i\in[m]} G_{i,op}$.

For the remaining part of the paper, we fix some $\omega \in (0,1)$ and initialize σ_0 corresponding to the ω in order to obtain the lower bound in Lemma 1.

Lemma 13 (Lipschitz smoothness and bounded gradients with respect to σ)**.** *For all $i \in [m]$ and $\boldsymbol{U}_i, \boldsymbol{U} \in St(d,r)$, within the domain $\sigma \in [\omega\sqrt{\frac{2\xi}{d}}, \infty]$, the function $f_i^{\text{pca}}(\boldsymbol{U}_i, \boldsymbol{U}, \sigma)$ is L_σ-Lipschitz smooth with respect to σ with constants*

$$L_\sigma := \frac{d^2}{2\xi\omega^2} + \frac{3d^2}{2\xi\omega^4} + \frac{3d^2}{\xi^2\omega^4}.$$

Lemma 14 (Lipschitz smoothness and bounded gradients with respect to $\boldsymbol{U}_i$)**.** *The function $f_i^{\text{pca}}(\boldsymbol{U}_i, \boldsymbol{U}, \sigma)$ is L_U-Lipschitz smooth with respect to $\boldsymbol{U}_i$ and $\|\nabla f_i^{\text{pca}}(\boldsymbol{U}_i, \boldsymbol{U}, \sigma)\|_2 \leq G_U$ for all $i \in [m]$ with constants*

$$L_U := \frac{n}{2}\left(\frac{1}{\sigma_\epsilon^2} + \frac{G_{max,op}}{\sigma_\epsilon^4} + \left(1 + \frac{2G_{max,op}}{\sigma_\epsilon^2}\right)\frac{2}{\sigma_\epsilon^4}\right) + \frac{d}{\xi\omega^2},$$
$$G_U := \frac{n}{2}\left(\frac{G_{max,op}}{\sigma_\epsilon^4} + \frac{1}{\sigma_\epsilon^2}\right) + \frac{d}{\xi\omega^2}.$$

Lemma 15 (Lipschitz smoothness and bounded gradients with respect to $\boldsymbol{U}$)**.** *The function $f^{\text{pca}}(\{\boldsymbol{U}_i\}_i, \boldsymbol{U}, \sigma)$ is L_V-Lipschitz smooth with respect to $\boldsymbol{U}$ and $\|\nabla f^{\text{pca}}(\{\boldsymbol{U}_i\}_i, \boldsymbol{U}, \sigma)\|_2 \leq G_V$ with constants*

$$L_V := \frac{12d}{\xi\omega^2},$$
$$G_V := \frac{3d}{\xi\omega^2}.$$

Lemma 16 (Lipschitz continuity of $\frac{\partial}{\partial\sigma} f_i^{\text{pca}}(\boldsymbol{U}, \boldsymbol{U}, \sigma)$ with respect to $\boldsymbol{U}, \boldsymbol{U}$)**.** *The function $\frac{\partial}{\partial\sigma} f_i^{\text{pca}}(\boldsymbol{U}, \boldsymbol{U}, \sigma)$ is $L_U^{(\sigma)}$-Lipschitz continuous with respect to $\boldsymbol{U}$ and $L_V^{(\sigma)}$-Lipschitz continuous with respect to $\boldsymbol{U}$ with*

$$L_U^{(\sigma)} = \frac{\sqrt{2d^3}}{\omega^3\sqrt{\xi^3}},$$

$$L_V^{(\sigma)} = \frac{2\sqrt{d^3}}{\omega^3\sqrt{2\xi^3}}.$$

Before showing the convergence results, we define the following terms. Let

$$\begin{aligned}
C_{\eta_1} &= C_1 G_1 + \frac{(L_U + G_U)(C_1^2 G_1^2 + 1)}{2}, \\
C\eta_2 &= C_2 G_2 + \frac{(L_V + G_V)(C_2^2 G_2^2 + 1)}{2}, \\
G_1 &= 2G_U\sqrt{d}, \\
G_2 &= 2G_V\sqrt{d}.
\end{aligned}$$

with some constants C_1, C_2 given by Lemma 11 and G_U, G_V given in Lemma 14, 15.

Lemma 17 (Sufficient Decrease)**.** *At any iteration t, we have*

$$\begin{aligned}
& f^{\text{pca}}(\{\boldsymbol{U}_{i,t}\}_i, \boldsymbol{U}_t, \sigma_t) - f^{\text{pca}}(\{\boldsymbol{U}_{i,t-1}\}_i, \boldsymbol{U}_{t-1}, \sigma_{t-1}) \\
\leq\ & \left(-\eta_1 + C_{\eta_1}\eta_1^2 + \eta_3 (L_U^{(\sigma)})^2\right) \frac{1}{m}\sum_{i=1}^m \| P_{\mathcal{T}_{\boldsymbol{U}_{i,t-1}}} \left(\nabla_{\boldsymbol{U}_{i,t-1}} f_i^{\text{pca}}(\boldsymbol{U}_{i,t-1}, \boldsymbol{U}_{t-1}, , \sigma_{t-1})\right)\|_F^2 \\
& + \left(-\eta_2 + C\eta_2\eta_2^2 + \eta_3 (L_V^{(\sigma)})^2\right) \| P_{\mathcal{T}_{\boldsymbol{U}_{t-1}}} \left(\nabla_{\boldsymbol{U}_{t-1}} f^{\text{pca}}(\{\boldsymbol{U}_{i,t}\}_i, \boldsymbol{U}_{t-1}, \sigma_{t-1})\right)\|_F^2 \\
& + \left(\frac{-\eta_3 + \eta_3^2 L_\sigma}{2}\right) \left[\frac{\partial}{\partial \sigma_{t-1}} f^{\text{pca}}(\{\boldsymbol{U}_{i,t-1}\}_i, \boldsymbol{U}_{t-1}, \sigma_{t-1})\right]^2.
\end{aligned}$$

Theorem 20. *By choosing $\eta_1 = \min\{\frac{1}{3C_{\eta_1}}, 1\}$, $\eta_2 = \min\{\frac{1}{3C\eta_2}, 1\}$, and $\eta_3 = \min\left\{\frac{\eta_1}{3(L_U^{(\sigma)})^2}, \frac{\eta_2}{3(L_V^{(\sigma)})^2}, \frac{1}{bL_\sigma}\right\}$, we have*

$$\sum_{t=1}^T \left(\left(\frac{1}{m}\sum_{i=1}^m \|\mathbf{g}_{i,t}^{\boldsymbol{U}}\|_F^2\right) + \|\mathbf{g}_t^{\boldsymbol{U}}\|_F^2 + (\mathbf{g}_t^\sigma)^2\right) \leq \frac{3\Delta_T}{\min\{\eta_1, \eta_2, \eta_3\}}$$

where

$$\begin{aligned}
\mathbf{g}_t^{\boldsymbol{U}} &= P_{\mathcal{T}_{\boldsymbol{U}_{t-1}}} \left(\nabla_{\boldsymbol{U}_{t-1}} f^{\text{pca}}(\{\boldsymbol{U}_{i,t}\}_i, \boldsymbol{U}_{t-1}, \sigma_{t-1})\right), \\
\mathbf{g}_{i,t}^{\boldsymbol{U}} &= P_{\mathcal{T}_{\boldsymbol{U}_{i,t-1}}} \left(\nabla_{\boldsymbol{U}_{i,t-1}} f_i^{\text{pca}}(\boldsymbol{U}_{i,t-1}, \boldsymbol{U}_{t-1}, , \sigma_{t-1})\right), \\
\mathbf{g}_t^\sigma &= \frac{\partial}{\partial \sigma_{t-1}} f^{\text{pca}}(\{\boldsymbol{U}_{i,t-1}\}_i, \boldsymbol{U}_{t-1}, \sigma_{t-1}), \\
\Delta_T &= f^{\text{pca}}(\{\boldsymbol{U}_{i,0}\}_i, \boldsymbol{U}_0, \sigma_0) - f^{\text{pca}}(\{\boldsymbol{U}_{i,T}\}_i, \boldsymbol{U}_T, \sigma_T).
\end{aligned}$$

C.3.3 Proofs

Fact 7. *The gradients of the local loss function with respect to the local and global PC's and σ are given as*

$$\begin{aligned}
\nabla_{\boldsymbol{U}_i} f_i^{\mathrm{pca}}(\boldsymbol{U}_i, \boldsymbol{U}, \sigma) &= -\frac{n}{2}(\boldsymbol{W}_i^{-1}\mathbf{S}_i\boldsymbol{W}_i^{-1}\boldsymbol{U}_i - \boldsymbol{W}_i^{-1}\boldsymbol{U}_i) + \frac{\mathcal{P}_{\mathcal{T}_{\boldsymbol{U}}}(\boldsymbol{U}_i)}{\sigma^2}, \\
\nabla_{\boldsymbol{U}} f_i^{\mathrm{pca}}(\boldsymbol{U}_i, \boldsymbol{U}) &= -\frac{\mathcal{P}_{\mathcal{T}_{\boldsymbol{U}}}(\boldsymbol{U}_i)(\boldsymbol{U}_i^\top \boldsymbol{U} + \boldsymbol{U}^\top \boldsymbol{U}_i)}{2\sigma^2}, \\
\frac{\partial}{\partial\sigma} f_i^{\mathrm{pca}}(\boldsymbol{U}_i, \boldsymbol{U}, \sigma) &= \frac{d}{\sigma} - \frac{2\xi + d^2(\boldsymbol{U}, \boldsymbol{U}_i)}{\sigma^3}, \\
\nabla_{\boldsymbol{U}_i}\left(\frac{\partial}{\partial\sigma} f_i^{\mathrm{pca}}(\boldsymbol{U}_i, \boldsymbol{U}, \sigma)\right) &= -\frac{2\mathcal{P}_{\mathcal{T}_{\boldsymbol{U}}}(\boldsymbol{U}_i)}{\sigma^3}, \\
\nabla_{\boldsymbol{U}}\left(\frac{\partial}{\partial\sigma} f_i^{\mathrm{pca}}(\boldsymbol{U}_i, \boldsymbol{U}, \sigma)\right) &= \frac{\mathcal{P}_{\mathcal{T}_{\boldsymbol{U}}}(\boldsymbol{U}_i)(\boldsymbol{U}_i^\top \boldsymbol{U} + \boldsymbol{U}^\top \boldsymbol{U}_i)}{\sigma^3}.
\end{aligned}$$

Fact 8. *For two matrices $\boldsymbol{A} \in \mathbb{R}^{a\times b}$ and $\boldsymbol{B} \in \mathbb{R}^{b\times c}$, we have*

$$\|\boldsymbol{AB}\|_F \le \|\boldsymbol{A}\|_{op}\|\boldsymbol{B}\|_F \text{ and } \|\boldsymbol{AB}\|_F \le \|\boldsymbol{A}\|_F\|\boldsymbol{B}\|_{op}.$$

Fact 9. *For matrix to matrix functions, $\{g_i\}_{i=1}^k$, with bounded output operator norms, $\max_{\mathbf{X}} \|g_i(\mathbf{X})\|_{op} \le M_i$, we have*

$$\|\prod_{i=1}^k g_i(\mathbf{X}) - \prod_{i=1}^k g_i(\mathbf{Y})\|_F \le \prod_{j=1}^k M_j \left(\sum_{i=1}^k \|g_i(\mathbf{X}) - g_i(\mathbf{Y})\|_F\right)$$

C.3.3.1 Proof of Lemma 1

Proof. We use mathematical induction to prove the lemma. For the base case, it is given that $\sigma_0 \ge \omega\sqrt{\frac{2\xi}{d}}$. Assume that for all $\tau \in \{0, 1, \ldots, t\}$,

$$\sigma_\tau \ge \omega\sqrt{\frac{2\xi}{d}}.$$

Then, we consider the following two cases. First, if $\sigma_t \in \left[\omega\sqrt{\frac{2\xi}{d}}, \sqrt{\frac{2\xi}{d}}\right]$, we have

$$\sigma_t \le \sqrt{\frac{2\xi}{d}} \quad \Rightarrow \quad d \le \frac{2\xi}{\sigma_t^2} \quad \Rightarrow \quad \frac{d}{\sigma_t} - \frac{2\xi}{\sigma_t^3} \le 0$$

$$\Rightarrow \quad \forall i \in [m]: \quad \frac{\partial}{\partial \sigma_t} f_i^{\text{pca}}(\boldsymbol{U}_{i,t}, \boldsymbol{U}_t, \sigma_t) = \frac{d}{\sigma_t} - \frac{2\xi + d^2(\boldsymbol{U}_t, \boldsymbol{U}_{i,t})}{\sigma_t^3} \leq \frac{d}{\sigma_t} - \frac{2\xi}{\sigma_t^3} \leq 0$$

$$\Rightarrow \quad \forall i \in [m]: \quad \sigma_{i,t+1} = \sigma_t - \eta_3 \frac{\partial}{\partial \sigma_t} f_i^{\text{pca}}(\boldsymbol{U}_{i,t}, \boldsymbol{U}_t, \sigma_t) \geq \sigma_t \geq \omega \sqrt{\frac{2\xi}{d}}$$

$$\Rightarrow \quad \sigma_{t+1} = \frac{1}{m} \sum_{i=1}^{m} \sigma_{i,t+1} \geq \omega \sqrt{\frac{2\xi}{d}}.$$

Otherwise, if we have $\sigma_t > \sqrt{\frac{2\xi}{d}}$, we have

$$\begin{aligned} \sigma_{i,t+1} &= \sigma_t - \eta_3 \frac{\partial}{\partial \sigma_t} f_i^{\text{pca}}(\boldsymbol{U}_{i,t}, \boldsymbol{U}_t, \sigma_t) = \sigma_t - \eta_3 \left(\frac{d}{\sigma_t} - \frac{2\xi + d^2(\boldsymbol{U}_t, \boldsymbol{U}_{i,t})}{\sigma_t^3} \right) \\ &\geq \sigma_t - \eta_3 \frac{d}{\sigma_t} \geq \sqrt{\frac{2\xi}{d}} - (1-\omega) \frac{2\xi}{d^2} \cdot \frac{d}{\sqrt{2\xi/d}} = \omega \sqrt{\frac{2\xi}{d}} \end{aligned}$$

and thus

$$\sigma_{t+1} = \frac{1}{m} \sum_{i=1}^{m} \sigma_{i,t+1} \geq \omega \sqrt{\frac{2\xi}{d}}.$$

Thus, by mathematical induction, we have

$$\forall t \in \mathbb{N} \ \ \forall i \in [m]: \quad \sigma_t \geq \omega \sqrt{\frac{2\xi}{d}} \quad \text{and} \quad \sigma_{i,t} \geq \omega \sqrt{\frac{2\xi}{d}}.$$

□

C.3.3.2 Proof of Lemma 13

Proof. For the Lipschitz smoothness, we have

$$\begin{aligned} \left| \frac{\partial^2}{\partial \sigma^2} f_i^{\text{pca}}(\boldsymbol{U}_i, \boldsymbol{U}, \sigma) \right| &= \left| \frac{6\xi + 3d^2(\boldsymbol{U}, \boldsymbol{U}_i)}{\sigma^4} - \frac{d}{\sigma^2} \right| \\ &\leq \left| \frac{6\xi + 3d^2(\boldsymbol{U}, \boldsymbol{U}_i)}{\sigma^4} \right| + \left| \frac{d}{\sigma^2} \right| \\ &\leq \frac{6\xi + 12}{4\xi^2\omega^4/d^2} + \frac{d}{2\xi\omega^2/d} \\ &= \frac{3d^2}{2\xi\omega^4} + \frac{3d^2}{\xi^2\omega^4} + \frac{d^2}{2\xi\omega^2} \\ &= L_\sigma \end{aligned}$$

for any $\boldsymbol{U}$, $\boldsymbol{U}_i$, and $\sigma \geq \omega \sqrt{\frac{2\xi}{d}}$. □

C.3.3.3 Proof of Lemma 14

Proof. For the bound on the gradient,

$$\begin{aligned}
&\left\| -\frac{n}{2}(\boldsymbol{W}_i^{-1}S_i\boldsymbol{W}_i^{-1}\boldsymbol{U}_i - \boldsymbol{W}_i^{-1}\boldsymbol{U}_i) + \frac{\mathcal{P}_{\mathcal{T}_U}(\boldsymbol{U}_i)}{\sigma^2} \right\|_{op} \\
\leq & \| -\frac{n}{2}(\boldsymbol{W}_i^{-1}S_i\boldsymbol{W}_i^{-1}\boldsymbol{U}_i - \boldsymbol{W}_i^{-1}\boldsymbol{U}_i)\|_{op} + \frac{2}{\sigma^2} \\
\leq & \frac{n}{2}(\|\boldsymbol{W}_i^{-1}S_i\boldsymbol{W}_i^{-1}\boldsymbol{U}_i\|_{op} + \|\boldsymbol{W}_i^{-1}\boldsymbol{U}_i\|_{op}) + \frac{2}{\sigma^2} \\
\leq & \frac{n}{2}\left(\frac{G_{max,op}}{\sigma_\epsilon^4} + \frac{1}{\sigma_\epsilon^2}\right) + \frac{d}{\xi\omega^2},
\end{aligned}$$

where in the last inequality we use $\|\boldsymbol{W}_i^{-1}\|_{op} \leq \frac{1}{\sigma_\epsilon^2}$. Therefore, we find that the norm of the gradient is bounded by $G_U := \frac{n}{2}(\frac{G_{max,op}}{\sigma_\epsilon^4} + \frac{1}{\sigma_\epsilon^2}) + \frac{d}{\xi\omega^2}$. For the Lipschitz continuity of the gradient, we omit the client index i and use $\boldsymbol{U}_1$ and $\boldsymbol{U}_2$ to denote two arbitrary points on $St(d,r)$ for simplicity. For any client i, we focus on the first term of the gradient,

$$\begin{aligned}
&\|\boldsymbol{W}_1^{-1}S_i\boldsymbol{W}_1^{-1}\boldsymbol{U}_1 - \boldsymbol{W}_1^{-1}\boldsymbol{U}_1 - \boldsymbol{W}_2^{-1}S_i\boldsymbol{W}_2^{-1}\boldsymbol{U}_2 + \boldsymbol{W}_2^{-1}\boldsymbol{U}_2\|_F \\
&\leq \left(\frac{1}{\sigma_\epsilon^2} + \frac{G_{max,op}}{\sigma_\epsilon^4} + \left(1 + \frac{2G_{max,op}}{\sigma_\epsilon^2}\right)\frac{2}{\sigma_\epsilon^4}\right)\|\boldsymbol{U}_2 - \boldsymbol{U}_1\|_F, \qquad \text{(C.2)}
\end{aligned}$$

where we defer the algebra to the Appendix. For the second part of the gradient we have

$$\begin{aligned}
&\frac{1}{\sigma^2}\|\mathcal{P}_{\mathcal{T}_U}(\boldsymbol{U}_1) - \mathcal{P}_{\mathcal{T}_U}(\boldsymbol{U}_2)\|_F \\
&= \frac{1}{\sigma^2}\|\boldsymbol{U}_1 - \boldsymbol{U}_2 - \frac{1}{2}\boldsymbol{U}(\boldsymbol{U}^\top(\boldsymbol{U}_1 - \boldsymbol{U}_2) + (\boldsymbol{U}_1^\top - \boldsymbol{U}_2^\top)\boldsymbol{U})\|_F \\
&\leq \frac{2}{\sigma^2}\|\boldsymbol{U}_1 - \boldsymbol{U}_2\|_F, \\
&\leq \frac{d}{\xi\omega^2}\|\boldsymbol{U}_1 - \boldsymbol{U}_2\|_F,
\end{aligned}$$

where in the last inequality we use Fact 8. As a result, we find that the gradient is Lipschitz continuous with $L_U := \frac{n}{2}\left(\frac{1}{\sigma_\epsilon^2} + \frac{G_{max,op}}{\sigma_\epsilon^4} + \left(1 + \frac{2G_{max,op}}{\sigma_\epsilon^2}\right)\frac{2}{\sigma_\epsilon^4}\right) + \frac{d}{\xi\omega^2}$. □

C.3.3.4 Proof of Lemma 15

Proof. For the Lipschitz constant

$$\begin{aligned}
&\frac{2}{\sigma^2}\|\mathcal{P}_{\mathcal{T}_{\boldsymbol{V}_1}}(\boldsymbol{U}_i)\mathrm{sym}(\boldsymbol{U}_i^\top\boldsymbol{U}_1)-\mathcal{P}_{\mathcal{T}_{\boldsymbol{V}_2}}(\boldsymbol{U}_i)\mathrm{sym}(\boldsymbol{U}_i^\top\boldsymbol{U}_2)\|_F \\
&= \frac{2}{\sigma^2}\Big\|\boldsymbol{U}_i\boldsymbol{U}_i^\top(\boldsymbol{U}_1-\boldsymbol{U}_2)+\boldsymbol{U}_i(\boldsymbol{U}_1-\boldsymbol{U}_2)\boldsymbol{U}_i^\top \\
&\qquad\qquad -\frac{1}{2}(\boldsymbol{U}_1(\boldsymbol{U}_1^\top\boldsymbol{U}_i+\boldsymbol{U}_i^\top\boldsymbol{U}_1)-\boldsymbol{U}_2(\boldsymbol{U}_2^\top\boldsymbol{U}_i+\boldsymbol{U}_i^\top\boldsymbol{U}_2))\Big\|_F \\
&\le \frac{24}{\sigma^2}\|\boldsymbol{U}_1-\boldsymbol{U}_2\|_F \\
&\le \frac{12d}{\xi\omega^2},
\end{aligned}$$

where $\mathrm{sym}(\boldsymbol{U}_i^\top\boldsymbol{U}) = \boldsymbol{U}_i^\top\boldsymbol{U}+\boldsymbol{U}^\top\boldsymbol{U}_i$ and we used Fact 9, hence $L_V = \frac{12d}{\xi\omega^2}$. For the gradient bound, is it straightforward to see that

$$\|\nabla_{\boldsymbol{U}} f_i^{\mathrm{pca}}(\boldsymbol{U},\boldsymbol{U},\sigma)\|_2 \le \frac{4}{\sigma^2} \le \frac{2d}{\xi\omega^2}.$$

□

C.3.3.5 Proof of Lemma 16

Proof. Using Fact 7, we have

$$\|\nabla_{\boldsymbol{U}_i}\left(\frac{\partial}{\partial\sigma}f_i^{\mathrm{pca}}(\boldsymbol{U}_i,\boldsymbol{U},\sigma)\right)\|_2 \le \frac{4}{\sigma^3} \le \frac{\sqrt{2d^3}}{\omega^3\sqrt{\xi^3}}$$

and

$$\|\nabla_{\boldsymbol{U}}\left(\frac{\partial}{\partial\sigma}f_i^{\mathrm{pca}}(\boldsymbol{U}_i,\boldsymbol{U},\sigma)\right)\|_2 \le \frac{8}{\sigma^3} \le \frac{2\sqrt{2d^3}}{\omega^3\sqrt{\xi^3}}$$

□

C.3.3.6 Proof of Lemma 17

Proof. We have

$$f^{\mathrm{pca}}(\{\boldsymbol{U}_{i,t}\}_i,\boldsymbol{U}_t,\sigma_t) - f^{\mathrm{pca}}(\{\boldsymbol{U}_{i,t-1}\}_i,\boldsymbol{U}_{t-1},\sigma_{t-1})$$

$$
\begin{aligned}
&= [f^{\text{pca}}(\{\boldsymbol{U}_{i,t}\}_i, \boldsymbol{U}_t, \sigma_t) - f^{\text{pca}}(\{\boldsymbol{U}_{i,t}\}_i, \boldsymbol{U}_t, \sigma_{t-1})] \\
&+ [f^{\text{pca}}(\{\boldsymbol{U}_{i,t}\}_i, \boldsymbol{U}_t, \sigma_{t-1}) - f^{\text{pca}}(\{\boldsymbol{U}_{i,t}\}_i, \boldsymbol{U}_{t-1}, \sigma_{t-1})] \\
&+ [f^{\text{pca}}(\{\boldsymbol{U}_{i,t}\}_i, \boldsymbol{U}_{t-1}, \sigma_{t-1}) - f^{\text{pca}}(\{\boldsymbol{U}_{i,t-1}\}_i, \boldsymbol{U}_{t-1}, \sigma_{t-1})]
\end{aligned}
$$

With a similar proof as Lemma 5 in [128], we have

$$
\begin{aligned}
&f^{\text{pca}}(\{\boldsymbol{U}_{i,t}\}_i, \boldsymbol{U}_{t-1}, \sigma_{t-1}) - f^{\text{pca}}(\{\boldsymbol{U}_{i,t-1}\}_i, \boldsymbol{U}_{t-1}, \sigma_{t-1}) \\
&\qquad \leq \frac{(-\eta_1 + C_{\eta_1}\eta_1^2)}{m} \sum_{i=1}^{m} \| P_{\mathcal{T}_{\boldsymbol{U}_{i,t-1}}} (\nabla_{\boldsymbol{U}_{i,t-1}} f_i^{\text{pca}}(\boldsymbol{U}_{i,t-1}, \boldsymbol{U}_{t-1}, \sigma_{i,t})) \|_F^2, \\
&f^{\text{pca}}(\{\boldsymbol{U}_{i,t}\}_i, \boldsymbol{U}_{t}, \sigma_{t-1}) - f^{\text{pca}}(\{\boldsymbol{U}_{i,t}\}_i, \boldsymbol{U}_{t-1}, \sigma_{t-1}) \\
&\qquad \leq (-\eta_2 + C\eta_2\eta_2^2) \| P_{\mathcal{T}_{\boldsymbol{U}_{t-1}}} (\nabla_{\boldsymbol{U}_{t-1}} f^{\text{pca}}(\{\boldsymbol{U}_{i,t}\}_i, \boldsymbol{U}_{t-1}, \sigma_{t-1})) \|_F^2
\end{aligned}
$$

where

$$
\begin{aligned}
C_{\eta_1} &= C_1 G_1 + \frac{L_{gu}(C_1^2 G_1^2 + 1)}{2}, \\
C\eta_2 &= C_2 G_2 + \frac{L_{gv}(C_2^2 G_2^2 + 1)}{2}, \\
G_1 &= 2G_U\sqrt{d}, \\
G_2 &= 2G_V\sqrt{d}
\end{aligned}
$$

with some constants C_1, C_2 given by Lemma 11 and G_U, G_V given in Lemma 14, 15. For the sufficient decrease with respect to σ, we have

$$
\begin{aligned}
&f^{\text{pca}}(\{\boldsymbol{U}_{i,t}\}_i, \boldsymbol{U}_t, \sigma_t) - f^{\text{pca}}(\{\boldsymbol{U}_{i,t}\}_i, \boldsymbol{U}_t, \sigma_{t-1}) \\
\leq\ & \frac{\partial}{\partial \sigma_{t-1}} f^{\text{pca}}(\{\boldsymbol{U}_{i,t}\}_i, \boldsymbol{U}_t, \sigma_{t-1})(\sigma_t - \sigma_{t-1}) + \frac{L_\sigma}{2}(\sigma_t - \sigma_{t-1})^2 \\
=\ & \left[\frac{\partial}{\partial \sigma_{t-1}} f^{\text{pca}}(\{\boldsymbol{U}_{i,t}\}_i, \boldsymbol{U}_t, \sigma_{t-1})\right] \left[-\eta_3 \frac{\partial}{\partial \sigma_{t-1}} f^{\text{pca}}(\{\boldsymbol{U}_{i,t-1}\}_i, \boldsymbol{U}_{t-1}, \sigma_{t-1})\right] \\
&+ \frac{\eta_3^2 L_\sigma}{2} \left[\frac{\partial}{\partial \sigma_{t-1}} f^{\text{pca}}(\{\boldsymbol{U}_{i,t-1}\}_i, \boldsymbol{U}_{t-1}, \sigma_{t-1})\right]^2 \qquad \text{(by the update rule of } \sigma_t) \\
=\ & (-\eta_3) \left[\frac{\partial}{\partial \sigma_{t-1}} f^{\text{pca}}(\{\boldsymbol{U}_{i,t}\}_i, \boldsymbol{U}_t, \sigma_{t-1}) - \frac{\partial}{\partial \sigma_{t-1}} f^{\text{pca}}(\{\boldsymbol{U}_{i,t-1}\}_i, \boldsymbol{U}_{t-1}, \sigma_{t-1})\right.
\end{aligned}
$$

$$
\begin{aligned}
&+ \frac{\partial}{\partial \sigma_{t-1}} f^{\text{pca}}(\{\boldsymbol{U}_{i,t-1}\}_i, \boldsymbol{U}_{t-1}, \sigma_{t-1})\Bigg] \left[\frac{\partial}{\partial \sigma_{t-1}} f^{\text{pca}}(\{\boldsymbol{U}_{i,t-1}\}_i, \boldsymbol{U}_{t-1}, \sigma_{t-1})\right] \\
&+ \frac{\eta_3^2 L_\sigma}{2} \left[\frac{\partial}{\partial \sigma_{t-1}} f^{\text{pca}}(\{\boldsymbol{U}_{i,t-1}\}_i, \boldsymbol{U}_{t-1}, \sigma_{t-1})\right]^2 \\
={}& (-\eta_3) \left[\frac{\partial}{\partial \sigma_{t-1}} f^{\text{pca}}(\{\boldsymbol{U}_{i,t}\}_i, \boldsymbol{U}_{t}, \sigma_{t-1}) - \frac{\partial}{\partial \sigma_{t-1}} f^{\text{pca}}(\{\boldsymbol{U}_{i,t-1}\}_i, \boldsymbol{U}_{t-1}, \sigma_{t-1})\right] \\
&\times \left[\frac{\partial}{\partial \sigma_{t-1}} f^{\text{pca}}(\{\boldsymbol{U}_{i,t-1}\}_i, \boldsymbol{U}_{t-1}, \sigma_{t-1})\right] \\
&- \left(\eta_3 - \frac{\eta_3^2 L_\sigma}{2}\right) \left[\frac{\partial}{\partial \sigma_{t-1}} f^{\text{pca}}(\{\boldsymbol{U}_{i,t-1}\}_i, \boldsymbol{U}_{t-1}, \sigma_{t-1})\right]^2 \\
\le{}& \frac{\eta_3}{2} \left[\frac{\partial}{\partial \sigma_{t-1}} f^{\text{pca}}(\{\boldsymbol{U}_{i,t}\}_i, \boldsymbol{U}_{t}, \sigma_{t-1}) - \frac{\partial}{\partial \sigma_{t-1}} f^{\text{pca}}(\{\boldsymbol{U}_{i,t-1}\}_i, \boldsymbol{U}_{t-1}, \sigma_{t-1})\right]^2 \\
&+ \frac{\eta_3}{2} \left[\frac{\partial}{\partial \sigma_{t-1}} f^{\text{pca}}(\{\boldsymbol{U}_{i,t-1}\}_i, \boldsymbol{U}_{t-1}, \sigma_{t-1})\right]^2 \\
&- \left(\eta_3 - \frac{\eta_3^2 L_\sigma}{2}\right) \left[\frac{\partial}{\partial \sigma_{t-1}} f^{\text{pca}}(\{\boldsymbol{U}_{i,t-1}\}_i, \boldsymbol{U}_{t-1}, \sigma_{t-1})\right]^2 \\
&\hspace{20em} (\text{since } 2ab \le a^2 + b^2 \text{ for any } a, b \in \mathbb{R}) \\
={}& \frac{\eta_3}{2} \left[\frac{\partial}{\partial \sigma_{t-1}} f^{\text{pca}}(\{\boldsymbol{U}_{i,t}\}_i, \boldsymbol{U}_{t}, \sigma_{t-1}) - \frac{\partial}{\partial \sigma_{t-1}} f^{\text{pca}}(\{\boldsymbol{U}_{i,t-1}\}_i, \boldsymbol{U}_{t-1}, \sigma_{t-1})\right]^2 \\
&- \left(\frac{\eta_3}{2} - \frac{\eta_3^2 L_\sigma}{2}\right) \left[\frac{\partial}{\partial \sigma_{t-1}} f^{\text{pca}}(\{\boldsymbol{U}_{i,t-1}\}_i, \boldsymbol{U}_{t-1}, \sigma_{t-1})\right]^2 \\
={}& \frac{\eta_3}{2} \Bigg[\frac{\partial}{\partial \sigma_{t-1}} f^{\text{pca}}(\{\boldsymbol{U}_{i,t}\}_i, \boldsymbol{U}_{t}, \sigma_{t-1}) - \frac{\partial}{\partial \sigma_{t-1}} f_i^{\text{pca}}(\boldsymbol{U}_{t}, \{\boldsymbol{U}_{i,t-1}\}_i, \sigma_{t-1}) \\
&+ \frac{\partial}{\partial \sigma_{t-1}} f^{\text{pca}}(\{\boldsymbol{U}_{i,t-1}\}_i, \boldsymbol{U}_{t}, \sigma_{t-1}) - \frac{\partial}{\partial \sigma_{t-1}} f^{\text{pca}}(\{\boldsymbol{U}_{i,t-1}\}_i, \boldsymbol{U}_{t-1}, \sigma_{t-1}) \Bigg]^2 \\
&- \left(\frac{\eta_3}{2} - \frac{\eta_3^2 L_\sigma}{2}\right) \left[\frac{\partial}{\partial \sigma_{t-1}} f^{\text{pca}}(\{\boldsymbol{U}_{i,t-1}\}_i, \boldsymbol{U}_{t-1}, \sigma_{t-1})\right]^2 \\
={}& \frac{\eta_3}{2} \Bigg[\left(\frac{1}{m} \sum_{i=1}^{m} \frac{\partial}{\partial \sigma_{t-1}} f_i^{\text{pca}}(\boldsymbol{U}_{i,t}, \boldsymbol{U}_{t}, \sigma_{t-1}) - \frac{\partial}{\partial \sigma_{t-1}} f_i^{\text{pca}}(\boldsymbol{U}_{i,t-1}, \boldsymbol{U}_{t}, \sigma_{t-1}) \right)
\end{aligned}
$$

$$
\begin{aligned}
&+ \frac{\partial}{\partial \sigma_{t-1}} f^{\mathrm{pca}}(\{\boldsymbol{U}_{i,t-1}\}_i, \boldsymbol{U}_t, \sigma_{t-1}) - \frac{\partial}{\partial \sigma_{t-1}} f^{\mathrm{pca}}(\{\boldsymbol{U}_{i,t-1}\}_i, \boldsymbol{U}_{t-1}, \sigma_{t-1}) \Bigg]^2 \\
&- \left(\frac{\eta_3}{2} - \frac{\eta_3^2 L_\sigma}{2}\right) \left[\frac{\partial}{\partial \sigma_{t-1}} f^{\mathrm{pca}}(\{\boldsymbol{U}_{i,t-1}\}_i, \boldsymbol{U}_{t-1}, \sigma_{t-1})\right]^2 \\
\leq\ & \frac{\eta_3}{2} \left[\left(\frac{1}{m}\sum_{i=1}^m L_U^{(\sigma)} \|\boldsymbol{U}_{i,t} - \boldsymbol{U}_{i,t-1}\|_F\right) + L_V^{(\sigma)} \|\boldsymbol{U}_{i,t} - \boldsymbol{U}_{i,t-1}\|_F\right]^2 \\
&- \left(\frac{\eta_3 - \eta_3^2 L_\sigma}{2}\right) \left[\frac{\partial}{\partial \sigma_{t-1}} f^{\mathrm{pca}}(\{\boldsymbol{U}_{i,t-1}\}_i, \boldsymbol{U}_{t-1}, \sigma_{t-1})\right]^2 \qquad \text{(from Lemma 16)} \\
\leq\ & \frac{\eta_3}{2} \left[2\left(\frac{1}{m}\sum_{i=1}^m L_U^{(\sigma)} \|\boldsymbol{U}_{i,t} - \boldsymbol{U}_{i,t-1}\|_F\right)^2 + 2\left(L_V^{(\sigma)} \|\boldsymbol{U}_{i,t} - \boldsymbol{U}_{i,t-1}\|_F\right)^2\right] \\
&- \left(\frac{\eta_3 - \eta_3^2 L_\sigma}{2}\right) \left[\frac{\partial}{\partial \sigma_{t-1}} f^{\mathrm{pca}}(\{\boldsymbol{U}_{i,t-1}\}_i, \boldsymbol{U}_{t-1}, \sigma_{t-1})\right]^2 \qquad \text{(since } (a+b)^2 \leq 2a^2 + 2b^2\text{)} \\
\leq\ & \eta_3 \left[\frac{1}{m}\sum_{i=1}^m \left(L_U^{(\sigma)} \|\boldsymbol{U}_{i,t} - \boldsymbol{U}_{i,t-1}\|_F\right)^2 + \left(L_V^{(\sigma)} \|\boldsymbol{U}_{i,t} - \boldsymbol{U}_{i,t-1}\|_F\right)^2\right] \\
&- \left(\frac{\eta_3 - \eta_3^2 L_\sigma}{2}\right) \left[\frac{\partial}{\partial \sigma_{t-1}} f^{\mathrm{pca}}(\{\boldsymbol{U}_{i,t-1}\}_i, \boldsymbol{U}_{t-1}, \sigma_{t-1})\right]^2 \qquad \text{(Cauchy–Schwarz inequality)} \\
=\ & \eta_3 (L_U^{(\sigma)})^2 \frac{1}{m} \left(\sum_{i=1}^m \|P_{\mathcal{T}_{\boldsymbol{U}_{i,t-1}}} \left(\nabla_{\boldsymbol{U}_{i,t-1}} f^{\mathrm{pca}}(\boldsymbol{U}_{i,t-1}, \boldsymbol{U}_{t-1}, \sigma_{t-1})\right)\|_F^2\right) \\
&+ \eta_3 (L_V^{(\sigma)})^2 \|P_{\mathcal{T}_{\boldsymbol{U}_{t-1}}} \left(\nabla_{\boldsymbol{U}_{t-1}} f^{\mathrm{pca}}(\{\boldsymbol{U}_{i,t}\}_i, \boldsymbol{U}_{t-1}, \sigma_{t-1})\right)\|_F^2 \\
&- \left(\frac{\eta_3 - \eta_3^2 L_\sigma}{2}\right) \left[\frac{\partial}{\partial \sigma_{t-1}} f^{\mathrm{pca}}(\{\boldsymbol{U}_{i,t-1}\}_i, \boldsymbol{U}_{t-1}, \sigma_{t-1})\right]^2.
\end{aligned}
$$

Thus, we have

$$
\begin{aligned}
& f^{\mathrm{pca}}(\{\boldsymbol{U}_{i,t}\}_i, \boldsymbol{U}_t, \sigma_t) - f^{\mathrm{pca}}(\{\boldsymbol{U}_{i,t-1}\}_i, \boldsymbol{U}_{t-1}, \sigma_{t-1}) \\
=\ & [f^{\mathrm{pca}}(\{\boldsymbol{U}_{i,t}\}_i, \boldsymbol{U}_{t-1}, \sigma_{t-1}) - f^{\mathrm{pca}}(\{\boldsymbol{U}_{i,t-1}\}_i, \boldsymbol{U}_{t-1}, \sigma_{t-1})] \\
&+ [f^{\mathrm{pca}}(\{\boldsymbol{U}_{i,t}\}_i, \boldsymbol{U}_t, \sigma_{t-1}) - f^{\mathrm{pca}}(\{\boldsymbol{U}_{i,t}\}_i, \boldsymbol{U}_{t-1}, \sigma_{t-1})] \\
&+ [f^{\mathrm{pca}}(\{\boldsymbol{U}_{i,t}\}_i, \boldsymbol{U}_t, \sigma_t) - f^{\mathrm{pca}}(\{\boldsymbol{U}_{i,t}\}_i, \boldsymbol{U}_t, \sigma_{t-1})] \\
\leq\ & \left(-\eta_1 + C_{\eta_1}\eta_1^2 + \eta_3 (L_U^{(\sigma)})^2\right) \frac{1}{m} \sum_{i=1}^m \|P_{\mathcal{T}_{\boldsymbol{U}_{i,t-1}}} \left(\nabla_{\boldsymbol{U}_{i,t-1}} f_i^{\mathrm{pca}}(\boldsymbol{U}_{i,t-1}, \boldsymbol{U}_{t-1}, , \sigma_{t-1})\right)\|_F^2
\end{aligned}
$$

$$+\left(-\eta_2+C\eta_2\eta_2^2+\eta_3(L_V^{(\sigma)})^2\right)\|P_{\mathcal{T}_{\boldsymbol{U}_{t-1}}}\left(\nabla_{\boldsymbol{U}_{t-1}}f^{\mathrm{pca}}(\{\boldsymbol{U}_{i,t}\}_i,\boldsymbol{U}_{t-1},\sigma_{t-1})\right)\|_F^2$$
$$+\left(\frac{-\eta_3+\eta_3^2L_\sigma}{2}\right)\left[\frac{\partial}{\partial\sigma_{t-1}}f^{\mathrm{pca}}(\{\boldsymbol{U}_{i,t-1}\}_i,\boldsymbol{U}_{t-1},\sigma_{t-1})\right]^2.$$

By choosing $\eta_1=\min\{\frac{1}{3C_{\eta_1}},1\}$, $\eta_2=\min\{\frac{1}{3C\eta_2},1\}$, and $\eta_3=\min\left\{\frac{\eta_1}{3(L_U^{(\sigma)})^2},\frac{\eta_2}{3(L_V^{(\sigma)})^2},\frac{1}{6L_\sigma}\right\}$, we have

$$-\eta_1+C_{\eta_1}\eta_1^2+\eta_3(L_U^{(\sigma)})^2\le\eta_1\left(C_{\eta_1}\eta_1-\frac{1}{3}\right)-\frac{2\eta_1}{3}+\frac{\eta_1}{3}=-\frac{\eta_1}{3},$$
$$-\eta_2+C\eta_2\eta_2^2+\eta_3(L_V^{(\sigma)})^2\le\eta_2\left(C\eta_2\eta_2-\frac{1}{3}\right)-\frac{2\eta_2}{3}+\frac{\eta_2}{3}=-\frac{\eta_2}{3},$$
$$\frac{-\eta_3+\eta_3^2L_\sigma}{2}=\eta_3\left(L_\sigma\eta_3-\frac{1}{6}\right)-\frac{\eta_3}{3}\le-\frac{\eta_3}{3}.$$

Therefore, we obtain

$$\begin{aligned}&f^{\mathrm{pca}}(\{\boldsymbol{U}_{i,t}\}_i,\boldsymbol{U}_t,\sigma_t)-f^{\mathrm{pca}}(\{\boldsymbol{U}_{i,t-1}\}_i,\boldsymbol{U}_{t-1},\sigma_{t-1})\\ \le&-\frac{\eta_1}{3}\left(\frac{1}{m}\sum_{i=1}^m\|P_{\mathcal{T}_{\boldsymbol{U}_{i,t-1}}}\left(\nabla_{\boldsymbol{U}_{i,t-1}}f_i^{\mathrm{pca}}(\boldsymbol{U}_{i,t-1},\boldsymbol{U}_{t-1},,\sigma_{t-1})\right)\|_F^2\right)\\ &-\frac{\eta_2}{3}\|P_{\mathcal{T}_{\boldsymbol{U}_{t-1}}}\left(\nabla_{\boldsymbol{U}_{t-1}}f^{\mathrm{pca}}(\{\boldsymbol{U}_{i,t}\}_i,\boldsymbol{U}_{t-1},\sigma_{t-1})\right)\|_F^2-\frac{\eta_3}{3}\left[\frac{\partial}{\partial\sigma_{t-1}}f^{\mathrm{pca}}(\{\boldsymbol{U}_{i,t-1}\}_i,\boldsymbol{U}_{t-1},\sigma_{t-1})\right]^2.\end{aligned}$$

□

C.3.3.7 Proof of Theorem 8

Proof. Following Lemma 17, by telescoping across the iterations, we have

$$\begin{aligned}&\frac{1}{T}\sum_{t=1}^T\Bigg[\|P_{\mathcal{T}_{\boldsymbol{U}_{t-1}}}\left(\nabla_{\boldsymbol{U}_{t-1}}f^{\mathrm{pca}}(\{\boldsymbol{U}_{i,t}\}_i,\boldsymbol{U}_{t-1},\sigma_{t-1})\right)\|_F^2\\ &+\left(\frac{1}{m}\sum_{i=1}^m\|P_{\mathcal{T}_{\boldsymbol{U}_{i,t-1}}}\left(\nabla_{\boldsymbol{U}_{i,t-1}}f_i^{\mathrm{pca}}(\boldsymbol{U}_{i,t-1},\boldsymbol{U}_{t-1},,\sigma_{t-1})\right)\|_F^2\right)\\ &+\left[\frac{\partial}{\partial\sigma_{t-1}}f^{\mathrm{pca}}(\{\boldsymbol{U}_{i,t-1}\}_i,\boldsymbol{U}_{t-1},\sigma_{t-1})\right]^2\Bigg]\end{aligned}$$

$$\leq \frac{1}{T\min\{\frac{\eta_1}{3}, \frac{\eta_2}{3}, \frac{\eta_3}{3}\}} \sum_{t=1}^{T} f^{\text{pca}}(\{\boldsymbol{U}_{i,t-1}\}_i, \boldsymbol{U}_{t-1}, \sigma_{t-1}) - f^{\text{pca}}(\{\boldsymbol{U}_{i,t}\}_i, \boldsymbol{U}_t, , \sigma_t)$$
$$= \frac{3\left(f^{\text{pca}}(\{\boldsymbol{U}_{i,0}\}_i, \boldsymbol{U}_0, \sigma_0) - f^{\text{pca}}(\{\boldsymbol{U}_{i,T}\}_i, \boldsymbol{U}_T, \sigma_T)\right)}{T\min\{\eta_1, \eta_2, \eta_3\}}.$$

□

C.4 Proofs for Adaptive AEs: ADEPT-AE

C.4.1 Proof Summary

Since the form of the adaptation in the loss function is similar to the PCA loss function, the lower bound on sigma (lemma 1) holds for ADEPT-AE as well. We utilize the lower bound to derive the Lipschitz smoothness constants with respect to $\boldsymbol{\mu}$ and $\boldsymbol{\sigma}$ (lemma 18), and the Lipschitz smoothness constant with respect to $\boldsymbol{\theta}$ is stated in Assumption 2. Then, we derive the sufficient decrease with respect to $\boldsymbol{\theta}$, $\boldsymbol{\mu}$, and σ with the Lipschitz constants. However, we have multiple local iterations for one communication round in ADEPT-AE. Thus, we have to deal with the sufficient decrease separately depending on whether the round is a communication round. With careful derivation under the two cases, we can combine them in the end and get to our theorem 9.

C.4.2 Lemmas

In the following proof, $d = d_\theta$ is the dimension of the models.

Lemma 18. *The loss function $f_i^{\text{ae}}(\boldsymbol{\theta}_i, \boldsymbol{\mu}, \sigma)$ is $L_{\boldsymbol{\mu}}$-smooth w.r.t. $\boldsymbol{\mu}$ and L_σ-smooth w.r.t. σ with*

$$L_{\boldsymbol{\mu}} = \frac{d}{2\xi\omega^2}$$
$$L_\sigma = \frac{3\xi d^2}{2\xi^2\omega^4} + \frac{3d^2B^2}{\xi^2\omega^4} + \frac{d^2}{2\xi\omega^2}.$$

Also, we have

$$\|\nabla_{\boldsymbol{\mu}} f_i^{\text{ae}}(\boldsymbol{\theta}, \boldsymbol{\mu}, \sigma_1) - \nabla_{\boldsymbol{\mu}} f_i^{\text{ae}}(\boldsymbol{\theta}, \boldsymbol{\mu}, \sigma_2)\| \leq L_\sigma^{(\boldsymbol{\mu})} |\sigma_1 - \sigma_2|$$

with

$$L_\sigma^{(\boldsymbol{\mu})} = \frac{B\sqrt{d^3}}{\omega^3\sqrt{2\xi^3}}.$$

Assumption 4. *Assume that there exists some $B > 0$ such that for the weights of the autoencoders, we have $\|\boldsymbol{\mu}\| \leq B$ and $\|\boldsymbol{\theta}_i\| \leq B$ for all $i \in [m]$.*

C.4.3 Proofs

C.4.3.1 Proof of Lemma 18

Proof. Following the same proof in Lemma 1, we have the same lower bound on σ_t if we initialized it in the same way.

The gradient w.r.t. $\boldsymbol{\mu}$ is

$$\nabla_{\boldsymbol{\mu}} f_i^{\text{ae}}(\boldsymbol{\theta}, \boldsymbol{\mu}, \sigma) = \frac{\boldsymbol{\mu} - \boldsymbol{\theta}}{2\sigma^2}.$$

Thus, we have

$$\left\| \frac{\boldsymbol{\mu}_1 - \boldsymbol{\theta}}{2\sigma^2} - \frac{\boldsymbol{\mu}_2 - \boldsymbol{\theta}}{2\sigma^2} \right\| = \left\| \frac{\boldsymbol{\mu}_1 - \boldsymbol{\mu}_2}{2\sigma^2} \right\| \leq \frac{d}{2\xi\omega^2} \|\boldsymbol{\mu}_1 - \boldsymbol{\mu}_2\| \leq L_{\boldsymbol{\mu}} \|\boldsymbol{\mu}_1 - \boldsymbol{\mu}_2\|.$$

For L_σ, we have

$$\begin{aligned}
\left| \frac{\partial^2}{\partial \sigma^2} f_i^{\text{ae}}(\boldsymbol{\theta}, \boldsymbol{\mu}, \sigma) \right| &= \left| \frac{6\xi + 3\|\boldsymbol{\mu} - \boldsymbol{\theta}\|^2}{\sigma^4} - \frac{d}{\sigma^2} \right| \\
&\leq \left| \frac{6\xi + 3\|\boldsymbol{\mu} - \boldsymbol{\theta}\|^2}{\sigma^4} \right| + \left| \frac{d}{\sigma^2} \right| \\
&\leq \frac{6\xi + 3(2B)^2}{4\xi^2\omega^4/d^2} + \frac{d^2}{2\xi\omega^2} \\
&= \frac{3\xi d^2}{2\xi^2\omega^4} + \frac{3d^2B^2}{\xi^2\omega^4} + \frac{d^2}{2\xi\omega^2} \\
&= L_\sigma.
\end{aligned}$$

For $L_\sigma^{(\boldsymbol{\mu})}$, we have

$$\begin{aligned}
\|\nabla_{\boldsymbol{\mu}} f_i^{\mathrm{ae}}(\boldsymbol{\theta}, \boldsymbol{\mu}, \sigma_1) - \nabla_{\boldsymbol{\mu}} f_i^{\mathrm{ae}}(\boldsymbol{\theta}, \boldsymbol{\mu}, \sigma_2)\| &= \left\| \frac{\boldsymbol{\mu} - \boldsymbol{\theta}}{2\sigma_1^2} - \frac{\boldsymbol{\mu} - \boldsymbol{\theta}}{2\sigma_2^2} \right\| \\
&= \left| \frac{1}{\sigma_1^2} - \frac{1}{\sigma_2^2} \right| \frac{\|\boldsymbol{\mu} - \boldsymbol{\theta}\|}{2} \\
&= |\sigma_1 - \sigma_2| \left| \frac{1}{\sigma_1^2 \sigma_2} + \frac{1}{\sigma_1 \sigma_2^2} \right| \frac{\|\boldsymbol{\mu} - \boldsymbol{\theta}\|}{2} \\
&\leq 2 \left(\omega \sqrt{\frac{2\xi}{d}} \right)^{-3} B |\sigma_1 - \sigma_2| \\
&= \frac{B\sqrt{d^3}}{\omega^3 \sqrt{2\xi^3}} |\sigma_1 - \sigma_2| \\
&= L_\sigma^{(\boldsymbol{\mu})} |\sigma_1 - \sigma_2|.
\end{aligned}$$

□

C.4.3.2 Proof of Theorem 9

Proof. Since $\boldsymbol{\mu}_t$ and σ_t are updated only when τ divides t, we consider the two cases separately.

When τ divides t

First, for the sufficient decrease of $\boldsymbol{\theta}_{t,i}$, we have

$$\begin{aligned}
&f_i^{\mathrm{ae}}(\boldsymbol{\theta}_{i,t}, \boldsymbol{\mu}_{t-1}, \sigma_{t-1}) - f_i^{\mathrm{ae}}(\boldsymbol{\theta}_{i,t-1}, \boldsymbol{\mu}_{t-1}, \sigma_{t-1}) \\
&\leq \left\langle \nabla_{\boldsymbol{\theta}_{i,t-1}} f_i^{\mathrm{ae}}(\boldsymbol{\theta}_{i,t-1}, \boldsymbol{\mu}_{t-1}, \sigma_{t-1}), \boldsymbol{\theta}_{i,t} - \boldsymbol{\theta}_{i,t-1} \right\rangle + \frac{L_{\boldsymbol{\theta}}}{2} \|\boldsymbol{\theta}_{i,t} - \boldsymbol{\theta}_{i,t-1}\|^2 \\
&= \left\langle \nabla_{\boldsymbol{\theta}_{i,t-1}} f_i^{\mathrm{ae}}(\boldsymbol{\theta}_{i,t-1}, \boldsymbol{\mu}_{t-1}, \sigma_{t-1}), -\eta_1 \nabla_{\boldsymbol{\theta}_{i,t-1}} f_i^{\mathrm{ae}}(\boldsymbol{\theta}_{i,t-1}, \boldsymbol{\mu}_{t-1}, \sigma_{t-1}) \right\rangle \\
&\quad + \frac{L_{\boldsymbol{\theta}}}{2} \left\| -\eta_1 \nabla_{\boldsymbol{\theta}_{i,t-1}} f_i^{\mathrm{ae}}(\boldsymbol{\theta}_{i,t-1}, \boldsymbol{\mu}_{i,t-1}, \sigma_{t-1}) \right\|^2 \\
&\leq \left(-\eta_1 + \eta_1^2 \frac{L_{\boldsymbol{\theta}}}{2} \right) \left\| \nabla_{\boldsymbol{\theta}_{i,t-1}} f_i^{\mathrm{ae}}(\boldsymbol{\theta}_{i,t-1}, \boldsymbol{\mu}_{t-1}, \sigma_{t-1}) \right\|^2.
\end{aligned}$$

Sum over the clients and we have

$$f^{\mathrm{ae}}(\{\boldsymbol{\theta}_{i,t}\}, \boldsymbol{\mu}_{t-1}, \sigma_{t-1}) - f^{\mathrm{ae}}(\{\boldsymbol{\theta}_{i,t-1}\}, \boldsymbol{\mu}_{t-1}, \sigma_{t-1})$$

$$\leq \left(-\eta_1 + \eta_1^2 \frac{L_{\boldsymbol{\theta}}}{2}\right) \left(\frac{1}{m}\sum_{i=1}^{m} \left\|\nabla_{\boldsymbol{\theta}_{i,t-1}} f_i^{\text{ae}}(\boldsymbol{\theta}_{i,t-1}, \boldsymbol{\mu}_{t-1}, \sigma_{t-1})\right\|^2\right). \tag{C.3}$$

Second, for the sufficient decrease of σ_t, define

$$g_t^{\sigma} = \frac{1}{m}\sum_{i=1}^{m} \frac{\partial}{\partial \sigma_{t-1}} f_i^{\text{ae}}(\boldsymbol{\theta}_{i,t}, \boldsymbol{\mu}_{t-1}, \sigma_{t-1}).$$

Thus, $\sigma_t = \sigma_{t-1} - \eta_2 g_t^{\sigma}$ and

$$\begin{aligned}
&f_i^{\text{ae}}(\boldsymbol{\theta}_{i,t}, \boldsymbol{\mu}_{t-1}, \sigma_t) - f_i^{\text{ae}}(\boldsymbol{\theta}_{i,t}, \boldsymbol{\mu}_{t-1}, \sigma_{t-1}) \\
&\leq \left(\frac{\partial}{\partial \sigma_{t-1}} f_i^{\text{ae}}(\boldsymbol{\theta}_{i,t}, \boldsymbol{\mu}_{t-1}, \sigma_{t-1})\right)(\sigma_t - \sigma_{t-1}) + \frac{L_{\sigma}}{2}(\sigma_t - \sigma_{t-1})^2 \\
&= \left(\frac{\partial}{\partial \sigma_{t-1}} f_i^{\text{ae}}(\boldsymbol{\theta}_{i,t}, \boldsymbol{\mu}_{t-1}, \sigma_{t-1})\right)(-\eta_2 g_t^{\sigma}) + \frac{L_{\sigma}}{2}(-\eta_2 g_t^{\sigma})^2.
\end{aligned}$$

Sum over the clients and we have

$$\begin{aligned}
f^{\text{ae}}(\{\boldsymbol{\theta}_{i,t}\}, \boldsymbol{\mu}_{t-1}, \sigma_t) - f^{\text{ae}}(\{\boldsymbol{\theta}_{i,t}\}, \boldsymbol{\mu}_{t-1}, \sigma_{t-1}) &\leq -\eta_2 (g_t^{\sigma})^2 + \eta_2^2 \frac{L_{\sigma}}{2} (g_t^{\sigma})^2 \\
&= \left(-\eta_2 + \eta_2^2 \frac{L_{\sigma}}{2}\right)(g_t^{\sigma})^2.
\end{aligned} \tag{C.4}$$

Then, for the sufficient decrease of $\boldsymbol{\mu}_t$, define

$$\begin{aligned}
\mathbf{g}_{i,t}^{\boldsymbol{\mu}} &= \nabla_{\boldsymbol{\mu}_{t-1}} f_i^{\text{ae}}(\boldsymbol{\theta}_{i,t}, \boldsymbol{\mu}_{t-1}, \sigma_{t-1}), \qquad & \mathbf{g}_t^{\boldsymbol{\mu}} &= \frac{1}{m}\sum_{i=1}^{m} \mathbf{g}_{i,t}^{\boldsymbol{\mu}}, \\
\tilde{\mathbf{g}}_{i,t}^{\boldsymbol{\mu}} &= \nabla_{\boldsymbol{\mu}_{t-1}} f_i^{\text{ae}}(\boldsymbol{\theta}_{i,t}, \boldsymbol{\mu}_{t-1}, \sigma_t), \qquad & \tilde{\mathbf{g}}_t^{\boldsymbol{\mu}} &= \frac{1}{m}\sum_{i=1}^{m} \tilde{\mathbf{g}}_{i,t}^{\boldsymbol{\mu}}.
\end{aligned}$$

We have

$$\begin{aligned}
&f^{\text{ae}}(\{\boldsymbol{\theta}_{i,t}\}, \boldsymbol{\mu}_t, \sigma_t) - f^{\text{ae}}(\{\boldsymbol{\theta}_{i,t}\}, \boldsymbol{\mu}_{t-1}, \sigma_t) \\
&\leq \langle \tilde{\mathbf{g}}_t^{\boldsymbol{\mu}}, \boldsymbol{\mu}_t - \boldsymbol{\mu}_{t-1} \rangle + \frac{L_{\boldsymbol{\mu}}}{2} \|\boldsymbol{\mu}_t - \boldsymbol{\mu}_{t-1}\|^2 \\
&= -\eta_3 \langle \tilde{\mathbf{g}}_t^{\boldsymbol{\mu}} - \mathbf{g}_t^{\boldsymbol{\mu}} + \mathbf{g}_t^{\boldsymbol{\mu}}, \mathbf{g}_t^{\boldsymbol{\mu}} \rangle + \frac{L_{\boldsymbol{\mu}} \eta_3^2}{2} \|\mathbf{g}_t^{\boldsymbol{\mu}}\|^2 \\
&= \left(-\eta_3 + \frac{L_{\boldsymbol{\mu}} \eta_3^2}{2}\right) \|\mathbf{g}_t^{\boldsymbol{\mu}}\|^2 + \eta_3 \langle \mathbf{g}_t^{\boldsymbol{\mu}} - \tilde{\mathbf{g}}_t^{\boldsymbol{\mu}}, \mathbf{g}_t^{\boldsymbol{\mu}} \rangle
\end{aligned}$$

$$
\begin{aligned}
&\leq \left(-\eta_3 + \frac{L_{\boldsymbol{\mu}}\eta_3^2}{2}\right)\|\mathbf{g}_t^{\boldsymbol{\mu}}\|^2 + \frac{\eta_3}{2}\|\mathbf{g}_t^{\boldsymbol{\mu}} - \tilde{\mathbf{g}}_t^{\boldsymbol{\mu}}\|^2 + \frac{\eta_3}{2}\|\mathbf{g}_t^{\boldsymbol{\mu}}\|^2 \\
&\leq \left(-\frac{\eta_3}{2} + \frac{L_{\boldsymbol{\mu}}\eta_3^2}{2}\right)\|\mathbf{g}_t^{\boldsymbol{\mu}}\|^2 + \frac{\eta_3 {L_\sigma^{(\boldsymbol{\mu})}}^2}{2}(\sigma_t - \sigma_{t-1})^2 \\
&\leq \left(-\frac{\eta_3}{2} + \frac{L_{\boldsymbol{\mu}}\eta_3^2}{2}\right)\|\mathbf{g}_t^{\boldsymbol{\mu}}\|^2 + \frac{\eta_3\eta_2^2 {L_\sigma^{(\boldsymbol{\mu})}}^2}{2}(g_t^\sigma)^2 \\
&\leq \left(-\frac{\eta_3}{2} + \frac{L_{\boldsymbol{\mu}}\eta_3^2}{2}\right)\|\mathbf{g}_t^{\boldsymbol{\mu}}\|^2 + \frac{\eta_2^2 {L_\sigma^{(\boldsymbol{\mu})}}^2}{2}(g_t^\sigma)^2.
\end{aligned}
$$

Finally, we have the overall decrease when τ divides t by summing with equation (C.4), (C.3),

$$
\begin{aligned}
&f^{\text{ae}}(\{\boldsymbol{\theta}_{i,t}\}, \boldsymbol{\mu}_t, \sigma_t) - f^{\text{ae}}(\{\boldsymbol{\theta}_{i,t}\}, \boldsymbol{\mu}_{t-1}, \sigma_{t-1}) \\
&\leq \left(-\eta_1 + \eta_1^2\frac{L_{\boldsymbol{\theta}}}{2}\right)\left(\frac{1}{m}\sum_{i=1}^m \left\|\nabla_{\boldsymbol{\theta}_{i,t-1}} f_i^{\text{ae}}(\boldsymbol{\theta}_{i,t-1}, \boldsymbol{\mu}_{t-1}, \sigma_{t-1})\right\|^2\right) \\
&\quad + \left(-\eta_2 + \eta_2^2\frac{L_\sigma + {L_\sigma^{(\boldsymbol{\mu})}}^2}{2}\right)(g_t^\sigma)^2 + \left(-\frac{\eta_3}{2} + \frac{L_{\boldsymbol{\mu}}\eta_3^2}{2}\right)\|\mathbf{g}_t^{\boldsymbol{\mu}}\|^2.
\end{aligned}
$$

When τ does not divide t

At time steps that are not communication rounds we simply have a decrease due to the updates of $\{\boldsymbol{\theta}_i\}$, that is,

$$
\begin{aligned}
&f^{\text{ae}}(\{\boldsymbol{\theta}_{i,t}\}, \boldsymbol{\mu}_t, \sigma_t) - f^{\text{ae}}(\{\boldsymbol{\theta}_{i,t-1}\}, \boldsymbol{\mu}_{t-1}, \sigma_{t-1}) \\
&= f^{\text{ae}}(\{\boldsymbol{\theta}_{i,t}\}, \boldsymbol{\mu}_{t-1}, \sigma_{t-1}) - f^{\text{ae}}(\{\boldsymbol{\theta}_{i,t-1}\}, \boldsymbol{\mu}_{t-1}, \sigma_{t-1}) \\
&= \frac{1}{m}\sum_{i=1}^m (f_i^{\text{ae}}(\boldsymbol{\theta}_{i,t}, \boldsymbol{\mu}_{t-1}, \sigma_{t-1}) - f_i^{\text{ae}}(\boldsymbol{\theta}_{i,t-1}, \boldsymbol{\mu}_{t-1}, \sigma_{t-1})) \\
&\leq \left(-\eta_1 + \eta_1^2\frac{L_{\boldsymbol{\theta}}}{2}\right)\left(\frac{1}{m}\sum_{i=1}^m \left\|\nabla_{\boldsymbol{\theta}_{i,t-1}} f_i^{\text{ae}}(\boldsymbol{\theta}_{i,t-1}, \boldsymbol{\mu}_{t-1}, \sigma_{t-1})\right\|^2\right).
\end{aligned}
$$

The final bound By choosing, $\eta_1 = \frac{1}{L_{\boldsymbol{\theta}}}, \eta_2 = \frac{1}{L_\sigma + {L_\sigma^{(\boldsymbol{\mu})}}^2}, \eta_3 = \min\{1, \frac{1}{L_{\boldsymbol{\mu}}}\}$, and by averaging over time steps while combining two type of decrease, we obtain

$$\frac{1}{T}\sum_{t=1}^{T}\left(\frac{1}{m}\sum_{i=1}^{m}\left\|\nabla_{\boldsymbol{\theta}_{i,t-1}} f_i^{\text{ae}}(\boldsymbol{\theta}_{i,t-1},\boldsymbol{\mu}_{t-1},\sigma_{t-1})\right\|^2\right)+\frac{1}{T}\sum_{\substack{t=1\\ t\%\tau=0}}^{T}\|\mathbf{g}_t^{\boldsymbol{\mu}}\|^2+\frac{1}{T}\sum_{\substack{t=1\\ t\%\tau=0}}^{T}(g_t^{\sigma})^2$$
$$\leq \frac{\max\{L_{\boldsymbol{\theta}}, L_\sigma + L_\sigma^{(\boldsymbol{\mu})^2}, L_{\boldsymbol{\mu}}, 1\}\Delta_T}{T},$$

where $\Delta_T = f^{\text{ae}}(\{\boldsymbol{\theta}_{i,0}\},\boldsymbol{\mu}_0,\sigma_0) - f^{\text{ae}}(\{\boldsymbol{\theta}_{i,T}\},\boldsymbol{\mu}_T,\sigma_T)$. Given that τ is a finite constant let us denote $R = T/\tau$ as the number of communication rounds. Then we have,

$$\begin{aligned}
&\frac{1}{T}\sum_{t=1}^{T}\left(\frac{1}{m}\sum_{i=1}^{m}\left\|\mathbf{g}_{i,t}^{\boldsymbol{\theta}}\right\|^2\right)+\frac{1}{T}\sum_{\substack{t=1\\ t\%\tau=0}}^{T}\|\mathbf{g}_t^{\boldsymbol{\mu}}\|^2+\frac{1}{T}\sum_{\substack{t=1\\ t\%\tau=0}}^{T}(g_t^{\sigma})^2\\
&\quad=\frac{R}{T}\left(\frac{1}{R}\sum_{t=1}^{T}\left(\frac{1}{m}\sum_{i=1}^{m}\left\|\mathbf{g}_{i,t}^{\boldsymbol{\theta}}\right\|^2\right)+\frac{1}{R}\sum_{\substack{t=1\\ t\%\tau=0}}^{T}\|\mathbf{g}_t^{\boldsymbol{\mu}}\|^2+\frac{1}{R}\sum_{\substack{t=1\\ t\%\tau=0}}^{T}(g_t^{\sigma})^2\right)\\
&\quad\geq\frac{R}{T}\left(\frac{1}{R}\sum_{\substack{t=1\\ t\%\tau=0}}^{T}\left(\frac{1}{m}\sum_{i=1}^{m}\left\|\mathbf{g}_{i,t}^{\boldsymbol{\theta}}\right\|^2\right)+\frac{1}{R}\sum_{\substack{t=1\\ t\%\tau=0}}^{T}\|\mathbf{g}_t^{\boldsymbol{\mu}}\|^2+\frac{1}{R}\sum_{\substack{t=1\\ t\%\tau=0}}^{T}(g_t^{\sigma})^2\right)\\
&\quad\geq\frac{R}{T}\min_{t\in[T],\tau|t}\left\{\left\|\mathbf{g}_{i,t}^{\boldsymbol{\theta}}\right\|^2+\|\mathbf{g}_t^{\boldsymbol{\mu}}\|^2+(g_t^{\sigma})^2\right\}.
\end{aligned}$$

Finally this yields,

$$\min_{t\in[T],\tau|t}\left\{\left\|\mathbf{g}_{i,t}^{\boldsymbol{\theta}}\right\|^2+\|\mathbf{g}_t^{\boldsymbol{\mu}}\|^2+(g_t^{\sigma})^2\right\}\leq\frac{\max\{L_{\boldsymbol{\theta}}, L_\sigma + L_\sigma^{(\boldsymbol{\mu})^2}, L_{\boldsymbol{\mu}}, 1\}\Delta_T^{\text{ae}}}{R},$$

where $\Delta_T^{\text{ae}} = f^{\text{ae}}(\{\boldsymbol{\theta}_{i,0}\}_i,\boldsymbol{\mu}_0,\sigma_0) - f^{\text{ae}}(\{\boldsymbol{\theta}_{i,T}\}_i,\boldsymbol{\mu}_T,\sigma_T)$.

□

Remark 16. *In the experiments for* `ADEPT-AE`*, we treat sigma as a vector* $\boldsymbol{\sigma}\in\mathbb{R}^d$ *so that each weight in the models can learn its own* σ_j *instead of sharing one* σ *across all the weights. The convergence for the modified algorithm is almost identical to the proof here. Moreover, it can be shown that for the Lipschitz smoothness constant* L_σ*, the dependence on the dimension in the numerators becomes* d *instead of* d^2*. This is because our lower bound in lemma 1 will not depend on* d *in this case.*

C.5 Details and Proofs for Adaptive Diffusion Model: ADEPT-DGM

The main change compared to ADEPT-AE is that we input a corrupted sample to the network during the forward pass (**line 11**) to train it as a denoiser. Accordingly, $f^{\text{df}}_{i,\boldsymbol{\alpha}}(\boldsymbol{\theta}_i, \boldsymbol{\mu}, \sigma) := \|\boldsymbol{\phi}(\boldsymbol{X}(1-\boldsymbol{\alpha}) + \boldsymbol{Z}\boldsymbol{\alpha}; \boldsymbol{\theta}_i) - \boldsymbol{X}\|^2 + \frac{2\xi + \|\boldsymbol{\mu}-\boldsymbol{\theta}_i\|^2}{2\sigma^2} + d\log\sigma$, α is the random noise amount.

C.5.1 Proof of Lemma 2

Proof of Lemma 2. Let $\beta_t = \frac{1}{T-t+\sigma_0^2}$. For the stochastic differential equation

$$d\boldsymbol{x}_t^{\leftarrow} + \beta_t(\boldsymbol{x}_t^{\leftarrow} - \hat{\boldsymbol{\theta}})dt = d\boldsymbol{w}_t, \quad \boldsymbol{x}_0^{\leftarrow} \sim \mathcal{N}(0, (\sigma_0^2 + T)\boldsymbol{I}_d),$$

we have

$$\begin{aligned} &d(e^{\int_0^t \beta_s ds}(\boldsymbol{x}_t^{\leftarrow} - \hat{\boldsymbol{\theta}})) = e^{\int_0^t \beta_s ds} d\boldsymbol{x}_t^{\leftarrow} + e^{\int_0^t \beta_s ds}\beta_t(\boldsymbol{x}_t^{\leftarrow} - \hat{\boldsymbol{\theta}})dt \\ &= e^{\int_0^t \beta_s ds}\left(d\boldsymbol{x}_t^{\leftarrow} + \beta_t(\boldsymbol{x}_t^{\leftarrow} - \hat{\boldsymbol{\theta}})dt\right) = e^{\int_0^t \beta_s ds} d\boldsymbol{w}_t. \end{aligned}$$

Note that

$$e^{\int_0^t \beta_s ds} = e^{\int_0^t \frac{1}{T-s+\sigma_0^2} ds} = e^{\ln(T+\sigma_0^2) - \ln(T-t+\sigma_0^2)} = \frac{T+\sigma_0^2}{T-t+\sigma_0^2}$$

and

$$\int_0^T e^{2\int_0^t \beta_s ds} dt = \int_0^T \frac{(T+\sigma_0^2)^2}{(T+\sigma_0^2-t)^2} dt = (T+\sigma_0^2)^2\left(\frac{1}{\sigma_0^2} - \frac{1}{T+\sigma_0^2}\right) = \left(1 + \frac{T}{\sigma_0^2}\right)T.$$

It then follows that

$$e^{\int_0^T \beta_s ds}(\boldsymbol{x}_T^{\leftarrow} - \hat{\boldsymbol{\theta}}) - (\boldsymbol{x}_0^{\leftarrow} - \hat{\boldsymbol{\theta}}) \sim \mathcal{N}\left(0, \int_0^T e^{2\int_0^t \beta_s ds} dt\right) = \mathcal{N}\left(0, \left(1 + \frac{T}{\sigma_0^2}\right)T\right),$$

and equivalently

$$\boldsymbol{x}_T^{\leftarrow} = \hat{\boldsymbol{\theta}} + \frac{\sigma_0^2}{\sigma_0^2 + T}(\boldsymbol{x}_0^{\leftarrow} - \hat{\boldsymbol{\theta}}) + \sqrt{\frac{\sigma_0^2 T}{\sigma_0^2 + T}}\boldsymbol{\epsilon},$$

where $\boldsymbol{\epsilon} \sim \mathcal{N}(0, \boldsymbol{I}_d)$.

Since $\boldsymbol{x}_0^{\leftarrow} \sim \mathcal{N}(0, (\sigma_0^2+T)\boldsymbol{I}_d)$, we have $p_{\boldsymbol{x}_T^{\leftarrow}|\hat{\boldsymbol{\theta}}} = \mathcal{N}\left(\hat{\boldsymbol{\theta}} - \frac{\sigma_0^2}{\sigma_0^2+T}\hat{\boldsymbol{\theta}}, \sigma_0^2\boldsymbol{I}_d\right)$. The KL-divergence between the target distribution $p_{\boldsymbol{x}|\boldsymbol{\theta}} = \mathcal{N}(\boldsymbol{\theta}, \sigma_0^2\boldsymbol{I}_d)$ and $p_{\boldsymbol{x}_T^{\leftarrow}|\hat{\boldsymbol{\theta}}}$ can then be calculated as $\left\|\boldsymbol{\theta} - \hat{\boldsymbol{\theta}} + \frac{\sigma_0^2}{\sigma_0^2+T}\hat{\boldsymbol{\theta}}\right\|^2$. □

C.5.2 Proof of Lemma 3

Proof of Lemma 3. The parameterized score function is $\boldsymbol{\phi}(x;\boldsymbol{\theta},t) = -\frac{\boldsymbol{x}-\boldsymbol{\theta}}{\sigma_0^2+t}$. Note that $\nabla_{\boldsymbol{x}_t} \ln p_{\boldsymbol{x}_t|\boldsymbol{x}_0}(\boldsymbol{x}_t|\boldsymbol{x}_0) = -\frac{\boldsymbol{x}_t-\boldsymbol{x}_0}{t}$ since $\boldsymbol{x}_t|\boldsymbol{x}_0 \sim \mathcal{N}(\boldsymbol{x}_0, t\boldsymbol{I}_d)$. The training loss of (4.10) can then be written as

$$\frac{1}{m}\sum_{i=1}^{m}\sum_{j=1}^{n}\int_{t=0}^{T}\mathbb{E}_{\boldsymbol{\epsilon}\sim\mathcal{N}(0,\boldsymbol{I}_d)}\left[\frac{1}{2}\left\|\boldsymbol{\phi}(\boldsymbol{x}_{ij}+\sqrt{t}\boldsymbol{\epsilon};\boldsymbol{\theta}_j,t)+\boldsymbol{\epsilon}/\sqrt{t}\right\|^2\right]dt + \sum_{j=1}^{m}\frac{2\xi+\|\boldsymbol{\theta}_j-\boldsymbol{\mu}\|^2}{2\sigma^2}+\frac{d}{2}\ln\sigma^2.$$

Note that

$$\begin{aligned}
&\int_{t=0}^{T}\mathbb{E}_{\boldsymbol{\epsilon}\sim\mathcal{N}(0,\boldsymbol{I}_d)}[\|\boldsymbol{\phi}(\boldsymbol{x}_{ij}+\sqrt{t}\boldsymbol{\epsilon};\boldsymbol{\theta},t)+\boldsymbol{\epsilon}/\sqrt{t}\|^2]dt\\
&=\int_{t=0}^{T}\mathbb{E}_{\boldsymbol{\epsilon}\sim\mathcal{N}(0,\boldsymbol{I}_d)}\left[\left\|-\frac{\boldsymbol{x}_{ij}+\sqrt{t}\boldsymbol{\epsilon}-\boldsymbol{\theta}}{\sigma_0^2+t}+\frac{\boldsymbol{\epsilon}}{\sqrt{t}}\right\|^2\right]dt\\
&=\int_{0}^{T}\mathbb{E}_{\boldsymbol{\epsilon}\sim\mathcal{N}(0,\boldsymbol{I}_d)}\left[\left\|\frac{\boldsymbol{\theta}-\boldsymbol{x}_{ij}}{\sigma_0^2+t}+\frac{\sigma_0^2}{\sqrt{t}(\sigma_0^2+t)}\boldsymbol{\epsilon}\right\|^2\right]dt\\
&=\left(\int_0^T\frac{1}{(\sigma_0^2+t)^2}dt\right)\|\boldsymbol{\theta}-\boldsymbol{x}_{ij}\|^2+\text{const.}
\end{aligned}$$

Since $\int_0^T \frac{1}{(\sigma_0^2+t)^2}dt = \frac{1}{\sigma_0^2}-\frac{1}{\sigma_0^2+T}$, the optimization of minimizing the training loss is equivalent to

$$\min_{\boldsymbol{\theta}_{1:m},\boldsymbol{\theta},\sigma^2}\quad \frac{1}{m}\sum_{i=1}^{m}\left(\sum_{j=1}^{n}\frac{\alpha}{2}\|\boldsymbol{\theta}_j-\boldsymbol{x}_{ij}\|^2+\frac{2\xi+\|\boldsymbol{\theta}_j-\boldsymbol{\mu}\|^2}{2\sigma^2}\right)+\frac{d}{2}\ln\sigma^2,$$

where $\alpha = \frac{1}{\sigma_0^2}-\frac{1}{\sigma_0^2+T}$. By the KKT condition that

$$\begin{aligned}
&\sum_{j=1}^{n}\alpha(\boldsymbol{\theta}_i-\boldsymbol{x}_{ij})+\frac{\boldsymbol{\theta}_i-\boldsymbol{\mu}}{\sigma^2}=0,\quad \forall i=1,2,\ldots,m,\\
&\boldsymbol{\mu}=\frac{1}{m}\sum_{i=1}^{m}\boldsymbol{\theta}_i,\\
&-\frac{1}{m}\sum_{i=1}^{m}\frac{2\xi+\|\boldsymbol{\theta}_j-\boldsymbol{\mu}\|^2}{2\sigma^4}+\frac{d}{2\sigma^2}=0,
\end{aligned}$$

We thus have

$$\hat{\boldsymbol{\mu}} = \frac{\sum_{i=1}^{m}\sum_{j=1}^{n}\boldsymbol{x}_{ij}}{mn}$$
$$\hat{\boldsymbol{\theta}}_i = \frac{n\alpha\hat{\sigma}^2}{n\alpha\hat{\sigma}^2+1}\frac{\sum_{j=1}^{n}\boldsymbol{x}_{ij}}{n} + \frac{\hat{\boldsymbol{\mu}}}{n\alpha\hat{\sigma}^2+1},$$

where $\hat{\sigma}^2$ satisfies $\hat{\sigma}^2 = \frac{2\xi}{d} + s^2(\frac{n\alpha\hat{\sigma}^2}{n\alpha\hat{\sigma}^2+1})^2$ with $s^2 = \frac{\sum_{i=1}^{m}\left\|\hat{\boldsymbol{\mu}}-\frac{1}{n}\sum_{j=1}^{n}\boldsymbol{x}_{ij}\right\|^2}{md}$. □

C.5.3 Proof of Theorem 10 and Corollary 1

Proof of Theorem 10. When $m\to\infty$, $s^2 \to \frac{\sigma_0^2}{n}+\sigma_*^2$ and $\hat{\boldsymbol{\mu}}\to\boldsymbol{\mu}_*$, a.s.. $\hat{\sigma}^2$ satisfies

$$\hat{\sigma}^2 = \frac{2\xi}{d} + (\frac{\sigma_0^2}{n}+\sigma_*^2)(\frac{\hat{\sigma}^2}{\hat{\sigma}^2+1/(n\alpha)})^2.$$

Since $\boldsymbol{\theta}_i$ are sampled i.i.d. from a population distribution $\mathcal{N}(\boldsymbol{\mu}_*, \sigma_*^2\boldsymbol{I}_d)$. $\alpha = 1/\sigma_0^2$ since $T\to\infty$. Let $\boldsymbol{x}\sim\mathcal{N}(\boldsymbol{\theta}, \frac{\sigma_0^2}{n}\boldsymbol{I}_d)$, $\boldsymbol{\theta}\sim\mathcal{N}(\boldsymbol{\mu}, \sigma^2\boldsymbol{I}_d)$, by Lemma 2, we have

$$\begin{aligned}
\frac{1}{m}\sum_{i=1}^{m} D_{KL}(p_{\boldsymbol{x}|\boldsymbol{\theta}_i}||p_{\boldsymbol{x}_T^{\leftarrow}|\hat{\boldsymbol{\theta}}_i}) &= \frac{1}{m}\sum_{i=1}^{m}\left\|\boldsymbol{\theta}_i - \hat{\boldsymbol{\theta}}_i + \frac{\sigma_0^2}{\sigma_0^2+T}\hat{\boldsymbol{\theta}}_i\right\|^2 \\
&= \mathbb{E}\left[\left\|\boldsymbol{\theta}-\boldsymbol{x}-\frac{1}{n\alpha\hat{\sigma}^2+1}(\boldsymbol{\mu}-\boldsymbol{x})\right\|^2\right] \\
&= (\frac{\sigma_0^2/n}{\hat{\sigma}^2+\sigma_0^2/n})^2\mathbb{E}\left[\|\boldsymbol{\theta}-\boldsymbol{\mu}\|^2\right] + (\frac{\hat{\sigma}^2}{\hat{\sigma}^2+\sigma_0^2/n})^2\mathbb{E}\left[\|\boldsymbol{\theta}-\boldsymbol{x}\|^2\right] \\
&= (\frac{\sigma_0^2/n}{\hat{\sigma}^2+\sigma_0^2/n})^2\sigma_*^2 + (\frac{\hat{\sigma}^2}{\hat{\sigma}^2+\sigma_0^2/n})^2\sigma_0^2/n \\
&= \frac{\sigma_0^2}{n} + \frac{(\sigma_*^2+\sigma_0^2/n)\sigma_0^2/n}{(\hat{\sigma}^2+\sigma_0^2/n)^2}\frac{\sigma_0^2}{n} - 2\frac{\sigma_0^2/n}{\hat{\sigma}^2+\sigma_0^2/n}\frac{\sigma_0^2}{n} \\
&= \frac{\sigma_0^2}{n} - \left(\frac{2\hat{\sigma}^2+\sigma_0^2/n-\sigma_*^2}{\hat{\sigma}^2+\sigma_0^2/n}\right)\frac{\sigma_0^2/n}{\hat{\sigma}^2+\sigma_0^2/n}\frac{\sigma_0^2}{n}
\end{aligned}$$

where the expectation is taken w.r.t. $\boldsymbol{x}, \boldsymbol{\theta}$.

Since without collaboration, the training of maximizing $ELBO_i$ for client-i leads to parameter $\hat{\boldsymbol{\theta}}_i = \frac{\sum_{j=1}^{n}\boldsymbol{x}_{ij}}{n}$ and the KL-divergence between the target distribution and the output distribution is $\frac{\sigma_0^2}{n}$, it follows that collaboration improves the performance as long as $\hat{\sigma}^2 > \frac{\sigma_*^2}{2} - \frac{\sigma_0^2}{2n}$.

The improvement is $\left(\frac{2\hat{\sigma}^2+\sigma_0^2/n-\sigma_*^2}{\hat{\sigma}^2+\sigma_0^2/n}\right)\frac{\sigma_0^2/n}{\hat{\sigma}^2+\sigma_0^2/n}\frac{\sigma_0^2}{n}$ and achieves the maximum when $\hat{\sigma}^2=\sigma_*^2$, i.e., the learned $\hat{\sigma}^2=\sigma_*^2$ □

Proof of Corollary 1. Under the same setting as in Theorem 10, by Lemma 3, we have $\hat{\sigma}^2 = \frac{2\xi}{d} + (\frac{\sigma_0^2}{n}+\sigma_*^2)(\frac{\hat{\sigma}^2}{\hat{\sigma}^2+\sigma_0^2/n})^2$. Taking $\xi > \frac{2\xi}{d} = \frac{3d\sigma_0^2}{2n}$ gives that $\hat{\sigma}^2 \geq \frac{3\sigma_0^2}{n}$, and thus $\hat{\sigma}^2 > (\frac{\sigma_0^2}{n}+\sigma_*^2)(\frac{\hat{\sigma}^2}{\hat{\sigma}^2+\sigma_0^2/n})^2 \geq \frac{9}{16}(\frac{\sigma_0^2}{n}+\sigma_*^2) > \frac{\sigma_*^2}{2} - \frac{\sigma_0^2}{2n}$, which guarantees strictly improvement by Theorem 10. □

C.6 Additional Experiments and Experimental Details

C.6.1 Experiments for ADEPT-PCA on synthetic data

We use synthetic datasets for the experiments of ADEPT-PCA. In the dataset, we first sample a global PC, $\boldsymbol{U}^* \in St(d,r)$, uniformly on the Steifel manifold. We sample $\{\hat{\boldsymbol{U}}_i^*\}_{i=1}^m$ where the entries of each $\hat{\boldsymbol{U}}_i^*$ follows Gaussian distribution with mean being $\boldsymbol{U}^*$ and variance σ^*. Then, we let $\boldsymbol{U}_i^* = \mathcal{R}_{\boldsymbol{U}^*}(P_{\mathcal{T}_{\boldsymbol{U}^*}}(\hat{\boldsymbol{U}}_i^*))$ so that it is in the Steifel manifold. Data on each client are then generated by $\mathbf{x} = \boldsymbol{U}_i^*\mathbf{z} + \boldsymbol{\epsilon}$.

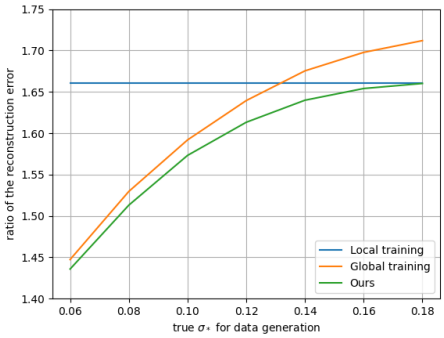


Figure C.1 Ratio of the reconstruction error to the optimal error for different values of σ^*. We have $d = 100$, $r = 20$, $m = 10$, and $n = 20$.

We compare the reconstruction error between Algorithm 15, local training, and global training. In the global training setting, we train a single global model with the average of local gradients in each iteration. In Figure C.1, the value in the y-axis is the ratio of the

reconstruction error of each training method to the true error, which is evaluated by $\{\boldsymbol{U}_i^*\}_{i=1}^m$. When σ^* is small, the heterogeneity of the data among the clients is low, and thus global training benefits from the sample size and performs better than pure local training. Our algorithm also makes use of the sample size and achieves an even smaller reconstruction error with the personalized models. When σ^* is large, the heterogeneity of the data among the clients is high and thus training a single global model for each client does not work well. In this case, our algorithm learns a larger σ and performs more like local training. In the scenario between the two cases, our algorithm also outperforms both global and local training.

C.6.2 Training and hyper-parameter Details

ADEPT-AE. We set $\xi = 1e{-}6$ for all experiments. For the synthetic experiments, we don't apply local iterations and just do distributed training. For the other experiments, we do 20 local iterations per communication round, and each communication round corresponds to an epoch. We use 300 global batch size for MNIST and 750 for CIFAR-10. For all datasets and methods, we use SGD with a constant learning rate of 0.01 after individually tuning in the set $\{0.5, 0.1, 0.05, 0.01, 0.005, 0.001\}$ and momentum coefficient of 0.9. For our method, we choose $\eta_2 = 0.01, \eta_3 = 0.001$ and use SGD without momentum. For MNIST and Fashion MNIST datasets, we train for 150 epochs/comm. rounds and for CIFAR-10 for 250 epochs. We initialize $\sigma = 1$ in MNIST and Fashion MNIST experiments, and $\sigma = 0.4$ in synthetic experiments, and do not update σ for the first two epochs. For CIFAR-10 we initialize $\sigma = 0.2$ and we do lazy updates that is we start updates after 200 epochs. We observed lazy updates with relatively small initial σ works better for deeper models, whereas simpler models do not require it. For pFedMe, we set the number of local optimization steps to 3 and use a $\lambda =$

For AE experiments we use simpler models (compared to Diffusion Model experiments) as AEs tend to overfit the data and deeper models do not necessarily provide a better performance in general. For synthetic experiments, we use a two-layer fully connected AE, and for MNIST and Fashion MNIST we also use a two-layer AE with 784 input dimension

with 10 and 20 latent dimensions depending on the experiment. For CIFAR-10 we use a symmetric convolutional AE whose input and output layers are convolutional layers with 16 channels, 3 kernel size, 2 strides, and no padding; the intermediate layers are fully connected layers that map 3600 dimensions to latent dimensions. We use 10 and 50 latent dimensions depending on the experiment. We use ReLU activation function after the first layer and sigmoid after the last layer. To improve the empirical performance and have a more stable training we make a few changes to algorithm 5. Namely, we keep individual σ for each scalar weight, for personalized and global models we clip the ℓ_∞ norm of the gradients by 1, and for σ by 10. We update σ at the first iteration instead of the last one. We also update the global model locally in each local iteration.

ADEPT-DGM. On MNIST and Fashion MNIST, we use 20 local iterations per communication round and epoch during training. We use Adam optimizer with $1e{-}3$ for all methods. For our method, we use Adam with 0.01 learning rate for the updates of the global model and SGD with 0.001 lr for σ. We do 100 epochs/comm. rounds in total. We initialize $\sigma = 0.8$ and do not update it for the first 2 epochs. We multiply the learning rates η_1, η_2 by 0.1 at the 75th epoch. We employ the same changes in algorithm 5 and algorithm 16 as well. For demonstration, we use a variance preserving SDE (as in algorithm 16) instead of variance exploding (as in section 4.4). On MNIST and Fashion MNIST, the models are trained to predict images instead of noise. On MNIST we use a cosine noise scheduler and on CIFAR-10 we use a linear noise scheduler. On CIFAR-10, for the results in table 4.3, we train the model using 50 local iterations per communication round. We use Adam optimizer with $1e{-}4$ for all methods, and we use Adam with $1e{-}4$ learning rate for the updates of the global model and SGD with $1e{-}4$ lr for σ and we initialize $\sigma = 0.35$. We train for 200 comm. rounds. For the results in table 4.3 we train for 100 comm. rounds in table C.3 and let each client access samples from 3 classes.

C.6.3 Additional Experiments

Convergence plots In both figs. C.2 and C.3 we observe that our theoretical findings from section 4.3 hold; that is, high variance imply faster convergence albeit it can result in inferior testing performance.

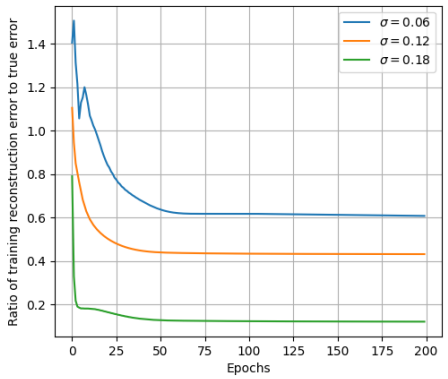


Figure C.2 Training loss vs epochs for `ADEPT-PCA` with a different true standard deviation of data.

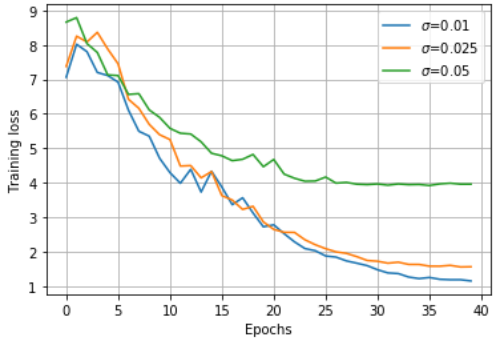


Figure C.3 Training loss vs epochs for `ADEPT-AE` with a different true standard deviation of data.

Sensitivity to ξ

As seen in table C.1, the sensitivity to ξ is quite low in our experiments.

Additional ADEPT-DGM results

In table C.2, compared to the results in table 4.3, we observe similar trends across the performance comparison of methods.

Table C.1 Energy captured (%) versus hyper-prior parameter for the setting of F. MNIST experiments with high latent dimensionality.

Value of ξ	KID
$1e{-}5$	85.6
$1e{-}6$	85.6
$1e{-}7$	85.5
$1e{-}8$	85.7
$1e{-}9$	85.8
$1e{-}10$	85.9

For CIFAR-10 when we have 50 comm. rounds and samples from 3 classes per client; we observe a significant performance difference between our method and competition; indicating faster convergence of our method in terms of image quality.

For F. MNIST, we look at a setting where each client has 200 samples from 2 classes; despite the low number of samples per client, our method is able to outperform other methods. We note that on F. MNIST, it is particularly hard to outperform local training due to less diverse training and test samples.

Table C.3 Diffusion model generation quality for generating CIFAR-10 samples (lower is better).

Method	KID
Baseline	0.062 ± 0.003
`ADEPT-DGM`	$\mathbf{0.067} \pm 0.003$
FedAvg+fine-tuning	0.082 ± 0.004
Local Training	0.075 ± 0.001

Method	KID
`ADEPT-DGM`	$\mathbf{0.086} \pm 0.003$
FedAvg+fine-tuning	0.104 ± 0.005
Local Training	0.111 ± 0.006

Table C.4 Diffusion model generation quality for generating F. MNIST samples (lower is better). 200 samples and 2 classes accessed per client, $m = 30$.

Method	KID
`ADEPT-DGM`	$\mathbf{0.110} \pm 0.008$
FedAvg+fine-tuning	0.118 ± 0.010
Local Training	0.119 ± 0.005

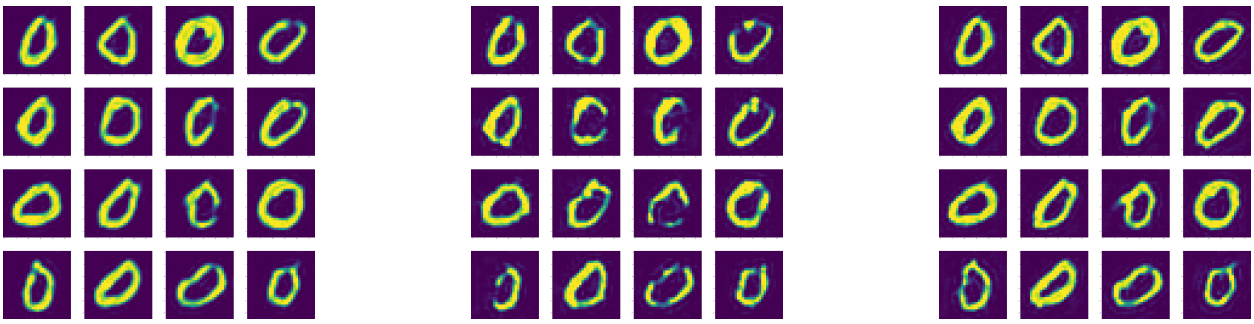


Figure C.4 Randomly chosen samples (Left:`ADEPT-DGM`, noise $\sigma = 0.024$; Middle:FedAvg+fine-tuning, noise $\sigma = 0.028$; Right: Local training, noise $\sigma = 0.032$) (models are trained and samples are chosen with the same seed across runs) from generated dataset for a client with data from '0' class. The pictures are also included in the supplementary material.

Algorithm 15 `ADEPT-PCA` Algorithm

Input: Number of iterations T and learning rates (η_1, η_2, η_3).

Initialize local PCs $\{\boldsymbol{U}_{i,0}\}_{i=1}^m$, global PC $\boldsymbol{U}_0$, and σ_0.
Broadcast $\boldsymbol{U}_0, \sigma_0$ to the clients
for $t = 1$ **to** T **do**
  On the clients:
  for $i = 1$ **to** m: **do**
    Receive $\boldsymbol{U}_{t-1}, \sigma_{t-1}$
    $\sigma_{i,t} = \sigma_{t-1} - \eta_3 \frac{\partial}{\partial \sigma_{t-1}} f_i^{\text{pca}}(\boldsymbol{U}_{i,t-1}, \boldsymbol{U}_{t-1}, \sigma_{t-1})$
    $\mathbf{g}_{i,t}^{\boldsymbol{U}} = P_{\mathcal{T}_{\boldsymbol{U}_{i,t-1}}}(\nabla_{\boldsymbol{U}_{i,t-1}} f_i^{\text{pca}}(\boldsymbol{U}_{i,t-1}, \boldsymbol{U}_{t-1}, \sigma_{t-1}))$
    $\boldsymbol{U}_{i,t} \leftarrow \mathcal{R}_{\boldsymbol{U}_{i,t-1}}(-\eta_1 \mathbf{g}_{i,t}^{\boldsymbol{U}})$
    $\mathbf{g}_{i,t}^{\boldsymbol{U}} = P_{\mathcal{T}_{\boldsymbol{U}_{t-1}}}(\nabla_{\boldsymbol{U}_{t-1}} f_i^{\text{pca}}(\boldsymbol{U}_{i,t}, \boldsymbol{U}_{t-1}, \sigma_{t-1}))$
    $\boldsymbol{U}_{i,t} \leftarrow \boldsymbol{U}_{t-1} - \eta_2 \mathbf{g}_{i,t}^{\boldsymbol{U}}$
    Send $\boldsymbol{U}_{i,t}$, $\sigma_{i,t}$ to the server
  end for
  At the server:
  Receive $\{\boldsymbol{U}_{i,t}\}_{i=1}^m$ and $\{\sigma_{i,t}\}_{i=1}^m$
  $\boldsymbol{U}_t = \mathcal{R}_{\boldsymbol{U}_{t-1}}\left(\frac{1}{m}\sum_{i=1}^m \boldsymbol{U}_{i,t} - \boldsymbol{U}_{t-1}\right)$
  $\sigma_t = \frac{1}{m}\sum_{i=1}^m \sigma_{i,t}$
  Broadcast $\boldsymbol{U}_t, \sigma_t$ to the clients
end for

Output: Personalized PCs $\{\boldsymbol{U}_{1,T}, \ldots, \boldsymbol{U}_{m,T}\}$.

Algorithm 16 Personalized Adaptive Diffusion Model: `ADEPT-DGM`

Input: Number of iterations T, learning rates (η_2, η_1, η_3), number of local iterations τ, sample corruption range $\gamma \in \mathbb{Z}^+$

Init local models $\{\boldsymbol{\theta}_{i,0}\}_{i=1}^m$, global model $\boldsymbol{\mu}_0$, and σ_0.
On server:
Broadcast $\boldsymbol{\mu}_0$, σ_0 to all clients
for $t = 1$ **to** T **do**
  On Clients:
  for $i = 1$ **to** m **do**
    if τ divides $t-1$ **then**
      Receive $\boldsymbol{\mu}_{t-1}$, σ_{t-1}
    end if
    Sample noise amount $\alpha_i \in$ Uniform$[1, \ldots, \gamma]$ independently for each sample and construct $\boldsymbol{\alpha} = [\alpha_1, \ldots, \alpha_n]$
    $\boldsymbol{\theta}_{i,t} = \boldsymbol{\theta}_{i,t-1} - \eta_1 \nabla_{\boldsymbol{\theta}_{i,t-1}} f_{i,\boldsymbol{\alpha}}^{\mathrm{df}}(\boldsymbol{\theta}_{i,t-1}, \boldsymbol{\mu}_{t-1}, \sigma_{t-1})$
    if τ divides t **then**
      $\boldsymbol{\mu}_{i,t} = \boldsymbol{\mu}_{t-1} - \eta_2 \nabla_{\boldsymbol{\mu}_{t-1}} f_{i,\boldsymbol{\alpha}}^{\mathrm{df}}(\boldsymbol{\theta}_{i,t}, \boldsymbol{\mu}_{t-1}, \sigma_{t-1})$
      $\sigma_{i,t} = \sigma_{t-1} - \eta_3 \frac{\partial}{\partial \sigma_{t-1}} f_{i,\boldsymbol{\alpha}}^{\mathrm{df}}(\boldsymbol{\theta}_{i,t}, \boldsymbol{\mu}_{t-1}, \sigma_{t-1})$
      Send $\boldsymbol{\mu}_{i,t}, \sigma_{i,t}$ to server
    else
      $\boldsymbol{\mu}_t = \boldsymbol{\mu}_{t-1}$, $\sigma_t = \sigma_{t-1}$
    end if
  end for
  At the Server:
  if τ divides t **then**
    Receive $\{\boldsymbol{\mu}_{i,t}\}_{i=1}^m$ and $\{\sigma_{i,t}\}_{i=1}^m$
    $\boldsymbol{\mu}_t = \frac{1}{m}\sum_{i=1}^m \boldsymbol{\mu}_{i,t}$, $\sigma_t = \frac{1}{m}\sum_{i=1}^m \sigma_{i,t}$
    Broadcast $\boldsymbol{\mu}_t$, σ_t to all clients
  end if
end for

Output: Personalized autoencoders $\{\boldsymbol{\theta}_{1,T}, \ldots, \boldsymbol{\theta}_{m,T}\}$.

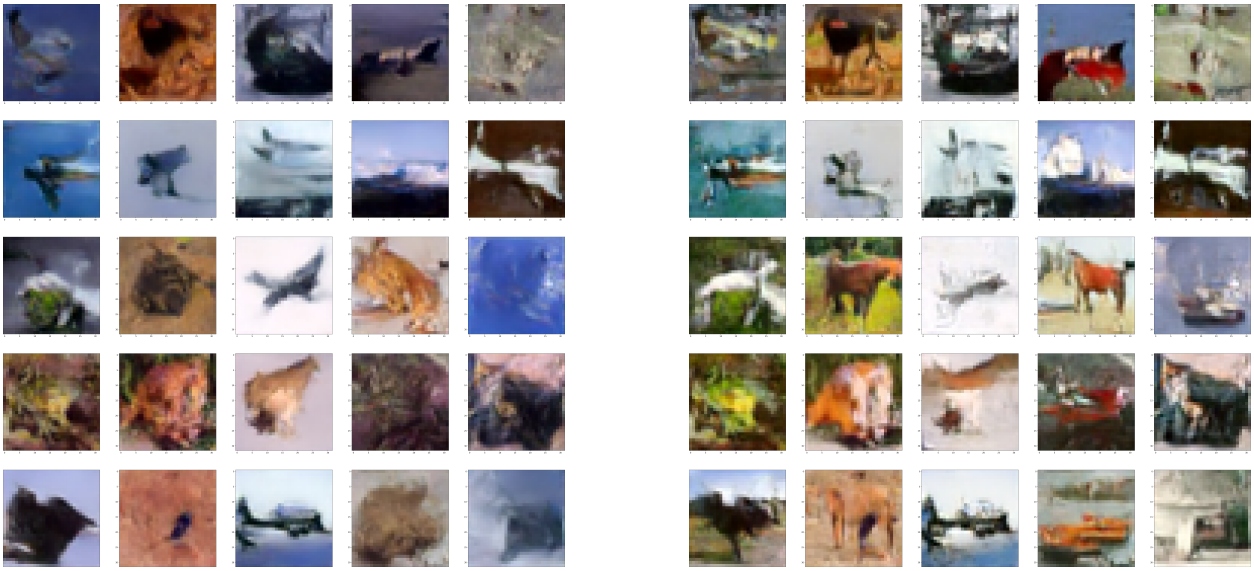


Figure C.5 Randomly chosen samples (Left:`ADEPT-DGM`; Right: FedAvg+fine-tuning (models are trained and samples are chosen with the same seed across runs) from the generated dataset for a client with data from 'frog' and 'airplane' class. We observe that FedAvg+fine-tuning hallucinates images from other classes/clients (e.g. horses); which is both a privacy concern and indicates the lack of performance in generating images from the target distribution.

APPENDIX D

Appendix of Chapter 5

D.1 Additional Related Work

Related Works. Beyond the studies highlighted in Section 5.1, the literature on Personalized Federated Learning (FL) has evolved along several complementary tracks. Meta-learning approaches aim to learn initialization points that quickly adapt to each client's data [1, 48, 87]. Regularization-based methods encourage personalization by adding client-specific penalties to the global objective [40, 65, 114]. Clustered FL partitions clients into groups with similar data distributions [56, 114, 115, 185], while knowledge-distillation techniques transfer information between local and global models [101, 131]. Multi-task formulations treat each client as a related task to optimize jointly [42, 157, 168, 183], and common-representation methods share feature extractors across clients [34, 44, 164]. A hierarchical Bayesian viewpoint has recently inspired new supervised FL algorithms with adaptation capabilities [28, 90, 126].

D.2 Proofs and other details for theoretical results

D.2.1 Diffusion model setup

We first start with the forward process. Ornstein-Uhlenbeck (OU) forward diffusion process describes the gradual transformation from a true data sample to noise through a SDE, it is defined as

$$\mathrm{d}X_t = -\delta_t X_t \,\mathrm{d}t + \sqrt{2\delta_t}\,\mathrm{d}W_t, \quad X_0 \sim q_0(x), \tag{D.1}$$

here $q_0(x)$ denotes the true underlying distribution, similarly $q_t(x)$ will be used to denote a latent distribution, δ_t is a drift coefficient, commonly assumed a constant e.g. 1. W_t is the standard Brownian motion. $X_t \sim q_t$ is a latent variable. Equivalently, it can be shown for some $\beta_t : \lim_{t\to\infty} \beta_t = 1, \beta_0 = 0$,

$$X_t = \sqrt{1-\beta_t^2} X_0 + \beta_t Z_t, \text{with } X_0 \sim q_0 \text{ and } Z_t \sim \mathcal{N}(0, \mathrm{I}).$$

Corresponding to the forward process, is the backward SDE process, which is defined to formalize recovering a data sample from true distribution, given a corrupted sample and score function,

$$\mathrm{d}\overleftarrow{X}_t = \left[\delta_{T-t}\overleftarrow{X}_t + 2\delta_{T-t}\nabla \log q_{T-t}(\overleftarrow{X}_t)\right]\mathrm{d}t + \sqrt{2\delta_{T-t}}\,\mathrm{d}W_t, \quad \overleftarrow{X}_0 \sim q_T(\cdot).$$

In the backward process, we use $\overleftarrow{X}_0$ to denote a sample from pure noise q_T. $\nabla \log q_{T-t}(\overleftarrow{X}_t)$ is the true score function at time $T-t$ (or at time t in terms of the forward process). Of course the true score function is unknown, hence we need to estimate it from the data using a fixed pure noise distribution. To that end, one constructs the score matching optimization criterion,

$$\min_\theta \int_\tau^T \mathbb{E}_{X_t\sim q_t(\cdot)}\left[\|s_\theta(x_t,t) - \nabla \log q_t(X_0)\|^2\right]\,\mathrm{d}t, \tag{D.2}$$

where θ is a neural network that is assumed to parameterize the score function. Equivalent to score matching is the DDPM view, which yields,

$$\min_\theta \int_\tau^T \mathbb{E}_{X_0\sim q_0(\cdot), Z_t}\left[\left\|s_\theta(X_t,t) + \frac{Z_t}{\beta_t}\right\|^2\right]\,\mathrm{d}t. \tag{D.3}$$

EM updates. EM algorithm is the most common approach to find parameters of GMMs. E-step is consists of finding the soft assignments or so called responsibilities, and M-step corresponds to updating the parameters. As can be seen in [92]:

$$\text{(E-step)}: \quad r_i(X) = \frac{w_i \exp(-\|X-\mu_i\|^2/2)}{\sum_{j=1}^k w_j \exp(-\|X-\mu_j\|^2/2)}, \text{ where subscripts denote cluster assignments}$$

$$\text{in our case:} \quad r_1(X) = \frac{w \exp(-\|X-\mu\|^2/2)}{w \exp(-\|X-\mu\|^2/2) + (1-w)\exp(-\|X+\mu\|^2/2)}, r_2(X) = 1 - r_1(X)$$

$$\text{(M-step)}: \quad w^+ = \mathbb{E}_X[r_1(X)], \quad \mu^+ = \mathbb{E}_X[r_1(X)X]/\mathbb{E}_X[r_1(X)] \text{ where } + \text{ denotes the next iteration.}$$

D.2.2 Proofs in Section 7.4

Lemma 19 (Solution of the OU forward SDE)**.** *Let $(W_t)_{t\geq 0}$ be a standard d-dimensional Brownian motion independent of the initialization $X_0 \sim q_0$. Consider the Ornstein–Uhlenbeck (OU) forward (noising) process*

$$dX_t \;=\; -\,X_t\,dt \;+\; \sqrt{2}\,dW_t, \qquad X_0 \sim q_0. \tag{D.4}$$

Then for every $t \geq 0$ the marginal distribution of X_t is

$$X_t \;=\; \mathrm{e}^{-t}\,X_0 \;+\; \sqrt{1-\mathrm{e}^{-2t}}\;Z_t, \qquad Z_t \sim \mathcal{N}\Big(0, \mathrm{I}\Big), \tag{D.5}$$

with Z_t independent of X_0.

Proof. Multiply (D.4) by the integrating factor e^t:

$$\mathrm{e}^t\,dX_t + \mathrm{e}^t\,X_t\,dt \;=\; \sqrt{2}\,\mathrm{e}^t\,dW_t \;\Longrightarrow\; d\Big(\mathrm{e}^t X_t\Big) = \sqrt{2}\,\mathrm{e}^t\,dW_t.$$

Integrating from 0 to t and dividing by e^t yields

$$X_t \;=\; \mathrm{e}^{-t}X_0 \;+\; \sqrt{2}\,\mathrm{e}^{-t}\int_0^t \mathrm{e}^s\,dW_s.$$

Because the Itô integral is Gaussian with mean 0, its covariance is

$$\mathrm{Cov}\left(\sqrt{2}\,\mathrm{e}^{-t}\int_0^t \mathrm{e}^s\,dW_s\right) = 2\,\mathrm{e}^{-2t}\int_0^t \mathrm{e}^{2s}\,ds = 1-\mathrm{e}^{-2t}.$$

Thus the stochastic integral can be written as $\sqrt{1-\mathrm{e}^{-2t}}\;Z_t$ with $Z_t \sim \mathcal{N}(0,\mathrm{I})$, independent of X_0. Substituting back gives (D.5), completing the proof. □

Lemma 20 (Two–component mixture is preserved by the OU forward process)**.** *Let the initial distribution be the symmetric two–component Gaussian mixture*

$$q_0(x) = w\,\mathcal{N}\Big(x;\mu,\mathbf{I}\Big) \;+\; (1-w)\,\mathcal{N}\Big(x;-\mu,\mathbf{I}\Big), \qquad 0<w<1,\;\; \mu\in\mathbb{R}^d.$$

Consider the Ornstein–Uhlenbeck (OU) forward SDE

$$X_t \;=\; e^{-t}X_0 \;+\; \sqrt{1-e^{-2t}}\,Z_t, \qquad \textit{with } Z_t\sim\mathcal{N}(0,\mathbf{I}) \textit{ independent of } X_0.$$

Then, for every $t\ge 0$, the law of X_t remains a two–component mixture

$$q_t(x) = w\,\mathcal{N}\Big(x;\mu_t,\mathbf{I}\Big) \;+\; (1-w)\,\mathcal{N}\Big(x;-\mu_t,\mathbf{I}\Big), \qquad \textit{where } \mu_t \;:=\; e^{-t}\mu.$$

Proof. Condition on the component of the mixture that generated X_0.

Component "$+\mu$". If $X_0\sim\mathcal{N}(\mu,\mathbf{I})$, then

$$X_t \;=\; e^{-t}X_0+\sqrt{1-e^{-2t}}\,Z_t \;\sim\; \mathcal{N}\Big(e^{-t}\mu,\; e^{-2t}\mathbf{I}+(1-e^{-2t})\mathbf{I}\Big) \;=\; \mathcal{N}\Big(e^{-t}\mu,\mathbf{I}\Big).$$

Component "$-\mu$". Analogously, if $X_0\sim\mathcal{N}(-\mu,\mathbf{I})$, we obtain

$$X_t \;\sim\; \mathcal{N}\Big(-e^{-t}\mu,\mathbf{I}\Big).$$

Mixture. Because the two cases occur with probabilities w and $1-w$, the unconditional distribution of X_t is the weighted sum of the two Gaussians derived above, namely

$$q_t(x) = w\,\mathcal{N}\Big(x;e^{-t}\mu,\mathbf{I}\Big) + (1-w)\,\mathcal{N}\Big(x;-e^{-t}\mu,\mathbf{I}\Big),$$

which coincides with the claimed form after defining $\mu_t=e^{-t}\mu$. □

Proof of Lemma 4. Recall that q_t is the symmetric two–component mixture

$$q_t(x) = w\,\mathcal{N}\Big(x;\mu_t,\mathrm{I}\Big) + (1-w)\,\mathcal{N}\Big(x;-\mu_t,\mathrm{I}\Big), \qquad 0<w<1, \quad \mu_t=e^{-t}\mu.$$

Because $q_t(x) = \sum_{k=1}^{2} \pi_k p_k(x)$ with $p_1(x) = \mathcal{N}(x; \mu_t, \mathrm{I})$, $p_2(x) = \mathcal{N}(x; -\mu_t, \mathrm{I})$ and weights $\pi_1 = w,\ \pi_2 = 1 - w$, the score can be written as

$$\nabla_x \log q_t(x) = \frac{\sum_{k=1}^{2} \pi_k \, p_k(x) \, \nabla_x \log p_k(x)}{\sum_{k=1}^{2} \pi_k p_k(x)}.$$

For a Gaussian with unit covariance, $\nabla_x \log \mathcal{N}(x; m, \mathrm{I}) = -(x - m)$. Hence

$$\nabla_x \log q_t(x) = -\Big[r_1(x)\Big(x - \mu_t\Big) + r_2(x)\Big(x + \mu_t\Big)\Big],$$

where

$$r_1(x) \;=\; \frac{w \, p_1(x)}{w \, p_1(x) + (1 - w) p_2(x)}, \qquad r_2(x) = 1 - r_1(x)$$

are the posterior ("belief") probabilities of the two components. A short algebraic simplification gives

$$\nabla_x \log q_t(x) \;=\; \Big(2r_1(x) - 1\Big)\, \mu_t \;-\; x. \tag{D.6}$$

Writing the Gaussian densities explicitly:

$$p_1(x) = \frac{1}{(2\pi)^{d/2}} \exp\Big[-\tfrac{1}{2}\|x - \mu_t\|^2 \Big], \quad p_2(x) = \frac{1}{(2\pi)^{d/2}} \exp\Big[-\tfrac{1}{2}\|x + \mu_t\|^2 \Big].$$

Then

$$\frac{r_1(x)}{1 - r_1(x)} = \frac{w \, p_1(x)}{(1 - w) \, p_2(x)} = \frac{w}{1 - w} \, \exp\Big[-\tfrac{1}{2}(\|x - \mu_t\|^2 - \|x + \mu_t\|^2) \Big].$$

Because $\|x - \mu_t\|^2 - \|x + \mu_t\|^2 = -4 \, \mu_t^\top x$, we obtain

$$\frac{r_1(x)}{1 - r_1(x)} = \exp\Big[2\mu_t^\top x \;+\; \log \frac{w}{1 - w} \Big].$$

Setting $z(x) := 2\mu_t^\top x \;+\; \log \frac{w}{1-w}$ and recalling that $r_1 = \sigma(z)$ with the logistic function $\sigma(z) = 1/(1 + e^{-z})$, we have $2r_1 - 1 = \tanh\Big(\frac{1}{2} z\Big)$. Therefore

$$2r_1(x) - 1 \;=\; \tanh\!\left(\mu_t^\top x + \tfrac{1}{2} \log \frac{w}{1 - w} \right) \;=\; \tanh\!\left(\mu_t^\top x - \tfrac{1}{2} \log \frac{1 - w}{w} \right).$$

Substituting the expression for $2r_1(x) - 1$ back into (D.6) yields

$$\nabla_x \log q_t(x) = \tanh\left(\mu_t^\top x - \tfrac{1}{2}\log\frac{w}{1-w}\right)\mu_t \; - \; x,$$

which is exactly the claimed form in Lemma 4. □

D.2.3 Algorithm for new client fine-tuning

Algorithm 17 Conditional personalized fine-tuning for new clients

Input: Number of iterations K, learning rate (η)

1: **Download** backbone model, initialize the embedding $e_{j,0}$
2: Set $\theta_{j,0} = [\theta_{bb}, e_{j,0}]$
3: **for** $k = 1$ **to** K **do**
4: Sample a batch of samples $\{X_i\}_{i=1}^b$ and diffusion timesteps $\{t_i\}_{i=1}^b$
5: Use forward process (D.2) to obtain a batch of noisy samples $\{X_{t_i}\}_{i=1}^b$
6: Take gradient step for embedding $\boldsymbol{e}_{j,k+1} = \boldsymbol{e}_{j,k} - \nabla_{\boldsymbol{e}_j} L_{ddpm}^{(j)}(\{X_{t_i}\}_{i=1}^b)$
7: **end for**

Algorithm 17 A new client downloads the current backbone (line 1), forms $\boldsymbol{\theta}_{j,0}$ by appending a freshly initialized embedding (line 2), and for K iterations (line 3) repeats: sampling data and timesteps (lines 4–5) and updating *only* its embedding via the DDPM loss (line 6). No further communication is required after the initial download, so personalization remains entirely on-device.

D.2.4 Proofs of Theorems 11, 12

Proof of Theorem 11. We start by taking the derivative of (5.10) with respect to w at a particular timepoint t. Let $u_t = \mu_t^\top X_t - \frac{1}{2}\log(w/(1-w))$, $f(X_t) = \tanh(u_t) - X_t + \frac{Z_t}{\sqrt{1-e^{-2t}}}$, by the chain rule,

$$
\begin{aligned}
\frac{\mathrm{d}L}{\mathrm{d}w} &= \frac{\mathrm{d}L}{\mathrm{d}f(X_t)} \cdot \frac{\mathrm{d}f(X_t)}{\mathrm{d}\tanh(u_t)} \cdot \frac{\mathrm{d}\tanh(u_t)}{\mathrm{d}u_t} \cdot \frac{\mathrm{d}u_t}{\mathrm{d}(\frac{1}{2}\log((1-w)/w))} \cdot \frac{\mathrm{d}(\frac{1}{2}\log((1-w)/w))}{w} \\
&= \mathbb{E}\left[2\mu_t^\top(\tanh(u_t)\mu_t - X_t + Z_t/(\sqrt{1-e^{-2t}})\tanh'(u_t)\frac{-1}{2w(1-w)}\right] \\
&= \frac{-1}{w(1-w)}\mathbb{E}\left[\mu_t^\top(\tanh(u_t)\mu_t - X_t + Z_t/\sqrt{1-e^{-2t}}) \cdot \tanh'(u_t)\right] \\
&\frac{-1}{w(1-w)}\mathbb{E}\left[\mu_t^\top(\tanh(u_t)\mu_t - X_t + Z_t/\sqrt{1-e^{-2t}}) \cdot \tanh'(u_t)\right] \\
&\frac{-1}{w(1-w)}\mathbb{E}\left[\|\mu_t\|^2\tanh(u_t)\tanh'(u_t) - \mu_t^\top X_t\tanh'(u_t) + \mu_t^\top Z_t/\sqrt{1-e^{-2t}}\tanh'(u_t)\right] \\
&\overset{(a)}{=} \frac{-1}{w(1-w)}\mathbb{E}\left[\|\mu_t\|^2\tanh(u_t)\tanh'(u_t) - \mu_t^\top X_t\tanh'(u_t) + \|\mu_t\|^2\tanh''(u_t)\right] \\
&\overset{(b)}{=} \frac{-1}{w(1-w)}\mathbb{E}\left[-\|\mu_t\|^2\tanh(u_t)\tanh'(u_t) - \mu_t^\top X_t\tanh'(u_t)\right] \\
&= \frac{1}{w(1-w)}\mathbb{E}\left[\tanh'(u_t)(\|\mu_t\|^2\tanh(u_t) + \mu_t^\top X_t)\right]
\end{aligned}
$$

Hence, at the critical point we have the following condition being satisfied,

$$
\mathbb{E}[\tanh(u_t)] = -\frac{\mu_t^\top \mathbb{E}\left[X_t\right]}{\|\mu_t\|^2}. \tag{D.7}
$$

Now looking at the EM algorithm and in particular M-step, we see that

$$
w^+ = \mathbb{E}[r_1(X_t)] = \mathbb{E}[\frac{1}{2}(1+\tanh(u_t))]
$$

hence, plugging the convergence point of score loss into EM algorithm,

$$
w^+ = \frac{1}{2}(1 - \frac{\mu_t^\top \mathbb{E}[X_t]}{\|\mu_t\|^2}), \text{ in other words EM algorithm is also converged, equivalently,}
$$

$$
\mathbb{E}[X_t] = \mu_t(2w-1), \text{ which is exactly the first order moment matching condition.}
$$

In other words, convergence of score estimation loss corresponds EM algorithm convergence that satisfies the first order moment equality, since the means and covariances are known this corresponds to the global maximum of the likelihood.

At the convergence point we can measure the mean squared error of estimating the mixing weight from samples. In particular, let us define $\hat{w} = \frac{1}{2}(1 - \frac{\mu_t^\top \hat{X}_t}{\|\mu_t\|^2})$ as the sample mixing weight, where $\hat{X}_t$ is the sample mean at time t. Then the L_2 error can be found to be:

$$
\begin{aligned}
\mathbb{E}\|w-\hat{w}\|^2 &= \frac{1}{4}\|\frac{\mu_t^\top}{\|\mu_t\|^2}(\mathbb{E}[X_t]-\hat{X}_t)\|^2 \\
&\leq \frac{1}{4\|\mu_t\|^2}\|\mathbb{E}[X_t]-\hat{X}_t\|^2 \\
&\leq \frac{1}{4\|\mu_t\|^2}\frac{tr(\Sigma)}{n}, \text{ where } \Sigma \text{ is true overall covariance vector} \\
&= \frac{1}{4\|\mu_t\|^2}\frac{d+4w(1-w)\|\mu_t\|^2}{n} \\
&= \frac{w(1-w)}{n}+\frac{d}{4\|\mu_t\|^2 n},
\end{aligned}
$$

which concludes the proof. □

Proof of Theorem 12. Let us define $u := X_t^\top \mu_t - \frac{1}{2}\log((1-w)/(w))$, and similarly for $\hat{u}$ with estimated parameters, and $b(w) = \frac{1}{2}\log((1-w)/w)$, we also assume there is a constant such that $\|\mu_t\|^2 \leq B$. Then we have,

$$
\begin{aligned}
L_{est}(\hat{\mu},\hat{w}) &= \mathbb{E}_{X_o\sim\mathcal{D}_i,Z_t}\Big\| \tanh(X_t^\top\mu_t - \frac{1}{2}\log((1-w)/(w)))\mu_t \\
&\quad - \tanh(X_t^\top\hat{\mu}_t - \frac{1}{2}\log((1-\hat{w})/(\hat{w})))\hat{\mu}_t\Big\|^2 \\
&= \mathbb{E}\|\tanh(u)\mu_t - \tanh(\hat{u})\mu_t + \tanh(\hat{u})\mu_t - \tanh(\hat{u})\hat{\mu}_t\|^2, \\
&\leq \|\mu_t\|^2(\tanh u - \tanh\hat{u})^2 + |\tanh\hat{u}|\|\mu_t-\hat{\mu}_t\|^2 \\
&\leq \mathbb{E}[\|\mu_t\|^2 2(u-\hat{u})^2 + \|\mu_t-\hat{\mu}_t\|^2] \\
&= \mathbb{E}[\|\mu_t\|^2 2(X_t^\top(\mu_t-\hat{\mu}_t)+b(w)-b(\hat{w}))^2 + \|\mu_t-\hat{\mu}_t\|^2], \\
&\leq 4\mathbb{E}[\|\mu_t\|^2\|X_t\|^2\|\mu_t-\hat{\mu}_t\|^2 + \frac{(w-\hat{w})^2}{2w_{min}^2(1-w_{min})^2} + \|\mu_t-\hat{\mu}_t\|^2] \\
&\leq 4B\mathbb{E}\|X_t\|^2\mathbb{E}\|\hat{\mu}_t-\mu_t\|^2 + \mathbb{E}(w-\hat{w})^2 + \mathbb{E}\|\mu_t-\hat{\mu}_t\|^2 \\
&\leq 4B(d+(4w(1-w)+1)B)\mathbb{E}\|\mu-\hat{\mu}\|^2 + \frac{1}{2w_{min}^2(1-w_{min})^2}\frac{w(1-w)}{n},
\end{aligned}
$$

From [151], at the convergence point of their algorithm, we have that with probability at least 1-$2d\exp(-\frac{n\epsilon^2\beta_t^2}{d^4B^6})$ the convergence is to a ball with $\|\mu_t-\hat{\mu}_t\|^2 = \mathcal{O}(\epsilon)$. Now we convert

this high probability bound to a expectation bound. In particular the non-vanishing ϵ term is introduced in Lemma E.7 of [151]. There we have the following high probability bound for bounding the difference between population and empirical gradient,

$$Pr[\left\|\nabla_{\hat{\mu}_t}\frac{1}{n}\sum_{j=1}^{n}L_t(s_{\hat{\mu}_t}(x_{j,t}))-L_t(s_{\hat{\mu}_t})\right\|>\epsilon]\leq 2d\exp\left(-\frac{n\epsilon^2\beta_t^2}{d^2B^6}\right)$$

equivalently for some constant $C>0$, at the end of convergence of algorithm (where exponential decaying contraction term has vanished), the result in [151] yields,

$$Pr[\|\mu_t-\hat{\mu}_t\|>C\epsilon]\leq 2d\exp\left(-\frac{n\epsilon^2\beta_t^2}{d^4B^6}\right)$$

For conversion, we use the tail definition of expectation, $\mathbb{E}[Y^2]=\int_0^\infty P(Y^2>y)dy=\int_0^\infty P(Y>\sqrt{y})dy$ which yields,

$$\mathbb{E}\|\mu_t-\hat{\mu}_t\|^2=\int_0^\infty 2Cd\exp\left(-\frac{ny\beta_t^2}{d^4B^6}\right)dy\leq C\frac{d^5B^6}{mn\beta_t^2},$$

where we used the fact that $\int_0^\infty\exp(-Cx)dx=\frac{1}{C}$ As a result, the overall bound is,

$$L(\hat{\mu},\hat{w})=\mathcal{O}\left(\frac{d^6B^8}{mn(1-e^{-2t})^2}\right)+\mathcal{O}\left(\frac{1}{n}\right) \tag{D.8}$$

where the first term is due to global estimation of mean and the second term is due to the local estimation of mixing weight.

□

D.3 Additional Details and Results on Experiments

Experiment details. We use Adam optimizer with constant learning rate following the original DDPM paper. For CelebA, CIFAR-10 we use learning rate of $1e-4$, and for

MNIST we use $1e-3$. During training, each round of local training corresponds to 1 epoch. For CelebA, we traing for a total of 800 epochs with 50 local iterations, for CIFAR-10 we train for 400 epochs with 100 local iterations, and for MNIST we train for 200 epochs using 20 local iterations. For fine-tuning, for competing methods $\times 0.1$ of original learning rate gives the best results, for our method we use 0.01 as the learning rate.

Model architecture. For every dataset we employ the same high-level design: a identity-embedding layer maps each identity token $y \in \{0, \ldots, m-1\}$ to a learnable vector of dimension d_{emb}; this vector is injected into an `UNet2DModel` backbone through modulating activations. Each backbone uses two residual blocks per scale, while its depth and channel widths are dataset-specific:

- **MNIST** (28×28, 1 channel): three scales with feature widths $(32, 64, 64)$. The encoder follows `DownBlock2D` $\rightarrow$ `AttnDownBlock2D` $\rightarrow$ `AttnDownBlock2D`; the decoder mirrors this with two `AttnUpBlock2D`s and a final `UpBlock2D`. This lightweight configuration suffices for the low-resolution digit images.
- **CIFAR-10** (32×32, 3 channels): four scales with widths $(64, 128, 128, 128)$. The down path is `DownBlock2D` $\rightarrow$ `AttnDownBlock2D` $\rightarrow$ `AttnDownBlock2D` $\rightarrow$ `DownBlock2D`; the up path is `UpBlock2D` $\rightarrow$ `AttnUpBlock2D` $\rightarrow$ `AttnUpBlock2D` $\rightarrow$ `UpBlock2D`. Both width and depth are doubled relative to MNIST to handle the richer natural-image content.
- **CelebA** (64×64, 3 channels): we reuse the CIFAR backbone.

Across all variants, the parameter count rises from MNIST to CIFAR/CelebA, yet the conditioning interface is unchanged. For our implementation we directly use label conditioning interface of UNet architecture.

CelebA qualitative results. In Figure D.1 we can observe generated images for the modesl trained on CelebA dataset.

Shared representation approach architecture. Inspired by [34], in our framework `SPIRE`, we introduce a novel *shared representation* approach as a strong competing baseline.

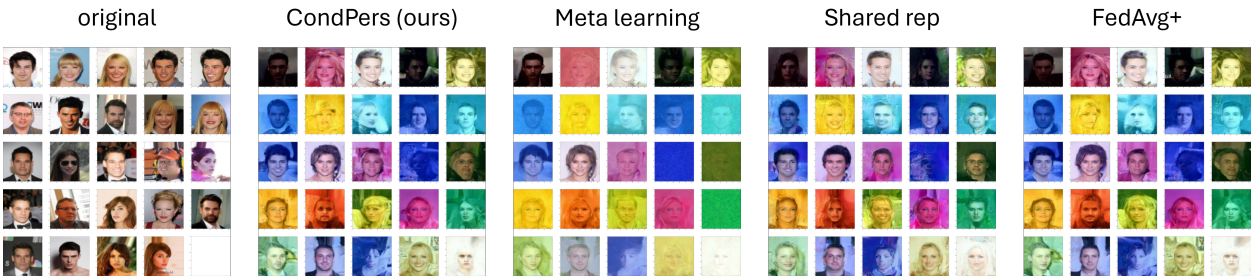


Figure D.1 Grid of original CelebA images and generated images by competing methods for particular client, who has sampled data from 5 different individuals; using the same random seed maybe to appendix.

Concretely, each client's U-Net is partitioned into three semantic parts: (i) the *down-sampling path*, responsible for hierarchical feature extraction; (ii) the *bottleneck (middle) block*, where the highest-level, most compressed latent representation is processed; and (iii) the *up-sampling path*, which decodes latents back into the pixel space. During federated training, the down- and up-sampling paths are synchronized across clients at every communication round in `SPIRE` —mirroring the global feature extractor in [34] so that low and high-level visual features are learned from the entire population's data as can be seen in Figure D.2. In contrast, the bottleneck block is kept *client-specific* and never uploaded to the server. This design has several advantages:

1. **Personalization without classification head swapping.** Unlike classifiers, diffusion U-Nets do not have a lightweight prediction head that can be individualized. By individualizing the bottleneck, we inject client-specific inductive bias at the deepest latent level, where semantic factors are most disentangled, enabling fine-grained customization of generation quality and style.

2. **Better fine-tuning performance.** In `SPIRE`, global synchronization of the encoder and decoder paths anchors each client's individual bottleneck in a common feature space, mitigating overfitting to narrow data distributions and preserving cross-client coherence—a problem that pure fine-tuning baselines (e.g., FedAvg+FT) often suffer from. Note that, as we see in Section 5, conditional personalization within `SPIRE` still

results in better fine-tuning performance.

Training hardware. For training we use a GPU server of 6 NVIDIA RTX2080 GPUs. The longest training is done on CelebA, which lasts around 60 hours.

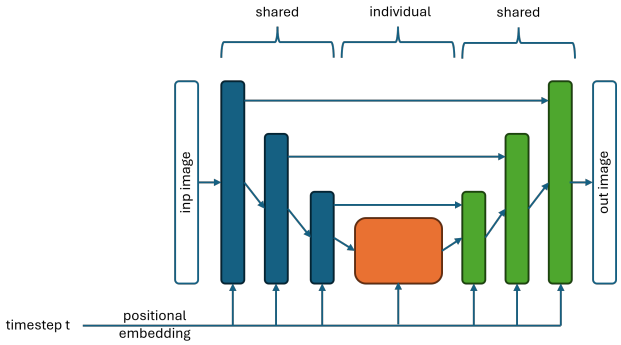


Figure D.2 Shared representation partitioning.

APPENDIX E

Appendix of Chapter 6.6

E.1 Setup, Details, and Proof for Theorem 13

Assumptions. (i) F is bounded below by F^*, (ii) stochastic gradients, that is $\forall x \in \mathbb{R}^d$, $\|\nabla f(x)\|_\infty \leq R$ a.s. , (iii) the objective function is smooth, that is $\forall x, y \in \mathbb{R}^d$, $\|\nabla F(x) - \nabla F(y)\|_2 \leq L\|x - y\|_2$.

Setup. Our setup follows [39]. Let $d \in \mathbb{N}$ be the dimensionality of the model. Given a function $h : \mathbb{R}^d \to \mathbb{R}$ we define $\nabla_i h$ as the i'th component of the gradient. For the stochastic loss, we assume we have access to an oracle providing i.i.d samples $(f_t)_{t\in\mathbb{N}}$. $\mathbb{E}_{t-1}$ is defined as the expectation conditioned upon observing the past stochastic function values: $f_1, \ldots, f_{t-1}$. Unlike [39], for simplicity of calculations, we define $\epsilon_t = \epsilon/t$ where ϵ is a small constant for stability. Next we define adaptive moment updates in a generic way, and then specialize them to AVGrad. These updates are different from those in Section 6.2, 6.4, but they are equivalent, and facilitates easier analysis.

Adaptive updates. Similar to [39] we define the following vectors iteratively.

$$m_{t,i} = \beta_1 m_{t-1,i} + \nabla_i f_t(x_{t-1})$$
$$v_{t,i} = \beta_2 v_{t-1,i} + \nabla_i f_t(x_{t-1})^2$$

Note that $\hat{m}_{t,i} = (1-\beta_1)m_{t,i}$ and $\hat{v}_{t,i} = (1-\beta_2)v_{t,i}$ give the updates in Adam. [39] uses the learning rate to absorb $(1-\beta)$ terms. In particular, if one has an update $x_{t,i} = x_{t-1,i} - \alpha_t \frac{m_{t,i}}{\sqrt{\epsilon + v_{t,i}}}$; Adam is recovered with $\alpha_t = \alpha\frac{1-\beta_1}{\sqrt{1-\beta_2}}$.

AVGrad updates. Let $\alpha_t = \frac{\alpha}{\sqrt{t}}$ be the learning rate. AVGrad is defined via the following iterative steps

$$v_{t,i}^{(sum)} = v_{t-1,i}^{(sum)} + v_{t,i} = \sum_{j=1}^{t} v_{j,i}, \text{ and } v_{t,i}^{(avg)} = \frac{v_{t,i}^{(sum)}}{t}$$

$$x_{t,i} = x_{t-1,i} - \alpha_t \frac{m_{t,i}}{\sqrt{\epsilon_t + v_{t,i}^{(avg)}}} = x_{t-1,i} - \alpha \frac{m_{t,i}}{\sqrt{\epsilon + v_{t,i}^{(sum)}}}.$$

Note that to obtain the AVGrad updates we also need to scale the learning rate with $\frac{1-\beta_1}{\sqrt{1-\beta_2}}$; but, for now, we assume it is already absorbed in the α for the sake of simplicity. We also drop the bias correction for simplicity as justified in [39].

Precise Statement of Theorem 13. Under the assumptions above and $\alpha_t = \frac{\alpha}{\sqrt{t}}$ for some $\alpha > 0$, and a constant $0 \leq \rho \leq 1$; when the second-order normalization term is in the form of the interpolation $\frac{\rho}{t} v_t + (1-\rho) v_t^{(avg)}$ we have:

$$\frac{1}{T}\sum_{t=1}^{T} \|\nabla F(x_t)\|^2 \leq \frac{2R\sqrt{\rho + (1-\rho)T}(F_0 - F^*)}{\alpha T \sqrt{1-\beta_2}} + \frac{2R\sqrt{\rho + (1-\rho)T}\, d}{\sqrt{1-\beta_2} T} (2R + \alpha L) \left(\ln\left(1 + \frac{R^2(\rho + (1-\rho)T)}{\epsilon(1-\beta_2)}\right) + T \left[\ln\left(\frac{\rho}{\beta_2}\right)\right]_+ \right)$$

E.1.1 Proof of Theorem 13

We multiply the normalization term of the update with t from the decaying learning rate; hence, there is a multiplying factor of t difference in definition of interpolated term in Theorem 13 and $\bar{\psi}$ in this proof. Let us start by defining $\bar{\psi}_{t,i}$

$$\bar{\psi}_{t,i} = \rho v_{t,i} + (1-\rho) v_{t,i}^{(sum)} \tag{E.1}$$

We also define $\tilde{\psi}_{t,i}$:

$$\begin{aligned} \tilde{\psi}_{t,i} = \rho \tilde{v}_{t,i} + (1-\rho)\tilde{v}_{t,i}^{(sum)} &= \rho(\beta_2 v_{t-1,i} + \mathbb{E}_{t-1} \nabla_i f_t(x_{t-1})^2) \qquad \text{(E.2)} \\ &\quad + (1-\rho)(v_{t-1,i}^{(sum)} + \beta_2 v_{t-1,i} + \mathbb{E}_{t-1} \nabla_i f_t(x_{t-1})^2) \\ &= (1-\rho) v_{t-1,i}^{(sum)} + \beta_2 v_{t-1,i} + \mathbb{E}_{t-1} \nabla_i f_t(x_{t-1})^2, \qquad \text{(E.3)} \end{aligned}$$

where the contribution of the last gradient is replaced by its expected value conditioned on the past iteration. Let us also define

$$u_{t,i} = \frac{\nabla_i f_t(x_{t-1})}{\sqrt{\epsilon + \overline{\psi}_{t,i}}}.$$

From the smoothness of the objective function F, we can use the Descent Lemma to obtain that

$$F(x_t) \leq F(x_{t-1}) - \alpha \nabla F(x_{t-1})^\top u_t + \frac{\alpha^2 L}{2} \|u_t\|^2.$$

Taking the expectation of both sides conditioned on $f_1, \ldots, f_{t-1}$, we have

$$\mathbb{E}_{t-1} F(x_t) \leq F(x_{t-1}) - \alpha \sum_{i \in [d]} \mathbb{E}_{t-1} \left[\nabla_i F(x_{t-1}) \frac{\nabla_i f_t(x_{t-1})}{\sqrt{\epsilon + \overline{\psi}_{t,i}}} \right] + \frac{\alpha^2 L}{2} + \mathbb{E}_{t-1}[\|u_t\|^2]. \tag{E.4}$$

To upper bound the second term on the right hand side, we use the following lemma which generalizes Lemma 5.1 in [39] to be used with interpolated optimizer and replaces v_t with $\psi_{t,i}$ as the normalization term.

Lemma 21 (Updates approximately follow a descent direction)**.**

$$\mathbb{E}_{t-1} \left[\nabla_i F(x_{t-1}) \frac{\nabla_i f_t(x_{t-1})}{\sqrt{\epsilon + \overline{\psi}_{t,i}}} \right] \geq \frac{\nabla_i F(x_{t-1})^2}{2\sqrt{\epsilon + \tilde{\psi}_{t,i}}} - 2R \mathbb{E}_{t-1} \left[\frac{\nabla_i f_t(x_{t-1})^2}{\epsilon + \psi_{t,i}} \right]. \tag{E.5}$$

Proof. For the sake of simplicity, we define $G = \nabla_i F(x_{t-1}),\ g = \nabla_i f_t(x_{t-1}), \psi = \bar{\psi}_{t,i}, \tilde{\psi} = \tilde{\psi}_{t,i}$. First obviously, we have

$$\frac{Gg}{\sqrt{\epsilon + \psi}} = \frac{Gg}{\sqrt{\epsilon + \tilde{\psi}}} + \underbrace{\frac{Gg}{\sqrt{\epsilon + \psi}} - \frac{Gg}{\sqrt{\epsilon + \tilde{\psi}}}}_{A}. \tag{E.6}$$

For the first term, we use the fact that $\mathbb{E}_{t-1}[g] = G$ to obtain that

$$\mathbb{E}_{t-1} \left[\frac{Gg}{\sqrt{\epsilon + \tilde{\psi}}} \right] = \frac{G^2}{\sqrt{\epsilon + \tilde{\psi}}}. \tag{E.7}$$

In order to bound A, we begin with

$$A = Gg\frac{\tilde{\psi}-\psi}{\sqrt{\epsilon+\psi}\sqrt{\epsilon+\tilde{\psi}}(\sqrt{\epsilon+\psi}+\sqrt{\epsilon+\tilde{\psi}})} = Gg\frac{\mathbb{E}_{t-1}g^2-g^2}{\sqrt{\epsilon+\psi}\sqrt{\epsilon+\tilde{\psi}}(\sqrt{\epsilon+\psi}+\sqrt{\epsilon+\tilde{\psi}})},$$

where the last equality follows from the fact that $\tilde{\psi}-\psi = \mathbb{E}_{t-1}[g^2]-g^2$ (see (E.3) and the definition of $\psi_{t,i}$). Hence, from the triangle inequality we have that

$$|A| \leq \underbrace{|Gg|\frac{\mathbb{E}_{t-1}g^2}{\sqrt{\epsilon+\psi}(\epsilon+\tilde{\psi})}}_{A_1} + \underbrace{|Gg|\frac{g^2}{\sqrt{\epsilon+\tilde{\psi}}(\epsilon+\psi)}}_{A_2},$$

where in the inequality follows from the fact that $\sqrt{\epsilon+\psi}+\sqrt{\epsilon+\tilde{\psi}} \geq \max(\sqrt{\epsilon+\psi}, \sqrt{\epsilon+\tilde{\psi}})$. A useful fact that will be used later is the following

$$\forall \lambda > 0,\ x, y \in \mathbb{R} \quad xy \leq \frac{\lambda x^2}{2} + \frac{y^2}{2\lambda}. \tag{E.8}$$

To bound A_1, we use (F.8) with $\lambda = \frac{\sqrt{\epsilon+\tilde{\psi}}}{2}, x = \frac{|G|}{\sqrt{\epsilon+\tilde{\psi}}}, y = |g|\frac{\mathbb{E}_{t-1}g^2}{2\sqrt{\epsilon+\tilde{\psi}}\sqrt{\epsilon+\psi}}$, which yields that

$$A_1 \leq \frac{G^2}{4\sqrt{\epsilon+\tilde{\psi}}} + \frac{g^2\mathbb{E}_{t-1}[g^2]^2}{(\epsilon+\tilde{\psi})^{3/2}(\epsilon+\psi)}.$$

Taking the expectation and noting $\epsilon+\tilde{\psi} \geq \mathbb{E}_{t-1}[g^2]$ ensures that

$$\mathbb{E}_{t-1}[A_1] \leq \frac{G^2}{4\sqrt{\epsilon+\tilde{\psi}}} + \frac{\mathbb{E}_{t-1}[g^2]}{\sqrt{\epsilon+\tilde{\psi}}}\mathbb{E}_{t-1}\left[\frac{g^2}{\epsilon+\psi}\right] \leq \frac{G^2}{4\sqrt{\epsilon+\tilde{\psi}}} + R\mathbb{E}_{t-1}\left[\frac{g^2}{\epsilon+\psi}\right]$$

where the last inequality uses our boundedness assumption $\sqrt{\mathbb{E}_{t-1}[g^2]} \leq R$ and the fact that $\sqrt{\epsilon+\tilde{\psi}} \geq \sqrt{\mathbb{E}_{t-1}[g^2]}$. Similarly, for A_2, we use (F.8) with $\lambda = \frac{\sqrt{\epsilon+\tilde{\psi}}}{2\mathbb{E}_{t-1}g^2}, x = \frac{|Gg|}{\sqrt{\epsilon+\tilde{\psi}}}, y = \frac{g^2}{\epsilon+\psi}$, which gives us (we used similar bounding arguments)

$$\mathbb{E}_{t-1}[A_2] \leq \frac{G^2}{4\sqrt{\epsilon+\tilde{\psi}}} + R\frac{\mathbb{E}_{t-1}[g^2]}{\epsilon+\psi}.$$

Combining both upper bounds we have,

$$\mathbb{E}_{t-1}[|A|] \leq \frac{G^2}{2\sqrt{\epsilon+\tilde{\psi}}} + 2R\frac{\mathbb{E}_{t-1}[g^2]}{\epsilon+\psi},$$

which can be equivalently written as follows

$$\mathbb{E}_{t-1}[-|A|] \geq -\left(\frac{G^2}{2\sqrt{\epsilon+\tilde{\psi}}} + 2R\frac{\mathbb{E}_{t-1}[g^2]}{\epsilon+\psi}\right),$$

Finally, combining further with (F.7) and substituting into (F.6) gives us,

$$\mathbb{E}_{t-1}\left[\frac{Gg}{\sqrt{\epsilon+\psi}}\right] \geq \frac{G^2}{\sqrt{\epsilon+\tilde{\psi}}} - \left(\frac{G^2}{2\sqrt{\epsilon+\tilde{\psi}}} + 2R\frac{\mathbb{E}_{t-1}[g^2]}{\epsilon+\psi}\right) = \frac{G^2}{2\sqrt{\epsilon+\tilde{\psi}}} - 2R\frac{\mathbb{E}_{t-1}[g^2]}{\epsilon+\psi},$$

which proves the desired result. □

Using Lemma 21 in (F.4) yields that

$$\mathbb{E}_{t-1}[F(x_t)] \leq F(x_{t-1}) - \left(\frac{\alpha\sqrt{1-\beta_2}}{2R\sqrt{\rho+(1-\rho)t}}\|\nabla F(x_{t-1})\|^2 - 2\alpha R\mathbb{E}_{t-1}[\|u_t\|^2]\right) + \frac{\alpha^2 L}{2}\mathbb{E}_{t-1}[\|u_t\|^2].$$

Summing both sides for $t = 1, \ldots, T$, and noting $\sqrt{t} \leq \sqrt{T}$ implies that

$$\mathbb{E}[F(x_T)] \leq F(x_0) - \frac{\alpha\sqrt{1-\beta_2}}{2R\sqrt{\rho+(1-\rho)T}}\sum_{t=0}^{T-1}\|\nabla F(x_t)\|^2 + \left(2\alpha R + \frac{\alpha^2 L}{2}\right)\sum_{t=0}^{T-1}\mathbb{E}[\|u_t\|^2].$$

Equivalently, we can write,

$$\frac{\alpha\sqrt{1-\beta_2}}{2R\sqrt{\rho+(1-\rho)T}}\sum_{t=0}^{T-1}\|\nabla F(x_t)\|^2 \leq F(x_0) - \mathbb{E}[F(x_T)] + \left(2\alpha R + \frac{\alpha^2 L}{2}\right)\sum_{t=0}^{T-1}\mathbb{E}[\|u_t\|^2]. \tag{E.9}$$

Now, we would like to bound the last term on the right hand side. For this, we introduce the following lemma that generalizes Lemma 5.2 in [39].

Lemma 22. *Define* $b_t = \rho\sum_{j=1}^{t}\beta_2^{t-j}a_j + (1-\rho)\sum_{\tau=1}^{t}\sum_{j=1}^{\tau}\beta_2^{\tau-j}a_j$ *for* $t > 0$ *and* $b_0 = 0$, *we have*

$$\sum_{t=1}^{T}\frac{a_t}{\epsilon+b_t} \leq \ln\left(1 + \frac{R^2(\rho+(1-\rho)T)}{\epsilon(1-\beta_2)}\right) + T\left[\ln\left(\frac{\rho}{\beta_2}\right)\right]_+.$$

Proof. Let $l_t = \sum_{j=1}^{t} \beta_2^{t-j} a_j$, $r_t = \sum_{\tau=1}^{t} \sum_{j=1}^{\tau} \beta_2^{\tau-j} a_j$, we have $b_t = \rho l_t + (1-\rho) r_t$. Since $b_t > a_t \geq 0$ we have that $1 - z \leq \exp^{-z}$ with $z = \frac{a_t}{\epsilon + b_t} < 1$. Hence,

$$
\begin{aligned}
\frac{a_t}{\epsilon + b_t} &\leq \ln(\epsilon + b_t) - \ln(\epsilon + b_t - a_t) \\
&= \ln(\epsilon + b_t) - \ln(\epsilon + \rho(l_t - a_t) + (1-\rho)(r_t - a_t)) \\
&= \ln(\epsilon + b_t) - \ln\left(\epsilon + \rho\beta_2 l_{t-1} + (1-\rho) r_{t-1} + (1-\rho)(\sum_{j=1}^{t} \beta_2^{t-j} a_j - a_t)\right) \\
&= \ln(\epsilon + b_t) - \ln\left(\epsilon + \rho\beta_2 l_{t-1} + (1-\rho) r_{t-1} + (1-\rho)\beta_2 \underbrace{\sum_{j=1}^{t-1} \beta_2^{t-1-j} a_j}_{l_{t-1}}\right) \\
&= \ln(\epsilon + b_t) - \ln\left(\epsilon + \beta_2 l_{t-1} + (1-\rho) r_{t-1}\right) \\
&= \ln\left(\frac{\epsilon + b_t}{\epsilon + b_{t-1}}\right) + \ln\left(\frac{\epsilon + \rho l_{t-1} + (1-\rho) r_{t-1}}{\epsilon + \beta_2 l_{t-1} + (1-\rho) r_{t-1}}\right) \\
&\leq \ln\left(\frac{\epsilon + b_t}{\epsilon + b_{t-1}}\right) + \ln\left(\max\{1, \frac{\rho}{\beta_2}\}\right) \\
&= \ln\left(\frac{\epsilon + b_t}{\epsilon + b_{t-1}}\right) + \left[\ln\left(\frac{\rho}{\beta_2}\right)\right]_+,
\end{aligned}
$$

where the first equality is due to definition of b_t. Summing the inequality above for $t = 1, 2, \ldots, T$, yields that (recall that $b_0 = 0$)

$$
\sum_{t=1}^{T} \frac{a_t}{\epsilon + b_t} \leq \ln\left(1 + \frac{b_T}{\epsilon}\right) + T\left[\ln\left(\frac{\rho}{\beta_2}\right)\right]_+
$$

Also note that $b_T \leq \frac{R^2(\rho + (1-\rho)T)}{1-\beta_2}$. □

Using Lemma 22 in (F.13), dividing both sides by T, and after some algebra we have

$$
\begin{aligned}
\frac{1}{T}\sum_{t=1}^{T} \|\nabla F(x_t)\|^2 \leq{}& \frac{2R\sqrt{\rho + (1-\rho)T}(F_0 - F^*)}{\alpha T\sqrt{1-\beta_2}} \\
&+ \frac{2R\sqrt{\rho + (1-\rho)T}d}{\sqrt{1-\beta_2 T}}(2R + \alpha L)\left(\ln\left(1 + \frac{R^2(\rho + (1-\rho)T)}{\epsilon(1-\beta_2)}\right) + T\left[\ln\left(\frac{\rho}{\beta_2}\right)\right]_+\right)
\end{aligned}
$$

which concludes the proof.

E.2 Convergence of AVGrad

Using the notation and assumptions in Section 6.3, we give the following convergence results for AVGrad.

Theorem 21 (Convergence of AVGrad without momentum)**.** *Under the above assumptions and* $\alpha_t = \frac{\alpha}{\sqrt{t}}$ *for some* $\alpha > 0$*, we have:*

$$\frac{1}{T}\sum_{t=1}^{T}\|\nabla F(x_t)\|^2 \leq \frac{2R(F_0 - F^*)}{\alpha\sqrt{T}\sqrt{1-\beta_2}} + \frac{2Rd}{\sqrt{T}\sqrt{1-\beta_2}}(2R + \alpha L)\ln\left(1 + \frac{R^2T}{(1-\beta_2)\epsilon}\right).$$

Theorem 22 (Convergence of AVGrad with momentum)**.** *Let* $\tau_T \in \{0, \ldots, T-1\}$ *denote a random index such that* $\forall j \in \mathbb{N}, j < T, \mathbb{P}[\tau = j] \propto 1 - \beta_1^{T-j}$*. Under the above assumptions and* $\alpha_t = \frac{\alpha}{\sqrt{t}}$ *for some* $\alpha > 0$*, we have:*

$$\mathbb{E}\|\nabla F(x_\tau)\|^2 \leq \frac{2(1-\beta_1)R\sqrt{T}}{\alpha\sqrt{1-\beta_2}\tilde{T}}(F(x_0) - F^*) + C\frac{\sqrt{T}d}{\tilde{T}}\ln\left(1 + \frac{R^2T}{(1-\beta_2)\epsilon}\right),$$

where $C = \frac{\alpha RL}{\sqrt{1-\beta_2}(1-\beta_1)} + \frac{2\beta_1\alpha^2L^2}{(1-\beta_2)(1-\beta_1)^3} + \frac{12R^2}{\sqrt{1-\beta_1}}$ *and* $\tilde{T} = T - \frac{\beta_1}{1-\beta_1}$.

Remark. Comparing our Theorems 21 and 22 to Theorems 3 and 4 in [39], we observe that AVGrad does not have the non-vanishing, constant term that Adam has (see [39], Theorem 2). Moreover, unlike the result for Adam, we do not require $\beta_2 > \beta_1$. Analogous to how momentum slows down the convergence bound of algorithms (by multiplicative factors) [39] AVGrad slows down the convergence of Adagrad by multiplicative factors of $(1 - \beta_2)$.

E.2.1 Proof of Theorem 21

Proof of convergence of AVGrad can be obtained by replacing the interpolated second order moment term $\bar{\psi}_t$, by $v_t^{(sum)}$ in the previous section. In other words, setting $\rho = 0$ in the Proof of Theorem 13, going through the analysis will give the desired result

$$\frac{1}{T}\sum_{t=1}^{T}\|\nabla F(x_t)\|^2 \leq \frac{2R(F_0 - F^*)}{\alpha\sqrt{T}\sqrt{1-\beta_2}} + \frac{2Rd}{\sqrt{1-\beta_2}\sqrt{T}}(2R + \alpha L)\ln\left(1 + \frac{TR^2}{(1-\beta_2)\epsilon}\right).$$

E.2.2 Proof of Theorem 22

The idea in this proof is to essentially change gradient terms in the Descent Lemma with first order moments, as the difference in consequent model weights will now depend on the moment term rather than a gradient at some time point. The necessary changes in the lemmas closely follow the proof for Theorem 3,4 in [39]. We start by redefining some iterative vectors.

$$
\begin{aligned}
m_{t,i} &= \beta_1 m_{t-1,i} + \nabla_i f_t(x_{t-1}) \\
v_{t,i} &= \beta_2 v_{t-1,i} + \nabla_i f_t(x_{t-1})^2 \\
x_{t,i} &= x_{t-1,i} - \alpha_t \frac{m_{t,i}}{\sqrt{\epsilon_t + v_{t,i}^{(avg)}}} = x_{t-1,i} - \alpha \frac{m_{t,i}}{\sqrt{\epsilon + v_{t,i}^{(sum)}}}
\end{aligned}
$$

Let us further define $G_t = \nabla F(x_{t-1}), g_t = \nabla f_t(x_{t-1}), u_{t,i} = \frac{m_{t,i}}{\sqrt{\epsilon + v_{t,i}^{(sum)}}}$ and $U_{t,i} = \frac{g_{t,i}}{\sqrt{\epsilon + v_{t,i}^{(sum)}}}$. And also define:

$$
\tilde{v}_{t,k,i}^{(sum)} = v_{t-k,i}^{(sum)} + \mathbb{E}_{t-k-1}[\sum_{\tau=t-k+1}^{t} \sum_{j=1}^{\tau} \beta_2^{\tau-j} g_{j,i}^2]
$$

note that for $j \leq t-k$ we have $\mathbb{E}_{t-k-1}[g_j^2] = g_j^2$, so $\tilde{v}_{t,k,i}^{(sum)}$ essentially replaces the contribution of last k gradients with their expected values. From the smoothness of the objective function F, we have

$$
F(x_t) \leq F(x_{t-1}) - \alpha G_t^\top u_t + \frac{\alpha^2 L}{2} \|u_t\|_2^2.
$$

Taking the expectations of both sides,

$$
\mathbb{E}[F(x_t)] \leq \mathbb{E}[F(x_{t-1})] - \alpha \sum_{i \in [d]} \mathbb{E}[G_{t,i}^\top u_{t,i}] + \frac{\alpha^2 L}{2} \mathbb{E}[\|u_t\|_2^2]. \tag{E.10}
$$

To bound the second term on the right hand side, we introduce the following approximate descent lemma, whose proof is provided at the end of section.

Lemma 23. *[Updates approximately follow a descent direction]*

$$\mathbb{E}\left[G_{t,i}\frac{m_{t,i}}{\sqrt{\epsilon+v_{t,i}^{(sum)}}}\right] \geq \frac{1}{2}\left(\sum_{i\in[d]}\sum_{k=0}^{t-1}\beta_1^k\mathbb{E}\left[\frac{G_{t-k,i}^2}{\sqrt{\epsilon+\tilde{v}_{t,k+1,i}^{(sum)}}}\right]\right) - \frac{3R\sqrt{1-\beta_2}}{\sqrt{1-\beta_1}}\sum_{k=0}^{t-1}\sqrt{k+1}\beta_1^k\mathbb{E}[\|U_{t-k}\|^2] - \frac{\alpha^2L^2}{4R\sqrt{1-\beta_2}}\sqrt{1-\beta_1}\sum_{l=1}^{t-1}\|u_{t-l}\|^2\sum_{k=l}^{t-1}\beta_1^k\sqrt{k}.$$

Using Lemma 23 in (E.10) for the second term on right hand side, yields that

$$\mathbb{E}[F(x_t)] \leq \mathbb{E}[F(x_{t-1})] - \frac{\alpha}{2}\left(\sum_{i\in[d]}\sum_{k=0}^{t-1}\beta_1^k\mathbb{E}\left[\frac{G_{t-k,i}^2}{\sqrt{\epsilon+\tilde{v}_{t,k+1,i}^{(sum)}}}\right]\right) \tag{E.11}$$

$$+\frac{3\alpha R\sqrt{1-\beta_2}}{\sqrt{1-\beta_1}}\sum_{k=0}^{t-1}\sqrt{k+1}\beta_1^k\mathbb{E}[\|U_{t-k}\|_2^2] + \frac{\alpha^3L^2}{4R\sqrt{1-\beta_2}}\sqrt{1-\beta_1}\sum_{l=1}^{t-1}\|u_{t-l}\|_2^2\sum_{k=l}^{t-1}\beta_1^k\sqrt{k} + \frac{\alpha^2L}{2}\mathbb{E}[\|u_t\|_2^2]. \tag{E.12}$$

Let us define $\Omega_t := \sqrt{\sum_{\tau=1}^{t}\sum_{j=1}^{\tau}\beta_2^{\tau-j}}$, hence, from our boundedness assumption, $\sqrt{\epsilon+\tilde{v}_{t,k+1,i}^{(sum)}} \leq R\sqrt{\sum_{\tau=1}^{t}\sum_{j=1}^{\tau}\beta_2^{\tau-j}} = R\Omega_t$, inserting in (E.12) implies that

$$\mathbb{E}[F(x_t)] \leq \mathbb{E}[F(x_{t-1})] - \frac{\alpha}{2R\Omega_t}\left(\sum_{k=0}^{t-1}\beta_1^k\mathbb{E}\left[G_{t-k,i}^2\right]\right) + \frac{\alpha^2L}{2}\mathbb{E}\|u_t\|_2^2 + \frac{3\alpha R\sqrt{1-\beta_2}}{\sqrt{1-\beta_1}}\sum_{k=0}^{t-1}\sqrt{k+1}\beta_1^k\mathbb{E}\|U_{t-k}\|_2^2 + \frac{\alpha^3L^2}{4R\sqrt{1-\beta_2}}\sqrt{1-\beta_1}\sum_{l=1}^{t-1}\|u_{t-l}\|_2^2\sum_{k=l}^{t-1}\beta_1^k\sqrt{k}$$

Summing for $t = 1, \ldots, T$ results in

$$\underbrace{\frac{\alpha}{2R}\sum_{t=1}^{T}\frac{1}{\Omega_t}\sum_{k=0}^{t-1}\beta_1^k\mathbb{E}\|G_{t-k}^2\|_2^2}_{A} \leq F(x_0) - F^* + \underbrace{\frac{\alpha^2L}{2}\sum_{t=1}^{T}\mathbb{E}[\|u_t\|^2]}_{B} \tag{E.13}$$

$$+\underbrace{\frac{\alpha^3 L^2}{4R\sqrt{1-\beta_2}}\sqrt{1-\beta_1}\sum_{t=1}^{T}\sum_{l=1}^{t-1}\mathbb{E}[\|u_{t-l}\|^2]\sum_{k=l}^{t-1}\beta_1^k\sqrt{k}}_{C}$$

$$+\underbrace{\frac{3\alpha R\sqrt{1-\beta_2}}{\sqrt{1-\beta_1}}\sum_{t=1}^{T}\sum_{k=0}^{t-1}\sqrt{k+1}\beta_1^k\mathbb{E}[\|U_{t-k}\|^2]}_{D}. \tag{E.14}$$

We will examine each term separately, for B, we state the following lemma (whose proof is given at the end of section) to bound the sum of squared norm term

Lemma 24. *Assume,* $0 \leq \beta_1 < 1$, $0 \leq \beta_2 < 1$ *and we have sequence of real numbers* $(a_t)_{t\in[T]}$. *Let* $c_t = \sum_{\tau=1}^{t}\beta_1^{t-\tau}a_t$, $b_t = \sum_{\tau=1}^{t}\sum_{j=1}^{\tau}\beta_2^{\tau-j}a_t^2$. *Then,*

$$\sum_{t=1}^{T}\frac{c_t^2}{\epsilon+b_t} \leq \frac{1}{(1-\beta_1)^2}\ln\left(1+\frac{b_T}{\epsilon}\right).$$

Lemma 24 implies that

$$\frac{\alpha^2 L}{2}\sum_{t=1}^{T}\mathbb{E}[\|u_t\|^2] \leq \frac{\alpha^2 L}{2(1-\beta_1)^2}\sum_{i\in[d]}\ln\left(1+\frac{v_{T,i}^{(sum)}}{\epsilon}\right) \leq \frac{\alpha^2 L}{2(1-\beta_1)^2}\sum_{i\in[d]}\ln\left(1+\frac{R^2T}{(1-\beta_2)\epsilon}\right). \tag{E.15}$$

Before moving on with other terms, we state some useful facts that are proven in [39].

Fact 10. *Given* $0 < a < 1$ *and* $Q \in \mathbb{N}$ *we have,*

$$\sum_{q=0}^{Q-1}a^q q \leq \frac{a}{(1-a)^2}.$$

Fact 11. *Given* $0 < a < 1$ *and* $Q \in \mathbb{N}$ *we have,*

$$\sum_{q=0}^{Q-1}a^q\sqrt{q} \leq \frac{2}{(1-a)^{3/2}}.$$

Fact 12. *Given* $0 < a < 1$ *and* $Q \in \mathbb{N}$ *we have,*

$$\sum_{q=0}^{Q-1}a^q\sqrt{q}(q+1) \leq \frac{4a}{(1-a)^{5/2}}.$$

Now we move onto examining other terms in (E.14). For C, we make the following change in the index $j = t - l$ which yields the following steps

$$\frac{\alpha^3 L^2}{4R\sqrt{1-\beta_2}}\sqrt{1-\beta_1}\sum_{t=1}^{T}\sum_{l=1}^{t-1}\mathbb{E}[\|u_{t-l}\|^2]\sum_{k=l}^{t-1}\beta_1^k\sqrt{k} \tag{E.16}$$

$$= \frac{\alpha^3 L^2}{4R\sqrt{1-\beta_2}}\sqrt{1-\beta_1}\sum_{t=1}^{T}\sum_{j=1}^{t}\mathbb{E}[\|u_j\|^2]\sum_{k=t-j}^{t-1}\beta_1^k\sqrt{k}$$

$$= \frac{\alpha^3 L^2}{4R\sqrt{1-\beta_2}}\sqrt{1-\beta_1}\sum_{j=1}^{T}\mathbb{E}[\|u_j\|^2]\sum_{t=j}^{T}\sum_{k=t-j}^{t-1}\beta_1^k\sqrt{k}$$

$$= \frac{\alpha^3 L^2}{4R\sqrt{1-\beta_2}}\sqrt{1-\beta_1}\sum_{j=1}^{T}\mathbb{E}[\|u_j\|^2]\sum_{k=0}^{T-1}\beta_1^k\sqrt{k}\sum_{t=j}^{j+k}1$$

$$= \frac{\alpha^3 L^2}{4R\sqrt{1-\beta_2}}\sqrt{1-\beta_1}\sum_{j=1}^{T}\mathbb{E}[\|u_j\|^2]\sum_{k=0}^{T-1}\beta_1^k\sqrt{k}(k+1)$$

$$\overset{(a)}{\leq} \frac{\alpha^3 L^2}{R\sqrt{1-\beta_2}}\sum_{j=1}^{T}\mathbb{E}[\|u_j\|^2]\frac{\beta_1}{(1-\beta_1)^2}$$

$$\overset{(b)}{\leq} \frac{\alpha^3 L^2}{R\sqrt{1-\beta_2}}\frac{\beta_1}{(1-\beta_1)^4}\sum_{i\in[d]}\ln\left(1+\frac{v_{T,i}^{(sum)}}{\epsilon}\right) \tag{E.17}$$

$$\overset{(c)}{\leq} \frac{\alpha^3 L^2}{R\sqrt{1-\beta_2}}\frac{\beta_1}{(1-\beta_1)^4}\sum_{i\in[d]}\ln\left(1+\frac{R^2T}{(1-\beta_2)\epsilon}\right), \tag{E.18}$$

where in (a) we use Fact 12, in (b) we use Lemma 24, and in (c) the definition of $v_{T,i}^{(sum)}$. For D, we make the following change in the index $j = t - k$, and obtain that

$$\frac{3\alpha R\sqrt{1-\beta_2}}{\sqrt{1-\beta_1}}\sum_{t=1}^{T}\sum_{k=0}^{t-1}\sqrt{k+1}\beta_1^k\mathbb{E}[\|U_{t-k}\|^2] = \frac{3\alpha R\sqrt{1-\beta_2}}{\sqrt{1-\beta_1}}\sum_{t=1}^{T}\sum_{j=1}^{t}\sqrt{1+t-j}\beta_1^{t-j}\mathbb{E}[\|U_j\|^2]$$

$$= \frac{3\alpha R\sqrt{1-\beta_2}}{\sqrt{1-\beta_1}}\sum_{j=1}^{T}\mathbb{E}[\|U_j\|^2]\sum_{t=j}^{T}\sqrt{1+t-j}\beta_1^{t-j}$$

$$\overset{(a)}{\leq} \frac{3\alpha R\sqrt{1-\beta_2}}{\sqrt{1-\beta_1}}\sum_{j=1}^{T}\mathbb{E}[\|U_j\|^2]\frac{2}{(1-\beta_1)^{3/2}}$$

$$\overset{(b)}{\leq} \frac{6\alpha R\sqrt{1-\beta_2}}{(1-\beta_1)^2}\sum_{i\in[d]}\ln\left(1+\frac{v_{T,i}^{(sum)}}{\epsilon}\right) \tag{E.19}$$

$$\overset{(b)}{\leq} \frac{6\alpha R\sqrt{1-\beta_2}}{(1-\beta_1)^2}\sum_{i\in[d]}\ln\left(1+\frac{R^2}{(1-\beta_2)\epsilon}\right), \quad \text{(E.20)}$$

where in (a) we use Fact 11 and in (b) we use Lemma 22. For A, first note that,

$$\Omega_t = \sqrt{\sum_{\tau=1}^{t}\sum_{j=1}^{\tau}\beta_2^{\tau-j}} \leq \sqrt{\frac{t}{(1-\beta_2)}} \leq \sqrt{\frac{T}{(1-\beta_2)}}. \quad \text{(E.21)}$$

Let us change index, $j = t-k$, and use (E.21):

$$\begin{aligned}
\frac{\alpha}{2R}\sum_{t=1}^{T}\frac{1}{\Omega_t}\sum_{k=0}^{t-1}\beta_1^k\mathbb{E}[\|G_{t-k}^2\|^2] &\geq \frac{\alpha\sqrt{1-\beta_2}}{2R\sqrt{T}}\sum_{t=1}^{T}\sum_{j=1}^{t}\beta_1^{t-j}\mathbb{E}[\|G_j\|^2] \\
&= \frac{\alpha\sqrt{1-\beta_2}}{2R\sqrt{T}}\sum_{j=1}^{T}\mathbb{E}[\|G_j\|^2]\sum_{t=j}^{T}\beta_1^{t-j} \\
&= \frac{\alpha\sqrt{1-\beta_2}}{2(1-\beta_1)R\sqrt{T}}\sum_{j=1}^{T}(1-\beta_1^{T-j+1})\mathbb{E}[\|G_j\|^2] \\
&= \frac{\alpha\sqrt{1-\beta_2}}{2(1-\beta_1)R\sqrt{T}}\sum_{j=0}^{T-1}(1-\beta_1^{T-j})\mathbb{E}[\|\nabla F(x_j)\|^2].
\end{aligned}$$

Note, $\sum_{j=0}^{T-1}(1-\beta_1^{T-j}) = T - \beta_1\frac{1-\beta_1^T}{1-\beta_1} \geq T - \frac{\beta_1}{1-\beta_1} = \tilde{T}$, and let $\tau \in \{0,\dots,T-1\}$ and $\mathbb{P}[\tau=j] \propto 1-\beta_1^{T-j}$, then we have:

$$A \geq \frac{\alpha\sqrt{1-\beta_2}}{2(1-\beta_1)R\sqrt{T}}\tilde{T}\mathbb{E}[\|\nabla F(x_\tau)\|^2]. \quad \text{(E.22)}$$

Inserting (E.15), (E.16), (E.19), (E.22) in (E.14) yields that

$$\underbrace{\frac{\alpha\sqrt{1-\beta_2}}{2(1-\beta_1)R\sqrt{T}}\tilde{T}\mathbb{E}[\|\nabla F(x_\tau)\|^2]}_{A} \leq F(x_0)-F^* + \underbrace{\frac{\alpha^2 L}{2}\frac{1}{(1-\beta_1)^2}\sum_{i\in[d]}\ln\left(1+\frac{R^2T}{(1-\beta_2)\epsilon}\right)}_{B}$$
$$+\underbrace{\frac{\alpha^3L^2}{R\sqrt{1-\beta_2}}\frac{\beta_1}{(1-\beta_1)^4}\sum_{i\in[d]}\ln\left(1+\frac{R^2T}{(1-\beta_2)\epsilon}\right)}_{C} + \underbrace{\frac{6\alpha R\sqrt{1-\beta_2}}{(1-\beta_1)^2}\sum_{i\in[d]}\ln\left(1+\frac{R^2}{(1-\beta_2)\epsilon}\right)}_{D}.$$

and after some algebra we have,

$$\mathbb{E}[\|\nabla F(x_\tau)\|^2] \leq \frac{2(1-\beta_1)R\sqrt{T}}{\alpha\sqrt{1-\beta_2}\tilde{T}}(F(x_0)-F^*) + \frac{\alpha RL\sqrt{T}d}{\sqrt{1-\beta_2}(1-\beta_1)\tilde{T}}\ln\left(1+\frac{R^2T}{(1-\beta_2)^2\epsilon}\right)$$

$$+\frac{2\beta_1\alpha^2L^2\sqrt{T}d}{(1-\beta_2)(1-\beta_1)^3\tilde{T}}\ln\left(1+\frac{R^2T}{(1-\beta_2)\epsilon}\right)+\frac{12R^2\sqrt{T}d}{\sqrt{1-\beta_1}\tilde{T}}\ln\left(1+\frac{R^2T}{(1-\beta_2)\epsilon}\right),$$

Equivalently,

$$\mathbb{E}\|\nabla F(x_\tau)\|^2\leq\frac{2(1-\beta_1)R\sqrt{T}}{\alpha\sqrt{1-\beta_2}\tilde{T}}(F(x_0)-F^*)+C\frac{\sqrt{T}d}{\tilde{T}}\ln\left(1+\frac{R^2T}{(1-\beta_2)\epsilon}\right),$$

where $C=\frac{\alpha RL}{\sqrt{1-\beta_2}(1-\beta_1)}+\frac{2\beta_1\alpha^2L^2}{(1-\beta_2)(1-\beta_1)^3}+\frac{12R^2}{\sqrt{1-\beta_1}}$ which concludes the proof.

Proof of lemma 23. We start by separating the main term into two:

$$G_{t,i}\frac{m_{t,i}}{\sqrt{\epsilon+v_{t,i}^{(sum)}}}=\sum_{k=0}^{t-1}G_{t,i}\beta_1^{t-k}\frac{g_{t-k,i}}{\sqrt{\epsilon+v_{t,i}^{(sum)}}} \tag{E.23}$$

$$=\underbrace{\sum_{k=0}^{t-1}G_{t-k,i}\beta_1^{t-k}\frac{g_{t-k,i}}{\sqrt{\epsilon+v_{t,i}^{(sum)}}}}_{A}+\underbrace{\sum_{k=0}^{t-1}(G_{t,i}-G_{t-k,i})\beta_1^{t-k}\frac{g_{t-k,i}}{\sqrt{\epsilon+v_{t,i}^{(sum)}}}}_{B}. \tag{E.24}$$

For B, we again utilize the fact (F.8) that $\forall\lambda>0,\ x,y\in\mathbb{R}\quad xy\leq\frac{\lambda x^2}{2}+\frac{y^2}{2\lambda}$, for each dimension with $\lambda=\frac{\sqrt{1-\beta_1}}{2R\sqrt{1-\beta_2}\sqrt{k+1}}, x=|G_{t,i}-G_{t-k,i}|, y=\frac{|g_{t-k,i}|}{\sqrt{\epsilon+v_{t,i}^{(sum)}}}$. Then,

$$|B|\leq\sum_{i\in[d]}\sum_{k=0}^{t-1}\beta_1^t\left(\frac{\sqrt{1-\beta_1}}{4R\sqrt{1-\beta_2}\sqrt{k+1}}(G_{t,i}-G_{t-k,i})^2+\frac{2R\sqrt{1-\beta_2}\sqrt{k+1}g_{t-k,i}^2}{\sqrt{1-\beta_1}(\epsilon+v_{t-k,i}^{(sum)})}\right).$$

Note that $\epsilon+v_{t,i}^{(sum)}\geq\epsilon+v_{t-k,i}^{(sum)}$, hence,

$$\frac{g_{t-k,i}^2}{\epsilon+v_{t,i}^{(sum)}}\leq\frac{g_{t-k,i}^2}{\epsilon+v_{t-k,i}^{(sum)}}=U_{t-k,i}^2.$$

Moreover, smoothness of objective function implies

$$\|G_t-G_{t-k}\|^2\leq L^2\|x_{t-1}-x_{t-k-1}\|^2=L^2\|\sum_{l=1}^{k}\alpha u_{t-l}\|^2\leq\alpha^2L^2k\sum_{l=1}^{k}\|u_{t-l}\|^2$$

As a result,

$$|B|\leq\sum_{k=0}^{t-1}\frac{\alpha^2L^2\beta_1^k\sqrt{1-\beta_1}\sqrt{k}}{4R\sqrt{1-\beta_2}}\sum_{l=1}^{k}\|u_{t-l}\|^2+\sum_{k=0}^{t-1}\frac{R\sqrt{1-\beta_2}\sqrt{k+1}}{\sqrt{1-\beta_1}}\beta_1^k\|U_{t-k}\|^2$$

$$= \frac{\alpha^2 L^2}{4R\sqrt{1-\beta_2}}\sqrt{1-\beta_1}\sum_{l=1}^{t-1}\|u_{t-l}\|^2\sum_{k=l}^{t-1}\beta_1^k\sqrt{k} + \frac{R\sqrt{1-\beta_2}}{\sqrt{1-\beta_1}}\sum_{k=0}^{t-1}\sqrt{k+1}\beta_1^k\|U_{t-k}\|^2. \quad \text{(E.25)}$$

For term A, we focus on the main term of the summation $\mathbb{E}\left[G_{t-k}\frac{g_{t-k,i}}{\sqrt{\epsilon+v_{t,i}^{(sum)}}}\right]$. Let $G = G_{t-k,i}, g = g_{t-k,i}, \tilde{v} = \tilde{v}_{t,k+1,i}^{(sum)}, v = v_{t,i}^{(sum)}$. Note that,

$$\begin{aligned}
\tilde{v} - v &= v_{t-k,i}^{(sum)} + \mathbb{E}_{t-k-1}\left[\sum_{\tau=t-k}^{t}\sum_{j=1}^{\tau}\beta_2^{\tau-j}g_j^2\right] - v_{t,i}^{(sum)} \\
&= \mathbb{E}_{t-k-1}[\sum_{\tau=t-k}^{t}\sum_{j=1}^{\tau}\beta_2^{\tau-j}g_j^2] - \sum_{\tau=t-k}^{t} v_\tau \\
&= \underbrace{\mathbb{E}_{t-k-1}[\sum_{\tau=t-k}^{t}\sum_{j=1}^{\tau}\beta_2^{\tau-j}g_j^2]}_{A_1^2} - \underbrace{\sum_{\tau=t-k}^{t}\sum_{j=1}^{\tau}\beta_2^{\tau-j}g_j^2}_{A_2^2}.
\end{aligned}$$

We continue similar to Lemma 21.

$$\frac{Gg}{\sqrt{\epsilon+v}} = \frac{G^2}{\sqrt{\epsilon+\tilde{v}}} + \underbrace{Gg\frac{A_1^2 - A_2^2}{\sqrt{\epsilon+v}\sqrt{\epsilon+\tilde{v}}(\sqrt{\epsilon+v}+\sqrt{\epsilon+\tilde{v}})}}_{C}. \quad \text{(E.26)}$$

We have

$$|C| \le \underbrace{\frac{|Gg|A_1^2}{\sqrt{\epsilon+v}(\epsilon+\tilde{v})}}_{C_1} + \underbrace{\frac{|Gg|A_2^2}{(\epsilon+v)\sqrt{\epsilon+\tilde{v}}}}_{C_2}.$$

We will utilize (F.8), for C_1, let $\lambda = \frac{\sqrt{(1-\beta_1)}\sqrt{\epsilon+\tilde{v}}}{2}$, $x = \frac{|G|}{\sqrt{\epsilon+\tilde{v}}}$, $y = \frac{|g|A_1^2}{\sqrt{\epsilon+\tilde{v}}\sqrt{\epsilon+v}}$. Then,

$$C_1 \le \frac{G^2}{4\sqrt{\epsilon+\tilde{v}}} + \frac{1}{\sqrt{1-\beta_1}}\frac{g^2A_1^4}{(\epsilon+\tilde{v})^{\frac{3}{2}}(\epsilon+v)}. \quad \text{(E.27)}$$

Given $\epsilon + \tilde{v} \ge A_1^2$ and taking the expectation:

$$\mathbb{E}_{t-k-1}[C_1] \le \frac{G^2}{4\sqrt{\epsilon+\tilde{v}}} + \frac{1}{\sqrt{1-\beta_1}}\frac{A_1^2}{\sqrt{\epsilon+\tilde{v}}}\mathbb{E}_{t-k-1}\left[\frac{g^2}{(\epsilon+v)}\right].$$

Similarly, for C_2, we let $\lambda = \frac{\sqrt{1-\beta_1}\sqrt{\epsilon+\tilde{v}}}{2A_1^2}$, $x = \frac{|GA_2|}{\sqrt{\epsilon+\tilde{v}}}$, $y = \frac{|A_2 g|}{\epsilon+v}$. Then,

$$C_2 \le \frac{G^2}{4\sqrt{\epsilon+\tilde{v}}}\frac{A_2^2}{A_1^2} + \frac{1}{\sqrt{1-\beta_1}}\frac{A_1^2}{\sqrt{\epsilon+\tilde{v}}}\frac{g^2A_2^2}{(\epsilon+v)^2}.$$

Using $\epsilon + v \geq A_2^2$ and $\mathbb{E}_{t-k-1}[\frac{A_2^2}{A_1^2}] = 1$,

$$\mathbb{E}_{t-k-1}[C_2] \leq \frac{G^2}{4\sqrt{\epsilon+\tilde{v}}} + \frac{1}{\sqrt{1-\beta_1}} \frac{A_1^2}{\sqrt{\epsilon+\tilde{v}}} \mathbb{E}_{t-k-1}\left[\frac{g^2}{(\epsilon+v)}\right].$$

Hence,

$$\mathbb{E}_{t-k-1}[|C|] \leq \frac{G^2}{2\sqrt{\epsilon+\tilde{v}}} + \frac{1}{\sqrt{1-\beta_1}} \frac{2A_1^2}{\sqrt{\epsilon+\tilde{v}}} \mathbb{E}_{t-k-1}\left[\frac{g^2}{(\epsilon+v)}\right].$$

Using $A_1 \leq \sqrt{\epsilon+\tilde{v}}$, thus, $A_1 \leq R\sqrt{k+1}\sqrt{1-\beta_2}$:

$$\mathbb{E}_{t-k-1}[|C|] \leq \frac{G^2}{2\sqrt{\epsilon+\tilde{v}}} + \frac{2R\sqrt{k+1}\sqrt{1-\beta_2}}{\sqrt{1-\beta_1}} \mathbb{E}_{t-k-1}\left[\frac{g^2}{(\epsilon+v)}\right].$$

Taking complete expectation, using $\epsilon + v_{t,i}^{(sum)} \geq \epsilon + v_{t-k,i}^{(sum)}$ and reintroducing the indices:

$$\mathbb{E}[|C|] \leq \frac{1}{2}\mathbb{E}\left[\frac{G_{t-k,i}^2}{\sqrt{\epsilon+\tilde{v}_{t,k+1,i}^{(sum)}}}\right] + \frac{2R\sqrt{k+1}\sqrt{1-\beta_2}}{\sqrt{1-\beta_1}} \mathbb{E}_{t-k-1}\left[\frac{g_{t-k,i}^2}{(\epsilon+v_{t-k,i}^{(sum)})}\right].$$

Equivalently,

$$\mathbb{E}[-|C|] \geq -\frac{1}{2}\mathbb{E}\left[\frac{G_{t-k,i}^2}{\sqrt{\epsilon+\tilde{v}_{t,k+1,i}^{(sum)}}}\right] - \frac{2R\sqrt{k+1}\sqrt{1-\beta_2}}{\sqrt{1-\beta_1}} \mathbb{E}_{t-k-1}\left[\frac{g_{t-k,i}^2}{(\epsilon+v_{t-k,i}^{(sum)})}\right] \tag{E.28}$$

Note,

$$\mathbb{E}[|A|] \geq \sum_{i\in[d]}\sum_{k=0}^{t-1} \beta_1^k \left(\mathbb{E}\left[\frac{G_{t-k,i}^2}{\sqrt{\epsilon+\tilde{v}_{t,k+1,i}^{(sum)}}}\right] + \mathbb{E}[-|C|]\right)$$

Then, inserting (E.28), we have:

$$\begin{aligned}
\mathbb{E}[|A|] \geq \sum_{i\in[d]}\sum_{k=0}^{t-1} \beta_1^k \Bigg(&\mathbb{E}\left[\frac{G_{t-k,i}^2}{\sqrt{\epsilon+\tilde{v}_{t,k+1,i}^{(sum)}}}\right] \qquad (E.29)\\
&- \left(\frac{1}{2}\mathbb{E}\left[\frac{G_{t-k,i}^2}{\sqrt{\epsilon+\tilde{v}_{t,k+1,i}^{(sum)}}}\right] + \frac{2R\sqrt{k+1}\sqrt{1-\beta_2}}{\sqrt{1-\beta_1}} \mathbb{E}_{t-k-1}\left[\frac{g^2}{(\epsilon+v_{t-k,i}^{(sum)})}\right]\right)\Bigg)\\
= \frac{1}{2}\left(\sum_{i\in[d]}\sum_{k=0}^{t-1} \beta_1^k \mathbb{E}\left[\frac{G_{t-k,i}^2}{\sqrt{\epsilon+\tilde{v}_{t,k+1,i}^{(sum)}}}\right]\right) &- \frac{2R\sqrt{1-\beta_2}}{\sqrt{1-\beta_1}} \sum_{k=0}^{t-1} \beta_1^k \sqrt{k+1}\mathbb{E}[\|U_{t-k}\|^2]. \qquad (E.30)
\end{aligned}$$

Note, in (E.23) we have,

$$G_{t,i}\frac{m_{t,i}}{\sqrt{\epsilon+v_{t,i}^{(sum)}}} = A + B \geq A - |B|,$$

taking the expectation of both sides yields

$$\mathbb{E}\left[G_{t,i}\frac{m_{t,i}}{\sqrt{\epsilon+v_{t,i}^{(sum)}}}\right] = \mathbb{E}[A] + \mathbb{E}[B] \geq \mathbb{E}[A] + \mathbb{E}[-|B|], \tag{E.31}$$

Inserting (E.29) and negated (E.25) into (E.31) results in

$$\begin{aligned}\mathbb{E}\left[G_{t,i}\frac{m_{t,i}}{\sqrt{\epsilon+v_{t,i}^{(sum)}}}\right] \geq \frac{1}{2}\left(\sum_{i\in[d]}\sum_{k=0}^{t-1}\beta_1^k\mathbb{E}\left[\frac{G_{t-k,i}^2}{\sqrt{\epsilon+\tilde{v}_{t,k+1,i}^{(sum)}}}\right]\right) - \frac{3R\sqrt{1-\beta_2}}{\sqrt{1-\beta_1}}\sum_{k=0}^{t-1}\sqrt{k+1}\beta_1^k\mathbb{E}\|U_{t-k}\|^2 \\ - \frac{\alpha^2L^2}{4R\sqrt{1-\beta_2}}\sqrt{1-\beta_1}\sum_{l=1}^{t-1}\|u_{t-l}\|^2\sum_{k=l}^{t-1}\beta_1^k\sqrt{k},\end{aligned}$$

which completes the proof. □

Proof of Lemma 24. For some t,

$$\frac{c_t^2}{\epsilon+b_t} \leq \frac{1}{1-\beta_1}\sum_{\tau=1}^{t}\beta_1^{t-\tau}\frac{a_\tau^2}{\epsilon+b_t}.$$

We have $\epsilon + b_t \geq \epsilon + b_\tau$ for any $\tau \leq t$, then,

$$\begin{aligned}\sum_{t=1}^{T}\frac{c_t^2}{\epsilon+b_t} &\leq \frac{1}{1-\beta_1}\sum_{t=1}^{T}\sum_{\tau=1}^{t}\beta_1^{t-\tau}\frac{a_\tau^2}{\epsilon+b_\tau} \\ &= \frac{1}{1-\beta_1}\sum_{\tau=1}^{T}\frac{a_\tau^2}{\epsilon+b_\tau}\sum_{\tau=1}^{t}\beta_1^{t-\tau} \\ &\leq \frac{1}{(1-\beta_1)^2}\sum_{\tau=1}^{T}\frac{a_\tau^2}{\epsilon+b_\tau} \\ &\leq \frac{1}{(1-\beta_1)^2}\ln\left(1+\frac{b_T}{\epsilon}\right),\end{aligned}$$

where in the last inequality we apply Lemma 22 with $\rho = 0$. Note, from the definition of b_T we also have that $b_T \leq \frac{R^2T}{1-\beta_2}$. □

E.3 Additional Experiments and Details

E.3.1 Overall Parameterization in the Experiments

In Section 7.5, we provided the overall parameterization verbally, here we give the vector updates for the overall parameterization. For the *first order moment term* we have,

$$\begin{aligned} m_t &= \beta_1 m_{t-1} + (1-\beta_1) g_t \\ n_t &= \beta_3 n_{t-1} + (1-\beta_3)(g_t - g_{t-1}) \\ m_t^{lion} &= \beta_2^{lion} m_{t-1}^{lion} + (1-\beta_2^{lion}) g_t \\ u_t &= \beta_1^{lion} m_{t-1}^{lion} + (1-\beta_1^{lion}) g_t, \end{aligned}$$

where β_1 controls the strength of heavy-ball momentum and β_3 extends the equation to Nesterov momentum. The first order moment equations essentially capture the updates for Adan and Adam. For the *second order moment term* we have,

$$\begin{aligned} \hat{g}_t &= g_t + \beta_3(g_t - g_{t-1}) \\ \tilde{g}_t^2 &= c_{t-1}\hat{g}_t^2 + (1-c_{t-1})(\bar{v}_{t-1} + \hat{g}_t^2 \text{sign}(\hat{g}_t^2 - \bar{v}_{t-1})) \\ \bar{v}_t &= \beta_2 \bar{v}_{t-1} + (1-\beta_2)\tilde{g}_t^2 \\ \tilde{v}_t &= (\bar{v}_t + (t-1)\tilde{v}_{t-1})/t \\ v_t &= \rho \bar{v}_t + (1-\rho)\tilde{v}_t \end{aligned}$$

where c controls whether the difference of consequent $\bar{v}_t$s grows with $g_t^2 \text{sign}(g_t^2 - \bar{v}_{t-1})$, as in YOGI, or $g_t^2(g_t^2 - \bar{v}_{t-1})$ as in Adam; the former, prevents rapid increases in effective learning rate and provides more controlled updates. For the second order moment term, ρ_t determines whether to use, less noisy, average of past v_ts (we call this method AVGrad and defer its formal introduction to the next section) or the current v_t, which may be more noisy but up to date. Lastly, the *update term* together with the decoupled weight decay is as follows,

$$x_{t-1} = x_{t-1} - \lambda \alpha_t x_{t-1}$$

$$x_t = x_{t-1} - \alpha_t \left(\gamma \frac{m_t + \beta_3 n_t}{\sqrt{v_t} + \epsilon} + (1 - \gamma) \text{sign}(u_t) \right),$$

where ϵ is the stability parameter, $\text{sign}(u_t)$ is the element-wise sign operator, γ controls whether to do a Lion type update or not.

E.3.2 Training setup for the experiments

On OpenWebText, we use a global batch size of 480 sequences, cosine learning rate schedule with the peak learning rate of 6×10^{-4} (1.5×10^{-4} for Lion) and the final learning rate of 1.5×10^{-5} (as chosen for Sophia algorithm in [103]). For Lion we use the tuned learning rate from (Liu, 2023) for the same setting. For Adam we found that employing Sophia's learning rate schedule results in better loss compared to employing Adam's learning rate schedule in [103]. For our methods: AVGrad, `MADA`, `MADA-FS`, we directly employ Adam's learning rate and learning schedule without any tuning since our goal is to demonstrate that `MADA` can be plugged in place of Adam without any changes in learning rate schedule.

We run the experiment for 100,000 iterations (first 2000 are warmup iterations) which corresponds to training on $\sim$ 492B tokens. We use a weight decay parameter of 0.1. On Shakespeare, we use a batch size of 64, cosine learning rate schedule with the peak learning rate of 10^{-3} and the final learning rate of 10^{-4}. We run the experiment for 5,000 iterations (first 100 are warmup iterations). We run our experiments on AWS `p5.48xlarge` instances equipped with 8 NVIDIA H100 GPUs. To be able utilize multiple GPU's we allreduced hyper-gradients across GPU's.

We initialize `MADA` with $\beta_3 = 0.9, \rho = 0$ on top of Adam's established β_1, β_2 parameters for OpenWebText experiments as it resulted in a better validation loss. For Shakespeare experiments, we compare `MADA` against Adam over a grid of (β_1, β_2) values, where in the case of `MADA` (β_1, β_2) represents the initial values. For the OpenWebText experiments, we do not update hyper-parameters for the first 50 iterations for the sake of stability. For both experiments we use SGD as the hyper-parameter optimizer. On Shakespeare, for 10M model

we use $2.5e{-}3$ learning rate and 0.5 momentum (we do not use momentum for γ) for learning the hyper-parameters. On OpenWebText (and for 1.5B model on Shakespeare) we use a learning rate of $5e{-}4$ for training β_1, β_2 and $1e{-}1$ for other hyper-parameters.

Vision Tasks. For 5-layer model experiments, we use a CNN whose first two layers are convolutional layers with 6 and 16 output channels and 5 kernel size; and last 3 layers are fully connected layers with 120, 84, and 10 output dimensionality. We use ReLU activation in all layers except the last one and maxpool on the outputs convolutional layers. We use a constant learning rate of $1e{-}3$ for MADA, Adam, HyperAdam; and $1e{-}2$ learning rate and 0.9 momentum coefficient for SGD with momentum. We initialize `MADA` from Adam state, we set hyper learning rate for β_1, β_2 to $1e{-}4$ and for the other variables to $1e{-}2$. We use batch size of 256 and train for 50 epochs. For ResNet-9 experiments, we use the model implementation from https://github.com/Moddy2024/ResNet-9. We use one cycle learning rate scheduler with $1e{-}2$ peak learning rate for `MADA`, Adam, HyperAdam and $1e{-}1$ peak learning rate for SGD with momentum. Again we initialize `MADA` from Adam, we set hyper learning rate for β_1 to $1e{-}4$, for β_2 to $1e{-}4$ and for the other variables to $1e{-}2$. We use batch size of 400 and train for 50 epochs.

E.3.3 Additional Experiments

Method	OpenWebText (validation loss)	OpenWebText	Wikitext	Lambada
`MADA`	2.8853	17.9084	61.4689	73.1291
`MADA`-FS	2.8838	17.8822	63.9886	75.6158

Table E.1 Validation loss on OpenWebText and validation perplexities on OpenWebText, Wikitext and Lambada datasets of GPT-2 (125M) models trained on OpenWebText with `MADA` when the initial optimizer state is AVGrad.

In these tables, we observe `MADA` outperforms Adam in training of GPT-2 (medium) as well. Moreover, we observe GPT-2 (small) training with different initialization (from AVGrad).

Method	OpenWebText (validation loss)	OpenWebText	Wikitext	Lambada
Adam	2.6527	14.1928	50.1410	53.7468
MADA	2.6422	14.0441	43.8067	53.7575

Table E.2 Validation loss on OpenWebText and validation perplexities on OpenWebText, Wikitext and Lambada datasets of GPT-2 (355M) models trained on OpenWebText with Adam and MADA. The initial state of MADA is AVGrad + Adan ($\rho = 0, \beta_3 = 0.9$).

Synthetic convex experiment. This is a famous example from [143], where Adam notably fails to converge to the optimal solution, $x = -1$; whereas AMSGrad (and AVGrad) reach the optimum. The online learning experiment given in [143] to motivate AMSGrad is as follows:

$$g_t(x) := \begin{cases} 1010x & \text{for } t \bmod 101 = 1 \\ -10x & \text{otherwise} \end{cases} \tag{E.32}$$

with constraint set $x \in [-1, 1]$ and t denotes the time index in the online learning setting. In Figure E.1, we plot the average regret which is defined by $(g_t(x) - g_t(-1))/t$, as well as the evolution of x and ρ with respect to iterations. Here, we reduce the effect of Lion by assigning a small hyper-learning rate to γ, since signSGD based methods neutralize the effect of large gradients which allow faster progress towards the optimum. We observe that even when we initialize MADA from Adam ($\rho_0 = 1$), it quickly recovers AVGrad ($\rho_t \to 0$), which is the right optimizer to use for this example. This experiment shows that MADA can learn to behave like AVGrad even when initialized from Adam.

E.4 Details on Computational Burden

As mentioned in the main text, the additional computation can be broken down into two components: additional per-parameter computational steps during optimizer update, and the

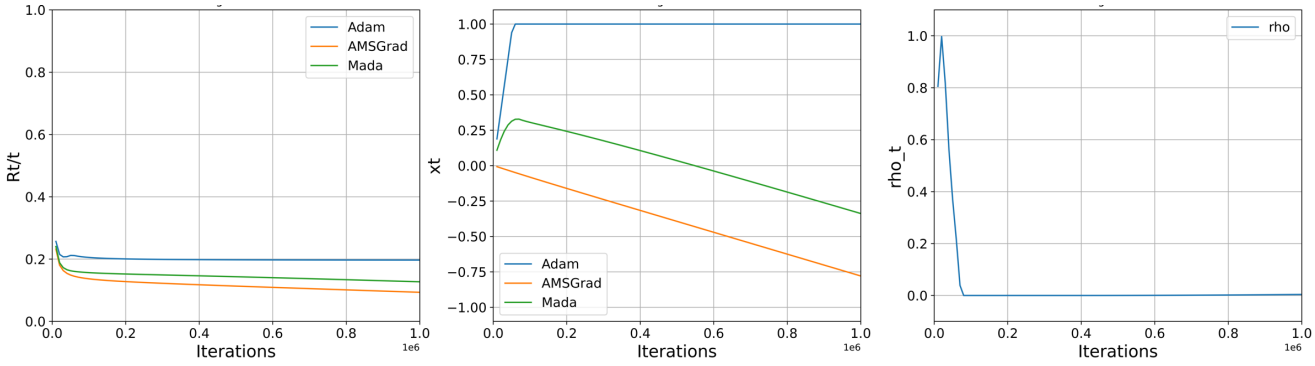


Figure E.1 Average regret, x_t, ρ_t with respect to iterations for `MADA` on simple convex function.

computation of hyper-gradients. Note that the first component is negligible compared to the overall FLOPs requirement of a single training step in a language model. This is because for a model with N parameters trained over a batch of T tokens, the model parameter update in the parameterized optimizer involves cN FLOPs (with c being on the range of 10-20 depending on the parameterization), while the forward-backward passes require approximately $6TN$ FLOPs, with T being on the order of thousands. To analyze the second component, consider the hyper-gradient example in (6.4), where the computation of hyper-gradient with respect to ρ involves several vector-level element-wise operations (note that the multiplication of the first term with second does not require a matrix-vector multiplication, since it can be implemented by a dot product with the numerator followed by element-wise division with the denominator), where each vector is of size N. As a result, the FLOPs requirement is still $O(N)$ (does not scale with T), and much smaller than the $6TN$ required for the forward-backward passes. The derivative with respect to the other coefficients can similarly be shown to have $O(N)$ complexity.

We also profile the memory footprints of the methods and find that the peak memory usage is 15.5 GB for Adam, and 24.9 GB for `MADA`; time per iteration is 0.65s for Adam and 0.85s for `MADA`. If we exclude LION (which contributes the least in this setting) MADA results in 22.2GB of memory usage and 0.72s time per iteration.

APPENDIX F

Appendix of Chapter 7.6

F.1 Implicit Regularization and Proof of Theorem 14

Proof of Lemma 5. Gradient perturbation can be defined as the effective perturbation on the gradient that is induced by the update with SR. In other words, for gradient descent as an example, we have the following equality:

$$x - \alpha(\nabla F(x) + \xi_\alpha) = Q_{SR}(x - \alpha\nabla F(x))$$

As a result ξ_α is defined as:

$$\xi_\alpha(x) := \begin{cases} \frac{1}{\alpha}\Big(\lceil (x-\alpha\nabla F(x)\rceil - (x-\alpha\nabla F(x))\Big), & \text{w.p. } \frac{(x-\alpha\nabla F(x) - \lfloor (x-\alpha\nabla F(x)\rfloor}{\Delta} \\ \frac{1}{\alpha}\Big(\lfloor (x-\alpha\nabla F(x)\rfloor - (x-\alpha\nabla F(x)\Big), & \text{w.p. } \frac{\lceil (x-\alpha\nabla F(x)\rceil - (x-\alpha\nabla F(x)}{\Delta} \end{cases}.$$

where $\Delta = \lceil x - \alpha\nabla F(x)\rceil - \lfloor x - \alpha\nabla F(x)\rfloor$ is the difference between quantization levels in terms of gradients. $\mathbb{E}[\|\xi_\alpha\|^2] = \frac{1}{\alpha^2}\Big(\lceil (x-\alpha\nabla F(x)\rceil - (x-\alpha\nabla F(x))\Big)^\top \Big(\lfloor (x - \alpha\nabla F(x)\rfloor - (x-\alpha\nabla F(x)\Big)$. Consequently, when $\alpha \to 0$, after some algebra, we have $\lim_{\alpha\to 0}\alpha\mathbb{E}[\|\xi_\alpha\|^2] = \sum_{i=1}^d \Delta_{x_i}|\nabla F(x)_i|$.

□

Theorem (Restatement of theorem 14.)**.** *Let $F(x)$ be the loss function to be minimized, α be a small learning rate, $\xi_\alpha(x_t)$ be random vector quantization error while doing updates with SR. Then, implicitly, the following expected modified loss, $F_{SR}(x)$, is being optimized:*

$$
\begin{aligned}
F_{SR}(x) &= F(x) + \frac{\alpha}{4}\|\nabla F(x)\|^2 + \frac{\alpha}{4}\mathbb{E}\|\xi_\alpha(x)\|^2 \\
&= F(x) + \frac{\alpha}{4}\|\nabla F(x)\|^2 \\
&\quad + \frac{\alpha}{4}\sum_{i=1}^{d}\Big(\lceil -\nabla F(x)\rceil_\alpha - (-\nabla F(x))\Big)_i\Big(-\nabla F(x) - \lfloor -\nabla F(x)\rfloor_\alpha\Big)_i
\end{aligned}
$$

Proof. Our proof follows similar logic to [10], but we consider the perturbation due to SR. We start by defining $\hat{l}(x) := l(x) + \xi_\alpha(x)$ which is the update due to gradient descent when SR is applied. To measure the introduced error due to taking steps using modified-perturbed flow we need to compare the continuous updates and discrete updates. Note, the discrete updates with perturbed gradient flow results in the following modified flow:

$$
\begin{aligned}
\tilde{l}(x(t)) &:= \hat{l}(x(t)) + \alpha\hat{l}_1(x(t)) + \alpha^2\hat{l}_2(x(t)) + \mathcal{O}(\alpha^3) \\
&= l(x) + \xi_\alpha(x) + \alpha\hat{l}_1(x(t)) + \alpha^2\hat{l}_2(x(t)) + \mathcal{O}(\alpha^3)
\end{aligned}
$$

Note that gradient descent (Euler method) is defined by $x_{t+1} = x_t + \alpha l(x_t)$; in this work we are interested in the average path the discrete updates follow, i.e., $\mathbb{E}_\xi x_{t+1} = x_t + \alpha\mathbb{E}_\xi[\tilde{l}(x_t)]$. The goal of backward error analysis is to compute $\hat{l}_i$s; here, we equate one step of gradient descent on original gradient flow with Taylor expansion via the modified-perturbed flow (in expectation). This yields,

$$
\begin{aligned}
x_t + \alpha l(x_t) &= \mathbb{E}_\xi[x_t + \alpha\tilde{l}(x_t) + \alpha^2\nabla\tilde{l}(x_t)\tilde{l}(x_t) + \mathcal{O}(\alpha^3)] \\
&= x(t) + \alpha l(x(t)) + \alpha^2(\hat{l}_1(x(t)) + \frac{1}{2}\mathbb{E}_\xi[\nabla(l(x(t)) + \xi_\alpha(x(t)))(l(x(t)) + \xi_\alpha(x(t))) + \mathcal{O}(\alpha^3)
\end{aligned}
$$

Assuming $l(x) \neq 0$ (this corresponds to case where the local minimum is achieved and derivative of ξ is not continuous) we have that expectation and differentiation operators commute due to Leibniz rule. Using zero mean property of ξ, therefore we have,

$$
x_t + \alpha l(x_t) = x(t) + \alpha l(x(t)) + \alpha^2(l_1(x(t)) + \frac{1}{2}\nabla(l(x(t))l(x(t)) + \frac{1}{2}\mathbb{E}[\nabla\xi_\alpha(x(t)))\xi_\alpha(x(t))])
$$

$$+ \mathcal{O}(\alpha^3)$$

Now we can compute $\hat{l}_1(x)$ as,

$$\hat{l}_1(x(t)) = -\frac{1}{2}\nabla f(x(t))f(x(t)) - \frac{1}{2}\mathbb{E}[\nabla\xi_\alpha(x(t)))\xi_\alpha(x(t))]\Bigg)$$
$$= -\nabla\|\nabla F(x)\|^2 - \nabla\mathbb{E}\|\xi_\alpha(x)\|^2$$

Then we found the average perturbed-modified flow as:

$$\mathbb{E}\dot{x} = l(x) + \mathbb{E}\xi_\alpha(x) + \mathbb{E}\alpha\hat{l}_1(x) + \mathcal{O}(\alpha^2) = -\nabla F(x) - \frac{\alpha}{4}\nabla\|\nabla F(x)\|^2 - \frac{\alpha}{4}\nabla\mathbb{E}\|\xi_\alpha(x)\|^2$$
$$= -\nabla\left(F(x) + \frac{\alpha}{4}\|\nabla F(x)\|^2 + \frac{\alpha}{4}\mathbb{E}\|\xi_\alpha(x)\|^2\right).$$

and we obtain the theorem statement. Note that with this modified flow, $\hat{l}_1$ is constructed such that there is a $\mathcal{O}(\alpha^3)$ error between the true solution of gradient flow and discrete update, i.e. $\|\mathbb{E}_\xi x_t - x(t)\| < \mathcal{O}(\alpha^3)$. This is similar to [10]; hence, on average SR does not introduce additional error as expected due the non-biasedness. For any biased rounding scheme, such as nearest, $\mathbb{E}\xi_\alpha(x)$ in the update step cannot be cancelled, as a result there will be a $\mathcal{O}(\alpha)$ error difference between the true updates and modified updates at every iteration.

□

F.2 Adam with quantized model updates convergence

F.2.1 Proof of Theorem 15

Theorem (Restated theorem 15)**.** *When SR is used for the update step to obtain quantized weights we have the following upper bound:*

$$\frac{1}{T}\sum_{t=1}^{T}\mathbb{E}\|\nabla F(x_t)\|^2 \leq \frac{2Rd}{T}\left(\frac{2R}{\sqrt{1-\beta_2}} + \frac{\alpha L}{1-\beta_2}\right)\left(\ln\left(1 + \frac{R^2}{\epsilon(1-\beta_2)}\right) + T\ln\left(\frac{1}{\beta_2}\right)\right)$$
$$+ \frac{2R(F_0 - F^*)}{\alpha T} + \frac{Rd\Delta L}{T\sqrt{1-\beta_2}}\left(\sqrt{T\ln\left(1 + \frac{R^2}{\epsilon(1-\beta_2)}\right)} + T\sqrt{\ln\left(\frac{1}{\beta_2}\right)}\right).$$

Adaptive updates. Similar to [39] we define the following vectors iteratively.

$$m_{t,i} = \beta_1 m_{t-1,i} + \nabla_i f_t(x_{t-1})$$
$$v_{t,i} = \beta_2 v_{t-1,i} + \nabla_i f_t(x_{t-1})^2$$

Note that $\hat{m}_{t,i} = (1-\beta_1)m_{t,i}$ and $\hat{v}_{t,i} = (1-\beta_2)v_{t,i}$ give the updates in Adam. we use the learning rate to absorb $(1-\beta_1), (1-\beta_2)$ terms for simplicity. In particular, if one has an update $x_{t,i} = x_{t-1,i} - \alpha_t \frac{m_{t,i}}{\sqrt{\epsilon + v_{t,i}}}$; Adam is recovered with $\alpha_t = \alpha \frac{1-\beta_1}{\sqrt{1-\beta_2}}$. If we include the bias correction terms, Adam is obtained with $\alpha_t = \alpha \frac{\sqrt{1-\beta_2^t}}{\sqrt{1-\beta_2}} \frac{1-\beta_1^t}{1-\beta_1}$. For the update step we further define:

$$x_{t,i} = Q_{SR}\left(x_{t-1,i} - \alpha_t \frac{m_{t,i}}{\sqrt{\epsilon + v_{t,i}}} \right).$$

Lemma 25. *Let us define $u_{t,i} := \frac{m_{t,i}}{\sqrt{\epsilon + v_{t,i}}}$, the perturbation due to SR updates, defined as $r_{t,i} := Q_{SR}(x_{t-1,i} - \alpha_t u_{t,i}) - (x_{t-1,i} - \alpha_t u_{t,i})$ at timestep t per dimension $i \in [d]$ has the following upper bound on l_2 norm,*

$$\mathbb{E}_p \|r_t\|_2^2 \leq \Delta \alpha_t \|u_t\|_1 \leq \Delta \sqrt{d} \alpha_t \|u_t\|_2 \tag{F.1}$$

where $\mathbb{E}_p$ is the expectation with respect to randomness of SR and $\Delta = \Delta_{x_{t-1,i} - \alpha_t u_{t,i}}$.

Proof. Here we consider the update terms in Adam instead of gradient terms, other than that the proof is similar to Lemma 1 in [97]. We redefine the perturbation due to SR updates for ease of analysis, $r_{t,i} = Q_{SR}(x_{t-1,i} - \alpha_t \frac{m_{t,i}}{\sqrt{\epsilon + v_{t,i}}}) - (x_{t-1,i} - \alpha_t \frac{m_{t,i}}{\sqrt{\epsilon + v_{t,i}}})$. Choose a random number $p \in [0,1]$, $r_{t,i}$ can be expressed as,

$$r_{t,i} = \Delta \cdot \begin{cases} \frac{-x_{t-1} + \alpha_t u_{t,i}}{\Delta} + \frac{\lfloor x_{t-1} - \alpha_t u_{t,i} \rfloor}{\Delta} + 1, & \text{when } p \leq \frac{x_{t-1} - \alpha_t u_{t,i}}{\Delta} - \frac{\lfloor x_{t-1} - \alpha_t u_{t,i} \rfloor}{\Delta} \\ \frac{-x_{t-1} + \alpha_t u_{t,i}}{\Delta} + \frac{\lfloor x_{t-1} - \alpha_t u_{t,i} \rfloor}{\Delta} & \text{otherwise.} \end{cases}$$

Defining $q = \frac{x_{t-1}-\alpha_t u_{t,i}}{\Delta} - \frac{\lfloor x_{t-1}-\alpha_t u_{t,i} \rfloor}{\Delta}$, note $q \in [0,1]$, we can rewrite,

$$r_{t,i} = \Delta \cdot \begin{cases} -q+1, & \text{when } p \leq q \\ -q & \text{otherwise.} \end{cases} \tag{F.2}$$

Consequently, $\mathbb{E}[r_{t,i}] = 0$, and it can be shown that

$$\begin{aligned} \mathbb{E}[r_{t,i}^2] &\leq \Delta^2 q(1-q) \\ &\leq \Delta^2 \min\{q, 1-q\} \\ &\leq \Delta^2 \alpha_t \frac{|u_{t,i}|}{\Delta} \\ &= \Delta \alpha_t |u_{t,i}|, \end{aligned}$$

where the last inequality is due to the fact that the update term will be at least as large as the error term. Then $\mathbb{E}\|r_t\|_2^2 \leq \Delta \alpha_t \|u_t\|_1 \leq \Delta\sqrt{d}\alpha_t \|u_t\|_2$. □

Let us start by defining $\tilde{v}_{t,i}$ which is a virtual sequence where the contribution of the last gradient is replaced by its expected value conditioned on past iterations.

$$\tilde{v}_{t,i} = \beta_2 v_{t-1,i} + \mathbb{E}_{t-1} \nabla_i f_t(x_{t-1})^2.$$

We also define the perturbed updates as follows:

$$\tilde{u}_{t,i} = \frac{\nabla_i f_t(x_{t-1})}{\sqrt{\epsilon + v_{t,i}}} + \frac{r_{t,i}}{\alpha_t}.$$

From the smoothness of the objective function F, we can use the Descent Lemma to obtain that

$$\begin{aligned} F(x_t) &\leq F(x_{t-1}) - \alpha_t \nabla F(x_{t-1})^\top \tilde{u}_t + \frac{\alpha^2 L}{2} \|\tilde{u}_t\|^2 \\ &= F(x_{t-1}) - \alpha_t \nabla F(x_{t-1})^\top \left(u_t + \frac{r_t}{\alpha_t} \right) + \frac{\alpha_t^2 L}{2} \left\| u_t + \frac{r_t}{\alpha_t} \right\|^2 \end{aligned}$$

where r_t is defined in lemma 25. Taking the expectation of both sides with respect to sampling randomness (conditioned on $l_1, \ldots, f_{t-1}$) and SR randomness, we have:

$$\begin{aligned}
\mathbb{E}_{p,t-1}F(x_t) &\leq F(x_{t-1}) - \alpha_t \sum_{i\in[d]} \mathbb{E}_{t-1}\left[\nabla_i F(x_{t-1})\frac{\nabla_i f_t(x_{t-1})}{\sqrt{\epsilon + v_{t,i}}}\right] + \frac{\alpha_t^2 L}{2}\mathbb{E}_{p,t-1}\left\|u_t + \frac{r_t}{\alpha_t}\right\|^2 \\
&= F(x_{t-1}) - \alpha_t \sum_{i\in[d]} \mathbb{E}_{t-1}\left[\nabla_i F(x_{t-1})\frac{\nabla_i f_t(x_{t-1})}{\sqrt{\epsilon + v_{t,i}}}\right] + \frac{\alpha_t^2 L}{2}\mathbb{E}_{t-1}\|u_t\|^2 + \frac{L}{2}\mathbb{E}_{p,t-1}\|r_t\|^2 \\
&\leq F(x_{t-1}) - \alpha_t \sum_{i\in[d]} \mathbb{E}_{t-1}\left[\nabla_i F(x_{t-1})\frac{\nabla_i f_t(x_{t-1})}{\sqrt{\epsilon + v_{t,i}}}\right] + \frac{\alpha_t^2 L}{2}\mathbb{E}_{t-1}\|u_t\|^2 \qquad \text{(F.3)} \\
&\quad + \frac{\alpha_t\sqrt{d}\Delta L}{2}\mathbb{E}_{t-1}\|u_t\|_2. \qquad \text{(F.4)}
\end{aligned}$$

where the first equality holds due to $\mathbb{E}[r_t] = 0$. To upper bound the second term on the right hand side, we use the following lemma which is a restatement of Lemma 5.1 in [39], we provide the proof for completeness.

Lemma 26 (Lemma 5.1 in [39], updates approximately follow a descent direction)**.**

$$\mathbb{E}_{t-1}\left[\nabla_i F(x_{t-1})\frac{\nabla_i f_t(x_{t-1})}{\sqrt{\epsilon + v_{t,i}}}\right] \geq \frac{\nabla_i F(x_{t-1})^2}{2\sqrt{\epsilon + \tilde{v}_{t,i}}} - 2R\mathbb{E}_{t-1}\left[\frac{\nabla_i f_t(x_{t-1})^2}{\epsilon + v_{t,i}}\right]. \qquad \text{(F.5)}$$

Proof. For the sake of simplicity, we define $G = \nabla_i F(x_{t-1}),\ g = \nabla_i f_t(x_{t-1}), v = \bar{v}_{t,i}, \tilde{v} = \tilde{v}_{t,i}$. First obviously, we have

$$\frac{Gg}{\sqrt{\epsilon + v}} = \frac{Gg}{\sqrt{\epsilon + \tilde{v}}} + \underbrace{\frac{Gg}{\sqrt{\epsilon + v}} - \frac{Gg}{\sqrt{\epsilon + \tilde{v}}}}_{A}. \qquad \text{(F.6)}$$

For the first term, we use the fact that $\mathbb{E}_{t-1}[g] = G$ to obtain that

$$\mathbb{E}_{t-1}\left[\frac{Gg}{\sqrt{\epsilon + \tilde{v}}}\right] = \frac{G^2}{\sqrt{\epsilon + \tilde{v}}}. \qquad \text{(F.7)}$$

In order to bound A, we begin with

$$A = Gg\frac{\tilde{v} - v}{\sqrt{\epsilon + v}\sqrt{\epsilon + \tilde{v}}(\sqrt{\epsilon + v} + \sqrt{\epsilon + \tilde{v}})} = Gg\frac{\mathbb{E}_{t-1}g^2 - g^2}{\sqrt{\epsilon + v}\sqrt{\epsilon + \tilde{v}}(\sqrt{\epsilon + v} + \sqrt{\epsilon + \tilde{v}})},$$

where the last equality follows from the fact that $\tilde{v} - v = \mathbb{E}_{t-1}[g^2] - g^2$ (see the definition of $v_{t,i}$). Hence, from the triangle inequality we have that

$$|A| \leq \underbrace{|Gg| \frac{\mathbb{E}_{t-1} g^2}{\sqrt{\epsilon + v}(\epsilon + \tilde{v})}}_{A_1} + \underbrace{|Gg| \frac{g^2}{\sqrt{\epsilon + \tilde{v}}(\epsilon + v)}}_{A_2},$$

where in the inequality follows from the fact that $\sqrt{\epsilon + v} + \sqrt{\epsilon + \tilde{v}} \geq \max(\sqrt{\epsilon + v}, \sqrt{\epsilon + \tilde{v}})$. A useful fact that will be used later is the following

$$\forall \lambda > 0,\ x, y \in \mathbb{R} \quad xy \leq \frac{\lambda x^2}{2} + \frac{y^2}{2\lambda}. \tag{F.8}$$

To bound A_1, we use (F.8) with $\lambda = \frac{\sqrt{\epsilon+\tilde{v}}}{2}, x = \frac{|G|}{\sqrt{\epsilon+\tilde{v}}}, y = |g| \frac{\mathbb{E}_{t-1} g^2}{2\sqrt{\epsilon+\tilde{v}}\sqrt{\epsilon+v}}$, which yields that

$$A_1 \leq \frac{G^2}{4\sqrt{\epsilon + \tilde{v}}} + \frac{g^2 \mathbb{E}_{t-1}[g^2]^2}{(\epsilon + \tilde{v})^{3/2}(\epsilon + v)}.$$

Taking the expectation and noting $\epsilon + \tilde{v} \geq \mathbb{E}_{t-1}[g^2]$ ensures that

$$\mathbb{E}_{t-1}[A_1] \leq \frac{G^2}{4\sqrt{\epsilon + \tilde{v}}} + \frac{\mathbb{E}_{t-1}[g^2]}{\sqrt{\epsilon + \tilde{v}}} \mathbb{E}_{t-1}\left[\frac{g^2}{\epsilon + v}\right] \leq \frac{G^2}{4\sqrt{\epsilon + \tilde{v}}} + R\mathbb{E}_{t-1}\left[\frac{g^2}{\epsilon + v}\right] \tag{F.9}$$

where the last inequality uses our boundedness assumption $\sqrt{\mathbb{E}_{t-1}[g^2]} \leq R$ and the fact that $\sqrt{\epsilon + \tilde{v}} \geq \sqrt{\mathbb{E}_{t-1}[g^2]}$. Similarly, for A_2, we use (F.8) with $\lambda = \frac{\sqrt{\epsilon+\tilde{v}}}{2\mathbb{E}_{t-1} g^2}, x = \frac{|Gg|}{\sqrt{\epsilon+\tilde{v}}}, y = \frac{g^2}{\epsilon+v}$, which gives us (we used similar bounding arguments)

$$\mathbb{E}_{t-1}[A_2] \leq \frac{G^2}{4\sqrt{\epsilon + \tilde{v}}} + R\frac{\mathbb{E}_{t-1}[g^2]}{\epsilon + v}. \tag{F.10}$$

Combining both upper bounds we have,

$$\mathbb{E}_{t-1}[|A|] \leq \frac{G^2}{2\sqrt{\epsilon + \tilde{v}}} + 2R\frac{\mathbb{E}_{t-1}[g^2]}{\epsilon + v},$$

which can be equivalently written as follows

$$\mathbb{E}_{t-1}[-|A|] \geq -\left(\frac{G^2}{2\sqrt{\epsilon + \tilde{v}}} + 2R\frac{\mathbb{E}_{t-1}[g^2]}{\epsilon + v}\right),$$

Finally, combining further with (F.7) and substituting into (F.6) gives us,

$$\mathbb{E}_{t-1}\left[\frac{Gg}{\sqrt{\epsilon + v}}\right] \geq \frac{G^2}{\sqrt{\epsilon + \tilde{v}}} - \left(\frac{G^2}{2\sqrt{\epsilon + \tilde{v}}} + 2R\frac{\mathbb{E}_{t-1}[g^2]}{\epsilon + v}\right) = \frac{G^2}{2\sqrt{\epsilon + \tilde{v}}} - 2R\frac{\mathbb{E}_{t-1}[g^2]}{\epsilon + v},$$

which proves the desired result. □

Note that for any $i \in [d]$, $\sqrt{\epsilon + \tilde{v}_{t,i}} \leq R\sqrt{1 + \sum_{j=0}^{t-1} \beta_2^t} \leq \frac{R}{\sqrt{1-\beta_2}}$. Hence,

$$\alpha_t \frac{(\nabla_i F(x_{t-1}))^2}{2\sqrt{\epsilon + \tilde{v}_{t,i}}} \geq \alpha \frac{\nabla_i F(x_{t-1})^2}{2R}. \tag{F.11}$$

Using (F.11) in conjunction with Lemma 26 in (F.4) further yields that

$$\begin{aligned}\mathbb{E}_{t-1}[F(x_t)] \leq F(x_{t-1}) - \left(\frac{\alpha}{2R}\|\nabla F(x_{t-1})\|^2 - 2\alpha_t R \mathbb{E}_{t-1}[\|u_t\|^2]\right) + \frac{\alpha_t^2 L}{2}\mathbb{E}_{t-1}[\|u_t\|^2] \\ + \frac{\alpha_t \sqrt{d}\Delta L}{2}\mathbb{E}_{t-1}[\|u_t\|_2].\end{aligned}$$

Summing both sides for $t = 1, \ldots, T$, using $\alpha_t \leq \frac{\alpha}{\sqrt{1-\beta_2}}$, and from the definition of l_2 norm and taking the overall expectation yields that

$$\begin{aligned}\mathbb{E}[F(x_T)] \leq F(x_0) - \frac{\alpha}{2R}\sum_{t=0}^{T-1}\mathbb{E}\|\nabla F(x_t)\|^2 + \left(\frac{2\alpha R}{\sqrt{1-\beta_2}} + \frac{\alpha^2 L}{2(1-\beta_2)}\right)\sum_{t=0}^{T-1}\mathbb{E}[\|u_t\|^2] \\ + \frac{\alpha\sqrt{d}\Delta L}{2\sqrt{1-\beta_2}}\sum_{t=0}^{T-1}\mathbb{E}[\sqrt{\|u_t\|^2}].\end{aligned}$$

Equivalently, we can write,

$$\frac{\alpha}{2R}\sum_{t=0}^{T-1}\mathbb{E}\|\nabla F(x_t)\|^2 \leq F(x_0) - \mathbb{E}[F(x_T)] + \left(\frac{2\alpha R}{\sqrt{1-\beta_2}} + \frac{\alpha^2 L}{2(1-\beta_2)}\right)\sum_{t=0}^{T-1}\mathbb{E}[\|u_t\|^2] \tag{F.12}$$

$$+ \frac{\alpha\sqrt{d}\Delta L}{2\sqrt{1-\beta_2}}\sum_{t=0}^{T-1}\mathbb{E}[\sqrt{\|u_t\|^2}]. \tag{F.13}$$

Now, we would like to bound the last two terms on the right hand side. For this, we extend Lemma 5.2 in [39].

Lemma 27. *Define $b_t = \sum_{j=1}^{t} \beta_2^{t-j} a_j$ for $t > 0$ and $b_0 = 0$, we have*

$$\sum_{t=1}^{T}\frac{a_t}{\epsilon + b_t} \leq \ln\left(1 + \frac{R^2}{\epsilon(1-\beta_2)}\right) + T\ln\left(\frac{1}{\beta_2}\right).$$

Proof. Note $b_t > a_t \geq 0$ and we have that $1 - z \leq \exp^{-z}$ with $z = \frac{a_t}{\epsilon + b_t} < 1$. Hence,

$$\frac{a_t}{\epsilon + b_t} \leq \ln(\epsilon + b_t) - \ln(\epsilon + b_t - a_t)$$

$$
\begin{aligned}
&= \ln(\epsilon + b_t) - \ln(\epsilon + \beta_2 b_{t-1}) \\
&= \ln\left(\frac{\epsilon + b_t}{\epsilon + b_{t-1}}\right) + \ln\left(\frac{\epsilon + b_{t-1}}{\epsilon + \beta_2 b_{t-1}}\right) \\
&\leq \ln\left(\frac{\epsilon + b_t}{\epsilon + b_{t-1}}\right) + \ln\left(\max\{1, \frac{1}{\beta_2}\}\right) \\
&= \ln\left(\frac{\epsilon + b_t}{\epsilon + b_{t-1}}\right) + \ln\left(\frac{1}{\beta_2}\right),
\end{aligned}
$$

where the first equality is due to definition of b_t. Summing the inequality above for $t = 1, 2, \ldots, T$, yields that (recall that $b_0 = 0$)

$$
\sum_{t=1}^{T} \frac{a_t}{\epsilon + b_t} \leq \ln\left(1 + \frac{b_T}{\epsilon}\right) + T \ln\left(\frac{1}{\beta_2}\right)
$$

Furthermore, we have

$$
\begin{aligned}
\sum_{t=1}^{T} \sqrt{\frac{a_t}{\epsilon + b_t}} &\leq \sqrt{T} \sqrt{\sum_{t=1}^{T} \frac{a_t}{\epsilon + b_t}} \\
&\leq \sqrt{T} \sqrt{\ln\left(1 + \frac{b_T}{\epsilon}\right) + T \ln\left(\frac{1}{\beta_2}\right)} \\
&\leq \sqrt{T \ln\left(1 + \frac{b_T}{\epsilon}\right)} + T \sqrt{\ln\left(\frac{1}{\beta_2}\right)}
\end{aligned}
$$

Also note that $b_T \leq \frac{R^2}{1-\beta_2}$. □

Using Lemma 27 in (F.13), i.e., $a_t = \nabla_i f_t(x_t)^2$ and $b_t = v_{t,i} = \sum_{j=1}^{t} \beta_2^{t-j} a_j$, and dividing both sides by T, and after some algebra we have the result.

F.2.2 Proof sketch with quantized second order moment.

For theorem 15 analysis we considered that the only quantized part is the model weights x_t, which was shown to be the most important to affect the empirical performance from the ablation studies in appendix F.3. Now we present a proof sketch in the case where second

order moment v_t is also quantized. Let δ_t denote the quantization error in computation of v_t. That is, $v_{t,i} = \beta_2 v_{t-1,i} + \nabla_i f_t(x_{t-1})^2 + \delta_t$. We would like to examine what happens to main lemmas under this error. For lemma 26, there is no changes in the proof until (F.9). For (F.9) we have $\epsilon + \tilde{v} \geq \mathbb{E}_{t-1}[g^2] + \delta^t$, which modifies the R in (F.9) to $R + \delta$ where we assume δ is an upper bound on $\|\delta_t\|_2$ which depends on quantization resolution. We have the identical effect on multiplicative term R in (F.10). For (F.11), we have again loosening effect, hence, $\sqrt{\epsilon + \tilde{v}_{t,i}} \leq \frac{R+\delta}{\sqrt{1-\beta_2}}$. We see that the aggregated effect of δ_ts are modifying R constants with $R + \delta$ until lemma 27. Looking at lemma 27, the perturbation will require us to have a term $\hat{b}_t = b_t + \sum_{j=1}^{t} \delta_t$; for the lemma to hold we need $\hat{b}_t - \sum_{j=1}^{t} \delta_t \geq 0$ which will impose an upper bound on the δ_t (hence, on resolution of quantization for this step) in terms of v_t, i.e. R such that it's magnitude cannot dominate b_t. Assuming such upper bound exists and continuing with lemma we will again end up modifying the R^2 term to $(R + \delta)^2$.

F.2.3 Proof of theorem 16

Proof. The analysis is similar to stochastic rounding except r is not random and $\|r_t\|_2 \leq \sqrt{d}\Delta$.

From the smoothness of the objective function F, we can use the Descent Lemma to obtain that

$$\begin{aligned} F(x_t) &\leq F(x_{t-1}) - \alpha_t \nabla F(x_{t-1})^\top \tilde{u}_t + \frac{\alpha^2 L}{2} \|\tilde{u}_t\|^2 \\ &= F(x_{t-1}) - \alpha_t \nabla F(x_{t-1})^\top \left(u_t + \frac{r_t}{\alpha_t} \right) + \frac{\alpha_t^2 L}{2} \left\| u_t + \frac{r_t}{\alpha_t} \right\|^2 \end{aligned}$$

Taking the expectation of both sides with respect to sampling randomness (conditioned on $f_1, \ldots, f_{t-1}$), we have:

$$\begin{aligned} \mathbb{E}_{t-1} F(x_t) &\leq F(x_{t-1}) - \alpha_t \sum_{i \in [d]} \mathbb{E}_{t-1} \left[\nabla_i F(x_{t-1}) \frac{\nabla_i f_t(x_{t-1})}{\sqrt{\epsilon + v_{t,i}}} \right] - \nabla F(x_{t-1})^\top r_t + \frac{\alpha_t^2 L}{2} \mathbb{E}_{t-1} \left\| u_t + \frac{r_t}{\alpha_t} \right\|^2 \\ &= F(x_{t-1}) - \alpha_t \sum_{i \in [d]} \mathbb{E}_{t-1} \left[\nabla_i F(x_{t-1}) \frac{\nabla_i f_t(x_{t-1})}{\sqrt{\epsilon + v_{t,i}}} \right] - \nabla F(x_{t-1})^\top r_t + \frac{\alpha_t^2 L}{2} \mathbb{E}_{t-1} \|u_t\|^2 \end{aligned}$$

$$
\begin{aligned}
&+ \alpha_t L \mathbb{E}_{t-1}[u_t^\top r_t] + \frac{L}{2}\|r_t\|^2 \\
&\leq F(x_{t-1}) - \alpha_t \sum_{i\in[d]} \mathbb{E}_{t-1}\left[\nabla_i F(x_{t-1}) \frac{\nabla_i f_t(x_{t-1})}{\sqrt{\epsilon + v_{t,i}}}\right] - \nabla F(x_{t-1})^\top r_t + \frac{\alpha_t^2 L}{2}\mathbb{E}_{t-1}\|u_t\|^2 \\
&+ \alpha_t L \mathbb{E}_{t-1}[\|u_t\|_2]\|r_t\| + \frac{L}{2}\|r_t\|^2 \\
&\leq F(x_{t-1}) - \alpha_t \sum_{i\in[d]} \mathbb{E}_{t-1}\left[\nabla_i F(x_{t-1}) \frac{\nabla_i f_t(x_{t-1})}{\sqrt{\epsilon + v_{t,i}}}\right] + \|\nabla F(x_{t-1})\|\|r_t\| + \frac{\alpha_t^2 L}{2}\mathbb{E}_{t-1}\|u_t\|^2 \\
&+ \alpha_t L \mathbb{E}_{t-1}[\|u_t\|_2]\|r_t\| + \frac{L}{2}\|r_t\|^2.
\end{aligned}
$$

Using the fact that $\|r_t\| \leq \sqrt{d}\Delta$ and l_∞ bound assumption, we have:

$$
\mathbb{E}_{t-1}F(x_t) \leq F(x_{t-1}) - \alpha_t \sum_{i\in[d]} \mathbb{E}_{t-1}\left[\nabla_i F(x_{t-1}) \frac{\nabla_i f_t(x_{t-1})}{\sqrt{\epsilon + v_{t,i}}}\right] + Rd\Delta \tag{F.14}
$$

$$
+ \alpha_t^2 L \mathbb{E}_{t-1}\|u_t\|^2 + \alpha_t L\sqrt{d}\Delta \mathbb{E}_{t-1}\|u_t\|_2 + Ld\Delta^2. \tag{F.15}
$$

Remark that the last term in (F.14) is due to variance of the nearest rounding and other error terms are due to bias. We observe that although variance is low ($\mathcal{O}(\Delta^2)$ dependence) the bias terms result in higher error compared to the proof with SR. Continuing as in proof of theorem 15 we obtain the end result.

□

F.3 Additional Details and Results for Experiments

F.3.1 Training hyper-parameters and details.

For all our experiments we use AWS *p4d.24xlarge* instances each with 8 A100-40GB GPUs. For GPT-2 (350M, 770M) we train, on a single node, for 49B and 46B tokens with 12 and 7 micro batch sizes and 480 and 448 global batch sizes respectively. For GPT-Neo (1.3B), we train for 40B tokens with 1024 global batch size and 16 micro batch sizes on 4 nodes;

for GPT-Neo (2.7B, 6.7B) we train for 20B tokens with 512 global batch size and 8 micro batch sizes on 8 nodes. For GPT-2 we do not use tensor parallelism (TP), and for GPT-Neo models we use TP=8.

Tuning the learning rate. For GPT-2 models we use 2000 warm-up iterations and for GPT-Neo 200 iterations as recommended. We utilize cosine scheduling. We tune the maximum learning rate individually for each method starting from the recommended/default learning rates, we leave the minimum learning rate as default. For GPT-2 models we search the learning rate in $\{2, 3, 4, 5, 6, 7, 8\} \times 10^{-4}$. For GPT-Neo (1.2B,2.7B) we search in $\{1.6, 3.2, 4, 5\} \times 10^{-4}$, and GPT-Neo (6.7B) $\{1.2, 3.6, 5.5\} \times 10^{-4}$.

F.3.2 Ablation studies.

Toy example in section 7.2.2

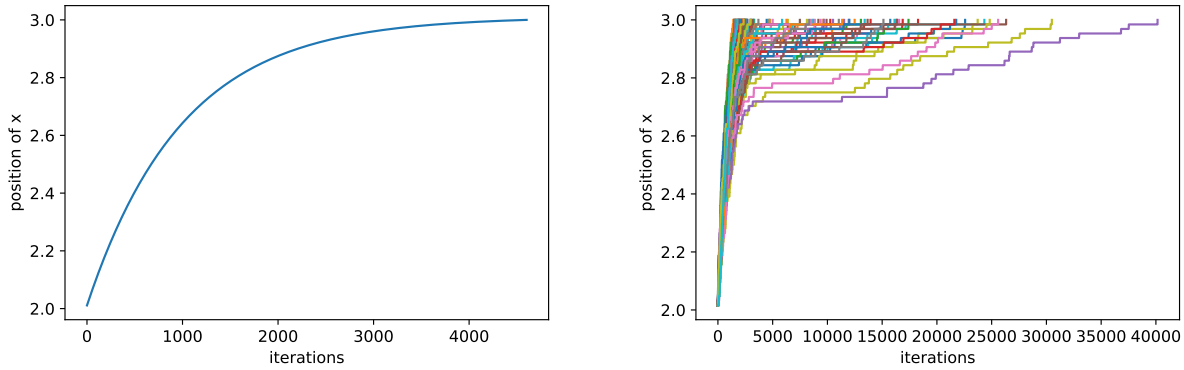


Figure F.1 Position of x vs. number of iterations for FP32 updates (Left) and BF16+SR updates (Right), each position curve represents the position vs iterations for an individual run. With the decaying updates SR can take a longer time to converge, here FP32 converges in 4600 steps and SR converges in on average 7700 (over 100 runs). We observe that an individual run with SR can converge in as much as 40,000 iterations.

Effect of full precision optimizer states and gradients.

In fig. F.2 we ablate the precisions of optimizer states, gradients and the update. In particular, every variable is in BF16 in our experiments for algorithm 8; here, the ablation

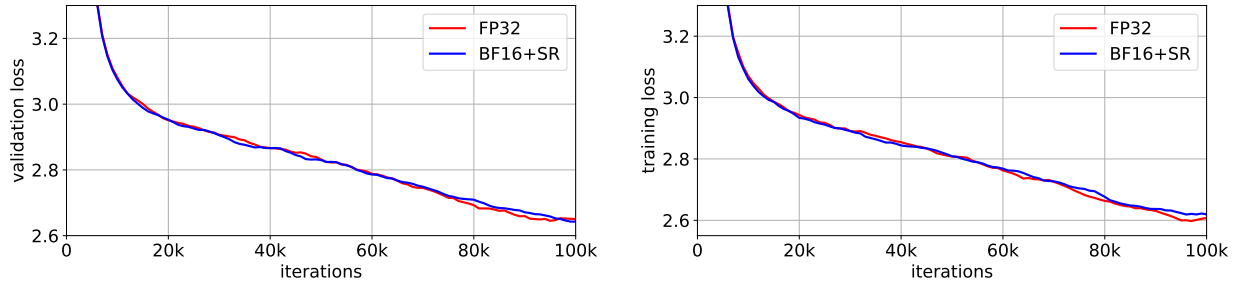


Figure F.2 Training and validation loss curves comparing FP32 training and BF16+SR strategy while training GPT-2 (350M).

indicates that the effect of keeping optimizer states and gradients in higher precision does not affect the performance significantly.

F.3.3 Loss curves.

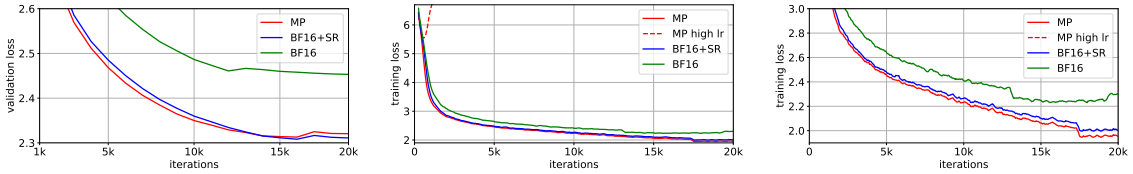


Figure F.3 Validation (left) and training losses (middle–zoomed out, right–zoomed in) for training GPT-Neo (6.7B).

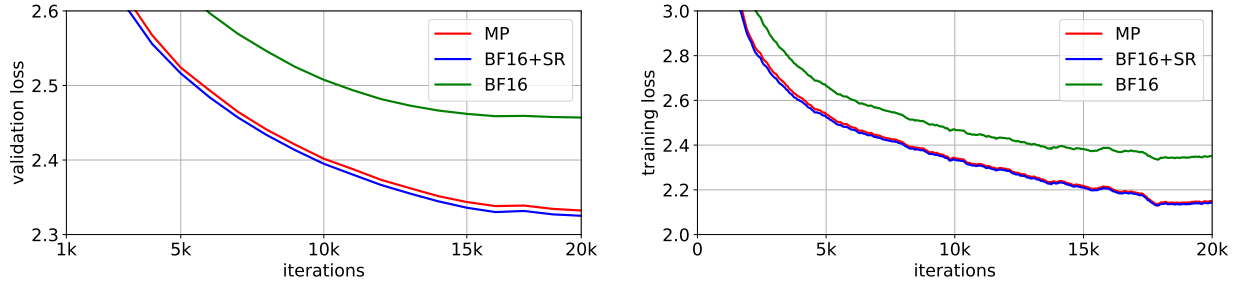


Figure F.4 Validation (left) and training losses (right) for training GPT-Neo (2.7B).

The convergence plots show BF16+SR training can converge to a better local minimum in terms of validation perplexity. Note that BF16+SR further gain can be provided when we

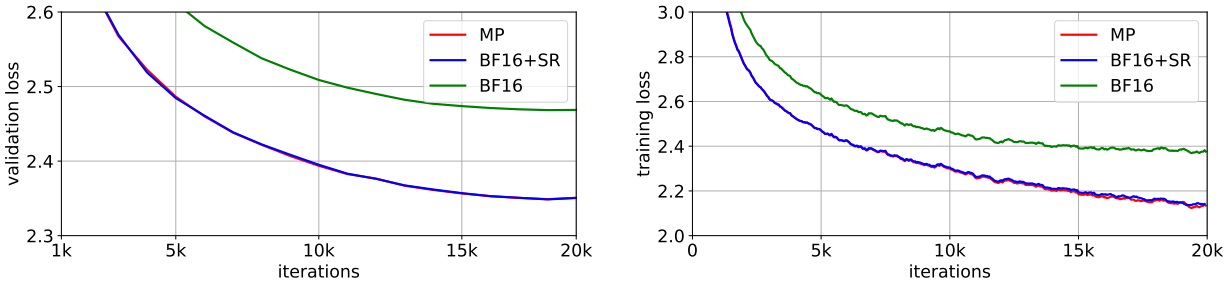


Figure F.5 Validation (left) and training losses (right) for training GPT-Neo (1.3B).

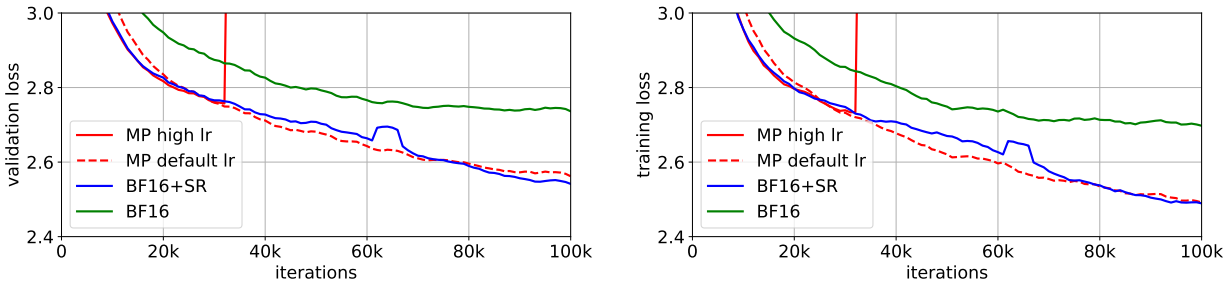


Figure F.6 Validation (left) and training losses (right) for training GPT-2 (770M).

equate the wall clock running times.

REFERENCES

[1] Durmus Alp Emre Acar, Yue Zhao, Ruizhao Zhu, Ramon Matas, Matthew Mattina, Paul Whatmough, and Venkatesh Saligrama. Debiasing model updates for improving personalized federated training. In *International Conference on Machine Learning*, pages 21–31. PMLR, 2021.

[2] Sara Ahmadian, Ashkan Norouzi-Fard, Ola Svensson, and Justin Ward. Better guarantees for k-means and euclidean k-median by primal-dual algorithms. *SIAM Journal on Computing*, 49(4):FOCS17–97, 2019.

[3] Dan Alistarh, Demjan Grubic, Jerry Li, Ryota Tomioka, and Milan Vojnovic. Qsgd: Communication-efficient sgd via gradient quantization and encoding. *Advances in Neural Information Processing Systems*, 30, 2017.

[4] Diogo Almeida, Clemens Winter, Jie Tang, and Wojciech Zaremba. A generalizable approach to learning optimizers. *arXiv preprint arXiv:2106.00958*, 2021.

[5] Luís B. Almeida, Thibault Langlois, Jose D. Amaral, and Alexander Plakhov. *Parameter Adaptation in Stochastic Optimization*, page 111–134. Publications of the Newton Institute. Cambridge University Press, 1999.

[6] Marcin Andrychowicz, Misha Denil, Sergio Gomez, Matthew W Hoffman, David Pfau, Tom Schaul, Brendan Shillingford, and Nando De Freitas. Learning to learn by gradient descent by gradient descent. *Advances in neural information processing systems*, 29, 2016.

[7] Yu Bai, Yu-Xiang Wang, and Edo Liberty. Proxquant: Quantized neural networks via proximal operators. In *International Conference on Learning Representations*, 2019.

[8] Borja Balle, Gilles Barthe, Marco Gaboardi, Justin Hsu, and Tetsuya Sato. Hypothesis testing interpretations and renyi differential privacy. In Silvia Chiappa and Roberto Calandra, editors, *International Conference on Artificial Intelligence and Statistics (AISTATS)*, volume 108 of *Proceedings of Machine Learning Research*, pages 2496–2506. PMLR, 2020.

[9] Leighton Pate Barnes, Yanjun Han, and Ayfer Ozgur. Lower bounds for learning distributions under communication constraints via fisher information. *Journal of Machine Learning Research*, 21(236):1–30, 2020.

[10] David GT Barrett and Benoit Dherin. Implicit gradient regularization. *arXiv preprint arXiv:2009.11162*, 2020.

[11] Antje Barth. Amazon ec2 trn1 instances for high-performance model training are now available, 2022.

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

[13] Atilim Gunes Baydin, Robert Cornish, David Martinez Rubio, Mark Schmidt, and Frank Wood. Online learning rate adaptation with hypergradient descent. In *International Conference on Learning Representations*, 2018.

[14] Jeremy Bernstein, Yu-Xiang Wang, Kamyar Azizzadenesheli, and Animashree Anandkumar. signsgd: Compressed optimisation for non-convex problems. In *International Conference on Machine Learning*, pages 560–569. PMLR, 2018.

[15] Mikołaj Bińkowski, Dougal J. Sutherland, Michael Arbel, and Arthur Gretton. Demystifying MMD GANs. In *International Conference on Learning Representations*, 2018.

[16] Sid Black, Leo Gao, Phil Wang, Connor Leahy, and Stella Biderman. GPT-Neo: Large Scale Autoregressive Language Modeling with Mesh-Tensorflow, March 2021.

[17] David M. Blei, Michael I. Jordan, and A. Ng. Hierarchical bayesian models for applications in information retrieval. 2003.

[18] Jérôme Bolte, Shoham Sabach, and Marc Teboulle. Proximal alternating linearized minimization for nonconvex and nonsmooth problems. *Math. Program.*, 146(1-2):459–494, 2014.

[19] Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *Advances in neural information processing systems*, 33:1877–1901, 2020.

[20] Mark Bun, Cynthia Dwork, Guy N. Rothblum, and Thomas Steinke. Composable and versatile privacy via truncated CDP. In *ACM SIGACT Symposium on Theory of Computing (STOC)*, pages 74–86, 2018.

[21] Sebastian Caldas, Sai Meher Karthik Duddu, Peter Wu, Tian Li, Jakub Konečnỳ, H Brendan McMahan, Virginia Smith, and Ameet Talwalkar. Leaf: A benchmark for federated settings. *arXiv preprint arXiv:1812.01097*, 2018.

[22] Clément L. Canonne, Gautam Kamath, and Thomas Steinke. The discrete gaussian for differential privacy. In *Neural Information Processing Systems (NeurIPS)*, 2020.

[23] Xingjian Cao, Gang Sun, Hongfang Yu, and Mohsen Guizani. Perfed-gan: Personalized federated learning via generative adversarial networks. *IEEE Internet of Things Journal*, 10(5):3749–3762, 2023.

[24] Matias D. Cattaneo, Jason Matthew Klusowski, and Boris Shigida. On the implicit bias of Adam. In Ruslan Salakhutdinov, Zico Kolter, Katherine Heller, Adrian Weller, Nuria Oliver, Jonathan Scarlett, and Felix Berkenkamp, editors, *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pages 5862–5906. PMLR, 21–27 Jul 2024.

[25] Kartik Chandra, Audrey Xie, Jonathan Ragan-Kelley, and Erik Meijer. Gradient descent: The ultimate optimizer. In Alice H. Oh, Alekh Agarwal, Danielle Belgrave, and Kyunghyun Cho, editors, *Advances in Neural Information Processing Systems*, 2022.

[26] Congliang Chen, Li Shen, Haozhi Huang, and Wei Liu. Quantized adam with error feedback, 2021.

[27] Huili Chen, Jie Ding, Eric Tramel, Shuang Wu, Anit Kumar Sahu, Salman Avestimehr, and Tao Zhang. Self-aware personalized federated learning. *CoRR*, abs/2204.08069, 2022.

[28] Huili Chen, Jie Ding, Eric William Tramel, Shuang Wu, Anit Kumar Sahu, Salman Avestimehr, and Tao Zhang. Self-aware personalized federated learning. In Alice H. Oh, Alekh Agarwal, Danielle Belgrave, and Kyunghyun Cho, editors, *Advances in Neural Information Processing Systems*, 2022.

[29] Jinghui Chen, Dongruo Zhou, Yiqi Tang, Ziyan Yang, Yuan Cao, and Quanquan Gu. Closing the generalization gap of adaptive gradient methods in training deep neural networks, 2020.

[30] Shixiang Chen, Alfredo Garcia, Mingyi Hong, and Shahin Shahrampour. Decentralized riemannian gradient descent on the stiefel manifold. In *International Conference on Machine Learning*, pages 1594–1605. PMLR, 2021.

[31] Sitan Chen, Sinho Chewi, Jerry Li, Yuanzhi Li, Adil Salim, and Anru Zhang. Sampling is as easy as learning the score: theory for diffusion models with minimal data assumptions. In *The Eleventh International Conference on Learning Representations*, 2023.

[32] Sitan Chen, Vasilis Kontonis, and Kulin Shah. Learning general gaussian mixtures with efficient score matching, 2024.

[33] Xiangning Chen, Chen Liang, Da Huang, Esteban Real, Kaiyuan Wang, Yao Liu, Hieu Pham, Xuanyi Dong, Thang Luong, Cho-Jui Hsieh, Yifeng Lu, and Quoc V. Le. Symbolic discovery of optimization algorithms, 2023.

[34] Liam Collins, Hamed Hassani, Aryan Mokhtari, and Sanjay Shakkottai. Exploiting shared representations for personalized federated learning. In Marina Meila and Tong Zhang, editors, *International Conference on Machine Learning (ICML)*, volume 139 of *Proceedings of Machine Learning Research*, pages 2089–2099. PMLR, 2021.

[35] Michael P. Connolly, Nicholas J. Higham, and Theo Mary. Stochastic rounding and its probabilistic backward error analysis. *SIAM Journal on Scientific Computing*, 43(1):A566–A585, 2021.

[36] Matteo Croci, Massimiliano Fasi, Nicholas J Higham, Theo Mary, and Mantas Mikaitis. Stochastic rounding: implementation, error analysis and applications. *Royal Society Open Science*, 9(3):211631, 2022.

[37] Constantinos Daskalakis, Christos Tzamos, and Manolis Zampetakis. Ten steps of em suffice for mixtures of two gaussians. In *Proceedings of the 2017 Conference on Learning Theory*, volume 65 of *Proceedings of Machine Learning Research*, pages 704–710, 2017.

[38] Hassan Dbouk, Hetul Sanghvi, Mahesh Mehendale, and Naresh Shanbhag. Dbq: A differentiable branch quantizer for lightweight deep neural networks. In *European Conference on Computer Vision*, pages 90–106. Springer, 2020.

[39] Alexandre Défossez, Leon Bottou, Francis Bach, and Nicolas Usunier. A simple convergence proof of adam and adagrad. *Transactions on Machine Learning Research*, 2022.

[40] Yuyang Deng, Mohammad Mahdi Kamani, and Mehrdad Mahdavi. Adaptive personalized federated learning. *arXiv preprint arXiv:2003.13461*, 2020.

[41] Prafulla Dhariwal and Alexander Quinn Nichol. Diffusion models beat GANs on image synthesis. In A. Beygelzimer, Y. Dauphin, P. Liang, and J. Wortman Vaughan, editors, *Advances in Neural Information Processing Systems*, 2021.

[42] Canh T. Dinh, Nguyen H. Tran, and Tuan Dung Nguyen. Personalized federated learning with moreau envelopes. In *Advances in Neural Information Processing Systems*, 2020.

[43] Timothy Dozat. Incorporating nesterov momentum into adam. 2016.

[44] Simon Shaolei Du, Wei Hu, Sham M. Kakade, Jason D. Lee, and Qi Lei. Few-shot learning via learning the representation, provably. In *International Conference on Learning Representations*, 2021.

[45] Cynthia Dwork, Frank McSherry, Kobbi Nissim, and Adam D. Smith. Calibrating noise to sensitivity in private data analysis. In *Theory of Cryptography Conference (TCC)*, pages 265–284, 2006.

[46] Cynthia Dwork and Aaron Roth. The algorithmic foundations of differential privacy. *Foundations and Trends in Theoretical Computer Science*, 9(3-4):211–407, 2014.

[47] Bradley Efron. *Large-Scale Inference: Empirical Bayes Methods for Estimation, Testing, and Prediction*. Institute of Mathematical Statistics Monographs. Cambridge University Press, 2010.

[48] Alireza Fallah, Aryan Mokhtari, and Asuman Ozdaglar. Personalized federated learning: A meta-learning approach. In *Advances in Neural Information Processing Systems*, 2020.

[49] Haozheng Fan, Hao Zhou, Guangtai Huang, Parameswaran Raman, Xinwei Fu, Gaurav Gupta, Dhananjay Ram, Yida Wang, and Jun Huan. Hlat: High-quality large language model pre-trained on aws trainium. *arXiv preprint arXiv:2404.10630*, 2024.

[50] Pierre Foret, Ariel Kleiner, Hossein Mobahi, and Behnam Neyshabur. Sharpness-aware minimization for efficiently improving generalization. In *International Conference on Learning Representations*, 2021.

[51] Luca Franceschi, Michele Donini, Paolo Frasconi, and Massimiliano Pontil. Forward and reverse gradient-based hyperparameter optimization. In *International Conference on Machine Learning*, pages 1165–1173. PMLR, 2017.

[52] Andrew Gelman, John B. Carlin, Hal S. Stern, David B. Dunson, Aki Vehtari, and Donald B. Rubin. *Bayesian Data Analysis*. Chapman and Hall/CRC, 2013.

[53] Robin C Geyer, Tassilo Klein, and Moin Nabi. Differentially private federated learning: A client level perspective. *arXiv preprint arXiv:1712.07557*, 2017.

[54] Badih Ghazi, Ravi Kumar, and Pasin Manurangsi. Differentially private clustering: Tight approximation ratios. *Advances in Neural Information Processing Systems*, 33:4040–4054, 2020.

[55] Badih Ghazi, Ravi Kumar, and Pasin Manurangsi. User-level differentially private learning via correlated sampling. In *Neural Information Processing Systems (NeurIPS)*, pages 20172–20184, 2021.

[56] Avishek Ghosh, Jichan Chung, Dong Yin, and Kannan Ramchandran. An efficient framework for clustered federated learning. In *Advances in Neural Information Processing Systems*, 2020.

[57] Richard Gill and Boris Levit. Applications of the van trees inequality: A bayesian cramér-rao bound. *Bernoulli*, 1:59–79, 03 1995.

[58] Antonious M. Girgis, Deepesh Data, and Suhas Diggavi. Distributed user-level private mean estimation. In *2022 IEEE International Symposium on Information Theory (ISIT)*, pages 2196–2201, 2022.

[59] Antonious M Girgis, Deepesh Data, Suhas Diggavi, Peter Kairouz, and Ananda Theertha Suresh. Shuffled model of federated learning: Privacy, accuracy and communication trade-offs. *IEEE Journal on Selected Areas in Information Theory*, 2(1):464–478, 2021.

[60] Antonious M. Girgis, Deepesh Data, Suhas N. Diggavi, Peter Kairouz, and Ananda Theertha Suresh. Shuffled model of differential privacy in federated learning. In *International Conference on Artificial Intelligence and Statistics (AISTATS)*, volume 130 of *Proceedings of Machine Learning Research*, pages 2521–2529. PMLR, 2021.

[61] Aaron Gokaslan and Vanya Cohen. Openwebtext corpus. http://Skylion007.github.io/OpenWebTextCorpus, 2019.

[62] Ruihao Gong, Xianglong Liu, Shenghu Jiang, Tianxiang Li, Peng Hu, Jiazhen Lin, Fengwei Yu, and Junjie Yan. Differentiable soft quantization: Bridging full-precision and low-bit neural networks. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 4852–4861, 2019.

[63] Suyog Gupta, Ankur Agrawal, Kailash Gopalakrishnan, and Pritish Narayanan. Deep learning with limited numerical precision. In *International conference on machine learning*, pages 1737–1746. PMLR, 2015.

[64] Filip Hanzely, Slavomír Hanzely, Samuel Horváth, and Peter Richtárik. Lower bounds and optimal algorithms for personalized federated learning. In *Advances in Neural Information Processing Systems*, 2020.

[65] Filip Hanzely and Peter Richtárik. Federated learning of a mixture of global and local models. *arXiv preprint arXiv:2002.05516*, 2020.

[66] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

[67] Byeongho Heo, Sanghyuk Chun, Seong Joon Oh, Dongyoon Han, Sangdoo Yun, Gyuwan Kim, Youngjung Uh, and Jung-Woo Ha. Adamp: Slowing down the slowdown for momentum optimizers on scale-invariant weights, 2021.

[68] Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. Gans trained by a two time-scale update rule converge to a local nash equilibrium. *Advances in neural information processing systems*, 30, 2017.

[69] Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, 2015.

[70] Timothy Hospedales, Antreas Antoniou, Paul Micaelli, and Amos Storkey. Meta-learning in neural networks: A survey. *IEEE transactions on pattern analysis and machine intelligence*, 44(9):5149–5169, 2021.

[71] Lu Hou, Quanming Yao, and James T. Kwok. Loss-aware binarization of deep networks. In *International Conference on Learning Representations*, 2017.

[72] Lu Hou, Ruiliang Zhang, and James T. Kwok. Analysis of quantized models. In *International Conference on Learning Representations*, 2019.

[73] Rui Hu, Yuanxiong Guo, Hongning Li, Qingqi Pei, and Yanmin Gong. Personalized federated learning with differential privacy. *IEEE Internet of Things Journal*, 7(10):9530–9539, 2020.

[74] John Immerkær. Fast noise variance estimation. *Computer Vision and Image Understanding*, 64(2):300–302, 1996.

[75] Prateek Jain, John Rush, Adam Smith, Shuang Song, and Abhradeep Guha Thakurta. Differentially private model personalization. In *Advances in Neural Information Processing Systems*, volume 34, 2021.

[76] Prateek Jain, John Rush, Adam Smith, Shuang Song, and Abhradeep Guha Thakurta. Differentially private model personalization. *Advances in Neural Information Processing Systems*, 34, 2021.

[77] William James and Charles Stein. Estimation with quadratic loss. In *Proceedings Berkeley Symposium on Mathematics and Statistics, Vol 1*, pages 361–379. University of California Press, 1961.

[78] Yihan Jiang, Jakub Konečnỳ, Keith Rush, and Sreeram Kannan. Improving federated learning personalization via model agnostic meta learning. *arXiv preprint arXiv:1909.12488*, 2019.

[79] William Kahan. How futile are mindless assessments of roundoff in floating-point computation ?, 2006. Accessed: 2024-10-08.

[80] Peter Kairouz, H Brendan McMahan, Brendan Avent, Aurélien Bellet, Mehdi Bennis, Arjun Nitin Bhagoji, Kallista Bonawitz, Zachary Charles, Graham Cormode, Rachel Cummings, et al. Advances and open problems in federated learning. *Foundations and Trends® in Machine Learning*, 14(1–2):1–210, 2021.

[81] Dhiraj Kalamkar, Dheevatsa Mudigere, Naveen Mellempudi, Dipankar Das, Kunal Banerjee, Sasikanth Avancha, Dharma Teja Vooturi, Nataraj Jammalamadaka, Jianyu Huang, Hector Yuen, Jiyan Yang, Jongsoo Park, Alexander Heinecke, Evangelos Georganas, Sudarshan Srinivasan, Abhisek Kundu, Misha Smelyanskiy, Bharat Kaul, and Pradeep Dubey. A study of bfloat16 for deep learning training, 2019.

[82] Sai Praneeth Karimireddy, Satyen Kale, Mehryar Mohri, Sashank Reddi, Sebastian Stich, and Ananda Theertha Suresh. Scaffold: Stochastic controlled averaging for federated learning. In *International Conference on Machine Learning*, pages 5132–5143. PMLR, 2020.

[83] Sai Praneeth Karimireddy, Quentin Rebjock, Sebastian Stich, and Martin Jaggi. Error feedback fixes signsgd and other gradient compression schemes. In *International Conference on Machine Learning*, pages 3252–3261. PMLR, 2019.

[84] Andrej Karpathy. The unreasonable effectiveness of recurrent neural networks. http://karpathy.github.io/2015/05/21/rnn-effectiveness/, 2015.

[85] Andrej Karpathy. NanoGPT. https://github.com/karpathy/nanoGPT, 2022.

[86] Shiva Prasad Kasiviswanathan, Homin K Lee, Kobbi Nissim, Sofya Raskhodnikova, and Adam Smith. What can we learn privately? *SIAM Journal on Computing*, 40(3):793–826, 2011.

[87] Mikhail Khodak, Maria-Florina F Balcan, and Ameet S Talwalkar. Adaptive gradient-based meta-learning methods. In *Advances in Neural Information Processing Systems*, 2019.

[88] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In Yoshua Bengio and Yann LeCun, editors, *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, 2015.

[89] Diederik P Kingma and Ruiqi Gao. Understanding the diffusion objective as a weighted integral of elbos. In *Neural Information Processing Systems (NeurIPS)*, 2023.

[90] Nikita Yurevich Kotelevskii, Maxime Vono, Alain Durmus, and Eric Moulines. Fedpop: A bayesian approach for personalised federated learning. In Alice H. Oh, Alekh Agarwal, Danielle Belgrave, and Kyunghyun Cho, editors, *Advances in Neural Information Processing Systems*, 2022.

[91] Alex Krizhevsky, Vinod Nair, and Geoffrey Hinton. Cifar-10 (canadian institute for advanced research). 2009.

[92] Jeongyeol Kwon and Constantine Caramanis. The em algorithm gives sample-optimality for learning mixtures of well-separated gaussians, 2020.

[93] Qi Le, Enmao Diao, Xinran Wang, Ali Anwar, Vahid Tarokh, and Jie Ding. Personalized federated recommender systems with private and partially federated autoencoders, 2022.

[94] Cong Leng, Zesheng Dou, Hao Li, Shenghuo Zhu, and Rong Jin. Extremely low bit neural network: Squeeze the last bit out with admm. In *Thirty-Second AAAI Conference on Artificial Intelligence*, 2018.

[95] Daniel Levy, Ziteng Sun, Kareem Amin, Satyen Kale, Alex Kulesza, Mehryar Mohri, and Ananda Theertha Suresh. Learning with user-level privacy. In *Neural Information Processing Systems (NeurIPS)*, pages 12466–12479, 2021.

[96] Daliang Li and Junpu Wang. Fedmd: Heterogenous federated learning via model distillation. *arXiv preprint arXiv:1910.03581*, 2019.

[97] Hao Li, Soham De, Zheng Xu, Christoph Studer, Hanan Samet, and Tom Goldstein. Training quantized nets: A deeper understanding. *Advances in Neural Information Processing Systems*, 30, 2017.

[98] Hao Li, Soham De, Zheng Xu, Christoph Studer, Hanan Samet, and Tom Goldstein. Training quantized nets: A deeper understanding. In I. Guyon, U. Von Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 30. Curran Associates, Inc., 2017.

[99] Jeffrey Li, Mikhail Khodak, Sebastian Caldas, and Ameet Talwalkar. Differentially private meta-learning. In *International Conference on Learning Representations*, 2020.

[100] Tian Li, Shengyuan Hu, Ahmad Beirami, and Virginia Smith. Ditto: Fair and robust federated learning through personalization. In *International Conference on Machine Learning*, pages 6357–6368. PMLR, 2021.

[101] Tian Li, Anit Kumar Sahu, Manzil Zaheer, Maziar Sanjabi, Ameet Talwalkar, and Virginia Smith. Federated optimization in heterogeneous networks. In *Proceedings of Machine Learning and Systems 2020, MLSys*, 2020.

[102] Tao Lin, Lingjing Kong, Sebastian U. Stich, and Martin Jaggi. Ensemble distillation for robust model fusion in federated learning. In *Advances in Neural Information Processing Systems*, 2020.

[103] Hong Liu, Zhiyuan Li, David Hall, Percy Liang, and Tengyu Ma. Sophia: A scalable stochastic second-order optimizer for language model pre-training, 2023.

[104] Liyuan Liu, Haoming Jiang, Pengcheng He, Weizhu Chen, Xiaodong Liu, Jianfeng Gao, and Jiawei Han. On the variance of the adaptive learning rate and beyond. In *International Conference on Learning Representations*, 2020.

[105] Yuhan Liu, Ananda Theertha Suresh, Felix X. Yu, Sanjiv Kumar, and Michael Riley. Learning discrete distributions: user vs item-level privacy. In *Neural Information Processing Systems (NeurIPS)*, 2020.

[106] Ziwei Liu, Ping Luo, Xiaogang Wang, and Xiaoou Tang. Deep learning face attributes in the wild. In *Proceedings of International Conference on Computer Vision (ICCV)*, December 2015.

[107] Stuart Lloyd. Least squares quantization in pcm. *IEEE transactions on information theory*, 28(2):129–137, 1982.

[108] Frederic M. Lord. Estimating true-score distributions in psychological testing (an empirical bayes estimation problem)*. *ETS Research Bulletin Series*, 1967(2):i–51, 1967.

[109] Ilya Loshchilov and Frank Hutter. Decoupled weight decay regularization. In *International Conference on Learning Representations*, 2019.

[110] Christos Louizos, Matthias Reisser, Tijmen Blankevoort, Efstratios Gavves, and Max Welling. Relaxed quantization for discretized neural networks. In *International Conference on Learning Representations*, 2019.

[111] Liangchen Luo, Yuanhao Xiong, and Yan Liu. Adaptive gradient methods with dynamic bound of learning rate. In *International Conference on Learning Representations*, 2019.

[112] Jian Ma, Junhao Liang, Chen Chen, and Haonan Lu. Subject-diffusion: Open domain personalized text-to-image generation without test-time fine-tuning. *arXiv preprint arXiv:2307.11410*, 2023.

[113] Dougal Maclaurin, David Duvenaud, and Ryan Adams. Gradient-based hyperparameter optimization through reversible learning. In *International conference on machine learning*, pages 2113–2122. PMLR, 2015.

[114] Yishay Mansour, Mehryar Mohri, Jae Ro, and Ananda Theertha Suresh. Three approaches for personalization with applications to federated learning. *arXiv preprint arXiv:2002.10619*, 2020.

[115] Othmane Marfoq, Giovanni Neglia, Aurélien Bellet, Laetitia Kameni, and Richard Vidal. Federated multi-task learning under a mixture of distributions. *Advances in Neural Information Processing Systems*, 34, 2021.

[116] Brendan McMahan, Eider Moore, Daniel Ramage, Seth Hampson, and Blaise Aguera y Arcas. Communication-efficient learning of deep networks from decentralized data. In *Artificial Intelligence and Statistics*, pages 1273–1282. PMLR, 2017.

[117] Stephen Merity, Caiming Xiong, James Bradbury, and Richard Socher. Pointer sentinel mixture models, 2016.

[118] Luke Metz, James Harrison, C Daniel Freeman, Amil Merchant, Lucas Beyer, James Bradbury, Naman Agrawal, Ben Poole, Igor Mordatch, Adam Roberts, et al. Velo: Training versatile learned optimizers by scaling up. *arXiv preprint arXiv:2211.09760*, 2022.

[119] Luke Metz, Niru Maheswaranathan, C Daniel Freeman, Ben Poole, and Jascha Sohl-Dickstein. Tasks, stability, architecture, and compute: Training more effective learned optimizers, and using them to train themselves. *arXiv preprint arXiv:2009.11243*, 2020.

[120] Paulius Micikevicius, Sharan Narang, Jonah Alben, Gregory Diamos, Erich Elsen, David Garcia, Boris Ginsburg, Michael Houston, Oleksii Kuchaiev, Ganesh Venkatesh, and Hao Wu. Mixed precision training, 2018.

[121] Ilya Mironov. Rényi differential privacy. In *IEEE Computer Security Foundations Symposium (CSF)*, pages 263–275, 2017.

[122] Ilya Mironov, Kunal Talwar, and Li Zhang. Rényi differential privacy of the sampled gaussian mechanism. *CoRR*, abs/1908.10530, 2019.

[123] Igor Molybog, Peter Albert, Moya Chen, Zachary DeVito, David Esiobu, Naman Goyal, Punit Singh Koura, Sharan Narang, Andrew Poulton, Ruan Silva, Binh Tang, Diana Liskovich, Puxin Xu, Yuchen Zhang, Melanie Kambadur, Stephen Roller, and Susan Zhang. A theory on adam instability in large-scale machine learning, 2023.

[124] Taehong Moon, Moonseok Choi, Gayoung Lee, Jung-Woo Ha, and Juho Lee. Fine-tuning diffusion models with limited data. In *NeurIPS 2022 Workshop on Score-Based Methods*, 2022.

[125] Aashiq Muhamed, Christian Bock, Rahul Solanki, Youngsuk Park, Yida Wang, and Jun Huan. Training large-scale foundation models on emerging ai chips. In *Proceedings of the 29th ACM SIGKDD Conference on Knowledge Discovery and Data Mining*, pages 5821–5822, 2023.

[126] Kaan Ozkara, Antonious Girgis, Deepesh Data, and Suhas Diggavi. A statistical framework for personalized federated learning and estimation: Theory, algorithms, and privacy. In *International Conference on Learning Representations*, 2023.

[127] Kaan Ozkara, Antonious M Girgis, Deepesh Data, and Suhas Diggavi. A generative framework for personalized learning and estimation: Theory, algorithms, and privacy. *arXiv preprint arXiv:2207.01771*, 2022, July.

[128] Kaan Ozkara, Bruce Huang, and Suhas Diggavi. Personalized PCA for federated heterogeneous data. In *2023 IEEE International Symposium on Information Theory (ISIT)*, pages 168–173. IEEE, 2023.

[129] Kaan Ozkara, Bruce Huang, Ruida Zhou, and Suhas Diggavi. ADEPT: Hierarchical bayes approach to personalized federated unsupervised learning. In *The 28th International Conference on Artificial Intelligence and Statistics*, 2025.

[130] Kaan Ozkara, Can Karakus, Parameswaran Raman, Mingyi Hong, Shoham Sabach, Branislav Kveton, and Volkan Cevher. MADA: Meta-adaptive optimizers through hyper-gradient descent. In *Proceedings of the 41st International Conference on Machine Learning*, volume 235 of *Proceedings of Machine Learning Research*, pages 38983–39008. PMLR, 21–27 Jul 2024.

[131] Kaan Ozkara, Navjot Singh, Deepesh Data, and Suhas Diggavi. Quped: Quantized personalization via distillation with applications to federated learning. *Advances in Neural Information Processing Systems*, 34, 2021.

[132] Adam Paszke, Sam Gross, Soumith Chintala, Gregory Chanan, Edward Yang, Zachary DeVito, Zeming Lin, Alban Desmaison, Luca Antiga, and Adam Lerer. Automatic differentiation in pytorch. 2017.

[133] Chao Peng, Yiming Guo, Yao Chen, Qilin Rui, Zhengfeng Yang, and Chenyang Xu. Fedgm: Heterogeneous federated learning via generative learning and mutual distillation. In *Euro-Par 2023: Parallel Processing: 29th International Conference on Parallel and Distributed Computing, Limassol, Cyprus, August 28 – September 1, 2023, Proceedings*, page 339–351, Berlin, Heidelberg, 2023. Springer-Verlag.

[134] Mirko Polato. Federated variational autoencoder for collaborative filtering. In *2021 International Joint Conference on Neural Networks (IJCNN)*, pages 1–8, 2021.

[135] Antonio Polino, Razvan Pascanu, and Dan Alistarh. Model compression via distillation and quantization. In *International Conference on Learning Representations*, 2018.

[136] Haotong Qin, Ruihao Gong, Xianglong Liu, Xiao Bai, Jingkuan Song, and Nicu Sebe. Binary neural networks: A survey. *Pattern Recognition*, 105:107281, Sep 2020.

[137] Alec Radford, Jeff Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. Language models are unsupervised multitask learners. 2019.

[138] Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, Ilya Sutskever, et al. Language models are unsupervised multitask learners. *OpenAI blog*, 1(8):9, 2019.

[139] Jack W. Rae et al. Scaling language models: Methods, analysis & insights from training gopher, 2022.

[140] Aniruddh Raghu, Maithra Raghu, Samy Bengio, and Oriol Vinyals. Rapid learning or feature reuse? towards understanding the effectiveness of MAML. In *8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020*. OpenReview.net, 2020.

[141] Samyam Rajbhandari, Jeff Rasley, Olatunji Ruwase, and Yuxiong He. Zero: Memory optimizations toward training trillion parameter models. In *SC20: International Conference for High Performance Computing, Networking, Storage and Analysis*, pages 1–16. IEEE, 2020.

[142] Esteban Real, Chen Liang, David So, and Quoc Le. AutoML-zero: Evolving machine learning algorithms from scratch. In Hal Daumé III and Aarti Singh, editors, *Proceedings of the 37th International Conference on Machine Learning*, volume 119 of *Proceedings of Machine Learning Research*, pages 8007–8019. PMLR, 13–18 Jul 2020.

[143] Sashank J. Reddi, Satyen Kale, and Sanjiv Kumar. On the convergence of adam and beyond. In *International Conference on Learning Representations*, 2018.

[144] Danilo Rezende and Shakir Mohamed. Variational inference with normalizing flows. In *International conference on machine learning*, pages 1530–1538. PMLR, 2015.

[145] Herbert Robbins. An empirical bayes approach to statistics. In *The Proceedings of Third Berkeley Symposium on Mathematical Statistics and Probability*, pages 157–163, 1956.

[146] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 10684–10695, 2022.

[147] Robin Rombach, Andreas Blattmann, Dominik Lorenz, Patrick Esser, and Björn Ommer. High-resolution image synthesis with latent diffusion models, 2022.

[148] Nataniel Ruiz, Yuanzhen Li, Varun Jampani, Yael Pritch, Michael Rubinstein, and Kfir Aberman. Dreambooth: Fine tuning text-to-image diffusion models for subject-driven generation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 22500–22510, 2023.

[149] Robin M Schmidt, Frank Schneider, and Philipp Hennig. Descending through a crowded valley - benchmarking deep learning optimizers. In Marina Meila and Tong Zhang, editors, *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pages 9367–9376. PMLR, 18–24 Jul 2021.

[150] L. Schuchman. Dither signals and their effect on quantization noise. *IEEE Transactions on Communication Technology*, 12(4):162–165, 1964.

[151] Kulin Shah, Sitan Chen, and Adam Klivans. Learning mixtures of gaussians using the DDPM objective. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023.

[152] Tao Shen, Jie Zhang, Xinkang Jia, Fengda Zhang, Gang Huang, Pan Zhou, Kun Kuang, Fei Wu, and Chao Wu. Federated mutual learning. *arXiv preprint arXiv:2006.16765*, 2020.

[153] Naichen Shi and Raed Al Kontar. Personalized pca: Decoupling shared and unique features. *arXiv preprint arXiv:2207.08041*, 2022.

[154] Mohammad Shoeybi, Mostofa Patwary, Raul Puri, Patrick LeGresley, Jared Casper, and Bryan Catanzaro. Megatron-lm: Training multi-billion parameter language models using model parallelism, 2020.

[155] Navjot Singh, Deepesh Data, Jemin George, and Suhas Diggavi. Squarm-sgd: Communication-efficient momentum sgd for decentralized optimization. *IEEE Journal on Selected Areas in Information Theory*, 2(3):954–969, 2021.

[156] Samuel L Smith, Benoit Dherin, David Barrett, and Soham De. On the origin of implicit regularization in stochastic gradient descent. In *International Conference on Learning Representations*, 2021.

[157] Virginia Smith, Chao-Kai Chiang, Maziar Sanjabi, and Ameet S. Talwalkar. Federated multi-task learning. In *Advances in Neural Information Processing Systems*, pages 4424–4434, 2017.

[158] Jascha Sohl-Dickstein, Eric A. Weiss, Niru Maheswaranathan, and Surya Ganguli. Deep unsupervised learning using nonequilibrium thermodynamics. In *Proceedings of the 32nd International Conference on Machine Learning, ICML 2015, Lille, France, 6-11 July 2015*, volume 37 of *JMLR Workshop and Conference Proceedings*, pages 2256–2265. JMLR.org, 2015.

[159] Yang Song, Jascha Sohl-Dickstein, Diederik P Kingma, Abhishek Kumar, Stefano Ermon, and Ben Poole. Score-based generative modeling through stochastic differential equations. In *International Conference on Learning Representations*, 2021.

[160] Charles Stein. Inadmissibility of the usual estimator for the mean of a multivariate normal distribution. In *Proceedings of the Third Berkeley Symposium on Mathematical Statistics and Probability, 1954–1955, vol. I*, pages 197–206. University of California Press, Berkeley-Los Angeles, Calif., 1956.

[161] Uri Stemmer. Locally private k-means clustering. In *SODA*, pages 548–559, 2020.

[162] Xiao Sun, Jungwook Choi, Chia-Yu Chen, Naigang Wang, Swagath Venkataramani, Vijayalakshmi Viji Srinivasan, Xiaodong Cui, Wei Zhang, and Kailash Gopalakrishnan. Hybrid 8-bit floating point (hfp8) training and inference for deep neural networks. *Advances in neural information processing systems*, 32, 2019.

[163] Kevin Tian, Weihao Kong, and Gregory Valiant. Learning populations of parameters. *Advances in neural information processing systems*, 30, 2017.

[164] Yonglong Tian, Yue Wang, Dilip Krishnan, Joshua B Tenenbaum, and Phillip Isola. Rethinking few-shot image classification: a good embedding is all you need? In *European Conference on Computer Vision*, pages 266–282. Springer, 2020.

[165] Michael E Tipping and Christopher M Bishop. Probabilistic principal component analysis. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 61(3):611–622, 1999.

[166] Hugo Touvron, Thibaut Lavril, Gautier Izacard, Xavier Martinet, Marie-Anne Lachaux, Timothée Lacroix, Baptiste Rozière, Naman Goyal, Eric Hambro, Faisal Azhar, et al. Llama: Open and efficient foundation language models. *arXiv preprint arXiv:2302.13971*, 2023.

[167] Ye Lin Tun, Chu Myaet Thwal, Ji Su Yoon, Sun Moo Kang, Chaoning Zhang, and Choong Seon Hong. Federated learning with diffusion models for privacy-sensitive vision tasks. In *2023 International Conference on Advanced Technologies for Communications (ATC)*, pages 305–310. IEEE, 2023.

[168] Paul Vanhaesebrouck, Aurélien Bellet, and Marc Tommasi. Decentralized collaborative learning of personalized models over networks. In *Artificial Intelligence and Statistics*, pages 509–517. PMLR, 2017.

[169] Ramya Korlakai Vinayak, Weihao Kong, Gregory Valiant, and Sham Kakade. Maximum likelihood estimation for learning populations of parameters. In *International Conference on Machine Learning*, pages 6448–6457. PMLR, 2019.

[170] Martin J Wainwright. *High-dimensional statistics: A non-asymptotic viewpoint*, volume 48. Cambridge University Press, 2019.

[171] Naigang Wang, Jungwook Choi, Daniel Brand, Chia-Yu Chen, and Kailash Gopalakrishnan. Training deep neural networks with 8-bit floating point numbers. *Advances in neural information processing systems*, 31, 2018.

[172] Olga Wichrowska, Niru Maheswaranathan, Matthew W Hoffman, Sergio Gomez Colmenarejo, Misha Denil, Nando Freitas, and Jascha Sohl-Dickstein. Learned optimizers that scale and generalize. In *International conference on machine learning*, pages 3751–3760. PMLR, 2017.

[173] Lu Xia, Stefano Massei, Michiel E. Hochstenbach, and Barry Koren. On the influence of stochastic roundoff errors and their bias on the convergence of the gradient descent method with low-precision floating-point computation, 2023.

[174] Xingyu Xie, Pan Zhou, Huan Li, Zhouchen Lin, and Shuicheng Yan. Adan: Adaptive nesterov momentum algorithm for faster optimizing deep models, 2023.

[175] Jiwei Yang, Xu Shen, Jun Xing, Xinmei Tian, Houqiang Li, Bing Deng, Jianqiang Huang, and Xian-sheng Hua. Quantization networks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019.

[176] Lei Yang, Jiaming Huang, Wanyu Lin, and Jiannong Cao. Personalized federated learning on non-iid data via group-based meta-learning. *ACM Transactions on Knowledge Discovery from Data*, 17(4):1–20, 2023.

[177] Penghang Yin, Shuai Zhang, Jiancheng Lyu, Stanley J. Osher, Yingyong Qi, and Jack Xin. Binaryrelax: A relaxation approach for training deep neural networks with quantized weights. *SIAM J. Imaging Sci.*, 11(4):2205–2223, 2018.

[178] Yang You, Igor Gitman, and Boris Ginsburg. Large batch training of convolutional networks, 2017.

[179] Yang You, Jing Li, Sashank Reddi, Jonathan Hseu, Sanjiv Kumar, Srinadh Bhojanapalli, Xiaodan Song, James Demmel, Kurt Keutzer, and Cho-Jui Hsieh. Large batch optimization for deep learning: Training bert in 76 minutes. In *International Conference on Learning Representations*, 2020.

[180] Tao Yu, Gaurav Gupta, Karthick Gopalswamy, Amith R Mamidala, Hao Zhou, Jeffrey Huynh, Youngsuk Park, Ron Diamant, Anoop Deoras, and Luke Huan. Collage: Lightweight low-precision strategy for LLM training. In *Forty-first International Conference on Machine Learning*, 2024.

[181] Manzil Zaheer, Sashank Reddi, Devendra Sachan, Satyen Kale, and Sanjiv Kumar. Adaptive methods for nonconvex optimization. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc., 2018.

[182] Pedram Zamirai, Jian Zhang, Christopher R. Aberger, and Christopher De Sa. Revisiting bfloat16 training, 2021.

[183] Valentina Zantedeschi, Aurélien Bellet, and Marc Tommasi. Fully decentralized joint learning of personalized models and collaboration graphs. In *International Conference on Artificial Intelligence and Statistics*, pages 864–874. PMLR, 2020.

[184] Lvmin Zhang, Anyi Rao, and Maneesh Agrawala. Adding conditional control to text-to-image diffusion models. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 3836–3847, 2023.

[185] Michael Zhang, Karan Sapra, Sanja Fidler, Serena Yeung, and Jose M. Alvarez. Personalized federated learning with first order model optimization. In *International Conference on Learning Representations*, 2021.

[186] Ying Zhang, Tao Xiang, Timothy M Hospedales, and Huchuan Lu. Deep mutual learning. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 4320–4328, 2018.

[187] Yuchen Zhang. *Distributed machine learning with communication constraints*. PhD thesis, EECS Department, University of California, Berkeley, May 2016.

[188] Juntang Zhuang, Tommy Tang, Yifan Ding, Sekhar C Tatikonda, Nicha Dvornek, Xenophon Papademetris, and James Duncan. Adabelief optimizer: Adapting stepsizes by the belief in observed gradients. In H. Larochelle, M. Ranzato, R. Hadsell, M.F. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 18795–18806. Curran Associates, Inc., 2020.