Communication Efficient and Differentially Private Optimization

Shuli Jiang

CMU-RI-TR-25-46

June 3, 2025

The Robotics Institute
School of Computer Science
Carnegie Mellon University
Pittsburgh, PA

Thesis Committee:
Gauri Joshi, *Chair*
Steven Wu
Giulia Fanti
Zachary Manchester
Swanand Kadhe, *IBM Research*

Submitted in partial fulfillment of the requirements for the degree of Doctor of Philosophy in Robotics.

To the ones who lit the bonfire.
In a world of gradients, noise, and shadow, a path was carved
—step by stochastic step—
toward the truth that awaits thee.

Abstract

In modern machine learning, the abundance of data generated across diverse and distributed sources has made distributed training a central paradigm, particularly in large-scale applications such as Federated Learning. However, two key challenges arise in distributed training: ensuring communication efficiency and preserving the privacy of sensitive data used during training. This thesis addresses these challenges by exploring the interplay between communication efficiency, differential privacy, and optimization algorithms—key elements for enabling scalable, efficient, and privacy-preserving distributed learning.

We first address communication efficiency in distributed optimization by introducing Rand-Proj-Spatial, a sparsification-based communication efficient estimator for distributed vector mean estimation that leverages cross-client correlation through random subspace projections using the Subsampled Randomized Hadamard Transform (SRHT), achieving significant improvements over conventional sparsification methods. Next, focusing on differential privacy in prediction tasks, we propose DaRRM, a unified framework for private majority ensembling that optimizes a data-dependent noise function to improve model utility under fixed privacy guarantees and demonstrates strong empirical performance in private image classification. Finally, examining the interplay between privacy and optimization, we analyze the limitations of differentially private shuffled gradient methods (DP-ShuffleG), a practical optimization algorithm for solving private empirical risk minimization (ERM), and introduce Interleaved-ShuffleG, a hybrid algorithm that incorporates public data to reduce empirical excess risk, supported by novel theoretical insights and superior empirical performance across diverse datasets and baselines.

Together, these contributions advance the understanding and design of communication-efficient and privacy-preserving optimization algorithms critical for scalable and secure distributed learning.

Acknowledgments

These people are awesome.

TODO. I will add this ASAP.

Funding

This work was supported by love and passion.

TODO. I will add this ASAP.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Distributed Learning

Data play a central role in training modern machine learning models, with model performance often improving as the volume and diversity of training samples increase [64]. In many real-world settings, however, vast amounts of data are generated and stored in a distributed fashion across multiple geographic or organizational locations—such as mobile devices, hospitals, or corporate data silos. For instance, in next-word prediction tasks, data are naturally collected on users' mobile phones [53], while in healthcare applications, training data for disease diagnosis are generated across diverse clinical sites and institutions [101].

However, directly aggregating these datasets on a central server is often impractical—limited by communication overhead and, more importantly, constrained by privacy concerns surrounding sensitive or proprietary information. Despite these limitations, there remains a strong incentive to leverage the richness of distributed data sources, as doing so enables models to generalize better across heterogeneous environments, capture rare patterns that may not exist in a single dataset, and reduce biases introduced by isolated data collection. To address these needs, a now well-established paradigm known as Federated Learning (FL) has emerged in distributed machine learning [79].

Federated Learning (FL)[79] is a collaborative machine learning paradigm that enables model training without requiring data to be centralized. In a typical FL

setup, as illustrated in Figure1.1, a central server (often referred to as a parameter server) coordinates the training process, while a set of distributed clients or users each hold local datasets from diverse and potentially non-identically distributed sources. Training proceeds in iterative communication rounds. In each round, clients independently train local models on their private data and send the resulting updates—such as gradients or model parameters—to the central server. The server then aggregates these updates to produce a new global model that reflects knowledge from across the distributed data sources. This global model is subsequently broadcast back to the clients, initiating the next round of training. By keeping raw data on-device and only exchanging model updates, FL preserves data locality and helps protect the privacy of sensitive user information.

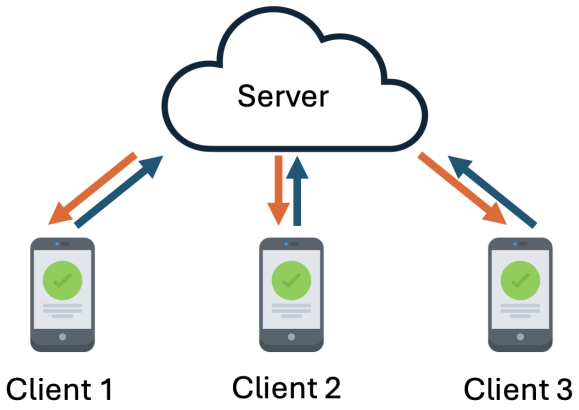

Figure 1.1: Illustration of a Federated Learning setup.

FL has been successfully applied across a wide range of domains, including—but not limited to—next-word prediction on mobile devices [53, 98, 114], digital healthcare [59, 72, 101], and distributed data analysis [111, 119]. Despite its broad adoption and demonstrated potential, FL continues to face two fundamental challenges: achieving communication efficiency and preserving client or user data privacy, which we discuss in details as follows. Addressing these challenges is critical to enabling the scalability and trustworthiness of FL systems in real-world deployments.

1.2 Challenge 1: Communication Efficiency

Although data are not directly transmitted between clients and the server, one of the central challenges remains communication efficiency, particularly in settings where user devices—such as smartphones or tablets—have limited bandwidth and power constraints. Modern machine learning models can be extremely large, often comprising millions or even billions of parameters, making it impractical for clients to transmit full model updates to a central server in every training round. This bottleneck becomes especially pronounced in cross-device FL scenarios, where thousands or millions of users participate asynchronously over unreliable connections.

To address this, a variety of communication-efficient techniques have been developed, including update sparsification [68], quantization [100], structured updates [68], and periodic averaging [133]. These methods aim to reduce the size and frequency of transmitted updates while preserving convergence guarantees, enabling scalable distributed optimization in bandwidth-constrained environments. In this work, we focus on a popular class of techniques that compress the actual data sent from clients to the server during each communication round—specifically, model parameters or gradient vectors. Among these, the two most widely used approaches are vector sparsification and quantization, which we detail below.

1.2.1 Sparsification and Quantization

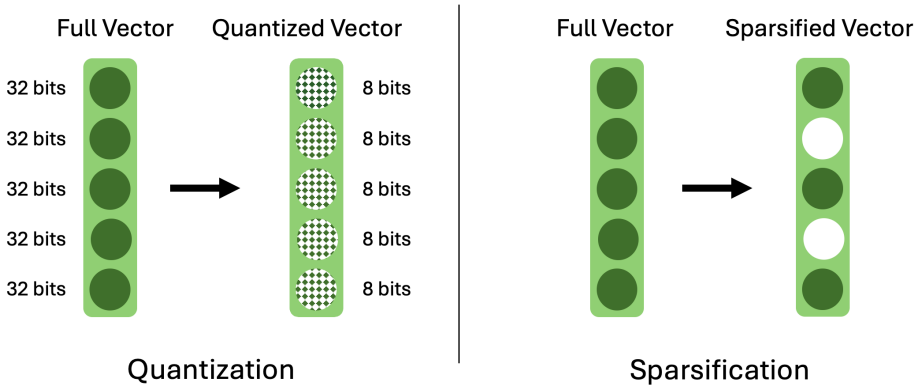

Figure 1.2: Illustration of quantization and sparsification.

We provide a visual illustration of quantization and sparsification techniques

in Figure 1.2, where vectors represent model parameters or gradients exchanged between the clients and the server during training. At a high level, quantization, e.g., [29, 115, 128] reduces the number of bits used to represent each coordinate of a vector, while sparsification, e.g. [5, 113, 135], reduces the number of coordinates transmitted from clients to the server.

Once the server receives these compressed vectors, it must reconstruct an estimate of their true aggregation to continue training. This naturally raises a key challenge: more aggressive compression—i.e., using fewer bits or transmitting fewer coordinates—reduces communication cost, but also degrades the fidelity of the transmitted vectors. As a result, the server's estimate of the aggregated client updates may become increasingly inaccurate, potentially leading to optimization instability or rendering the final model useless for downstream tasks.

In essence, a fundamental trade-off exists between communication efficiency and the accuracy of the server's estimate. This leads to our first question (Q1): Can we improve this trade-off?

Specifically, under a fixed communication budget, is it possible to design compression schemes that yield more accurate estimate at the server—bringing us closer to the true aggregate of client vectors while preserving scalability?

1.3 Challenge 2: Data Privacy

The second major challenge in distributed learning is protecting the privacy of clients' data during training. While FL mitigates some risks by keeping raw data on clients' devices, this local-data approach alone does not guarantee privacy. Prior research has shown that sensitive information can still leak through the shared model updates—such as gradient vectors or parameters—that are exchanged between the server and the clients during training. In particular, a long line of research has shown that FL is vulnerable to membership inference attacks, e.g., [8, 108, 117, 152], which can determine whether a specific data point was used in training, while data reconstruction attacks, e.g., [134, 142, 153] can partially or fully reconstruct users' private inputs by exploiting model gradients or parameters. These vulnerabilities

highlight that mere data locality is insufficient for strong privacy guarantees.

To address this challenge, Differential Privacy (DP) [35]—a rigorous mathematical framework for reasoning about and quantifying information leakage—has been widely adopted in the context of federated learning (FL) [62, 90, 99, 136]. In FL, privacy guarantees are most commonly enforced by adding carefully calibrated noise to model updates on the client side before they are shared with the central server. This approach helps limit the server's ability to infer sensitive information about a client's local data from the shared updates. That said, client-side noise injection is not the only mechanism for achieving differential privacy (DP) guarantees. Different DP configurations offer varying levels of privacy guarantees—ranging from central DP, where noise is added after aggregation at the server, to the more stringent local DP, where noise is added directly to each individual's data or update [87], and even group DP [92], which provides guarantees over subsets of users. A detailed discussion of these variants is beyond the scope of this thesis, and we refer interested readers to the cited works for further exploration.

Moreover, DP has proven effective in mitigating membership inference attacks [87, 117], as well as data reconstruction attacks [118, 155], improving privacy guarantees of distributed learning systems.

We provide a more detailed introduction to DP in the following section.

1.3.1 Differential Privacy (DP)

Differential Privacy (DP), first introduced by [35], provides a mathematically rigorous framework for quantifying the leakage of individual-level information.

At its core, differential privacy ensures that the output of a computation, such as the output model parameters used in downstream applications, remains nearly indistinguishable whether or not any single individual's data is used in the input, such as the data used to train the model.

Formally, a randomized algorithm $\mathcal{M} : \mathcal{D} \to \mathcal{R}$ is said to satisfy (ε, δ)-differential privacy if, for all neighboring datasets $D, D' \in \mathcal{D}$ that differ in exactly one record, and for all measurable subsets $\mathcal{S} \subseteq \mathcal{R}$, the following inequality holds:

$$\Pr[\mathcal{M}(\mathcal{D}) \in \mathcal{S}] \leq e^{\epsilon} \Pr[\mathcal{M}(\mathcal{D}') \in \mathcal{S}] + \delta \tag{1.1}$$

This condition ensures that the output distributions of $\mathcal{M}$ on D and D' are statistically close: the presence or absence of a single individual's data in the input has a limited effect on the distribution of the output. The parameter $\varepsilon \geq 0$ is known as the privacy loss parameter, with smaller values corresponding to stronger privacy guarantees. The parameter $\delta \geq 0$ allows for a small probability of the guarantee failing—i.e., a small chance that the output distributions differ by more than a factor of e^{ε}.

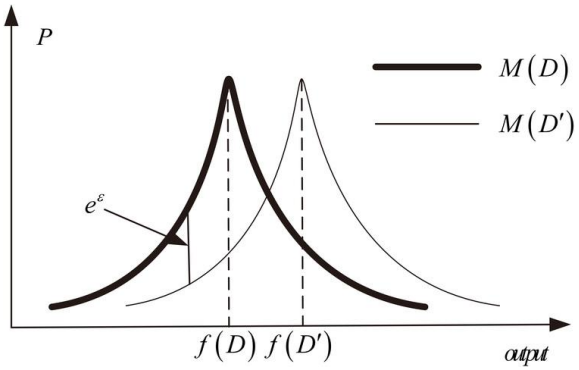

Figure 1.3: Illustration of the output distributions of a randomized algorithm $\mathcal{M}$ when applied to two neighboring datasets D and D' differing in one data point. Here, P denotes probability. $f(D)$ and $f(D')$ are the true outputs of some function f. $\mathcal{M}$ denotes a randomized mechanism that introduces noise to preserve privacy. $\mathcal{M}(\mathcal{D})$ and $\mathcal{M}(\mathcal{D}')$ represent the output distributions on the two neighboring datasets. The distributions are close in the sense of ε-differential privacy, meaning that $\mathcal{M}$ is ε-DP. Figure adapted from [31].

When $\delta = 0$, the definition is referred to as pure differential privacy; when $\delta > 0$, it is called approximate differential privacy. Figure 1.3 illustrates how differential privacy bounds the change in output distributions between neighboring datasets, enforcing stability in the algorithm's behavior with respect to individual data points. The DP guarantee intuitively implies that the presence or absence of any single individual's data has a limited effect on the output distribution of $\mathcal{A}$, thereby limiting the information that can be inferred about any one person.

However, DP guarantees do not come without cost. The injection of noise—central to DP's privacy guarantees—inevitably degrades the utility of the resulting model compared to its non-private counterpart. Here, utility refers to the effectiveness of the model in downstream tasks, which may be measured by classification accuracy, regression error (e.g., mean squared error), optimization

convergence rates, or other task-specific metrics.

This inherent trade-off between privacy and model utility is fundamental in many private learning problems: stronger privacy generally implies reduced performance, e.g., [12, 146, 149]. This observation naturally leads to our second question (Q2): Can we improve this trade-off?

That is, given a fixed privacy budget, can we design better algorithms or make use of available information that yield models with higher utility while still preserving the same level of privacy?

1.4 Thesis Statement

This thesis explores strategies to address the two central challenges—communication efficiency and privacy-utility trade-offs—across three distinct learning settings, and provides affirmative answers to both questions (Q1 and Q2) within each context.

This thesis advances distributed learning by improving **communication efficiency** via cross-client correlation and improving **privacy-utility trade-offs** in differentially private majority ensembling, and through the use of public data in differentially private shuffled gradient methods. These contributions enable more scalable, trustworthy and privacy-preserving model training in distributed settings.

In this thesis, we answer Q1 and Q2, and address the two core challenges through the following contributions:

1. **Improving communication cost-accuracy trade-offs**: we use practically available side information, i.e., cross-client correlation, to improve this trade-off in distributed vector mean estimation, a critical subroutine in distributed optimization algorithms.
2. **Improve the privacy-model utility trade-offs**:
   (a) We study how to improve this trade-off in the problem of private prediction

with majority ensembling by drawing connections and generalizing classical algorithms and baselines.

(b) We analyze and understand this trade-off in private shuffled gradient methods, a practical variant of optimization algorithms used in practice. We then propose to improve this trade-off in this problem using public data.

1.5 Summary of Contributions and Thesis Organization

We summarize the main contributions of this thesis and provide an overview below.

Rand-Proj-Spatial : Communication-Efficient Correlated Distributed Mean Estimation (Chapter 2)

1. We propose **the Rand-Proj-Spatial family estimator** for solving **distributed vector mean estimation** with a more flexible encoding-decoding procedure.
2. Rand-Proj-Spatial better leverages **the cross-client correlation information** to achieve a more general and improved mean estimator compared to existing approaches.
3. We propose to use **Subsampled Randomized Hadamard Transform (SRHT)** as the random linear maps in Rand-Proj-Spatial an theoretically show Rand-Proj-Spatial with SRHT achieves **reduced mean estimation error (MSE)** when the correlation is known.
4. We propose a practical configuration, Rand-Proj-Spatial (Avg) when the correlation is unknown.
5. We empirically demonstrate the superior performance of Rand-Proj-Spatial (Avg) in common distributed optimization tasks.

DaRRM: Improving Privacy-Utility Trade-offs in Private Prediction with Majority Ensembling (Chapter 3)

1. We generalize the classical Randomized Response (RR) mechanism and the commonly used subsampling baseline for solving private prediction with majority ensembling and propose the **Data-dependent Randomized Response Majority (DaRRM)**, which comes with a customizable noise function γ.
2. We show that DaRRM is **general**—it captures all algorithms computing the majority whose outputs are at least as good as a random guess, by choosing different γ functions.
3. Leveraging structural observations, we demonstrate that, under mild conditions, selecting the γ functions strategically yields **a 2× privacy amplification** compared to standard composition in the pure differential privacy setting.
4. We develop an optimization-based method to find a utility-maximizing, data-dependent γ function for DaRRM by reformulating its infinite privacy constraints into a finite, tractable set.
5. We empirically demonstrate that DaRRM with an optimized γ outperforms PATE [95] in private label aggregation under equal privacy budgets, **achieving up to 30% higher accuracy** in downstream private image classification tasks when the number of teachers is small.

***DP-ShuffleG*: Improving the Privacy-Convergence Trade-offs in Private Shuffled Gradient Methods with Public Data (Chapter 4)**

1. We present **a generalized shuffled gradient framework** that allows surrogate objectives (based on public samples) and varying noise addition across epochs.
2. We derive the general convergence result of the generalized shuffled gradient framework, based on **a novel dissimilarity metric**.
3. We show the empirical excess risk of *DP-ShuffleG*, a special case of the generalized shuffled gradient framework.
4. Based on the general framework, we propose ***Interleaved-ShuffleG***, an algorithm that uses both private and public samples within each epoch, that achieves **lower empirical excess risk** compared to *DP-ShuffleG* and other

approaches that use public samples.

5. We empirically demonstrate the superior performance of *Interleaved-ShuffleG* compared to the baselines in three tasks on diverse datasets.

1.6 Bibliographic Notes

This thesis is mainly based on the following works:

1. Chapter 2:
   - **Shuli Jiang**, Pranay Sharma, Gauri Joshi. "**Correlation Aware Sparsified Mean Estimation Using Random Projection**". *The Thirty-seventh Conference on Neural Information Processing Systems (NeurIPS), 2023.* [paper]
2. Chapter 3:
   - **Shuli Jiang**, Qiuyi Richard Zhang, Gauri Joshi. "**Optimized Tradeoffs for Private Prediction with Majority Ensembling**". *Transaction on Machine Learning Research (TMLR), 2024.* [paper]
3. Chapter 4:
   - **Shuli Jiang**, Pranay Sharma, Steven Wu, Gauri Joshi. **"Improving the Convergence of Private Shuffled Gradient Methods with Public Data"**. *In Submission, 2025.* [paper]

Additionally, I have contributed to several other works during my Ph.D. journey that fall outside the main scope of this thesis; these are omitted from the main discussion and are listed below, grouped by topic.

Data compression / Sketching / Randomized Numerical Linear Algebra:

- (alphabetical order) **Shuli Jiang**, Dongyu Li, Irene Mengze Li, Arvind V. Mahankali, David P. Woodruff. "**Streaming and Distributed Algorithms for Robust Column Subset Selection**". *The Thirty-eighth International Conference on Machine Learning (ICML), 2021.* [paper]

- (alphabetical order) **Shuli Jiang**, Hai Thanh Pham, David P. Woodruff, Qiuyi Richard Zhang. "**Optimal Sketching for Trace Estimation**". *The Thirty-fifth Conference on Neural Information Processing Systems (NeurIPS), 2021*. [paper]

Outlier Detection:

- **Shuli Jiang**, Robson Leonardo Ferreira Cordeiro, Leman Akoglu. "**D.MCA: Outlier Detection with Explicit Micro-Cluster Assignments**". *The Twenty-second IEEE International Conference on Data Mining (ICDM), 2022*. [paper]

Attacking Large Language Models (LLMs):

- **Shuli Jiang**, Swanand Kadhe, Yi Zhou, Ling Cai, Nathalie Baracaldo. "**Forcing Generative Models to Degenerate Ones: The Power of Data Poisoning Attacks**". *NeurIPS 2023 Workshop on Backdoors in Deep Learning - The Good, the Bad, and the Ugly*. [paper]
- **Shuli Jiang**, Swanand Ravindra Kadhe, Yi Zhou, Farhan Ahmed, Ling Cai, Nathalie Baracaldo. "**Turning Generative Models Degenerate: The Power of Data Poisoning Attacks**". *Full version of the above paper, arXiv, 2024*. [paper]

Differential Privacy with Public Features:

- **Shuli Jiang**, Walid Krichene, Nicolas Mayoraz. "**Private Learning with Public Feature Conditioning**". *In submission, 2025*.

Chapter 2

Communication Efficiency: Correlated Distributed Mean Estimation

This chapter is based on the following work:

> ***Shuli Jiang**, Pranay Sharma, Gauri Joshi. **"Correlation Aware Sparsified Mean Estimation Using Random Projection"**. The Thirty-seventh Conference on Neural Information Processing Systems (NeurIPS), 2023. [paper]*

Clients involved in a distributed training of a machine learning model often face constraints in communication bandwidth. Consequently, in scenarios where the global model to be trained is large, it becomes essential for each client to compress their local vectors to be sent to the server. This chapter focuses on improving the communication efficiency through practically available side information in distributed learning.

Abstract. We study the problem of communication-efficient distributed vector mean estimation[1], a commonly used subroutine in distributed optimization and Federated Learning (FL). Rand-k sparsification is a commonly used technique to reduce communication cost, where each client sends $k < d$ of its coordinates to the server. However, Rand-k is agnostic to any correlations, that might exist between clients in practical scenarios. The recently proposed Rand-k-Spatial estimator leverages the

[1]Note we focus on the empirical mean estimation, where the vectors at the clients are fixed. This is as opposed to statistical mean estimation, where the vectors are drawn from some distribution.

cross-client correlation information at the server to improve Rand-k's performance. Yet, the performance of Rand-k-Spatial is suboptimal. We propose the Rand-Proj-Spatial estimator with a more flexible encoding-decoding procedure, which generalizes the encoding of Rand-k by projecting the client vectors to a random k-dimensional subspace. We utilize Subsampled Randomized Hadamard Transform (SRHT) as the projection matrix and show that Rand-Proj-Spatial with SRHT outperforms Rand-k-Spatial, using the correlation information more efficiently. Furthermore, we propose an approach to incorporate varying degrees of correlation and suggest a practical variant of Rand-Proj-Spatial when the correlation information is not available to the server. Experiments on real-world distributed optimization tasks showcase the superior performance of Rand-Proj-Spatial compared to Rand-k-Spatial and other more sophisticated sparsification techniques.

2.1 Introduction

In modern machine learning applications, data is naturally distributed across a large number of edge devices or clients. The underlying learning task in such settings is modeled by distributed optimization or the recent paradigm of Federated Learning (FL) [63, 67, 79, 130]. A crucial subtask in distributed learning is for the server to compute the mean of the vectors sent by the clients. In FL, for example, clients run training steps on their local data and once-in-a-while send their local models (or local gradients) to the server, which averages them to compute the new global model. However, with the ever-increasing size of machine learning models [23, 110], and the limited battery life of the edge clients, communication cost is often the major constraint for the clients. This motivates the problem of (empirical) *distributed mean estimation* (DME) under communication constraints, as illustrated in Figure 2.1. Each of the n clients holds a vector $\mathbf{x}_i \in \mathbb{R}^d$, on which there are no distributional assumptions. Given a communication budget, each client sends a compressed version $\widehat{\mathbf{x}}_i$ of its vector to the server, which utilizes these to compute an estimate of the mean vector $\frac{1}{n}\sum_{i=1}^{n}\mathbf{x}_i$.

Quantization and sparsification are two major techniques for reducing the communication costs of DME. Quantization [29, 50, 115, 128] involves compressing each coordinate of the client vector to a given precision and aims to reduce the number

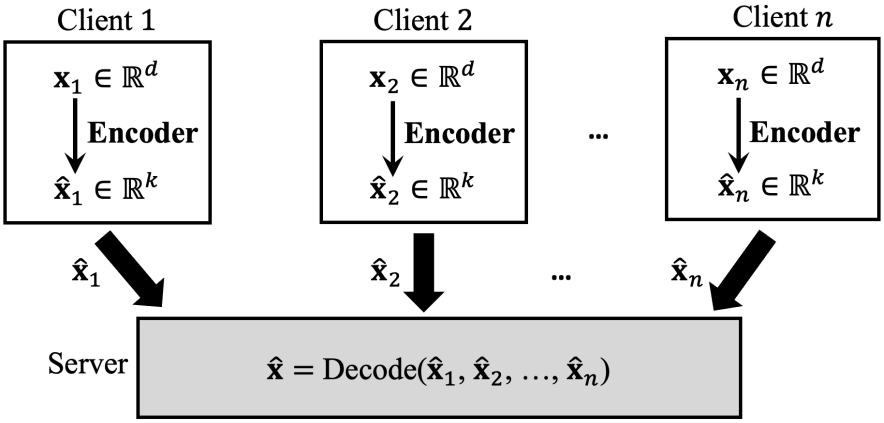

Figure 2.1: The problem of distributed mean estimation under limited communication. Each client $i \in [n]$ encodes its vector $\mathbf{x}_i$ as $\widehat{\mathbf{x}}_i$ and sends this compressed version to the server. The server decodes them to compute an estimate of the true mean $\frac{1}{n}\sum_{i=1}^{n}\mathbf{x}_i$.

of bits to represent each coordinate, achieving a constant reduction in the communication cost. However, the communication cost still remains $\Theta(d)$. Sparsification, on the other hand, aims to reduce the number of coordinates each clinet sends and compresses each client vector to only $k \ll d$ of its coordinates (e.g. Rand-k [66]). As a result, sparsification reduces communication costs more aggressively compared to quantization, achieving better communication efficiency at a cost of only $O(k)$. While in practice, one can use a combination of quantization and sparsification techniques for communication cost reduction, in this work, we focus on the more aggressive sparsification techniques. We call k, the dimension of the vector each client sends to the server, the *per-client* communication budget.

Most existing works on sparsification ignore the potential correlation (or similarity) among the client vectors, which often exists in practice. For example, the data of a specific client in federated learning can be similar to that of multiple clients. Hence, it is reasonable to expect their models (or gradients) to be similar as well. To the best of our knowledge, [57] is the first work to account for *spatial* correlation across individual client vectors. They propose the Rand-k-Spatial family of unbiased estimators, which generalizes Rand-k and achieves a better estimation error in the presence of cross-client correlation. However, their approach is focused only on the server-side decoding procedure, while the clients do simple Rand-k encoding.

In this work, we consider a more general encoding scheme that directly compresses a vector from $\mathbb{R}^d$ to $\mathbb{R}^k$ using a (random) linear map. The encoded vector consists of k linear combinations of the original coordinates. Intuitively, this has a higher chance of capturing the large-magnitude coordinates ("heavy hitters") of the vector than randomly sampling k out of the d coordinates (Rand-k), which is crucial for the estimator to recover the true mean vector. For example, consider a vector where only a few coordinates are heavy hitters. For small k, Rand-k has a decent chance of missing all the heavy hitters. But with a linear-maps-based general encoding procedure, the large coordinates are more likely to be encoded in the linear measurements, resulting in a more accurate estimator of the mean vector. Guided by this intuition, we ask:

Can we design an improved joint encoding-decoding scheme that utilizes the correlation information and achieves an improved estimation error?

One naïve solution is to apply the same random rotation matrix $\boldsymbol{G} \in \mathbb{R}^{d\times d}$ to each client vector, before applying Rand-k or Rand-k-Spatial encoding. Indeed, such preprocessing is applied to improve the estimator using quantization techniques on heterogeneous vectors [115, 116]. However, as we see in Appendix A.1.1, for sparsification, we can show that this leads to no improvement. But what happens if every client uses a different random matrix, or applies a random $k \times d$-dimensional linear map? How to design the corresponding decoding procedure to leverage cross-client correlation? As there is no way for one to directly apply the decoding procedure of Rand-k-Spatial in such cases. To answer these questions, we propose the Rand-Proj-Spatial family estimator. We propose a flexible encoding procedure in which each client applies its own random linear map to encode the vector. Further, our novel decoding procedure can better leverage cross-client correlation. The resulting mean estimator generalizes and improves over the Rand-k-Spatial family estimator.

Next, we discuss some reasonable restrictions we expect our mean estimator to obey. 1) *Unbiased*. An unbiased mean estimator is theoretically more convenient compared to a biased one [55]. 2) *Non-adaptive*. We focus on an encoding procedure that does not depend on the actual client data, as opposed to the *adaptive* ones, e.g. Rand-k with vector-based sampling probability [66, 135]. Designing a data-adaptive encoding procedure is computationally expensive as this might require using an iterative procedure to find out the sampling probabilities [66]. In practice, however, clients often have limited computational power compared to the server. Further,

as discussed earlier, mean estimation is often a subroutine in more complicated tasks. For applications with streaming data [89], the additional computational overhead of adaptive schemes is challenging to maintain. Note that both Rand-k and Rand-k-Spatial family estimator [57] are *unbiased* and *non-adaptive*.

In this paper, we focus on the severely communication-constrained case $nk \leq d$, when the server receives very limited information about any single client vector. If $nk \gg d$, we see in Appendix A.1.2 that the cross-client information has no additional advantage in terms of improving the mean estimate under both Rand-k-Spatial or Rand-Proj-Spatial, with different choices of random linear maps. Furthermore, when $nk \gg d$, the performance of both the estimators converges to that of Rand-k. Intuitively, this means when the server receives sufficient information regarding the client vectors, it does not need to leverage cross-client correlation to improve the mean estimator.

2.1.1 Our Contributions

Our contributions can be summarized as follows:

1. We propose the Rand-Proj-Spatial family estimator with a more flexible encoding-decoding procedure, which can better leverage the cross-client correlation information to achieve a more general and improved mean estimator compared to existing ones.
2. We show the benefit of using Subsampled Randomized Hadamard Transform (SRHT) as the random linear maps in Rand-Proj-Spatial in terms of better mean estimation error (MSE). We theoretically analyze the case when the correlation information is known at the server (see Theorems 2.4.3, 2.4.4 and Section 2.4.3). Further, we propose a practical configuration called Rand-Proj-Spatial(Avg) when the correlation is unknown.
3. We conduct experiments on common distributed optimization tasks, and demonstrate the superior performance of Rand-Proj-Spatial compared to existing sparsification techniques.

2.2 Related Work

2.2.1 Quantization and Sparsification

Commonly used techniques to achieve communication efficiency are quantization, sparsification, or more generic compression schemes, which generalize the former two [17]. Quantization involves either representing each coordinate of the vector by a small number of bits [4, 18, 29, 100, 115, 128], or more involved vector quantization techniques [44, 107]. Sparsification [5, 65, 104, 113, 135], on the other hand, involves communicating a small number $k < d$ of coordinates, to the server. Common protocols include Rand-k [66], sending k uniformly randomly selected coordinates; Top-k [106], sending the k largest magnitude coordinates; and a combination of the two [11]. Some recent works, with a focus on distributed learning, further refine these communication-saving mechanisms [91] by incorporating temporal correlation or error feedback [55, 65].

2.2.2 Distributed Mean Estimation (DME)

DME has wide applications in distributed optimization and FL. Most of the existing literature on DME either considers statistical mean estimation [45, 148], assuming that the data across clients is generated i.i.d. according to the same distribution, or empirical mean estimation [25, 57, 66, 78, 115, 127, 129], without making any distributional assumptions on the data. A recent line of work on empirical DME considers applying additional information available to the server, to further improve the mean estimate. This side information includes cross-client correlation [57, 116], or the memory of the past updates sent by the clients [73].

2.2.3 Subsampled Randomized Hadamard Transformation (SRHT)

SRHT was introduced for random dimensionality reduction using sketching [3, 70, 123]. Common applications of SRHT include faster computation of matrix problems, such as low-rank approximation [9, 22], and machine learning tasks, such as ridge

regression [77], and least square problems [52, 69, 120]. SRHT has also been applied to improve communication efficiency in distributed optimization [56] and FL [51, 102].

2.3 Preliminaries

Notation. We use bold lowercase (uppercase) letters, e.g. $\mathbf{x}$ ($\boldsymbol{G}$) to denote vectors (matrices). $\mathbf{e}_j \in \mathbb{R}^d$, for $j \in [d]$, denotes the j-th canonical basis vector. $\|\cdot\|_2$ denotes the Euclidean norm. For a vector $\mathbf{x}$, $\mathbf{x}(j)$ denotes its j-th coordinate. Given integer m, we denote by $[m]$ the set $\{1, 2, \ldots, m\}$.

2.3.1 Problem Setup

Consider n geographically separated clients coordinated by a central server. Each client $i \in [n]$ holds a vector $\mathbf{x}_i \in \mathbb{R}^d$, while the server wants to estimate the mean vector $\bar{\mathbf{x}} \triangleq \frac{1}{n}\sum_{i=1}^n \mathbf{x}_i$. Given a per-client communication budget of $k \in [d]$, each client i computes $\widehat{\mathbf{x}}_i$ and sends it to the central server. $\widehat{\mathbf{x}}_i$ is an approximation of $\mathbf{x}_i$ that belongs to a random k-dimensional subspace. Each client also sends a random seed to the server, which conveys the subspace information, and can usually be communicated using a negligible amount of bits. Having received the encoded vectors $\{\widehat{\mathbf{x}}_i\}_{i=1}^n$, the server then computes $\widehat{\mathbf{x}} \in \mathbb{R}^d$, an estimator of $\bar{\mathbf{x}}$. We consider the severely communication-constrained setting where $nk \leq d$, when only a limited amount of information about the client vectors is seen by the server.

2.3.2 Error Metric

We measure the quality of the decoded vector $\widehat{\mathbf{x}}$ using the Mean Squared Error (MSE) $\mathbb{E}\left[\|\widehat{\mathbf{x}} - \bar{\mathbf{x}}\|_2^2\right]$, where the expectation is with respect to all the randomness in the encoding-decoding scheme. Our goal is to design an encoding-decoding algorithm to achieve an unbiased estimate $\widehat{\mathbf{x}}$ (i.e. $\mathbb{E}[\widehat{\mathbf{x}}] = \bar{\mathbf{x}}$) that minimizes the MSE, given the per-client communication budget k. To consider an example, in rand-k sparsification, each client sends randomly selected k out of its d coordinates to the server. The server then computes the mean estimate as $\widehat{\mathbf{x}}^{(\text{Rand-}k)} = \frac{1}{n}\frac{d}{k}\sum_{i=1}^n \widehat{\mathbf{x}}_i$. By [57, Lemma

1], the MSE of Rand-k sparsification is given by

$$\mathbb{E}\left[\|\widehat{\mathbf{x}}^{(\text{Rand-}k)} - \bar{\mathbf{x}}\|_2^2\right] = \frac{1}{n^2}\left(\frac{d}{k} - 1\right)\sum_{i=1}^{n}\|\mathbf{x}_i\|_2^2 \tag{2.1}$$

2.3.3 The Rand-k-Spatial Family Estimator

For large values of $\frac{d}{k}$, the Rand-k MSE in Eq. 2.1 can be prohibitive. [57] proposed the Rand-k-Spatial family estimator, which achieves an improved MSE, by leveraging the knowledge of the correlation between client vectors at the server. The encoded vectors $\{\widehat{\mathbf{x}}_i\}$ are the same as in Rand-k. However, the j-th coordinate of the decoded vector is given as

$$\widehat{\mathbf{x}}^{(\text{Rand-}k\text{-Spatial})}(j) = \frac{1}{n}\frac{\bar{\beta}}{T(M_j)}\sum_{i=1}^{n}\widehat{\mathbf{x}}_i(j) \tag{2.2}$$

Here, $T : \mathbb{R} \to \mathbb{R}$ is a pre-defined transformation function of M_j, the number of clients which sent their j-th coordinate, and $\bar{\beta}$ is a normalization constant to ensure $\widehat{\mathbf{x}}$ is an unbiased estimator of $\mathbf{x}$. The resulting MSE is given by

$$\mathbb{E}\left[\|\widehat{\mathbf{x}}^{(\text{Rand-}k\text{-Spatial})} - \bar{\mathbf{x}}\|_2^2\right] = \frac{1}{n^2}\left(\frac{d}{k} - 1\right)\sum_{i=1}^{n}\|\mathbf{x}_i\|_2^2 + \left(c_1\sum_{i=1}^{n}\|\mathbf{x}_i\|_2^2 - c_2\sum_{i=1}^{n}\sum_{l\neq i}\langle\mathbf{x}_i, \mathbf{x}_l\rangle\right) \tag{2.3}$$

where c_1, c_2 are constants dependent on n, d, k and T, but independent of client vectors $\{\mathbf{x}_i\}_{i=1}^n$. When the client vectors are orthogonal, i.e., $\langle\mathbf{x}_i, \mathbf{x}_l\rangle = 0$, for all $i \neq l$, [57] show that with appropriately chosen T, the MSE in Eq. 2.3 reduces to Eq. 2.1. However, if there exists a positive correlation between the vectors, the MSE in Eq. 2.3 is strictly smaller than that for Rand-k Eq. 2.1.

2.4 The Rand-Proj-Spatial Family Estimator

While the Rand-k-Spatial family estimator proposed in [57] focuses only on improving the decoding at the server, we consider a more general encoding-decoding scheme. Rather than simply communicating k out of the d coordinates of its vector $\mathbf{x}_i$ to

the server, client i applies a (random) linear map $\boldsymbol{G}_i \in \mathbb{R}^{k\times d}$ to $\mathbf{x}_i$ and sends $\widehat{\mathbf{x}}_i = \boldsymbol{G}_i\mathbf{x}_i \in \mathbb{R}^k$ to the server. The decoding process on the server first projects the *encoded* vectors $\{\boldsymbol{G}_i\mathbf{x}_i\}_{i=1}^n$ back to the d-dimensional space and then forms an estimate $\widehat{\mathbf{x}}$. We motivate our new decoding procedure with the following regression problem:

$$\widehat{\mathbf{x}}^{(\text{Rand-Proj})} = \operatorname*{argmin}_{\mathbf{x}} \sum_{i=1}^{n} \|\boldsymbol{G}_i\mathbf{x} - \boldsymbol{G}_i\mathbf{x}_i\|_2^2 \tag{2.4}$$

To understand the motivation behind Eq. 2.4, first consider the special case where $\boldsymbol{G}_i = \boldsymbol{I}_d$ for all $i \in [n]$, that is, the clients communicate their vectors without compressing. The server can then exactly compute the mean $\bar{\mathbf{x}} = \frac{1}{n}\sum_{i=1}^n \mathbf{x}_i$. Equivalently, $\bar{\mathbf{x}}$ is the solution of $\operatorname{argmin}_{\mathbf{x}} \sum_{i=1}^n \|\mathbf{x} - \mathbf{x}_i\|_2^2$. In the more general setting, we require that the mean estimate $\widehat{\mathbf{x}}$ when encoded using the map $\boldsymbol{G}_i$, should be "close" to the encoded vector $\boldsymbol{G}_i\mathbf{x}_i$ originally sent by client i, for all clients $i \in [n]$.

We note the above intuition can also be translated into different regression problems to motivate the design of the new decoding procedure. We discuss in Appendix A.2.2 intuitive alternatives which, unfortunately, either do not enable the usage of cross-client correlation information, or do not use such information effectively. We choose the formulation in Eq. 2.4 due to its analytical tractability and its direct relevance to our target error metric MSE. We note that it is possible to consider the problem in Eq. 2.4 in the other norms, such as the sum of ℓ_2 norms (without the squares) or the ℓ_∞ norm. We leave this as a future direction to explore.

The solution to Eq. 2.4 is given by $\widehat{\mathbf{x}}^{(\text{Rand-Proj})} = (\sum_{i=1}^n \boldsymbol{G}_i^T\boldsymbol{G}_i)^\dagger \sum_{i=1}^n \boldsymbol{G}_i^T\boldsymbol{G}_i\mathbf{x}_i$, where $\dagger$ denotes the Moore-Penrose pseudo inverse [48]. However, while $\widehat{\mathbf{x}}^{(\text{Rand-Proj})}$ minimizes the error of the regression problem, our goal is to design an *unbiased* estimator that also improves the MSE. Therefore, we make the following two modifications to $\widehat{\mathbf{x}}^{(\text{Rand-Proj})}$: First, to ensure that the mean estimate is unbiased, we scale the solution by a normalization factor $\bar{\beta}$[2]. Second, to incorporate varying degrees of correlation among the clients, we propose to apply a scalar transformation function $T : \mathbb{R} \to \mathbb{R}$ to each of the eigenvalues of $\sum_{i=1}^n \boldsymbol{G}_i^T\boldsymbol{G}_i$. The resulting Rand-Proj-Spatial

[2]We show that it suffices for $\bar{\beta}$ to be a scalar in Appendix A.2.1.

family estimator is given by

$$\widehat{\mathbf{x}}^{(\text{Rand-Proj-Spatial})} = \bar{\beta}\Big(T(\sum_{i=1}^{n} \boldsymbol{G}_i^T \boldsymbol{G}_i)\Big)^{\dagger} \sum_{i=1}^{n} \boldsymbol{G}_i^T \boldsymbol{G}_i \mathbf{x}_i \tag{2.5}$$

Though applying the transformation function T in Rand-Proj-Spatial requires computing the eigendecomposition of $\sum_{i=1}^{n} \boldsymbol{G}_i^T \boldsymbol{G}_i$. However, this happens only at the server, which has more computational power than the clients. Next, we observe that for appropriate choice of $\{\boldsymbol{G}_i\}_{i=1}^n$, the Rand-Proj-Spatial family estimator reduces to the Rand-k-Spatial family estimator [57].

Lemma 2.4.1 (Recovering Rand-k-Spatial)**.** *Suppose client i generates a subsampling matrix $\boldsymbol{E}_i = \begin{bmatrix} \mathbf{e}_{i_1}, & \ldots, & \mathbf{e}_{i_k} \end{bmatrix}^\top$, where $\{\mathbf{e}_j\}_{j=1}^d$ are the canonical basis vectors, and $\{i_1, \ldots, i_k\}$ are sampled from $\{1, \ldots, d\}$ without replacement. The encoded vectors are given as $\widehat{\mathbf{x}}_i = \boldsymbol{E}_i \mathbf{x}_i$. Given a function T, $\widehat{\mathbf{x}}$ computed as in Eq. 2.5 recovers the Rand-k-Spatial estimator.*

The proof details are in Appendix A.3.5. We discuss the choice of T and how it compares to Rand-k-Spatial in detail in Section 2.4.3.

Remark 2.4.2. *In the simple case when $\boldsymbol{G}_i$'s are subsampling matrices (as in Rand-k-Spatial [57]), the j-th diagonal entry of $\sum_{i=1}^{n} \boldsymbol{G}_i^T \boldsymbol{G}_i$, M_j conveys the number of clients which sent the j-th coordinate. Rand-k-Spatial incorporates correlation among client vectors by applying a function T to M_j. Intuitively, it means scaling different coordinates differently. This is in contrast to Rand-k, which scales all the coordinates by d/k. In our more general case, we apply a function T to the eigenvalues of $\sum_{i=1}^{n} \boldsymbol{G}_i^T \boldsymbol{G}_i$ to similarly incorporate correlation in Rand-Proj-Spatial.*

To showcase the utility of the Rand-Proj-Spatial family estimator, we propose to set the random linear maps $\boldsymbol{G}_i$ to be scaled Subsampled Randomized Hadamard Transform (SRHT, e.g. [123]). Assuming d to be a power of 2, the linear map $\boldsymbol{G}_i$ is given as

$$\boldsymbol{G}_i = \frac{1}{\sqrt{d}} \boldsymbol{E}_i \boldsymbol{H} \boldsymbol{D}_i \in \mathbb{R}^{k \times d} \tag{2.6}$$

where $\boldsymbol{E}_i \in \mathbb{R}^{k \times d}$ is the subsampling matrix, $\boldsymbol{H} \in \mathbb{R}^{d \times d}$ is the (deterministic) Hadamard matrix and $\boldsymbol{D}_i \in \mathbb{R}^{d \times d}$ is a diagonal matrix with independent Rademacher

random variables as its diagonal entries. We choose SRHT due to its superior performance compared to other random matrices. Other possible choices of random matrices for Rand-Proj-Spatial estimator include sketching matrices commonly used for dimensionality reduction, such as Gaussian [122, 137], row-normalized Gaussian, and Count Sketch [80], as well as error-correction coding matrices, such as Low-Density Parity Check (LDPC) [43] and Fountain Codes [109]. However, in the absence of correlation between client vectors, all these matrices suffer a higher MSE.

In the following, we first compare the MSE of Rand-Proj-Spatial with SRHT against Rand-k and Rand-k-Spatial in two extreme cases: when all the client vectors are identical, and when all the client vectors are orthogonal to each other. In both cases, we highlight the transformation function T used in Rand-Proj-Spatial (Eq. 2.5) to incorporate the knowledge of cross-client correlation. We define

$$\mathcal{R} := \frac{\sum_{i=1}^{n} \sum_{l \neq i} \langle \mathbf{x}_i, \mathbf{x}_l \rangle}{\sum_{i=1}^{n} \|\mathbf{x}_i\|_2^2} \tag{2.7}$$

to measure the correlation between the client vectors. Note that $\mathcal{R} \in [-1, n-1]$. $\mathcal{R} = 0$ implies all client vectors are orthogonal, while $\mathcal{R} = n - 1$ implies identical client vectors.

2.4.1 Case I: Identical Client Vectors ($\mathcal{R} = n - 1$)

When all the client vectors are identical ($\mathbf{x}_i \equiv \mathbf{x}$), [57] showed that setting the transformation T to identity, i.e., $T(m) = m$, for all m, leads to the minimum MSE in the Rand-k-Spatial family of estimators. The resulting estimator is called Rand-k-Spatial (Max). Under the same setting, using the same transformation T in Rand-Proj-Spatial with SRHT, the decoded vector in Eq. 2.5 simplifies to

$$\widehat{\mathbf{x}}^{(\text{Rand-Proj-Spatial})} = \bar{\beta}\Big(\sum_{i=1}^{n} \boldsymbol{G}_i^T \boldsymbol{G}_i\Big)^{\dagger} \sum_{i=1}^{n} \boldsymbol{G}_i^T \boldsymbol{G}_i \mathbf{x} = \bar{\beta} \boldsymbol{S}^{\dagger} \boldsymbol{S} \mathbf{x}, \tag{2.8}$$

where $\boldsymbol{S} := \sum_{i=1}^{n} \boldsymbol{G}_i^T \boldsymbol{G}_i$. By construction, $\text{rank}(\boldsymbol{S}) \leq nk$, and we focus on the case $nk \leq d$.

Limitation of Subsampling matrices. As mentioned above, with $\boldsymbol{G}_i = \boldsymbol{E}_i, \forall\ i \in [n]$, we recover the Rand-$k$-Spatial family of estimators. In this case, $\boldsymbol{S}$ is

a diagonal matrix, where each diagonal entry $\boldsymbol{S}_{jj} = M_j$, $j \in [d]$. M_j is the number of clients which sent their j-th coordinate to the server. To ensure $\text{rank}(\boldsymbol{S}) = nk$, we need $\boldsymbol{S}_{jj} \leq 1, \forall j$, i.e., each of the d coordinates is sent by *at most* one client. If all the clients sample their matrices $\{\boldsymbol{E}_i\}_{i=1}^n$ independently, this happens with probability $\frac{\binom{d}{nk}}{\binom{d}{k}^n}$. As an example, for $k = 1$, $\text{Prob}(\text{rank}(\boldsymbol{S}) = n) = \frac{\binom{d}{n}}{d^n} \leq \frac{1}{n!}$ (because $\frac{d^n}{n^n} \leq \binom{d}{n} \leq \frac{d^n}{n!}$). Therefore, to guarantee that $\boldsymbol{S}$ is full-rank, each client would need the subsampling information of all the other clients. This not only requires additional communication but also has serious privacy implications. Essentially, the limitation with subsampling matrices $\boldsymbol{E}_i$ is that the eigenvectors of $\boldsymbol{S}$ are restricted to be canonical basis vectors $\{\mathbf{e}_j\}_{j=1}^d$. Generalizing $\boldsymbol{G}_i$'s to general rank k matrices relaxes this constraint and hence we can ensure that $\boldsymbol{S}$ is full-rank with high probability. In the next result, we show the benefit of choosing $\boldsymbol{G}_i$ as SRHT matrices. We call the resulting estimator Rand-Proj-Spatial(Max).

Theorem 2.4.3 (MSE under Full Correlation)**.** *Consider n clients, each holding the same vector $\mathbf{x} \in \mathbb{R}^d$. Suppose we set $T(\lambda) = \lambda$, $\bar{\beta} = \frac{d}{k}$ in Eq. 2.5, and the random linear map $\boldsymbol{G}_i$ at each client to be an SRHT matrix. Let δ be the probability that $\boldsymbol{S} = \sum_{i=1}^n \boldsymbol{G}_i^T \boldsymbol{G}_i$ does not have full rank. Then, for $nk \leq d$,*

$$\mathbb{E}\left[\|\widehat{\mathbf{x}}^{(\textit{Rand-Proj-Spatial(Max)})} - \bar{\mathbf{x}}\|_2^2\right] \leq \left[\frac{d}{(1-\delta)nk + \delta k} - 1\right]\|\mathbf{x}\|_2^2 \tag{2.9}$$

The proof details are in Appendix A.3.1. To compare the performance of Rand-Proj-Spatial(Max) against Rand-k, we show in Appendix A.3.2 that for $n \geq 2$, as long as $\delta \leq \frac{2}{3}$, the MSE of Rand-Proj-Spatial(Max) is less than that of Rand-k. Furthermore, in Appendix A.3.3 we empirically demonstrate that with $d \in \{32, 64, 128, \ldots, 1024\}$ and different values of $nk \leq d$, the rank of $\boldsymbol{S}$ is full with high probability, i.e., $\delta \approx 0$. This implies $\mathbb{E}[\|\widehat{\mathbf{x}}^{(\text{Rand-Proj-Spatial(Max)})} - \bar{\mathbf{x}}\|_2^2] \approx (\frac{d}{nk} - 1)\|\mathbf{x}\|_2^2$.

Furthermore, since setting $\boldsymbol{G}_i$ as SRHT significantly increases the probability of recovering nk coordinates of $\mathbf{x}$, the MSE of Rand-Proj-Spatial with SRHT (Eq. 2.4.3) is strictly less than that of Rand-k-Spatial (Eq. 2.3). We also compare the MSEs of the three estimators in Figure 2.2 in the following setting: $\|\mathbf{x}\|_2 = 1$, $d = 1024, n \in \{10, 20, 50, 100\}$ and small k values such that $nk < d$.

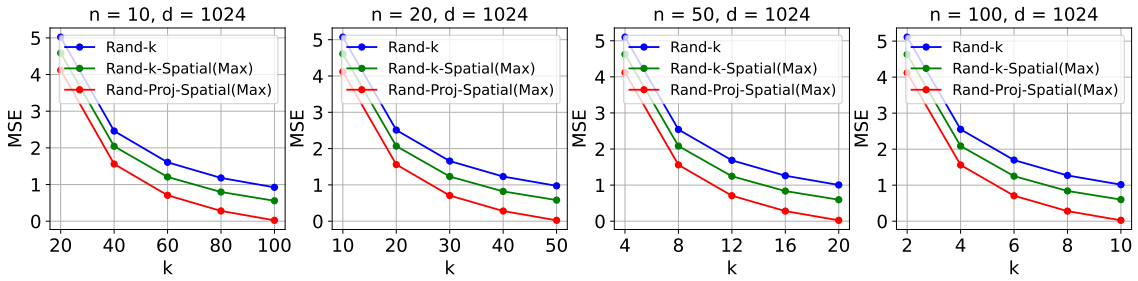

Figure 2.2: MSE comparison of Rand-k, Rand-k-Spatial(Max) and Rand-Proj-Spatial(Max) estimators, when all clients have identical vectors (maximum inter-client correlation).

2.4.2 Case II: Orthogonal Client Vectors ($\mathcal{R} = 0$)

When all the client vectors are orthogonal to each other, [57] showed that Rand-k has the lowest MSE among the Rand-k-Spatial family of decoders. We show in the next result that if we set the random linear maps $\boldsymbol{G}_i$ at client i to be SRHT, and choose the fixed transformation $T \equiv 1$ as in [57], Rand-Proj-Spatial achieves the same MSE as that of Rand-k.

Theorem 2.4.4 (MSE under No Correlation)**.** *Consider n clients, each holding a vector $\mathbf{x}_i \in \mathbb{R}^d$, $\forall i \in [n]$. Suppose we set $T \equiv 1$, $\bar{\beta} = \frac{d^2}{k}$ in Eq. 2.5, and the random linear map $\boldsymbol{G}_i$ at each client to be an SRHT matrix. Then, for $nk \leq d$,*

$$\mathbb{E}\left[\|\widehat{\mathbf{x}}^{(\textit{Rand-Proj-Spatial})} - \bar{\mathbf{x}}\|_2^2\right] = \frac{1}{n^2}\left(\frac{d}{k} - 1\right)\sum_{i=1}^{n}\|\mathbf{x}_i\|_2^2. \tag{2.10}$$

The proof details are in Appendix A.3.4. Theorem 2.4.4 above shows that with zero correlation among client vectors, Rand-Proj-Spatial achieves the same MSE as that of Rand-k.

2.4.3 Incorporating Varying Degrees of Correlation

In practice, it unlikely that all the client vectors are either identical or orthogonal to each other. In general, there is some "imperfect" correlation among the client vectors, i.e., $\mathcal{R} \in (0, n-1)$. Given correlation level $\mathcal{R}$, [57] shows that the estimator from the Rand-k-Spatial family that minimizes the MSE is given by the following

transformation.

$$T(m) = 1 + \frac{\mathcal{R}}{n-1}(m-1) \tag{2.11}$$

Recall from Section 2.4.1 (Section 2.4.2) that setting $T(m) = 1$ ($T(m) = m$) leads to the estimator among the Rand-k-Spatial family that minimizes MSE when there is zero (maximum) correlation among the client vectors. We observe the function T defined in Eq. 2.11 essentially interpolates between the two extreme cases, using the normalized degree of correlation $\frac{\mathcal{R}}{n-1} \in [-\frac{1}{n-1}, 1]$ as the weight. This motivates us to apply the same function T defined in Eq. 2.11 on the eigenvalues of $\boldsymbol{S} = \sum_{i=1}^{n} \boldsymbol{G}_i^T \boldsymbol{G}_i$ in Rand-Proj-Spatial. As we shall see in our results, the resulting Rand-Proj-Spatial family estimator improves over the MSE of both Rand-k and Rand-k-Spatial family estimator.

We note that deriving a closed-form expression of MSE for Rand-Proj-Spatial with SRHT in the general case with the transformation function T (Eq. 2.11) is hard (we elaborate on this in Appendix A.2.3), as this requires a closed form expression for the non-asymptotic distributions of eigenvalues and eigenvectors of the random matrix $\boldsymbol{S}$. To the best of our knowledge, previous analyses of SRHT, for example in [3, 69, 70, 71, 123], rely on the asymptotic properties of SRHT, such as the limiting eigen spectrum, or concentration bounds on the singular values, to derive asymptotic or approximate guarantees. However, to analyze the MSE of Rand-Proj-Spatial, we need an exact, non-asymptotic analysis of the eigenvalues and eigenvectors distribution of SRHT. Given the apparent intractability of the theoretical analysis, we compare the MSE of Rand-Proj-Spatial, Rand-k-Spatial, and Rand-k via simulations.

Simulations. In each experiment, we first simulate $\bar{\beta}$ in Eq. 2.5, which ensures our estimator is unbiased, based on 1000 random runs. Given the degree of correlation $\mathcal{R}$, we then compute the squared error, i.e. $\|\widehat{\mathbf{x}}^{(\text{Rand-Proj-Spatial})} - \bar{\mathbf{x}}\|_2^2$, where Rand-Proj-Spatial has $\boldsymbol{G}_i$ as SRHT matrix (Eq. 2.6) and T as in Eq. 2.11. We plot the average over 1000 random runs as an approximation to MSE. Each client holds a d-dimensional base vector $\mathbf{e}_j$ for some $j \in [d]$, and so two clients either hold the same or orthogonal vectors. We control the degree of correlation $\mathcal{R}$ by changing the number of clients which hold the same vector. We consider $d = 1024$, $n \in \{21, 51\}$. We consider positive correlation values, where $\mathcal{R}$ is chosen to be linearly spaced

within $[0, n-1]$. Hence, for $n = 21$, we use $\mathcal{R} \in \{4, 8, 12, 16\}$ and for $n = 51$, we use $\mathcal{R} \in \{10, 20, 30, 40\}$. All results are presented in Figure 2.3. As expected, given $\mathcal{R}$, Rand-Proj-Spatial consistently achieves a lower MSE than the lowest possible MSE from the Rand-k-Spatial family decoder. Additional results with different values of n, d, k, including the setting $nk \ll d$, can be found in Appendix A.2.4.

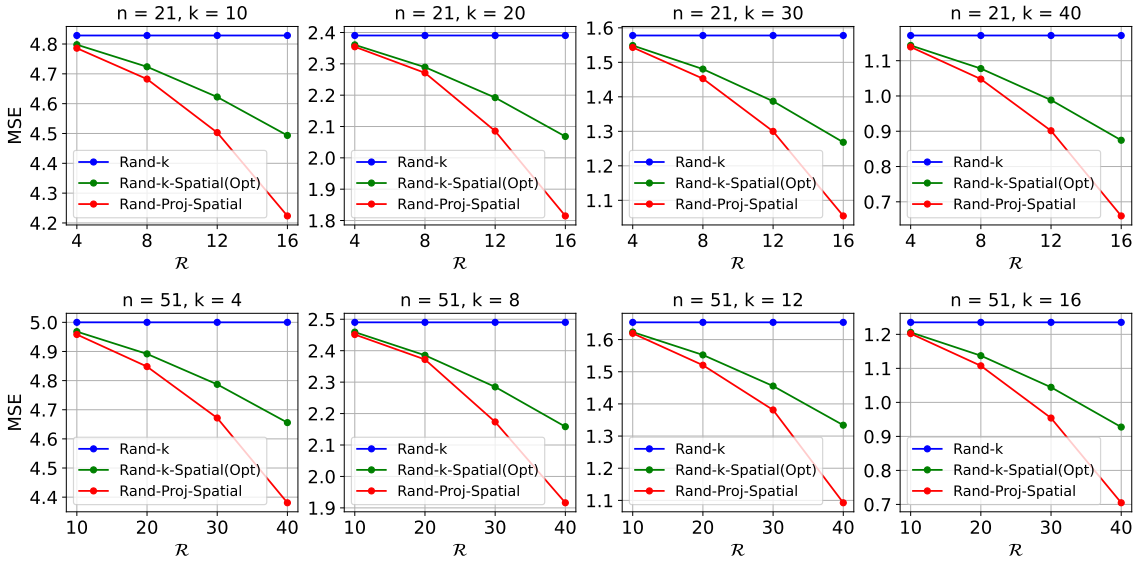

Figure 2.3: MSE comparison of estimators Rand-k, Rand-k-Spatial(Opt), Rand-Proj-Spatial, given the degree of correlation $\mathcal{R}$. Rand-k-Spatial(Opt) denotes the estimator that gives the lowest possible MSE from the Rand-k-Spatial family. We consider $d = 1024$, number of clients $n \in \{21, 51\}$, and k values such that $nk < d$. In each plot, we fix n, k, d and vary the degree of positive correlation $\mathcal{R}$. The y-axis represents MSE. Notice since each client has a fixed $\|\mathbf{x}_i\|_2 = 1$, and Rand-k does not leverage cross-client correlation, the MSE of Rand-k in each plot remains the same for different $\mathcal{R}$.

A Practical Configuration. In reality, it is hard to know the correlation information $\mathcal{R}$ among the client vectors. [57] uses the transformation function which interpolates to the middle point between the full correlation and no correlation cases, such that $T(m) = 1 + \frac{n}{2}\frac{m-1}{n-1}$. Rand-$k$-Spatial with such T is called Rand-k-Spatial(Avg). Following this approach, we evaluate Rand-Proj-Spatial with SRHT using this T, and call it Rand-Proj-Spatial(Avg) in practical settings (see Figure 2.4).

2.5 Experiments

We consider three practical distributed optimization tasks for evaluation: distributed power iteration, distributed k-means and distributed linear regression. We compare Rand-Proj-Spatial(Avg) against Rand-k, Rand-k-Spatial(Avg), and two more sophisticated but widely used sparsification schemes: non-uniform coordinate-wise gradient sparsification [135] (we call it Rand-k(Wangni)) and the Induced compressor with Rand-k + Top-k [55]. The results are presented in Figure 2.4.

2.5.1 Dataset

For both distributed power iteration and distributed k-means, we use the test set of the `Fashion-MNIST` dataset [141] consisting of 10000 samples. The original images from `Fashion-MNIST` are 28×28 in size. We preprocess and resize each image to be 32×32. Resizing images to have their dimension as a power of 2 is a common technique used in computer vision to accelerate the convolution operation. We use the `UJIndoor` dataset [3] for distributed linear regression. We subsample 10000 data points, and use the first 512 out of the total 520 features on signals of phone calls. The task is to predict the longitude of the location of a phone call. In all the experiments in Figure 2.4, the datasets are split IID across the clients via random shuffling. In Appendix A.4.1, we have additional results for non-IID data split across the clients.

2.5.2 Setup and Metric

Recall that n denotes the number of clients, k the per-client communication budget, and d the vector dimension. For Rand-Proj-Spatial, we use the first 50 iterations to estimate $\bar{\beta}$ (see Eq. 2.5). Note that $\bar{\beta}$ only depends on n, k, d, and T (the transformation function in Eq. 2.5), but is independent of the dataset. We repeat the experiments across 10 independent runs, and report the mean MSE (solid lines) and one standard deviation (shaded regions) for each estimator. For each task, we plot the squared error of the mean estimator $\widehat{\mathbf{x}}$, i.e., $\|\widehat{\mathbf{x}} - \bar{\mathbf{x}}\|_2^2$, and the values of the task-specific loss function, detailed below.

[3] https://archive.ics.uci.edu/ml/datasets/ujiindoorloc

2.5.3 Tasks and Settings

1. **Distributed power iteration.** We estimate the principle eigenvector of the covariance matrix, with the dataset (`Fashion-MNIST`) distributed across the n clients. In each iteration, each client computes a local principle eigenvector estimate based on a single power iteration and sends an encoded version to the server. The server then computes a global estimate and sends it back to the clients. The task-specific loss here is $\|\mathbf{v}_t - \mathbf{v}_{top}\|_2$, where $\mathbf{v}_t$ is the global estimate of the principal eigenvector at iteration t, and $\mathbf{v}_{top}$ is the true principle eigenvector.
2. **Distributed k-means.** We perform k-means clustering [10] with the data distributed across n clients (`Fashion-MNIST`, 10 classes) using Lloyd's algorithm. At each iteration, each client performs a single iteration of k-means to find its local centroids and sends the encoded version to the server. The server then computes an estimate of the global centroids and sends them back to the clients. We report the average squared mean estimation error across 10 clusters, and the k-means loss, i.e., the sum of the squared distances of the data points to the centroids.

   For both distributed power iterations and distributed k-means, we run the experiments for 30 iterations and consider two different settings: $n = 10, k = 102$ and $n = 50, k = 20$.
3. **Distributed linear regression.** We perform linear regression on the `UJIndoor` dataset distributed across n clients using SGD. At each iteration, each client computes a local gradient and sends an encoded version to the server. The server computes a global estimate of the gradient, performs an SGD step, and sends the updated parameter to the clients. We run the experiments for 50 iterations with learning rate 0.001. The task-specific loss is the linear regression loss, i.e. empirical mean squared error. To have a proper scale that better showcases the difference in performance of different estimators, we plot the results starting from the 10th iteration.

2.5.4 Results

It is evident from Figure 2.4 that Rand-Proj-Spatial(Avg), our estimator with the practical configuration T (see Section 2.4.3) that does not require the knowledge of the actual degree of correlation among clients, consistently outperforms the other estimators in all three tasks. Additional experiments for the three tasks are included in Appendix A.4.1. Furthermore, we present the wall-clock time to encode and decode client vectors using different sparsification schemes in Figure 2.5. Though Rand-Proj-Spatial(Avg) has the longest decoding time, the encoding time of Rand-Proj-Spatial(Avg) is less than that of the *adaptive* Rand-k(Wangni) sparsifier. In practice, the server has more computational power than the clients and hence can afford a longer decoding time. Therefore, it is more important to have efficient encoding procedures.

2.6 Limitations

We note two practical limitations of the proposed Rand-Proj-Spatial.

1) Computation Time of Rand-Proj-Spatial. The encoding time of Rand-Proj-Spatial is $O(kd)$, while the decoding time is $O(d^2 \cdot nk)$. The computation bottleneck in decoding is computing the eigendecomposition of the $d \times d$ matrix $\sum_{i=1}^{n} \boldsymbol{G}_i^T \boldsymbol{G}_i$ of rank at most nk. Improving the computation time for both the encoding and decoding schemes is an important direction for future work.

2) Perfect Shared Randomness. It is common to assume perfect shared randomness between the server and the clients in distributed settings [150]. However, to perfectly simulate randomness using Pseudo Random Number Generator (PRNG), at least $\log_2 d$ bits of the seed need to be exchanged in practice. We acknowledge this gap between theory and practice.

2.7 Conclusion

In this work, we propose the Rand-Proj-Spatial estimator, a novel encoding-decoding scheme, for communication-efficient distributed mean estimation. The proposed client-side encoding generalizes and improves the commonly used Rand-k sparsi-

fication, by utilizing projections onto general k-dimensional subspaces. On the server side, cross-client correlation is leveraged to improve the approximation error. Compared to existing methods, the proposed scheme consistently achieves better mean estimation error across a variety of tasks. Potential future directions include improving the computation time of Rand-Proj-Spatial and exploring whether the proposed Rand-Proj-Spatial achieves the optimal estimation error among the class of *non-adaptive* estimators, given correlation information. Furthermore, combining sparsification and quantization techniques and deriving such algorithms with the optimal communication cost-estimation error trade-offs would be interesting.

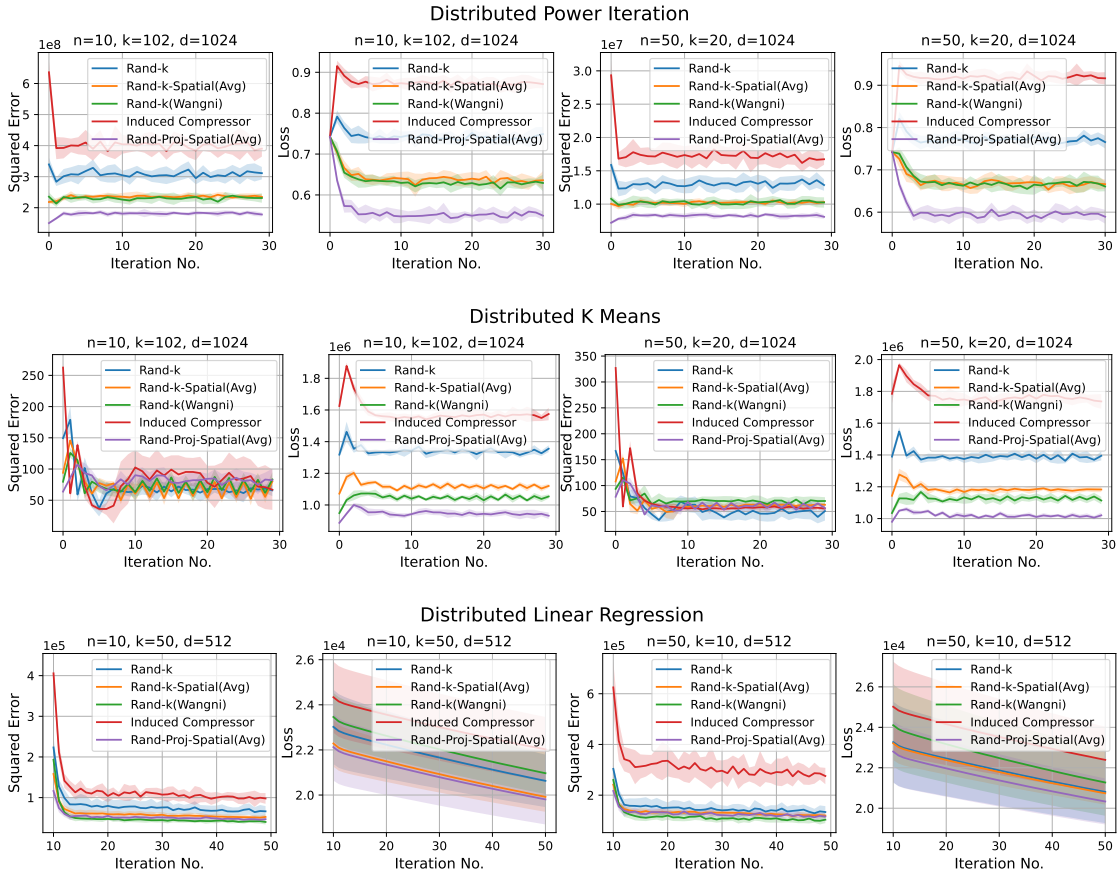

Figure 2.4: Experiment results on three distributed optimization tasks: distributed power iteration, distributed k-means, and distributed linear regression. The first two use the `Fashion-MNIST` dataset with the images resized to 32×32, hence $d = 1024$. Distributed linear regression uses `UJIndoor` dataset with $d = 512$. All the experiments are repeated for 10 random runs, and we report the mean as the solid lines, and one standard deviation using the shaded region. The violet line in the plots represents our proposed Rand-Proj-Spatial(Avg) estimator.

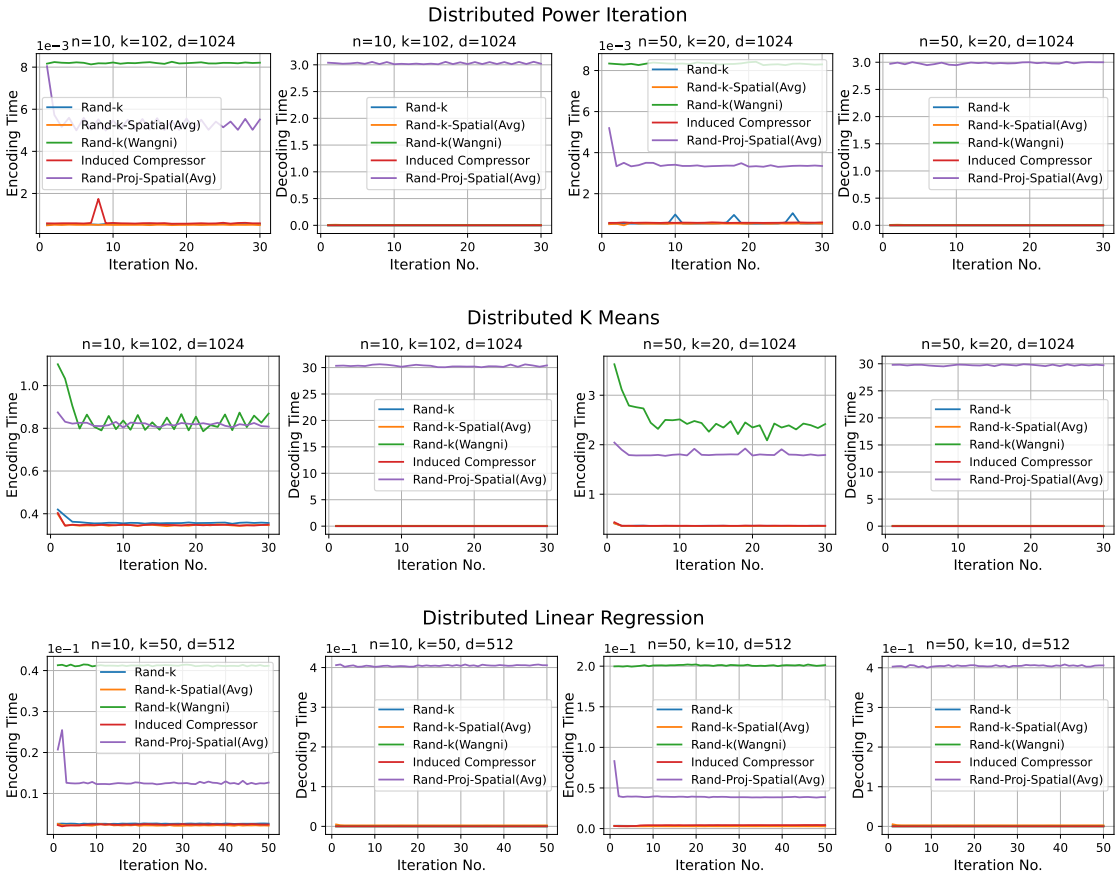

Figure 2.5: The corresponding wall-clock time to encode and decode client vectors (in seconds) using different sparsification schemes, across the three tasks.

Chapter 3

Differential Privacy: Private Majority Ensembling

This chapter is based on the following work:

> ***Shuli Jiang**, Richard Zhang, Gauri Joshi. **"Optimized Tradeoffs for Private Prediction with Majority Ensembling"**. Transactions on Machine Learning Research (TMLR), 2024. [paper]*

Beyond communication efficiency, another fundamental challenge in distributed learning is preserving the privacy of client data. Differential Privacy (DP) is a widely adopted framework to address this issue by providing formal privacy guarantees. However, achieving these guarantees typically comes at the cost of reduced model utility, leading to degraded performance in downstream tasks. Therefore, a key research goal is to improve the trade-off between privacy and model utility. In this chapter, we discuss how to improve such trade-offs in one specific learning problem.

Abstract. We study a classical problem in private prediction, the problem of computing an $(m\epsilon, \delta)$-differentially private majority of K (ϵ, Δ)-differentially private algorithms for $1 \leq m \leq K$ and $1 > \delta \geq \Delta \geq 0$. Standard methods such as subsampling or randomized response are widely used, but do they provide optimal privacy-utility tradeoffs? To answer this, we introduce the Data-dependent Randomized Response Majority (DaRRM) algorithm. It is parameterized by a data-dependent

noise function γ, and enables efficient utility optimization over the class of all private algorithms, encompassing those standard methods. We show that maximizing the utility of an $(m\epsilon, \delta)$-private majority algorithm can be computed tractably through an optimization problem for any $m \leq K$ by a novel structural result that reduces the infinitely many privacy constraints into a polynomial set. In some settings, we show that DaRRM provably enjoys a privacy gain of a factor of 2 over common baselines, with fixed utility. Lastly, we demonstrate the strong empirical effectiveness of our first-of-its-kind privacy-constrained utility optimization for ensembling labels for private prediction from private teachers in image classification. Notably, our DaRRM framework with an optimized γ exhibits substantial utility gains when compared against several baselines.

3.1 Introduction

Differential privacy (DP) is a widely applied framework for formally reasoning about privacy leakage when releasing statistics on a sensitive database [28, 37]. Differential privacy protects data privacy by obfuscating algorithmic output, ensuring that query responses look similar on adjacent datasets while preserving utility as much as possible [35].

Privacy in practice often requires aggregating or composing multiple private procedures that are distributed for data or training efficiency. For example, it is common to aggregate multiple private algorithmic or model outputs in methods such as boosting or calibration [103]. In federated learning, model training is distributed across multiple edge devices. Those devices need to send local information, such as labels or gradients [68], to an aggregating server, which is often honest but curious about the local training data. Hence, the output from each model at an edge device needs to be privatized locally before being sent to the server. When translating from a local privacy guarantee to a centralized one, one needs to reason about the composition of the local privacy leakage [86]. Therefore, we formally ask the following:

Problem 3.1.1 (Private Majority Ensembling (Illustrated in Figure 3.1))**.** *Consider $K \geq 1$ (ϵ, Δ)-differentially private mechanisms $M_1, \ldots, M_K$ for K odd. Given a dataset $\mathcal{D}$, each mechanism outputs a binary answer — that is, $M_i : \mathcal{D} \to \{0, 1\}$, $\forall i \in [K]$. Given a privacy* ***allowance*** *$1 \leq m \leq K$, $m \in \mathbb{R}$ and a failure probability*

$\delta \geq \Delta \geq 0$, $\delta, \Delta \in [0, 1)$, *how can one maximize the utility of an* $(m\epsilon, \delta)$*-differentially private mechanism* $\mathcal{A}$ *to compute the majority function* $g(S_1, S_2, \ldots, S_K)$*, where* $S_i \sim M_i(\mathcal{D})$*?*

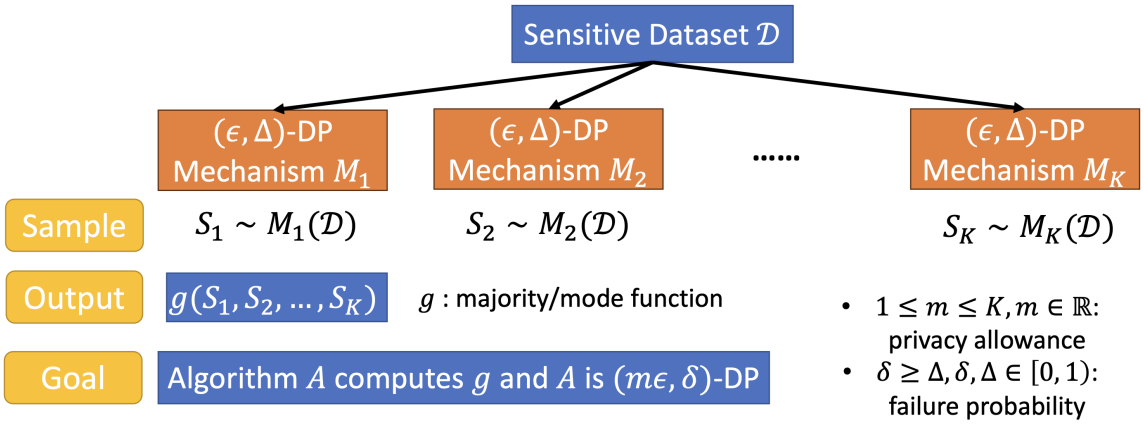

Figure 3.1: An illustration of the problem setting. The inputs are the dataset $\mathcal{D}$ and K (ϵ, Δ)-differentially private mechanisms $M_1, \ldots, M_K$. One draws samples $S_i \sim M_i(\mathcal{D})$ and computes an aggregated output $g(S_1, \ldots, S_K)$ based on all observed samples. Our goal is to design a randomized algorithm $\mathcal{A}$ that approximately computes g and is $(m\epsilon, \delta)$-differentially private for $1 \leq m \leq K$ and $\delta \geq \Delta \geq 0$. We focus on g being the majority function .

The majority function g is often used in private prediction, where one studies the privacy cost of releasing one prediction [33] and exploits the fact that releasing only the aggregated output on sharded models is significantly more private than releasing each prediction. For example, this occurs in semi-supervised knowledge transfer with private aggregated teacher ensembles (PATE) [94, 95], in ensemble learning algorithms [58, 140], machine unlearning [21], private distributed learning algorithms such as Stochastic Sign-SGD [139], and in ensemble feature selection [75]. Private prediction is also shown to be a competitive technique in data-adaptive settings, where the underlying dataset is changing slowly over time, to quickly adjust to online dataset updates [154]. Furthermore, to address the large privacy loss of private prediction under the many-query regime, there has been recent works in everlasting private prediction that extends privacy guarantees with repeated, possibly infinite, queries without suffering a linear increase in privacy loss [85, 112].

These works, however, rely often on the standard sensitivity analysis of g to provide

a private output and thus generally provide limited utility guarantees. This is because the maximum sensitivity of g can be too pessimistic in practice, as observed in the problem of private hyperparameter optimization [74]. On the other hand, for private model ensembling, a naive way to bound privacy loss without restrictive assumptions is to apply simple composition (Theorem 3.3.2) or general composition (Theorem 3.3.3, a tighter version compared to advanced composition) to reason about the final privacy loss after aggregation. A black-box application of the simple composition theorem to compute g would incur a $K\epsilon$ privacy cost in the pure differential privacy setting, that is, $\delta = 0$, or if one is willing to tolerate some failure probability δ, general composition would yield a $O(\sqrt{K}\epsilon)$ privacy cost [60]. Thus, a natural baseline algorithm $\mathcal{A}$ that is $(m\epsilon, m\Delta)$-differentially private applies privacy amplification by subsampling and randomly chooses m of the K mechanisms to aggregate and returns the majority of the subsampled mechanisms. This technique is reminiscent of the subsampling procedure used for the maximization function g [74] or some general techniques for privacy amplification in the federated setting via shuffling [38].

However, standard composition analysis and privacy amplication techniques can be suboptimal for computing a private majority, in terms of both utility and privacy. Observe that if there is a clear majority among the outputs of $M_1(\mathcal{D}), \ldots, M_K(\mathcal{D})$, one can add less noise. This is because each mechanism M_i is (ϵ, Δ)-differentially private already, and hence, is less likely to change its output on a neighboring dataset by definition. This implies the majority outcome is unlikely to change based on single isolated changes in $\mathcal{D}$. Furthermore, composition theorems make two pessimistic assumptions: 1) the worst-case function g and the dataset $\mathcal{D}$ are considered, and 2) all intermediate mechanism outputs $M_1(\mathcal{D}), \ldots, M_K(\mathcal{D})$ are released, rather than just the final aggregate. Based on these observations, is it possible then to improve the utility of computing a private majority, under a fixed privacy loss?

3.1.1 Our Contributions

We give a (perhaps surprising) affirmative answer to the above question by using our novel data-dependent randomized response framework (DaRRM), which captures all private majority algorithms, we introduce a tractable noise optimization procedure that maximizes the privacy-utility tradeoffs. Furthermore, we can provably achieve

a constant factor improvement in utility over simple subsampling by applying data-dependent noise injection when M_i's are i.i.d. and $\delta = 0$. To our knowledge, this is the first of its work of its kind that gives a tractable utility optimization over the possibly infinite set of privacy constraints.

Data-dependent Randomized Response Majority (DaRRM). We generalize the classical Randomized Response (RR) mechanism and the commonly used subsampling baseline for solving Problem 3.1.1 and propose a general randomized response framework DaRRM (see Algorithm 1), which comes with a customizable noise function γ. We show that DaRRM actually captures all algorithms computing the majority whose outputs are at least as good as a random guess (see Lemma 3.4.3), by choosing different γ functions.

Designing γ with Provable Privacy Amplification. The choice of the γ function in DaRRM allows us to explicitly optimize noise while trading off privacy and utility. Using structural observations, we show privacy amplification by a factor of 2 under mild conditions over applying simple composition in the pure differential privacy setting when the mechanisms M_i's are i.i.d. (see Theorem 3.5.1).

Finding the Best γ through Dimension-Reduced Optimization. We further exploit the generality of DaRRM by applying a novel optimization-based approach that applies constrained optimization to find a data-dependent γ that maximizes some measure of utility. One challenge is that there are infinitely many privacy constraints, which are necessary for DaRRM with the optimized γ to satisfy the given privacy loss. We show that we can reformulate the privacy constraints, which are infinite dimensional, to a finite polynomial-sized constraint set, allowing us to efficiently constrain the optimization problem to find the best γ, even for approximate differential privacy (see Lemma 3.6.1). Empirically, we show that with a small m and ϵ, the optimized γ (see γ_{opt} in Figure 3.2) achieves the best utility among all γ functions, even compared to the subsampling and the data-independent baseline. To our knowledge, this is the first utility maximization algorithm that optimizes over all private algorithms by constrained optimization with dimension reduction.

Experiments. In downstream tasks, such as semi-supervised knowledge transfer for private image classification, we compare our DaRRM with an optimized γ to compute the private label majority from private teachers against PATE [95], which computes the private label majority from non-private teachers. We fix the privacy

loss of the output of both algorithms to be the same and find that when the number of teachers K is small, DaRRM indeed has a higher utility than PATE, achieving 10%-15% and 30% higher accuracy on datasets `MNIST` and `Fashion-MNIST`, respectively.

3.2 Related Work

3.2.1 Private Composition

Blackbox privacy composition analysis often leads to pessimistic utility guarantees. In the blackbox composition setting, one can do no better than the $O(K\epsilon)$ privacy analysis for pure differential privacy [36]. For approximate differential privacy, previous work has found optimal constants for advanced composition by reducing to the binary case of hypothesis testing with randomized response; and optimal tradeoffs between ϵ, δ for black box composition are given in [60], where there could be a modest improvement 20%.

Thus, for specific applications, previous work has turned to white-box composition analysis for improved utility. This includes, for example, moment accountant for private SGD [2] and the application of contractive maps in stochastic convex optimization [40]. For the specific case of model ensembles, [95] shows a data-dependent privacy bound that vanishes as the probability of disagreement goes to 0. Their method provides no utility analysis but they empirically observed less privacy loss when there is greater ensemble agreement.

When g is the maximization function, some previous work shows that an approximately maximum value can be outputted with high probability while incurring $O(\epsilon)$ privacy loss, independently of K. [74] proposed a random stopping mechanism for $m = 1$ that draws samples uniformly at random from $M_i(\mathcal{D})$ at each iteration. In any given iteration, the sampling halts with probability γ and the final output is computed based on the samples collected until that time. This leads to a final privacy cost of only 3ϵ for the maximization function g, which can be improved to 2ϵ [93]. In addition to the aforementioned works, composing top-k and exponential mechanisms also enjoy slightly improved composition analysis via a bounded-range analysis [30, 32].

3.2.2 Bypassing the Global Sensitivity

To ensure differential privacy, it is usually assumed the query function g has bounded global sensitivity — that is, the output of g does not change much on *any* adjacent input datasets differing in one entry. The noise added to the output is then proportional to the global sensitivity of g. If the sensitivity is large, the output utility will thus be terrible due to a large amount of noises added. However, the worst case global sensitivity can be rare in practice, and this observation has inspired a line of works on designing private algorithms with data-dependent sensitivity bound to reduce the amount of noises added.

Instead of using the maximum global sensitivity of g on any dataset, the classical Propose-Test-Release framework of Dwork [34] uses a local sensitivity value for robust queries that is tested privately and if the sensitivity value is too large, the mechanism is halted before the query release. The halting mechanism incurs some failure probability but deals with the worst-case sensitivity situations, while allowing for lower noise injection in most average-case cases.

One popular way to estimate average-case sensitivity is to use the Subsample-and-Aggregate framework by introducing the notion of *perturbation stability*, also known as *local sensitivity* of a function g on a dataset $\mathcal{D}$ [36, 121], which represents the minimum number of entries in $\mathcal{D}$ needs to be changed to change $g(\mathcal{D})$. One related concept is *smooth sensitivity*, a measure of variability of g in the neighborhood of each dataset instance. To apply the framework under *smooth sensitivity*, one needs to privately estimate a function's local sensitivity L_s and adapt noise injection to be order of $O(\frac{L_s}{\epsilon})$, where L_s can often be as small as $O(e^{-n})$, where $n = |\mathcal{D}|$, the total dataset size [88]. Generally, the private computation of the smooth sensitivity of a blackbox function is nontrivial but is aided by the Subsample and Aggregate approach for certain functions.

These techniques hinge on the observation that a function with higher stability on $\mathcal{D}$ requires less noise to ensure worst case privacy. Such techniques are also applied to answer multiple online functions/queries in model-agnostic learning [14]. However, we highlight two key differences in our setting with a weaker stability assumption. First, in order to estimate the *perturbation stability* of g on $\mathcal{D}$, one needs to downsample or split $\mathcal{D}$ into multiple blocks [14, 36, 121], $\hat{\mathcal{D}}_1, \ldots, \hat{\mathcal{D}}_B$, and estimate the *perturbation*

stability based on the mode of $g(\hat{\mathcal{D}}_1), \ldots, g(\hat{\mathcal{D}}_B)$. This essentially reduces the amount of change in the output of g due to a single entry in $\mathcal{D}$, with high probability and replaces the hard-to-estimate *perturbation stability* of g with an easy-to-compute *perturbation stability* of the mode. Such a notion of stability has also been successfully applied, along with the sparse vector technique, for model-agnostic private learning to handle exponentially number of queries to a model [14]. Note that in these cases, since a private stochastic test is applied, one cannot achieve pure differential privacy [36]. In practice, e.g. federated learning, however, one does not have direct access to $\mathcal{D}$, and thus it is impractical to draw samples from or to split $\mathcal{D}$. Second, to ensure good utility, one relies on a key assumption, i.e. the *subsampling stability* of g, which requires $g(\hat{\mathcal{D}}) = g(\mathcal{D})$ with high probability over the draw of subsamples $\hat{\mathcal{D}}$.

Although our intuition in designing DaRRM also relies on the stability of the mode function g, previous usage of stability to improve privacy-utility tradeoffs, e.g., propose-test-release [36, 125], requires the testing of such stability, based on which one adds a larger (constant) noise γ. This can still lead to adding redundant noise in our case.

3.2.3 Optimal Randomized Response

[54] and [60] show that the classical Randomized Response (RR) mechanism with a constant probability of faithfully revealing the true answer is optimal in certain private estimation problems. Our proposed DaRRM framework and our problem setting is a generalized version of the ones considered in both [54] and [60], which not only subsumes RR but also enables a data-dependent probability, or noise addition.

While RR with a constant probability can be shown optimal in problems such as private count queries or private estimation of trait possession in a population, it is not optimal in other problems, such as private majority ensembling, since unlike the former problems, changing one response of the underlying mechanisms does not necessarily change the output of the majority. To explicitly compute the minimum amout of noise required, one needs the output distributions of the underlying mechanisms but this is unknown. To resolve this, our proposed DaRRM framework adds the amount of noise dependent on the set of observed outcomes from the underlying private mechanisms, $\mathcal{S}$, which is a random variable of the dataset and is hence a proxy. This

enables DaRRM to calibrate the amount of noise based on whether the majority output is likely to change. The amount of noise is automatically reduced when the majority output is not likely to change.

Second, [54] and [60] both consider a special case in our setting where all K private mechanisms are i.i.d., while our approach focuses on the more general setting where each private mechanism can have a different output distribution.

3.2.4 Learning A Good Noise Distribution

There have been limited works that attempt to derive or learn a good noise distribution that improves the utility. For deep neural networks inference, [81] attempts to learn the best noise distribution to maximizing utility subject to an entropy Lagrangian, but no formal privacy guarantees were derived. For queries with bounded sensitivity, [46] demonstrate that the optimal noise distribution is in fact a staircase distribution that approaches the Laplacian distribution as $\epsilon \to 0$.

3.2.5 Private Prediction

Instead of releasing a privately trained model as in private learning, private prediction hides the models and only releases private outputs. Private prediction has been shown as a practical alternative compared to private learning, as performing private prediction is much easier compared to private learning on a wide range of tasks [33, 85, 126]. Although a privately trained model can make infinitely many predictions at the inference time without incurring additional privacy loss, since differential privacy is closed under post-processing, it has been shown recently that it is indeed possible to make infinitely many private predictions [85] with a finite privacy loss for specific problems.

3.3 Preliminaries

We first introduce the definition of differential privacy, simple composition and general composition as follows. The general composition [60] gives a near optimal and closed-form bound on privacy loss under adaptive composition, which improves upon advanced composition [36].

Definition 3.3.1 (Differential Privacy (DP) [36]). *A randomized mechanism $\mathcal{M} : \mathcal{D} \to \mathcal{R}$ with a domain $\mathcal{D}$ and range $\mathcal{R}$ satisfies (ϵ, δ)-differential privacy for $\epsilon, \delta \geq 0$ if for any two* **adjacent datasets** *$\mathcal{D}, \mathcal{D}'$ and for any subset of outputs $S \subseteq \mathcal{R}$ it holds that $\Pr[\mathcal{M}(\mathcal{D}) \in S] \leq e^{\epsilon} \Pr[\mathcal{M}(\mathcal{D}') \in S] + \delta$. $\delta = 0$ is often called pure differential privacy; while $\delta > 0$ is often called approximate differential privacy.*

Theorem 3.3.2 (Simple Composition [36]). *For any $\epsilon > 0$ and $\delta \in [0, 1]$, the class of (ϵ, δ)-differentially private mechanisms satisfy $(k\epsilon, k\delta)$-differential privacy under k-fold adaptive composition.*

Theorem 3.3.3 (General Composition (Theorem 3.4 of [60])). *For any $\epsilon > 0, \delta \in [0, 1]$ and $\delta' \in (0, 1]$, the class of (ϵ, δ)-differentially private mechanisms satisfies $(\epsilon', 1 - (1 - \delta)^k(1 - \delta'))$-differential privacy under k-fold adaptive composition for*

$$\epsilon' = \min\left\{k\epsilon, \frac{(e^{\epsilon} - 1)\epsilon k}{e^{\epsilon} + 1} + \epsilon\sqrt{2k\log(e + \frac{\sqrt{k\epsilon^2}}{\delta'})}, \frac{(e^{\epsilon} - 1)\epsilon k}{e^{\epsilon} + 1} + \epsilon\sqrt{2k\log(1/\delta')}\right\}$$

We then formalize the error and utility metric in our problem as follows:

Definition 3.3.4 (Error Metric and Utility Metric). *For the problem setting in Definition 3.1.1, let the observed (random) outcomes set be $\mathcal{S} = \{S_1, .., S_k\}$, where $S_i \sim M_i(\mathcal{D})$. For a fixed $\mathcal{D}$, we define the error of an algorithm $\mathcal{A}$, i.e., $\mathcal{E}(\mathcal{A})$, in computing the majority function g as the Total Variation (TV) distance between $g(\mathcal{S})$ and $\mathcal{A}(\mathcal{D})$. Specifically,*

$$\mathcal{E}(\mathcal{A}) = \mathcal{D}_{TV}(g(\mathcal{S}) \parallel \mathcal{A}(\mathcal{D})) = |\Pr[\mathcal{A}(\mathcal{D}) = 1] - \Pr[g(\mathcal{S}) = 1]|$$

and the utility is defined as $1 - \mathcal{E}(\mathcal{A})$.

Notation. Throughout the paper, we use the same notations defined in Problem 3.1.1 and Definition 3.3.4. Furthermore, let $\mathcal{D}$ and $\mathcal{D}'$ to denote a pair of adjacent datasets with one entry being different. Also, let $p_i = \Pr[M_i(\mathcal{D}) = 1]$ and $p_i' = \Pr[M_i(\mathcal{D}') = 1]$, $\forall i \in [K]$. We omit the subscript i when all p_i's or p_i''s are equal. $\mathbb{I}\{\cdot\}$ denotes the indicator function and $[K] = \{1, 2, \ldots, K\}$. For the purpose of analysis, let $\mathcal{L}(\mathcal{D}) = \sum_{i=1}^{K} M_i(\mathcal{D}) \in \{0, 1, \ldots, K\}$, i.e. the (random) sum of all observed outcomes on dataset $\mathcal{D}$. $\mathcal{D}$ is omitted when the context is clear. Unless specified, we use the noise function $\gamma : \{0, 1, \ldots, K\} \to [0, 1]$ as input to our

algorithms to calibrate the probabilistic noise injection. Unless specified, the privacy allowance $m \in \mathbb{R}$.

3.4 Private Majority Algorithms

3.4.1 Randomized Response (RR)

The very first approach to consider when solving private majority ensembling (Problem 3.1.1), since the output is binary, is the classical Randomized Response (RR) mechanism [36], where one flips a biased coin with a *constant* probability $p_{const} \in [0, 1]$. If the coin lands on head with probability p_{const}, output the true majority base on K samples; if not, then simply output a noisy random answer. However, to make the output $(m\epsilon, \delta)$-differential private, the success probability p_{const} can be at most $O(\frac{m}{K})$ (or $O(\frac{m}{\sqrt{K}})$) when $\delta = 0$ (or $\delta > 0$) (see Appendix B.1.1), which is too small for any reasonable utility.

The key observation for improved utility is that the probability of success should not be a *constant*, but should depend on the *unpublished* set of observed outcomes from the mechanisms $\mathcal{S}$. If we see many 1's or 0's in $\mathcal{S}$, then there should be a clear majority even on adjacent datasets. On the other hand, if we see about half 1's and half 0's, this means the majority is highly volatile to data changes, which implies we need more noise to ensure privacy. In summary, if we can calibrate the success probability based on $\mathcal{S}$ to smoothly increase when there is a clear majority, we can improve the utility without affecting privacy.

3.4.2 Subsampling

One natural baseline is outputting the majority of m out of K randomly subsampled mechanisms (without replacement), given a privacy allowance $m \in [K]$. Suppose $\delta \geq m\Delta$, the privacy loss of the aggregated output can be reasoned through simple composition or general composition. Interestingly, we show outputting the majority of m out of K subsampled mechanisms corresponds to RR with a *non-constant* probability $p_\gamma = \gamma_{Sub}(\mathcal{L}(\mathcal{D}))$, which is set by a polynomial function $\gamma_{Sub} : \{0, \ldots, K\} \to [0, 1]$ based on the sum of observed outcomes $\mathcal{L}(\mathcal{D})$ in Lemma 3.4.1 (see a full proof in

Appendix B.1.2). Intuitively, subsampling may be seen as implicitly adding noise by only outputting based on a randomly chosen subset of the mechanisms; therefore this implicit noise is inherently *data-dependent* on $\mathcal{L}(\mathcal{D})$.

Lemma 3.4.1. *Consider Problem 3.1.1, with the privacy allowance $m \in [K]$. Consider the data-dependent algorithm that computes $\mathcal{L}(\mathcal{D})$ and then applies RR with probability p_γ. If $p_\gamma = \gamma_{Sub}(l)$, where $l \in \{0, 1, \ldots, K\}$ is the value of $\mathcal{L}(\mathcal{D})$, i.e., the (random) sum of observed outcomes on dataset $\mathcal{D}$, and $\gamma_{Sub} : \{0, 1, \ldots, K\} \to [0, 1]$ is*

$$\gamma_{Sub}(l) = \gamma_{Sub}(K - l) = \begin{cases} 1 - 2\sum_{j=\frac{m+1}{2}}^{m} \frac{\binom{l}{j}\binom{K-l}{m-j}}{\binom{K}{m}} & \text{if } m \text{ is odd} \\ 1 - 2\sum_{j=\frac{m}{2}+1}^{m} \frac{\binom{l}{j}\binom{K-l}{m-j}}{\binom{K}{m}} - \frac{\binom{l}{\frac{m}{2}}\binom{K-l}{\frac{m}{2}}}{\binom{K}{m}} & \text{if } m \text{ is even} \end{cases}$$

then the majority of m out of K subsampled mechanisms without replacement and the output of our data-dependent RR algorithm have the same distribution.

One thing special about subsampling is that when $m = 1$, it indeed results in the optimal error, which we show in Lemma 3.4.2 as follows. See a full proof in Appendix B.1.3. Note that when $m = 1$, subsampling outputs a majority of 1 with probability exactly $\frac{1}{K}\sum_{i=1}^{K} p_i$.

Lemma 3.4.2 (Lower Bound on Error when $m = 1$)**.** *Let $\mathcal{A}$ be an (ϵ, δ)-differentially private algorithm, where $0 \leq \epsilon < c$ for some constant $c > 0$ and $\delta \in [0, \frac{1}{2})$, that computes the majority of K (ϵ, δ)-differentially private mechanisms $M_1, \ldots, M_K$, where $M_i : \mathcal{D} \to \{0, 1\}$ on dataset $\mathcal{D}$ and $\Pr[M_i(\mathcal{D}) = 1] = p_i, \forall i \in [K]$. Then, the error $\mathcal{E}(\mathcal{A}) \geq |\Pr[g(\mathcal{S}) = 1] - \frac{1}{K}\sum_{i=1}^{K} p_i|$, where $g(\mathcal{S})$ is the probability of the true majority output being 1 as defined in Definition 3.1.1.*

3.4.3 Data-dependent Randomized Response (DaRRM)

Does subsampling give optimal utility when $m > 1$? Inspired by the connection between RR and subsampling, we propose Data-dependent Randomized Response Majority (DaRRM) in Algorithm 1, to study optimizing privacy-utility tradeoffs in private majority ensembling. In particular, DaRRM has a *non-constant* success probability p_γ that is set by a parameterized noise function γ, which in turn depends on the set of observed outcomes $\mathcal{S} = \{S_1, \ldots, S_K\}$. In fact, we can show that DaRRM is general: any *reasonable* algorithm $\mathcal{A}$, name one whose output is at least as good as

Algorithm 1 DaRRM$(\cdot)$: Data-dependent Randomized Response Majority

Input: K (ϵ, Δ)-DP mechanisms $\{M_i\}_{i=1}^K$, noise function $\gamma : \{0,1\}^{K+1} \to [0,1]$ (in our specific setting $\gamma : \{0,1,\ldots,K\} \to [0,1]$), dataset $\mathcal{D}$, privacy allowance $1 \leq m \leq K$, failure probability $\delta \geq \Delta \geq 0$
Output: $(m\epsilon, \delta)$-DP majority vote of $\{M_i\}_{i=1}^K$
$\mathcal{S} = \{S_1, .., S_K\}$, where $S_i \sim M_i(\mathcal{D})$
$\mathcal{L} = \sum_{i=1}^K S_i$
Set probability $p_\gamma \leftarrow \gamma(\mathcal{S})$ (in our setting $p_\gamma \leftarrow \gamma(\mathcal{L})$)
Flip the p_γ- biased coin
if Head (with probability p_γ) **then**
    Output $\mathbb{I}\{\frac{1}{K}\mathcal{L} \geq \frac{1}{2}\}$
else
    Output 0/1 with equal probability
end if

a random guess, can be captured by the DaRRM framework in Lemma 3.4.3 (see a full proof in Appendix B.1.4). We denote DaRRM instantiated with a specific noise function γ by DaRRM$_\gamma$.

Lemma 3.4.3 (Generality of DaRRM)**.** *Let $\mathcal{A}$ be any randomized algorithm to compute the majority function g on $\mathcal{S}$ such that for all $\mathcal{S}$, $\Pr[\mathcal{A}(\mathcal{S}) = g(\mathcal{S})] \geq 1/2$ (i.e. $\mathcal{A}$ is at least as good as a random guess). Then, there exists a a general function $\gamma : \{0,1\}^{K+1} \to [0,1]$ such that if one sets p_γ by $\gamma(\mathcal{S})$ in DaRRM, the output distribution of DaRRM$_\gamma$ is the same as the output distribution of $\mathcal{A}$.*

Designing the γ Function. With the DaRRM framework, we ask: how to design a good γ function that maximizes the utility? First, we introduce two characteristics of γ that do not affect the utility, while simplifying the analysis and the empirical optimization:

(a) **A function of the sum of observed samples**: Since the observed samples set $\mathcal{S}$ is a permutation-invariant set, a sufficient statistic that captures the full state of $\mathcal{S}$ is $\mathcal{L} = \sum_{i=1}^K S_i$, the sum of observed outcomes. This allows us to reduce $\gamma(\mathcal{S}) = \gamma(\mathcal{L})$. *Hence, in the rest of the paper, we focus on $\gamma : \{0,1,\ldots,K\} \to [0,1]$.*

(b) **Symmetric around $\frac{K}{2}$**: If γ is asymmetric, we can symmetrize by reflecting one region about $\frac{K}{2}$ and achieve better or equal expected utility, where the utility is summed over symmetric distributions of p_i.

Note that γ_{Sub} satisfies both characteristics. Now, recall $\mathcal{L}(\mathcal{D})$ and $\mathcal{L}(\mathcal{D}')$ are the sum of observed outcomes on adjacent datasets $\mathcal{D}$ and $\mathcal{D}'$. Also, recall $p_i = \Pr[M_i(\mathcal{D}) = 1]$ and $p_i' = \Pr[M_i(\mathcal{D}') = 1]$ are the output probabilities of the mechanism M_i on $\mathcal{D}, \mathcal{D}'$. To design a good noise function γ in DaRRM, we start by deriving conditions for a γ function such that DaRRM_γ is $(m\epsilon, \delta)$-differentially private in Lemma 3.4.4 (see a full proof in Appendix B.1.5).

Lemma 3.4.4 (γ privacy condition)**.** *Consider using DaRRM (Algorithm 1) to solve Problem 3.1.1, let $\alpha_l = \Pr[\mathcal{L}(\mathcal{D}) = l]$ and $\alpha_l' = \Pr[\mathcal{L}(\mathcal{D}') = l]$, where $\mathcal{D}$ and $\mathcal{D}'$ are adjacent datasets and $l \in \{0, \dots, K\}$. For a noise function $\gamma : \{0, 1, \dots, K\} \to [0, 1]$ such that $\gamma(l) = \gamma(K - l), \forall l$, DaRRM_γ is $(m\epsilon, \delta)$-differentially private if and only if for all α_l, α_l', the following holds,*

$$f(p_1, \dots, p_K, p_1', \dots, p_K'; \gamma) \leq e^{m\epsilon} - 1 + 2\delta \tag{3.1}$$

where f is called the ***privacy cost objective*** *and*

$$f(p_1, \dots, p_K, p_1', \dots, p_K'; \gamma) := \sum_{l=0}^{\frac{K-1}{2}} (e^{m\epsilon}\alpha_l' - \alpha_l) \cdot \gamma(l) + \sum_{l=\frac{K+1}{2}}^{K} (\alpha_l - e^{m\epsilon}\alpha_l') \cdot \gamma(l)$$

3.5 Provable Privacy Amplification

We theoretically demonstrate that privacy is provably amplified under improved design of γ in our DaRRM framework. Specifically, we show when the mechanisms are i.i.d. and $\delta = 0$, we gain privacy amplification by a factor of 2 compared to the naïve subsampling baseline by carefully designing γ.

Theorem 3.5.1 (Provable Privacy Amplification by 2)**.** *Consider using DaRRM (Algorithm 1) to solve Problem 3.1.1, with i.i.d. mechanisms $\{M_i\}_{i=1}^K$, i.e., $p_i = p$, $p_i' = p'$, $\forall i \in [K]$, the privacy allowance $m \in [K]$ and $\delta = \Delta = 0$. Let the noise function $\gamma : \{0, 1, \dots, K\} \to [0, 1]$ be that:*

if $m \geq \frac{K+1}{2}$, $\gamma(l) = 1$ *and if* $m \leq \frac{K-1}{2}$,

$$\gamma(l) = \begin{cases} 1 - 2h(l) & \forall l \leq \frac{K-1}{2} \\ 2h(l) - 1 & \forall l \geq \frac{K+1}{2} \end{cases}$$

where $h(l) = \sum_{i=m}^{2m-1} \frac{\binom{l}{i}\binom{K-l}{2m-1-i}}{\binom{K}{2m-1}}$, *then* DaRRM_γ *is* $m\epsilon$*-differentially private.*

3.5.1 Interpretation

First, when $m \leq \frac{K-1}{2}$ is small, the $\gamma(l)$ in Theorem 3.5.1 corresponds to outputting the majority based on subsampling $2m-1$ outcomes, from Lemma 3.4.1. However, the subsampling baseline, whose privacy loss is reasoned through simple composition, would have indicated that one can only output the majority based on m outcomes, therefore implying a 2x privacy gain. When $m \geq \frac{K+1}{2}$, the above theorem indicates that we can set a constant $\gamma = 1$, which implies we are optimally outputting the true majority with no noise while still surprisingly ensuring $m\epsilon$ privacy.

3.5.2 Intuition

This 2x privacy gain is intuitively possible because the majority is only dependent on half of the mechanisms' outputs, therefore the privacy leakage is also halved. To see this, we start by analyzing the privacy cost objective in Eq. B.12, where with a careful analysis of its gradient, we show that the maximum indeed occurs $(p^*, p'^*) = (0, 0)$ when γ satisfies certain conditions. Now, when $(p^*, p'^*) \to 0$, note that the probability ratio of outputting 1 with $2m-1$ outcomes is approximately $e^{m\epsilon}$, where dependence on m follows because the probability of outputting 1 is dominated by the probability that exactly m mechanisms output 1. To rigorize this, we derive sufficient conditions for γ functions that satisfy $\max_{(p,p')} f(p, p'; \gamma) = f(0, 0; \gamma) \leq e^{m\epsilon} - 1$ as indicated by Lemma 3.4.4, to ensure DaRRM to be $m\epsilon$-differentially private and a more detailed overview and the full proof can be found in Appendix B.2.

3.6 Optimizing the Noise Function γ in DaRRM

Theoretically designing γ and extending privacy amplification results to the $\delta > 0$ case is difficult and it is likely that our crafted γ is far from optimal. On the other hand, one can optimize for such γ^* that maximizes the utility but this involves solving a "Semi-infinite Programming" problem, due to the infinitely many privacy constraints, which are the constraints in the optimization problem necessary to ensure DaRRM with the optimized γ satisfy a given privacy loss. Solving a "Semi-infinite Programming" problem in general is non-tractable, but we show that in our specific setting this is in fact tractable, proposing a novel learning approach based on DaRRM that can optimize the noise function to maximize the utility. To the best of our knowledge, such optimization, presented as follows, is the first of its kind:

$$\min_{\gamma \in [0,1]^{K+1}} \mathbb{E}_{p_1,p_2,\ldots,p_K \sim \mathcal{T}}[\mathcal{E}(\mathsf{DaRRM}_\gamma)] \tag{3.2}$$

$$\text{s.t.} \max_{\{(p_i,p_i') \in \mathcal{F}_i\}_{i=1}^K} f(p_1,\ldots,p_K,p_1',\ldots,p_K';\gamma) \leq e^{m\epsilon} - 1 + 2\delta \tag{3.3}$$

$$\gamma(l) = \gamma(K-l), \forall l \in \{0,1,\ldots,K\}$$

where f is the privacy cost objective as defined in Lemma 3.4.4, $\mathcal{F}_i$ is the feasible region where (p_i, p_i') lies due to each mechanism M_i being ϵ-differentially private. Observe that since γ is symmetric around $\frac{K}{2}$, we only need to optimize $\frac{K+1}{2}$ variables instead of $K+1$ variables. $\mathcal{T}$ is the distribution from which $p_1,\ldots,p_K$ are drawn. We want to stress that no prior knowledge about the dataset or the amount of consensus among the private mechanisms is required to use our optimization framework. When there is no prior knowledge about $p_1,\ldots,p_K$, $\mathcal{T}$ is set to be the uniform distribution for maximizing the expected utility. Note the above optimization problem also enables the flexibility of incorporating prior knowledge about the mechanisms by choosing a prior distribution $\mathcal{T}$ to further improve the utility.

3.6.1 Optimizing Over All Algorithms

We want to stress that by solving the above optimization problem, we are indeed optimizing over *all* algorithms for maximal utility, since we show in Lemma 3.4.3 DaRRM that captures *all reasonable* algorithms computing a private majority.

3.6.2 Linear Optimization Objective

Perhaps surprisingly, it turns out that optimizing for γ^* is a Linear Programming (LP) problem! Indeed, after expanding the optimization objective in Eq. 3.2 by the utility definition (see Definition 3.3.4), optimizing the above objective is essentially same as optimizing:

$$\min_{\gamma\in[0,1]^{K+1}} -\frac{1}{2}\sum_{l=\frac{K+1}{2}}^{K} \mathbb{E}_{p_1,p_2,\ldots,p_K\sim\mathcal{T}}\left[(\alpha_l - \alpha_{K-l})\right]\cdot\gamma(l)$$

where $\alpha_l = \Pr[\mathcal{L}(\mathcal{D}) = l], \forall l \in \{0,1,\ldots,K\}$ and observe $\mathcal{L}(\mathcal{D}) \sim \text{PoissonBinomial}(p_1,\ldots,p_K)$. The above objective is linear in γ. See a full derivation in Appendix B.3.1.

Although taking the expectation over $p_1,\ldots,p_K$ involves integrating over K variables and this can be computationally expensive, we discuss how to formulate a computationally efficient approximation of the objective in Appendix B.3.2, which we later use in the experiments. Note that the objective only for maximizing the utility and hence approximating the objective does not affect the privacy guarantee.

3.6.3 Reducing Infinitely Many Constraints to A Polynomial Set

The constraints in the optimization problem (Eq. 3.3) is what makes sure the output of DaRRM_γ is $m\epsilon$-differentially private. We thus call them *the privacy constraints*. Note that the privacy constraints are linear in γ.

Though it appears we need to solve for infinitely many such privacy constraints since p_i's and p_i''s are continuous, we show that through a structural understanding of DaRRM, we can reduce the number of privacy constraints from infinitely many to exponentially many, and further to a polynomial set. First, we observe

the privacy cost objective f is linear in each independent pair of (p_i, p_i') fixing all (p_j, p_j'), $\forall j \neq i$, and hence finding the worst case probabilities in (p_i, p_i') given any γ, $(p_i^*, p_i'^*) = \text{argmax}_{(p_i,p_i')} f(p_1, \ldots, p_K, p_1', \ldots, p_K'; \gamma)$ is a linear programming (LP) problem. Furthermore, since p_i and p_i' are the probability of outputting 1 from the i-th (ϵ, Δ)-differentially private mechanism M_i on adjacent datasets, by definition, they are close and lie in a feasible region $\mathcal{F}_i$, which we show has 8 corners if $\delta > 0$ (and only 4 corners if $\delta = 0$). This implies $(p_i^*, p_i'^*)$ only happens at one of the corners of $\mathcal{F}_i$, and hence the number of constraints reduces to K^8 (and K^4 if $\delta = 0$). Second, observe that α_l and α_l' in the privacy cost objective f are the pmf of two Poisson Binomial distributions at $l \in \{0, \ldots, K\}$. Notice that the Poisson Binomial is invariant under the permutation of its parameters, i.e. PoissonBinomial$(p_1, \ldots, p_K)$ has the same distribution as PoissonBinomial$(\pi(p_1, \ldots, p_K))$, under some permutation π. Based on this observation, we show the number of constraints can be further reduced to $O(K^7)$ if $\delta > 0$ (and $O(K^3)$ if $\delta = 0$). We formalize the two-step reduction of the number of privacy constraints in Lemma 3.6.1 as follows. See a full proof in Appendix B.3.3. [1]

Lemma 3.6.1. *Consider using DaRRM (Algorithm 1) to solve Problem 3.1.1 and let f be the privacy cost objective as defined in Lemma 3.4.4. Given an arbitrary noise function γ, let the worst case probabilities be $(p_1^*, \ldots, p_K^*, p_1'^*, \ldots, p_K'^*) = \text{argmax}_{\{(p_i,p_i')\}_{i=1}^K} f(p_1, \ldots, p_K, p_1', \ldots, p_K'; \gamma)$.*

$$(p_1^*, \ldots, p_K^*, p_1'^*, \ldots, p_K'^*) = \underset{\{(p_i,p_i')\}_{i=1}^K}{\text{argmax}} f(p_1, \ldots, p_K, p_1', \ldots, p_K'; \gamma)$$

Then, each pair $(p_i^*, p_i'^*), \forall i \in [K]$ satisfies

$$(p_i^*, p_i'^*) \in \{(0,0), (1,1), (0,\Delta), (\Delta,0), (1-\Delta,1), (1,1-\Delta), (\frac{e^\epsilon+\Delta}{e^\epsilon+1}, \frac{1-\Delta}{e^\epsilon+1}), (\frac{1-\Delta}{e^\epsilon+1}, \frac{e^\epsilon+\Delta}{e^\epsilon+1})\}$$

Furthermore, when $\delta > 0$, there exists a finite vector set $\mathcal{P}$ of size $O(K^7)$ such that if $\beta = \max_{\{(p_i,p_i')\}_{i=1}^K \in \mathcal{P}} f(p_1, \ldots, p_K, p_1', \ldots, p_K'; \gamma)$, then $f(p_1^*, \ldots, p_K^*, p_1'^*, \ldots, p_K'^*; \gamma) \leq$

[1] **Practical Limitation.** Although the number of constraints is polynomial in K and optimizing γ in DaRRM is an LP, $O(K^7)$ can still make the number of constraints intractably large when K is large. In practice, we observe with the `Gurobi` optimizer, one can optimize γ for $K \leq 41$ on a laptop if $\delta > 0$. But if $\delta = 0$, since the number of privacy constraints is $O(K^3)$, one can optimize for K over 100.

β. When $\delta = 0$, the size of $\mathcal{P}$ can be reduced to $O(K^3)$.

3.7 Experiments

We empirically solve the above optimization problem (Eq. 3.2) using the `Gurobi`[2] solver and first present the shape of the optimized γ function, which we call γ_{opt}, and its utility in Section 3.7.1. Then, we demonstrate the compelling effectiveness of DaRRM with an optimized γ function, i.e., $\mathsf{DaRRM}_{\gamma_{opt}}$, in ensembling labels for private prediction from private teachers through the application of semi-supervised knowledge transfer for private image classification in Section 3.7.2.

3.7.1 Optimized γ in Simulations

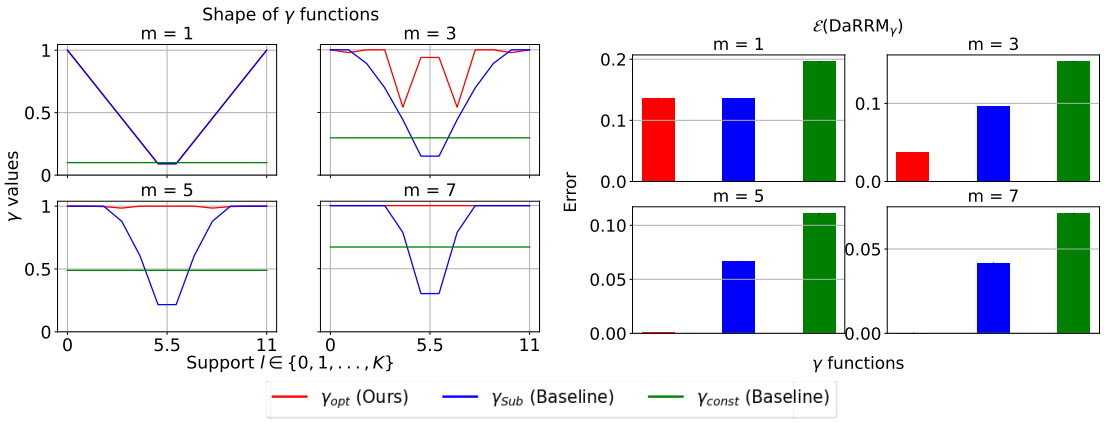

Figure 3.2: Plots of the shape and $\mathcal{E}(\mathsf{DaRRM}_\gamma)$ of different γ functions: the optimized γ_{opt}, and the baselines γ_{Sub} (corresponding to subsampling) and γ_{const} (corresponding to RR). Here, $K = 11, m \in \{1, 3, 5, 7\}$, $\epsilon = 0.1$, $\Delta = 10^{-5}$ and $\delta = 1-(1-\Delta)^m \approx m\Delta$.

We compare the shape and the error $\mathcal{E}(\mathsf{DaRRM}_\gamma)$ of different γ functions: an optimized γ_{opt} and the subsampling γ_{Sub} as in Lemma 3.4.1[3]. We also compare against

[2] https://www.gurobi.com/

[3] Note the subsampling mechanism from Section 3.5, which enjoys a privacy amplification by a factor of 2, only applies to pure differential privacy settings (i.e., when $\Delta = \delta = 0$). However, we focus on the more general approximate differential privacy settings (with $\Delta > 0$) in the experiments, and hence, the subsampling baseline we consider throughout this section is the basic version without privacy amplification. To see how the subsampling mechanism from Section 3.5 with privacy amplification compares against the other algorithms, please refer to Appendix B.4.

p_{const} in the classical baseline RR (see Section B.1.1) and $\mathcal{E}(\mathsf{RR})$. Here, p_{const} can be viewed as a constant noise function $\gamma_{const}(l) = p_{const}, \forall l \in \{0, 1, \ldots, K\}$; and $\mathcal{E}(\mathsf{RR})$ is the same as $\mathcal{E}(\mathsf{DaRRM}_{\gamma_{const}})$.

We present the results with $K = 11, \epsilon = 0.1, \Delta = 10^{-5}$ and $m \in \{1, 3, 5, 7\}$. We assume there is no prior knowledge about the mechanisms $\{M_i\}_{i=1}^K$, and set the prior distribution from which p_i's are drawn, $\mathcal{T}$, to be the uniform distribution, in the optimization objective (Eq. 3.2) searching for γ_{opt}. To ensure a fair comparison against the subsampling baseline, we set δ to be the one by m-fold general composition (see Theorem 3.3.3), which in this case, is $\delta = 1 - (1 - \Delta)^m \approx m\Delta$. We plot each γ functions over the support $\{0, 1, \ldots, K\}$ and the corresponding error of each algorithm in Figure 3.2.

Discussion

In summary, at $m = 1$, the optimized noise function γ_{opt} overlaps with γ_{sub} which corresponds to the subsampling baseline. This agrees with our lower bound on the error in Lemma 3.4.2, which implies that at $m = 1$, subsampling indeed gives the optimal error. When $m > 1$, the optimized noise function γ_{opt} has the highest probability of outputting the true majority over the support than the γ functions corresponding to the baselines. This implies $\mathsf{DaRRM}_{\gamma_{opt}}$ has the lowest error (and hence, highest utility), which is verified on the bottom set of plots. More results on comparing the $\mathsf{DaRRM}_{\gamma_{opt}}$ optimized under the uniform $\mathcal{T}$ against the baselines by general composition (Theorem 3.3.3) and in pure differential privacy settings (i.e., $\Delta = \delta = 0$) for large K and m can be found in Appendix B.4 and B.4. Furthermore, we include results optimizing γ using a non-uniform $\mathcal{T}$ prior in Appendix B.4.

3.7.2 Private Semi-Supervised Knowledge Transfer

Semi-supervised Knowledge Transfer

We apply our DaRRM framework in the application of semi-supervised knowledge transfer for private image classification. We follow a similar setup as in PATE [94, 95], where one trains K teachers, each on a subset of a sensitive dataset, and at the inference time, queries the teachers for the majority of their votes, i.e., the predicted

Dataset	MNIST			Dataset	Fashion-MNIST		
# Queries	GNMax (Baseline)	DaRRM$_{\gamma_{Sub}}$ (Baseline)	DaRRM$_{\gamma_{opt}}$ (Ours)	# Queries	GNMax (Baseline)	DaRRM$_{\gamma_{Sub}}$ (Baseline)	DaRRM$_{\gamma_{opt}}$ (Ours)
$Q = 20$	0.63 (0.09)	0.76 (0.09)	**0.79 (0.09)**	$Q = 20$	0.65 (0.11)	0.90 (0.07)	**0.96 (0.03)**
$Q = 50$	0.66 (0.06)	0.75 (0.06)	**0.79 (0.05)**	$Q = 50$	0.59 (0.06)	0.94 (0.03)	**0.96 (0.02)**
$Q = 100$	0.64 (0.04)	0.76 (0.04)	**0.80 (0.04)**	$Q = 100$	0.64 (0.04)	0.93 (0.02)	**0.96 (0.02)**

Table 3.1: Accuracy of the predicted labels of Q query samples on datasets `MNIST` (on the left) and `Fashion-MNIST` (on the right). We report the mean and one std. in parentheses over 10 random draws of the query samples from the test dataset. Note each prediction on the query sample is $(\epsilon_{query}, \delta_{query})$-differentially private. With the same per query privacy loss (and hence the same total privacy loss over Q samples), DaRRM$_{\gamma_{opt}}$ achieves the highest accuracy compared to the other two baselines.

labels, of a test sample. Each time the teachers are queried, there is a privacy loss, and we focus on this private prediction subroutine in this section. To limit the total privacy loss over all queries, the student model is also trained on a public dataset without labels. The student model queries the labels of a small portion of the samples in this dataset from the teachers and is then trained using semi-supervised learning algorithms on both the labeled and unlabeled samples from the public dataset.

Baselines

We want the privacy loss per query of a test sample to the teachers to be $(\epsilon_{query}, \delta_{query})$. This can be achieved via two ways: 1) Train K non-private teachers, add Gaussian noise to the number of predicted labels from the teachers in each output class, and output the majority of the noisy votes. This is exactly the GNMax algorithm from PATE [95]. 2) Train K (ϵ, Δ)-differentially private teachers and output the majority of the teachers' votes by adding a smaller amount of noise. This can be computed using DaRRM with an appropriate noise function γ. We compare the performance of GNMax and DaRRM with two γ functions: γ_{opt} (i.e., the optimized γ), and γ_{Sub} (i.e., the subsampling baseline). The overall privacy loss over Q queries to the teachers can be computed by general composition (Theorem 3.3.3).

Experiment Setup

We use samples from two randomly chosen classes — class 5 and 8 — from the `MNIST` and `Fashion-MNIST` datasets to form our training and testing datasets. Our `MNIST`

has a total of 11272 training samples and 1866 testing samples; our `Fashion-MNIST` has 10000 training samples and 2000 testing samples. We train $K = 11$ teachers on equally divided subsets of the training datasets. Each teacher is a CNN model. The non-private and private teachers are trained using SGD and DP-SGD [2], respectively, for 5 epochs. *DaRRM Setup:* The Gaussian noise in DP-SGD has zero mean and std. $\sigma_{dpsgd} = 12$; the gradient norm clipping threshold is $C = 1$. This results in each private teacher, trained on `MNIST` and `Fashion-MNIST`, being $(\epsilon, \Delta) = (0.0892, 10^{-4})$ and $(0.0852, 10^{-4})$-differentially private, respectively, after 5 epochs. We set the privacy allowance $m = 3$[4] and the privacy loss per query is then computed using general composition under m-fold, which give the same privacy loss in the high privacy regime, resulting in $(\epsilon_{query}, \delta_{query}) = (0.2676, 0.0003)$ on `MNIST` and $(0.2556, 0.0003)$ on `Fashion-MNIST`. *GNMax Setup:* We now compute the std. σ of the Gaussian noise used by GNMax to achieve a per-query privacy loss of $(m\epsilon, m\Delta)$, as in the DaRRM setup. We optimize σ according to the Renyi differential privacy loss bound of Gaussian noise. Although [95] gives a potentially tighter data-dependent privacy loss bound for majority ensembling *non-private* teachers, we found when K and the number of output classes are small as in our case, even if all teachers agree on a single output class, the condition of the data-dependent bound is not satisfied. Hence, we only use the privacy loss bound of Gaussian noise here to set σ in GNMax. See Appendix B.4.1 for more details, including the σ values and other parameters. Finally, the per sample privacy loss and the total privacy loss over Q queries, which is computed by advanced composition, are reported in Table 3.2.

The testing dataset is treated as the public dataset on which one trains a student model. [95] empirically shows querying $Q = 1\%N$ samples from a public dataset of size N suffices to train a student model with a good performance. Therefore, we pick $Q \in \{20, 50, 100\}$. We repeat the selection of Q samples 10 times and report the mean test accuracy with one std. in parentheses in Table 3.1. The Q queries serve as the

[4]Here, we present results with privacy allowance $m = 3$ because we think this is a more interesting case. $m = 1$ is less interesting, since one cannot get improvement compared to the subsampling baseline. m close to a $\frac{K}{2} \approx 5$ is also less interesting, as this case seems too easy for our proposed method (the optimized γ function is very close to 1, meaning very little noise needs to be added in this case). Hence, we pick $m = 3$, which is a case when improvement is possible, and is also potentially challenging for our optimization framework. This is also realistic as most applications would only want to tolerate a constant privacy overhead. See more results with different privacy allowance m's in this setting in Appendix B.4.1.

labeled samples in training the student model. The higher the accuracy of the labels from the queries, the better the final performance of the student model. We skip the actual training of the student model using semi-supervised learning algorithms here.

Dataset	# Queries	Privacy loss per query $(\epsilon_{query}, \delta_{query})$	Total privacy loss over Q queries $(\epsilon_{total}, \delta_{total})$
MNIST	$Q = 20$	$(0.2676, 0.0003)$	$(5.352, 0.006)$
	$Q = 50$		$(9.901, 0.015)$
	$Q = 100$		$(15.044, 0.030)$
Fashion MNIST	$Q = 20$	$(0.2556, 0.0003)$	$(5.112, 0.006)$
	$Q = 50$		$(9.382, 0.015)$
	$Q = 100$		$(14.219, 0.030)$

Table 3.2: The privacy loss per query to the teachers and the total privacy loss over Q queries. Note the total privacy loss is computed by general composition (see Theorem 3.3.3), where we set $\delta' = 0.0001$.

Discussion

Table 3.1 shows DaRRM$_{\gamma_{opt}}$ achieves the highest accuracy (i.e., utility) compared to the two baselines on both datasets. First, comparing to DaRRM$_{\gamma_{Sub}}$, we verify that subsampling does not achieve a tight privacy-utility tradeoff, and we can optimize the noise function γ in DaRRM to maximize the utility given a target privacy loss. Second, comparing to GNMax, the result shows there are regimes where ensembling private teachers gives a higher utility than directly ensembling non-private teachers, assuming the outputs in both settings have the same privacy loss. Intuitively, this is because ensembling private teachers adds fine-grained noise during both training the teachers and aggregation of teachers' votes, while ensembling non-private teachers adds a coarser amount of noise only to the teachers' outputs. This further motivates private prediction from private teachers and the practical usage of DaRRM, in addition to the need of aggregating private teachers in federated learning settings with an honest-but-curious server.

3.8 Conclusion

In computing a private majority from K private mechanisms, we propose the DaRRM framework, which is provably general, with a customizable γ function. We show a privacy amplification by a factor of 2 in the i.i.d. mechanisms and a pure differential privacy setting. For the general setting, we propose an tractable optimization algorithm that maximizes utility while ensuring privacy guarantees. Furthermore, we demonstrate the empirical effectiveness of DaRRM with an optimized γ. We hope that this work inspires more research on the intersection of privacy frameworks and optimization.

Chapter 4

Private Optimization: Differentially Private Shuffled Gradient Methods

This chapter is based on the following work:

> ***Shuli Jiang**, Pranay Sharma, Steven Wu, Gauri Joshi. **"Improving the Convergence of Private Shuffled Gradient Methods with Public Data"**. In Submission, 2025. [paper]*

In this chapter, we turn our attention to differentially private optimization algorithms, which serve as the backbone of many privacy-preserving machine learning systems. Building on the previous discussion of privacy-utility trade-offs, we investigate how to improve these trade-offs in the context of practical optimization algorithms. In particular, we focus on shuffled gradient methods, a class of algorithms widely adopted in real-world applications and implemented in many machine learning libraries and codebases.

Abstract. We consider the problem of differentially private (DP) convex empirical risk minimization (ERM). While the standard DP-SGD algorithm is theoretically well-established, practical implementations often rely on shuffled gradient methods that traverse the training data sequentially rather than sampling with replacement in each iteration. Despite their widespread use, the theoretical privacy-accuracy trade-offs of private shuffled gradient methods (*DP-ShuffleG*) remain poorly understood,

leading to a gap between theory and practice. In this work, we leverage privacy amplification by iteration (PABI) and a novel application of Stein's lemma to provide the first empirical excess risk bound of *DP-ShuffleG*. Our result shows that data shuffling results in worse empirical excess risk for *DP-ShuffleG* compared to DP-SGD. To address this limitation, we propose *Interleaved-ShuffleG*, a hybrid approach that integrates public data samples in private optimization. By alternating optimization steps that use private and public samples, *Interleaved-ShuffleG* effectively reduces empirical excess risk. Our analysis introduces a new optimization framework with surrogate objectives, varying levels of noise injection, and a dissimilarity metric, which can be of independent interest. Our experiments on diverse datasets and tasks demonstrate the superiority of *Interleaved-ShuffleG* over several baselines.

4.1 Introduction

Differential privacy (DP) [36] has become a cornerstone of privacy-preserving machine learning, providing robust guarantees against the leakage of sensitive information in training datasets. In this work, we revisit the classical problem of (ϵ, δ)-differentially private convex empirical risk minimization (ERM) [13], a framework that underpins many privacy-preserving machine learning tasks. Given the training dataset $\mathsf{D} = \{\mathbf{d}_1, \ldots, \mathbf{d}_n\}$, the private ERM problem can be formulated as:

$$\min_{\mathbf{x}\in\mathbb{R}^d} \Big\{ G(\mathbf{x}; \mathsf{D}) = F(\mathbf{x}; \mathsf{D}) + \psi(\mathbf{x}) \Big\}, \text{ where } F(\mathbf{x}; \mathsf{D}) = \frac{1}{n} \sum_{i=1}^{n} \Big\{ f_i(\mathbf{x}) := f(\mathbf{x}; \mathbf{d}_i) \Big\}, \tag{4.1}$$

while ensuring (ϵ, δ)-differential privacy. Here, $\mathbf{x}$ represents the model parameters, ψ is a convex regularization and f_i's, for all i, are assumed to be convex, smooth, and Lipschitz-continuous[1]. For clarity of presentation, we consider twice differentiable ψ (e.g., $\psi(\mathbf{x}) = \|\mathbf{x}\|^2$) in the main paper[2].

A well-known approach to address the above problem is DP-SGD [1, 13], the

[1]Convexity and smoothness are standard assumptions in the optimization literature. Lipschitzness is used only for privacy analysis [39, 143], and is not required for convergence analysis. Indeed, the Lipschitzness assumption can be removed by using gradient clipping [1] in practice. For simplicity, we retain the Lipschitz assumption.

[2]In our experiments, we consider ψ as ℓ_1 regularizer, ℓ_2 regularizer and the projection operator.

private variant of stochastic gradient descent. In DP-SGD, at each iteration, a gradient is computed using a randomly picked sample from the training data, followed by the addition of Gaussian noise to ensure differential privacy. However, the reliance on i.i.d. sampling introduces practical challenges. Consequently, DP-SGD in its original form is seldom implemented in practice. Instead, as noted in [26, 27, 96], *shuffled gradient methods*, which traverse samples from the training dataset sequentially, are often used in private optimization codebases and libraries, such as *Tensorflow Privacy* [97] and *PyTorch Opacus* [144]. However, their privacy parameters are often incorrectly set based on the analysis of DP-SGD.

In the non-private setting, shuffled gradient methods (a class of methods, which we abbreviate with *ShuffleG*) converge provably faster than SGD [76]. However, the convergence of *private* shuffled gradient methods (which we denote by *DP-ShuffleG*), and how it compares to DP-SGD is poorly understood. This gap motivates our first key question:

What is the privacy-convergence trade-off of private shuffled gradient methods?

To evaluate the privacy-convergence trade-off for a private optimization algorithm, we fix the privacy loss and measure the empirical excess risk, a standard metric in ERM. This metric captures the trade-off by accounting for both the error from noise injection for privacy preservation and the optimization error.

The privacy analysis of private *ShuffleG* presents unique challenges. For DP-SGD, the optimal privacy-convergence trade-off is achieved using a technique called privacy amplification by subsampling (PABS) [13]. However, PABS requires independent sampling of data points at every iteration, hence is not applicable to shuffled gradient methods. Instead, privacy amplification by iteration (PABI), where privacy is amplified by hiding the intermediate parameters, emerges as a viable alternative in the convex setting. While prior work [6, 39, 143] has studied the privacy guarantees of PABI and its use in analyzing the convergence of DP-SGD [41], its application in the context of private *ShuffleG* remains unexplored. Moreover, key challenge in analyzing private *ShuffleG* stems from the interaction between privacy noise and the bias in gradient estimates. Unlike DP-SGD, which uses unbiased gradients, *ShuffleG* accumulates biased gradients over each epoch. This bias, when combined with injected noise, creates nontrivial error terms that couple noise and model parameters. To handle this, we introduce a novel analysis using Stein's lemma. Our approach

addresses a challenge unique to private shuffled methods and goes beyond standard techniques.

Excess Risk for Private Shuffled Gradient Methods. Addressing these challenges, we establish for the first time that the empirical excess risk of private shuffled gradient methods (*DP-ShuffleG*) is $\widetilde{\mathcal{O}}\Big(\frac{1}{n^{2/3}}\big(\frac{\sqrt{d}}{\epsilon}\big)^{4/3}\Big)$, given that the algorithm satisfies (ϵ, δ)-differential privacy. This rate is worse than the empirical excess risk of DP-SGD, $\widetilde{\mathcal{O}}\big(\frac{\sqrt{d}}{n\epsilon}\big)$, with matching lower bound [13]. The worse excess risk for *DP-ShuffleG* matches similar empirical observations in [27]. This disparity can perhaps be intuitively understood: shuffled gradient methods outperform SGD in non-private settings by trading in the bias of the gradient estimator for reduced variance. However, this also implies reduced inherent randomness, resulting in a worse privacy guarantee.

A promising direction to improve the empirical excess risk of private shuffled gradient methods is to leverage public data. The use of public samples, which can be accessed cheaply in many real-world scenarios, has been shown to improve utility in private learning problems, both theoretically [15, 20, 124] and empirically [24, 145]. However, no prior work has explored the use of public samples in the context of private shuffled gradient methods. We consider the practical setting where the public and private datasets may come from different distributions. While using public samples enhances the privacy guarantee, leading to less noise being added, it also risks greater divergence from the target objective. This trade-off motivates our second key question:

Can public samples help improve the privacy-convergence trade-offs of private shuffled gradient methods?

To answer this question, we propose the novel **generalized shuffled gradient framework** (see Algorithm 3), which introduces flexibility along several dimensions. First, instead of using a fixed objective function across all epochs, this framework allows optimizing a potentially different surrogate objective $G(\mathbf{x}; \mathsf{D}^{(s)} \cup \mathsf{P}^{(s)})$ in each epoch s, where the dataset $\mathsf{D}^{(s)} \cup \mathsf{P}^{(s)}$ contains both private ($\mathsf{D}^{(s)}$) and public ($\mathsf{P}^{(s)}$) samples. Second, it supports varying levels of noise injection across epochs to ensure the desired privacy guarantee. For example, when optimizing with only public samples, no noise needs to be added, while noise is necessary when using private samples. Further, to analyze this setup, we introduce a novel metric to measure the dissimilarity

between the true and the surrogate objective (see Assumption 4.4.6), specifically designed for shuffled gradient methods. Unlike prior metrics, it explicitly accounts for the sample order used in gradient computations, enabling tighter convergence bounds.

Using the generalized shuffled gradient framework, we study three algorithms (see Section 4.6): 1) *Priv-Pub-ShuffleG*, where the initial few epochs involve optimizing only using private samples, followed by some epochs on public samples; 2) in *Pub-Priv-ShuffleG*, the order of using public and private samples is reversed compared to *Priv-Pub-ShuffleG*; and 3) *Interleaved-ShuffleG*, which involves using both private and public samples within each epoch. We show that *Interleaved-ShuffleG* achieves lower empirical excess risk than both *Priv-Pub-ShuffleG* and *Pub-Priv-ShuffleG* (4.4), as well as *DP-ShuffleG*.

4.1.1 Our Contributions

1. **Generalized Shuffled Gradient Framework.** In Section 4.4, we present a generalized shuffled gradient framework (Algorithm 3) that allows surrogate objectives (based on public samples) and varying noise addition across epochs. We state the general convergence result, based on a novel dissimilarity metric, in Theorem 4.4.7.
2. **Empirical Excess Risk of *DP-ShuffleG*.** In Section 4.5, we show the empirical excess risk of *DP-ShuffleG*, a special case of Algorithm 3.
3. **Effective Improvement of *DP-ShuffleG* with Public Samples.** Based on the general framework, in Section 4.6, we propose *Interleaved-ShuffleG* (see Algorithm 6), an algorithm that uses both private and public samples within each epoch. *Interleaved-ShuffleG* achieves lower empirical excess risk compared to *DP-ShuffleG* and other approaches that use public samples. A summary of the empirical excess risk comparisons across algorithms is presented in Table 4.4.
4. **Experiments.** In Section 4.7, we empirically demonstrate the superior performance of *Interleaved-ShuffleG* compared to the baselines in three tasks on diverse datasets.

4.2 Related Work

4.2.1 Private Optimization

The privacy loss and convergence of DP-SGD is well understood [1], including tight upper and lower bounds for solving private empirical risk minimization problems in convex settings [13]. However, recent work has observed the gap between theory and practice: shuffled gradient methods are widely implemented in codebases, while the amount of noise applied to the gradients to ensure privacy guarantees is computed based on the analysis of DP-SGD [26, 27]. This line of work, however, focuses on comparisons of the privacy loss between *DP-ShuffleG* and DP-SGD. There is no unified analyses that consider both optimization and privacy.

4.2.2 Shuffled Gradient Methods in Non-private Settings

While the convergence rate of SGD in non-private settings is well understood [105], understanding the convergence of shuffled gradient methods, particularly Random Reshuffling (RR), has been a more recent development. Significant advances include characterizing the convergence rate of RR [83, 84] and establishing last-iterate convergence results for shuffled gradient methods in general [76]. It is known that the best convergence rate by SGD in the non-private setting is $\mathcal{O}\left(\frac{1}{\sqrt{T}}\right)$ for T gradient steps, while [76] shows that the convergence of shuffled gradient methods is $\mathcal{O}\left(\frac{1}{K^{2/3}}\right)$, where $K = T/n$ is the number of epochs, each consisting of n gradient steps based on n samples. The results suggest that shuffled gradient methods converge faster than SGD in the non-private setting. However, it is unclear how their performances compare in the private setting, and we address this gap in this work.

4.2.3 Privacy Amplification by Iteration (PABI)

In many applications, only the last iterate model parameter is used during inference, while intermediate model parameters generated during training are discarded. However, the common privacy analyses based on composition of privacy loss per gradient step implicitly assumes that all intermediate model parameters are released. This discrepancy has motivated a line of research investigating the privacy loss of releasing

only the last iterate model parameter [6, 39, 143] and the privacy amplification that arises by hiding intermediate model parameters is referred to as privacy amplification by iteration (PABI).

Most prior work on PABI focuses on privacy guarantees, with limited attention to its implication on the convergence of private optimizers in solving stochastic convex optimization (SCO) [39, 41], where the algorithms studied are often impractical. [39] relies on convergence bounds for average iterates, contradicting the PABI setting where only the last iterate is released. To align with PABI, they analyze unrealistic algorithm variants like Skip-PNSGD and Stop-PNSGD, which randomly skip or stop gradient steps. While public data is discussed as a means for further privacy amplification, they do not address distributional shifts between public and private datasets. In a related work, [41] propose a DP-SGD based algorithm that utilizes PABI and achieves a tight upper bound on both the excess risk and the number of gradient computation. Its implementation, however, will be non-standard due to the use of exponentially decaying learning rates and varying batch sizes. In practice, fixed batch sizes are preferred for their simplicity, and efficiency in hardware acceleration. Moreover, the first few batches in the algorithm require sizes of $\frac{n}{2}, \frac{n}{4}, \ldots$, where n is the number of samples, which may be too large to fit into memory in practice. *The goal of our work is to analyze and to understand a practical variant of private optimization algorithm, i.e., DP-ShuffleG.*

[6] shows that DP-SGD applied to convex, smooth, and Lipschitz objectives with a bounded domain $\mathcal{W}$ incurs a finite privacy loss, rather an infinite privacy loss as privacy composition indicates. However, their analysis critically depends on privacy amplification by subsampling, specific to DP-SGD, and and the assumption that all model parameters remain within $\mathcal{W}$ at every gradient step. Specifically, the update sequence analyzed is $\mathbf{x}_{t+1} = \text{Proj}_{\mathcal{W}}\left(\mathbf{x}_t - \eta(\nabla f_i(\mathbf{x}_t) + \rho)\right)$, $\forall t \in [T]$, where $\mathbf{x}_t$ is the model parameter at t-th gradient step, $i \in [n]$ is the index of the sample used for gradient computation, ρ is the Gaussian noise vector and Proj is the projection operator, or a special case of regularization. This update ensures $\mathbf{x}_t \in \mathcal{W}$, $\forall t \in [T]$, a key condition for applying their privacy bound. However, in shuffled gradient methods, if regularization (e.g., projection) is applied after each gradient step, one would no longer approximate the full gradient after an epoch to ensure convergence to the target objective and hence, it is crucial to apply the regularization only at the

end of every epoch [83]. This implies that in shuffled gradient methods, we cannot guarantee $\mathbf{x}_t \in \mathcal{W}$ for every gradient step. These differences make the results of [6] inapplicable to private shuffled gradient methods. Another work [143] shows that in a more restricted setting where the objective function is strongly convex, even without a bounded domain, hiding intermediate model parameters leads to a finite privacy loss.

4.2.4 Public Data Assisted Private Learning

There is a long line of work on using public samples to improve statistical learning tasks, e.g., [15, 20, 124]. In machine learning, public data is commonly used to improve model performance by either identifying gradient subspaces [61, 151] or through public pre-training [24, 145]. Empirical studies have also explored the use of public samples in DP-SGD to solve ERM problems [132]. Limited attention has been given to addressing distribution shifts between private and public datasets in statistical learning tasks [16, 19]. None of these works investigate the use of public samples in the context of private shuffled gradient methods for solving ERM or tackle distributional differences between public and private datasets in this specific setting.

4.2.5 Optimization on a Surrogate Objective

The use of surrogate objectives in optimization is studied in the non-private setting in [138], which analyzes SGD using average-iterate methods. However, no prior work has investigated the use of surrogate objectives in the context of shuffled gradient methods.

4.3 Problem Formulation

Notation. Given a positive integer m, $[m] \triangleq \{1, 2, \ldots, m\}$. π denotes a permutation, π_j denotes the j-th element in permutation π, while Π_n denotes the set of all permutations of $[n]$. $\|\cdot\|$ denotes the ℓ_2 norm. $\mathbb{I}_d$ denotes the identity matrix of dimension d. $\mathbf{x}^* = \operatorname{argmin}_{\mathbf{x}\in\mathbb{R}^d} G(\mathbf{x}; \mathsf{D})$ denotes the minimizer of the true objective in equation 4.1. Additionally, we denote the Bregman divergence induced by a

real-valued convex function $g(\mathbf{x}) : \mathbb{R}^d \to \mathbb{R} \cup \{+\infty\}$ as $B_g(\mathbf{x}, \mathbf{y}) = g(\mathbf{x}) - g(\mathbf{y}) - \langle \nabla g(\mathbf{y}), \mathbf{x} - \mathbf{y} \rangle, \forall \mathbf{x}, \mathbf{y} \in \mathbb{R}^d$.

4.3.1 Differential Privacy

We solve the optimization problem given by (4.1) under differential privacy, formally defined as follows:

Definition 4.3.1 (Differential Privacy (DP) [36]). *A randomized mechanism $\mathcal{M} : \mathcal{W} \to \mathcal{R}$ satisfies (ϵ, δ)-differential privacy, for $\epsilon \geq 0, \delta \in (0, 1)$, if for any two* **adjacent datasets** *D, D' and for any subset of outputs $S \subseteq \mathcal{R}$, it holds that $\Pr[\mathcal{M}(\mathsf{D}) \in S] \leq e^{\epsilon} \Pr[\mathcal{M}(\mathsf{D}') \in S] + \delta$. ϵ and δ are called the privacy loss of the algorithm $\mathcal{M}$.*

Additionally, in the privacy analysis, we make use of Rényi Differential Privacy (RDP). The definition of RDP, the conversion between standard DP and RDP and the composition theorem are presented as follows.

Definition 4.3.2 (Renyi Divergence). *For two probability distributions P and Q defined over $\mathcal{R}$, the Renyi divergence of order $\alpha > 1$ is $D_\alpha(P \parallel Q) := \frac{1}{\alpha - 1} \log \mathbb{E}_{x \sim Q} \left(\frac{P(x)}{Q(x)} \right)^{\alpha}$.*

Definition 4.3.3 ((α, ϵ)-Renyi Differential Privacy (RDP) [82]). *A randomized mechanism $f : \mathcal{D} \to \mathcal{R}$ is said to have ϵ-Renyi differential privacy of order α, or (α, ϵ)-RDP for short, if for any adjacent $D, D' \in \mathcal{D}$, it holds that $D_\alpha(f(D) \parallel f(D')) \leq \epsilon$.*

Proposition 4.3.4 (From RDP to DP (Proposition 3 of [82])). *If f is an (α, ϵ)-RDP mechanism, it also satisfies $(\epsilon + \frac{\log 1/\delta}{\alpha - 1}, \delta)$-DP for any $0 < \delta < 1$.*

Proposition 4.3.5 (RDP Composition (Proposition 1 of [82])). *Let $f : \mathcal{D} \to \mathcal{R}_1$ be (α, ϵ_1)-RDP and $g : \mathcal{R}_1 \times \mathcal{D} \to \mathcal{R}_2$ be (α, ϵ_2)-RDP, then the mechanism defined as (X, Y), where $X \sim f(D)$ and $Y \sim g(X, D)$, satisfies $(\alpha, \epsilon_1 + \epsilon_2)$-RDP.*

4.3.2 PABI

We use PABI in the privacy analysis for improved privacy-convergence trade-offs. At a high level, the privacy amplification arises due to hiding intermediate parameters and only release the last-iterate parameter in an optimization procedure. Our analysis builds on the results of PABI in [39]. We begin by introducing the concept of contractive noisy iterations, the key setting where PABI applies, and how the

optimization steps in private shuffled gradient methods fall under this setting.

Definition 4.3.6 (Contraction (Definition 16 of [39])). *For a Banach space $(\mathcal{Z}, \|\cdot\|)$ A function $g : \mathcal{Z} \to \mathcal{Z}$ is said to be contractive if it is 1-Lipschitz, i.e., $\forall \mathbf{x}, \mathbf{y} \in \mathcal{Z}$, $\|g(\mathbf{x}) - g(\mathbf{y})\| \leq \|\mathbf{x} - \mathbf{y}\|$.*

Remark 4.3.7. *As shown in [39], taking one gradient step of a convex and L-smooth objective f, i.e., $g(\mathbf{x}) = \mathbf{x} - \eta \nabla_{\mathbf{x}} f(\mathbf{x})$, where the learning rate $\eta \leq 2/L$, is contractive.*

Definition 4.3.8 (Contractive Noisy Iteration (Definition 19 of [39])). *Given a random initial state $X_0 \in \mathcal{Z}$, a sequence of contractive functions $g_t : \mathcal{Z} \to \mathcal{Z}$, and a sequence of noise distribution $\{\rho_t\}_{t=1}^T$, the contractive noisy iteration (CNI) is defined by the update rule: $X_{t+1} = g_{t+1}(X_t) + Z_{t+1}$, where Z_{t+1}, $\forall t \in [T]$, is drawn independently from ρ_{t+1}. The random variable output by this process after T steps is denoted as $CNI(X_0, \{g_t\}_{t=1}^T, \{\rho_t\}_{t=1}^T)$.*

Theorem 4.3.9 (Privacy Amplification by Iteration (Theorem 22 of [39] with Gaussian Noise)). *Let X_T and X'_T denote the output of $CNI_T(X_0, \{g_t\}_{t=1}^T, \{\rho_t\}_{t=1}^T)$ and $CNI_T(X_0, \{g'_t\}_{t=1}^T, \{\rho_t\}_{t=1}^T)$. Let $s_t := \sup_{\mathbf{x}} \|g_t(\mathbf{x}) - g'_t(\mathbf{x})\|$, where $\rho_t \sim \mathcal{N}(0, \sigma^2 \mathbb{I}_d)$ for all t. Let $a_1, \ldots, a_T$ be a sequence of reals and let $z_t := \sum_{i \leq t} s_i - \sum_{i \leq t} a_i$. If $z_t \geq 0$ for all t and $z_T = 0$, then*

$$D_\alpha(X_T \parallel X'_T) \leq \sum_{t=1}^{T} \frac{\alpha a_t^2}{2\sigma^2}$$

4.3.3 Shuffled Gradient Methods

In this work, we study private shuffled gradient methods (*DP-ShuffleG*), which optimize the objective in (4.1) over K epochs. During epoch $k \in [K]$, the i-th update is given by $\mathbf{x}_{i+1}^{(k)} = \mathbf{x}_i^{(k)} - \eta(\nabla f_j(\mathbf{x}_i^{(k)}) + \rho_i^{(k)})$, $\forall i \in [n]$, where η is the learning rate, $\rho_i^{(k)} \sim \mathcal{N}(0, \sigma^2 \mathbb{I}_d)$ is the Gaussian noise vector, and $j \in [n]$ is the index of the sample selected for gradient computation. Each sample is used exactly once per epoch, with regularization ψ being applied only at the end of every epoch, to ensure convergence [83].

Next, we discuss the three most commonly studied variants of shuffled gradient methods (see Algorithm 2), which differ in how samples are selected in each epoch. **Incremental Gradient (IG)** method processes samples in the same *pre-determined*

Algorithm 2 (Vanilla) Shuffled Gradient Methods

Input: Initial point $\mathbf{x}_1^{(1)}$, learning rate η, number of epochs K. Dataset D of size n.

IG: Fix an order π, set $\pi^{(k)} = \pi, \forall k \in [K]$
SO: Generate permutation π of $[n]$, set $\pi^{(k)} = \pi, \forall k \in [K]$
for $k = 1, 2, \ldots, K$ **do**
  RR: Generate permutation $\pi^{(k)}$ of $[n]$
  for $i = 1, 2, \ldots, n$ **do**
    $\mathbf{x}_{i+1}^{(k)} \leftarrow \mathbf{x}_i^{(k)} - \eta \nabla f(\mathbf{x}_i^{(k)}; \mathbf{d}_{\pi_i^{(k)}})$
  end for
  $\mathbf{x}_1^{(k+1)} \leftarrow \mathbf{x}_{n+1}^{(k)}$
end for
return $\mathbf{x}_1^{(K+1)}$

order across epochs. **Shuffle Once (SO)** also follows the same order across epochs, but the order is a random permutation π of $[n]$. Finally, **Random Reshuffling (RR)** picks a new random permutation $\pi^{(k)}$ at the beginning of each epoch s, which determines the order for that epoch.

To understand the privacy-convergence trade-offs of *DP-ShuffleG*, we define the *empirical excess risk*

$$\mathbb{E}\left[G(\mathbf{x}; \mathsf{D}) - G(\mathbf{x}^*; \mathsf{D})\right] \tag{4.2}$$

where $\mathsf{D} = \{\mathbf{d}_1, \ldots, \mathbf{d}_n\}$ is a private training dataset, and $G(\mathbf{x}; \mathsf{D})$ is the *target objective*. The empirical excess risk captures the trade-off between privacy and convergence, reflecting the optimization error incurred to ensure some fixed privacy guarantee.

Our second goal is to effectively use public samples to improve the empirical excess risk of *DP-ShuffleG*. Alongside D, we have access to some public dataset P, with a potentially different distribution. To allow the flexibility of using varying proportions of samples from both datasets, we formulate the optimization objective as a sequence of surrogate objectives that capture the proportion of public and private data used in each epoch. In each epoch k, we use $n_d^{(k)} (\leq n)$ private samples from D and $n - n_d^{(k)}$ public samples from P. The private dataset used in epoch k, denoted $\mathsf{D}^{(k)}$ is formed by generating a random permutation $\pi^{(k)}$ of $[n]$ and selecting the first

$n_d^{(k)}$ samples in $\pi^{(k)}(\mathsf{D})$, namely, $\mathsf{D}^{(k)} := \{\mathbf{d}_{\pi_i^{(k)}}\}_{i=1}^{n_d^{(k)}}$. The public data used in epoch s is $\mathsf{P}^{(k)} := \{\mathbf{p}_j^{(k)}\}_{j=1}^{n-n_d^{(k)}} \subseteq \mathsf{P}$ with $|\mathsf{P}^{(k)}| = n - n_d^{(k)}$. The *surrogate objective* function used in epoch $k \in [K]$ is

$$G(\mathbf{x}; \mathsf{D}^{(k)} \cup \mathsf{P}^{(k)}) = F(\mathbf{x}; \mathsf{D}^{(k)} \cup \mathsf{P}^{(k)}) + \psi(\mathbf{x}), \tag{4.3}$$
$$F(\mathbf{x}; \mathsf{D}^{(k)} \cup \mathsf{P}^{(k)}) = \frac{1}{n}\Big(\sum_{\mathbf{d} \in \mathsf{D}^{(k)}} f(\mathbf{x}; \mathbf{d}) + \sum_{\mathbf{p} \in \mathsf{P}^{(k)}} f(\mathbf{x}; \mathbf{p}) \Big).$$

The above objective generalizes the target objective $G(\mathbf{x}; \mathsf{D})$ in (4.1) and recovers the objective of *DP-ShuffleG*, when $n_d^{(k)} = n$ and $\mathsf{P}^{(k)} = \emptyset, \forall k \in [K]$. It also allows flexible use of private and public samples, supporting schemes like public pre-training followed by private fine-tuning or mixed usage of private and public samples within an epoch. See Section 4.6 for the discussion on some such approaches. We also define the objective difference between the target objective (4.1) and the surrogate objective used in epoch k (4.3),

$$H^{(k)}(\mathbf{x}) = G(\mathbf{x}; \mathsf{D}) - G(\mathbf{x}; \mathsf{D}^{(k)} \cup \mathsf{P}^{(k)}). \tag{4.4}$$

4.4 Generalized Shuffled Gradient Framework

In this section, we introduce the generalized shuffled gradient framework (Algorithm 3), which incorporates surrogate objectives, and noise injection for privacy preservation. This framework unifies the shuffled gradient methods (*ShuffleG*) and their private variant *DP-ShuffleG*. Specifically, *DP-ShuffleG* corresponds to using only the true private dataset across all epochs ($n_d^{(k)} = n$ and $\mathsf{P}^{(k)} = \emptyset$, $\forall k \in [K]$), while in the non-private version, no noise is added to the gradients ($\sigma^{(k)} = 0$). We provide the convergence analysis of the general framework here, with the convergence of *DP-ShuffleG* as a corollary and its empirical excess risk derived in Section 4.5. We first introduce the assumptions and notation in Section 4.4.1. To analyze the impact of surrogate objectives on convergence, we introduce a novel dissimilarity measure in Section 4.4.2 to measure the difference between the target objective, i.e., $G(\mathbf{x}; \mathsf{D})$, and surrogate objectives, i.e., $G(\mathbf{x}; \mathsf{D}^{(k)} \cup \mathsf{P}^{(k)})$. Using this measure, we present the convergence result in Section 4.4.3.

Algorithm 3 Generalized Shuffled Gradient Framework

Input: Initial point $\mathbf{x}_1^{(1)}$, learning rate η, number of epochs K. Private dataset D. Number of private samples to use $\{n_d^{(k)}\}_{k=1}^K$, $n_d^{(k)} \in \{0\} \bigcup [n]$. Public datasets $\{\mathsf{P}^{(k)}\}_{k=1}^K$ with $|\mathsf{P}^{(k)}| = n - n_d^{(k)}$. Noise standard deviation $\{\sigma^{(k)}\}_{k=1}^K$.
IG: Fix an order π, and set $\pi^{(k)} = \pi, \forall k \in [K]$
SO: Generate permutation π of $[n]$, and set $\pi^{(k)} = \pi, \forall k \in [K]$
for $k = 1, 2, \ldots, K$ **do**
  RR: Generate permutation $\pi^{(k)}$ of $[n]$
  for $i = 1, 2, \ldots, n_d^{(k)}$ **do**
    $\mathbf{x}_{i+1}^{(k)} \leftarrow \mathbf{x}_i^{(k)} - \eta\Big(\nabla f(\mathbf{x}_i^{(k)}; \mathbf{d}_{\pi_i^{(k)}}) + \rho_i^{(k)}\Big)$, where noise $\rho_i^{(k)} \sim \mathcal{N}(0, (\sigma^{(k)})^2 \mathbb{I}_d)$
  end for
  for $i = n_d^{(k)} + 1, n_d^{(k)} + 2, \ldots, n$ **do**
    $\mathbf{x}_{i+1}^{(k)} \leftarrow \mathbf{x}_i^{(k)} - \eta\Big(\nabla f(\mathbf{x}_i^{(k)}; \mathbf{p}_{i-n_d}^{(k)}) + \rho_i^{(k)}\Big)$, where noise $\rho_i^{(k)} \sim \mathcal{N}(0, (\sigma^{(k)})^2 \mathbb{I}_d)$
  end for
  $\mathbf{x}_1^{(k+1)} \leftarrow \operatorname{argmin}_{\mathbf{x} \in \mathbb{R}^d} n\psi(\mathbf{x}) + \frac{\|\mathbf{x} - \mathbf{x}_{n+1}^{(k)}\|^2}{2\eta}$
end for
return $\mathbf{x}_1^{(K+1)}$

4.4.1 Assumptions and Notation

We emphasize that only Assumptions 4.4.1, 4.4.2, and 4.4.4 are required for convergence analysis. Assumption 4.4.3 is standard for privacy analysis and can be removed by using gradient clipping in practice. Recall that D is the training dataset in the target objective (4.1), and P is the public dataset.

Assumption 4.4.1 (Convexity). *$f(\mathbf{x}; \mathbf{d})$ is convex, for all $\mathbf{d} \in \mathsf{D} \cup \mathsf{P}$.*

Assumption 4.4.2 (Smoothness). *A function $f : \mathbb{R}^d \to \mathbb{R}$ is L-smooth if $\|\nabla f(\mathbf{x}) - \nabla f(\mathbf{y})\| \leq L\|\mathbf{x} - \mathbf{y}\|$, for some $L \geq 0$, for all $\mathbf{x}, \mathbf{y}$. $f(\mathbf{x}; \mathbf{d}_i)$ is L_i-smooth, $\forall i \in [n]$ and $\mathbf{d}_i \in \mathcal{D}$. $f(\mathbf{x}; \mathbf{p}_j^{(k)})$ is $\widetilde{L}_j^{(k)}$-smooth, $\forall j \in [n - n_d^{(k)}]$, $\mathbf{p}_j^{(k)} \in \mathcal{P}$ and $\forall k \in [K]$.*

Assumption 4.4.3 (Lipschitz Continuity). *A convex function $f : \mathbb{R}^d \to \mathbb{R}$ is G-Lipschitz if $\|\nabla f(\mathbf{x})\| \leq G$. $f(\mathbf{x}, \mathbf{d})$ is G-Lipschitz, $\forall \mathbf{d} \in \mathsf{D}$; $f(\mathbf{x}; \mathbf{p})$ is $\widetilde{G}$-Lipschitz, for all $\mathbf{p} \in \mathsf{P}$.*

Assumption 4.4.4. *The regularization function ψ is twice differentiable and $\mu_\psi(\geq 0)$-strongly convex.*

Additionally, we define several smoothness and Lipschitz constants relevant to

our analysis. The average smoothness constant of the target objective is given by $L = \frac{1}{n}\sum_{i=1}^{n} L_i$, while the average smoothness constant of the objective used in the k-th epoch is defined as

$$\widehat{L}^{(k)} = \frac{1}{n}\Big(\sum_{i=1}^{n_d^{(k)}} L_{\pi_i^{(k)}} + \sum_{j=1}^{n-n_d^{(k)}} \widetilde{L}_j^{(k)}\Big) \tag{4.5}$$

Similarly, the maximum smoothness constant of the target objective is $L^* = \max_{i\in[n]}\{L_i\}$ and for the k-th eopch, it is

$$\widehat{L}^{(k)*} = \max\{\{L_{\pi_i^{(k)}}\}_{i=1}^{n_d^{(k)}} \cup \{\widetilde{L}_i^{(k)}\}_{i=1}^{n-n_d^{(k)}}\}$$

Finally, we define the maximum Lipschitz parameter as $G^* = \max\{G, \widetilde{G}\}$.

4.4.2 Dissimilarity Measure

Next, we measure the dissimilarity between the target objective function $G(\mathbf{x}; \mathsf{D})$ (4.1) and the surrogate objective function $G(\mathbf{x}; \mathsf{D}^{(k)} \cup \mathsf{P}^{(k)})$ in epoch $k \in [K]$ (4.3), based on the smoothness and the Lipschitzness of the objective difference $H^{(k)}(\mathbf{x})$ defined in (4.4). It follows from Assumptions 4.4.2 and 4.4.3 that $H^{(k)}$ is $(L + L^*)$-smooth, and $(G + G^*)$-Lipschitz continuous. However, these constants can be too large, leading to loose convergence bounds. For example, when the dataset used in every epoch is exactly the same as the training dataset D, i.e., $n_d^{(k)} = n, \mathsf{P}^{(k)} = \emptyset$, $\forall k \in [K]$, then $H^{(k)} \equiv 0$. In this case, the smoothness and the Lipschitzness parameters of $H^{(k)}$ are both 0. Therefore, for sharper analysis, we explicitly model the smoothness and Lipschitzness of $H^{(k)}$.

Assumption 4.4.5. *$H^{(k)}$ is $L_H^{(k)}$-smooth, for all $k \in [K]$.*

Assumption 4.4.6. *For epoch $k \in [K]$, there exists constants $\{C_i^{(k)}\}_{i=n_d^{(k)}+1}^{n}$ such that*

$$\max_{\pi\in\Pi_n} \mathbb{E}\Big[\Big\|\sum\nolimits_{j=n_d^{(k)}+1}^{i}\Big(\nabla f(\mathbf{x}; \mathbf{d}_{\pi_j}) - \nabla f(\mathbf{x}; \mathbf{p}_{j-n_d^{(k)}}^{(k)})\Big)\Big\|\Big] \le C_i^{(k)} \tag{4.6}$$

This measure of dissimilarity is inspired by prior work on distributed SGD-based optimization in non-private settings [131], and optimization with surrogate objectives

[138]. These works define dissimilarity by directly comparing the gradients evaluated at individual samples. For example, in our case, this would be $\|\nabla f(\mathbf{x};\mathbf{d}) - \nabla f(\mathbf{x};\mathbf{p})\| \leq C$, for two private and public samples, $\mathbf{d}, \mathbf{p}$, respectively. However, this crude notion of dissimilarity is less suitable for analyzing shuffled gradient methods and can lead to overly loose bounds.

To illustrate this, consider an extreme case where the public dataset P is simply a permuted version of the private dataset D under some fixed permutation $\widehat{\pi} \in \Pi_n$, i.e., $\mathbf{p}_i = \mathbf{d}_{\widehat{\pi}_i}, \forall i \in [n]$, and $n_d^{(k)} = 0$, for all $k \in [K]$. In other words, the public dataset is identical to the private dataset. The typical dissimilarity measure based on the gradients of individual samples would imply $\|\nabla H^{(k)}(\mathbf{x})\| \leq C$, suggesting nonzero dissimilarity. However, the two datasets are essentially identical. Our proposed dissimilarity measure in Assumption 4.4.6 using $C_n^{(k)} = 0$ correctly captures this, which implies $\|\nabla H^{(k)}(\mathbf{x})\| \equiv 0, \forall k \in [K]$, more accurately reflecting the underlying intuition.

4.4.3 Convergence

Based on the above dissimilarity measure, we present the convergence results of *generalized shuffled gradient framework* in Algorithm 3. In the convergence bound, we use $\sigma_{any}^2 = \frac{1}{n}\sum_{i=1}^n \|\nabla f_i(\mathbf{x}^*)\|^2$ to measure the optimization uncertainty, following [76]. See Section 4.4.4 for a proof sketch and see Appendix C.1 for the full proof.

Theorem 4.4.7 (Convergence of Generalized Shuffled Gradient Framework)**.** *Under Assumptions 4.4.1, 4.4.4, 4.4.6, 4.4.2 and 4.4.5, for $\beta > 0$, if $\mu_\psi \geq L_H^{(k)} + \beta$, $\forall k \in [K]$, and $\eta \leq \frac{1}{2n\sqrt{10\bar{L}^* \max_{k\in[K]} \widehat{L}^{(k)*}(1+\log K)}}$, where $\bar{L}^* = \max\{L, \max_{k\in[K]} \widehat{L}^{(k)}\}$, Algorithm 3 guarantees*

$$\mathbb{E}\left[G(\mathbf{x}_1^{(K+1)})\right] - G(\mathbf{x}^*) \leq \underbrace{\frac{\|\mathbf{x}_1^{(1)} - \mathbf{x}^*\|^2}{\eta n K}}_{\textit{Due to Initialization}} + \underbrace{10\eta^2 n^2 \sigma_{any}^2 (1+\log K) \max_{k\in[K]} \widehat{L}^{(k)}}_{\textit{Optimization Uncertainty}} + 2M \tag{4.7}$$

where

$$M = \max_{s\in[K]} \Bigg(\underbrace{\frac{1}{2n^2\beta}\sum_{k=1}^{s}\frac{(C_n^{(k)})^2}{s+1-k}}_{\text{Non-vanishing Dissimilarity}} + \underbrace{5\eta^2\sum_{k=1}^{s}\frac{\widehat{L}^{(k)*}\frac{1}{n}\sum_{i=n_d^{(k)}+1}^{n-1}(C_i^{(k)})^2}{s+1-k}}_{\text{Vanishing Dissimilarity}} + \underbrace{6\eta^2 nd\sum_{k=1}^{s}\frac{(\sigma^{(k)})^2\widehat{L}^{(k)*}}{s+1-k}}_{\text{Injected Noise}} \Bigg)$$

and the expectation is taken w.r.t. the injected noise $\{\rho_i^{(k)}\}$ *and the order of samples* $\pi^{(k)}$, $\forall i \in [n], k \in [K]$.

4.4.4 Proof Sketch

We now provide a proof sketch of Theorem 4.4.7, highlighting all the key steps.

One Epoch Convergence

We begin by analyzing the convergence for each epoch and presenting two lemmas—Lemma 4.4.8 and Lemma 4.4.9—which generalize Lemmas D.1 and D.2 from [76]. The generalization is along two dimensions: (1) the use of surrogate objectives, and (2) the incorporation of additional noise for privacy preservation. We then combine these results to establish the one-epoch convergence guarantee, stated in Lemma 4.4.10. See proof details in Appendix C.1.2.

Lemma 4.4.8. *Under Assumptions 4.4.1 and 4.4.4, for any epoch* $k \in [K]$, *permutation* $\pi^{(k)}$ *and* $\mathbf{z} \in \mathbb{R}^d$, *Algorithm 3 guarantees*

$$\begin{aligned}
G(\mathbf{x}_1^{(k+1)}) - G(\mathbf{z}) \le{}& H^{(k)}(\mathbf{x}_1^{(k+1)}) - H^{(k)}(\mathbf{z}) + \frac{\|\mathbf{z}-\mathbf{x}_1^{(k)}\|^2}{2n\eta} \\
&- \Big(\frac{1}{2n\eta} + \frac{\mu_\psi}{2}\Big)\|\mathbf{z}-\mathbf{x}_1^{(k+1)}\|^2 - \frac{1}{2n\eta}\|\mathbf{x}_1^{(k+1)}-\mathbf{x}_1^{(k)}\|^2 \\
&+ \frac{1}{n}\Bigg(\sum_{i=1}^{n_d^{(k)}}\Big(B_{f_{\pi_i^{(k)}}}(\mathbf{x}_1^{(k+1)},\mathbf{x}_i^{(k)}) - B_{f_{\pi_i^{(k)}}}(\mathbf{z},\mathbf{x}_i^{(k)})\Big) \\
&+ \sum_{i=n_d^{(k)}+1}^{n}\Big(B_{f^{(k,pub)}_{i-n_d^{(k)}}}(\mathbf{x}_1^{(k+1)},\mathbf{x}_i^{(k)}) - B_{f^{(k,pub)}_{i-n_d^{(k)}}}(\mathbf{z},\mathbf{x}_i^{(k)})\Big)\Bigg) + \frac{1}{n}\sum_{i=1}^{n}\langle -\rho_i^{(k)}, \mathbf{x}_1^{(k+1)}-\mathbf{z}\rangle
\end{aligned}$$

Lemma 4.4.9. *Under Assumptions 4.4.1, 4.4.6 and 4.4.2, for any epoch* $k \in [K]$,

permutation $\pi^{(k)}$ *and* $\mathbf{z} \in \mathbb{R}^d$, *if the learning rate* $\eta \leq \frac{1}{n\sqrt{10\widehat{L}^{(k)}\widehat{L}^{(k)*}}}$, *Algorithm 3 guarantees*

$$
\begin{aligned}
&\frac{1}{n}\Big(\sum_{i=1}^{n_d^{(k)}}\Big(B_{f_{\pi_i^{(k)}}}(\mathbf{x}_1^{(k+1)},\mathbf{x}_i^{(k)}) - B_{f_{\pi_i^{(k)}}}(\mathbf{z},\mathbf{x}_i^{(k)})\Big)\\
&\quad+\sum_{i=n_d^{(k)}+1}^{n}\Big(B_{f^{(k,pub)}_{i-n_d^{(k)}}}(\mathbf{x}_1^{(k+1)},\mathbf{x}_i^{(k)}) - B_{f^{(k,pub)}_{i-n_d^{(k)}}}(\mathbf{z},\mathbf{x}_i^{(k)})\Big)\Big)\\
&\leq \widehat{L}^{(k)}\|\mathbf{x}_1^{(k+1)}-\mathbf{x}_1^{(k)}\|^2 + 10\eta^2 n^2\widehat{L}^{(k)}LB_F(\mathbf{z},\mathbf{x}^*)\\
&\quad+5\eta^2\frac{1}{n}\Big(\sum_{i=2}^{n_d^{(k)}}L_{\pi_i^{(k)}}\Big\|\sum_{j=1}^{i-1}\nabla f_{\pi_j^{(k)}}(\mathbf{x}^*)\Big\|^2 + \sum_{i=n_d^{(k)}+1}^{n}\widetilde{L}^{(k)}_{i-n_d^{(k)}}\Big\|\sum_{j=1}^{i-1}\nabla f_{\pi_j^{(k)}}(\mathbf{x}^*)\Big\|^2\Big)\\
&\quad+5\eta^2\frac{1}{n}\Big(\sum_{i=2}^{n_d^{(k)}}L_{\pi_i^{(k)}}\Big\|\sum_{j=1}^{i-1}\rho_j^{(k)}\Big\|^2 + \sum_{i=n_d^{(k)}+1}^{n}\widetilde{L}^{(k)}_{i-n_d^{(k)}}\Big\|\sum_{j=1}^{i-1}\rho_j^{(k)}\Big\|^2\Big)\\
&\quad+5\eta^2 L^{(k)*}\frac{1}{n}\sum_{i=n_d^{(k)}+1}^{n}\Big\|\sum_{j=n_d^{(k)}+1}^{i-1}\nabla f^{(k,pub)}_{i-n_d^{(k)}}(\mathbf{z}) - \sum_{j=n_d^{(k)}+1}^{i-1}\nabla f_{\pi_j^{(k)}}(\mathbf{z})\Big\|^2
\end{aligned}
$$

Lemma 4.4.10 (One Epoch Convergence)**.** *Under Assumptions 4.4.1, 4.4.4, 4.4.6, 4.4.2 and 4.4.5, for any epoch* $k \in [K]$, $\beta > 0$, *and* $\forall \mathbf{z} \in \mathbb{R}^d$, *if* $\eta \leq \frac{1}{n\sqrt{10\widehat{L}^{(k)}\widehat{L}^{(k)*}}}$, *Algorithm 3 guarantees*

$$
\begin{aligned}
G(\mathbf{x}_1^{(k+1)}) - G(\mathbf{z}) &\leq \frac{1}{2n\eta}(\|\mathbf{z}-\mathbf{x}_1^{(k)}\|^2 - \|\mathbf{z}-\mathbf{x}_1^{(k+1)}\|^2) \qquad (4.8)\\
&\quad+(\frac{L_H^{(k)}+\beta}{2} - \frac{\mu_\psi}{2})\|\mathbf{z}-\mathbf{x}_1^{(k+1)}\|^2 + 10\eta^2 n^2\widehat{L}^{(k)}LB_F(\mathbf{z},\mathbf{x}^*)\\
&\quad\underbrace{+5\eta^2\frac{1}{n}\Big(\sum_{i=2}^{n_d^{(k)}}L_{\pi_i^{(k)}}\Big\|\sum_{j=1}^{i-1}\nabla f_{\pi_j^{(k)}}(\mathbf{x}^*)\Big\|^2 + \sum_{i=n_d^{(k)}+1}^{n}\widetilde{L}^{(k)}_{i-n_d^{(k)}}\Big\|\sum_{j=1}^{i-1}\nabla f_{\pi_j^{(k)}}(\mathbf{x}^*)\Big\|^2\Big)}_{\textit{Optimization Uncertainty}}\\
&\quad\underbrace{+\frac{1}{n}\sum_{i=1}^{n}\langle -\rho_i^{(k)}, \mathbf{x}_1^{(k+1)}-\mathbf{z}\rangle + 5\eta^2\frac{1}{n}\Big(\sum_{i=2}^{n_d^{(k)}}L_{\pi_i^{(k)}}\Big\|\sum_{j=1}^{i-1}\rho_j^{(k)}\Big\|^2 + \sum_{i=n_d^{(k)}+1}^{n}\widetilde{L}^{(k)}_{i-n_d^{(k)}}\Big\|\sum_{j=1}^{i-1}\rho_j^{(k)}\Big\|^2\Big)}_{\textit{Injected Noise}}
\end{aligned}
$$

$$+ \underbrace{\frac{1}{2n^2\beta}(C_n^{(k)})^2}_{\text{Non-vanishing Dissimilarity}} + \underbrace{5\eta^2 L^{(k)*}\frac{1}{n}\sum_{i=n_d^{(k)}+1}^{n}\Big\|\sum_{j=n_d^{(k)}+1}^{i-1}\nabla f_{i-n_d^{(k)}}^{(k,pub)}(\mathbf{z}) - \sum_{j=n_d^{(k)}+1}^{i-1}\nabla f_{\pi_j^{(k)}}(\mathbf{z})\Big\|^2}_{\text{Vanishing Dissimilarity}}$$

Expected One Epoch Convergence

We next analyze the expected convergence over one epoch, accounting for two sources of randomness: the random shuffling of samples and the injected noise added for privacy. Notably, the noise injection introduces an additional error term, as characterized in Lemma 4.4.11. This term does not appear in standard analyses of private optimization algorithms, such as that of DP-SGD. To handle this, we develop a novel analysis leveraging Stein's lemma to bound the bias introduced by noise, as detailed in Lemma 4.4.11. We further bound the noise variance in Lemma 4.4.12. Finally, we incorporate the effect of sample shuffling—quantified using the variance parameter σ_{any}^2 —and combine all components to derive the expected convergence for one epoch in Lemma 4.4.13. See proof details in Appendix C.1.3.

Lemma 4.4.11 (Additional Error)**.** *For any epoch $s \in [K]$ and $\forall \mathbf{z} \in \mathbb{R}^d$, consider the injected noise $\rho_i^{(k)} \sim \mathcal{N}(0, (\sigma^{(k)})^2\mathbb{I}_d)$, $\forall i \in [n]$, if the regularization function ψ is twice differentiable and $\mathbf{z}$ is independent of $\rho_i^{(k)}, \forall i \in [n]$, then the error caused by noise injection in epoch k is*

$$\mathbb{E}\left[\frac{1}{n}\sum_{i=1}^{n}\langle \rho_i^{(k)}, \mathbf{x}_1^{(k+1)} - \mathbf{z}\rangle\right] \leq (\sigma^{(k)})^2 nd\eta^2 \widehat{L}^{(k)*} \tag{4.9}$$

where the expectation is taken w.r.t. the injected noise $\{\rho_i^{(k)}\}_{i=1}^n$.

Lemma 4.4.12 (Noise Variance)**.** *For any epoch $k \in [K]$ and $\forall \mathbf{z} \in \mathbb{R}^d$, consider the injected noise $\rho_i^{(k)} \sim \mathcal{N}(0, (\sigma^{(k)})^2\mathbb{I}_d)$, $\forall i \in [n]$, the variance caused by noise injection in epoch k is, $\forall i \in [n]$,*

$$\mathbb{E}\left[\Big\|\sum_{j=1}^{i}\rho_j^{(k)}\Big\|^2\right] \leq id(\sigma^{(k)})^2$$

where the expectation is taken w.r.t. the injected noise $\{\rho_i^{(k)}\}_{i=1}^n$.

Lemma 4.4.13 (Expected One Epoch Convergence). *Under Assumptions 4.4.1, 4.4.4, 4.4.6, 4.4.2 and 4.4.5, for any epoch $k \in [K]$, $\beta > 0$ and $\forall \mathbf{z} \in \mathbb{R}^d$, if $\eta \leq \frac{1}{n\sqrt{10\widehat{L}^{(k)}\widehat{L}^{(k)*}}}$ and $\mathbf{z}$ is independent of $\rho_i^{(k)}$, $\forall i \in [n]$, Algorithm 3 guarantees*

$$\begin{aligned}\mathbb{E}\left[G(\mathbf{x}_1^{(k+1)})\right] - \mathbb{E}\left[G(\mathbf{z})\right] \leq \frac{1}{2n\eta}\Big(\mathbb{E}\left[\|\mathbf{z}-\mathbf{x}_1^{(k)}\|^2\right] - \mathbb{E}\left[\|\mathbf{z}-\mathbf{x}_1^{(k+1)}\|^2\right]\Big) \\ + \Big(\frac{L_H^{(k)}+\beta}{2} - \frac{\mu_\psi}{2}\Big)\mathbb{E}\left[\|\mathbf{z}-\mathbf{x}_1^{(k+1)}\|^2\right] + 10\eta^2 n^2 \widehat{L}^{(k)} L \mathbb{E}\left[B_F(\mathbf{z},\mathbf{x}^*)\right] \\ + \underbrace{5\eta^2 n^2 \widehat{L}^{(k)}\sigma_{any}^2}_{\text{Optimization Uncertainty}} + \underbrace{\frac{1}{2n^2\beta}(C_n^{(k)})^2}_{\text{Non-vanishing Dissimilarity}} + \underbrace{5\eta^2\widehat{L}^{(k)*}\frac{1}{n}\sum_{i=n_d^{(k)}+1}^{n-1}(C_i^{(k)})^2}_{\text{Vanishing Dissimilarity}} \\ + \underbrace{6\eta^2 nd(\sigma^{(k)})^2\widehat{L}^{(k)*}}_{\text{Injected Noise}}\end{aligned} \tag{4.10}$$

where the expectation is taken w.r.t. both the injected noise within epoch k, i.e., $\{\rho_i^{(k)}\}_{i=1}^n$, and the shuffling operator $\pi^{(k)}$.

Convergence Across K Epochs

Finally, we extend the one-epoch convergence result to K epochs. Our analysis follows a similar approach to that of [76] for last-iterate convergence. We begin by computing convergence across an arbitrary number of s epochs in Lemma 4.4.14. To derive the final convergence bound over K epochs in Theorem 4.4.7, we apply a weighted telescoping sum and recursive argument—departing from the unweighted approach commonly used in classical analyses. See proof details in Appendix C.1.4.

Lemma 4.4.14 (Convergence Across Arbitrary Epochs). *Under Assumptions 4.4.1, 4.4.4, 4.4.6, 4.4.2 and 4.4.5, for any number of epochs $s \in [K]$ and $\beta > 0$, if $\mu_\psi \geq L_H^{(k)} + \beta$, $\forall k \in [s]$, and $\eta \leq \frac{1}{n\sqrt{10\max_{k\in[s]}(\widehat{L}^{(k)}\widehat{L}^{(k)*})}}$, Algorithm 3 guarantees*

$$\mathbb{E}\left[G(\mathbf{x}_1^{(s+1)})\right] - G(\mathbf{x}^*) \leq \underbrace{\frac{\|\mathbf{x}^*-\mathbf{x}_1^{(1)}\|^2}{2\eta n s}}_{\text{Due to Initialization}}$$

$$+\underbrace{10\eta^2 n^2 L \max_{k\in[s]} \widehat{L}^{(k)} \sum_{k=2}^{s} \frac{1}{s+2-k} \mathbb{E}\left[B_F(\mathbf{x}_1^{(k)}, \mathbf{x}^*)\right]}_{\textit{Recursive Relationship}}$$

$$+\underbrace{5\eta^2 n^2 \sigma_{any}^2 \sum_{k=1}^{s} \frac{\widehat{L}^{(k)}}{s+1-k}}_{\textit{Optimization Uncertainty}} + \underbrace{\frac{1}{2n^2\beta} \sum_{k=1}^{s} \frac{(C_n^{(k)})^2}{s+1-k}}_{\textit{Non-vanishing Dissimilarity}}$$

$$+\underbrace{5\eta^2 \sum_{k=1}^{s} \frac{\widehat{L}^{(k)*} \frac{1}{n} \sum_{i=n_d^{(k)}+1}^{n-1} (C_i^{(k)})^2}{s+1-k}}_{\textit{Vanishing Dissimilarity}} + \underbrace{6\eta^2 nd \sum_{k=1}^{s} \frac{(\sigma^{(k)})^2 \widehat{L}^{(k)*}}{s+1-k}}_{\textit{Injected Noise}}$$

4.4.5 Convergence Bound for Non-Private Shuffled Gradient Methods

In the special case where we only use private data, i.e., $n_d^{(k)} = n$, and no noise is injected, $\sigma^{(k)} = 0$, and the dissimilarity is $C_i^{(k)} = 0$, $\forall i \in \{n_d^{(k)} + 1, \ldots, n\}$, $\forall k \in [K]$. Consequently, the bound in (4.7) recovers the convergence rate of non-private shuffled gradient methods in Theorem 4.4 of [76]. This is formalized in the following corollary. Recall $L = \frac{1}{n}\sum_{i=1}^{n} L_i$ and $L^* = \max_{i\in[n]}\{L_i\}_{i=1}^{n}$ are the average and maximum smoothness constants of the target objective.

Corollary 4.4.15 (Convergence of Non-private Shuffled Gradient Methods)**.** *Under Assumptions 4.4.1, 4.4.4, 4.4.6 and 4.4.2, if the learning rate satisfies* $\eta \leq \frac{1}{2n\sqrt{10LL^*}}$, *Algorithm 3 guarantees*

$$\mathbb{E}\left[G(\mathbf{x}_1^{(K+1)})\right] - G(\mathbf{x}^*) \leq \underbrace{\frac{\|\mathbf{x}_1^{(1)} - \mathbf{x}^*\|^2}{\eta n K}}_{\textit{Initialization}} + \underbrace{10\eta^2 n^2 \sigma_{any}^2 (1+\log K) L}_{\textit{Optimization Uncertainty}} \tag{4.11}$$

Furthermore, choosing the learning rate as $\eta = \min\{\frac{1}{n\sqrt{LL^*(1+\log K)}}, \frac{\|\mathbf{x}^* - \mathbf{x}_1^{(1)}\|^{2/3}}{nL^{1/3}\sigma_{any}^{2/3} K^{1/3}(1+\log K)^{1/3}}\}$ *yields the following convergence bound:*

$$\mathbb{E}\left[G(\mathbf{x}_1^{(K+1)})\right] - G(\mathbf{x}^*) \tag{4.12}$$
$$= \mathcal{O}\left(\frac{\|\mathbf{x}^* - \mathbf{x}_1^{(1)}\|^2 \sqrt{LL^*(1+\log K)}}{K}\right) + \mathcal{O}\left(\frac{\|\mathbf{x}^* - \mathbf{x}_1^{(1)}\|^{4/3} L^{1/3} \sigma_{any}^{2/3} (1+\log K)^{1/3}}{K^{2/3}}\right)$$

Proof of Corollary 4.4.15. The convergence bound in Eq. 4.11 follows directly from the general result in Theorem 4.4.7 (Eq. 4.11) by setting: the number of private samples per epoch $n_d^{(k)} = n$, dissimilarity $C_n^{(k)} = 0$, noise $\sigma^{(k)} = 0$, and using the appropriate smoothness constants—maximum $L^* = \max_{i\in[n]} L_i$ and average per epoch $\widehat{L}^{(k)} = \frac{1}{n}\sum_{i=1}^n L_i = L$ for all $k \in [K]$. With these settings, Eq. 4.11 recovers the convergence bound for non-private shuffled gradient methods under smooth objectives prior to specifying the learning rate η, as shown in Theorem C.1 of [76]. Adopting the same learning rate choice as in [76] yields the exact result stated in their Theorem 4.4 and full version Theorem C.1. □

4.4.6 Impact of Dissimilarity

In the convergence bound (4.7), we identify two dissimilarity terms: one *vanishing* and the other *non-vanishing*. When the dataset used for optimization, $\mathcal{D}^{(k)} \cup \mathcal{P}^{(k)}$, differs from the original dataset $\mathcal{D}$, Algorithm 3 cannot be expected to converge exactly to the true solution $\mathbf{x}^*$. This discrepancy is captured by the *Non-vanishing Dissimilarity* term, which remains nonzero when $C_n^{(k)} \neq 0$ (Assumption 4.4.6). Conversely, if the datasets $\mathcal{D}^{(k)} \cup \mathcal{P}^{(k)}$ used across all epochs are permutations of the original dataset $\mathcal{D}$ (e.g., see the discussion following Assumption 4.4.6 in Section 4.4.2), then $C_n^{(k)} = 0, \forall k$ and the *Non-vanishing Dissimilarity* term disappears. Nevertheless, even with identical data, different sample orderings can still lead to varying optimization trajectories in *ShuffleG*. This effect is reflected through $\{C_i^{(k)} > 0\}$ in the *Vanishing Dissimilarity* term. However, the η^2 scaling in this term ensures it does not dominate the overall convergence bound.

4.5 Private Shuffled Gradient Methods

In this section, we derive the convergence rate and the empirical excess risk of *DP-ShuffleG*. As discussed earlier, *DP-ShuffleG* is a special case of the *generalized shuffled gradient framework* (Algorithm 3) where the surrogate objectives are identical to the target objective, i.e., $n_d^{(k)} = n, \mathsf{P}^{(k)} = \emptyset$, and $\sigma^{(k)} = \sigma$, $\forall k \in [K]$. We first present the convergence bound of *DP-ShuffleG* as a corollary of Theorem 4.4.7 in Corollary 4.5.1. In Lemma 4.5.2, we state the differential privacy guarantee of

DP-ShuffleG, in terms of the noise variance σ. Finally, we discuss the choice of the learning rate η and the number of epochs K to achieve the minimal empirical excess risk while ensuring (ϵ, δ)-DP.

4.5.1 Convergence of *DP-ShuffleG*

We give the convergence bound of *DP-ShuffleG* as follows. Since the full $\mathcal{D}$ is used across all epochs, $n_d^{(k)} = n$ and the dissimilarity measure (see Assumption 4.4.6) is $C_n^{(k)} = 0$, $\forall k \in [K]$. By Theorem 4.4.7, the convergence of *DP-ShuffleG* is

Corollary 4.5.1 (Convergence of *DP-ShuffleG*[3])**.** *Under the conditions in Theorem 4.4.7, with* $n_d^{(k)} = n$, $\mathcal{P}^{(k)} = \emptyset$, *and* $\sigma^{(k)} = \sigma$ *for all* $k \in [K]$, *Algorithm 3 (DP-ShuffleG) guarantees*

$$\mathbb{E}\left[G(\mathbf{x}_1^{(K+1)})\right] - G(\mathbf{x}^*) \leq \underbrace{\frac{\|\mathbf{x}_1^{(1)} - \mathbf{x}^*\|^2}{\eta n K}}_{\text{Initialization}} + \underbrace{10\eta^2 n^2 \sigma_{any}^2 L(1 + \log K)}_{\text{Optimization Uncertainty}} + \underbrace{12\eta^2 n d \sigma^2 L^*(1 + \log K)}_{\text{Noise Injection}}$$

4.5.2 Privacy of *DP-ShuffleG*

We use privacy amplification by iteration (PABI [39]) to bound the privacy loss within an epoch. The privacy amplification arises because the intermediate iterates within an epoch $\{\mathbf{x}_i^{(k)}\}_{i=1}^n$ are hidden. However, PABI requires the update steps to be "contractive" (Definition 4.3.6). While each gradient step within an epoch (line 8 of Algorithm 3) satisfies this property, the regularization step at the end of each epoch (line 14 of Algorithm 3) need not be contractive. This prevents us from using PABI *across* epochs. Hence, we use composition (Proposition 4.3.5) to bound the total privacy loss across the K epochs. See Appendix C.2.1 for additional comments on the privacy loss due to random shuffling.

Lemma 4.5.2 (Privacy of *DP-ShuffleG*)**.** *Under Assumptions 4.4.1, 4.4.2 and 4.4.3, given the learning rate* $\eta \leq 1/L$, *DP-ShuffleG is* $(\frac{2\alpha G^2 K}{\sigma^2} + \frac{\log 1/\delta}{\alpha - 1}, \delta)$*-DP, for* $\alpha > 1, \delta \in (0, 1)$.

[3]One can set $\beta = 0$ when $L_H^{(k)} = 0$, which is the case here since no surrogate datasets is used. This implies $\mu_\psi \geq 0$, as indicated in Assumption 4.4.4, suffices to ensure convergence.

Proof of Lemma 4.5.2. We first show the privacy loss per epoch by using privacy amplification by iteration (PABI) in Renyi Differential Privacy (RDP, see Definition 4.3.3), and then use the composition theorem of RDP (see Proposition 4.3.5) to derive the total privacy loss across K epochs. Finally, we convert the privacy loss in RDP to DP based on Proposition 4.3.4.

By Remark 4.3.7, if one sets the learning rate in *DP-ShuffleG* as $\eta \leq 1/L^*$, each gradient update step in epoch $k \in [K]$, i.e., $\mathbf{x}_{i+1}^{(k)} = \mathbf{x}_i^{(k)} - \eta(\nabla f_{\pi_i^{(k)}}(\mathbf{x}_i^{(k)}) + \rho_i^{(k)})$, $\forall i \in [n]$, (line 8 of Algorithm 3) satisfies a "noisy contractive sequence" (CNI, see Definition 4.3.8) and hence, we can apply Theorem 4.3.9 to reason about the privacy loss of epoch k, $\forall k \in [K]$.

Let $\mathcal{D} = \{\mathbf{d}_1, \ldots, \mathbf{d}_t, \ldots, \mathbf{d}_n\}$ and $\mathcal{D}' = \{\mathbf{d}_1, \ldots, \mathbf{d}'_t, \ldots, \mathbf{d}_n\}$ be two neighboring datasets that differ at some index $t \in [n]$. On dataset $\mathcal{D}$, the CNI is defined by the initial point $\mathbf{x}_1^{(k)}$, sequence of functions $g_i(\mathbf{x}) = \mathbf{x}_i^{(k)} - \eta \nabla f(\mathbf{x}; \mathbf{d}_{\pi_i^{(k)}})$, for all $\mathbf{x}$, and sequence of noise distributions $\mathcal{N}(0, (\eta\sigma)^2 \mathbb{I}_d)$. Similarly, on dataset $\mathcal{D}'$, the CNI is defined in the same way with the exception of $g'_j(\mathbf{x}) = \mathbf{x} - \eta \nabla f(\mathbf{x}; \mathbf{d}'_{\pi_j^{(k)}})$ for the index j such that $\pi_j^{(k)} = t$. Let $\mathbf{x}_{n+1}^{(k)}$, $(\mathbf{x}_{n+1}^{(k)})'$ be the output of the CNI on dataset $\mathcal{D}$ and $\mathcal{D}'$, respectively.

By Assumption 4.4.3, $f(\mathbf{x}; \mathbf{d}_i)$ is G-Lipschitz, $\forall i \in [n]$, and hence,

$$\sup_{\mathbf{w}} \|g_j(\mathbf{x}) - g'_j(\mathbf{x})\| = \sup_{\mathbf{w}} \|\eta \nabla f(\mathbf{x}; \mathbf{d}_t) - \eta \nabla f(\mathbf{x}; \mathbf{d}'_t)\| \leq 2\eta G$$

We now apply Theorem 4.3.9 with $a_1, \ldots, a_{j-1} = 0$ and $a_j, \ldots, a_n = \frac{2\eta G}{n-j+1}$. Note that $s_{\pi_j^{(k)}} = 2\eta G$ and $s_i = 0$, $\forall i \neq \pi_j^{(k)}$. In addition, $z_i \geq 0$ for all $i \leq n$ and $z_n = 0$. Hence,

$$D_\alpha(\mathbf{x}_{n+1}^{(k)} \parallel (\mathbf{x}_{n+1}^{(k)})') \leq \sum_{i=j}^{n} \frac{\alpha}{2\eta^2\sigma^2} \cdot \frac{4\eta^2 G^2}{(n-j+1)^2} = \frac{2\alpha G^2}{\sigma^2(n-j+1)}$$

The maximum privacy loss happens when $j = n$, that is, when the sample $\mathbf{d}_t$ is the last one being processed in epoch k. And it is not hard to see that $\max_{j \in [n]} D_\alpha(\mathbf{x}_{n+1}^{(k)} \parallel (\mathbf{x}_{n+1}^{(k)})') \leq \frac{2\alpha G^2}{\sigma^2}$, which implies $\mathbf{x}_{n+1}^{(k)}$ in Algorithm 3 is $(\alpha, \frac{2\alpha G^2}{\sigma^2})$-RDP, for $\alpha > 1$. The output of epoch k, $\mathbf{x}_1^{(k+1)}$ can be seen as a post-processing step of $\mathbf{x}_{n+1}^{(k)}$, and hence, $\mathbf{x}_1^{(k+1)}$ is also $(\alpha, \frac{2\alpha G^2}{\sigma^2})$-RDP.

By Proposition 4.3.5, the output $\mathbf{x}_1^{(K+1)}$ is $(\alpha, \frac{2\alpha G^2 K}{\sigma^2})$-RDP. And by Proposition 4.3.4, $\mathbf{x}_1^{(K+1)}$ is $(\frac{2\alpha G^2 K}{\sigma^2} + \frac{\log 1/\delta}{\alpha - 1}, \delta)$-DP, for $\alpha > 1$ and $\delta \in (0, 1)$.

□

4.5.3 Empirical Excess Risk

To ensure *DP-ShuffleG* satisfies (ϵ, δ)-DP, we set $\alpha = \frac{\sigma\sqrt{\log(1/\delta)}}{G\sqrt{2K}}$ based on Lemma 4.5.2 to minimize the overall privacy loss. This choice implies $\sigma = \mathcal{O}\left(\frac{G\sqrt{K\log(1/\delta)}}{\epsilon}\right)$. The learning rate is set to be $\eta = \mathcal{O}\left(\frac{\|\mathbf{x}_1^{(1)} - \mathbf{x}^*\|^{2/3}}{nL^*(K(1+\log K))^{1/3}}\right)$ to optimize the convergence bound, while satisfying the conditions of both Corollary 4.5.1 (convergence) and Lemma 4.5.2 (privacy). After choosing the learning rate, the convergence bound of *DP-ShuffleG* is now given by

$$\begin{aligned}
&\mathbb{E}\left[G(\mathbf{x}_1^{(K+1)})\right] - \mathbb{E}\left[G(\mathbf{x}^*)\right] \\
&\leq \mathcal{O}\Big(\|\mathbf{x}_1^{(1)} - \mathbf{x}^*\|^{4/3}(1+\log K)^{1/3}\Big(\frac{L^*}{K^{2/3}} + \frac{1}{L^* K^{2/3}} + \frac{dG^2 K^{1/3}\log 1/\delta}{nL^*\epsilon^2}\Big)\Big)
\end{aligned}$$

Finally, setting the number of epochs as $K = \mathcal{O}\left(\frac{n\epsilon^2}{d}\right)$ to minimize the above bound, resulting in the following empirical excess risk of *DP-ShuffleG*:

$$\mathbb{E}\left[G(\mathbf{x}_1^{(K+1)})\right] - \mathbb{E}\left[G(\mathbf{x}^*)\right] = \widetilde{O}\left(\frac{1}{n^{2/3}}(\frac{\sqrt{d}}{\epsilon})^{4/3}\right) \tag{4.13}$$

with respect to n, d, and ϵ, while ignoring other terms. Here, $\widetilde{\mathcal{O}}$ hides logarithmic factors in $(n, d, 1/\delta)$.

4.5.4 Comparison with DP-(S)GD

The lower bound for empirical excess risk when minimizing convex, smooth objectives is $\Omega(\sqrt{d}/(n\epsilon))$ [13]. DP-GD and DP-SGD achieve matching upper bounds. However, the above bound suggests a worse empirical excess risk for *DP-ShuffleG*. This aligns partially with the empirical findings of [27] that shuffled gradient methods (SO and RR) underperform DP-SGD in private binary classification tasks with the same

privacy guarantees. Their setting, however, allows intermediate model parameter releases and does not require convex objectives. The worse privacy guarantee of *DP-ShuffleG* can be intuitively explained: the gradient estimators in shuffled gradient methods [76] have a smaller variance compared to SGD. Consequently, to ensure the same privacy guarantee, these methods need a larger noise variance.

4.6 Leveraging Public Data

Given the pessimistic empirical excess risk of *DP-ShuffleG* discussed above, how can it be improved? In this section, we explore leveraging public data samples P in the context of private shuffled gradient methods. We propose a novel approach that interleaves the usage of public and private samples during training, demonstrating its effectiveness in reducing empirical excess risk.

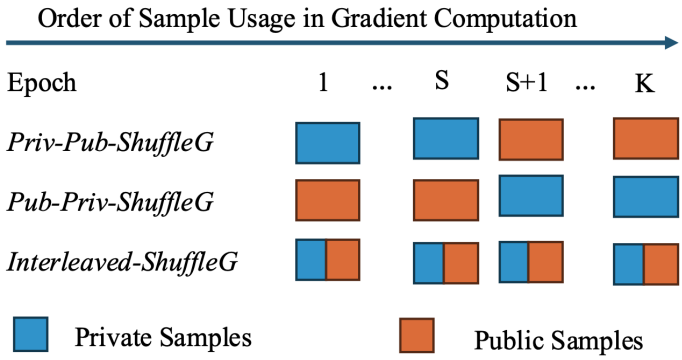

Figure 4.1: Illustration of algorithms using public data.

We begin with presenting two common baselines, *Priv-Pub-ShuffleG* in Section 4.6.1, and *Pub-Priv-ShuffleG* in Section 4.6.2. We then propose *Interleaved-ShuffleG* in Section 4.6.3. These algorithms, which leverage public samples, are specific instantiations of Algorithm 3. An illustration of these algorithms is provided in Figure 4.1.

4.6.1 Algorithm 1: *Priv-Pub-ShuffleG*

The pseudocode of *Priv-Pub-ShuffleG* is presented in Algorithm 4.

Priv-Pub-ShuffleG trains models only using the private dataset D for the first S epochs, where $S \in [K-1]$. For the remaining $K-S$ epochs, it trains models only

Algorithm 4 *Priv-Pub-ShuffleG*

Input: Initial point $\mathbf{x}_1^{(1)}$, learning rate η, number of epochs K. Private dataset $\mathcal{D} = \{\mathbf{d}_i\}_{i=1}^n$. Public datasets $\mathcal{P} = \{\mathbf{p}_i\}_{i=1}^n$. Number of epochs S using private samples only. Noise standard deviation σ_{prp}.
IG: Fix an order π, and set $\pi^{(k)} = \pi, \forall k \in [K]$
SO: Generate permutation π of $[n]$, and set $\pi^{(k)} = \pi, \forall k \in [K]$
for Private epochs $k = 1, 2, \ldots, S$ **do**
  RR: Generate permutation $\pi^{(k)}$ of $[n]$
  for $i = 1, 2, \ldots, n$ **do**
    $\mathbf{x}_{i+1}^{(k)} \leftarrow \mathbf{x}_i^{(k)} - \eta\Big(\nabla f(\mathbf{x}_i^{(k)}; \mathbf{d}_{\pi_i^{(k)}}) + \rho_i^{(k)}\Big)$, where noise $\rho_i^{(k)} \sim \mathcal{N}(0, (\sigma_{\text{prp}})^2 \mathbb{I}_d)$
  end for
  $\mathbf{x}_1^{(k+1)} \leftarrow \text{argmin}_{\mathbf{x}\in\mathbb{R}^d}\, n\psi(\mathbf{x}) + \frac{\|\mathbf{x}-\mathbf{x}_{n+1}^{(k)}\|^2}{2\eta}$
end for
for Public epochs $k = S+1, \ldots, K$ **do**
  for $i = 1, 2, \ldots, n$ **do**
    $\mathbf{x}_{i+1}^{(k)} \leftarrow \mathbf{x}_i^{(k)} - \eta \nabla f(\mathbf{x}_i^{(k)}; \mathbf{p}_i)$
  end for
  $\mathbf{x}_1^{(k+1)} \leftarrow \text{argmin}_{\mathbf{x}\in\mathbb{R}^d}\, n\psi(\mathbf{x}) + \frac{\|\mathbf{x}-\mathbf{x}_{n+1}^{(k)}\|^2}{2\eta}$
end for
return $\mathbf{x}_1^{(K+1)}$

using the public dataset P. Specifically, in Algorithm 3:

1. For the first S epochs ($k \le S$): $n_d^{(k)} = n$ and $\mathsf{P}^{(k)} = \emptyset$,
2. For the remaining $K - S$ epochs ($k \ge S + 1$): $n_d^{(k)} = 0$ and $\mathsf{P}^{(k)} = \mathsf{P}$.

Consequently, during the first S epochs, there is no non-vanishing dissimilarity. In addition, to preserve privacy, noise with variance $\sigma^2_{\text{priv-pub}}$ is added during these epochs. During the last $K - S$ epochs, exclusively using the public data P results in the non-vanishing dissimilarity $C_n^{(k)} = C_n^{\text{full}}$, $\forall k \in \{S+1, \ldots, K\}$. No noise is needed during these epochs.

4.6.2 Algorithm 2: *Pub-Priv-ShuffleG*

The pseudocode of *Pub-Priv-ShuffleG* is presented in Algorithm 5.

Pub-Priv-ShuffleG trains models only on the public dataset P for the first S epochs, then switches to the private dataset D for the remaining $K - S$ epochs. In Algorithm 3:

Algorithm 5 *Pub-Priv-ShuffleG*

Input: Initial point $\mathbf{x}_1^{(1)}$, learning rate η, number of epochs K. Private dataset $\mathcal{D} = \{\mathbf{d}_i\}_{i=1}^n$. Public datasets $\mathcal{P} = \{\mathbf{p}_i\}_{i=1}^n$. Number of epochs S using public samples only. Noise standard deviation σ_{pup}.
IG: Fix an order π, and set $\pi^{(k)} = \pi, \forall k \in [K]$
SO: Generate permutation π of $[n]$, and set $\pi^{(k)} = \pi, \forall k \in [K]$
for Public epochs $k = 1, 2, \ldots, S$ **do**
  for $i = 1, 2, \ldots, n$ **do**
    $\mathbf{x}_{i+1}^{(k)} \leftarrow \mathbf{x}_i^{(k)} - \eta \nabla f(\mathbf{x}_i^{(k)}; \mathbf{p}_i)$
  end for
  $\mathbf{x}_1^{(k+1)} \leftarrow \text{argmin}_{\mathbf{x} \in \mathbb{R}^d} n\psi(\mathbf{x}) + \frac{\|\mathbf{x} - \mathbf{x}_{n+1}^{(k)}\|^2}{2\eta}$
end for
for Private epochs $k = S+1, \ldots, K$ **do**
  RR: Generate permutation $\pi^{(k)}$ of $[n]$
  for $i = 1, 2, \ldots, n$ **do**
    $\mathbf{x}_{i+1}^{(k)} \leftarrow \mathbf{x}_i^{(k)} - \eta\Big(\nabla f(\mathbf{x}_i^{(k)}; \mathbf{d}_{\pi_i^{(k)}}) + \rho_i^{(k)}\Big)$, where noise $\rho_i^{(k)} \sim \mathcal{N}(0, (\sigma_{\text{pup}})^2 \mathbb{I}_d)$
  end for
  $\mathbf{x}_1^{(k+1)} \leftarrow \text{argmin}_{\mathbf{x} \in \mathbb{R}^d} n\psi(\mathbf{x}) + \frac{\|\mathbf{x} - \mathbf{x}_{n+1}^{(k)}\|^2}{2\eta}$
end for
return $\mathbf{x}_1^{(K+1)}$

1. For the first S epochs ($k \leq S$), $n_d^{(k)} = 0$ and $\mathsf{P}^{(k)} = \mathsf{P}$,
2. For the remaining $K - S$ epochs ($k \geq S + 1$), $n_d^{(k)} = n$ and $\mathsf{P}^{(k)} = \emptyset$.

Consequently, during the first S epochs, the non-vanishing dissimilarity is $C_n^{(k)} = C_n^{\text{full}}$, $\forall k \in [S]$, and no additive noise is added. During the last $K - S$ epochs, there is no non-vanishing dissimilarity (i.e., $C_n^{(k)} = 0$, for $k \geq S + 1$), and noise with variance $\sigma^2_{\text{pub-priv}}$ is added.

4.6.3 Algorithm 3: *Interleaved-ShuffleG*

The pseudocode of *Interleaved-ShuffleG* is presented in Algorithm 6.

In each epoch $k \in [K]$, we fix $n_d^{(k)} = n_d$, where the first $n_d \in [n]$ steps use samples from the private dataset D for gradient computation, followed by $n - n_d$ steps using samples from the public dataset P. As a result, during every epoch, the first n_d steps involve no non-vanishing dissimilarity, while the remaining $n - n_d$ steps introduce dissimilarity arising from the use of samples from the public dataset. Specifically,

Algorithm 6 *Interleaved-ShuffleG*

Input: Initial point $\mathbf{x}_1^{(1)}$, learning rate η, number of epochs K. Private dataset $\mathcal{D} = \{\mathbf{d}_i\}_{i=1}^{n}$. Number of private samples to use per epoch $n_d \in (0, n)$. Public datasets $\mathcal{P} = \{\mathbf{p}_i\}_{i=1}^{n-n_d}$. Noise standard deviation σ_{int}.
IG: Fix an order π, and set $\pi^{(k)} = \pi, \forall k \in [K]$
SO: Generate permutation π of $[n]$, and set $\pi^{(k)} = \pi, \forall k \in [K]$
for $k = 1, 2, \ldots, K$ **do**
  RR: Generate permutation $\pi^{(k)}$ of $[n]$
  for $i = 1, 2, \ldots, n_d$ **do**
    $\mathbf{x}_{i+1}^{(k)} \leftarrow \mathbf{x}_i^{(k)} - \eta\Big(\nabla f(\mathbf{x}_i^{(k)}; \mathbf{d}_{\pi_i^{(k)}}) + \rho_i^{(k)}\Big)$, where noise $\rho_i^{(k)} \sim \mathcal{N}(0, (\sigma_{\text{int}})^2 \mathbb{I}_d)$
  end for
  for $i = n_d + 1, n_d + 2, \ldots, n$ **do**
    $\mathbf{x}_{i+1}^{(k)} \leftarrow \mathbf{x}_i^{(k)} - \eta\Big(\nabla f(\mathbf{x}_i^{(k)}; \mathbf{p}_{i-n_d}) + \rho_i^{(k)}\Big)$, where noise $\rho_i^{(k)} \sim \mathcal{N}(0, (\sigma_{\text{int}})^2 \mathbb{I}_d)$
  end for
  $\mathbf{x}_1^{(k+1)} \leftarrow \operatorname{argmin}_{\mathbf{x} \in \mathbb{R}^d} n\psi(\mathbf{x}) + \frac{\|\mathbf{x} - \mathbf{x}_{n+1}^{(k)}\|^2}{2\eta}$
end for
return $\mathbf{x}_1^{(K+1)}$

we denote the non-vanishing dissimilarity as $C_n^{(k)} = C_n^{\text{part}}$, $\forall k \in [K]$. Noise with variance σ_{int}^2 is added at *every* gradient step, even when using public samples, to enable privacy amplification by a factor of $1/(n - n_d + 1)$ [39]. The idea is to use noise from public gradients to further obscure the influence of private data.

4.6.4 Convergence and Privacy

We summarize the key parameters of each algorithm in Table 4.1. The specific parameter choices for each algorithm result in different dissimilarity measures and the maximum smoothness parameter, both of which are critical factors in the convergence bound. Their values are summarized in Table 4.2. We present the convergence bound and the privacy analysis of each algorithm in the following sections.

Convergence

We first present the convergence of each algorithm, as corollaries of Theorem 4.4.7, in Corollary 4.6.1. Note that to ensure the following bounds are tight, we enforce that the number of pre-determined epochs to be $S \in \{1, 2, \ldots, K-1\}$.

Parameters \ Algorithm	*Priv-Pub-ShuffleG*	*Pub-Priv-ShuffleG*	*Interleaved-ShuffleG*
# private samples used: $n_d^{(k)}$	$= \begin{cases} n & \text{if } k \le S^{\S} \\ 0 & \text{if } S < k \le K \end{cases}$	$= \begin{cases} 0 & \text{if } k \le S \\ n & \text{if } S < k \le K \end{cases}$	$n_d^{(k)} = n_d^{\dagger}, \forall k \in [K]$
Noise variance: $(\sigma^{(k)})^2$	$= \begin{cases} (\sigma_{\text{prp}})^2 & \text{if } k \le S \\ 0 & \text{if } S < k \le K \end{cases}$	$= \begin{cases} 0 & \text{if } k \le S \\ (\sigma_{\text{pup}})^2 & \text{if } S < k \le K \end{cases}$	$= (\sigma_{\text{int}})^2, \forall k \in [K]$

§ $S \in \{1, 2, \ldots, K-1\}$ is a pre-determined number of epochs.
† $n_d \in [n]$ is a pre-determined number of private samples to use in every epoch.

Table 4.1: Parameters of different algorithms that leverage public data samples.

Terms \ Algorithm	*Priv-Pub-ShuffleG*	*Pub-Priv-ShuffleG*	*Interleaved-ShuffleG*
Dissimilarity (Non-vanishing): $C_n^{(k)}$	$= \begin{cases} 0 & \text{if } k \le S \\ C_n^{\text{full}} & \text{if } S < k \le K \end{cases}$	$= \begin{cases} C_n^{\text{full}} & \text{if } k \le S \\ 0 & \text{if } k < S \le K \end{cases}$	$= C_n^{\text{part}}, \forall k \in [K]$
Dissimilarity: $\frac{1}{n}\sum_{i=1}^{n-1}(C_i^{(k)})^2$	$= \begin{cases} 0 & \text{if } k \le S \\ \frac{1}{n}\sum_{i=1}^{n-1}(C_i^{\text{full}})^2 & \text{if } S < k \le K \end{cases}$	$= \begin{cases} \frac{1}{n}\sum_{i=1}^{n-1}(C_i^{\text{full}})^2 & \text{if } k \le S \\ 0 & \text{if } S < k \le K \end{cases}$	$= \frac{1}{n}\sum_{i=n_d+1}^{n-1}(C_i^{\text{part}})^2, \forall k \in [K]$
Max smoothness parameter $\widehat{L}^{(k)*}$	$= \begin{cases} L^* & \text{if } k \le S \\ \widetilde{L}^* & \text{if } S < k \le K \end{cases}$	$= \begin{cases} \widetilde{L}^* & \text{if } k \le S \\ L^* & \text{if } S < k \le K \end{cases}$	$= \max\{L^*, \widetilde{L}^*\}$

Table 4.2: The resulting dissimilarity measures and the maximum smoothness parameters of different algorithms. Here, C_n^{full} measures the dissimilarity between $\mathcal{D}$ and $\mathcal{P}$ over the full datasets. C_n^{part} measures the dissimilarity between $\mathcal{D}$ and using the first $n - n_d$ samples from $\mathcal{P}$. This notion similarly extends to C_i^{part} and C_i^{full}, for $i < n$.

Corollary 4.6.1 (Convergence of Public Data Assisted Optimization)**.** *If one instantiates Algorithm 3 with parameters indicated in Table 4.1, then under Assumptions 4.4.1, 4.4.2, 4.4.4, 4.4.5 and 4.4.6, for $\beta > 0$, if $\mu_\psi \ge L_H^{(k)} + \beta$, $\forall k \in [K]$, and $\eta \le \frac{1}{2n \max\{L^*, \widetilde{L}^*\}\sqrt{10(1+\log K)}}$, Algorithm 3 guarantees convergence*

$$\mathbb{E}\left[G(\mathbf{x}_1^{(K+1)})\right] - \mathbb{E}\left[G(\mathbf{x}^*)\right] \le \underbrace{\frac{\|\mathbf{x}_1^{(1)} - \mathbf{x}^*\|^2}{\eta n K}}_{\textit{Initialization}} + \underbrace{10\eta^2 n^2 \sigma_{any}^2 (1 + \log K) \max\{L, \widetilde{L}\}}_{\textit{Optimization Uncertainty}} + 2M$$

where

- *For* ***Priv-Pub-ShuffleG****,*

$$M = \underbrace{\frac{1 + \log(K - S)}{2n^2\beta}(C_n^{full})^2}_{\textit{Non-vanishing Dissimilarity}} + \underbrace{5\eta^2(1 + \log(K - S))\frac{1}{n}\sum_{i=1}^{n-1}(C_i^{full})^2\widetilde{L}^*}_{\textit{Vanishing Dissimilarity}}$$

$$+\underbrace{6\eta^2 nd(1+\log S)(\sigma_{prp})^2 L^*}_{\textit{Injected Noise}}$$

- *For* ***Pub-Priv-ShuffleG****,*

$$M = \underbrace{\frac{1+\log S}{2n^2\beta}(C_n^{full})^2}_{\textit{Non-vanishing Dissimilarity}} + \underbrace{5\eta^2(1+\log S)\frac{1}{n}\sum_{i=1}^{n-1}(C_i^{full})^2\widetilde{L}^*}_{\textit{Vanishing Dissimilarity}}$$
$$+\underbrace{6\eta^2 nd(1+\log(K-S))(\sigma_{pup})^2 L^*}_{\textit{Injected Noise}}$$

- *For* ***Interleaved-ShuffleG****,*

$$M = \underbrace{\frac{1+\log K}{2n^2\beta}(C_n^{part})^2}_{\textit{Non-vanishing Dissimilarity}} + \underbrace{5\eta^2(1+\log K)\frac{1}{n}\sum_{i=n_d+1}^{n-1}(C_i^{part})^2\max\{L^*,\widehat{L}^*\}}_{\textit{Vanishing Dissimilarity}}$$
$$+\underbrace{6\eta^2 nd(1+\log K)(\sigma_{int})^2\max\{L^*,\widehat{L}^*\}}_{\textit{Injected Noise}}$$

Privacy Analysis

In this section, we derive the privacy guarantees of the three algorithms that leverage public samples: *Priv-Pub-ShuffleG*, *Pub-Priv-ShuffleG* and *Interleaved-ShuffleG*.

Lemma 4.6.2 (Privacy of Public Data Assisted Optimization)**.** *Under Assumptions 4.4.1, 4.4.2 and 4.4.3, if the learning rate is* $\eta \le 1/L^*$*,*

- *Priv-Pub-ShuffleG is* $(\frac{2\alpha G^2 S}{(\sigma_{prp})^2} + \frac{\log 1/\delta}{\alpha-1}, \delta)$*-DP.*
- *Pub-Priv-ShuffleG is* $(\frac{2\alpha G^2 (K-S)}{(\sigma_{pup})^2} + \frac{\log 1/\delta}{\alpha-1}, \delta)$*-DP.*

and if the learning rate is $\eta \le 1/\max\{L^*, \widetilde{L}^*\}$*,*

- *Interleaved-ShuffleG is* $(\frac{2\alpha (G^*)^2 K}{(n+1-n_d)(\sigma_{int})^2} + \frac{\log 1/\delta}{\alpha-1}, \delta)$*-DP.*

Proof of Lemma 4.6.2. The proof is similar to the proof of Lemma 4.5.2 showing the privacy loss of *DP-ShuffleG*. If the learning rate is set as $\eta \le 1/L^*$ for *Priv-Pub-ShuffleG* and *Pub-Priv-ShuffleG* and $\eta \le 1/\max\{L, L^*\}$ for *Interleaved-ShuffleG*, it is then guaranteed that each gradient step in one epoch is "contractive", which enables us to apply PABI (Theorem 4.3.9) to reason about the per-epoch privacy loss.

The per-epoch privacy loss of *Priv-Pub-ShuffleG* and *Pub-Priv-ShuffleG* is the same as the per-epoch privacy loss in *DP-ShuffleG* whenever the private dataset $\mathcal{D}$ is used during the epoch. Specifically, for $\alpha > 1$,

- For *Priv-Pub-ShuffleG*, $\mathbf{x}_1^{(k+1)}$ is $(\alpha, \frac{2\alpha G^2}{(\sigma_{\text{prp}})^2})$-RDP, if $k \leq S$ and there is no privacy loss 0 otherwise.
- For *Pub-Priv-ShuffleG*, $\mathbf{x}_1^{(k+1)}$ is $(\alpha, \frac{2\alpha G^2}{(\sigma_{\text{pup}})^2})$-RDP, if $S+1 \leq k \leq K$, and there is no privacy loss 0 otherwise.

By applying composition (Proposition 4.3.5) across K and $K-S$ epochs for *Priv-Pub-ShuffleG* and *Pub-Priv-ShuffleG*, respectively, and subsequently converting the RDP bound to a DP bound using Proposition 4.3.4, we obtain the overall privacy loss as stated in the lemma statement.

For *Interleaved-ShuffleG*, we consider two neighboring datasets $\mathcal{D} = \{\mathbf{d}_1, \ldots, \mathbf{d}_t, \ldots, \mathbf{d}_n\}$ and $\mathcal{D}' = \{\mathbf{d}_1, \ldots, \mathbf{d}'_t, \ldots, \mathbf{d}_n\}$ that differ at some index $t \in [n]$. In every epoch, only the first n_d steps use samples from the private dataset $\mathcal{D}$, and the remaining steps all use public samples. Hence, t can only occur at step $j \leq n_d$ in the sequence of updates in every epoch.

We apply Theorem 4.3.9 with $a_1, \ldots, a_{j-1} = 0$ and $a_j, \ldots, a_n = \frac{2\eta G^*}{n-j+1}$, where $j \leq n_d$. Note that $s_{\pi_j^{(k)}} = 2\eta G^*$ and $s_i = 0, \forall i \neq \pi_j^{(k)}$. In addition, $z_i \geq 0$ for all $i \leq n$ and $z_n = 0$. Hence,

$$D_\alpha(\mathbf{x}_{n+1}^{(k)} \parallel (\mathbf{x}_{n+1}^{(k)})') \leq \sum_{i=j}^{n} \frac{\alpha}{2\eta^2(\sigma_{\text{int}})^2} \cdot \frac{4\eta^2(G^*)^2}{(n-j+1)^2} = \frac{2\alpha(G^*)^2}{(\sigma_{\text{int}})^2(n-j+1)}$$

where $(\mathbf{x}_{n+1}^{(k)})'$ is the point obtained by optimizing on the neighboring dataset $\mathcal{D}'$.

Since $j \leq n_d$, the maximum privacy loss happens at $j = n_d$, that is, when the sample $\mathbf{d}_t$ is used at step n_d in one epoch. And so

$$\max D_\alpha(\mathbf{x}_{n+1}^{(k)} \parallel (\mathbf{x}_{n+1}^{(k)})') \leq \frac{2\alpha(G^*)^2}{(\sigma_{\text{int}})^2(n-n_d+1)}$$

which implies $\mathbf{x}_{n+1}^{(k)}$ and hence, $\mathbf{x}_1^{(k+1)}$, is $(\alpha, \frac{2\alpha(G^*)^2}{(\sigma_{\text{int}})^2(n-n_d+1)})$-RDP.

Applying composition (Proposition 4.3.5) across K epochs and converting the RDP bound to a DP bound using Proposition 4.3.4 leads to the overall privacy loss

as stated in the lemma statement.

□

4.6.5 Computing the Empirical Excess Risk

In this section, we derive the empirical excess risk of the three algorithms that leverage public samples: *Priv-Pub-ShuffleG*, *Pub-Priv-ShuffleG* and *Interleaved-ShuffleG*.

We begin by determining the optimal order α in in the RDP bound that minimizes the privacy loss, and the resulting amount of noise required for each algorithm to ensure the algorithm satisfies (ϵ, δ)-DP, as summarized in Table 4.3.

Algorithm \ Term	Renyi Order α	Noise Variance
Priv-Pub-ShuffleG	$\alpha = \frac{\sigma_{\text{prp}}\sqrt{\log 1/\delta}}{G\sqrt{2S}}$	$(\sigma_{\text{prp}})^2 = \mathcal{O}\left(\frac{G^2 S \log 1/\delta}{\epsilon^2}\right)$
Pub-Priv-ShuffleG	$\alpha = \frac{\sigma_{\text{pup}}\sqrt{\log 1/\delta}}{G\sqrt{2(K-S)}}$	$(\sigma_{\text{pup}})^2 = \mathcal{O}\left(\frac{G^2 (K-S) \log 1/\delta}{\epsilon^2}\right)$
Interleaved-ShuffleG	$\alpha = \frac{\sigma_{\text{int}}\sqrt{(n+1-n_d)\log 1/\delta}}{G^*\sqrt{2K}}$	$(\sigma_{\text{int}})^2 = \mathcal{O}\left(\frac{(G^*)^2 K \log 1/\delta}{\epsilon^2 (n+1-n_d)}\right)$

Table 4.3: Choices of the order α in the RDP bound (Lemma 4.6.2) and the resulting amount of noise required for each algorithm to ensure the output $\mathbf{x}_1^{(K+1)}$ satisfies (ϵ, δ)-DP.

Based on Corollary 4.6.1, we set the learning rate as $\eta = \mathcal{O}\left(\frac{\|\mathbf{x}_1^{(1)} - \mathbf{x}^*\|^{2/3}}{n \max\{L^*, \widetilde{L}^*\}(K(1+\log K))^{1/3}}\right)$ in *Priv-Pub-ShuffleG*, *Pub-Priv-ShuffleG* and *Interleaved-ShuffleG* to minimize the convergence bound, while satisfying the conditions of both Corollary 4.6.1 (convergence) and Lemma 4.6.2 (privacy). After choosing the learning rate, the convergence bounds of the algorithms are now given by

- *Priv-Pub-ShuffleG*:

$$\mathbb{E}\left[G(\mathbf{x}_1^{(K+1)})\right] - \mathbb{E}\left[G(\mathbf{x}^*)\right] \leq \underbrace{\mathcal{O}\left(\frac{\|\mathbf{x}_1^{(1)} - \mathbf{x}^*\|^{4/3} \max\{L^*, \widetilde{L}^*\}(1+\log K)^{1/3}}{K^{2/3}}\right)}_{\text{Initialization}}$$
$$+ \underbrace{\mathcal{O}\left(\frac{\|\mathbf{x}_1^{(1)} - \mathbf{x}^*\|^{4/3} \sigma_{any}^2 (1+\log K)^{1/3}}{K^{2/3} \max\{L^*, \widetilde{L}^*\}}\right)}_{\text{Optimization Uncertainty}} + \underbrace{\mathcal{O}\left(\frac{1+\log(K-S)}{n^2\beta}(C_n^{\text{full}})^2\right)}_{\text{Non-vanishing Dissimilarity}}$$

$$+\underbrace{\mathcal{O}\left(\frac{\|\mathbf{x}_1^{(1)}-\mathbf{x}^*\|^{4/3}\frac{1}{n}\sum_{i=1}^n (C_i^{\text{full}})^2}{n^2\max\{L^*,\widetilde{L}^*\}K^{2/3}}\cdot\frac{(1+\log(K-S))}{(1+\log K)^{1/3}}\right)}_{\text{Vanishing Dissimilarity}}$$

$$+\underbrace{\mathcal{O}\left(\frac{\|\mathbf{x}_1^{(1)}-\mathbf{x}^*\|^{4/3}dG^2\log(1/\delta)S}{n\max\{L^*,\widetilde{L}^*\}\epsilon^2K^{2/3}}\cdot\frac{(1+\log S)}{(1+\log K)^{2/3}}\right)}_{\text{Noise Injection}}$$

- *Pub-Priv-ShuffleG*:

$$\mathbb{E}\left[G(\mathbf{x}_1^{(K+1)})\right]-\mathbb{E}\left[G(\mathbf{x}^*)\right]\leq\underbrace{\mathcal{O}\left(\frac{\|\mathbf{x}_1^{(1)}-\mathbf{x}^*\|^{4/3}\max\{L^*,\widetilde{L}^*\}(1+\log K)^{1/3}}{K^{2/3}}\right)}_{\text{Initialization}}$$

$$+\underbrace{\mathcal{O}\left(\frac{\|\mathbf{x}_1^{(1)}-\mathbf{x}^*\|^{4/3}\sigma_{any}^2(1+\log K)^{1/3}}{K^{2/3}\max\{L^*,\widetilde{L}^*\}}\right)}_{\text{Optimization Uncertainty}}+\underbrace{\mathcal{O}\left(\frac{1+\log S}{n^2\beta}(C_n^{\text{full}})^2\right)}_{\text{Non-vanishing Dissimilarity}}$$

$$+\underbrace{\mathcal{O}\left(\frac{\|\mathbf{x}_1^{(1)}-\mathbf{x}^*\|^{4/3}\frac{1}{n}\sum_{i=1}^n (C_i^{\text{part}})^2}{n^2\max\{L^*,\widetilde{L}^*\}K^{2/3}}\cdot\frac{1+\log S}{(1+\log K)^{2/3}}\right)}_{\text{Vanishing Dissimilarity}}$$

$$+\underbrace{\mathcal{O}\left(\frac{\|\mathbf{x}_1^{(1)}-\mathbf{x}^*\|^{4/3}dG^2\log(1/\delta)(K-S)}{n\max\{L^*,\widetilde{L}^*\}\epsilon^2K^{2/3}}\cdot\frac{1+\log(K-S)}{(1+\log K)^{2/3}}\right)}_{\text{Noise Injection}}$$

- *Interleaved-ShuffleG*:

$$\mathbb{E}\left[G(\mathbf{x}_1^{(K+1)})\right]-\mathbb{E}\left[G(\mathbf{x}^*)\right]\leq\underbrace{\mathcal{O}\left(\frac{\|\mathbf{x}_1^{(1)}-\mathbf{x}^*\|^{4/3}\max\{L^*,\widetilde{L}^*\}(1+\log K)^{1/3}}{K^{2/3}}\right)}_{\text{Initialization}}$$

$$+\underbrace{\mathcal{O}\left(\frac{\|\mathbf{x}_1^{(1)}-\mathbf{x}^*\|^{4/3}\sigma_{any}^2(1+\log K)^{1/3}}{K^{2/3}\max\{L^*,\widetilde{L}^*\}}\right)}_{\text{Optimization Uncertainty}}+\underbrace{\mathcal{O}\left(\frac{1+\log K}{n^2\beta}(C_n^{\text{part}})^2\right)}_{\text{Non-vanishing Dissimilarity}}$$

$$+\underbrace{\mathcal{O}\left(\frac{\|\mathbf{x}_1^{(1)}-\mathbf{x}^*\|^{4/3}\frac{1}{n}\sum_{i=n_d+1}^{n-1}(C_i^{\text{part}})^2}{n^2\max\{L^*,\widetilde{L}^*\}K^{2/3}}(1+\log K)^{1/3}\right)}_{\text{Vanishing Dissimilarity}}$$

$$+\underbrace{\mathcal{O}\left(\frac{\|\mathbf{x}_1^{(1)}-\mathbf{x}^*\|^{4/3}d(G^*)^2\log(1/\delta)K^{1/3}}{n\max\{L^*,\widetilde{L}^*\}\epsilon^2(n-1+n_d)}(1+\log K)^{1/3}\right)}_{\text{Noise Injection}}$$

4.6.6 Empirical Excess Risk Comparison

Using the above convergence bounds with appropriate choices of the noise variance $\{(\sigma_{\text{prp}})^2,(\sigma_{\text{pup}})^2,(\sigma_{\text{int}})^2\}$, the learning rate η and We now compare the empirical excess risk of the three algorithms by fixing the number of gradient steps in all three algorithms: K epochs, each with n gradient steps. Let p denote the fraction of gradient steps computed using private samples. Therefore, in *Priv-Pub-ShuffleG*, $S=pK$; in *Pub-Priv-ShuffleG*, $K-S=pK$; and in *Interleaved-ShuffleG*, $n_d^{(k)}=pn,\forall k$. For simplicity, we assume both pK and pn are integers, and restrict p to $[\frac{1}{K},1]$.

We ensure all the algorithms satisfy the same (ϵ,δ)-DP guarantee, and compare their empirical excess risk bounds in Table 4.4. The bounds in Table 4.4 illustrate a trade-off when using public samples. The first term, which reflects the cost of privacy, is reduced compared to *DP-ShuffleG* (since $p\leq 1$). However, due to the dissimilarity between the public and private datasets, we get an additional non-vanishing term.

Algorithm	Empirical Excess Risk
Priv-Pub-ShuffleG	$\widetilde{\mathcal{O}}\left(\left(\frac{p}{n}\right)^{2/3}\left(\frac{\sqrt{d}}{\epsilon}\right)^{4/3}+\frac{(C_n^{\text{full}})^2}{n^2\beta}\right)$
Pub-Priv-ShuffleG	$\widetilde{\mathcal{O}}\left(\left(\frac{p}{n}\right)^{2/3}\left(\frac{\sqrt{d}}{\epsilon}\right)^{4/3}+\frac{(C_n^{\text{full}})^2}{n^2\beta}\right)$
Interleaved-ShuffleG	$\widetilde{\mathcal{O}}\left(\left(\frac{1}{n[(1-p)n+1]}\right)^{2/3}\left(\frac{\sqrt{d}}{\epsilon}\right)^{4/3}+\frac{(C_n^{\text{part}})^2}{n^2\beta}\right)$

Table 4.4: Empirical excess risk in terms of dataset size n, model dimension d, privacy parameter ϵ, the fraction of gradient steps that use private samples $p\in[\frac{1}{K},1]$, and the dissimilarity measures C_n^{full} and C_n^{part}, defined in 4.6.1, 4.6.2 and 4.6.3. The notation $\widetilde{\mathcal{O}}$ suppresses logarithmic factors.

First, *Interleaved-ShuffleG* reduces the privacy-related (first) term more aggres-

sively than the other two schemes when $(\frac{1}{n[(1-p)n+1]})^{2/3} \leq (\frac{p}{n})^{2/3}$, which holds for $p \geq 1/n$. This improvement, as reflected in Lemma 4.6.2, results from the more effective use of PABI within each epoch, which causes a reduction in privacy loss by a factor of $\frac{1}{n+1-n_d}$. On the other hand, the privacy loss bounds for *Priv-Pub-ShuffleG* and *Pub-Priv-ShuffleG* remain independent of n.

To compare the second terms in 4.4, recall that C_n^{full} and C_n^{part} measure the dissimilarity when, respectively, all or a part of the n gradient steps in an epoch are computed using samples from the public dataset P. Clearly $C_n^{\text{part}} \leq C_n^{\text{full}}$, hence, the dissimilarity term for *Interleaved-ShuffleG* is lower compared to the other two schemes.

To summarize, *Interleaved-ShuffleG* achieves a lower empirical excess risk than the other two baselines, *Priv-Pub-ShuffleG* and *Pub-Priv-ShuffleG*. It also reduces the privacy-related term compared to *DP-ShuffleG*, at the cost of an additional error term due to dissimilarity.

4.7 Experiments

We evaluate the above discussed algorithms in various tasks and datasets. All experiments were run on a 2021 MacBook Pro laptop, with an Apple M1 Pro chip.

4.7.1 Tasks

We consider three tasks, each with a distinct objective function. For every task, we describe the component function $f(\mathbf{x}; \mathbf{q})$ on a given sample $\mathbf{q} \in \mathcal{D} \cup \mathcal{P}$, and the regularizer $\psi(\mathbf{x})$. The true and the surrogate objective functions are constructed based on f and ψ accordingly.

1. **Mean Estimation**: $f(\mathbf{x}; \mathbf{q}) = \frac{1}{2}\|\mathbf{x} - \mathbf{q}\|^2$. $\psi(\mathbf{x}) = \mathcal{I}\{\mathbf{x} \in \mathcal{B}_C\}$, where $\mathcal{B}_C$ is a ball of radius C at the origin.
2. **Ridge Regression**: Let $\mathbf{q} = (\mathbf{a}, y)$, where $\mathbf{a}$ and y represent the feature vector and the response, respectively. $f(\mathbf{x}; \mathbf{q}) = (\langle \mathbf{x}, \mathbf{a} \rangle - y)^2$, $\psi(\mathbf{x}) = \frac{\lambda_r}{2}\|\mathbf{x}\|^2$ for $\lambda_r > 0$.
3. **Lasso Logistic Regression**: Let $\mathbf{q} = (\mathbf{a}, y)$, for $y \in \{\pm 1\}$, represent the feature vector and label, respectively. $f(\mathbf{x}; \mathbf{q}) = -y \log(h(\mathbf{x}; \mathbf{a})) - (1 - y) \log(h(\mathbf{x}; \mathbf{a}))$,

where $h(\mathbf{x}; \mathbf{a}) = \frac{1}{1+\exp(-\langle \mathbf{x}, \mathbf{a} \rangle)}$. $\psi(\mathbf{x}) = \lambda_l \|\mathbf{x}\|_1$ for $\lambda_l > 0$.

4.7.2 Datasets

We use `MNIST-69` for mean estimation, `CIFAR-10` and `Crime` for linear regression, and `COMPAS` and `CreditCard` for logistic regression. A summary of the datasets is presented in Table 4.5.

Task	Dataset	n	d
Mean Estimation	`MNIST-69`	1000	784
Ridge Regression	`CIFAR-10`	1000	3072
	`Crime`	159	124
Lasso Logistic Regression	`COMPAS`	2013	11
	`CreditCard`	200	21

Table 4.5: A summary of datasets.

We construct the private ($\mathcal{D}$) and public ($\mathcal{P}$) sets of samples from each dataset for each task as follows:

1. **Mean Estimation.**
   - `MNIST-69`. $n = 1000, d = 784$. We want to estimate the average pixel intensity of a given digit. $\mathcal{D}$ consists of the first 1000 training samples of digit 6. $\mathcal{P}$ consists of the first 1000 training samples of digit 9, with each sample rotated 180° to mimic digit 6.
2. **Ridge Regression**:
   - `CIFAR-10`. $n = 1000$, $d = 3072$. The task is to predict the class of a given image. $\mathcal{D}$ contains 200 samples per class across 10 classes. $\mathcal{P}$ simulates a real-world scenario where collecting data from certain classes is difficult, containing samples from only the first 4 classes (250 samples per class).
   - `Crime`[4]. $n = 159$, $d = 124$. The task is to predict per capita violent crimes in a region. Data with missing entries is removed and split into two halves. $\mathcal{D}$ consists of one half, while $\mathcal{P}$ simulates corrupted data with a small random rotation: $\mathcal{P} = \boldsymbol{X}_0 \boldsymbol{R}$, where $\boldsymbol{R} = \mathbb{I}_d + \mathcal{N}(0, \mathbb{I}_d)$ and $\boldsymbol{X}_0$ represents the other half of the original dataset.

[4] Communities and Crime

3. **Lasso Logistic Regression**:

   - `COMPAS` [5]. $n = 2103, d = 11$. The task is to predict whether a criminal defendant will reoffend within two years. The dataset, known for biases in predictions across ethnic groups, is split into African-American ($\mathcal{P}$) and Caucasian ($\mathcal{D}$) groups. This split reflects real-world disparities in data distributions.
   - `CreditCard` [6]. $n = 200, d = 21$. The task is to predict whether a client defaults on their credit card payment. The dataset is split by education level: university-level ($\mathcal{P}$) and below high school ($\mathcal{D}$). The private dataset ($\mathcal{D}$) has a higher default rate, creating a balanced class distribution, while the public dataset ($\mathcal{P}$) exhibits an extremely low default rate.

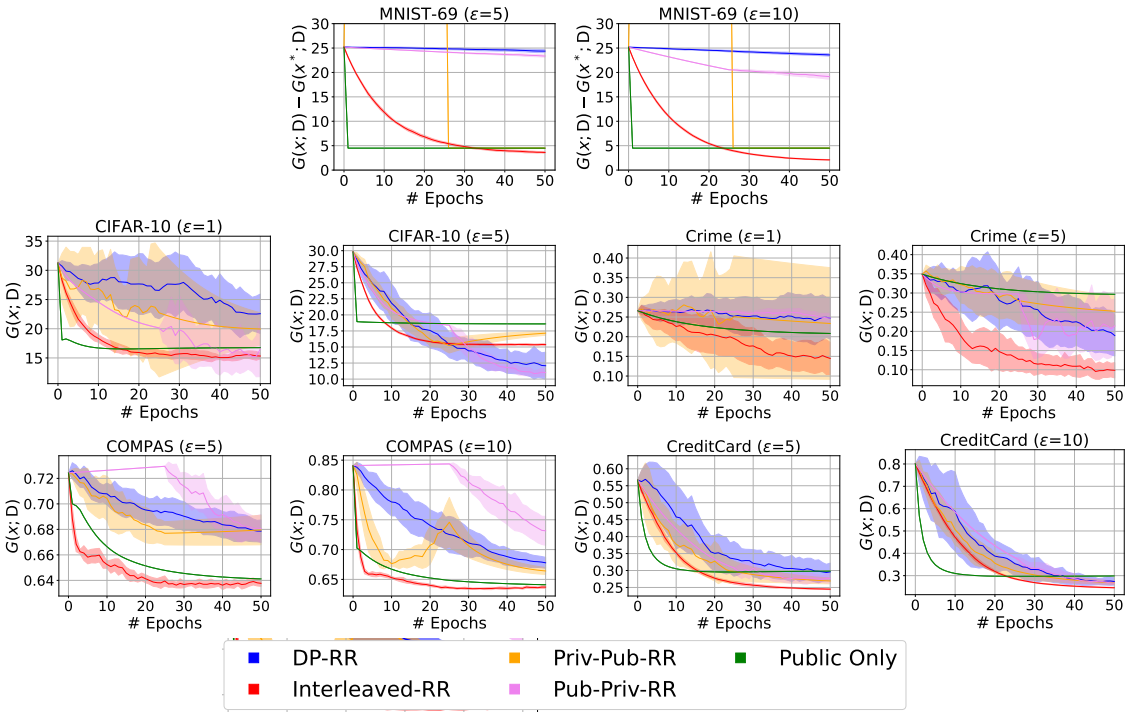

Figure 4.2: Results on each dataset across different tasks. Each algorithm runs for $K = 50$ epochs, with privacy loss $\epsilon \in \{1, 5, 10\}$ and $\delta = 10^{-6}$. The solid lines represent the mean performance, while the shaded regions denote one std. across 10 random runs.

[5] ProPublica Recidivism Dataset
[6] Default of Credit Card Clients

4.7.3 Baselines

In our experiments, all optimization algorithms apply Random Reshuffling (RR) to private samples. Thus, we replace "*ShuffleG*" in their names with RR, resulting in *Interleaved-RR*, *Priv-Pub-RR*, *Pub-Priv-RR* and *DP-RR*. And we include one additional baseline: *Public Only*, which uses only the public dataset without noise injection.

4.7.4 Hyperparameters

In algorithms that use public samples, we set percentage of private sample usage as $p = 0.5$. We set regularization parameters as $C = 10, \lambda_r = 0.1$, $\lambda_l = 0.1$. The number of epochs is $K = 50$. To ensure the Lipschitz continuity of the objectives, we apply gradient clipping with a norm of 10. The privacy parameters are $\delta = 10^{-6}$, with $\epsilon \in \{5, 10\}$ in mean estimation and lasso logistic regression, and $\epsilon \in \{1, 5\}$ in ridge regression. We perform a grid search on the learning rate $\eta \in \{0.5, 0.1, 0.05, 0.01, \ldots, 5 \times 10^{-9}, 10^{-9}\}$. Each experiment is repeated for 10 runs.

4.7.5 Results

All results are shown in Figure 4.2, with each color corresponding to a specific optimization algorithm. Solid lines represent the mean performance over 10 runs, and shaded areas indicate one standard deviation. Results are reported for the best hyperparameter setting, chosen based on the lowest last-iterate loss. Note that, due to this selection, results from the first half of epochs in *Pub-Priv-ShuffleG* or *Priv-Pub-ShuffleG* may not fully align with those of *Public Only* or *DP-ShuffleG*. Additional results of using other variants of *ShuffleG* to private samples and varying p can be found in Appendix C.3.

Discussion. Optimizing solely on the public dataset often leads to suboptimal solutions when the private and public datasets have slight distributional differences, as shown by the green curves. Conversely, relying only on the private dataset (i.e., *DP-ShuffleG*) is also suboptimal in high-privacy regimes, as shown by the blue curve, where excessive noise addition slows convergence. This is evident across all plots, except for `CIFAR-10` at $\epsilon = 5$, where the exception arises because larger ϵ values

require less noise and hence, and the benefits of incorporating public data are reduced. Moreover, in regimes with a smaller ϵ, *Interleaved-ShuffleG* consistently outperforms the baselines, as shown by the red curves. This is consistent with our theoretical findings.

4.8 Broader Impacts and Limitations

4.8.1 Broader Impacts

Our work is among the first to analyze the privacy-utility trade-offs of private shuffled gradient methods (*DP-ShuffleG*), which are widely used in modern deep learning frameworks but differ subtly from the theoretically studied DP-SGD. By focusing on *DP-ShuffleG*—implemented in libraries like TensorFlow Privacy—we highlight a critical yet often overlooked gap between theory and practice. Our analysis shows that *DP-ShuffleG* can suffer from worse empirical excess risk, underscoring the importance of understanding how implementation choices, such as using shuffling instead of i.i.d. sampling for gradient computation, impact privacy and utility. This has practical implications for developers and researchers who may unknowingly rely on mismatched assumptions when deploying private optimization methods. Moreover, we propose a novel training strategy that interleaves private and public samples to reduce empirical excess risk and improve the privacy-utility trade-off in shuffled gradient methods.

4.8.2 Limitations

Our analysis provides only an upper bound on the empirical excess risk of *DP-ShuffleG*, without a matching lower bound. While we have made a concerted effort to tighten this upper bound using all techniques available to us, to the best of our knowledge, the gap leaves open the possibility that the observed worse performance of *DP-ShuffleG* is an artifact of the analysis rather than an inherent limitation. Deriving a matching lower bound is challenging, as it requires fundamentally different tools and is non-standard—unlike typical lower bounds for problems, we seek one tailored to a specific algorithm. This makes the task highly non-trivial, and we leave it as an open problem.

Nevertheless, we conjecture that *DP-ShuffleG* inherently incurs higher empirical

excess risk than DP-SGD. As mentioned in the paragraph on **Comparison with DP-(S)GD** in Section 4.5, our result partially aligns with prior empirical findings [27], which show that even with minimal noise, *DP-ShuffleG* does not outperform DP-SGD on private binary classification tasks. See this section also for the intuition on why *DP-ShuffleG* achieves a higher empirical excess risk compared to DP-SGD.

4.9 Conclusion

We study private convex ERM problems solved via shuffled gradient methods (*DP-ShuffleG*) and provide the first empirical excess risk bound, which is larger than the lower bound. To reduce this risk, we incorporate public samples, and propose *Interleaved-ShuffleG*, which interleaves the usage of private and public samples during training. We demonstrate its superior performance compared to *DP-ShuffleG* and other baselines, theoretically and empirically.

Chapter 5

Conclusion

5.1 Summary

This thesis addresses two central challenges in federated and distributed learning: improving communication efficiency and improving privacy-model utility trade-offs in differentially private learning. Through three distinct yet thematically aligned works, we propose principled algorithmic techniques that contribute both practical and theoretical advancements.

First, in Chapter 2, we address the challenge of communication efficiency in distributed learning by proposing the Rand-Proj-Spatial estimator, a novel encoding-decoding scheme for distributed vector mean estimation. On the client side, Rand-Proj-Spatial generalizes the widely used Rand-k sparsification method and the previously proposed Rand-k-Spatial estimator by projecting vectors onto arbitrary k-dimensional subspaces, rather than selecting coordinates uniformly. On the server side, the decoding process explicitly leverages cross-client correlation to reduce the estimation error of the global mean. This approach yields significantly improved performance in terms of reduced mean squared error (MSE) over existing estimators both theoretically and empirically across a range of tasks. Looking ahead, promising directions of follow-up works include optimizing the computational efficiency of Rand-Proj-Spatial, characterizing its optimality within the class of non-adaptive estimators under correlation structure, and integrating quantization with sparsification to achieve tighter communication-accuracy trade-offs.

Second, in Chapter 3, we shift our focus to differential privacy in prediction tasks, and study privacy-utility trade-offs in the problem of computing a private majority from the outputs of K differentially private mechanisms. We propose propose the DaRRM framework—a general framework parameterized by a customizable noise function γ. We show that in the setting where client mechanisms are independent and identically distributed (i.i.d.) and satisfy pure differential privacy, the DaRRM framework can achieve privacy amplification by a factor of two over simple composition, under mild conditions. In the more general, non-i.i.d. setting, we introduce an efficient optimization-based algorithm that selects the optimal γ to maximize utility while satisfying differential privacy constraints. Empirical results further demonstrate that DaRRM with an optimized γ achieves substantially higher accuracy than existing baselines such as PATE [95], when the number of underlying private mechanisms K is small. This work opens new directions at the intersection of privacy theory and constrained optimization, and offers a new lens on how to aggregate private outputs with provable guarantees and improved utility.

Finally, in Chapter 4, we turn our attention to private convex empirical risk minimization (ERM) using shuffled gradient methods (*DP-ShuffleG*), a class of practical algorithms widely implemented in privacy-preserving machine learning applications. We provide the first empirical excess risk bound for *DP-ShuffleG*, revealing a gap between the achievable performance and known lower bounds. To mitigate this gap, we introduce *Interleaved-ShuffleG*, a novel method that strategically interleaves private and public data samples during training. This approach leverages the availability of public data to improve model utility without compromising privacy guarantees. We demonstrate both theoretically and empirically that *Interleaved-ShuffleG* outperforms *DP-ShuffleG* and other strong baselines, including the ones using public data, in terms of utility, evaluated using task-specific metrics, under the same privacy budget, highlighting its promise for improving privacy-utility trade-offs in practical learning systems.

Taken together, these contributions provide a multifaceted response to the dual challenges of communication efficiency and privacy in modern machine learning. They illustrate how problem-specific structure, careful algorithmic design, and hybrid data settings can be harnessed to build more scalable, trustworthy and privacy-preserving learning systems.

5.2 Future Work

While this thesis advances the goals of communication efficiency and data privacy, it also uncovers new challenges and opens up promising avenues for future exploration.

5.2.1 Communication Efficiency in Decentralized Settings and LLM-based Agent Systems

While the proposed Rand-Proj-Spatial estimator in this thesis targets distributed settings with a central server and multiple clients, real-world distributed learning increasingly involves decentralized architectures and Large Language Model (LLM)-based agent systems, where new communication challenges arise.

In decentralized settings, where clients exchange information over a dynamic communication graph without a central coordinator, it becomes critical to design communication-efficient protocols that adapt to changing topologies and ensure convergence under limited and potentially asymmetric bandwidth.

Furthermore, in multi-agent systems powered by large language models (LLMs), communication efficiency must account for the roles, relationships, and task-specific relevance of agents. Some agents may act as experts for particular subtasks and require more frequent or higher-priority communication, while others may share overlapping knowledge bases, making redundant communication wasteful. Future work could explore role-aware and topology-adaptive communication schemes that dynamically allocate bandwidth and compress shared information based on task dependencies, agent importance, and collaboration dynamics. Such approaches could significantly improve efficiency in emerging distributed learning paradigms beyond the classical client-server model.

5.2.2 Communication Efficiency with Algorithm-Hardware Co-design

Another promising direction is to approach communication efficiency not solely as an algorithmic challenge, but as a co-design problem involving both algorithms and hardware. As modern distributed and edge learning systems are increasingly deployed

on heterogeneous hardware platforms—ranging from smartphones and IoT devices to specialized AI chips, such as GPUs and TPUs—understanding how data is processed, stored, and transmitted at the hardware level becomes crucial.

Algorithm-hardware co-design aims to jointly optimize encoding schemes, memory access patterns, and communication protocols to match the underlying hardware constraints and capabilities. For example, certain sparsification or quantization strategies may be more efficient when aligned with hardware-friendly operations like SIMD instructions or GPU kernel scheduling. Future work could explore how to design communication-efficient algorithms that are aware of hardware bottlenecks and leverage architectural features such as low-precision arithmetic, on-chip memory hierarchies, or programmable network interfaces.

5.2.3 Protecting Data Privacy Beyond Differential Privacy

While differential privacy (DP) offers strong theoretical guarantees, its application in utility-sensitive domains like recommendation systems often leads to significant performance degradation. For instance, in recommendation systems, even a slight drop in prediction accuracy can result in substantial revenue losses, making practitioners hesitant to adopt DP-based methods. Moreover, while DP primarily guards against specific types of attacks, such as membership inference attacks, it may not sufficiently protect against other threats like attribute inference attacks [42]. These limitations highlight the need for alternative or complementary privacy-preserving techniques and more defense mechanisms against diverse attack vectors— for example, personalized privacy frameworks that tailor protection levels to individual user preferences and contexts may offer a more balanced trade-off between privacy and utility.

Appendix A

Correlated Distributed Mean Estimation

A.1 Additional Details on Motivation in Introduction

A.1.1 Preprocssing all client vectors by the same random matrix does not improve performance

Consider n clients. Suppose client i holds a vector $\mathbf{x}_i \in \mathbb{R}^d$. We want to apply Rand-$k$ or Rand-k-Spatial, while also making the encoding process more flexible than just randomly choosing k out of d coordinates. One naïve way of doing this is for each client to pre-process its vector by applying an orthogonal matrix $\boldsymbol{G} \in \mathbb{R}^{d \times d}$ that is the *same* across all clients. Such a technique might be helpful in improving the performance of quantization because the MSE due to quantization often depends on how uniform the coordinates of $\mathbf{x}_i$'s are, i.e. whether the coordinates of $\mathbf{x}_i$ have values close to each other. $\boldsymbol{G}$ is designed to be the random matrix (e.g. SRHT) that rotates $\mathbf{x}_i$ and makes its coordinates uniform.

Each client sends the server $\widehat{\mathbf{x}}_i = \boldsymbol{E}_i \boldsymbol{G} \mathbf{x}_i$, where $\boldsymbol{E}_i \in \mathbb{R}^{k \times d}$ is the subsamaping matrix. If we use Rand-k, the server can decode each client vector by first applying the decoding procedure of Rand-k and then rotating it back to the original space,

i.e., $\widehat{\mathbf{x}}_i^{(\text{Naïve})} = \frac{d}{k}\boldsymbol{G}^T\boldsymbol{E}_i^T\boldsymbol{E}_i\boldsymbol{G}\mathbf{x}_i$. Note that

$$\begin{aligned}\mathbb{E}[\widehat{\mathbf{x}}_i^{(\text{Naïve})}] &= \frac{d}{k}\mathbb{E}[\boldsymbol{G}^T\boldsymbol{E}_i^T\boldsymbol{E}_i\boldsymbol{G}\mathbf{x}_i] \\ &= \frac{d}{k}\boldsymbol{G}^T\frac{k}{d}\boldsymbol{I}_d\boldsymbol{G}\mathbf{x}_i \\ &= \mathbf{x}_i.\end{aligned}$$

Hence, $\widehat{\mathbf{x}}_i^{(\text{Naïve})}$ is unbiased. The MSE of $\widehat{\mathbf{x}}^{(\text{Naïve})} = \frac{1}{n}\sum_{i=1}^n \widehat{\mathbf{x}}_i^{(\text{Naïve})}$ is given as

$$\begin{aligned}&\mathbb{E}\left\|\bar{\mathbf{x}} - \widehat{\mathbf{x}}^{(\text{Naïve})}\right\|_2^2 = \mathbb{E}\left\|\frac{1}{n}\sum_{i=1}^n \mathbf{x}_i - \frac{1}{n}\frac{d}{k}\sum_{i=1}^n \boldsymbol{G}^T\boldsymbol{E}_i^T\boldsymbol{E}_i\boldsymbol{G}\mathbf{x}_i\right\|_2^2 \\ &= \frac{1}{n^2}\mathbb{E}\left\|\sum_{i=1}^n \mathbf{x}_i - \frac{d}{k}\sum_{i=1}^n \boldsymbol{G}^T\boldsymbol{E}_i^T\boldsymbol{E}_i\boldsymbol{G}\mathbf{x}_i\right\|_2^2 \\ &= \frac{1}{n^2}\left\{\frac{d^2}{k^2}\mathbb{E}\left\|\sum_{i=1}^n \boldsymbol{G}^T\boldsymbol{E}_i^T\boldsymbol{E}_i\boldsymbol{G}\mathbf{x}_i\right\|^2 - \left\|\sum_{i=1}^n \mathbf{x}_i\right\|^2\right\} \\ &= \frac{1}{n^2}\left\{\frac{d^2}{k^2}\left(\sum_{i=1}^n \mathbb{E}\|\boldsymbol{G}^T\boldsymbol{E}_i^T\boldsymbol{E}_i\boldsymbol{G}\mathbf{x}_i\|_2^2 + \sum_{i\neq j}\mathbb{E}\left\langle\boldsymbol{G}^T\boldsymbol{E}_i^T\boldsymbol{E}_i\boldsymbol{G}\mathbf{x}_i, \boldsymbol{G}\boldsymbol{E}_l^T\boldsymbol{E}_l\boldsymbol{G}\mathbf{x}_l\right\rangle\right) - \left\|\sum_{i=1}^n \mathbf{x}_i\right\|^2\right\}.\end{aligned} \tag{A.1}$$

Next, we bound the first term in Eq. A.1.

$$\begin{aligned}&\mathbb{E}\|\boldsymbol{G}^T\boldsymbol{E}_i^T\boldsymbol{E}_i\boldsymbol{G}\mathbf{x}_i\|_2^2 = \mathbb{E}[\mathbf{x}_i^T\boldsymbol{G}^T\boldsymbol{E}_i^T\boldsymbol{E}_i\boldsymbol{G}\boldsymbol{G}^T\boldsymbol{E}_i^T\boldsymbol{E}_i\boldsymbol{G}\mathbf{x}_i] = \mathbb{E}[\mathbf{x}_i^T\boldsymbol{G}^T\boldsymbol{E}_i^T\boldsymbol{E}_i\boldsymbol{E}_i^T\boldsymbol{E}_i\boldsymbol{G}\mathbf{x}_i] \\ &= \mathbf{x}_i^T\boldsymbol{G}^T\mathbb{E}[(\boldsymbol{E}_i^T\boldsymbol{E}_i)^2]\boldsymbol{G}\mathbf{x}_i \\ &= \mathbf{x}_i^T\frac{k}{d}\boldsymbol{I}_d\mathbf{x}_i \qquad (\because (\boldsymbol{E}_i^T\boldsymbol{E}_i)^2 = \boldsymbol{E}_i^T\boldsymbol{E}_i) \\ &= \frac{k}{d}\|\mathbf{x}_i\|_2^2\end{aligned} \tag{A.2}$$

The second term in Eq. A.1 can also be simplified as follows.

$$\begin{aligned}&\mathbb{E}[\langle\boldsymbol{G}^T\boldsymbol{E}_i^T\boldsymbol{E}_i\boldsymbol{G}\mathbf{x}_i, \boldsymbol{G}^T\boldsymbol{E}_l^T\boldsymbol{E}_l\boldsymbol{G}\mathbf{x}_l\rangle] \\ &= \langle\boldsymbol{G}^T\mathbb{E}[\boldsymbol{E}_i^T\boldsymbol{E}_i]\boldsymbol{G}\mathbf{x}_i, \boldsymbol{G}^T\mathbb{E}[\boldsymbol{E}_l^T\boldsymbol{E}_l]\boldsymbol{G}\mathbf{x}_l\rangle\end{aligned}$$

$$= \langle \boldsymbol{G}^T \frac{k}{d} \boldsymbol{I}_d \boldsymbol{G} \mathbf{x}_i, \boldsymbol{G}^T \frac{k}{d} \boldsymbol{I}_d \boldsymbol{G} \mathbf{x}_l \rangle$$
$$= \frac{k^2}{d^2} \langle \mathbf{x}_i, \mathbf{x}_l \rangle. \tag{A.3}$$

Plugging Eq. A.2 and Eq. A.3 into Eq. A.1, we get the MSE is

$$\begin{aligned}
&\mathbb{E}\|\bar{\mathbf{x}} - \widehat{\mathbf{x}}^{(\text{Naïve})}\|_2^2 \\
&= \frac{1}{n^2}\Big\{\frac{d^2}{k^2}\Big(\sum_{i=1}^n \frac{k}{d}\|\mathbf{x}_i\|_2^2 + 2\sum_{i=1}^n \sum_{l=i+1}^n \frac{k^2}{d^2}\langle \mathbf{x}_i, \mathbf{x}_l\rangle\Big) - \Big\|\sum_{i=1}^n \mathbf{x}_i\Big\|^2\Big\} \\
&= \frac{1}{n^2}(\frac{d}{k} - 1)\sum_{i=1}^n \|\mathbf{x}_i\|_2^2,
\end{aligned}$$

which has exactly the same MSE as that of Rand-k. The problem is that if each client applies the same rotational matrix $\boldsymbol{G}$, simply rotating the vectors will not change the ℓ_2 norm of the decoded vector, and hence the MSE. Similarly, if one applies Rand-k-Spatial, one ends up having exactly the same MSE as that of Rand-k-Spatial as well. Hence, we need to design a new decoding procedure when the encoding procedure at the clients are more flexible.

A.1.2 $nk \gg d$ is not interesting

One can rewrite $\sum_{i=1}^n \boldsymbol{G}_i^T \boldsymbol{G}_i$ in the Rand-Proj-Spatial estimator (Eq. 2.5) as $\sum_{i=1}^n \boldsymbol{G}_i^T \boldsymbol{G}_i = \sum_{j=1}^{nk} \mathbf{g}_i \mathbf{g}_i^T$, where $\mathbf{g}_j \in \mathbb{R}^d$ and $\mathbf{g}_{ik}, \mathbf{g}_{ik+1}, \ldots, \mathbf{g}_{(i+1)k}$ are the rows of $\boldsymbol{G}_i$. Since when $nk \gg d$, $\sum_{j=1}^{nk} \mathbf{g}_j \mathbf{g}_j^T \to \mathbb{E}[\sum_{j=1}^n \mathbf{g}_j \mathbf{g}_j^T]$ due to Law of Large Numbers, one way to see the limiting MSE of Rand-Proj-Spatial when nk is large is to approximate $\sum_{i=1}^n \sum_{j=1}^{nk} \mathbf{g}_i \mathbf{g}_i^T$ by its expectation.

By Lemma 2.4.1, when $\boldsymbol{G}_i = \boldsymbol{E}_i$, Rand-Proj-Spatial recovers Rand-k-Spatial. We now discuss the limiting behavior of Rand-k-Spatial when $nk \gg d$ by leveraging our proposed Rand-Proj-Spatial. In this case, each $\mathbf{g}_j$ can be viewed as a random based vector $\mathbf{e}_w$ for w randomly chosen in $[d]$. $\sum_{i=1}^{nk} \mathbf{g}_j \mathbf{g}_j^T \to \mathbb{E}[\sum_{i=1}^{nk} \mathbf{g}_j \mathbf{g}_j^T] = \sum_{i=1}^{nk} \frac{1}{d}\boldsymbol{I}_d = \frac{nk}{d}\boldsymbol{I}_d$. And so the scalar $\bar{\beta}$ in Eq. 2.5 to ensure an unbiased estimator is computed as

$$\bar{\beta}\mathbb{E}[(\frac{nk}{d}\boldsymbol{I}_d)^\dagger \boldsymbol{G}_i^T \boldsymbol{G}_i] = \boldsymbol{I}_d$$

$$\bar{\beta}\frac{d}{nk}\boldsymbol{I}_d\mathbb{E}[\boldsymbol{G}_i^T\boldsymbol{G}_i] = \boldsymbol{I}_d$$
$$\bar{\beta}\frac{d}{nk}\frac{k}{d} = \boldsymbol{I}_d$$
$$\bar{\beta} = n$$

And the MSE is now

$$\mathbb{E}\Big[\|\bar{\mathbf{x}} - \hat{\mathbf{x}}\|\Big] = \mathbb{E}\Big[\|\frac{1}{n}\sum_{i=1}^n \mathbf{x}_i - \frac{1}{n}\bar{\beta}\frac{d}{nk}\boldsymbol{I}_d\sum_{i=1}^n \boldsymbol{E}_i^T\boldsymbol{E}_i\mathbf{x}_i\|_2^2\Big]$$
$$= \frac{1}{n^2}\Big\{\bar{\beta}^2\frac{d^2}{n^2k^2}\mathbb{E}\Big[\|\sum_{i=1}^n \boldsymbol{E}_i^T\boldsymbol{E}_i\mathbf{x}_i\|_2^2\Big] - \|\sum_{i=1}^n \mathbf{x}_i\|_2^2\Big\}$$
$$= \frac{1}{n^2}\Big\{n^2\frac{d^2}{n^2k^2}\Big(\sum_{i=1}^n \mathbb{E}\Big[\|\boldsymbol{E}_i^T\boldsymbol{E}_i\mathbf{x}_i\|_2^2\Big] + 2\sum_{i=1}^n\sum_{l=i+1}^n \langle \boldsymbol{E}_i^T\boldsymbol{E}_i\mathbf{x}_i, \boldsymbol{E}_l^T\boldsymbol{E}_l\mathbf{x}_l\rangle\Big]\Big) - \|\sum_{i=1}^n \mathbf{x}_i\|_2^2\Big\}$$
$$= \frac{1}{n^2}\Big\{\frac{d^2}{k^2}\Big(\sum_{i=1}^n \mathbb{E}\Big[\mathbf{x}_i^T(\boldsymbol{E}_i^T\boldsymbol{E}_i)^2\mathbf{x}_i\Big] + 2\sum_{i=1}^n\sum_{l=i+1}^n \frac{k^2}{d^2}\langle \mathbf{x}_i, \mathbf{x}_l\rangle\Big) - \|\sum_{i=1}^n \mathbf{x}_i\|_2^2\Big\}$$
$$= \frac{1}{n^2}\Big\{\frac{d^2}{k^2}\Big(\sum_{i=1}^n \frac{k}{d}\|\mathbf{x}_i\|_2^2 + 2\sum_{i=1}^n\sum_{l=i+1}^n \frac{k^2}{d^2}\langle \mathbf{x}_i, \mathbf{x}_l\rangle\Big) - \sum_{i=1}^n \|\mathbf{x}_i\|_2^2 - 2\sum_{i=1}^n\sum_{l=i+1}^n \langle \mathbf{x}_i, \mathbf{x}_l\rangle\Big\}$$
$$= \frac{1}{n^2}(\frac{d}{k} - 1)\sum_{i=1}^n \|\mathbf{x}_i\|_2^2$$

which is exactly the same MSE as Rand-k. This implies when nk is large, the MSE of Rand-k-Spatial does not get improved compared to Rand-k with correlation information. Intuitively, this implies when $nk \gg d$, the server gets enough amount of information from the client, and does not need correlation to improve its estimator. Hence, we focus on the more interesting case when $nk < d$ — that is, when the server does not have enough information from the clients, and thus wants to use additional information, i.e. cross-client correlation, to improve its estimator.

A.2 Additional Details on the Rand-Proj-Spatial Family Estimator

A.2.1 $\bar{\beta}$ is a scalar

From Eq. A.9 in the proof of Theorem 2.4.3 and Eq. A.14 in the proof of Theorem 2.4.4, it is evident that the unbiasedness of the mean estimator $\widehat{\mathbf{x}}^{\text{Rand-Proj-Spatial}}$ is ensured collectively by

- The random sampling matrices $\{\boldsymbol{E}_i\}$.
- The orthogonality of scaled Hadamard matrices $\boldsymbol{H}^T\boldsymbol{H} = d\boldsymbol{I}_d = \boldsymbol{H}\boldsymbol{H}^T$.
- The rademacher diagonal matrices, with the property $(\boldsymbol{D}_i)^2 = \boldsymbol{I}_d$.

A.2.2 Alternative motivating regression problems

Alternative motivating regression problem 1.

Let $\boldsymbol{G}_i \in \mathbb{R}^{k\times d}$ and $\boldsymbol{W}_i \in \mathbb{R}^{d\times k}$ be the encoding and decoding matrix for client i. One possible alternative estimator that translates the intuition that the decoded vector should be close to the client's original vector, for all clients, is by solving the following regression problem,

$$\begin{aligned}\hat{\mathbf{x}} &= \underset{\boldsymbol{W}}{\operatorname{argmin}}\, f(\boldsymbol{W}) = \mathbb{E}[\|\bar{\mathbf{x}} - \frac{1}{n}\sum_{i=1}^{n}\boldsymbol{W}_i\boldsymbol{G}_i\mathbf{x}_i\|_2^2] \\ &\text{subject to } \bar{\mathbf{x}} = \frac{1}{n}\sum_{i=1}^{n}\mathbb{E}[\boldsymbol{W}_i\boldsymbol{G}_i\mathbf{x}_i] \end{aligned} \tag{A.4}$$

where $\boldsymbol{W} = (\boldsymbol{W}_1, \boldsymbol{W}_2, \ldots, \boldsymbol{W}_n)$ and the constraint enforces unbiasedness of the estimator. The estimator is then the solution of the above problem. However, we note that optimizing a decoding matrix $\boldsymbol{W}_i$ for each client leads to performing individual decoding of each client's compressed vector instead of a joint decoding process that considers all clients' compressed vectors. Only a joint decoding process can achieve the goal of leveraging cross-client information to reduce the estimation error. Indeed, we show as follows that solving the above optimization problem in Eq. A.4 recovers the MSE of our baseline Rand-k. Note

$$
\begin{aligned}
f(\boldsymbol{W}) &= \mathbb{E}[\|\frac{1}{n}\sum_{i=1}^{n}(\mathbf{x}_i - \boldsymbol{W}_i\boldsymbol{G}_i\mathbf{x}_i)\|_2^2] = \mathbb{E}[\|\frac{1}{n}\sum_{i=1}^{n}(\boldsymbol{I}_d - \boldsymbol{W}_i\boldsymbol{G}_i)\mathbf{x}_i\|_2^2] \\
&= \mathbb{E}\Big[\frac{1}{n^2}\Big(\sum_{i=1}^{n}\|(\boldsymbol{I}_d - \boldsymbol{W}_i\boldsymbol{G}_i\mathbf{x}_i)\|_2^2 + \sum_{i\neq j}\Big\langle(\boldsymbol{I}_d - \boldsymbol{W}_i\boldsymbol{G}_i)\mathbf{x}_i, (\boldsymbol{I}_d - \boldsymbol{W}_j\boldsymbol{G}_j)\mathbf{x}_j\Big\rangle\Big)\Big] \\
&= \frac{1}{n^2}\Big(\sum_{i=1}^{n}\mathbb{E}\Big[\|(\boldsymbol{I}_d - \boldsymbol{W}_i\boldsymbol{G}_i)\mathbf{x}_i\|_2^2\Big] + \sum_{i\neq j}\mathbb{E}\Big[\Big\langle(\boldsymbol{I}_d - \boldsymbol{W}_i\boldsymbol{G}_i)\mathbf{x}_i, (\boldsymbol{I}_d - \boldsymbol{W}_j\boldsymbol{G}_j)\mathbf{x}_j\Big\rangle\Big]\Big).
\end{aligned}
\tag{A.5}
$$

By the constraint of unbiasedness, i.e., $\bar{\mathbf{x}} = \frac{1}{n}\sum_{i=1}^{n}\mathbf{x}_i = \frac{1}{n}\sum_{i=1}^{n}\mathbb{E}[\boldsymbol{W}_i\boldsymbol{G}_i\mathbf{x}_i]$, there is

$$
\frac{1}{n}\sum_{i=1}^{n}\mathbf{x}_i - \frac{1}{n}\sum_{i=1}^{n}\mathbb{E}[\boldsymbol{W}_i\boldsymbol{G}_i\mathbf{x}_i] = 0 \Leftrightarrow \frac{1}{n}\sum_{i=1}^{n}\mathbb{E}[(\boldsymbol{I}_d - \boldsymbol{W}_i\boldsymbol{G}_i)\mathbf{x}_i] = 0.
$$

We now show that a sufficient and necessary condition to satisfy the above unbiasedness constraint is that for all $i \in [n]$, $\mathbb{E}[\boldsymbol{W}_i\boldsymbol{G}_i] = \boldsymbol{I}_d$.

Sufficiency. It is obvious that if for all $i \in [n]$, $\mathbb{E}[\boldsymbol{W}_i\boldsymbol{G}_i] = \boldsymbol{I}_d$, then we have $\frac{1}{n}\mathbb{E}[(\boldsymbol{I}_d - \boldsymbol{W}_i\boldsymbol{G}_i)\mathbf{x}_i] = 0$.

Necessity. Consider the special case that for some $i \in [n]$ and $\lambda \in [d]$, $\mathbf{x}_i = n\mathbf{e}_\lambda$, where $\mathbf{e}_\lambda$ is the λ-th canonical basis vector, and $\mathbf{x}_j = 0$, and for all $j \in [n] \setminus \{i\}$. Then,

$$
\mathbf{e}_\lambda = \bar{\mathbf{x}} = \frac{1}{n}\sum_{i=1}^{n}\mathbb{E}[\boldsymbol{W}_i\boldsymbol{G}_i\mathbf{x}_i] = \frac{1}{n}\mathbb{E}[\boldsymbol{W}_i\boldsymbol{G}_i]\mathbf{e}_\lambda = [\mathbb{E}[\boldsymbol{W}_i\boldsymbol{G}_i]]_\lambda,
$$

where $[\cdot]_\lambda$ denotes the λ-th column of matrix $\mathbb{E}[\boldsymbol{W}_i\boldsymbol{G}_i]$.

Since our approach is agnostic to the choice of vectors, we need this choice of decoder matrices, by varying λ over $[d]$, we see that we need $\mathbb{E}[\boldsymbol{W}_i\boldsymbol{G}_i] = \boldsymbol{I}_d$. And by varying i over $[n]$, we see that we need $\mathbb{E}[\boldsymbol{W}_j\boldsymbol{G}_j] = \boldsymbol{I}_d$ for all $j \in [n]$.

Therefore, $\bar{\mathbf{x}} = \frac{1}{n}\sum_{i=1}^{n}\mathbb{E}[\boldsymbol{W}_i\boldsymbol{G}_i\mathbf{x}_i] \Leftrightarrow \forall i \in [n], \mathbb{E}[\boldsymbol{W}_i\boldsymbol{G}_i] = \boldsymbol{I}_d$.

This implies the second term of $f(\boldsymbol{W})$ in Eq. A.5 is 0, that is,

$$
\sum_{i\neq j}\mathbb{E}\Big[\Big\langle(\boldsymbol{I}_d - \boldsymbol{W}_i\boldsymbol{G}_i)\mathbf{x}_i, (\boldsymbol{I}_d - \boldsymbol{W}_j\boldsymbol{G}_j)\mathbf{x}_j\Big\rangle = 0.
$$

Hence, we only need to solve

$$\hat{\mathbf{x}} = \underset{\boldsymbol{W}}{\operatorname{argmin}} f_2(\boldsymbol{W}) = \sum_{i=1}^{n} \mathbb{E}\Big[\|(\boldsymbol{I}_d - \boldsymbol{W}_i\boldsymbol{G}_i)\mathbf{x}_i\|_2^2\Big] \tag{A.6}$$

Since each $\boldsymbol{W}_i$ appears in $f_2(\boldsymbol{W})$ separately, each $\boldsymbol{W}_i$ can be optimized separately, via solving

$$\min_{\boldsymbol{W}_i} \mathbb{E}\Big[\|(\boldsymbol{I}_d - \boldsymbol{W}_i\boldsymbol{G}_i)\mathbf{x}_i\|_2^2\Big] \quad \text{subject to } \mathbb{E}[\boldsymbol{W}_i\boldsymbol{G}_i] = \boldsymbol{I}_d.$$

One natural solution is to take $\boldsymbol{W}_i = \frac{d}{k}\boldsymbol{G}_i^{\dagger}$, $\forall i \in [n]$. For $i \in [n]$, let $\boldsymbol{G}_i = \boldsymbol{V}_i\Lambda_i\boldsymbol{U}_i^T$ be its SVD, where $\boldsymbol{V}_i \in \mathbb{R}^{k\times d}$ and $\boldsymbol{U}_i \in \mathbb{R}^{d\times d}$ are orthogonal matrices. Then,

$$\boldsymbol{W}_i\boldsymbol{G}_i = \frac{d}{k}\boldsymbol{U}_i\Lambda_i^{\dagger}\boldsymbol{V}_i^T\boldsymbol{V}_i\Lambda\boldsymbol{U}^T = \frac{d}{k}\boldsymbol{U}_i\Lambda_i^{\dagger}\Lambda\boldsymbol{U}^T = \frac{d}{k}\boldsymbol{U}_i\Sigma\boldsymbol{U}_i^T,$$

where Σ is a diagonal matrix with 0s and 1s on the diagonal.

For simplicity, we assume the random matrix $\boldsymbol{U}_i$ follows a continuous distribution. $\boldsymbol{U}_i$ being discrete follows a similar analysis. Let $\mu(\boldsymbol{U}_i)$ be the measure of $\boldsymbol{U}_i$.

$$\begin{aligned}
\mathbb{E}[\boldsymbol{W}_i\boldsymbol{G}_i] &= \frac{d}{k}\mathbb{E}[\boldsymbol{U}_i\Sigma\boldsymbol{U}_i^T] = \frac{d}{k}\int_{\boldsymbol{U}_i}\mathbb{E}[\boldsymbol{U}_i\Sigma_i\boldsymbol{U}_i^T \mid \boldsymbol{U}_i]\cdot d\mu(\boldsymbol{U}_i)\\
&= \frac{d}{k}\int_{\boldsymbol{U}_i}\boldsymbol{U}_i\mathbb{E}[\Sigma_i \mid \boldsymbol{U}_i]\boldsymbol{U}_i^T\cdot\mu(\boldsymbol{U}_i)\\
&= \frac{d}{k}\int_{\boldsymbol{U}_i}\boldsymbol{U}_i\frac{k}{d}\boldsymbol{I}_d\boldsymbol{U}_i^T\cdot d\mu(\boldsymbol{U}_i)\\
&= \frac{d}{k}\frac{k}{d}\boldsymbol{I}_d = \boldsymbol{I}_d,
\end{aligned}$$

which means the estimator $\frac{1}{n}\sum_{i=1}^{n}\frac{k}{d}\boldsymbol{G}_i^{\dagger}\boldsymbol{G}_i$ satisfies unbiasedness. The MSE is now

$$\begin{aligned}
MSE &= \mathbb{E}\Big[\|\bar{\mathbf{x}} - \frac{1}{n}\sum_{i=1}^{n}\boldsymbol{W}_i\boldsymbol{G}_i\mathbf{x}_i\|_2^2\Big] = \frac{1}{n^2}\sum_{i=1}^{n}\mathbb{E}\Big[\|(\boldsymbol{I}_d - \boldsymbol{W}_i\boldsymbol{G}_i)\mathbf{x}_i\|_2^2\Big]\\
&= \frac{1}{n^2}\sum_{i=1}^{n}\Big(\|\mathbf{x}_i\|_2^2 + \mathbb{E}[\|\boldsymbol{W}_i\boldsymbol{G}_i\mathbf{x}_i\|_2^2] - 2\langle\mathbf{x}_i, \mathbb{E}[\boldsymbol{W}_i\boldsymbol{G}_i]\mathbf{x}_i\rangle\Big)
\end{aligned}$$

$$
\begin{aligned}
&= \frac{1}{n^2}\sum_{i=1}^{n}\Big(\|\mathbf{x}_i\|_2^2 + \mathbb{E}[\|\boldsymbol{W}_i\boldsymbol{G}_i\mathbf{x}_i\|_2^2] - 2\langle \mathbf{x}_i, \mathbf{x}_i\rangle\Big)\\
&= \frac{1}{n^2}\sum_{i=1}^{n}\Big(\mathbb{E}[\|\boldsymbol{W}_i\boldsymbol{G}_i\mathbf{x}_i\|_2^2 - \|\mathbf{x}_i\|_2^2]\Big)\\
&= \frac{1}{n^2}\sum_{i=1}^{n}\Big(\mathbf{x}_i\mathbb{E}[(\boldsymbol{W}_i\boldsymbol{G}_i)^T(\boldsymbol{W}_i\boldsymbol{G}_i)]\mathbf{x}_i - \|\mathbf{x}_i\|_2^2\Big).
\end{aligned}
$$

Again, let $\boldsymbol{G}_i = \boldsymbol{V}_i\Lambda_i\boldsymbol{U}_i^T$ be its SVD and consider $\boldsymbol{W}_i\boldsymbol{G}_i = \frac{d}{k}\boldsymbol{U}_i\Sigma_i\boldsymbol{U}_i^T$, where Σ_i is a diagonal matrix with 0s and 1s. Then,

$$
\begin{aligned}
MSE &= \frac{1}{n^2}\sum_{i=1}^{n}\sum_{i=1}^{n}\Big(\mathbf{x}_i^T\frac{d^2}{k^2}\mathbb{E}[\boldsymbol{U}_i\Sigma_i\boldsymbol{U}_i^T\boldsymbol{U}_i\Sigma_i\boldsymbol{U}_i^T]\mathbf{x}_i - \|\mathbf{x}_i\|_2^2\Big)\\
&= \frac{1}{n^2}\sum_{i=1}^{n}\Big(\frac{d^2}{k^2}\mathbf{x}_i^T\mathbb{E}[\boldsymbol{U}_i\Sigma^2\boldsymbol{U}_i^T]\mathbf{x}_i - \|\mathbf{x}_i\|_2^2\Big).
\end{aligned}
$$

Since $\boldsymbol{G}_i$ has rank k, Σ_i is a diagonal matrix with k out of d entries being 1 and the rest being 0. Let $\mu(\boldsymbol{U}_i)$ be the measure of $\boldsymbol{U}_i$. Hence, for $i \in [n]$,

$$
\begin{aligned}
\mathbb{E}[\boldsymbol{U}_i\Sigma_i^2\boldsymbol{U}_i^T] &= \int_{\boldsymbol{U}_i}\mathbb{E}[\boldsymbol{U}_i\Sigma_i^2\boldsymbol{U}_i^T \mid \boldsymbol{U}_i]d\mu(\boldsymbol{U}_i)\\
&= \int_{\boldsymbol{U}_i}\boldsymbol{U}_i\mathbb{E}[\Sigma_i^2 \mid \boldsymbol{U}_i]\boldsymbol{U}_i^T d\mu(\boldsymbol{U}_i)\\
&= \int_{\boldsymbol{U}_i}\frac{k}{d}\boldsymbol{U}_i\boldsymbol{I}_d\boldsymbol{U}_i^T d\mu(\boldsymbol{U}_i)\\
&= \frac{k}{d}\int_{\boldsymbol{U}_i}\boldsymbol{I}_d d\mu(\boldsymbol{U}_i)\\
&= \frac{k}{d}\boldsymbol{I}_d.
\end{aligned}
$$

Therefore, the MSE of the estimator, which is the solution of the optimization problem in Eq. A.4, is

$$
MSE = \frac{1}{n^2}\sum_{i=1}^{n}\Big(\frac{d^2}{k^2}\mathbf{x}_i^T\frac{k}{d}\boldsymbol{I}_d\mathbf{x}_i - \|\mathbf{x}_i\|_2^2\Big) = \frac{1}{n^2}(\frac{d}{k} - 1)\sum_{i=1}^{n}\|\mathbf{x}_i\|_2^2,
$$

which is the same MSE as that of Rand-k.

Alternative motivating regression problem 2.

Another motivating regression problem based on which we can design our estimator is

$$\widehat{\mathbf{x}} = \underset{\mathbf{x}}{\operatorname{argmin}} \|\frac{1}{n}\sum_{i=1}^{n} \mathbf{G}_i\mathbf{x} - \frac{1}{n}\sum_{i=1}^{n} \mathbf{G}_i\mathbf{x}_i\|_2^2 \tag{A.7}$$

Note that $\boldsymbol{G}_i \in \mathbb{R}^{k\times d}, \forall i \in [n]$, and so the solution to the above problem is

$$\widehat{\mathbf{x}}^{(\text{solution})} = \Big(\frac{1}{n}\sum_{i=1}^{n} \boldsymbol{G}_i\Big)^{\dagger}\Big(\frac{1}{n}\sum_{i=1}^{n} \boldsymbol{G}_i\mathbf{x}_i\Big),$$

and to ensure unbiasedness of the estimator, we can set $\bar{\beta} \in \mathbb{R}$ and have the estimator as

$$\widehat{\mathbf{x}}^{(\text{estimator})} = \bar{\beta}\Big(\frac{1}{n}\sum_{i=1}^{n} \boldsymbol{G}_i\Big)^{\dagger}\Big(\frac{1}{n}\sum_{i=1}^{n} \boldsymbol{G}_i\mathbf{x}_i\Big).$$

It is not hard to see this estimator does not lead to an MSE as low as Rand-Proj-Spatial does. Consider the full correlation case, i.e., $\mathbf{x}_i = \mathbf{x}, \forall i \in [n]$, for example, the estimator is now

$$\widehat{\mathbf{x}}^{(\text{estimator})} = \bar{\beta}\Big(\frac{1}{n}\sum_{i=1}^{n} \boldsymbol{G}_i\Big)^{\dagger}\Big(\frac{1}{n}\sum_{i=1}^{n} \boldsymbol{G}_i\Big)\mathbf{x}.$$

Note that $\operatorname{rank}(\frac{1}{n}\sum_{i=1}^{n} \boldsymbol{G}_i)$ is at most k, since $\boldsymbol{G}_i \in \mathbb{R}^{k\times d}$, $\forall i \in [k]$. This limits the amount of information of $\mathbf{x}$ the server can recover.

While recall that in this case, the Rand-Proj-Spatial estimator is

$$\widehat{\mathbf{x}}^{(\text{Rand-Proj-Spatial})} = \bar{\beta}\Big(\sum_{i=1}^{n} \boldsymbol{G}_i^T\boldsymbol{G}_i\Big)^{\dagger}\sum_{i=1}^{n} \boldsymbol{G}_i^T\boldsymbol{G}_i\mathbf{x} = \bar{\beta}\boldsymbol{S}^{\dagger}\boldsymbol{S}\mathbf{x},$$

where $\boldsymbol{S}$ can have rank at most nk.

A.2.3 Why deriving the MSE of Rand-Proj-Spatial with SRHT is hard

To analyze Eq. 2.11, one needs to compute the distribution of eigendecomposition of $\boldsymbol{S} = \sum_{i=1}^{n} \boldsymbol{G}_i^T \boldsymbol{G}_i$, i.e. the sum of the covariance of SRHT. To the best of our knowledge, there is no non-trivial closed form expression of the distribution of eigen-decomposition of even a single $\boldsymbol{G}_i^T \boldsymbol{G}_i$, when $\boldsymbol{G}_i$ is SRHT, or other commonly used random matrices, e.g. Gaussian. When $\boldsymbol{G}_i$ is SRHT, since $\boldsymbol{G}_i^T \boldsymbol{G}_i = \boldsymbol{D}_i \boldsymbol{H} \boldsymbol{E}_i^T \boldsymbol{E}_i \boldsymbol{H} \boldsymbol{D}_i$ and the eigenvalues of $\boldsymbol{E}_i^T \boldsymbol{E}_i$ are just diagonal entries, one might attempt to analyze $\boldsymbol{H}\boldsymbol{D}_i$. While the hardmard matrix $\boldsymbol{H}$'s eigenvalues and eigenvectors are known[1], the result can hardly be applied to analyze the distribution of singular values or singular vectors of $\boldsymbol{H}\boldsymbol{D}_i$.

Even if one knows the eigen-decomposition of a single $\boldsymbol{G}_i^T \boldsymbol{G}_i$, it is still hard to get the eigen-decomposition of $\boldsymbol{S}$. The eigenvalues of a matrix $\boldsymbol{A}$ can be viewed as a non-linear function in the $\boldsymbol{A}$, and hence it is in general hard to derive closed form expressions for the eigenvalues of $\boldsymbol{A} + \boldsymbol{B}$, given the eigenvalues of $\boldsymbol{A}$ and that of $\boldsymbol{B}$. One exception is when $\boldsymbol{A}$ and $\boldsymbol{B}$ have the same eigenvector and the eigenvalues of $\boldsymbol{A} + \boldsymbol{B}$ becomes a sum of the eigenvalues of $\boldsymbol{A}$ and $\boldsymbol{B}$. Recall when $\boldsymbol{G}_i = \boldsymbol{E}_i$, Rand-Proj-Spatial recovers Rand-k-Spatial. Since $\boldsymbol{E}_i^T \boldsymbol{E}_i$'s all have the same eigenvectors (i.e. same as $\boldsymbol{I}_d$), the eigenvalues of $\boldsymbol{S} = \sum_{i=1}^{n} \boldsymbol{E}_i^T \boldsymbol{E}_i$ are just the sum of diagonal entries of $\boldsymbol{E}_i^T \boldsymbol{E}_i$'s. Hence, deriving the MSE for Rand-k-Spatial is not hard compared to the more general case when $\boldsymbol{G}_i^T \boldsymbol{G}_i$'s can have different eigenvectors.

Since one can also view $\boldsymbol{S} = \sum_{i=1}^{nk} \mathbf{g}_i \mathbf{g}_i^T$, i.e. the sum of nk rank-one matrices, one might attempt to recursively analyze the eigen-decomposition of $\sum_{i=1}^{n'} \mathbf{g}_i \mathbf{g}_i^T + \mathbf{g}_{n'+1} \mathbf{g}_{n'+1}^T$ for $n' \leq n$. One related problem is eigen-decomposition of a low-rank updated matrix in perturbation analysis: Given the eigen-decomposition of a matrix $\boldsymbol{A}$, what is the eigen-decomposition of $\boldsymbol{A} + \boldsymbol{V}\boldsymbol{V}^T$, where $\boldsymbol{V}$ is low-rank matrix (or more commonly rank-one)? To compute the eigenvalues of $\boldsymbol{A} + \boldsymbol{V}\boldsymbol{V}^T$ directly from that of $\boldsymbol{A}$, the most effective and widely applied solution is to solve the so-called secular equation, e.g. [7, 47, 49]. While this can be done computationally efficiently, it is hard to get a closed form expression for the eigenvalues of $\boldsymbol{A} + \boldsymbol{V}\boldsymbol{V}^T$ from the secular equation.

[1] See this note https://core.ac.uk/download/pdf/81967428.pdf

The previous analysis of SRHT in e.g. [3, 69, 70, 71, 123] is based on asymptotic properties of SRHT, such as the limiting eigen-spectrum, or concentration bounds that bounds the singular values. To analyze the MSE of Rand-Proj-Spatial, however, we need an exact, non-asymptotic analysis of the distribution of SRHT. Concentration bounds does not apply, since computing the pseudo-inverse in Eq. 2.5 naturally bounds the eigenvalues, and applying concentration bounds will only lead to a loose upper bound on MSE.

A.2.4 More simulation results on incorporating various degrees of correlation

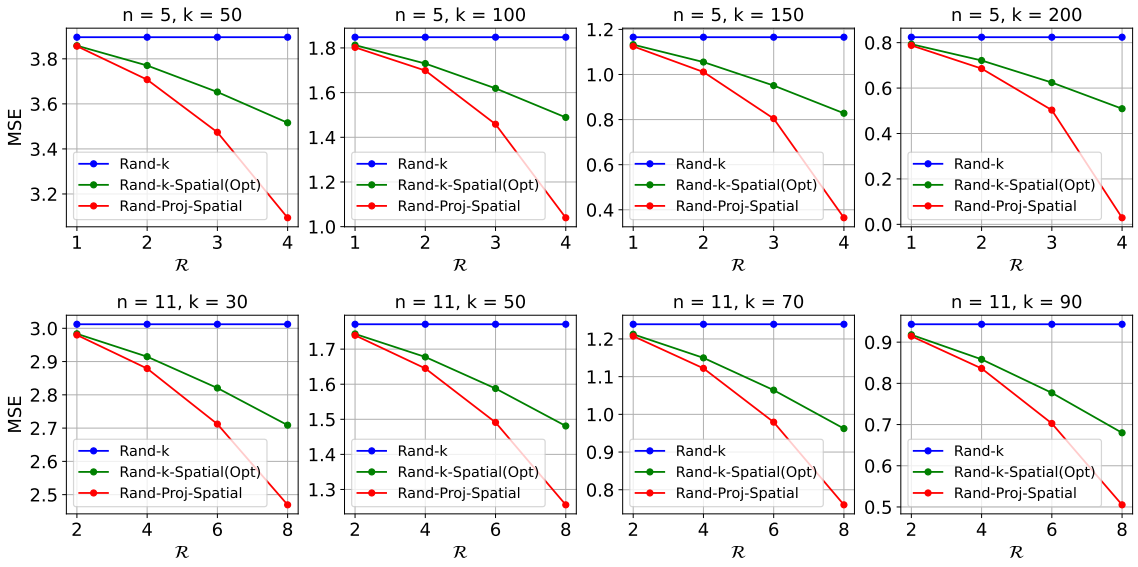

Figure A.1: MSE comparison of estimators Rand-k, Rand-k-Spatial(Opt), Rand-Proj-Spatial, given the degree of correlation $\mathcal{R}$. Rand-k-Spatial(Opt) denotes the estimator that gives the lowest possible MSE from the Rand-k-Spatial family. We consider $d = 1024$, a smaller number of clients $n \in \{5, 11\}$, and k values such that $nk < d$. In each plot, we fix n, k, d and vary the degree of positive correlation $\mathcal{R}$. Note the range of $\mathcal{R}$ is $\mathcal{R} \in [0, n-1]$. We choose $\mathcal{R}$ with equal space in this range.

A.3 All Proof Details

A.3.1 Proof of Theorem 2.4.3

Theorem 4.3 (MSE under Full Correlation)**.** *Consider n clients, each holding the same vector $\mathbf{x} \in \mathbb{R}^d$. Suppose we set $T(\lambda) = \lambda$, $\bar{\beta} = \frac{d}{k}$ in Eq. 2.5, and the random linear map $\boldsymbol{G}_i$ at each client to be an SRHT matrix. Let δ be the probability that $\boldsymbol{S} = \sum_{i=1}^n \boldsymbol{G}_i^T \boldsymbol{G}_i$ does not have full rank. Then, for $nk \leq d$,*

$$\mathbb{E}\Big[\|\widehat{\mathbf{x}}^{(\textit{Rand-Proj-Spatial(Max)})} - \bar{\mathbf{x}}\|_2^2\Big] \leq \Big[\frac{d}{(1-\delta)nk + \delta k} - 1\Big]\|\mathbf{x}\|_2^2 \tag{A.8}$$

Proof. All clients have the same vector $\mathbf{x}_1 = \mathbf{x}_2 = \cdots = \mathbf{x}_n = \mathbf{x} \in \mathbb{R}^d$. Hence, $\bar{\mathbf{x}} = \frac{1}{n}\sum_{i=1}^n \mathbf{x}_i = \mathbf{x}$, and the decoding scheme is

$$\widehat{\mathbf{x}}^{(\text{Rand-Proj-Spatial(Max)})} = \bar{\beta}\Big(\sum_{i=1}^n \boldsymbol{G}_i^T \boldsymbol{G}_i\Big)^\dagger \sum_{i=1}^n \boldsymbol{G}_i^T \boldsymbol{G}_i \mathbf{x} = \bar{\beta}\boldsymbol{S}^\dagger \boldsymbol{S}\mathbf{x},$$

where $\boldsymbol{S} = \sum_{i=1}^n \boldsymbol{G}_i^T \boldsymbol{G}_i$. Let $\boldsymbol{S} = \boldsymbol{U}\Lambda\boldsymbol{U}^T$ be its eigendecomposition. Since $\boldsymbol{S}$ is a real symmetric matrix, $\boldsymbol{U}$ is orthogonal, i.e., $\boldsymbol{U}^T\boldsymbol{U} = \boldsymbol{I}_d = \boldsymbol{U}\boldsymbol{U}^T$. Also, $\boldsymbol{S}^\dagger = \boldsymbol{U}\Lambda^\dagger\boldsymbol{U}^T$, where $\Lambda^\dagger$ is a diagonal matrix, such that

$$[\Lambda^\dagger]_{ii} = \begin{cases} 1/[\Lambda]_{ii} & \text{if } \Lambda_{ii} \neq 0, \\ 0 & \text{else.} \end{cases}$$

Let δ_c be the probability that $\boldsymbol{S}$ has rank c, for $c \in \{k, k+1, \ldots, nk-1\}$. Note that $\delta = \sum_{c=k}^{nk-1} \delta_c$. For vector $\mathbf{m} \in \mathbb{R}^d$, we use $\text{diag}(\mathbf{m}) \in \mathbb{R}^{d\times d}$ to denote the matrix whose diagonal entries correspond to the coordinates of $\mathbf{m}$ and the rest of the entries are zeros.

Computing $\bar{\beta}$. First, we compute $\bar{\beta}$. To ensure that our estimator $\widehat{\mathbf{x}}^{(\text{Rand-Proj-Spatial(Max)})}$ is unbiased, we need $\bar{\beta}\mathbb{E}[\boldsymbol{S}^\dagger\boldsymbol{S}\mathbf{x}] = \mathbf{x}$. Consequently,

$$\begin{aligned} \mathbf{x} &= \bar{\beta}\mathbb{E}[\boldsymbol{U}\Lambda^\dagger\boldsymbol{U}^T\boldsymbol{U}\Lambda\boldsymbol{U}^T]\mathbf{x} \\ &= \bar{\beta}\left[\sum_{\boldsymbol{U}=\Phi} \Pr[\boldsymbol{U} = \Phi]\mathbb{E}[\boldsymbol{U}\Lambda^\dagger\Lambda\boldsymbol{U}^T \mid \boldsymbol{U} = \Phi]\right]\mathbf{x} \end{aligned}$$

$$
\begin{aligned}
&= \bar{\beta}\left[\sum_{\boldsymbol{U}=\Phi} \Pr[\boldsymbol{U}=\Phi]\boldsymbol{U}\mathbb{E}[\Lambda^{\dagger}\Lambda \mid \boldsymbol{U}=\Phi]\boldsymbol{U}^T\right]\mathbf{x} \\
&\overset{(a)}{=} \bar{\beta}\left[\sum_{\boldsymbol{U}=\Phi} \Pr[\boldsymbol{U}=\Phi]\boldsymbol{U}\mathbb{E}[\mathrm{diag}(\mathbf{m}) \mid \boldsymbol{U}=\Phi]\boldsymbol{U}^T\right]\mathbf{x} \\
&\overset{(b)}{=} \bar{\beta}\sum_{\boldsymbol{U}=\Phi} \Pr[\boldsymbol{U}=\Phi]\left[\boldsymbol{U}\Big((1-\delta)\frac{nk}{d}\boldsymbol{I}_d + \sum_{c=k}^{nk-1}\delta_c\frac{c}{d}\boldsymbol{I}_d\Big)\boldsymbol{U}^T\right]\mathbf{x} \\
&= \bar{\beta}\Big[(1-\delta)\frac{nk}{d} + \sum_{c=k}^{nk-1}\delta_c\frac{c}{d}\Big]\mathbf{x} \\
\Rightarrow \bar{\beta} &= \frac{d}{(1-\delta)nk + \sum_{c=k}^{nk-1}\delta_c c}
\end{aligned} \tag{A.9}
$$

where in (a), $\mathbf{m} \in \mathbb{R}^d$ such that

$$
\mathbf{m}_i = \begin{cases} 1 & \text{if } \Lambda_{jj} > 0 \\ 0 & \text{else.} \end{cases}
$$

Also, by construction of $\boldsymbol{S}$, $\mathrm{rank}(\mathrm{diag}(\mathbf{m})) \leq nk$. Further, (b) follows by symmetry across the d dimensions.

Since $\delta k \leq \sum_{c=k}^{nk-1}\delta_c c \leq \delta(nk-1)$, there is

$$
\frac{d}{(1-\delta)nk + \delta(nk-1)} \leq \bar{\beta} \leq \frac{d}{(1-\delta)nk + \delta k} \tag{A.10}
$$

Computing the MSE. Next, we use the value of $\bar{\beta}$ in Eq. A.9 to compute MSE.

$$
\begin{aligned}
&MSE(\text{Rand-Proj-Spatial(Max)}) = \mathbb{E}[\|\widehat{\mathbf{x}}^{(\text{Rand-Proj-Spatial(Max)})} - \bar{\mathbf{x}}\|_2^2] = \mathbb{E}[\|\bar{\beta}\boldsymbol{S}^{\dagger}\boldsymbol{S}\mathbf{x} - \mathbf{x}\|_2^2] \\
&= \bar{\beta}^2\mathbb{E}[\|\boldsymbol{S}^{\dagger}\boldsymbol{S}\mathbf{x}\|_2^2] + \|\mathbf{x}\|_2^2 - 2\Big\langle\bar{\beta}\mathbb{E}[\boldsymbol{S}^{\dagger}\boldsymbol{S}\mathbf{x}], \mathbf{x}\Big\rangle \\
&= \bar{\beta}^2\mathbb{E}[\|\boldsymbol{S}^{\dagger}\boldsymbol{S}\mathbf{x}\|_2^2] - \|\mathbf{x}\|_2^2 \qquad \text{(Using unbiasedness of } \widehat{\mathbf{x}}^{(\text{Rand-Proj-Spatial(Max)})}\text{)} \\
&= \bar{\beta}^2\mathbf{x}^T\mathbb{E}[\boldsymbol{S}^T(\boldsymbol{S}^{\dagger})^T\boldsymbol{S}^{\dagger}\boldsymbol{S}]\mathbf{x} - \|\mathbf{x}\|_2^2.
\end{aligned} \tag{A.11}
$$

Using $\boldsymbol{S}^{\dagger} = \boldsymbol{U}\Lambda^{\dagger}\boldsymbol{U}^T$,

$$
\mathbb{E}[\boldsymbol{S}^T(\boldsymbol{S}^{\dagger})^T\boldsymbol{S}^{\dagger}\boldsymbol{S}] = \mathbb{E}[\boldsymbol{U}\Lambda\boldsymbol{U}^T\boldsymbol{U}\Lambda^{\dagger}\boldsymbol{U}^T\boldsymbol{U}\Lambda^{\dagger}\boldsymbol{U}^T\boldsymbol{U}\Lambda\boldsymbol{U}^T]
$$

$$
\begin{aligned}
&= \mathbb{E}[\boldsymbol{U}\Lambda(\Lambda^\dagger)^2\Lambda\boldsymbol{U}^T] \\
&= \sum_{\boldsymbol{U}=\Phi} \boldsymbol{U}\mathbb{E}[\Lambda(\Lambda^\dagger)^2\Lambda]\boldsymbol{U}^T \cdot \Pr[\boldsymbol{U}=\Phi] \\
&= \sum_{\boldsymbol{U}=\Phi} \boldsymbol{U}\Big[(1-\delta)\frac{nk}{d}\boldsymbol{I}_d + \sum_{c=k}^{nk-1}\delta_c\frac{c}{d}\boldsymbol{I}_d\Big]\boldsymbol{U}^T \cdot \Pr[\boldsymbol{U}=\Phi] \\
&= \Big[(1-\delta)\frac{nk}{d} + \sum_{c=k}^{nk-1}\delta_c\frac{c}{d}\Big] \cdot \sum_{\boldsymbol{U}=\Phi}\boldsymbol{U}\boldsymbol{U}^T \cdot \Pr[\boldsymbol{U}=\Phi] \\
&= \Big[(1-\delta)\frac{nk}{d} + \sum_{c=k}^{nk-1}\delta_c\frac{c}{d}\Big]\boldsymbol{I}_d \\
&= \frac{1}{\bar{\beta}}\boldsymbol{I}_d
\end{aligned} \tag{A.12}
$$

Substituting Eq. A.12 in Eq. A.11, we get

$$
\begin{aligned}
MSE(\text{Rand-Proj-Spatial(Max)}) &= \bar{\beta}^2\mathbf{x}^T\frac{1}{\bar{\beta}}\boldsymbol{I}_d\mathbf{x} - \|\mathbf{x}\|_2^2 = (\bar{\beta}-1)\|\mathbf{x}\|_2^2 \\
&\leq \Big[\frac{d}{(1-\delta)nk+\delta k} - 1\Big]\|\mathbf{x}\|_2^2,
\end{aligned}
$$

where the inequality is by Eq A.10. □

A.3.2 Comparing against Rand-k

Next, we compare the MSE of Rand-Proj-Spatial(Max) with the MSE of the baseline Rand-k analytically in the full-correlation case. Recall that in this case,

$$
MSE(\text{Rand-}k) = \frac{1}{n}(\frac{d}{k}-1)\|\mathbf{x}\|_2^2.
$$

We have

$$
\begin{aligned}
&MSE(\text{Rand-Proj-Spatial(Max)}) \leq MSE(\text{Rand-}k) \\
\Leftrightarrow\ &\frac{d}{(1-\delta)nk+\delta k} - 1 \leq \frac{1}{n}(\frac{d}{k}-1) \\
\Leftrightarrow\ &\frac{d}{k}\frac{n-(1-\delta)n-\delta}{n((1-\delta)n+\delta)} \leq 1 - \frac{1}{n}
\end{aligned}
$$

$$
\begin{aligned}
&\Leftrightarrow \frac{d}{k} \cdot \frac{\delta - \delta/n}{(1-\delta)n + \delta} \leq \frac{n-1}{n} \\
&\Leftrightarrow d\delta(1 - \frac{1}{n})n \leq k(n-1) \cdot ((1-\delta)n + \delta) \\
&\Leftrightarrow d\delta \leq k \cdot ((1-\delta)n + \delta) \\
&\Leftrightarrow d\delta + kn\delta - k\delta \leq kn \\
&\Leftrightarrow \delta \leq \frac{kn}{d + kn - k} \\
&\Leftrightarrow \delta \leq \frac{1}{\frac{d}{kn} + 1 - \frac{1}{n}}
\end{aligned}
$$

Since $nk \leq d$, for $n \geq 2$, the above implies when

$$\delta \leq \frac{1}{1 + \frac{1}{2}} = \frac{2}{3},$$

the MSE of Rand-Proj-Spatial(Max) is always less than that of Rand-k.

A.3.3 $\boldsymbol{S}$ has full rank with high probability

We empirically verify that $\delta \approx 0$. With $d \in \{32, 64, 128, \ldots, 1024\}$ and 4 different nk value such that $nk \leq d$ for each d, we compute rank($\boldsymbol{S}$) for 10^5 trials for each pair of (nk, d) values, and plot the results for all trials. All results are presented in Figure A.2. As one can observe from the plots, rank($\boldsymbol{S}$) $= nk$ with high probability, suggesting $\delta \approx 0$.

This implies the MSE of Rand-Proj-Spatial(Max) is

$$MSE(\text{Rand-Proj-Spatial(Max)}) \approx (\frac{d}{nk} - 1)\|\mathbf{x}\|_2^2,$$

in the full correlation case.

A.3.4 Proof of Theorem 2.4.4

Theorem 4.4 (MSE under No Correlation)**.** *Consider n clients, each holding a vector $\mathbf{x}_i \in \mathbb{R}^d$, $\forall i \in [n]$. Suppose we set $T \equiv 1$, $\bar{\beta} = \frac{d^2}{k}$ in Eq. 2.5, and the random linear*

map $\boldsymbol{G}_i$ *at each client to be an SRHT matrix. Then, for* $nk \leq d$*,*

$$\mathbb{E}\Big[\|\widehat{\mathbf{x}}^{(\textit{Rand-Proj-Spatial})} - \bar{\mathbf{x}}\|_2^2\Big] = \frac{1}{n^2}\Big(\frac{d}{k} - 1\Big)\sum_{i=1}^{n}\|\mathbf{x}_i\|_2^2.$$

Proof. When the client vectors are all orthogonal to each other, we define the transformation function on the eigenvalue to be $T(\lambda) = 1, \forall \lambda \geq 0$. We show that by considering the above constant T, SRHT becomes the same as rand k. Recall $\boldsymbol{S} = \sum_{i=1}^{n}\boldsymbol{G}_i^T\boldsymbol{G}_i$ and let $\boldsymbol{G}^T\boldsymbol{G} = \boldsymbol{U}\Lambda\boldsymbol{U}^T$ be its eigendecompostion. Then,

$$T(\boldsymbol{S}) = \boldsymbol{U}T(\Lambda)\boldsymbol{U}^T = \boldsymbol{U}\boldsymbol{I}_d\boldsymbol{U}^T = \boldsymbol{I}_d.$$

Hence, $(T(\boldsymbol{S}))^\dagger = \boldsymbol{I}_d$. And the decoded vector for client i becomes

$$\begin{aligned}
\widehat{\mathbf{x}}_i &= \bar{\beta}\Big(T(\boldsymbol{G}^T\boldsymbol{G})\Big)^\dagger\boldsymbol{G}_i^T\boldsymbol{G}_i\mathbf{x}_i = \bar{\beta}\boldsymbol{G}_i^T\boldsymbol{G}_i\mathbf{x}_i = \bar{\beta}\frac{1}{d}\boldsymbol{D}_i\boldsymbol{H}^T\boldsymbol{E}_i^T\boldsymbol{E}_i\boldsymbol{H}\boldsymbol{D}_i\mathbf{x}_i, \\
\widehat{\mathbf{x}} &= \frac{1}{n}\sum_{i=1}^{n}\widehat{\mathbf{x}}_i = \frac{1}{n}\bar{\beta}\sum_{i=1}^{n}\frac{1}{d}\boldsymbol{D}_i\boldsymbol{H}^T\boldsymbol{E}_i^T\boldsymbol{E}_i\boldsymbol{H}\boldsymbol{D}_i\mathbf{x}_i
\end{aligned} \tag{A.13}$$

$\boldsymbol{D}_i$ is a diagonal matrix. Also, $\boldsymbol{E}_i^T\boldsymbol{E}_i \in \mathbb{R}^{d\times d}$ is a diagonal matrix, where the i-th entry is 0 or 1.

Computing $\bar{\beta}$**.** To ensure that $\widehat{\mathbf{x}}$ is an unbiased estimator, from Eq. A.13

$$\begin{aligned}
\mathbf{x}_i &= \bar{\beta}\mathbb{E}[\boldsymbol{G}_i^T\boldsymbol{G}_i]\mathbf{x}_i \\
&= \frac{\bar{\beta}}{d}\mathbb{E}[\boldsymbol{D}_i\boldsymbol{H}^T\boldsymbol{E}_i^T\boldsymbol{E}_i\boldsymbol{H}\boldsymbol{D}_i]\mathbf{x}_i \\
&= \frac{\bar{\beta}}{d}\mathbb{E}_{\boldsymbol{D}_i}\Big[\boldsymbol{D}_i\boldsymbol{H}^T\underbrace{\mathbb{E}[\boldsymbol{E}_i^T\boldsymbol{E}_i]}_{=(k/d)\boldsymbol{I}_d}\boldsymbol{H}\boldsymbol{D}_i\Big]\mathbf{x}_i && (\because \boldsymbol{E}_i \text{ is independent of } \boldsymbol{D}_i) \\
&= \frac{\bar{\beta}}{d}k\mathbb{E}_{\boldsymbol{D}_i}\left[\boldsymbol{D}_i^2\right]\mathbf{x}_i && (\because \boldsymbol{H}^T\boldsymbol{H} = d\boldsymbol{I}_d) \\
&= \frac{\bar{\beta}k}{d}\mathbf{x}_i && (\because \boldsymbol{D}_i^2 = \boldsymbol{I} \text{ is now deterministic.}) \\
\Rightarrow \bar{\beta} &= \frac{d}{k}.
\end{aligned} \tag{A.14}$$

Computing the MSE.

$$
\begin{aligned}
MSE &= \mathbb{E}\left\|\widehat{\mathbf{x}} - \bar{\mathbf{x}}\right\|_2^2 \\
&= \mathbb{E}\left\|\frac{1}{n}\bar{\beta}\sum_{i=1}^{n}\frac{1}{d}\boldsymbol{D}_i\boldsymbol{H}^T\boldsymbol{E}_i^T\boldsymbol{E}_i\boldsymbol{H}\boldsymbol{D}_i\mathbf{x}_i - \frac{1}{n}\sum_{i=1}^{n}\mathbf{x}_i\right\|_2^2 \\
&= \frac{1}{n^2}\left\{\mathbb{E}\left\|\bar{\beta}\sum_{i=1}^{n}\frac{1}{d}\boldsymbol{D}_i\boldsymbol{H}^T\boldsymbol{E}_i^T\boldsymbol{E}_i\boldsymbol{H}\boldsymbol{D}_i\mathbf{x}_i\right\|_2^2 + \left\|\sum_{i=1}^{n}\mathbf{x}_i\right\|_2^2\right. \\
&\qquad\qquad \left. -2\left\langle\bar{\beta}\mathbb{E}[\sum_{i=1}^{n}\frac{1}{d}\boldsymbol{D}_i\boldsymbol{H}^T\boldsymbol{E}_i^T\boldsymbol{E}_i\boldsymbol{H}\boldsymbol{D}_i\mathbf{x}_i], \sum_{i=1}^{n}\mathbf{x}_i\right\rangle\right\} \\
&= \frac{1}{n^2}\left\{\bar{\beta}^2\mathbb{E}\left\|\sum_{i=1}^{n}\frac{1}{d}\boldsymbol{D}_i\boldsymbol{H}^T\boldsymbol{E}_i^T\boldsymbol{E}_i\boldsymbol{H}\boldsymbol{D}_i\mathbf{x}_i\right\|_2^2 - \left\|\sum_{i=1}^{n}\mathbf{x}_i\right\|_2^2\right\} \qquad (\because \mathbb{E}[\widehat{\mathbf{x}}] = \bar{\mathbf{x}}) \\
&= \frac{1}{n^2}\left\{\sum_{i=1}^{n}\frac{\bar{\beta}^2}{d^2}\mathbb{E}\left\|\boldsymbol{D}_i\boldsymbol{H}^T\boldsymbol{E}_i^T\boldsymbol{E}_i\boldsymbol{H}\boldsymbol{D}_i\mathbf{x}_i\right\|_2^2 - \sum_{i=1}^{n}\left\|\mathbf{x}_i\right\|_2^2 \right. \qquad (A.15) \\
&\quad \left. +2\sum_{i=1}^{n}\sum_{l=i+1}^{n}\frac{\bar{\beta}^2}{d^2}\left\langle\mathbb{E}[\boldsymbol{D}_i\boldsymbol{H}^T\boldsymbol{E}_i^T\boldsymbol{E}_i\boldsymbol{H}\boldsymbol{D}_i\mathbf{x}_i], \mathbb{E}[\boldsymbol{D}_l\boldsymbol{H}^T\boldsymbol{E}_l^T\boldsymbol{E}_l\boldsymbol{H}\boldsymbol{D}_l\mathbf{x}_l]\right\rangle - 2\sum_{i=1}^{n}\sum_{l=i+1}^{n}\left\langle\mathbf{x}_i, \mathbf{x}_l\right\rangle\right\}.
\end{aligned}
$$

Note that in Eq. A.15

$$
\begin{aligned}
&\mathbb{E}\left\|\boldsymbol{D}_i\boldsymbol{H}^T\boldsymbol{E}_i^T\boldsymbol{E}_i\boldsymbol{H}\boldsymbol{D}_i\mathbf{x}_i\right\|_2^2 = \mathbb{E}[\mathbf{x}_i^T\boldsymbol{D}_i\boldsymbol{H}^T\boldsymbol{E}_i^T\boldsymbol{E}_i\boldsymbol{H}\boldsymbol{D}_i\boldsymbol{D}_i\boldsymbol{H}^T\boldsymbol{E}_i^T\boldsymbol{E}_i\boldsymbol{H}\boldsymbol{D}_i\mathbf{x}_i] \\
&= d\mathbb{E}[\mathbf{x}_i^T\boldsymbol{D}_i\boldsymbol{H}^T(\boldsymbol{E}_i^T\boldsymbol{E}_i)^2\boldsymbol{H}\boldsymbol{D}_i\mathbf{x}_i] \qquad (\because \boldsymbol{D}_i^2 = \boldsymbol{I}_d; \boldsymbol{H}^T\boldsymbol{H} = \boldsymbol{H}\boldsymbol{H}^T = d\boldsymbol{I}_d) \\
&= d\mathbf{x}_i^T\mathbb{E}_{\boldsymbol{D}_i}\left[\boldsymbol{D}_i\boldsymbol{H}^T\mathbb{E}[\boldsymbol{E}_i^T\boldsymbol{E}_i]\boldsymbol{H}\boldsymbol{D}_i\right]\mathbf{x}_i \quad (\boldsymbol{E}_i, \boldsymbol{D}_i \text{ are independent; } (\boldsymbol{E}_i^T\boldsymbol{E}_i)^2 = \boldsymbol{E}_i^T\boldsymbol{E}_i) \\
&= kd\|\mathbf{x}_i\|_2^2, \qquad (A.16)
\end{aligned}
$$

since $\mathbb{E}[\boldsymbol{E}_i^T\boldsymbol{E}_i] = (k/d)\boldsymbol{I}_d$, $\boldsymbol{H}^T\boldsymbol{H} = d\boldsymbol{I}_d$ and for $i \neq l$

$$
\left\langle\mathbb{E}[\boldsymbol{D}_i\boldsymbol{H}^T\boldsymbol{E}_i^T\boldsymbol{E}_i\boldsymbol{H}\boldsymbol{D}_i\mathbf{x}_i], \mathbb{E}[\boldsymbol{D}_l\boldsymbol{H}^T\boldsymbol{E}_l^T\boldsymbol{E}_l\boldsymbol{H}\boldsymbol{D}_l\mathbf{x}_l]\right\rangle = \left\langle k\mathbf{x}_i, k\mathbf{x}_l\right\rangle = k^2\left\langle\mathbf{x}_i, \mathbf{x}_l\right\rangle. \qquad (A.17)
$$

Substituting Eq. A.16, A.17 in Eq. A.15, we get

$$
MSE = \frac{1}{n^2}\left\{\left(\frac{\bar{\beta}^2}{d^2}\sum_{i=1}^{n}kd\|\mathbf{x}_i\|_2^2 + 2\sum_{i=1}^{n}\sum_{l=i+1}^{n}\frac{\bar{\beta}^2k^2}{d^2}\left\langle\mathbf{x}_i, \mathbf{x}_l\right\rangle\right) - \sum_{i=1}^{n}\left\|\mathbf{x}_i\right\|_2^2 - 2\sum_{i=1}^{n}\sum_{l=i+1}^{n}\left\langle\mathbf{x}_i, \mathbf{x}_l\right\rangle\right\}
$$

$$= \frac{1}{n^2}\left(\frac{d}{k} - 1\right)\sum_{i=1}^{n} \|\mathbf{x}_i\|_2^2,$$

which is exactly the same as the MSE of rand k. □

A.3.5 Rand-Proj-Spatial recovers Rand-k-Spatial (Proof of Lemma 4.1)

Lemma 4.1 (Recovering Rand-k-Spatial)**.** *Suppose client i generates a subsampling matrix $\boldsymbol{E}_i = \left[\mathbf{e}_{i_1}, \ \ldots, \ \mathbf{e}_{i_k}\right]^\top$, where $\{\mathbf{e}_j\}_{j=1}^d$ are the canonical basis vectors, and $\{i_1, \ldots, i_k\}$ are sampled from $\{1, \ldots, d\}$ without replacement. The encoded vectors are given as $\widehat{\mathbf{x}}_i = \boldsymbol{E}_i\mathbf{x}_i$. Given a function T, $\widehat{\mathbf{x}}$ computed as in Eq. 2.5 recovers the Rand-k-Spatial estimator.*

Proof. If client i applies $\boldsymbol{E}_i \in \mathbb{R}^{k\times d}$ as the random matrix to encode $\mathbf{x}_i$ in Rand-Proj-Spatial, by Eq. 2.5, client i's encoded vector is now

$$\hat{\mathbf{x}}_i^{\text{(Rand-Proj-Spatial)}} = \bar{\beta}\left(T(\sum_{i=1}^{n} \boldsymbol{E}_i^T\boldsymbol{E}_i)\right)^\dagger \boldsymbol{E}_i^T\boldsymbol{E}_i\mathbf{x}_i \tag{A.18}$$

Notice $\boldsymbol{E}_i^T\boldsymbol{E}_i$ is a diagonal matrix, where the j-th diagonal entry is 1 if coordinate j of $\mathbf{x}_i$ is chosen. Hence, $\boldsymbol{E}_i^T\boldsymbol{E}_i\mathbf{x}_i$ can be viewed as choosing k coordinates of $\mathbf{x}_i$ without replacement, which is exactly the same as Rand-k-Spatial's (and Rand-k's) encoding procedure.

Notice $\sum_{i=1}^n \boldsymbol{E}_i^T\boldsymbol{E}_i$ is also a diagonal matrix, where the j-th diagonal entry is exactly M_j, i.e. the number of clients who selects the j-th coordinate as in Rand-k-Spatial [57]. Furthermore, notice $\left(T(\sum_{i=1}^n \boldsymbol{E}_i^T\boldsymbol{E}_i)\right)^\dagger$ is also a diagonal matrix, where the j-th diagonal entry is $\frac{1}{T(M_j)}$, which recovers the scaling factor used in Rand-k-Spatial's decoding procedure.

Rand-Proj-Spatial computes $\bar{\beta}$ as $\bar{\beta}\mathbb{E}\left[\left(T(\sum_{i=1}^n \boldsymbol{E}_i^T\boldsymbol{E}_i)\right)^\dagger \boldsymbol{E}_i^T\boldsymbol{E}_i\mathbf{x}_i\right] = \mathbf{x}_i$. Since $\left(T(\sum_{i=1}^n \boldsymbol{E}_i^T\boldsymbol{E}_i)\right)^\dagger$ and $\boldsymbol{E}_i^T\boldsymbol{E}_i\mathbf{x}_i$ recover the scaling factor and the encoding procedure of Rand-k-Spatial, and $\bar{\beta}$ is computed in exactly the same way as Rand-k-Spatial does, $\bar{\beta}$ will be exactly the same as in Rand-k-Spatial.

Therefore, $\hat{\mathbf{x}}_i^{(\text{Rand-Proj-Spatial})}$ in Eq. A.18 with $\boldsymbol{E}_i$ as the random matrix at client i recovers $\hat{\mathbf{x}}_i^{(\text{Rand-}k\text{-Spatial})}$. This implies Rand-Proj-Spatial recovers Rand-k-Spatial in this case. □

A.4 Additional Experiment Details and Results

Implementation. All experiments are conducted in a cluster of 20 machines, each of which has 40 cores. The implementation is in `Python`, mainly based on `numpy` and `scipy`. All code used for the experiments can be found at https://github.com/11hifish/Rand-Proj-Spatial.

Data Split. For the non-IID dataset split across the clients, we follow [**?**] to split `Fashion-MNIST`, which is used in distributed power iteration and distributed k-means. Specifically, the data is first sorted by labels and then divided into $2n$ shards with each shard corresponding to the data of a particular label. Each client is then assigned 2 shards (i.e., data from 2 classes). However, this approach only works for datasets with discrete labels (i.e. datasets used in classification tasks). For the other dataset `UJIndoor`, which is used in distributed linear regression, we first sort the dataset by the ground truth prediction and then divides the sorted dataset across the clients.

A.4.1 Additional experimental results

For each one of the three tasks, distributed power iteration, distributed k-means, and distributed linear regression, we provide additional results when the data split is IID across the clients for smaller n, k values in Section A.4.1, and when the data split is Non-IID across the clients in Section A.4.1. For the Non-IID case, we use the same settings (i.e. n, k, d values) as in the IID case.

Discussion. For smaller n, k values compared to the data dimension d, there is less information or less correlation from the client vectors. Hence, both Rand-k-Spatial and Rand-Proj-Spatial perform better as nk increases. When n, k is small, one might notice Rand-Proj-Spatial performs worse than Rand-k-Wangni in some settings. However, Rand-k-Wangni is an *adaptive* estimator, which optimizes the sampling weights for choosing the client vector coordinates through an iterative process. That

means Rand-k-Wangni requires more computation from the clients, while in practice, the clients often have limited computational power. In contrast, our Rand-Proj-Spatial estimator is *non-adaptive* and the server does more computation instead of the clients. This is more practical since the central server usually has more computational power than the clients in applications like FL. See the introduction for more discussion.

In most settings, we observe the proposed Rand-Proj-Spatial has a better performance compared to Rand-k-Spatial. Furthermore, as one would expect, both Rand-k-Spatial and Rand-Proj-Spatial perform better when the data split is IID across the clients since there is more correlation among the client vectors in the IID case than in the Non-IID case.

More results in the IID case

Distributed Power Iteration and Distribued K-Means. We use the `Fashion-MNIST` dataset for both distributed power iteration and distributed k-means, which has a dimension of $d = 1024$. We consider more settings for distributed power iteration and distributed k-means here: $n = 10, k \in \{5, 25, 51\}$, and $n = 50, k \in \{5, 10\}$. The results are presented in Figure A.3 and A.4.

Distributed Linear Regression. We use the `UJIndoor` dataset distributed linear regression, which has a dimension of $d = 512$. We consider more settings here: $n = 10, k \in \{5, 25\}$ and $n = 50, k \in \{1, 5\}$. The results are presented in Figure A.5.

Additional results in the Non-IID case

In this section, we report results when the dataset split across the clients are Non-IID, using the same datasets as in the IID case. We choose exactly the same set of n, k values as in the IID case.

Distributed Power Iteration and Distributed K-Means. Again, both distributed power iteration and distributed k-means use the `Fashion-MNIST` dataset, with a dimension $d = 1024$. We consider the following settings for both tasks: $n = 10, k \in \{5, 25, 51, 102\}$ and $n = 50, k \in \{5, 10, 20\}$. The results are presented in Figure A.6 and A.7.

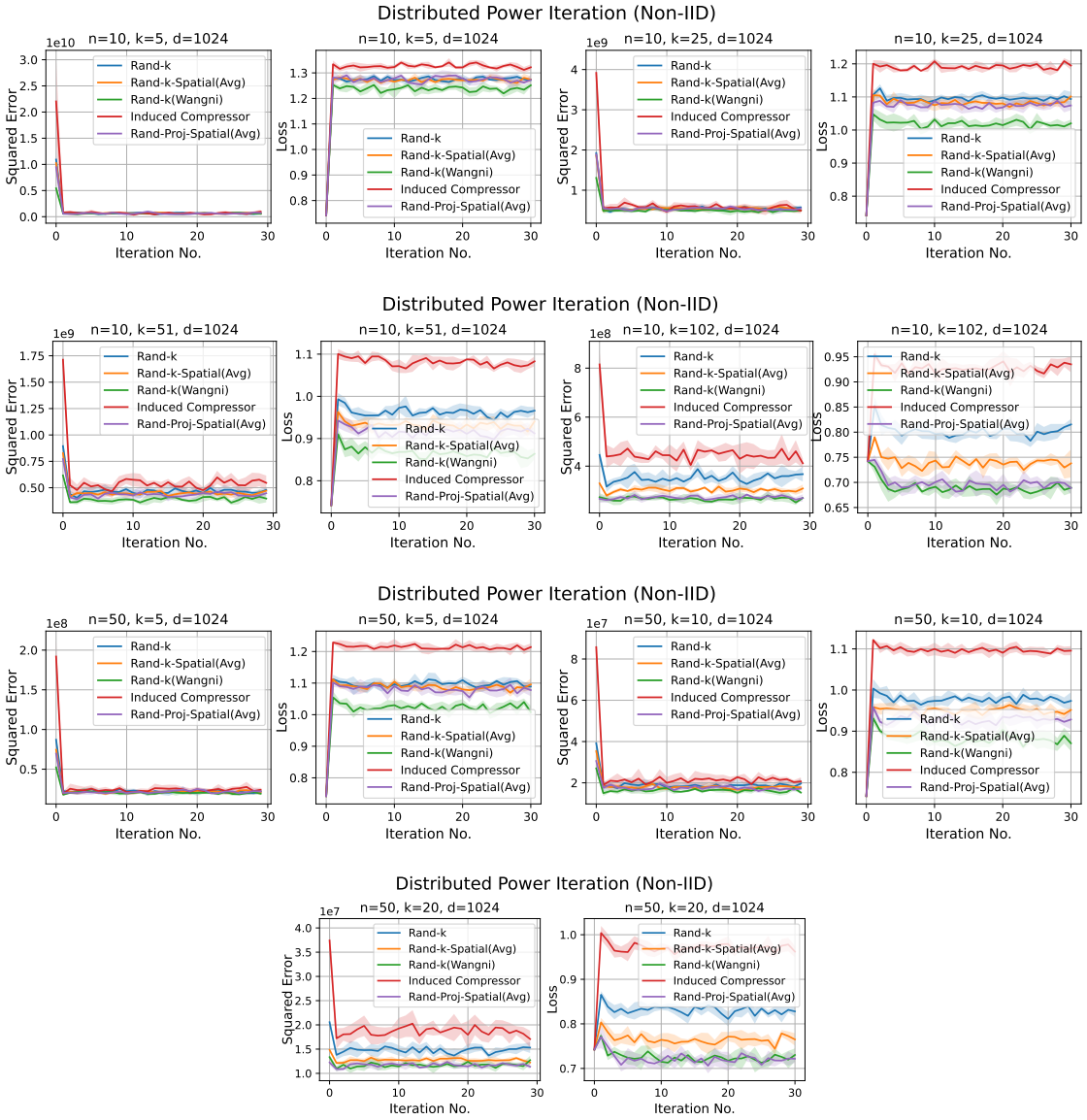

Figure A.6: Results of distributed power iteration when the data split is Non-IID. $n = 10, k \in \{5, 25, 51, 102\}$ and $n = 50, k \in \{5, 10, 20\}$.

Distributed Linear Regression. Again, we use the `UJIndoor` dataset for distributed linear regression, which has a dimension $d = 512$. We consider the following settings: $n = 10, k \in \{5, 25, 50\}$ and $n = 50, k \in \{1, 5, 50\}$. The results are presented in Figure A.8.

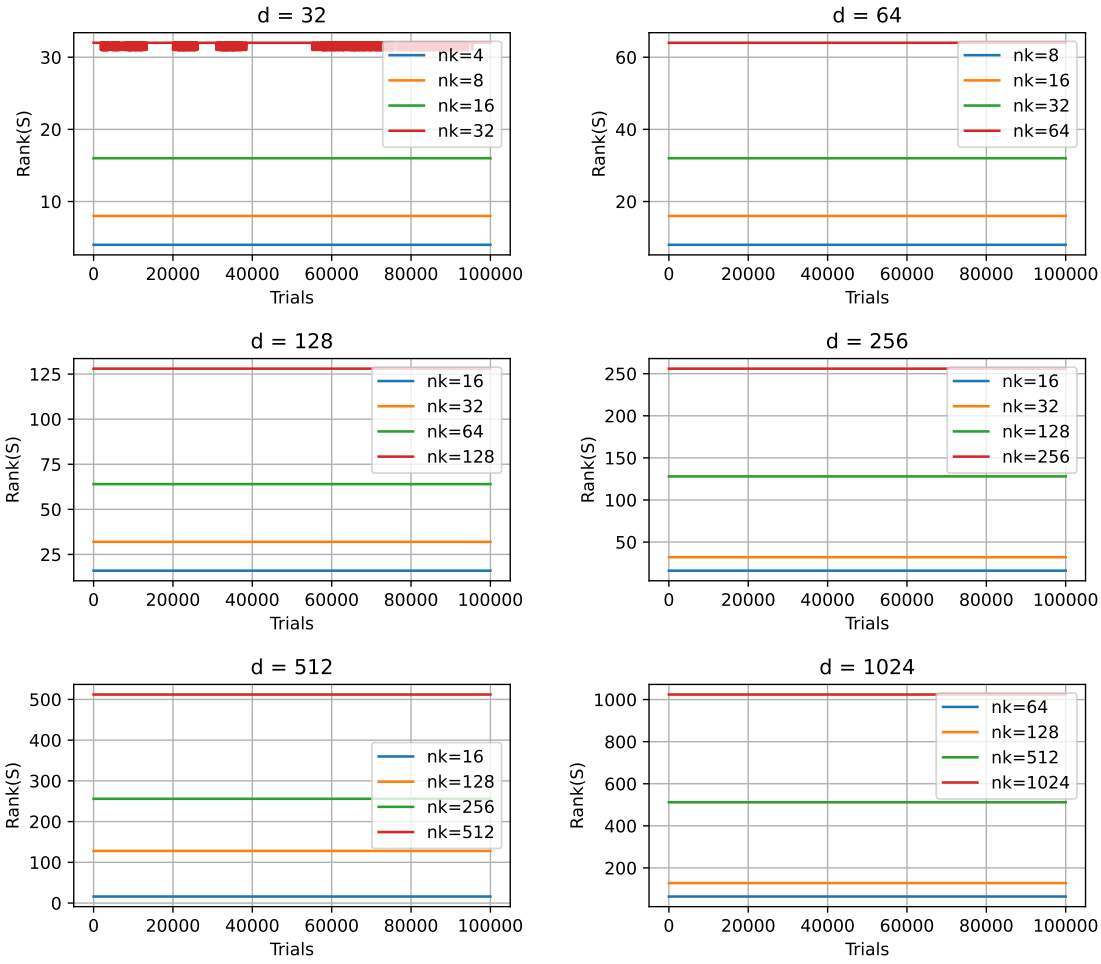

Figure A.2: Simulation results of rank($\boldsymbol{S}$), where $\boldsymbol{S} = \sum_{i=1}^{n} \boldsymbol{G}_i^T \boldsymbol{G}_i$, with $\boldsymbol{G}_i$ being SRHT. With $d \in \{32, 64, 128, \ldots, 1024\}$ and 4 different nk values such that $nk \leq d$ for each d, we compute rank($\mathbf{S}$) for 10^5 trials for each pairs of (nk, d) values and plot the results for all trials. When $d = 32$ and $nk = 32$ in the first plot, rank($\boldsymbol{S}$) $= 31$ in 2100 trials, and rank($\boldsymbol{S}$) $= nk = 32$ in all the rest of the trials. For all other (nk, d) pairs, $\boldsymbol{S}$ always has rank nk in the 10^5 trials. This verifies that $\delta = \Pr[\text{rank}(\boldsymbol{S}) < nk] \approx 0$.

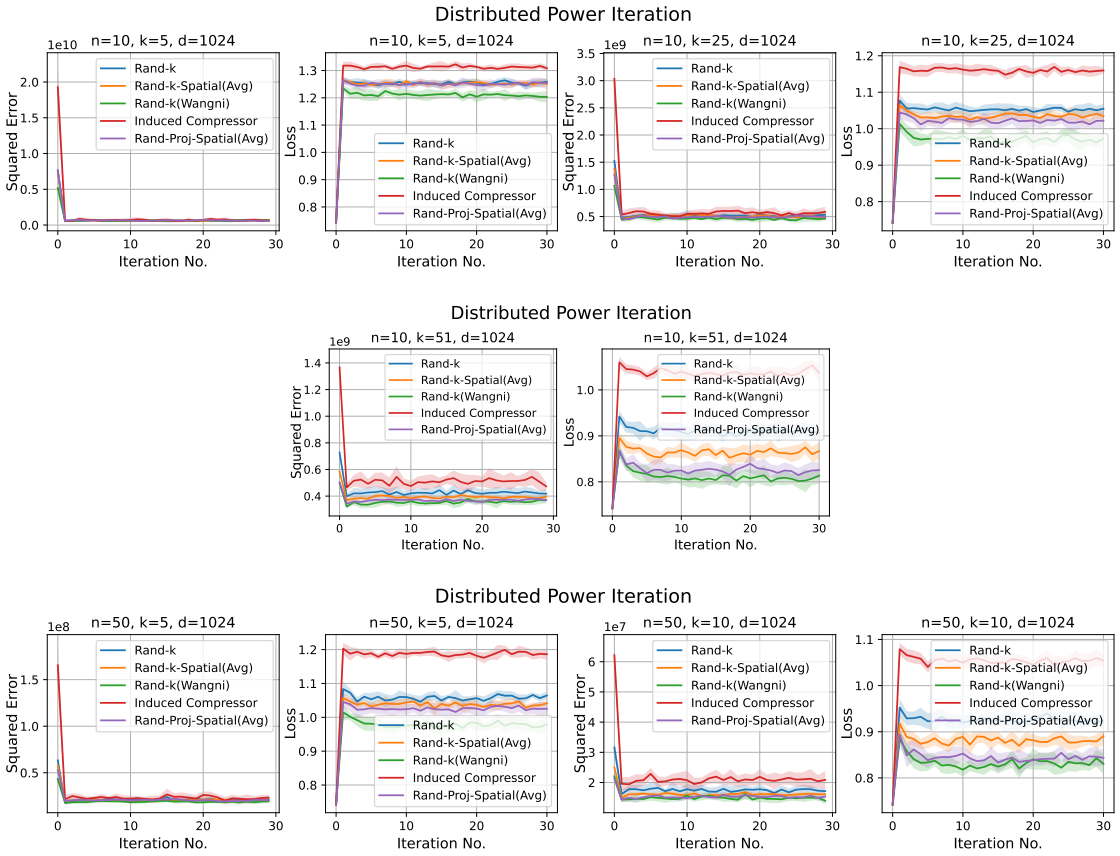

Figure A.3: More results of distributed power iteration on `Fashion-MNIST` (IID data split) with $d = 1024$ when $n = 10$, $k \in \{5, 25, 51\}$ and when $n = 50$, $k \in \{5, 10\}$.

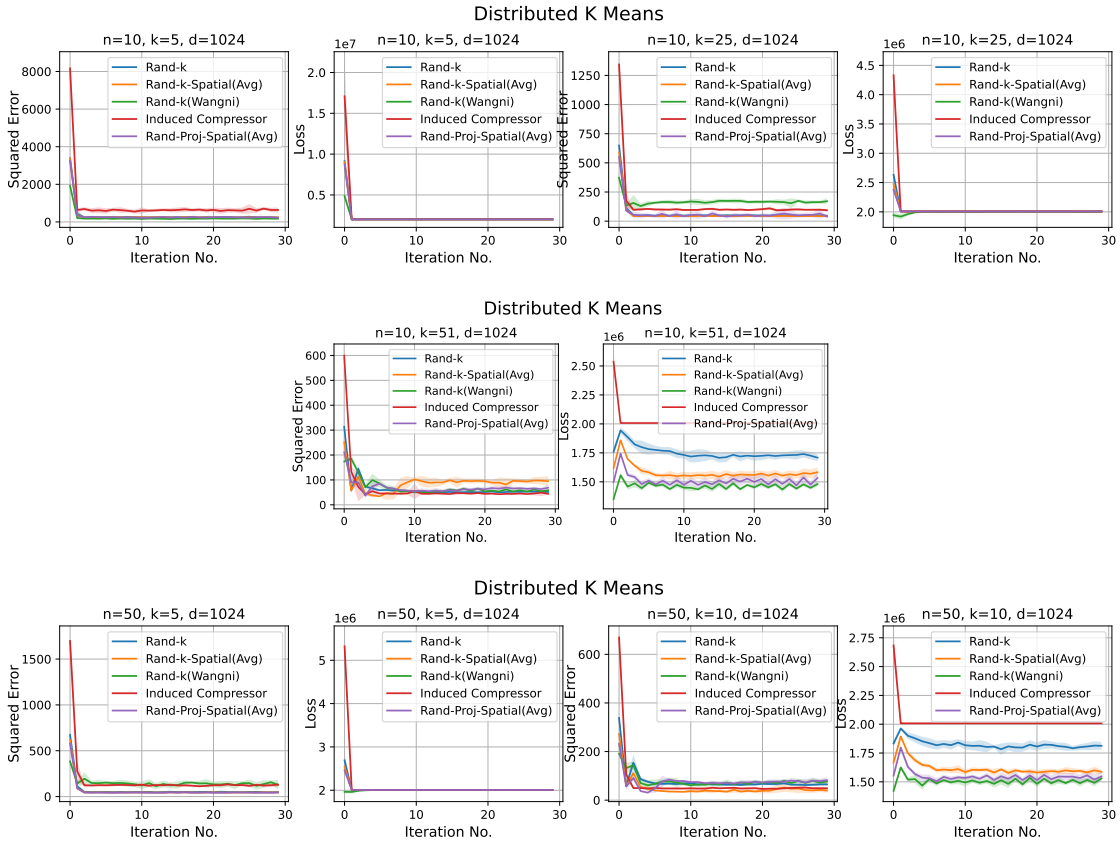

Figure A.4: More results on distributed k-means on `Fashion-MNIST` (IID data split) with $d = 1024$ when $n = 10, k \in \{5, 25, 51\}$ and when $n = 50, k \in \{10, 51\}$.

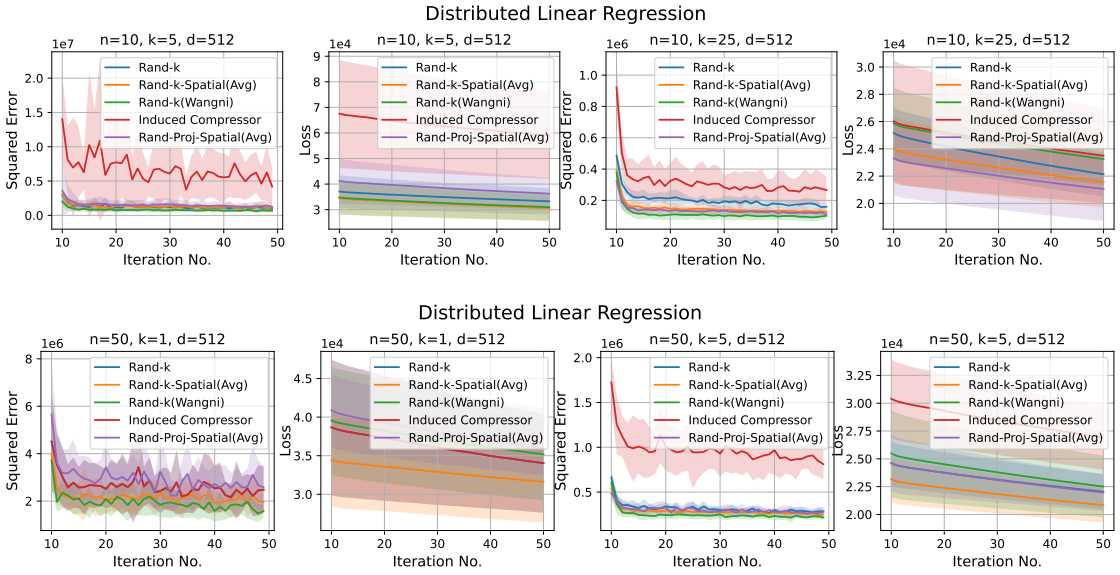

Figure A.5: More results of distributed linear regression on `UJIndoor` (IID data split) with $d = 512$, when $n = 10, k \in \{5, 25\}$ and when $n = 50, k \in \{1, 5\}$. Note when $k = 1$, the Induced estimator is the same as Rand-k.

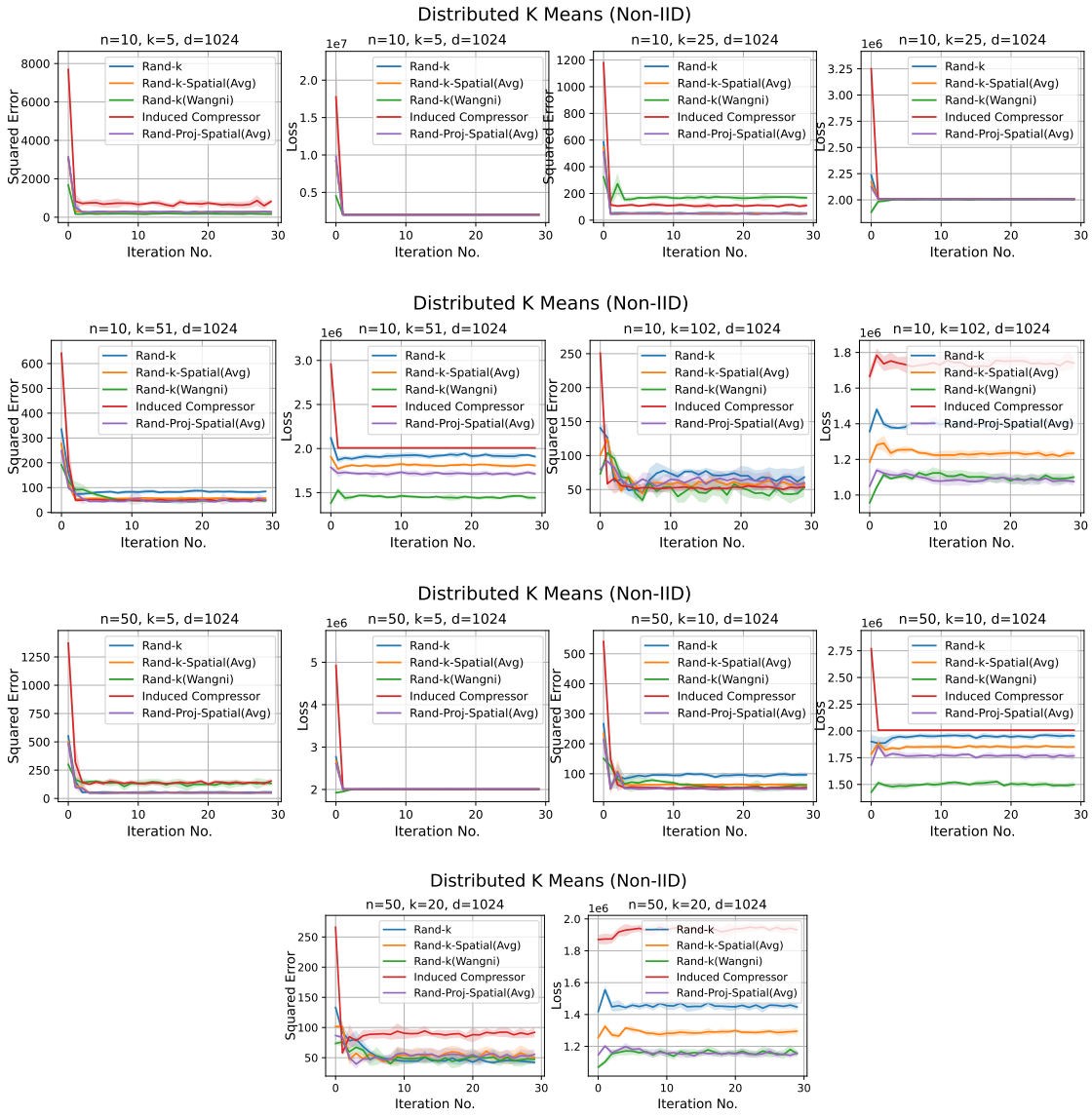

Figure A.7: Results of distributed k-means when the data split is Non-IID. $n = 10, k \in \{5, 25, 51, 102\}$ and $n = 50, k \in \{5, 10, 20\}$.

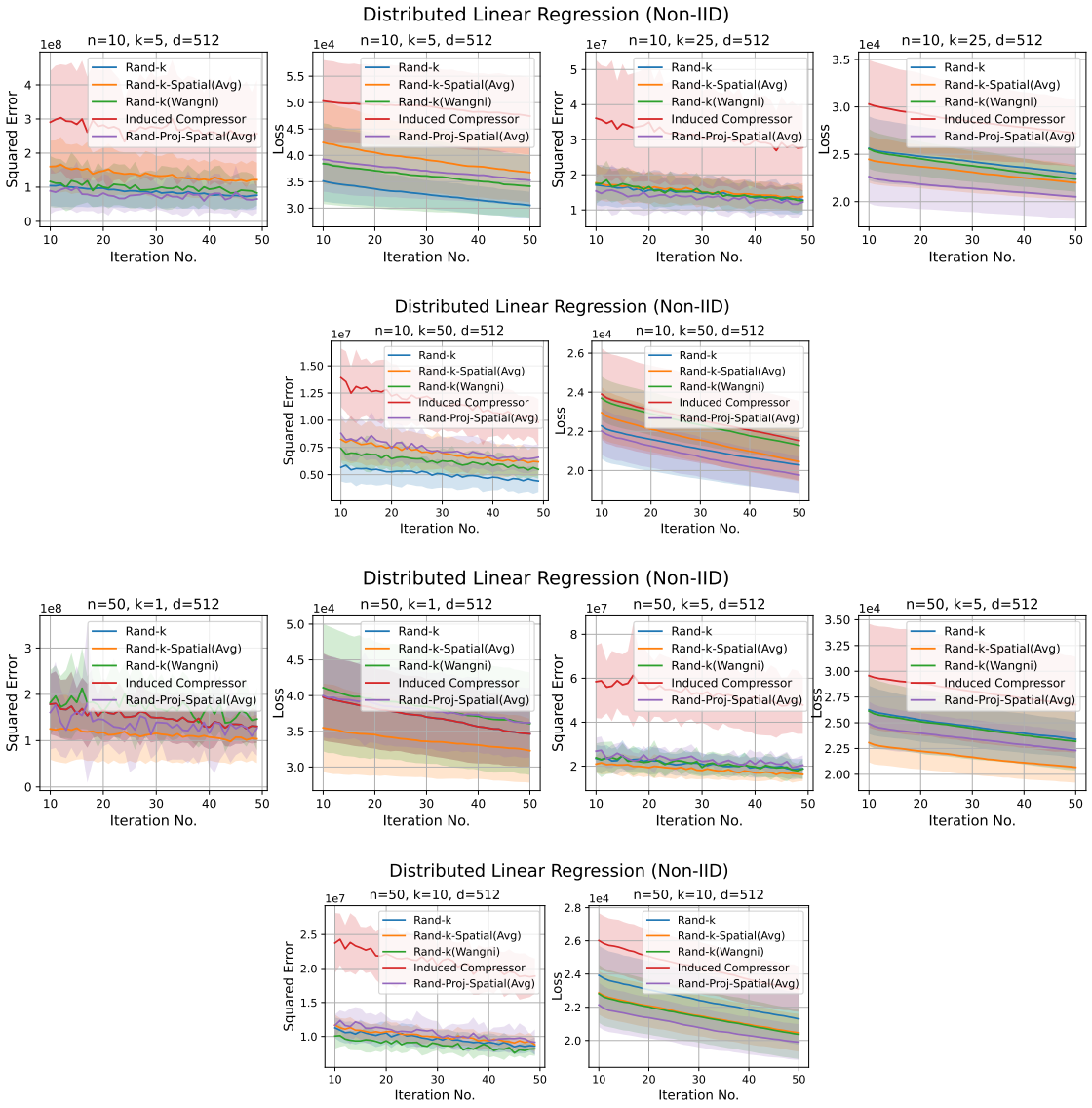

Figure A.8: Results of distributed linear regression when the data split is Non-IID. $n = 10, k \in \{5, 25, 50\}$ and $n = 50, k \in \{1, 5, 50\}$.

Appendix B

Private Majority Ensembling

B.1 Details of Section 3.4

B.1.1 Randomized Response with Constant Probability p_{const}

We show the magnitude of p_{const} in RR (Algorithm 7) to solve Problem 3.1.1, such that the output is $(m\epsilon, \delta)$-DP, in Lemma B.1.1.

Lemma B.1.1. *Consider using RR (Algorithm 7) to solve Problem 3.1.1. Let the majority of K (ϵ, Δ)-differentially private mechanisms be $(\tau\epsilon, \lambda)$-differentially private, where $\tau \in [1, K]$ and $\lambda \in [0, 1)$ are computed by simple composition (Theorem 3.3.2) or general composition (Theorem 3.3.3). If*

$$p_{const} \leq \frac{e^{m\epsilon} - 1 + 2\delta}{\frac{2(e^{\tau\epsilon} - e^{m\epsilon} + (1+e^{m\epsilon})\lambda)}{e^{\tau\epsilon}+1}} + e^{m\epsilon} - 1$$

then RR is $(m\epsilon, \delta)$-differentially private.

Proof of Lemma B.1.1. Let $x \in \{0, 1\}$ denote the output of RR. Let $q_x = \Pr[\mathcal{L}(\mathcal{D}) = x]$ and $q'_x = \Pr[\mathcal{L}(\mathcal{D}') = x]$, where $\mathcal{L}(\mathcal{D}) = \sum_{i=1}^{K} M_i(\mathcal{D})$, $\mathcal{L}(\mathcal{D}') = \sum_{i=1}^{K} M_i(\mathcal{D}')$ and $\mathcal{D}$, $\mathcal{D}'$ are adjacent datasets. Recall each mechanism M_i is (ϵ, Δ)-differentially private, and the majority of the outputs of $\{M_i\}_{i=1}^{K}$ is $(\tau\epsilon, \lambda)$-differentially private. When $\Delta = 0$, using simple composition, $\tau = K$ and $\lambda = 0$. When $\Delta > 0$, using general composition $\tau \approx \sqrt{K}$ and $\lambda \approx K\Delta$. By definition of differential privacy

Algorithm 7 Randomized Response Majority (RR)

Input: K (ϵ, Δ)-DP mechanisms $\{M_i\}_{i=1}^K$, noise function $\gamma : \{0, \dots, K\} \to [0, 1]$, dataset $\mathcal{D}$, privacy allowance $1 \leq m \leq K$, failure probability $\delta \geq \Delta \geq 0$
Output: $(m\epsilon, \delta)$-DP majority vote of $\{M_i\}_{i=1}^K$
Compute a *constant* probability $p_{const} \in [0, 1]$
Flip the p_{const}- biased coin
if Head (with probability p_{const}) **then**
    $\mathcal{S} = \{S_1, .., S_k\}$, where $S_i \sim M_i(\mathcal{D})$
    $\mathcal{L} = \sum_{i=1}^K S_i$
    Output $\mathbb{I}\{\frac{1}{K}\mathcal{L} \geq \frac{1}{2}\}$
else
    Output 0/1 with equal probability
end if

(Definition 3.3.1), all of the following four constraints on q_x, q'_x apply:

$$q_x \leq e^{\tau\epsilon} q'_x + \lambda, \quad \text{and} \quad 1 - q'_x \leq e^{\tau\epsilon}(1 - q_x) + \lambda$$
$$q'_x \leq e^{\tau\epsilon} q_x + \lambda, \quad \text{and} \quad 1 - q_x \leq e^{\tau\epsilon}(1 - q'_x) + \lambda$$

To ensure RR is $(m\epsilon, \delta)$-differentially private, p_{const} needs to be such that for all possible $q_x, q'_x \in [0, 1]$,

$$\begin{aligned}
&\Pr[\mathsf{RR}(\mathcal{D}) = x] \leq e^{m\epsilon} \Pr[\mathsf{RR}(\mathcal{D}') = x] + \delta \\
&p_{const} \cdot q_x + \frac{1}{2}(1 - p_{const}) \leq e^{m\epsilon}(p_{const} \cdot q'_x + \frac{1}{2}(1 - p_{const})) + \delta \\
&(q_x - e^{m\epsilon} q'_x + \frac{1}{2}e^{m\epsilon} - \frac{1}{2}) \cdot p_{const} \leq \frac{1}{2}e^{m\epsilon} - \frac{1}{2} + \delta
\end{aligned} \tag{B.1}$$

Let $h(q_x, q'_x) := q_x - e^{m\epsilon} q'_x + \frac{1}{2}e^{m\epsilon} - \frac{1}{2}$. The above inequality of p_{const} (Eq. B.1) needs to hold for worst case output probabilities q^*_x, q'^*_x that cause the maximum privacy loss. That is, p_{const} needs to satisfy

$$p_{const} \cdot max_{q_x, q'_x} h(q_x, q'_x) \leq \frac{1}{2}e^{m\epsilon} - \frac{1}{2} + \delta \tag{B.2}$$

To find the worst case output probabilities, we solve the following Linear Pro-

gramming (LP) problem:

$$\begin{aligned} &\text{Objective:} && \max_{q_x, q'_x} \quad h(q_x, q'_x) := q_x - e^{m\epsilon} q'_x + \frac{1}{2} e^{m\epsilon} - \frac{1}{2} \\ &\text{Subject to:} && 0 \le q_x \le 1, 0 \le q'_x \le 1 \\ & && q_x \le e^{\tau\epsilon} q'_x + \lambda, 1 - q'_x \le e^{\tau\epsilon}(1 - q_x) + \lambda \\ & && q'_x \le e^{\tau\epsilon} q_x + \lambda, 1 - q_x \le e^{\tau\epsilon}(1 - q'_x) + \lambda \end{aligned} \tag{B.3}$$

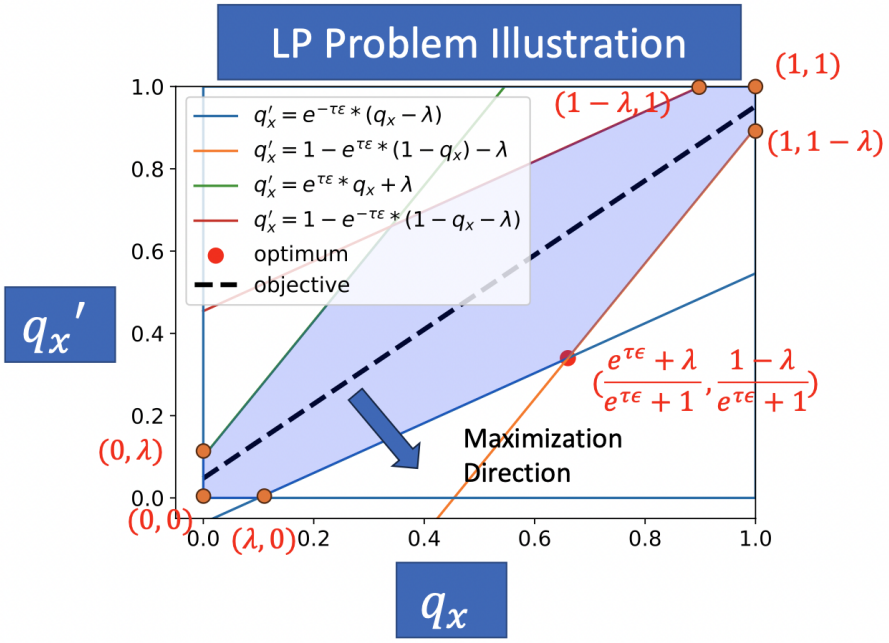

Figure B.1: A visualization of the above LP problem.

The optimum of any LP problem is at the corners of the feasible region, which is bounded by the optimization constraints. We plot the feasible region $\mathcal{F}$ and the objective of the above LP problem in Figure B.1. Here, $(q_x^*, q_x'^*) = \text{argmax}_{q_x, q'_x} h(q_x, q'_x) \in \{(0,0), (1,1), (0,\lambda), (\lambda,0), (1-\lambda,1), (1,1-\lambda), (\frac{1-\lambda}{e^{\tau\epsilon}+1}, \frac{e^{\tau\epsilon}+\lambda}{e^{\tau\epsilon}+1}), (\frac{e^{\tau\epsilon}+\lambda}{e^{\tau\epsilon}+1}, \frac{1-\lambda}{e^{\tau\epsilon}+1})\}$. The optimum of the LP problem – that is, the worse case probabilities $q_x^*, q_x'^*$ – is,

$$q_x^* = \frac{e^{\tau\epsilon} + \lambda}{e^{\tau\epsilon} + 1}, \quad q_x'^* = \frac{1 - \lambda}{e^{\tau\epsilon} + 1}$$

Algorithm 8 Subsampling Majority (SubMaj)

1: Input: K (ϵ, Δ)-DP mechanisms $\{M_i\}_{i=1}^K$, noise function $\gamma : \{0, \dots, K\} \to [0, 1]$, dataset $\mathcal{D}$, privacy allowance $1 \le m \le K$, failure probability $\delta \ge \Delta \ge 0$
2: Output: $(m\epsilon, \delta)$-DP majority vote of $\{M_i\}_{i=1}^K$
3: $\mathcal{S} = \{S_1, .., S_k\}$, where $S_i \sim M_i(\mathcal{D})$
4: $\mathcal{J}_m \leftarrow m$ indices chosen uniformly at random from $[K]$ without replacement
5: $\widehat{\mathcal{L}} = \sum_{j \in \mathcal{J}} S_j$
6: Output $\mathbb{I}\{\frac{1}{m}\widehat{\mathcal{L}} \ge \frac{1}{2}\}$

By Eq. B.2,

$$p_{const} \cdot \left(\frac{e^{\tau\epsilon} + \lambda}{e^{\tau\epsilon} + 1} - e^{m\epsilon} \frac{1 - \lambda}{e^{\tau\epsilon} + 1} + \frac{1}{2} e^{m\epsilon} - \frac{1}{2} \right) \le \frac{1}{2}(e^{m\epsilon} - 1) + \delta$$
$$p_{const} \cdot \left(\frac{e^{\tau\epsilon} - e^{m\epsilon} + (1 + e^{m\epsilon})\lambda}{e^{\tau\epsilon} + 1} + \frac{1}{2}(e^{m\epsilon} - 1) \right) \le \frac{1}{2}(e^{m\epsilon} - 1) + \delta$$
$$p_{const} \le \frac{e^{m\epsilon} - 1 + 2\delta}{\frac{2(e^{\tau\epsilon} - e^{m\epsilon} + (1 + e^{m\epsilon})\lambda)}{e^{\tau\epsilon} + 1} + e^{m\epsilon} - 1}$$

For small m, ϵ, K, using the approximation $e^y \approx 1 + y$ and that $\tau\epsilon < 2$,

$$p_{const} \approx \frac{m\epsilon + 2\delta}{\frac{2(\tau\epsilon - m\epsilon + (2 + m\epsilon)\lambda)}{\tau\epsilon + 2} + m\epsilon} \approx \frac{m\epsilon + 2\delta}{\tau\epsilon + (2 + m\epsilon)\lambda}$$

In the pure differential privacy setting, $\delta = 0, \lambda = 0, \tau = K$, and so $p_{const} \approx \frac{m}{K}$; and in the approximate differential privacy setting, $\lambda \approx 0, \delta \approx 0, \tau \approx \sqrt{K}$, and so $p_{const} \approx \frac{m}{\sqrt{K}}$. □

B.1.2 Proof of Lemma 3.4.1

Lemma B.1.2 (Restatement of Lemma 3.4.1)**.** *Consider Problem 3.1.1, with the privacy allowance $m \in [K]$. Consider the data-dependent algorithm that computes $\mathcal{L}(\mathcal{D})$ and then applies RR with probability p_γ. If $p_\gamma = \gamma_{Sub}(l)$, where $l \in \{0, 1, \dots, K\}$ is the value of $\mathcal{L}(\mathcal{D})$, i.e., the (random) sum of observed outcomes on dataset $\mathcal{D}$, and $\gamma_{Sub} : \{0, 1, \dots, K\} \to [0, 1]$ is*

$$\gamma_{Sub}(l) = \gamma_{Sub}(K - l)$$

$$= \begin{cases} 1 - 2\sum_{j=\frac{m+1}{2}}^{m} \frac{\binom{l}{j}\binom{K-l}{m-j}}{\binom{K}{m}} & \text{if } m \text{ is odd} \\ 1 - 2\sum_{j=\frac{m}{2}+1}^{m} \frac{\binom{l}{j}\binom{K-l}{m-j}}{\binom{K}{m}} - \frac{\binom{l}{\frac{m}{2}}\binom{K-l}{\frac{m}{2}}}{\binom{K}{m}} & \text{if } m \text{ is even} \end{cases}$$

then the majority of m out of K subsampled mechanisms without replacement and the output of our data-dependent RR algorithm have the same distribution.

Proof of Lemma 3.4.1. Let $\mathcal{L} = \sum_{i=1}^{K} S_i$ be the sum of observed outcomes from K mechanisms. Following Algorithm 8, $\mathcal{J}_m$ denotes the m indices chosen uniformly at random from $[K]$ without replacement. Conditioned on $\mathcal{L}$, notice the output of SubMaj follows a hypergeometric distribution. The output probability of SubMaj is

$$\begin{aligned} \Pr[\mathsf{SubMaj}(\mathcal{D}) = 1] &= \sum_{l=0}^{K} \Pr[\mathsf{SubMaj}(\mathcal{D}) = 1 \mid \mathcal{L} = l] \cdot \Pr[\mathcal{L} = l] \\ &= \sum_{l=0}^{K} \Pr[\sum_{j \in \mathcal{J}_m} S_j \geq \frac{m}{2} \mid \mathcal{L} = l] \cdot \Pr[\mathcal{L} = l] \qquad \text{(B.4)} \\ &= \begin{cases} \sum_{l=0}^{K} (\sum_{j=\frac{m+1}{2}}^{m} \frac{\binom{l}{j}\binom{K-l}{m-j}}{\binom{K}{m}}) \cdot \Pr[\mathcal{L} = l] & \text{if } m \text{ is odd} \\ \sum_{l=0}^{K} (\sum_{j=\frac{m}{2}+1}^{m} \frac{\binom{l}{j}\binom{K-l}{m-j}}{\binom{K}{m}} + \frac{1}{2}\frac{\binom{l}{\frac{m}{2}}\binom{K-l}{\frac{m}{2}}}{\binom{K}{m}}) \cdot \Pr[\mathcal{L} = l] & \text{if } m \text{ is even} \end{cases} \end{aligned}$$

Consider an arbitrary noise function $\gamma_{Sub} : \{0, 1, \ldots, K\} \to [0, 1]$. Let $\mathsf{RR\text{-}d}(\mathcal{D})$ denote the output of the data-dependent RR-d on dataset $\mathcal{D}$, where RR-d has the *non-constant* probability set by γ_{Sub}. The output probability of RR is,

$$\begin{aligned} \Pr[\mathsf{RR\text{-}d}(\mathcal{D}) = 1] &= \sum_{l=0}^{K} \Pr[\mathsf{RR\text{-}d}(\mathcal{D}) = 1 \mid \mathcal{L} = l] \cdot \Pr[\mathcal{L} = l] \\ &= \sum_{l=0}^{K} (\gamma_{Sub}(l) \cdot \mathbb{I}\{l \geq \frac{K+1}{2}\} + \frac{1}{2}(1 - \gamma_{Sub}(l))) \cdot \Pr[\mathcal{L} = l] \qquad \text{(B.5)} \end{aligned}$$

We want $\Pr[\mathsf{RR\text{-}d}(\mathcal{D}) = 1] = \Pr[\mathsf{Submaj}(\mathcal{D}) = 1]$.

If m is odd, for any $l \leq \frac{K-1}{2}$, this is

$$\begin{aligned} \frac{1}{2}(1-\gamma_{Sub}(l)) &= \sum_{j=\frac{m+1}{2}}^{m} \frac{\binom{l}{j}\binom{K-l}{m-j}}{\binom{K}{m}} \\ \Rightarrow \gamma_{Sub}(l) &= 1 - 2\sum_{j=\frac{m+1}{2}}^{m} \frac{\binom{l}{j}\binom{K-l}{m-j}}{\binom{K}{m}} \end{aligned} \tag{B.6}$$

and for any $l \geq \frac{K+1}{2}$, this is

$$\begin{aligned} \frac{1}{2} + \frac{1}{2}\gamma_{Sub}(l) &= \sum_{j=\frac{m+1}{2}}^{m} \frac{\binom{l}{j}\binom{K-l}{m-j}}{\binom{K}{m}} \\ \Rightarrow \gamma_{Sub}(l) &= 2\sum_{j=\frac{m+1}{2}}^{m} \frac{\binom{l}{j}\binom{K-l}{m-j}}{\binom{K}{m}} - 1 \end{aligned} \tag{B.7}$$

Similarly, if m is even, for any $l \leq \frac{K-1}{2}$, this is

$$\begin{aligned} \frac{1}{2}(1-\gamma_{Sub}(l)) &= \sum_{j=\frac{m}{2}+1}^{m} \frac{\binom{l}{j}\binom{K-l}{m-j}}{\binom{K}{m}} + \frac{1}{2}\frac{\binom{l}{\frac{m}{2}}\binom{K-l}{\frac{m}{2}}}{\binom{K}{m}} \\ \Rightarrow \gamma_{Sub}(l) &= 1 - 2\sum_{j=\frac{m}{2}+1}^{m} \frac{\binom{l}{j}\binom{K-l}{m-j}}{\binom{K}{m}} - \frac{\binom{l}{\frac{m}{2}}\binom{K-l}{\frac{m}{2}}}{\binom{K}{m}} \end{aligned} \tag{B.8}$$

and for any $l \geq \frac{K+1}{2}$, this is

$$\begin{aligned} \frac{1}{2} + \frac{1}{2}\gamma_{Sub}(l) &= \sum_{j=\frac{m}{2}+1}^{m} \frac{\binom{l}{j}\binom{K-l}{m-j}}{\binom{K}{m}} + \frac{1}{2}\frac{\binom{l}{\frac{m}{2}}\binom{K-l}{\frac{m}{2}}}{\binom{K}{m}} \\ \Rightarrow \gamma_{Sub}(l) &= 2\sum_{j=\frac{m}{2}+1}^{m} \frac{\binom{l}{j}\binom{K-l}{m-j}}{\binom{K}{m}} + \frac{\binom{l}{\frac{m}{2}}\binom{K-l}{\frac{m}{2}}}{\binom{K}{m}} - 1 \end{aligned} \tag{B.9}$$

Next, we show the above γ_{Sub} is indeed symmetric around $\frac{K}{2}$. For any $l \leq \frac{K-1}{2}$,

there is $K-l \geq \frac{K+1}{2}$. If m is odd,

$$\begin{aligned}\gamma_{Sub}(K-l) &= 2\sum_{j=\frac{m+1}{2}}^{m}\frac{\binom{K-l}{j}\binom{l}{m-j}}{\binom{K}{m}} - 1 = 2\Big(1-\sum_{j=1}^{\frac{m-1}{2}}\frac{\binom{K-l}{j}\binom{l}{m-j}}{\binom{K}{m}}\Big) - 1 \\ &= 1 - 2\sum_{j=1}^{\frac{m-1}{2}}\frac{\binom{K-l}{j}\binom{l}{m-j}}{\binom{K}{m}} = 1 - 2\sum_{j=\frac{m+1}{2}}^{m}\frac{\binom{l}{j}\binom{K-l}{m-j}}{\binom{K}{m}} \\ &= \gamma_{Sub}(l) \end{aligned} \tag{B.10}$$

Similarly, if m is even,

$$\begin{aligned}\gamma_{Sub}(K-l) &= 2\sum_{j=\frac{m}{2}+1}^{m}\frac{\binom{K-l}{j}\binom{l}{m-j}}{\binom{K}{m}} + \frac{\binom{l}{\frac{m}{2}}\binom{K-l}{\frac{m}{2}}}{\binom{K}{m}} - 1 \\ &= 2\Big(1-\sum_{j=1}^{\frac{m}{2}-1}\frac{\binom{K-l}{j}\binom{l}{m-j}}{\binom{K}{m}} - \frac{1}{2}\frac{\binom{l}{\frac{m}{2}}\binom{K-l}{\frac{m}{2}}}{\binom{K}{m}}\Big) - 1 \\ &= 1 - 2\sum_{j=1}^{\frac{m}{2}-1}\frac{\binom{K-l}{j}\binom{l}{m-j}}{\binom{K}{m}} - \frac{\binom{l}{\frac{m}{2}}\binom{K-l}{\frac{m}{2}}}{\binom{K}{m}} = 1 - 2\sum_{j=\frac{m}{2}+1}^{m}\frac{\binom{l}{j}\binom{K-l}{m-j}}{\binom{K}{m}} - \frac{\binom{l}{\frac{m}{2}}\binom{K-l}{\frac{m}{2}}}{\binom{K}{m}} \\ &= \gamma_{Sub}(l) \end{aligned} \tag{B.11}$$

Now, combining Eq. B.6, Eq. B.7 and Eq. B.10, if m is odd, setting γ_{Sub} as

$$\gamma_{Sub}(l) = \gamma_{Sub}(K-l) = 1 - 2\sum_{j=\frac{m+1}{2}}^{m}\frac{\binom{l}{j}\binom{K-l}{m-j}}{\binom{K}{m}}$$

makes RR-d have the same output distribution as SubMaj.

Similarly, combining Eq. B.8, Eq. B.9 and Eq. B.11, if m is even, setting γ_{Sub} as

$$\gamma_{Sub}(l) = \gamma_{Sub}(K-l) = 1 - 2\sum_{j=\frac{m}{2}+1}^{m}\frac{\binom{l}{j}\binom{K-l}{m-j}}{\binom{K}{m}} - \frac{\binom{l}{\frac{m}{2}}\binom{K-l}{\frac{m}{2}}}{\binom{K}{m}}$$

makes RR-d have the same output distribution as SubMaj.

□

B.1.3 Proof of Lemma 3.4.2

Lemma B.1.3 (Restatement of Lemma 3.4.2)**.** *Let $\mathcal{A}$ be an (ϵ, δ)-differentially private algorithm, where $0 \leq \epsilon < c$ for some constant $c > 0$ and $\delta \in [0, \frac{1}{2})$, that computes the majority of K (ϵ, δ)-differentially private mechanisms $M_1, \ldots, M_K$, where $M_i : \mathcal{D} \to \{0, 1\}$ on dataset $\mathcal{D}$ and $\Pr[M_i(\mathcal{D}) = 1] = p_i, \forall i \in [K]$. Then, the error $\mathcal{E}(\mathcal{A}) \geq |\Pr[g(\mathcal{S}) = 1] - \frac{1}{K}\sum_{i=1}^{K} p_i|$, where $g(\mathcal{S})$ is the probability of the true majority output being 1 as defined in Definition 3.1.1.*

Proof. Consider the setting where M_i's are i.i.d., i.e., $\Pr[M_i(\mathcal{D}) = 1] = p, \forall i \in [K]$ for some $p \in [0, 1]$ on any dataset $\mathcal{D}$. Then, it suffices to show $\mathcal{E}(\mathcal{A}) \geq |\Pr[g(\mathcal{S})] = 1 - p|$.

Consider a dataset $\mathcal{D}_0$ such that $\Pr[\mathcal{M}_i(\mathcal{D}_0) = 1] = \Pr[\mathcal{M}_i(\mathcal{D}_0) = 0] = \frac{1}{2}$ and without loss of generality, we may assume $\Pr[\mathcal{A}(\mathcal{D}_0) = 1] \leq \frac{1}{2}$.

Now, we construct a sequence of datasets $\mathcal{D}_1, \mathcal{D}_2, \ldots, \mathcal{D}_L$, such that $\mathcal{D}_j$ and $\mathcal{D}_{j+1}$ are neighboring datasets and $\Pr[M_i(\mathcal{D}_j) = 1] = \frac{1}{2}e^{j\epsilon} + \sum_{l=0}^{j-1} e^{l\epsilon}\delta$, $\forall i \in [K]$, $\forall j \in [L]$. Choose $L \in \mathbb{N}$ such that $\frac{1}{2}e^{L\epsilon} + \sum_{l=0}^{L-1} e^{\epsilon l}\delta = p$, for some $p > \frac{1}{2}$.

Now, by definition of differential privacy,

$$\begin{aligned}
&\Pr[\mathcal{A}(\mathcal{D}_1) = 1] \leq e^{\epsilon} \Pr[\mathcal{A}(\mathcal{D}_0) = 1] + \delta \\
&\Pr[\mathcal{A}(\mathcal{D}_2) = 1] \leq e^{\epsilon} \Pr[\mathcal{A}(\mathcal{D}_1) = 1] + \delta \leq e^{2\epsilon} \Pr[\mathcal{A}(\mathcal{D}_0) = 1] + e^{\epsilon}\delta + \delta \\
&\ldots \\
&\Pr[\mathcal{A}(\mathcal{D}_L) = 1] \leq e^{L\epsilon} \Pr[\mathcal{A}(\mathcal{D}_0) = 1] + \sum_{l=0}^{L-1} e^{\epsilon l}\delta \leq e^{L\epsilon}\frac{1}{2} + \sum_{l=0}^{L-1} e^{\epsilon l}\delta = p
\end{aligned}$$

Since the probability of true majority being 1 on dataset $\mathcal{D}_L$ is $\Pr[g(\mathcal{S}) = 1] \geq p > \frac{1}{2}$, there is

$$\mathcal{E}(\mathcal{A}) = |\Pr[g(\mathcal{S}) = 1] - \Pr[\mathcal{A}(\mathcal{D}_L) = 1]| \geq \Pr[g(\mathcal{S}) = 1] - p$$

□

B.1.4 Proof of Lemma 3.4.3

Lemma B.1.4 (Restatement of Lemma 3.4.3)**.** *Let $\mathcal{A}$ be any randomized algorithm to compute the majority function g on $\mathcal{S}$ such that for all $\mathcal{S}$, $\Pr[\mathcal{A}(\mathcal{S}) = g(\mathcal{S})] \geq 1/2$ (i.e. $\mathcal{A}$ is at least as good as a random guess). Then, there exists a a general function $\gamma : \{0,1\}^{K+1} \to [0,1]$ such that if one sets p_γ by $\gamma(\mathcal{S})$ in DaRRM, the output distribution of DaRRM_γ is the same as the output distribution of $\mathcal{A}$.*

Proof of Lemma 3.4.3. For some $\mathcal{D}$ and conditioned on $\mathcal{S}$, we see that by definition $\Pr[\mathsf{DaRRM}_\gamma(\mathcal{S}) = g(\mathcal{S})] = \gamma(\mathcal{S}) + (1/2)(1 - \gamma(\mathcal{S}))$. We want to set γ such that $\Pr[\mathsf{DaRRM}_\gamma(\mathcal{S}) = g(\mathcal{S})] = \Pr[\mathcal{A}(\mathcal{S}) = g(\mathcal{S})]$. Therefore, we set $\gamma(\mathcal{S}) = 2\Pr[\mathcal{A}(\mathcal{S}) = g(\mathcal{S})] - 1$.

Lastly, we need to justify that $\gamma \in [0,1]$. Clearly, $\gamma(\mathcal{S}) \leq 2 - 1 \leq 1$ since $\Pr[\mathcal{A}(\mathcal{S}) = g(\mathcal{S})] \leq 1$. Note that the non-negativity follows from assumption. □

B.1.5 Proof of Lemma 3.4.4

Lemma B.1.5 (Restatement of Lemma 3.4.4)**.** *Consider using DaRRM (Algorithm 1) to solve Problem 3.1.1, let $\alpha_l = \Pr[\mathcal{L}(\mathcal{D}) = l]$ and $\alpha'_l = \Pr[\mathcal{L}(\mathcal{D}') = l]$, where $\mathcal{D}$ and $\mathcal{D}'$ are adjacent datasets and $l \in \{0, \ldots, K\}$. For a noise function $\gamma : \{0, 1, \ldots, K\} \to [0,1]$ such that $\gamma(l) = \gamma(K - l), \forall l$, DaRRM_γ is $(m\epsilon, \delta)$-differentially private if and only if for all α_l, α'_l, the following holds,*

$$f(p_1, \ldots, p_K, p'_1, \ldots, p'_K; \gamma) \leq e^{m\epsilon} - 1 + 2\delta \tag{B.12}$$

where f is called the ***privacy cost objective*** *and*

$$f(p_1, \ldots, p_K, p'_1, \ldots, p'_K; \gamma) := \sum_{l=0}^{\frac{K-1}{2}} (e^{m\epsilon}\alpha'_l - \alpha_l) \cdot \gamma(l) + \sum_{l=\frac{K+1}{2}}^{K} (\alpha_l - e^{m\epsilon}\alpha'_l) \cdot \gamma(l)$$

Proof of Lemma 3.4.4. By the definition of differential privacy (Definition 3.3.1),

DaRRM_γ is $(m\epsilon, \delta)$-differentially private

$\iff \Pr[\mathsf{DaRRM}_\gamma(\mathcal{D}) = 1] \leq e^{m\epsilon}\Pr[\mathsf{DaRRM}_\gamma(\mathcal{D}') = 1] + \delta,$

$$\text{and } \Pr[\mathsf{DaRRM}_\gamma(\mathcal{D}) = 0] \leq e^{m\epsilon} \Pr[\mathsf{DaRRM}_\gamma(\mathcal{D}') = 0] + \delta, \quad \forall \text{ adjacent datasets } \mathcal{D}, \mathcal{D}' \tag{B.13}$$

Let random variables $\mathcal{L}(\mathcal{D}) = \sum_{i=1}^{K} S(\mathcal{D})$ and $\mathcal{L}(\mathcal{D}') = \sum_{i=1}^{K} S(\mathcal{D}')$ be the sum of observed outcomes on adjacent datasets $\mathcal{D}$ and $\mathcal{D}'$, based on which one sets p_γ in DaRRM. Let $\alpha_l = \Pr[\mathcal{L}(\mathcal{D}) = l]$ and $\alpha'_l = \Pr[\mathcal{L}(\mathcal{D}') = l]$, $\forall l \in \{0, 1, \ldots, K\}$.

Consider the output being 1.

$$\begin{aligned}
&\Pr[\mathsf{DaRRM}_\gamma(\mathcal{D}) = 1] \leq e^{m\epsilon} \Pr[\mathsf{DaRRM}_\gamma(\mathcal{D}') = 1] + \delta \qquad \text{(B.14)}\\
&\iff \sum_{l=0}^{K} \Pr[\mathsf{DaRRM}_\gamma(\mathcal{D}) = 1 \mid \mathcal{L}(\mathcal{D}) = l] \cdot \Pr[\mathcal{L}(\mathcal{D}) = l]\\
&\quad \leq e^{m\epsilon} \Big(\sum_{l=0}^{K} \Pr[\mathsf{DaRRM}_\gamma(\mathcal{D}') = 1 \mid \mathcal{L}(\mathcal{D}') = l] \cdot \Pr[\mathcal{L}(\mathcal{D}') = l] \Big) + \delta\\
&\iff \sum_{l=0}^{K} \Big(\gamma(l) \cdot \mathbb{I}\{l \geq \frac{K}{2}\} + \frac{1}{2}(1 - \gamma(l)) \Big) \cdot \Pr[\mathcal{L}(\mathcal{D}) = l]\\
&\quad \leq e^{m\epsilon} \Big(\sum_{l=0}^{K} \Big(\gamma(l) \cdot \mathbb{I}\{l \geq \frac{K}{2}\} + \frac{1}{2}(1 - \gamma(l)) \Big\} \Big) \cdot \Pr[\mathcal{L}(\mathcal{D}') = l] \Big) + \delta\\
&\iff \sum_{l=\frac{K+1}{2}}^{K} \Big(\gamma(l) + \frac{1}{2}(1 - \gamma(l)) \Big) \cdot \Pr[\mathcal{L}(\mathcal{D}) = l] + \sum_{l=0}^{\frac{K-1}{2}} \frac{1}{2}(1 - \gamma(l)) \cdot \Pr[\mathcal{L}(\mathcal{D}) = l]\\
&\quad \leq e^{m\epsilon} \Big(\sum_{l=\frac{K+1}{2}}^{K} \Big(\gamma(l) + \frac{1}{2}(1 - \gamma(l)) \Big) \cdot \Pr[\mathcal{L}(\mathcal{D}) = l] \Big)\\
&\quad + e^{m\epsilon} \Big(\sum_{l=0}^{\frac{K-1}{2}} \frac{1}{2}(1 - \gamma(l)) \cdot \Pr[\mathcal{L}(\mathcal{D}') = l] \Big) + \delta\\
&\iff \sum_{l=\frac{K+1}{2}}^{K} \frac{1}{2}\gamma(l)\alpha_l - \sum_{l=0}^{\frac{K-1}{2}} \frac{1}{2}\gamma(l)\alpha_l + \frac{1}{2}\\
&\quad \leq e^{m\epsilon} \sum_{l=\frac{K+1}{2}}^{K} \frac{1}{2}\gamma(l)\alpha'_l - e^{m\epsilon} \sum_{l=0}^{\frac{K-1}{2}} \frac{1}{2}\gamma(l)\alpha'_l + \frac{1}{2}e^{m\epsilon} + \delta
\end{aligned}$$

$$\iff \sum_{l=\frac{K+1}{2}}^{K} (\alpha_l - e^{m\epsilon}\alpha'_l)\gamma(l) - \sum_{l=0}^{\frac{K-1}{2}} (\alpha_l - e^{m\epsilon}\alpha'_l)\gamma(l) \leq e^{m\epsilon} - 1 + 2\delta \tag{B.15}$$

Similarly, consider the output being 0.

$$\begin{aligned}
&\Pr[\mathsf{DaRRM}_\gamma(\mathcal{D}) = 0] \leq e^{m\epsilon} \Pr[\mathsf{DaRRM}_\gamma(\mathcal{D}') = 0] + \delta \\
&\iff \sum_{l=0}^{K} \Pr[\mathsf{DaRRM}_\gamma(\mathcal{D}) = 0 \mid \mathcal{L}(\mathcal{D}) = l] \cdot \Pr[\mathcal{L}(\mathcal{D}) = l] \\
&\quad \leq e^{m\epsilon} \Big(\sum_{l=0}^{K} \Pr[\mathsf{DaRRM}_\gamma(\mathcal{D}') = 0 \mid \mathcal{L}(\mathcal{D}') = l] \cdot \Pr[\mathcal{L}(\mathcal{D}') = l] \Big) + \delta \\
&\iff \sum_{l=0}^{K} \Big(\gamma(l) \cdot \mathbb{I}\{l < \frac{K}{2}\} + \frac{1}{2}(1 - \gamma(l)) \Big) \cdot \Pr[\mathcal{L}(\mathcal{D}) = l] \\
&\quad \leq e^{m\epsilon} \Big(\sum_{l=0}^{K} \gamma(l) \cdot \mathbb{I}\{l < \frac{K}{2}\} + \frac{1}{2}(1 - \gamma(l)) \Big) \cdot \Pr[\mathcal{L}(\mathcal{D}') = l] + \delta \\
&\iff \sum_{l=0}^{\frac{K-1}{2}} \Big(\gamma(l) + \frac{1}{2}(1 - \gamma(l)) \Big) \cdot \Pr[\mathcal{L}(\mathcal{D}) = l] + \sum_{l=\frac{K+1}{2}}^{K} \frac{1}{2}(1 - \gamma(l)) \cdot \Pr[\mathcal{L}(\mathcal{D}) = l] \\
&\quad \leq e^{m\epsilon} \Big(\sum_{l=0}^{\frac{K-1}{2}} \Big(\gamma(l) + \frac{1}{2}(1 - \gamma(l)) \Big) \cdot \Pr[\mathcal{L}(\mathcal{D}') = l] + \sum_{l=\frac{K+1}{2}}^{K} \frac{1}{2}(1 - \gamma(l)) \cdot \Pr[\mathcal{L}(\mathcal{D}') = l] \Big) + \delta \\
&\iff \sum_{l=0}^{\frac{K-1}{2}} \frac{1}{2}\gamma(l)\alpha_l - \sum_{l=\frac{K+1}{2}}^{K} \frac{1}{2}\gamma(l)\alpha_l + \frac{1}{2} \\
&\quad \leq e^{m\epsilon} \sum_{l=0}^{\frac{K-1}{2}} \frac{1}{2}\gamma(l)\alpha'_l - e^{m\epsilon} \sum_{l=\frac{K+1}{2}}^{K} \frac{1}{2}\gamma(l)\alpha'_l + \frac{1}{2}e^{m\epsilon} + \delta \\
&\iff \sum_{l=0}^{\frac{K-1}{2}} (\alpha_l - e^{m\epsilon}\alpha'_l)\gamma(l) - \sum_{l=\frac{K+1}{2}}^{K} (\alpha_l - e^{m\epsilon}\alpha'_l)\gamma(l) \leq e^{m\epsilon} - 1 + 2\delta
\end{aligned} \tag{B.16}$$

Therefore, plugging Eq. B.15 and Eq. B.16 into Eq. B.13,

DaRRM_γ is $(m\epsilon, \delta)$-differentially private

$$\iff \sum_{l=\frac{K+1}{2}}^{K} (\alpha_l - e^{m\epsilon}\alpha'_l)\gamma(l) - \sum_{l=0}^{\frac{K-1}{2}} (\alpha_l - e^{m\epsilon}\alpha'_l)\gamma(l) \leq e^{m\epsilon} - 1 + 2\delta \tag{B.17}$$

$$\text{and} \sum_{l=0}^{\frac{K-1}{2}} (\alpha_l - e^{m\epsilon}\alpha'_l)\gamma(l) - \sum_{l=\frac{K+1}{2}}^{K} (\alpha_l - e^{m\epsilon}\alpha'_l)\gamma(l) \leq e^{m\epsilon} - 1 + 2\delta \tag{B.18}$$

where $\alpha_l = \Pr[\mathcal{L}(\mathcal{D}) = l]$ and $\alpha'_l = \Pr[\mathcal{L}(\mathcal{D}') = l]$, $\forall l \in \{0, 1, \ldots, K\}$ and $\mathcal{D}, \mathcal{D}'$ are any adjacent datasets.

Next, we show if γ is symmetric around $\frac{K}{2}$, i.e., $\gamma(l) = \gamma(K - l)$, satisfying either one of Eq. B.17 or Eq. B.18 implies satisfying the other one. Following Eq. B.17,

$$\begin{aligned}
&\sum_{l=\frac{K+1}{2}}^{K} (\alpha_l - e^{m\epsilon}\alpha'_l)\gamma(l) - \sum_{l=0}^{\frac{K-1}{2}} (\alpha_l - e^{m\epsilon}\alpha'_l)\gamma(l) \leq e^{m\epsilon} - 1 + 2\delta \\
&\iff \sum_{l=0}^{\frac{K-1}{2}} (\alpha_{K-l} - e^{m\epsilon}\alpha'_{K-l}) \cdot \gamma(K-l) - \sum_{l=\frac{K-1}{2}}^{K} (\alpha_{K-l} - e^{m\epsilon}\alpha'_{K-l}) \cdot \gamma(K-l) \\
&\quad \leq e^{m\epsilon} - 1 + 2\delta \\
&\iff \sum_{l=0}^{\frac{K-1}{2}} (\alpha_{K-l} - e^{m\epsilon}\alpha'_{K-l}) \cdot \gamma(l) - \sum_{l=\frac{K-1}{2}}^{K} (\alpha_{K-l} - e^{m\epsilon}\alpha'_{K-l}) \cdot \gamma(l) \leq e^{m\epsilon} - 1 + 2\delta
\end{aligned} \tag{B.19}$$

Since $\gamma(l) = \gamma(K - l)$

For analysis purpose, we rewrite Eq. B.18 as

$$\sum_{l=0}^{\frac{K-1}{2}} (\widetilde{\alpha}_l - e^{m\epsilon}\widetilde{\alpha}'_l) \cdot \gamma(l) - \sum_{l=\frac{K-1}{2}}^{K} (\widetilde{\alpha}_l - e^{m\epsilon}\widetilde{\alpha}'_l) \cdot \gamma(l) \leq e^{m\epsilon} - 1 + 2\delta \tag{B.20}$$

and proceed by showing Eq. B.19 $\iff$ Eq. B.20.

Recall $p_i = \Pr[M_i(\mathcal{D}) = 1]$ and $p'_i = \Pr[M_i(\mathcal{D}') = 1]$. Observe $\mathcal{L}(\mathcal{D}) \sim \text{PoissonBinomial}(\{p_i\}_{i=1}^K)$ and $\mathcal{L}(\mathcal{D}') \sim \text{PoissonBinomial}(\{p'_i\}_{i=1}^K)$. Let $F_l = \{\mathcal{A} : |\mathcal{A}| = l, \mathcal{A} \subseteq [K]\}$, for any $l \in \{0, \dots, K\}$, denote the set of all subsets of l integers that can be selected from $[K]$. Let $\mathcal{A}^c = [K] \setminus \mathcal{A}$ be $\mathcal{A}$'s complement set. Notice $F_{K-l} = \{\mathcal{A}^c : \mathcal{A} \in F_l\}$.

Since α denotes the pmf of the Poisson Binomial distribution at l, it follows that

$$\alpha_l = \Pr[\mathcal{L}(\mathcal{D}) = l] = \sum_{\mathcal{A} \in F_l} \Pi_{i \in \mathcal{A}} p_i \Pi_{j \in \mathcal{A}^c}(1 - p_j) \tag{B.21}$$

Consider $\beta_i = 1 - p_i, \forall i \in [K]$ and a new random variable $\mathcal{L}^\beta \sim \text{PoissonBinomial}(\{\beta_i\}_{i=1}^K)$, and let $\widetilde{\alpha}_l = \Pr[\mathcal{L}^\beta = 1]$. Observe that

$$\begin{aligned}
\widetilde{\alpha}'_l = \Pr[\mathcal{L}^\beta = l] &= \sum_{\mathcal{A} \in F_l} \Pi_{j \in \mathcal{A}} \beta_i \Pi_{i \in \mathcal{A}^c}(1 - \beta_i) = \sum_{\mathcal{A} \in F_l} \Pi_{j \in \mathcal{A}}(1 - p_j) \Pi_{i \in \mathcal{A}^c} p_i \\
&= \sum_{\mathcal{A}^c \in F_{K-l}} \Pi_{j \in \mathcal{A}}(1 - p_i) \Pi_{i \in \mathcal{A}^c} p_i = \sum_{\mathcal{A} \in F_{K-l}} \Pi_{i \in \mathcal{A}} p_i \Pi_{j \in \mathcal{A}^c}(1 - p_i) \\
&= \alpha_{K-l}
\end{aligned} \tag{B.22}$$

Similarly, consider $\beta'_i = 1 - p'_i, \forall i \in [K]$ and a new random variable $\mathcal{L}'^\beta \sim \text{PoissonBinomial}(\beta'_i\}_{i=1}^L)$, and let $\widetilde{\alpha}'_l = \Pr[\mathcal{L}'^\beta = 1]$. Then, $\widetilde{\alpha}'_l = \alpha'_{K-l}$.

Since Eq. B.19 holds for all possible α_{K-l}, α'_{K-l}, Eq. B.20 then holds for all $\widetilde{\alpha}_l, \widetilde{\alpha}'_l$ in the K-simplex, and so Eq. B.20 follows by relabeling α_{K-l} as $\widetilde{\alpha}_l$ and α'_{K-l} as $\widetilde{\alpha}'_l$.

The above implies Eq. B.17 $\iff$ Eq. B.18. Therefore,

$$\begin{aligned}
&\mathsf{DaRRM}_\gamma \text{ is } (m\epsilon, \delta)\text{-differentially private} \\
\iff &\underbrace{\sum_{l=\frac{K+1}{2}}^{K} (\alpha_l - e^{m\epsilon}\alpha'_l)\gamma(l) - \sum_{l=0}^{\frac{K-1}{2}} (\alpha_l - e^{m\epsilon}\alpha'_l)\gamma(l)}_{:= f(p_1, \dots, p_K, p'_1, \dots, p'_K; \gamma)} \leq e^{m\epsilon} - 1 + 2\delta
\end{aligned} \tag{B.23}$$

□

B.2 Details of Section 3.5: Provable Privacy Amplification

In this section, we consider Problem 3.1.1 in the pure differential privacy and i.i.d. mechanisms setting. That is, $\delta = \Delta = 0$ and $p = p_i = \Pr[M_i(\mathcal{D}) = 1], p' = p'_i = \Pr[M_i(\mathcal{D}') = 1], \forall i \in [K]$. Our goal is to search for a good noise function γ such that: 1) DaRRM_γ is $m\epsilon$-DP, and 2) DaRRM_γ achieves higher utility than that of the baselines (see Section 3.4) under a fixed privacy loss. Our main finding of such a γ function is presented in Theorem 3.5.1, which states given a privacy allowance $m \in [K]$, one can indeed output the majority of $2m-1$ subsampled mechanisms, instead of just m as indicated by simple composition. Later, we formally verify in Lemma B.2.11, Section B.2.3 that taking the majority of more mechanisms strictly increases the utility.

To start, by Lemma 3.4.4, for any noise function γ, γ satisfying goal 1) is equivalent to satisfying

$$f(p, p'; \gamma) \leq e^\epsilon - 1 \tag{B.24}$$

where $f(p, p'; \gamma) = \sum_{l=0}^{\frac{K-1}{2}} (e^{m\epsilon}\alpha'_l - \alpha_l) \cdot \gamma(l) + \sum_{l=\frac{K+1}{2}}^{K} (\alpha_l - e^{m\epsilon}\alpha'_l) \cdot \gamma(l)$ refers to the privacy cost objective (see Lemma 3.4.4) in the i.i.d. mechanisms setting, and recall $\alpha_l = \Pr[\mathcal{L}(\mathcal{D}) = l]$ and $\alpha'_l = \Pr[\mathcal{L}(\mathcal{D}') = l]$, $\forall l \in \{0, 1, \ldots, K\}$. Notice in this setting, $\mathcal{L}(\mathcal{D}) \sim \text{Binomial}(p)$, and $\mathcal{L}(\mathcal{D}') \sim \text{Binomial}(p')$.

Monotonicity Assumption. For analysis, we restrict our search for a γ function with good utility to the class with a mild monotonicity assumption: $\gamma(l) \geq \gamma(l+1), \forall l \leq \frac{K-1}{2}$ and $\gamma(l) \leq \gamma(l+1), \forall l \geq \frac{K+1}{2}$. This matches our intuition that as $\mathcal{L}(\mathcal{D}) = \sum_{i=1}^{K} S_i$, i.e., the number of mechanisms outputting 1, approaches 0 or K, there is a clearer majority and so not much noise is needed to ensure privacy, which implies a larger value of γ.

Roadmap of Proof of Theorem 3.5.1. Since γ needs to enable Eq. B.24 to be satisfied for all $p, p' \in [0, 1]$, we begin by showing characteristics of **the worst case probabilities**, i.e., $(p^*, p'^*) = \text{argmax}_{(p,p')} f(p, p'; \gamma)$, given any $\gamma : \{0, 1, \ldots, K\} \to [0, 1]$ that is symmetric around $\frac{K}{2}$ and that satisfies the above monotonicity assumption,

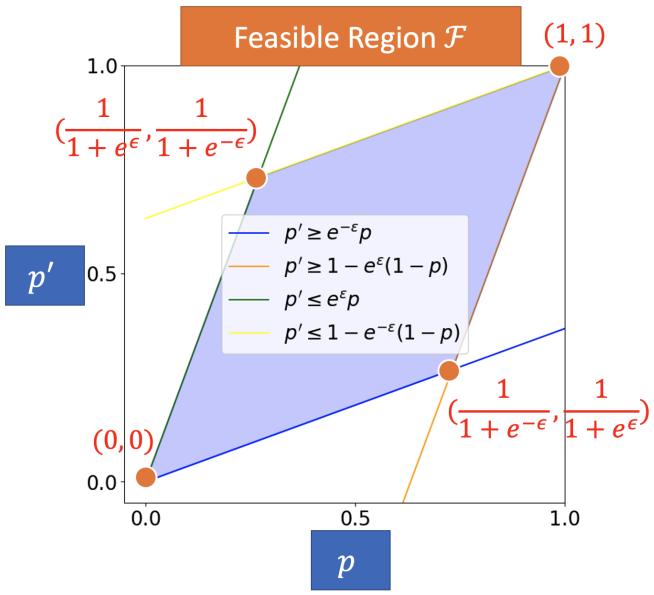

Figure B.2: The feasible region $\mathcal{F}$ is plotted as the blue area. The four boundaries are implied by p, p' satisfying ϵ-differential privacy.

in Lemma B.2.1, Section B.2.1. We call (p^*, p'^*) the worst case probabilities, since they incur the largest privacy loss. Later in Section B.2.2, we present the main proof of Theorem 3.5.1, where we focus on searching for a good γ that enables $f(p^*, p'^*; \gamma) \leq e^{\epsilon} - 1$, based on the characteristics of (p^*, p'^*) in Lemma B.2.1, to ensure DaRRM_{γ} is $m\epsilon$-differentially private.

B.2.1 Characterizing the Worst Case Probabilities

First, note (p, p') are close to each other and lie in a feasible region $\mathcal{F}$, due to each mechanism M_i being ϵ-differentially private; and so does (p^*, p'^*). The feasible region, as illustrated in Figure B.2, is bounded by (a) $p' \leq e^{\epsilon}p$ (b) $p \leq e^{\epsilon}p'$ (c) $1 - p' \leq e^{\epsilon}(1 - p)$, and (d) $1 - p \leq e^{\epsilon}(1 - p')$, where the four boundaries are derived from the definition of differential privacy. Therefore, we only need to search for $(p^*, p'^*) = \text{argmax}_{(p,p') \in \mathcal{F}} f(p, p'; \gamma)$.

Next, we show that given γ satisfying certain conditions, (p^*, p'^*) can only be on two of the four boundaries of $\mathcal{F}$ in Lemma B.2.1 — that is, either $p^* = e^{\epsilon}p'$, i.e., on the blue line in Figure B.2, or $1 - p'^* = e^{\epsilon}(1 - p^*)$, i.e., on the orange line in

Figure B.2.

Lemma B.2.1 (Characteristics of worst case probabilities)**.** *For any noise function* $\gamma : \{0, 1, \ldots, K\} \to [0, 1]$ *that is 1) symmetric around* $\frac{K}{2}$*, 2) satisfies the monotonicity assumption, and 3)* $\gamma(\frac{K-1}{2}) > 0$ *and* $\gamma(\frac{K+1}{2}) > 0$*, the worst case probabilities given* γ*,* $(p^*, p'^*) = \text{argmax}_{(p,p')\in\mathcal{F}} f(p, p'; \gamma)$*, must satisfy one of the following two equalities:*

$$\begin{aligned} & p^* = e^\epsilon p'^*, && \forall p^* \in [0, \frac{1}{e^{-\epsilon}+1}], p'^* \in [0, \frac{1}{1+e^\epsilon}] \\ \textit{or} \quad & 1 - p'^* = e^\epsilon (1 - p^*), && \forall p^* \in [\frac{1}{1+e^{-\epsilon}}, 1], p'^* \in [\frac{1}{1+e^\epsilon}, 1] \end{aligned}$$

To show Lemma B.2.1, we first show in Lemma B.2.2 that the search of (p^*, p'^*) can be refined to one of the four boundaries of $\mathcal{F}$, via a careful gradient analysis of $f(p, p'; \gamma)$ in $\mathcal{F}$, and then show in Lemma B.2.3 that the search of (p^*, p'^*) can be further refined to two of the four boundaries, due to symmetry of p, p'. Lemma B.2.1 directly follows from the two.

Lemma B.2.2. *For any noise function* $\gamma : \{0, 1, \ldots, K\} \to [0, 1]$ *that is 1) symmetric around* $\frac{K}{2}$*, 2) satisfies the monotonicity assumption, and 3)* $\gamma(\frac{K-1}{2}) > 0$ *and* $\gamma(\frac{K+1}{2}) > 0$*, the worst case probabilities given* γ*,* $(p^*, p'^*) = \text{argmax}_{(p,p')\in\mathcal{F}} f(p, p'; \gamma)$*, must satisfy one of the following four equalities:*

$$\begin{aligned} & p'^* = e^\epsilon p^*, && \forall p^* \in [0, \frac{1}{1+e^\epsilon}], p'^* \in [0, \frac{1}{1+e^{-\epsilon}}] \\ & p^* = e^\epsilon p'^*, && \forall p^* \in [0, \frac{1}{e^{-\epsilon}+1}], p'^* \in [0, \frac{1}{1+e^\epsilon}] \\ & 1 - p^* = e^\epsilon (1 - p'^*), && \forall p^* \in [\frac{1}{1+e^\epsilon}, 1], p'^* \in [\frac{1}{1+e^{-\epsilon}}, 1] \\ & 1 - p'^* = e^\epsilon (1 - p^*), && \forall p^* \in [\frac{1}{1+e^{-\epsilon}}, 1], p'^* \in [\frac{1}{1+e^\epsilon}, 1] \end{aligned}$$

Proof of Lemma B.2.2. Recall the privacy cost objective (as defined in Lemma 3.4.4) is now

$$f(p, p'; \gamma) = \sum_{l=0}^{\frac{K-1}{2}} (e^{m\epsilon}\alpha'_l - \alpha_l) \cdot \gamma(l) + \sum_{l=\frac{K+1}{2}}^{K} (\alpha_l - e^{m\epsilon}\alpha'_l) \cdot \gamma(l)$$

where $\alpha_l = \Pr[\mathcal{L}(\mathcal{D}) = l]$ and $\alpha'_l = \Pr[\mathcal{L}(\mathcal{D}') = l]$, $\forall l \in \{0, 1, \ldots, K\}$. Since

$\mathcal{L}(\mathcal{D}) \sim \text{Binomial}(p)$ and $\mathcal{L}(\mathcal{D}') \sim \text{Binomial}(p')$ in the i.i.d. mechanisms setting, and using the pmf of the Binomial distribution, f can be written as

$$f(p,p';\gamma) = \sum_{l=0}^{\frac{K-1}{2}} (e^{m\epsilon}\binom{K}{l}p'^l(1-p')^{K-l} - \binom{K}{l}p^l(1-p)^{K-l}) \cdot \gamma(l) + \sum_{l=\frac{K+1}{2}}^{K} (\binom{K}{l}p^l(1-p)^{K-l} - e^{m\epsilon}\binom{K}{l}p'^l(1-p')^{K-l})$$

The gradients w.r.t. p and p' are

$$\nabla_p f(p,p';\gamma) = \underbrace{\sum_{l=0}^{\frac{K-1}{2}} -\binom{K}{l}\gamma(l) \cdot (lp^{l-1}(1-p)^{K-l} - p^l(K-l)(1-p)^{K-l-1})}_{:=A} + \underbrace{\sum_{l=\frac{K+1}{2}}^{K} \binom{K}{l}\gamma(l) \cdot (lp^{l-1}(1-p)^{K-l} - p^l(K-l)(1-p)^{K-l-1})}_{:=B} \tag{B.25}$$

and

$$\nabla_{p'} f(p,p';\gamma) = \sum_{l=0}^{\frac{K-1}{2}} e^{m\epsilon}\binom{K}{l}\gamma(l) \cdot (lp'^{l-1}(1-p')^{K-l} - p'^l(K-l)(1-p')^{K-l-1}) + \sum_{l=\frac{K+1}{2}}^{K} -e^{m\epsilon}\binom{K}{l}\gamma(l) \cdot (lp'^{l-1}(1-p')^{K-l} - p'^l(K-l)(1-p')^{K-l-1})$$

We show in the following $\forall p \in (0,1)$, $\nabla_p f(p,p';\gamma) > 0$ and $\nabla_{p'} f(p,p';\gamma) < 0$. This implies there is no local maximum inside $\mathcal{F}$, and so $(p^*, p'^*) = \text{argmax}_{p,p'} f(p,p';\gamma)$ must be on one of the four boundaries of $\mathcal{F}$. Also, if $p = 0$, then $p' = 0$, and $(0,0)$ is a corner point at the intersection of two boundaries. Similarly, if $p = 1$, then $p' = 1$, and $(1,1)$ is also a corner point. This concludes $\forall p \in [0,1]$, $(p^*, p'^*) = \text{argmax}_{p,p'} f(p,p';\gamma)$ must be on one of the four boundaries of $\mathcal{F}$.

To show $\nabla_p f(p,p';\gamma) > 0$ for $p \in (0,1)$, we write $\nabla_p f(p,p';\gamma) = A + B$ as in Eq. B.25, and show that $A > 0$ and $B > 0$.

To show $A > 0$, first note

$$
\begin{aligned}
A &:= \sum_{l=0}^{\frac{K-1}{2}} \gamma(l) \binom{K}{l} \cdot (p^l (K-l)(1-p)^{K-l-1} - l p^{l-1}(1-p)^{K-l}) > 0 \qquad \text{(B.26)} \\
&\iff \sum_{l=0}^{\frac{K-1}{2}} \gamma(l) \binom{K}{l} \cdot p^l (K-l)(1-p)^{K-l-1} > \sum_{l=0}^{\frac{K-1}{2}} \gamma(l) \binom{K}{l} \cdot l p^{l-1}(1-p)^{K-l} \\
&\iff \sum_{l=0}^{\frac{K-1}{2}} \gamma(l) \binom{K-1}{l} \frac{K}{K-l} \cdot p^l (K-l)(1-p)^{K-l-1} \\
&\quad > \sum_{l=1}^{\frac{K-1}{2}} \gamma(l) \binom{K-1}{l-1} \frac{K}{l} \cdot l p^{l-1}(1-p)^{K-l} \\
&\iff K \sum_{l=0}^{\frac{K-1}{2}} \gamma(l) \binom{K-1}{l} p^l (1-p)^{K-l-1} > K \sum_{l=1}^{\frac{K-1}{2}} \gamma(l) \binom{K-1}{l-1} p^{l-1}(1-p)^{K-l} \\
&\iff \sum_{l=0}^{\frac{K-1}{2}} \gamma(l) \binom{K-1}{l} p^l (1-p)^{K-l-1} > \sum_{l=0}^{\frac{K-1}{2}-1} \gamma(l+1) \binom{K-1}{l} p^l (1-p)^{K-l-1} \qquad \text{(B.27)}
\end{aligned}
$$

Since $\forall l \leq \frac{K-1}{2}$, $\gamma(l) \geq \gamma(l+1)$ and $p \in (0,1)$, there is for $l \in \{0, \ldots, \frac{K-1}{2} - 1\}$,

$$
\gamma(l) \binom{K-1}{l} p^l (1-p)^{K-l-1} \geq \gamma(l+1) \binom{K-1}{l} p^l (1-p)^{K-l-1} \qquad \text{(B.28)}
$$

Furthermore, since $\gamma(\frac{K-1}{2}) > 0$ and $p \in (0,1)$,

$$
\gamma(\frac{K-1}{2}) \binom{K-1}{\frac{K-1}{2}} p^{\frac{K-1}{2}} (1-p)^{\frac{K-1}{2}} > 0 \qquad \text{(B.29)}
$$

Eq. B.28 and Eq. B.29 combined implies

$$
\gamma(\frac{K-1}{2}) \binom{K-1}{\frac{K-1}{2}} p^{\frac{K-1}{2}} (1-p)^{\frac{K-1}{2}} + \sum_{l=0}^{\frac{K-1}{2}-1} \gamma(l) \binom{K-1}{l} p^l (1-p)^{K-l-1}
$$

$$> \sum_{l=0}^{\frac{K-1}{2}-1} \gamma(l+1)\binom{K-1}{l} p^l (1-p)^{K-l-1}$$

and hence, Eq. B.27 holds. This further implies $A > 0$.

Next, to show $B > 0$, note that

$$\begin{aligned}
B &:= \sum_{l=\frac{K+1}{2}}^{K} \binom{K}{l} \gamma(l) \cdot (l p^{l-1}(1-p)^{K-l} - p^l (K-l)(1-p)^{K-l-1}) > 0 \qquad \text{(B.30)} \\
&\iff \sum_{l=\frac{K+1}{2}}^{K} \binom{K}{l} \gamma(l) \cdot l p^{l-1}(1-p)^{K-l} > \sum_{l=\frac{K+1}{2}}^{K} \binom{K}{l} p^l (K-l)(1-p)^{K-l-1} \\
&\iff \sum_{l=\frac{K+1}{2}}^{K} \gamma(l) \binom{K-1}{l-1} \frac{K}{l} \cdot l p^{l-1}(1-p)^{K-l} \\
&> \sum_{l=\frac{K+1}{2}}^{K-1} \gamma(l) \binom{K-1}{l} \frac{K}{K-l} \cdot p^l (K-l)(1-p)^{K-l-1} \\
&\iff K \sum_{l=\frac{K+1}{2}}^{K} \gamma(l) \binom{K-1}{l-1} \cdot p^{l-1}(1-p)^{K-l} \\
&> K \sum_{l=\frac{K+1}{2}}^{K-1} \gamma(l) \binom{K-1}{l} \cdot p^l (1-p)^{K-l-1} \\
&\iff \sum_{l=\frac{K+1}{2}}^{K} \gamma(l) \binom{K-1}{l-1} \cdot p^{l-1}(1-p)^{K-l} > \sum_{l=\frac{K+1}{2}+1}^{K} \gamma(l-1) \binom{K-1}{l-1} \cdot p^{l-1}(1-p)^{K-l}
\end{aligned}$$

(B.31)

Since $\forall l \geq \frac{K+1}{2}$, $\gamma(l) \geq \gamma(l-1)$ and $p \in (0,1)$, there is for $l \in \{\frac{K+1}{2}+1, \ldots, K\}$,

$$\gamma(l)\binom{K-1}{l-1} p^{l-1}(1-p)^{K-l} \geq \gamma(l-1)\binom{K-1}{l-1} p^{l-1}(1-p)^{K-l} \qquad \text{(B.32)}$$

Furthermore, since $\gamma(\frac{K+1}{2}) > 0$ and $p \in (0,1)$,

$$\gamma(\frac{K+1}{2})\binom{K-1}{\frac{K-1}{2}} p^{\frac{K-1}{2}}(1-p)^{\frac{K-1}{2}} > 0 \qquad \text{(B.33)}$$

Eq. B.32 and Eq. B.33 combined implies

$$\gamma(\frac{K+1}{2})\binom{K-1}{\frac{K-1}{2}}p^{\frac{K-1}{2}}(1-p)^{\frac{K-1}{2}} + \sum_{l=\frac{K+1}{2}+1}^{K}\gamma(l)\binom{K-1}{l-1}\cdot p^{l-1}(1-p)^{K-l}$$
$$> \sum_{l=\frac{K+1}{2}+1}^{K}\gamma(l-1)\binom{K-1}{l-1}\cdot p^{l-1}(1-p)^{K-l}$$

and hence Eq. B.31 holds. This further implies $B > 0$.

Following Eq.B.25, for $p \in (0,1)$ and γ satisfying the three assumptions,

$$\nabla_p f(p, p'; \gamma) = A + B > 0$$

Following similar techniques, one can show for $p \in (0,1)$ and γ satisfying the three conditions,

$$\nabla_{p'} f(p, p'; \gamma) < 0$$

This implies there is no local minima or local maxima inside the feasible region $\mathcal{F}$. Also recall $(p, p') \in \{(0,0),(1,1)\}$ are two special cases where (p, p') is at the intersection of two boundaries. Hence, we conclude the worst case probability $(p^*, p'^*) = \text{argmax}_{p,p' \in \mathcal{F}} f(p, p'; \gamma)$ is on one of the four boundaries of $\mathcal{F}$ — that is, (p^*, p'^*) satisfy one of the following:

$$\begin{aligned}
&p'^* = e^\epsilon p^*, && \forall p \in [0, \frac{1}{1+e^\epsilon}], p' \in [0, \frac{1}{1+e^{-\epsilon}}] \\
&p^* = e^\epsilon p'^*, && \forall p \in [0, \frac{1}{e^{-\epsilon}+1}], p' \in [0, \frac{1}{1+e^\epsilon}] \\
&1-p^* = e^\epsilon(1-p'^*), && \forall p \in [\frac{1}{1+e^\epsilon}, 1], p' \in [\frac{1}{1+e^{-\epsilon}}, 1] \\
&1-p'^* = e^\epsilon(1-p^*), && \forall p \in [\frac{1}{1+e^{-\epsilon}}, 1], p' \in [\frac{1}{1+e^\epsilon}, 1]
\end{aligned}$$

□

Lemma B.2.3. *For any noise function $\gamma : \{0, 1, \ldots, K\} \to [0,1]$ function that is 1) symmetric around $\frac{K}{2}$ and 2) satisfies the monotonicity assumption, the privacy cost*

objective $f(p, p'; \gamma)$ *is maximized when* $p \geq p'$.

Proof of Lemma B.2.3. Following Eq. B.14 and Eq. B.15 in the proof of Lemma 3.4.4, and that $\delta = 0$,

$$\begin{aligned}
&\Pr[\mathsf{DaRRM}_\gamma(\mathcal{D}) = 1] \leq e^{m\epsilon} \Pr[\mathsf{DaRRM}_\gamma(\mathcal{D}') = 1] \\
\iff &\underbrace{\sum_{l=\frac{K+1}{2}}^{K} (\alpha_l - e^{m\epsilon}\alpha'_l)\gamma(l) - \sum_{l=0}^{\frac{K-1}{2}} (\alpha_l - e^{m\epsilon}\alpha'_l)\gamma(l)}_{=f(p,p';\gamma)} \leq e^{m\epsilon} - 1
\end{aligned}$$

where $\alpha_l = \Pr[\mathcal{L}(\mathcal{D}) = l]$ and $\alpha'_l = \Pr[\mathcal{L}(\mathcal{D}') = l]$, $\forall l \in \{0, 1, \ldots, K\}$. This implies

$$f(p, p'; \gamma) = \frac{\Pr[\mathsf{DaRRM}_\gamma(\mathcal{D}) = 1]}{\Pr[\mathsf{DaRRM}_\gamma(\mathcal{D}') = 1]} - 1$$

Hence, $f(p, p'; \gamma)$ is maximized when $\Pr[\mathsf{DaRRM}_\gamma(\mathcal{D}) = 1] \geq \Pr[\mathsf{DaRRM}_\gamma(\mathcal{D}') = 1]$.

$$\begin{aligned}
&\Pr[\mathsf{DaRRM}_\gamma(\mathcal{D}) = 1] \\
&= \sum_{l=0}^{K} \Pr[\mathsf{DaRRM}_\gamma(\mathcal{D}) = 1 \mid \mathcal{L}(\mathcal{D}) = 1] \cdot \Pr[\mathcal{L}(\mathcal{D}) = l] \\
&= \sum_{l=0}^{K} \left(\gamma(l) \cdot \mathbb{I}\{l \geq \frac{K}{2}\} + \frac{1}{2}(1 - \gamma(l))\right) \cdot \Pr[\mathcal{L}(\mathcal{D}) = l] \\
&= \sum_{l=0}^{\frac{K-1}{2}} \frac{1}{2}(1 - \gamma(l)) \cdot \alpha_l + \sum_{l=\frac{K+1}{2}}^{K} \left(\gamma(l) + \frac{1}{2}(1 - \gamma(l))\right) \cdot \alpha_l \\
&= \frac{1}{2} \sum_{l=\frac{K+1}{2}}^{K} \gamma(l) \binom{K}{l} p^l (1-p)^{K-l} - \frac{1}{2} \sum_{l=0}^{\frac{K-1}{2}} \gamma(l) \binom{K}{l} p^l (1-p)^{K-l-1} + \frac{1}{2}
\end{aligned}$$

where the last line follows from the observation that in the i.i.d. mechanisms setting, $\mathcal{L}(\mathcal{D}) \sim \text{Binomial}(p)$ and α_l is hence the pmf of the Binomial distribution at l.

Similarly,

$$\Pr[\mathsf{DaRRM}_\gamma(\mathcal{D}') = 1]$$

$$= \frac{1}{2}\sum_{l=\frac{K+1}{2}}^{K}\gamma(l)\binom{K}{l}p'^{l}(1-p')^{K-l} - \frac{1}{2}\sum_{l=0}^{\frac{K-1}{2}}\gamma(l)\binom{K}{l}p'^{l}(1-p')^{K-l-1} + \frac{1}{2}$$

Now define the objective

$$h(\beta) = \frac{1}{2}\sum_{l=\frac{K+1}{2}}^{K}\gamma(l)\binom{K}{l}\beta^{l}(1-\beta)^{K-l} - \frac{1}{2}\sum_{l=0}^{\frac{K-1}{2}}\gamma(l)\binom{K}{l}\beta^{l}(1-\beta)^{K-l-1} + \frac{1}{2} \quad \text{(B.34)}$$

for $\beta \in [0,1]$ and it follows that $\Pr[\mathsf{DaRRM}_\gamma(\mathcal{D}) = 1] = h(p)$ and $\Pr[\mathsf{DaRRM}_\gamma(\mathcal{D}') = 1] = h(p')$. We now analyze the monotonicity of $h(\beta)$ in β.

For ease of presentation, define $g(l) := \begin{cases} -\frac{1}{2}\gamma(l) & \forall l \le \frac{K}{2} \\ \frac{1}{2}\gamma(l) & \forall l \ge \frac{K}{2} \end{cases}$. Since $\gamma(l) \ge \gamma(l+1), \forall l \le \frac{K}{2}$ and $\gamma(l+1) \ge \gamma(l), \forall l \ge \frac{K}{2}$, there is $g(l+1) \ge g(l), \forall l \in \{0,\dots,K\}$. And replacing $\gamma(l)$ with $g(l)$ in Eq. B.34,

$$h(\beta) = \sum_{l=0}^{K} g(l)\binom{K}{l}\beta^{l}(1-\beta)^{K-l}$$

and

$$\begin{aligned}
&\nabla_\beta h(\beta) \\
&= \sum_{l=0}^{K} g(l)\binom{K}{l}\left(l\beta^{l-1}(1-\beta)^{K-l} - (K-l)\beta^{l}(1-\beta)^{K-l-1}\right) \\
&= \sum_{l=1}^{K} g(l)\binom{K-1}{l-1}\frac{K}{l}l\beta^{l-1}(1-\beta)^{K-l} - \sum_{l=0}^{K-1}\binom{K-1}{l}\frac{K}{K-l}(K-l)\beta^{l}(1-\beta)^{K-l-1} \\
&= K\sum_{l=1}^{K}\binom{K-1}{l-1}\beta^{l-1}(1-\beta)^{K-l} - K\sum_{l=0}^{K-1}\binom{K-1}{l}\beta^{l}(1-\beta)^{K-l-1} \\
&= K\sum_{l=0}^{K-1} g(l+1)\binom{K-1}{l}\beta^{l}(1-\beta)^{K-l-1} - K\sum_{l=0}^{K-1} g(l)\binom{K-1}{l}\beta^{l}(1-\beta)^{K-l-1} \\
&= K\sum_{l=0}^{K-1}\left(g(l+1) - g(l)\right)\binom{K-1}{l}\beta^{l}(1-\beta)^{K-l-1}
\end{aligned}$$

Since $g(l+1) \geq g(l)$ and $\binom{K-1}{l}\beta^l(1-\beta)^{K-l-1} \geq 0$, $\nabla_\beta h(\beta) \geq 0$. This implies $h(\beta)$ is monotonically non-decreasing in β and hence,

$$\Pr[\mathsf{DaRRM}_\gamma(\mathcal{D}) = 1] \geq \Pr[\mathsf{DaRRM}_\gamma(\mathcal{D}') = 1] \iff p \geq p' \tag{B.35}$$

Therefore, $f(p, p'; \gamma)$ is maximized when $p \geq p'$. □

B.2.2 Proof of Privacy Amplification (Theorem 3.5.1)

Theorem B.2.4 (Restatement of Theorem 3.5.1)**.** *Consider using DaRRM (Algorithm 1) to solve Problem 3.1.1, with i.i.d. mechanisms $\{M_i\}_{i=1}^K$, i.e., $p_i = p$, $p_i' = p'$, $\forall i \in [K]$, the privacy allowance $m \in [K]$ and $\delta = \Delta = 0$. Let the noise function $\gamma : \{0, 1, \ldots, K\} \to [0, 1]$ be that:*
if $m \geq \frac{K+1}{2}$,

$$\gamma(l) = 1$$

and if $m \leq \frac{K-1}{2}$,

$$\gamma(l) = \begin{cases} 1 - 2h(l) & \forall l \leq \frac{K-1}{2} \\ 2h(l) - 1 & \forall l \geq \frac{K+1}{2} \end{cases}$$

where $h(l) = \sum_{i=m}^{2m-1} \frac{\binom{l}{i}\binom{K-l}{2m-1-i}}{\binom{K}{2m-1}}$, then DaRRM$_\gamma$ is $m\epsilon$-differentially private.

Roadmap. Theorem 3.5.1 consists of two parts: γ under a large privacy allowance $m \geq \frac{K+1}{2}$ and γ under a small privacy allowance $m \leq \frac{K-1}{2}$. We first show in Lemma B.2.5, Section B.2.2 that if $m \geq \frac{K+1}{2}$, setting $\gamma = 1$ suffices to ensure DaRRM_γ to be $m\epsilon$-differentially private, and hence one can always output the true majority of K mechanisms. In contrast, simple composition indicates only when $m = K$ can one output the true majority of K mechanisms. Next, we show in Lemma B.2.10, Section B.2.2 that if $m \leq \frac{K-1}{2}$, one can set γ to be γ_{DSub}, which corresponds to outputting the majority of $2m-1$ subsampled mechanisms (and hence the name "Double Subsampling", or DSub). In contrast, simple compositon indicates one can only output the majority of m subsampled mechanisms to make

sure the output is $m\epsilon$-differentially private. **Theorem 3.5.1 follows directly from combining Lemma B.2.5 and Lemma B.2.10.**

Privacy Amplification Under A Large Privacy Allowance $m \geq \frac{K+1}{2}$

The proof of Lemma B.2.5 is straightforward. We show that given the constant $\gamma_{max}(l) = 1$, if $m \geq \frac{K+1}{2}$, the worst case probabilities are $(p^*, p'^*) = \text{argmax}_{(p,p')\in\mathcal{F}} f(p, p'; \gamma_{max}) = (0, 0)$ and notice that $f(0, 0; \gamma_{max}) = e^{m\epsilon} - 1$, which satisfies the condition in Lemma 3.4.4. Hence, $\mathsf{DaRRM}_{\gamma_{max}}$ is $m\epsilon$-differentially private.

Lemma B.2.5 (Privacy amplification, $m \geq \frac{K+1}{2}$)**.** *Consider using DaRRM (Algorithm 1) to solve Problem 3.1.1, with i.i.d. mechanisms $\{M_i\}_{i=1}^K$, i.e., $p_i = p$, $p'_i = p'$, $\forall i \in [K]$, the privacy allowance $m \geq \frac{K+1}{2}, m \in \mathbb{Z}$ and $\delta = \Delta = 0$. Let the noise function be the constant $\gamma_{max}(l) = 1, \forall l \in \{0, 1, \ldots, K\}$. Then, $\mathsf{DaRRM}_{\gamma_{max}}$ is $m\epsilon$-differentially private.*

Proof of Lemma B.2.5. First, notice $\gamma_{max}(l) = 1, \forall l \in \{0, 1, \ldots, K\}$ is: 1) symmetric around $\frac{K}{2}$, 2) satisfies the monotonicity assumption, and 3) $\gamma_{max}(\frac{K-1}{2}) > 0$ and $\gamma_{max}(\frac{K+1}{2}) > 0$. Therefore, by Lemma B.2.1, the worst case probabilities given γ_{max}, i.e., $(p^*, p'^*) = \text{argmax}_{(p,p')\in\mathcal{F}} f(p, p'; \gamma_{max})$, are on one of the two boundaries of $\mathcal{F}$, satisfying

$$\begin{aligned} & p^* = e^\epsilon p'^*, && \forall p^* \in [0, \frac{1}{e^{-\epsilon}+1}], p'^* \in [0, \frac{1}{1+e^\epsilon}] \\ \text{or} \quad & 1 - p'^* = e^\epsilon(1 - p^*), && \forall p^* \in [\frac{1}{1+e^{-\epsilon}}, 1], p'^* \in [\frac{1}{1+e^\epsilon}, 1] \end{aligned}$$

We now find the local maximums on the two possible boundaries, i.e.,

$$(p^*_{local}, p'^*_{local}) = \underset{(p,p'):p=e^\epsilon p', p\in[0,\frac{1}{e^{-\epsilon}+1}]}{\text{argmax}} f(p, p'; \gamma_{max})$$

and

$$(p^*_{local}, p'^*_{local}) = \underset{(p,p'):1-p'=e^\epsilon(1-p), p\in[\frac{1}{1+e^{-\epsilon}},1]}{\text{argmax}} f(p, p'; \gamma_{max})$$

separately.

Part I: Local worst case probabilities on the boundary $p = e^{\epsilon}p'$.

Plugging $p = e^{\epsilon}p'$ into the privacy cost objective $f(p, p'; \gamma_{max})$, one gets

$$f(p';\gamma_{max}) = \sum_{l=0}^{\frac{K-1}{2}}(e^{m\epsilon}\binom{K}{l}p'^{l}(1-p')^{K-l} - \binom{K}{l}(e^{\epsilon}p')^{l}(1-e^{\epsilon}p')^{K-l}) \tag{B.36}$$
$$+ \sum_{l=\frac{K+1}{2}}^{K}(\binom{K}{l}(e^{\epsilon}p')^{l}(1-e^{\epsilon}p')^{K-l} - e^{m\epsilon}\binom{K}{l}p'^{l}(1-p')^{K-l})$$

The gradient w.r.t. p' is

$$\begin{aligned}
&\nabla_{p'} f(p';\gamma_{max}) \\
&= \sum_{l=0}^{\frac{K-1}{2}}\Big(e^{m\epsilon}\binom{K}{l}(lp'^{l-1}(1-p')^{K-l} - p'^{l}(K-l)(1-p')^{K-l-1}) \\
&\quad - e^{\epsilon}\binom{K}{l}(l(e^{\epsilon}p')^{l-1}(1-e^{\epsilon}p')^{K-l} - e^{\epsilon l}p'^{l}(K-l)(1-e^{\epsilon}p')^{K-l-1})\Big) \\
&\quad + \sum_{l=\frac{K+1}{2}}^{K}\Big(e^{\epsilon}\binom{K}{l}(l(e^{\epsilon}p')^{l-1}(1-e^{\epsilon}p')^{K-l} - e^{\epsilon l}p'^{l}(K-l)(1-e^{\epsilon}p')^{K-l-1}) \\
&\quad - e^{m\epsilon}\binom{K}{l}(lp'^{l-1}(1-p')^{K-l} - p'^{l}(K-l)(1-p')^{K-l-1})\Big) \\
&= -K\sum_{l=0}^{\frac{K-1}{2}}e^{m\epsilon}\binom{K-1}{l}p'^{l}(1-p')^{K-l-1} + K\sum_{l=\frac{K+1}{2}}^{K-1}e^{m\epsilon}\binom{K-1}{l}p'^{l}(1-p')^{K-l-1} \\
&\quad + K\sum_{l=0}^{\frac{K-1}{2}}e^{\epsilon}\binom{K-1}{l}(\epsilon p')^{\epsilon}(1-e^{\epsilon}p')^{K-l-1} - K\sum_{l=\frac{K+1}{2}}^{K-1}e^{\epsilon}\binom{K-1}{l}(e^{\epsilon}p')^{l}(1-e^{\epsilon}p')^{K-l-1} \\
&\quad + K\sum_{l=0}^{\frac{K-1}{2}-1}e^{m\epsilon}\binom{K-1}{l}p'^{l}(1-p')^{K-l-1} - K\sum_{l=\frac{K-1}{2}}^{K-1}e^{m\epsilon}\binom{K-1}{l}p'^{l}(1-p')^{K-l-1} \\
&\quad - K\sum_{l=0}^{\frac{K-1}{2}-1}e^{\epsilon}\binom{K-1}{l}(e^{\epsilon}p')^{l}(1-e^{\epsilon}p')^{K-l-1} + K\sum_{l=\frac{K-1}{2}}^{K-1}e^{\epsilon}\binom{K-1}{l}(e^{\epsilon}p')^{l}(1-e^{\epsilon}p')^{K-l-1}
\end{aligned}$$

$$= -2K \underbrace{e^{m\epsilon}\binom{K-1}{\frac{K-1}{2}} p'^{\frac{K-1}{2}}(1-p')^{\frac{K-1}{2}}}_{:=A} + 2K \underbrace{e^{\epsilon}\binom{K-1}{\frac{K-1}{2}}(e^{\epsilon}p')^{\frac{K-1}{2}}(1-e^{\epsilon}p')^{\frac{K-1}{2}}}_{:=B}$$

Notice that

$$\frac{A}{B} = \frac{e^{m\epsilon}\binom{K-1}{\frac{K-1}{2}} p'^{\frac{K-1}{2}}(1-p')^{\frac{K-1}{2}}}{e^{\epsilon}\binom{K-1}{\frac{K-1}{2}}(e^{\epsilon}p')^{\frac{K-1}{2}}(1-e^{\epsilon}p')^{\frac{K-1}{2}}} = \frac{e^{m\epsilon}}{e^{\frac{K+1}{2}\epsilon}} \cdot \left(\frac{1-p'}{1-e^{\epsilon}p'}\right)^{\frac{K-1}{2}}$$

Since $\frac{1-p'}{1-e^{\epsilon}p'} \geq 1$ and $m \geq \frac{K+1}{2}$, $\frac{A}{B} \geq 1$. This implies $\nabla_{p'} f(p'; \gamma_{max}) \leq 0$. Hence, $f(p'; \gamma_{max})$ is monotonically non-increasing on the boundary, for $p' \in [0, \frac{1}{1+e^{\epsilon}}]$.

Therefore, $\operatorname{argmax}_{p':p' \in [0, \frac{1}{1+e^{\epsilon}}]} f(p'; \gamma_{max}) = 0$. Since $p = e^{\epsilon}p'$, $p' = 0$ implies $p = 0$.

Hence,

$$(p^*_{local}, p'^*_{local}) = \operatorname*{argmax}_{(p,p'):p=e^{\epsilon}p', p \in [0, \frac{1}{e^{-\epsilon}+1}]} f(p, p'; \gamma_{max}) = (0, 0)$$

and

$$\max_{(p,p'):p=e^{\epsilon}p', p \in [0, \frac{1}{e^{-\epsilon}+1}]} f(p, p'; \gamma_{max}) = f(0, 0; \gamma_{max}) = e^{m\epsilon} - 1$$

Part II: Local worst case probabilities on the boundary $1 - p' = e^{\epsilon}(1-p)$.

For simplicity, let $q = 1 - p$ and $q' = 1 - p'$. Note on this boundary $p \in [\frac{1}{1+e^{-\epsilon}}, 1]$ and $p' \in [\frac{1}{1+e^{\epsilon}}, 1]$, and hence, $q \in [0, \frac{1}{1+e^{\epsilon}}]$ and $q' \in [0, \frac{1}{1+e^{-\epsilon}}]$.

Plugging q and q' into the privacy cost objective $f(p, p'; \gamma_{max})$, one gets a new objective in q, q' as

$$\begin{aligned} f(q, q'; \gamma_{max}) &= \sum_{l=0}^{\frac{K-1}{2}} \left(e^{m\epsilon}\binom{K}{l}(1-q')^l q'^{K-l} - \binom{K}{l}(1-q)^l q^{K-l} \right) \cdot \gamma_{max}(l) \\ &\quad + \sum_{l=\frac{K+1}{2}}^{K} \left(\binom{K}{l}(1-q)^l q^{K-l} - e^{m\epsilon}\binom{K}{l}(1-q')^l q'^{K-l} \right) \cdot \gamma_{max}(l) \\ &= \sum_{l=0}^{\frac{K-1}{2}} \left(e^{m\epsilon}\binom{K}{l}(1-q')^l q'^{K-l} - \binom{K}{l}(1-q)^l q^{K-l} \right) \end{aligned}$$

$$+\sum_{l=\frac{K+1}{2}}^{K}\Big(\binom{K}{l}(1-q)^l q^{K-l}-e^{m\epsilon}\binom{K}{l}(1-q')^l q'^{K-l}\Big)$$

Since on this boundary, $1-p'=e^{\epsilon}(1-p)$, writing this in q,q', this becomes $q'=e^{\epsilon}q$. Plugging $q'=e^{\epsilon}q$ into $f(q,q';\gamma_{max})$, one gets

$$\begin{aligned}f(q;\gamma_{max})=&\sum_{l=0}^{\frac{K-1}{2}}\Big(e^{m\epsilon}\binom{K}{l}(1-e^{\epsilon}q)^l(e^{\epsilon}q)^{K-l}-\binom{K}{l}(1-q)^l q^{K-l}\Big)\\&+\sum_{l=\frac{K+1}{2}}^{K}\Big(\binom{K}{l}(1-q)^l q^{K-l}-e^{m\epsilon}\binom{K}{l}(1-e^{\epsilon}q)^l(e^{\epsilon}q)^{K-l}\Big)\end{aligned}$$

The gradient w.r.t. q is

$$\begin{aligned}&\nabla_q f(q)\\&=\sum_{l=0}^{\frac{K-1}{2}}\Big(e^{m\epsilon}\binom{K}{l}\Big((-e^{\epsilon})l(1-e^{\epsilon}q)^{l-1}(e^{\epsilon}q)^{K-l}+e^{\epsilon}(K-l)(1-e^{\epsilon}q)^l(e^{\epsilon}q)^{K-l-1}\Big)\\&\quad-\binom{K}{l}\Big(-l(1-q)^{l-1}q^{K-l}+(K-l)(1-q)^l q^{K-l-1}\Big)\Big)\\&\quad+\sum_{l=\frac{K+1}{2}}^{K}\Big(\binom{K}{l}\Big(-l(1-q)^{l-1}q^{K-l}+(K-l)(1-q)^l q^{K-l-1}\Big)\\&\quad-e^{m\epsilon}\binom{K}{l}\Big((-e^{\epsilon})l(1-e^{\epsilon}q)^{l-1}(e^{\epsilon}q)^{K-l}+e^{\epsilon}(K-l)(1-e^{\epsilon}q)^l(e^{\epsilon}q)^{K-l-1}\Big)\Big)\\&=-\sum_{l=1}^{\frac{K-1}{2}}e^{(m+1)\epsilon}\binom{K-1}{l-1}\frac{K}{l}l(1-e^{\epsilon}q)^{l-1}(e^{\epsilon}q)^{K-l}\\&\quad+\sum_{l=0}^{\frac{K-1}{2}}e^{(m+1)\epsilon}\binom{K-1}{l}\frac{K}{K-l}(K-l)(1-e^{\epsilon}q)^l(e^{\epsilon}q)^{K-l-1}\\&\quad+\sum_{l=1}^{\frac{K-1}{2}}\binom{K-1}{l-1}\frac{K}{l}l(1-q)^{l-1}q^{K-l}-\sum_{l=0}^{\frac{K-1}{2}}\binom{K-1}{l}\frac{K}{K-l}(K-l)(1-q)^l q^{K-l-1}\end{aligned}\tag{B.37}$$

$$
\begin{aligned}
&- \sum_{l=\frac{K+1}{2}}^{K} \binom{K-1}{l-1} \frac{K}{l} l(1-q)^{l-1} q^{K-l} + \sum_{l=\frac{K+1}{2}}^{K-1} \binom{K-1}{l} \frac{K}{K-l} (K-l)(1-q)^{l} q^{K-l-1} \\
&+ \sum_{l=\frac{K+1}{2}}^{K} e^{(m+1)\epsilon} \binom{K-1}{l-1} \frac{K}{l} l(1-e^{\epsilon}q)^{l-1} (e^{\epsilon}q)^{K-l} \\
&- \sum_{l=\frac{K+1}{2}}^{K-1} e^{(m+1)\epsilon} \binom{K-1}{l} \frac{K}{K-l} (K-l)(1-e^{\epsilon}q)^{l} (e^{\epsilon}q)^{K-l-1} \\
= &-K \sum_{l=1}^{\frac{K-1}{2}} e^{(m+1)\epsilon} \binom{K-1}{l-1} (1-e^{\epsilon}q)^{l-1} (e^{\epsilon}q)^{K-l} \\
&+ K \sum_{l=0}^{\frac{K-1}{2}} e^{(m+1)\epsilon} \binom{K-1}{l} (1-e^{\epsilon}q)^{l} (e^{\epsilon}q)^{K-l-1} \\
&+ K \sum_{l=1}^{\frac{K-1}{2}} \binom{K-1}{l-1} (1-q)^{l-1} q^{K-l} - K \sum_{l=0}^{\frac{K-1}{2}} \binom{K-1}{l} (1-q)^{l} q^{K-l-1} \\
&- K \sum_{l=\frac{K+1}{2}}^{K} \binom{K-1}{l-1} (1-q)^{l-1} q^{K-l} + K \sum_{l=\frac{K+1}{2}}^{K-1} \binom{K-1}{l} (1-q)^{l} q^{K-l-1} \\
&+ K \sum_{l=\frac{K+1}{2}}^{K} e^{(m+1)\epsilon} \binom{K-1}{l-1} (1-e^{\epsilon}q)^{l-1} (e^{\epsilon}q)^{K-l} \\
&- K \sum_{l=\frac{K+1}{2}}^{K-1} e^{(m+1)\epsilon} \binom{K-1}{l} (1-e^{\epsilon}q)^{l} (e^{\epsilon}q)^{K-l-1} \\
= &\, 2K e^{(m+1)\epsilon} \binom{K-1}{\frac{K-1}{2}} (1-e^{\epsilon}q)^{\frac{K-1}{2}} (e^{\epsilon}q)^{\frac{K-1}{2}} - 2K \binom{K-1}{\frac{K-1}{2}} (1-q)^{\frac{K-1}{2}} q^{\frac{K-1}{2}}
\end{aligned}
$$

Recall $q \in [0, \frac{1}{1+e^{\epsilon}}]$ and so $(1-e^{\epsilon}q)(e^{\epsilon}q) \geq (1-q)q$. Furthermore, since $e^{(m+1)\epsilon} \geq 1$, there is $\nabla_q f(q) \geq 0$. This implies $f(q)$ is monotonically non-decreasing in q, and so the local maximum on this boundary is

$$
(q^*_{local}, q'^*_{local}) = \underset{(q,q'):q'=e^{\epsilon}q, q\in[0,\frac{1}{1+e^{\epsilon}}]}{\operatorname{argmax}} f(q, q'; \gamma_{max}) = (\frac{1}{1+e^{\epsilon}}, \frac{1}{1+e^{-\epsilon}})
$$

That is,

$$(p^*_{local}, p'^*_{local}) = \underset{(p,p'):1-p'=e^\epsilon(1-p), p\in[\frac{1}{1+e^{-\epsilon}},1]}{\operatorname{argmax}} f(p,p';\gamma_{max})$$
$$= (1-q^*_{local}, 1-q'^*_{local}) = (\frac{1}{1+e^{-\epsilon}}, \frac{1}{1+e^{\epsilon}})$$

Part III: The global worst case probabilities.

Notice that $(\frac{1}{1+e^{-\epsilon}}, \frac{1}{1+e^{\epsilon}})$, the maximum on the second boundary $1-p' = e^\epsilon(1-p), \forall p \in [\frac{1}{1+e^{-\epsilon}}, 1]$, is indeed the minimum on the first boundary $p = e^\epsilon p', \forall p \in [0, \frac{1}{1+e^{-\epsilon}+1}]$.

Therefore, the global maximum given γ_{max} is

$$(p^*, p'^*) = \underset{(p,p')\in\mathcal{F}}{\operatorname{argmax}} f(p,p';\gamma_{max}) = \underset{(p,p'):p=e^\epsilon p', p\in[0,\frac{1}{1+e^{-\epsilon}}]}{\operatorname{argmax}} f(p,p';\gamma_{max}) = (0,0)$$

and recall that $f(0,0;\gamma_{max}) = e^{m\epsilon} - 1$.

Hence, if $m \geq \frac{K+1}{2}$, by Lemma 3.4.4 $\mathsf{DaRRM}_{\gamma_{max}}$ is $m\epsilon$-differentially private.

□

Privacy Amplification Under A Small Privacy Allowance $m \leq \frac{K-1}{2}$

The proof of Lemma B.2.10 is slightly more involved. First, recall by Lemma 3.4.1, γ_{Sub}, the noise function that makes the output of $\mathsf{DaRRM}_{\gamma_{Sub}}$ and the subsampling baseline the same, is

$$\gamma_{Sub}(l) = \gamma_{Sub}(K=l)$$
$$= \begin{cases} 1 - 2\sum_{j=\frac{m+1}{2}}^{m} \frac{\binom{l}{j}\binom{K-l}{m-j}}{\binom{K}{m}} & \text{if } m \text{ is odd} \\ 1 - 2\sum_{j=\frac{m}{2}+1}^{m} \frac{\binom{l}{j}\binom{K-l}{m-j}}{\binom{K}{m}} - \frac{\binom{l}{\frac{m}{2}}\binom{K-l}{\frac{m}{2}}}{\binom{K}{m}} & \text{if } m \text{ is even} \end{cases}$$

for $l \in \{0,1,\ldots,K\}$, suppose the privacy allowance $m \in \mathbb{Z}$.

If we define $h(l) := \begin{cases} \sum_{j=\frac{m+1}{2}}^{m} \frac{\binom{l}{j}\binom{K-l}{m-j}}{\binom{K}{m}} & \text{if } m \text{ is odd} \\ \sum_{j=\frac{m}{2}+1}^{m} \frac{\binom{l}{j}\binom{K-l}{m-j}}{\binom{K}{m}} - \frac{\binom{l}{\frac{m}{2}}\binom{K-l}{\frac{m}{2}}}{\binom{K}{m}} & \text{if } m \text{ is even} \end{cases}$, then $\gamma_{Sub}(l)$ can

be written as $\gamma_{Sub}(l) = \begin{cases} 1-2h(l) & \text{if } l \leq \frac{K-1}{2} \\ 2h(l)-1 & \text{if } l \geq \frac{K+1}{2} \end{cases}$.

This can be generalized to a broader class of γ functions — which we call the "symmetric form family" — as follows

Definition B.2.6. *$\gamma : \{0, 1, \ldots, K\} \to [0, 1]$ is a member of the "symmetric form family" if γ follows*

$$\gamma(l) = \begin{cases} 1-2h(l) & \text{if } l \leq \frac{K-1}{2} \\ 2h(l)-1 & \text{if } l \geq \frac{K+1}{2} \end{cases} \tag{B.38}$$

where $h : \{0, 1, \ldots, K\} \to [0, 1]$ and

$$h(l) + h(K-l) = 1, \quad h(l+1) \geq h(l), \quad \forall l \in \{0, 1, \ldots, K\},$$
$$\text{and} \quad \gamma(\frac{K-1}{2}) > 0, \gamma(\frac{K+1}{2}) > 0$$

It is easy to verify any γ function that belongs to the "symmetric form family" satisfies: 1) symmetric around $\frac{K}{2}$ and 2) the monotonicity assumption. Hence, Lemma B.2.1 can be invoked to find the worst case probabilities given such γ, i.e., $(p^*, p'^*) = \text{argmax}_{(p,p') \in \mathcal{F}} f(p, p'; \gamma)$, which in turn gives us the guarantee of DaRRM_γ being $m\epsilon$-differentially private.

Roadmap. In this section, we restrict our search of a good γ that maximizes the utility of DaRRM_γ to in the "symmetric form family". To show the main privacy amplification result under a small m in Lemma B.2.10, Section B.2.2, we need a few building blocks, shown in Section B.2.2. We first show in Lemma B.2.7, Section B.2.2 two clean sufficient conditions that if a "symmetric form family" γ satisfies, then DaRRM_γ is $m\epsilon$-differentially private, in terms of the expectation of the γ function applied to Binomial random variables. The Binomial random variables appear in the lemma, because recall the sum of the observed outcomes on a dataset $\mathcal{D}$, $\mathcal{L}(\mathcal{D})$, follows a Binomial distribution in the i.i.d. mechanisms setting. Next, we show a recurrence relationship that connects the expectation of Binomial random variables to Hypergeometric random variables in Lemma B.2.9. This is needed because observe that for γ functions that makes DaRRM_γ have the same output as the majority of subsampled mechanisms, the h function is now a sum of pmfs of the Hypergeometric

random variable.

Finally, the proof of the main result under a small m (Lemma B.2.10) is presented in Section B.2.2, based on Lemma B.2.7 and Lemma B.2.9. We show in Lemma B.2.10 that γ_{DSub}, i.e., the γ function that enables the output of $\mathsf{DaRRM}_{\gamma_{DSub}}$ and outputting the majority of $2m-1$ subsampled mechanisms to be the same, belongs to the "symmetric form family" and satisfies the sufficient conditions as stated in Lemma B.2.7, implying $\mathsf{DaRRM}_{\gamma_{DSub}}$ being $m\epsilon$-differentially private.

Building Blocks

Lemma B.2.7 (Privacy conditions of the "symmetric form family" functions)**.** *Let random variables $X \sim Binomial(K-1, p')$, $Y \sim Binomial(K-1, e^\epsilon p')$, $\hat{X} \sim Binomial(K-1, 1-e^\epsilon(1-p))$ and $\hat{Y} \sim Binomial(K-1, p)$. For a function $\gamma : \{0, 1, \ldots, K\} \to [0, 1]$ that belongs to the "symmetric form family" (Definition B.2.6), if γ also satisfies both conditions as follows:*

$$e^{m\epsilon}\mathbb{E}_X[h(X+1) - h(X)] \geq e^\epsilon \mathbb{E}_Y[h(Y+1) - h(Y)], \quad \forall p' \in [0, \frac{1}{1+e^\epsilon}] \tag{B.39}$$

$$e^{(m+1)\epsilon}\mathbb{E}_{\hat{X}}[h(\hat{X}+1) - h(\hat{X})] \geq \mathbb{E}_{\hat{Y}}[h(\hat{Y}+1) - h(\hat{Y})], \quad \forall p \in [\frac{1}{1+e^{-\epsilon}}, 1] \tag{B.40}$$

then Algorithm DaRRM_γ is $m\epsilon$-differentially private.

Proof of Lemma B.2.7. Since $h(l+1) \geq h(l)$ on $l \in \{0, \ldots, K\}$, $\gamma(l) \geq \gamma(l+1), \forall l \leq \frac{K}{2}$ and $\gamma(l+1) \geq \gamma(l), \forall l \geq \frac{K}{2}$. Furthermore, since $h(l) + h(K-l) = 1$, $\gamma(\frac{K-1}{2}) = 1 - 2h(\frac{K-1}{2}) = 1 - 2(1 - h(\frac{K+1}{2})) = 2h(\frac{K+1}{2}) - 1$. Hence, any γ that belongs to the "symmetric form family" satisfies: 1) symmetric around $\frac{K}{2}$, 2) the monotonicity assumption, and 3) $\gamma(\frac{K-1}{2}) = \gamma(\frac{K+1}{2}) > 0$.

Therefore, by Lemma B.2.1, the worst case probabilities $(p^*, p'^*) = \text{argmax}_{(p,p')\in\mathcal{F}} f(p, p'; \gamma)$ are on one of the two boundaries of $\mathcal{F}$, satisfying

$$p^* = e^\epsilon p'^*, \qquad \forall p^* \in [0, \frac{1}{e^{-\epsilon}+1}], p'^* \in [0, \frac{1}{1+e^\epsilon}] \tag{B.41}$$

$$\text{or} \quad 1 - p'^* = e^\epsilon(1-p^*), \qquad \forall p^* \in [\frac{1}{1+e^{-\epsilon}}, 1], p'^* \in [\frac{1}{1+e^\epsilon}, 1] \tag{B.42}$$

We now derive the sufficient conditions that if any γ from the "symmetric form family" satisfy, then DaRRM_γ is $m\epsilon$-differentially private, from the two boundaries as in Eq. B.41 and Eq. B.42 separately.

Part I: Deriving a sufficient condition from Eq. B.41 for "symmetric form family" γ.

Consider the boundary of $\mathcal{F}$, $p = e^\epsilon p'$, $\forall p \in [0, \frac{1}{1+e^{-\epsilon}}], p' \in [0, \frac{1}{1+e^\epsilon}]$.

Given any γ, plugging $p = e^\epsilon p'$ into the privacy cost objective $f(p, p'; \gamma)$, one gets

$$
\begin{aligned}
f(p';\gamma) = &\sum_{l=0}^{\frac{K-1}{2}} (e^{m\epsilon}\binom{K}{l} p'^l (1-p')^{K-l} - \binom{K}{l}(e^\epsilon p')^l (1-e^\epsilon p')^{K-l}) \cdot \gamma(l) \\
&+ \sum_{l=\frac{K+1}{2}}^{K} (\binom{K}{l}(e^\epsilon p')^l (1-e^\epsilon p')^{K-l} - e^{m\epsilon}\binom{K}{l} p'^l (1-p')^{K-l}) \cdot \gamma(l)
\end{aligned}
$$

The gradient w.r.t. p' is

$$
\begin{aligned}
&\frac{\nabla_{p'} f(p';\gamma)}{K} \\
&= e^{m\epsilon} \sum_{l=0}^{\frac{K-1}{2}-1} \binom{K-1}{l} p'^l (1-p')^{K-l-1} \Big(\gamma(l+1) - \gamma(l)\Big) \\
&\quad - 2e^{m\epsilon} \binom{K-1}{\frac{K-1}{2}} p'^{\frac{K-1}{2}} (1-p')^{\frac{K-1}{2}} \gamma(\frac{K-1}{2}) \\
&\quad + e^{m\epsilon} \sum_{l=\frac{K+1}{2}}^{K-1} \binom{K-1}{l} p'^l (1-p')^{K-l-1} \Big(\gamma(l) - \gamma(l+1)\Big) \\
&\quad + e^{\epsilon} \sum_{l=0}^{\frac{K-1}{2}-1} \binom{K-1}{l} (e^\epsilon p')^l (1-e^\epsilon p')^{K-l-1} \Big(\gamma(l) - \gamma(l+1)\Big) \\
&\quad + 2e^{\epsilon} \binom{K-1}{\frac{K-1}{2}} (e^\epsilon p')^{\frac{K-1}{2}} (1-e^\epsilon p')^{\frac{K-1}{2}} \gamma(\frac{K-1}{2}) \\
&\quad + e^{\epsilon} \sum_{l=\frac{K+1}{2}}^{K-1} \binom{K-1}{l} (e^\epsilon p')^l (1-e^\epsilon p')^{K-l-1} \Big(\gamma(l+1) - \gamma(l)\Big)
\end{aligned}
\tag{B.43}
$$

Consider $l \in \{0, 1, \ldots, K\}$ in the above Eq. B.43. For any function γ that belongs to the "symmetric form family",

1. If $l \leq \frac{K}{2}$, $\gamma(l) - \gamma(l+1) = (1 - 2h(l)) - (1 - 2h(l+1)) = 2h(l+1) - 2h(l)$
2. If $l \geq \frac{K}{2}$, $\gamma(l+1) - \gamma(l) = (2h(l+1) - 1) - (2h(l) - 1) = 2h(l+1) - 2h(l)$
3. Since $\gamma(\frac{K-1}{2}) = \gamma(\frac{K+1}{2})$,

$$\begin{aligned} 2\gamma(\frac{K-1}{2}) &= \Big(\gamma(\frac{K-1}{2}) + \gamma(\frac{K+1}{2})\Big) \\ &= \Big(1 - 2h(\frac{K-1}{2}) + 2h(\frac{K+1}{2}) - 1\Big) \\ &= 2h(\frac{K+1}{2}) - 2h(\frac{K-1}{2}) \end{aligned}$$

Hence, following Eq. B.43, the gradient, $\nabla_{p'} f(p'; \gamma)$, given a "symmetric form family" γ can be written as

$$\begin{aligned} \frac{\nabla_{p'} f(p'; \gamma)}{K} &= -e^{m\epsilon} \sum_{l=0}^{K-1} \binom{K-1}{l} p'^{l} (1-p')^{K-l} \Big(2h(l+1) - 2h(l)\Big) \\ &\quad + e^{\epsilon} \sum_{l=0}^{K-1} \binom{K-1}{l} (e^{\epsilon}p')^{l} (1 - e^{\epsilon}p')^{K-l-1} \Big(2h(l+1) - 2h(l)\Big) \\ &= -2e^{m\epsilon} \mathbb{E}_X[h(X+1) - h(X)] + 2e^{\epsilon} \mathbb{E}_Y[h(Y+1) - h(Y)] \end{aligned}$$

where $X \sim \text{Binomial}(K-1, p')$ and $Y \sim \text{Binomial}(K-1, e^{\epsilon}p')$. The above implies

$$\nabla_{p'} f(p'; \gamma) \leq 0 \iff e^{\epsilon} \mathbb{E}_Y[h(Y+1) - h(Y)] \leq e^{m\epsilon} \mathbb{E}_X[h(X+1) - h(X)] \quad \text{(B.44)}$$

If $\nabla_{p'} f(p'; \gamma) \leq 0$, then we know the local worst case probabilities on the boundary $p = e^{\epsilon}p', \forall p \in [0, \frac{1}{1+e^{-\epsilon}}]$ given any γ is $(p^*_{local}, p'^*_{local}) = \text{argmax}_{(p,p'): p = e^{\epsilon}p', p \in [0, \frac{1}{1+e^{-\epsilon}}]} f(p, p'; \gamma) = (0, 0)$. Furthermore, recall the privacy cost objective given any γ is

$$\begin{aligned} & f(p, p'; \gamma) \\ &= \sum_{l=0}^{\frac{K-1}{2}} (e^{m\epsilon}\alpha'_l - \alpha_l) \cdot \gamma(l) + \sum_{l=\frac{K+1}{2}}^{K} (\alpha_l - e^{m\epsilon}\alpha'_l) \cdot \gamma(l) \\ &= \sum_{l=0}^{\frac{K-1}{2}} \left(e^{m\epsilon} \binom{K}{l} p'^{l} (1-p')^{K-l} - \binom{K}{l} p^{l} (1-p)^{K-l}\right) \cdot \gamma(l) \end{aligned}$$

$$+\sum_{l=\frac{K+1}{2}}^{K}\left(\binom{K}{l}p^l(1-p)^{K-l}-e^{m\epsilon}\binom{K}{l}p'^l(1-p')^{K-l}\right)\cdot\gamma(l)$$

and so for any γ,

$$f(0,0;\gamma)=(e^{m\epsilon}-1)\cdot\gamma(0)\leq e^{m\epsilon}-1 \tag{B.45}$$

Also, notice the local minimum on this boundary is

$$(p_{min},p'_{min})=\underset{(p,p'):p=e^{\epsilon}p',p\in[0,\frac{1}{1+e^{-\epsilon}}]}{\operatorname{argmin}} f(p,p'l;\gamma)=(\frac{1}{1+e^{-\epsilon}},\frac{1}{1+e^{\epsilon}}) \tag{B.46}$$

Part II: Deriving a sufficient condition from Eq. B.42 for "symmetric form family" γ.

Consider the boundary of $\mathcal{F}$, $1-p'=e^{\epsilon}(1-p)$, $\forall p\in[\frac{1}{1+e^{-\epsilon}},1],p'\in[\frac{1}{1+e^{\epsilon}},1]$. For simplicity, let $q=1-p\in[0,\frac{1}{1+e^{\epsilon}}]$ and $q'=1-p'\in[0,\frac{1}{1+e^{-\epsilon}}]$. Plugging $q'=e^{\epsilon}q$ into the privacy cost objective, one gets, given any γ,

$$\begin{aligned}f(q;\gamma)=&\sum_{l=0}^{\frac{K-1}{2}}\left(e^{m\epsilon}\binom{K}{l}(1-e^{\epsilon}q)^l(e^{\epsilon}q)^{K-l}-\binom{K}{l}(1-q)^lq^{K-l}\right)\cdot\gamma(l)\\&+\sum_{l=\frac{K+1}{2}}^{K}\left(\binom{K}{l}(1-q)^lq^{K-l}-e^{m\epsilon}\binom{K}{l}(1-e^{\epsilon}q)^l(e^{\epsilon}q)^{K-l}\right)\cdot\gamma(l)\end{aligned}$$

The gradient w.r.t. q is

$$\begin{aligned}&\frac{\nabla_q f(q;\gamma)}{K}\\&=\sum_{l=0}^{\frac{K-1}{2}-1}e^{(m+1)\epsilon}\binom{K-1}{l}(1-e^{\epsilon}q)^l(e^{\epsilon}q)^{K-l-1}\cdot\Big(\gamma(l)-\gamma(l+1)\Big)\\&+\sum_{l=\frac{K+1}{2}}^{K-1}\binom{K-1}{l}(1-e^{\epsilon}q)^l(e^{\epsilon}q)^{K-l-1}\cdot\Big(\gamma(l+1)-\gamma(l)\Big)\\&+2e^{(m+1)\epsilon}\binom{K-1}{\frac{K-1}{2}}(1-e^{\epsilon}q)^{\frac{K-1}{2}}(e^{\epsilon}q)^{\frac{K-1}{2}}\cdot\gamma(\frac{K-1}{2})\end{aligned}$$

$$+ \sum_{l=0}^{\frac{K-1}{2}-1} \binom{K-1}{l}(1-q)^l q^{K-l-1} \cdot \Big(\gamma(l+1) - \gamma(l)\Big)$$
$$+ \sum_{l=\frac{K+1}{2}}^{K-1} (1-q)^l q^{K-l-1} \cdot \Big(\gamma(l) - \gamma(l+1)\Big) - 2\binom{K-1}{\frac{K-1}{2}}(1-q)^{\frac{K-1}{2}} q^{\frac{K-1}{2}} \cdot \gamma(\frac{K-1}{2})$$

For any function γ that belongs to the "symmetric form family", the gradient $\nabla_q f(q;\gamma)$ can be written as

$$\begin{aligned}\frac{\nabla_q f(q;\gamma)}{K} &= e^{(m+1)\epsilon} \sum_{l=0}^{K-1} \binom{K-1}{l}(1-e^\epsilon q)^l (e^\epsilon q)^{K-l-1} \cdot \Big(2h(l+1) - 2h(l)\Big) \\ &\quad - \sum_{l=0}^{K} \binom{K-1}{l}(1-q)^l q^{K-l-1} \cdot \Big(2h(l+1) - 2h(l)\Big) \\ &= 2e^{(m+1)\epsilon}\mathbb{E}_{\hat{X}}[h(\hat{X}+1) - h(\hat{X})] - 2\mathbb{E}_{\hat{Y}}[h(\hat{Y}+1) - h(\hat{Y})]\end{aligned}$$

where $\hat{X} \sim \text{Binomial}(K-1, 1-e^\epsilon(1-p))$ and $\hat{Y} \sim \text{Binomial}(K-1, p)$. The above implies

$$\nabla_q f(q;\gamma) \geq 0 \iff e^{(m+1)\epsilon}\mathbb{E}_{\hat{X}}[h(\hat{X}+1) - h(\hat{X})] \geq \mathbb{E}_{\hat{Y}}[h(\hat{Y}+1) - h(\hat{Y})] \quad \text{(B.47)}$$

If $\nabla_q f(q;\gamma) \geq 0$, then since $q \in [0, \frac{1}{1+e^\epsilon}]$, we know that the local maximum given any γ is $(q^*_{local}, q'^*_{local}) = \text{argmax}_{(q,q'):q'=e^\epsilon q, q\in[0,\frac{1}{1+e^\epsilon}]} f(q,q';\gamma) = (\frac{1}{1+e^\epsilon}, \frac{1}{1+e^{-\epsilon}})$. That is,

$$\begin{aligned}(p^*_{local}, p'^*_{local}) &= \underset{(p,p'):1-p'=e^\epsilon(1-p), p\in[\frac{1}{1+e^{-\epsilon}},1]}{\text{argmax}} f(p,p';\gamma) \\ &= (1-q^*_{local}, 1-q'^*_{local}) = (\frac{1}{1+e^{-\epsilon}}, \frac{1}{1+e^\epsilon})\end{aligned}$$

Notice by Eq. B.46, the above $(\frac{1}{1+e^{-\epsilon}}, \frac{1}{1+e^\epsilon})$ is the local minimum on the first boundary $p = e^\epsilon p'$, $\forall p \in [0, \frac{1}{1+e^{-\epsilon}}]$.

Therefore, given an arbitrary γ function, if it satisfies both of the following:

1. On the boundary $p = e^\epsilon p', \forall p \in [0, \frac{1}{1+e^{-\epsilon}}]$, $\nabla_{p'} f(p';\gamma) \leq 0$
2. On the boundary $1-p' = e^\epsilon(1-p), \forall p \in [\frac{1}{1+e^{-\epsilon}}, 1]$, $\nabla_{q'} f(q';\gamma) \geq 0$ where $q' = 1-p'$

then the global worst case probabilities given this γ is $(p^*, p'^*) = \text{argmax}_{(p,p')\in\mathcal{F}} f(p, p'; \gamma) = (0, 0)$. Furthermore, since by Eq. B.45, $f(0, 0; \gamma) \leq e^{m\epsilon} - 1$ for any γ, this implies DaRRM_γ is $m\epsilon$-differentially private by Lemma 3.4.4.

Now, if γ belongs to the "symmetric form family", by Eq. B.44 and Eq. B.47, the sufficient conditions for γ that enables DaRRM_γ to be $m\epsilon$-differentially private are hence

$$e^\epsilon \mathbb{E}_Y[h(Y+1) - h(Y)] \leq e^{m\epsilon}\mathbb{E}_X[h(X+1) - h(X)], \quad \forall p' \in [0, \frac{1}{1+e^\epsilon}]$$
$$\text{and} \quad e^{(m+1)\epsilon}\mathbb{E}_{\hat{X}}[h(\hat{X}+1) - h(\hat{X})] \geq \mathbb{E}_{\hat{Y}}[h(\hat{Y}+1) - h(\hat{Y})], \quad \forall p \in [\frac{1}{1+e^{-\epsilon}}, 1]$$

where $X \sim \text{Binomial}(K-1, p')$, $Y \sim \text{Binomial}(K-1, e^\epsilon p')$, $\hat{X} \sim \text{Binomial}(K-1, 1-e^\epsilon(1-p))$ and $\hat{Y} \sim \text{Binomial}(K-1, p)$.

□

Lemma B.2.8 (Binomial Expectation Recurrence Relationship (Theorem 2.1 of [147]))**.** *Let $X_{(K-1)} \sim$ Binomial$(K-1, p)$ and $X_{(K)} \sim$ Binomial(K, p). Let $g(x)$ be a function with $-\infty < \mathbb{E}[g(X_{(K-1)})] < \infty$ and $-\infty < g(-1) < \infty$, then*

$$Kp\mathbb{E}_{X_{(K-1)}}[g(X_{(K-1)})] = \mathbb{E}_{X_{(K)}}[X_{(K)}g(X_{(K)} - 1)] \tag{B.48}$$

Lemma B.2.9. *Given $i, m, K \in \mathbb{Z}$, $K \geq 1$, $0 \leq i \leq m \leq K$, let $X_{(K)} \sim$ Binomial(K, p) for some $p \in [0, 1]$, there is*

$$\frac{1}{\binom{K}{m}}\mathbb{E}_{X_{(K)}}\left[\binom{X}{i}\binom{K-X}{m-i}\right] = \binom{m}{i}p^i(1-p)^{m-i} \tag{B.49}$$

Proof of Lemma B.2.9. We show the above statement in Eq. B.49 by induction on K and m.

Base Case: $K = 1$.

1. If $m = 0$, then $i = 0$. $\frac{1}{\binom{1}{0}}\mathbb{E}_{X_{(1)}}[\binom{X}{0}\binom{1-X}{0}] = \mathbb{E}_{X_{(1)}}[1] = 1$, and $\binom{0}{0}p^0(1-p)^0 = 1$.
2. If $m = 1$,
   (a) $i = 0$, $\frac{1}{\binom{1}{1}}\mathbb{E}_{X_{(1)}}[\binom{X}{0}\binom{1-X}{1}] = \mathbb{E}_{X_{(1)}}[1-X] = 1-p$, and $\binom{1}{0}p^0(1-p)^1 = 1-p$
   (b) $i = 1$, $\frac{1}{\binom{1}{1}}\mathbb{E}_{X_{(1)}}[\binom{X}{1}\binom{1-X}{0}] = \mathbb{E}_{X_{(1)}}[X] = p$, and $\binom{1}{1}p^1(1-p)^0 = p$.

Hence, Eq. B.49 holds for the base case.

Induction Hypothesis: Suppose the statement holds for some $K \geq 1$ and $0 \leq i \leq m \leq K$. Consider $1 \leq i \leq m \leq K+1$,

$$\frac{1}{\binom{K+1}{m}}\mathbb{E}_{X_{(K+1)}}\left[\binom{X}{i}\binom{K+1-X}{m-i}\right]$$
$$= \frac{1}{\binom{K+1}{m}}\mathbb{E}_{X_{(K+1)}}[\frac{X!}{i!(X-i)!}\frac{(K+1-X)!}{(m-i)!(K+1-X-(m-i))!}]$$
$$= \frac{1}{\binom{K+1}{m}i!(m-i)!}\mathbb{E}_{X_{(K+1)}}[X\frac{(X-1)!}{((X-1)-(i-1))!}\frac{(K-(X-1))!}{(K-(X-1)-((m-1)-(i-1)))!}]$$
$$= \frac{1}{\binom{K+1}{m}i!(m-i)!}\mathbb{E}_{X_{(K)}}[\frac{X!}{(X-(i-1))!}\frac{(K-X)!}{(K-X-((m-1)-(i-1)))!}]$$

(By Lemma B.2.8)

$$= \frac{(i-1)!(m-i)!}{\binom{K+1}{m}i!(m-i)!}\mathbb{E}_{X_{(K)}}[\binom{X}{i-1}\binom{K-X}{(m-1)-(i-1)}]$$
$$= \frac{(i-1)!}{\binom{K+1}{m}i!}(K+1)p\binom{K}{m-1}\binom{m-1}{i-1}p^{i-1}(1-p)^{m-i}$$

(By Induction Hypothesis)

$$= \frac{m!(K+1-m)!}{(K+1)!i}\frac{K!}{(m-1)!(K-m+1)!}\frac{(m-1)!}{(i-1)!(m-i)!}(K+1)p^i(1-p)^{m-i}$$
$$= \frac{m!}{i!(m-i)!}p^i(1-p)^{m-i} = \binom{m}{i}p^i(1-p)^{m-i}$$

Now we consider the edge cases when $0 = i \leq m$.

If $i = 0$ and $m = 0$,

$$\frac{1}{\binom{K+1}{0}}\mathbb{E}_{X_{(K+1)}}[\binom{X}{0}\binom{K+1-X}{0}] = 1 \cdot \mathbb{E}_{X_{(K+1)}}[1] = 1 = \binom{0}{0}p^0(1-p)^0$$

If $i = 0$ and $m > 0$,

$$\frac{1}{\binom{K+1}{m}}\mathbb{E}_{X_{(K+1)}}[\binom{K+1-X}{m}]$$

$$= \frac{1}{\binom{K+1}{m}} \sum_{x=0}^{K+1} \binom{K+1-x}{m} \binom{K+1}{x} p^x (1-p)^{K+1-x}$$

$$= \frac{1}{\binom{K+1}{m}} \sum_{x=0}^{K+1} \binom{K+1-x}{m} \left(\binom{K}{x} + \binom{K}{x-1} \mathbb{I}\{x \geq 1\} \right) p^x (1-p)^{K+1-x}$$

$$= \frac{1}{\binom{K+1}{m}} \sum_{x=0}^{K} \binom{K+1-x}{m} \binom{K}{x} p^x (1-p)^{K+1-x}$$

$$+ \frac{1}{\binom{K+1}{m}} \sum_{x=1}^{K+1} \binom{K+1-x}{m} \binom{K}{x-1} p^x (1-p)^{K+1-x}$$

(Since when $x = K+1$ and $m > 0$, $\binom{K+1-x}{m} = 0$)

$$= \frac{1}{\binom{K+1}{m}} \left(\sum_{x=0}^{K} \binom{K-x}{m} \binom{K}{x} p^x (1-p)^{K+1-x} + \sum_{x=0}^{K} \binom{K-x}{m-1} \binom{K}{x} p^x (1-p)^{K+1-x} \right)$$

$$+ \frac{1}{\binom{K+1}{m}} \sum_{x=0}^{K} \binom{K-x}{m} \binom{K}{x} p^{x+1} (1-p)^{K-x}$$

(Since $\binom{K+1-x}{m} = \binom{K-x}{m} + \binom{K-x}{m-1}$)

$$= \frac{1}{\binom{K+1}{m}} \left((1-p) \mathbb{E}_{X_{(K)}} [\binom{K-X}{m}] + (1-p) \mathbb{E}_{X_{(k)}} [\binom{K-X}{m-1}] \right)$$

$$+ \frac{1}{\binom{K+1}{m}} p \mathbb{E}_{X_{(K)}} [\binom{K-X}{m}]$$

$$= \frac{1}{\binom{K+1}{m}} \left(\mathbb{E}_{X_{(K)}} [\binom{K-X}{m}] + (1-p) \mathbb{E}_{X_{(K)}} [\binom{K-X}{m-1}] \right)$$

$$= \frac{1}{\binom{K+1}{m}} \left(\binom{K}{m} (1-p)^m + (1-p) \binom{K}{m-1} (1-p)^{m-1} \right)$$

(By Induction Hypothesis)

$$= \frac{1}{\binom{K+1}{m}} \binom{K+1}{m} (1-p)^m$$

$$= (1-p)^m$$

Hence, Eq. B.49 holds for all $K \geq 1$ and $0 \leq i \leq m \leq K$.

□

Main Result: Privacy Amplification Under a Small m

Lemma B.2.10 (Privacy amplification, $m \leq \frac{K-1}{2}$)**.** *Consider using DaRRM (Algorithm 1) to solve Problem 3.1.1, with i.i.d. mechanisms $\{M_i\}_{i=1}^K$, $p_i = p$, $p_i' = p'$, $\forall i \in [K]$, the privacy allowance $1 \leq m \leq \frac{K-1}{2}, m \in \mathbb{Z}$ and $\delta = \Delta = 0$. Let the noise function be that*

$$\gamma_{DSub}(l) = \begin{cases} 1 - 2h(l) & \forall l \in \{0, 1, \ldots, \frac{K-1}{2}\} \\ 2h(l) - 1 & \forall l \in \{\frac{K+1}{2}, \ldots, K\} \end{cases} \tag{B.50}$$

where $h : \{0, 1, \ldots, K\} \to [0, 1]$ and $h(l) = \sum_{i=m}^{2m-1} \frac{\binom{l}{i}\binom{K-l}{2m-1-i}}{\binom{K}{2m-1}}$, $\forall l \in \{0, 1, \ldots, K\}$, then Algorithm $\mathsf{DaRRM}_{\gamma_{DSub}}$ is $m\epsilon$-differentially private.

Proof of Lemma B.2.10. First, note γ_{DSub} belongs to the "symmetric form family". We show γ_{DSub} satisfies the two sufficient conditions in Lemma B.2.7 and hence by Lemma B.2.7, $\mathsf{DaRRM}_{\gamma_{DSub}}$ is $m\epsilon$-differentially private. Specifically, we consider $h(l) = \sum_{i=m}^{2m-1} \frac{\binom{l}{i}\binom{K-l}{2m-1-i}}{\binom{K}{2m-1}}$, $\forall l \in \{0, 1, \ldots, K\}$ and $1 \leq m \leq K$.

Two show the first condition is satisfied, let $X_{(K-1)} \sim \text{Binomial}(K-1, p)$ and $Y_{(K-1)} \sim \text{Binomial}(K-1, e^\epsilon p)$, and consider $p \in [0, \frac{1}{1+e^\epsilon}]$.

$$\begin{aligned}
&\mathbb{E}_{X_{(K-1)}}[h(X+1)] \\
&= \frac{1}{\binom{K}{2m-1}} \sum_{i=m}^{2m-1} \mathbb{E}_{X_{(K-1)}}[\binom{X+1}{i}\binom{K-X-1}{2m-1-i}] \\
&= \frac{1}{\binom{K}{2m-1}} \sum_{i=m}^{2m-1} \mathbb{E}_{X_{(K-1)}}[\binom{X}{i}\binom{K-X-1}{2m-1-i} + \binom{X}{i-1}\binom{K-X-1}{2m-1-i}] \\
&\text{(Since } \binom{X+1}{i} = \binom{X}{i} + \binom{X}{i-1}\mathbb{I}\{i \geq 1\}) \\
&= \frac{1}{\binom{K}{2m-1}} \sum_{i=m}^{2m-1} \Big(\mathbb{E}_{X_{(K-1)}}[\binom{X}{i}\binom{K-1-X}{2m-1-i}] \\
&\quad + \mathbb{E}_{X_{(K-1)}}[\binom{X}{i-1}\binom{K-1-X}{(2m-2)-(i-1)}]\Big)
\end{aligned}$$

$$= \frac{1}{\binom{K}{2m-1}} \sum_{i=m}^{2m-1} \Big(\binom{K-1}{2m-1} \binom{2m-1}{i} p^i (1-p)^{2m-1-i} + \binom{K-1}{2m-2} \binom{2m-2}{i-1} p^{i-1} (1-p)^{2m-1-i} \Big) \tag{B.51}$$

(By Lemma B.2.9)

and

$$\begin{aligned}
&\mathbb{E}_{X_{(K-1)}}[h(X)] \\
&= \frac{1}{\binom{K}{2m-1}} \sum_{i=m}^{2m-1} \mathbb{E}_{X_{(K-1)}}[\binom{X}{i} \binom{K-X}{2m-1-i}] \\
&(\text{Since } \binom{K-X}{2m-1-i} = \binom{K-1-X}{2m-1-i} + \binom{K-1-X}{2m-2-i}) \\
&= \frac{1}{\binom{K}{2m-1}} \sum_{i=m}^{2m-1} \Big(\mathbb{E}_{X_{(K-1)}}[\binom{X}{i} \binom{K-1-X}{2m-1-i}] \\
&\quad + \mathbb{E}_{X_{(K-1)}}[\binom{X}{i} \binom{K-1-X}{2m-2-i}] \mathbb{I}\{i \le 2m-2\} \Big) \\
&= \frac{1}{\binom{K}{2m-1}} \sum_{i=m}^{2m-1} \Big(\binom{K-1}{2m-1} \binom{2m-1}{i} p^i (1-p)^{2m-1-i} \\
&\quad + \binom{K-1}{2m-2} \binom{2m-2}{i} p^i (1-p)^{2m-2-i} \mathbb{I}\{i \le 2m-2\} \Big)
\end{aligned} \tag{B.52}$$

(By Lemma B.2.9)

Hence, following Eq. B.51 and Eq. B.52,

$$\begin{aligned}
&\mathbb{E}_{X_{(K-1)}}[h(X+1) - h(X)] \\
&= \frac{1}{\binom{K}{2m-1}} \Big(\sum_{i=m}^{2m-1} \binom{K-1}{2m-2} \binom{2m-2}{i-1} p^{i-1} (1-p)^{2m-1-i} \\
&\quad - \sum_{i=m}^{2m-2} \binom{K-1}{2m-2} \binom{2m-2}{i} p^i (1-p)^{2m-2-i} \Big) \\
&= \frac{1}{\binom{K}{2m-1}} \Big(\sum_{i=m-1}^{2m-2} \binom{K-1}{2m-2} \binom{2m-2}{i} p^i (1-p)^{2m-2-i}
\end{aligned}$$

$$
\begin{aligned}
&- \sum_{i=m}^{2m-2} \binom{K-1}{2m-2} \binom{2m-2}{i} p^i (1-p)^{2m-2-i} \Big) \\
&= \frac{2m-1}{K} \binom{2m-2}{m-1} p^{m-1} (1-p)^{m-1}
\end{aligned}
$$

Similarly,

$$
\mathbb{E}_{Y_{(K-1)}}[h(Y+1) - h(Y)] = \frac{2m-1}{K} \binom{2m-2}{m-1} (e^\epsilon p)^{m-1} (1-e^\epsilon p)^{m-1}
$$

Since $p \in [0, \frac{1}{1+e^\epsilon}]$, there is $p(1-p) \geq e^{-\epsilon} e^\epsilon p(1-e^\epsilon p)$. Hence,

$$
\begin{aligned}
e^{(m-1)\epsilon} \mathbb{E}_{X_{(K-1)}}[h(X+1) - h(X)] &= \frac{2m-1}{K} \binom{2m-2}{m-1} e^{(m-1)\epsilon} p^{m-1} (1-p)^{m-1} \\
&\geq \frac{2m-1}{K} \binom{2m-2}{m-1} e^{(m-1)\epsilon} (e^{-\epsilon} e^\epsilon p(1-e^\epsilon p))^{m-1} \\
&= \frac{2m-1}{K} \binom{2m-2}{m-1} (e^\epsilon p)^{m-1} (1-e^\epsilon p)^{m-1} \\
&= \mathbb{E}_{Y_{(K-1)}}[h(Y+1) - h(Y)]
\end{aligned}
$$

implying

$$
e^{m\epsilon} \mathbb{E}_{X_{(K-1)}}[h(X+1) - h(X)] \geq e^\epsilon \mathbb{E}_{Y_{(K-1)}}[h(Y+1) - h(Y)]
$$

and the first condition is satisfied.

To show the second condition is satisfied, let $\hat{X}_{(K-1)} \sim \text{Binom}(K-1, 1-e^\epsilon(1-p))$ and $\hat{Y}_{(K-1)} \sim \text{Binom}(K-1, p)$, and consider $p \in [\frac{1}{1+e^{-\epsilon}}, 1]$.

$$
\begin{aligned}
&\mathbb{E}_{\hat{X}_{(K-1)}}[h(\hat{X}+1)] \\
&= \frac{1}{\binom{K}{2m-1}} \sum_{i=m}^{2m-1} \Big(\mathbb{E}_{\hat{X}_{(K-1)}}[\binom{\hat{X}}{i} \binom{K-1-\hat{X}}{2m-1-i}] \\
&\quad + \mathbb{E}_{\hat{X}_{(K-1)}}[\binom{\hat{X}}{i-1} \binom{K-1-\hat{X}}{(2m-2)-(i-1)}] \Big)
\end{aligned}
$$

$$= \frac{1}{\binom{K}{2m-1}} \sum_{i=m}^{2m-1} \Big(\binom{K-1}{2m-1} \binom{2m-1}{i} (1-e^{\epsilon}(1-p))^{i} (e^{\epsilon}(1-p))^{2m-1-i} \tag{B.53}$$
$$+ \binom{K-1}{2m-2} \binom{2m-2}{i-1} (1-e^{\epsilon}(1-p))^{i-1} (e^{\epsilon}(1-p))^{2m-1-i} \Big)$$

By Lemma B.2.9

and

$$\begin{aligned}
&\mathbb{E}_{\hat{X}_{(K-1)}}[h(\hat{X})] \\
&= \frac{1}{\binom{K}{2m-1}} \sum_{i=m}^{2m-1} \Big(\mathbb{E}_{\hat{X}_{(K-1)}}[\binom{\hat{X}}{i} \binom{K-1-\hat{X}}{2m-1-i}] \\
&\quad + \mathbb{E}_{\hat{X}_{(K-1)}}[\binom{\hat{X}}{i} \binom{K-1-\hat{X}}{2m-2-i}] \mathbb{I}\{i \leq 2m-2\} \Big) \\
&= \frac{1}{\binom{K}{2m-1}} \sum_{i=m}^{2m-1} \Big(\binom{K-1}{2m-1} \binom{2m-1}{i} (1-e^{\epsilon}(1-p))^{i} (e^{\epsilon}(1-p))^{2m-1-i} \qquad \text{(B.54)} \\
&\quad + \binom{K-1}{2m-2} \binom{2m-2}{i} (1-e^{\epsilon}(1-p))^{i} (e^{\epsilon}(1-p))^{2m-2-i} \mathbb{I}\{i \leq 2m-2\} \Big)
\end{aligned}$$

By Lemma B.2.9

Hence, following Eq. B.53 and Eq. B.54,

$$\begin{aligned}
&\mathbb{E}_{\hat{X}_{(K-1)}}[h(\hat{X}+1) - h(\hat{X})] \\
&= \frac{1}{\binom{K}{2m-1}} \Big(\sum_{i=m}^{2m-1} \binom{K-1}{2m-2} \binom{2m-2}{i-1} (1-e^{\epsilon}(1-p))^{i-1} (e^{\epsilon}(1-p))^{2m-1-i} \\
&\quad - \sum_{i=m}^{2m-2} \binom{K-1}{2m-2} \binom{2m-2}{i} (1-e^{\epsilon}(1-p))^{i} (e^{\epsilon}(1-p))^{2m-2-i} \Big) \\
&= \frac{1}{\binom{K}{2m-1}} \Big(\sum_{i=m-1}^{2m-2} \binom{K-1}{2m-2} \binom{2m-2}{i} (1-e^{\epsilon}(1-p))^{i} (e^{\epsilon}(1-p))^{2m-2-i} \\
&\quad - \sum_{i=m}^{2m-2} \binom{K-1}{2m-2} \binom{2m-2}{i} (1-e^{\epsilon}(1-p))^{i} (e^{\epsilon}(1-p))^{2m-2-i} \Big) \\
&= \frac{2m-1}{K} \binom{2m-2}{m-1} (1-e^{\epsilon}(1-p))^{m-1} (e^{\epsilon}(1-p))^{m-1}
\end{aligned}$$

Similarly,

$$\mathbb{E}_{\hat{Y}_{(K-1)}}[h(\hat{Y}+1)-h(\hat{Y})] = \frac{2m-1}{K}\binom{2m-2}{m-1}p^{m-1}(1-p)^{m-1}$$

Hence,

$$\begin{aligned}
&e^{(m+1)\epsilon}\mathbb{E}_{\hat{X}_{(K-1)}}[h(\hat{X}+1)-h(\hat{X})] \\
&= e^{(m+1)\epsilon}\frac{2m-1}{K}\binom{2m-2}{m-1}(1-e^{\epsilon}(1-p))^{m-1}(e^{\epsilon}(1-p))^{m-1} \\
&\geq \frac{2m-1}{K}\binom{2m-2}{m-1}(1-e^{\epsilon}(1-p))^{m-1}e^{(m-1)\epsilon}(1-p)^{m-1} \\
&= \frac{2m-1}{K}\binom{2m-2}{m-1}(e^{\epsilon}-e^{2\epsilon}(1-p))^{m-1}(1-p)^{m-1} \qquad \text{(B.55)}
\end{aligned}$$

Note that

$$\begin{aligned}
&e^{\epsilon}-e^{2\epsilon}(1-p) = e^{\epsilon}-e^{2\epsilon}+e^{2\epsilon}p \geq p \\
\iff\ &(e^{\epsilon}+1)(e^{\epsilon}-1)p \geq e^{\epsilon}(e^{\epsilon}-1) \\
\iff\ &p \geq \frac{e^{\epsilon}}{e^{\epsilon}+1} = \frac{1}{1+e^{-\epsilon}}
\end{aligned}$$

and the condition needs to hold for $p \in [\frac{1}{1+e^{-\epsilon}}, 1]$.

Therefore, following Eq. B.55,

$$\begin{aligned}
e^{(m+1)\epsilon}\mathbb{E}_{\hat{X}_{(K-1)}}[h(\hat{X}+1)-h(\hat{X})] &\geq \frac{2m-1}{K}\binom{2m-2}{m-1}p^{m-1}(1-p)^{m-1} \\
&= \mathbb{E}_{\hat{Y}_{(K-1)}}[h(\hat{Y}+1)-h(\hat{Y})]
\end{aligned}$$

implying the second condition is satisfied.

Therefore, by Lemma B.2.7, $\mathsf{DaRRM}_{\gamma_{DSub}}$ is $m\epsilon$-differentially private.

□

B.2.3 Comparing the Utility of Subsampling Approaches

Intuitively, if we subsample $2m-1$ mechanisms, the utility is higher than that of the naïve subsampling approach which outputs the majority based on only m mechanisms.

To complete the story, we formally compare the utility of outputting the majority of $2m-1$ subsampled mechanisms (Theorem 3.5.1) and outputting the majority of m subsampled mechanisms (simple composition, Theorem 3.3.2) in the i.i.d. mechanisms and pure differential privacy setting, fixing the output privacy loss to be $m\epsilon$.

Lemma B.2.11. *Consider Problem 3.1.1 with i.i.d. mechanisms $\{M_i\}_{i=1}^K$, i.e., $p = p_i = \Pr[M_i(\mathcal{D}) = 1], p' = p'_i = \Pr[M_i(\mathcal{D}') = 1], \forall i \in [K]$. Let $\gamma_1 : \{0, 1, \ldots, K\} \to [0, 1], \gamma_2 : \{0, 1, \ldots, K\} \to [0, 1]$ be two functions that are both symmetric around $\frac{K}{2}$. If $1 \geq \gamma_1(l) \geq \gamma_2(l) \geq 0, \forall l \in \{0, \ldots, K\}$, then $\mathcal{E}(\mathsf{DaRRM}_{\gamma_1}) \leq \mathcal{E}(\mathsf{DaRRM}_{\gamma_2})$.*

Proof. Recall $\mathcal{S} = \{S_1, \ldots, S_K\}$, where $S_i \sim M_i(\mathcal{D})$, is the set of observed outcomes from the mechanisms $\{M_i\}_{i=1}^K$. By Definition 3.3.4, for any γ that is symmetric around $\frac{K}{2}$, the error of DaRRM_γ is

$$\begin{aligned}
\mathcal{E}(\mathsf{DaRRM}_\gamma) &= \Big|\Pr[\mathsf{DaRRM}_\gamma(\mathcal{D}) = 1] - \Pr[g(\mathcal{S}) = 1]\Big| \\
&= \Big|\sum_{l=\frac{K+1}{2}}^{K} \Big(\gamma(l) + \frac{1}{2}(1-\gamma(l))\Big) \cdot \alpha_l + \sum_{l=0}^{\frac{K-1}{2}} \frac{1}{2}(1-\gamma(l)) \cdot \alpha_l - \sum_{l=\frac{K+1}{2}}^{K} \alpha_l\Big| \\
&= \Big|\sum_{l=\frac{K+1}{2}}^{K} \Big(\frac{1}{2}\gamma(l) - \frac{1}{2}\Big) \cdot \alpha_l + \sum_{l=0}^{\frac{K-1}{2}} \Big(\frac{1}{2} - \frac{1}{2}\gamma(l)\Big) \cdot \alpha_l\Big| \\
&= \Big|\frac{1}{2}\sum_{l=\frac{K+1}{2}}^{K} (1-\gamma(l)) \cdot (\alpha_l - \alpha_{K-l})\Big|
\end{aligned}$$

where $\alpha_l = \binom{K}{l} p^l (1-p)^{K-l}$, $\forall l \in \{0, 1, \ldots, K\}$ and recall $p = \Pr[M_i(\mathcal{D}) = 1]$, $\forall i \in [K]$.

For any $l \geq \frac{K+1}{2}$,

1. If $p = 0$ or $p = 1$, $\alpha_l = \alpha_{K-l}$.
2. Otherwise, for $p \in (0, 1)$,
   (a) If $p \geq \frac{1}{2}$,

$$\frac{\alpha_l}{\alpha_{K-l}} = \frac{p^l(1-p)^{K-l}}{p^{K-l}(1-p)^l} = p^{2l-K}(1-p)^{K-2l} = (\underbrace{\frac{p}{1-p}}_{\geq 1})^{\underbrace{2l-K}_{\geq 0}} \geq 1,$$

$$\Rightarrow \alpha_l \geq \alpha_{K-l}$$

(b) If $p < \frac{1}{2}$,

$$\frac{\alpha_l}{\alpha_{K-l}} = (\underbrace{\frac{p}{1-p}}_{\leq 1})^{\underbrace{2l-K}_{\geq 0}} \leq 1, \quad \Rightarrow \alpha_l \leq \alpha_{K-l}$$

Hence, if $p \geq \frac{1}{2}$, then $\alpha_l \geq \alpha_{K-l}, \forall l \geq \frac{K+1}{2}$. Since $\gamma_1(l) \geq \gamma_2(l), \forall l \in \{0, \ldots, K\}$, $1 - \gamma_1(l) \leq 1 - \gamma_2(l)$, and so

$$\begin{aligned}\mathcal{E}(\mathsf{DaRRM}_{\gamma_1}) &= \sum_{l=\frac{K+1}{2}}^{K} \frac{1}{2}(1-\gamma_1(l)) \cdot (\alpha_l - \alpha_{K-l}) \\ &\leq \sum_{l=\frac{K+1}{2}}^{K} \frac{1}{2}(1-\gamma_2(l)) \cdot (\alpha_l - \alpha_{K-l}) = \mathcal{E}(\mathsf{DaRRM}_{\gamma_2})\end{aligned}$$

Similarly, if $p < \frac{1}{2}$, then $\alpha_l \leq \alpha_{K-l}, \forall l \geq \frac{K+1}{2}$ and

$$\begin{aligned}\mathcal{E}(\mathsf{DaRRM}_{\gamma_1}) &= \sum_{l=\frac{K+1}{2}}^{K} \frac{1}{2}(1-\gamma_1(l)) \cdot (\alpha_{K-l} - \alpha_l) \\ &\leq \sum_{l=\frac{K+1}{2}}^{K} \frac{1}{2}(1-\gamma_2(l)) \cdot (\alpha_{K-l} - \alpha_l) = \mathcal{E}(\mathsf{DaRRM}_{\gamma_2}))\end{aligned}$$

Therefore,

$$\mathcal{E}(\mathsf{DaRRM}_{\gamma_1}) \leq \mathcal{E}(\mathsf{DaRRM}_{\gamma_2})$$

□

Since $\gamma_{DSub}(l) \geq \gamma_{Sub}(l)$, $\forall l \in \{0, 1, \ldots, K\}$, by Lemma B.2.11, $\mathcal{E}(\mathsf{DaRRM}_{\gamma_{DSub}}) \leq \mathcal{E}(\mathsf{DaRRM}_{\gamma_{Sub}})$ — that is, outputting $2m-1$ mechanisms has a higher utility than outputting m mechanisms.

B.3 Details of Section 3.6: Optimizing the Noise Function γ in DaRRM

B.3.1 Deriving the Optimization Objective

For any γ function that is symmetric around $\frac{K}{2}$, we can write the optimization objective as

$$
\begin{aligned}
&\mathbb{E}_{p_1,p_2,\ldots,p_K\sim\mathcal{T}}[\mathcal{E}(\mathsf{DaRRM}_\gamma)] \\
&= \mathbb{E}_{p_1,p_2,\ldots,p_K\sim\mathcal{T}}[|\Pr[\mathsf{DaRRM}_\gamma(\mathcal{D})=1]-\Pr[g(\mathcal{S})=1]|] \\
&= \mathbb{E}_{p_1,p_2,\ldots,p_K\sim\mathcal{T}}\left[\left|\sum_{l=\frac{K+1}{2}}^{K}\left(\alpha_l\cdot(\gamma(l)+\frac{1}{2}(1-\gamma(l)))-\alpha_l\right)+\sum_{l=0}^{\frac{K-1}{2}}\alpha_l\cdot\frac{1}{2}(1-\gamma(l))\right|\right] \\
&= \mathbb{E}_{p_1,p_2,\ldots,p_K\sim\mathcal{T}}\left[\left|\sum_{l=0}^{\frac{K-1}{2}}\alpha_l(\frac{1}{2}\gamma(l)-\frac{1}{2})+\sum_{l=\frac{K+1}{2}}^{K}\alpha_l(\frac{1}{2}-\frac{1}{2}\gamma(l))\right|\right]
\end{aligned}
$$

The above follows by conditioning on $\mathcal{L}=l\in\{0,1,\ldots,K\}$, i.e. the sum of observed outcomes in $\mathcal{S}$

$$
= \mathbb{E}_{p_1,p_2,\ldots,p_K\sim\mathcal{T}}\left[\left|\frac{1}{2}\sum_{l=\frac{K+1}{2}}^{K}(\alpha_l-\alpha_{K-l})\,(1-\gamma(l))\right|\right] \tag{B.56}
$$

The above follows by symmetry of γ

Furthermore, notice the objective is symmetric around 0, and can be written as

$$
\mathbb{E}_{p_1,p_2,\ldots,p_K\sim\mathcal{T}}\left[\frac{1}{2}\sum_{l=\frac{K+1}{2}}^{K}(\alpha_l-\alpha_{K-l})\,(1-\gamma(l))\right] \tag{B.57}
$$

$$
= \frac{1}{2}\mathbb{E}_{p_1,p_2,\ldots,p_K\sim\mathcal{T}}\left[\sum_{l=\frac{K+1}{2}}^{K}\left((\alpha_l-\alpha_{K-l})-(\alpha_l-\alpha_{K-l})\gamma(l)\right)\right]
$$

$$= \underbrace{\frac{1}{2}\mathbb{E}_{p_1,p_2,\ldots,p_K\sim\mathcal{T}}\left[\sum_{l=\frac{K+1}{2}}^{K}(\alpha_l - \alpha_{K-l})\right]}_{:=A} \underbrace{-\frac{1}{2}\mathbb{E}_{p_1,p_2,\ldots,p_K\sim\mathcal{T}}\left[\sum_{l=\frac{K+1}{2}}^{K}(\alpha_l - \alpha_{K-l})\gamma(l)\right]}_{:=B} \tag{B.58}$$

Since expression A in Eq. B.58 does not involve γ, we only need to optimize expression B in Eq. B.58. That is,

$$-\frac{1}{2}\mathbb{E}_{p_1,p_2,\ldots,p_K\sim\mathcal{T}}\left[\sum_{l=\frac{K+1}{2}}^{K}(\alpha_l - \alpha_{K-l})\gamma(l)\right] \tag{B.59}$$

$$= -\frac{1}{2}\sum_{l=\frac{K+1}{2}}^{K}\mathbb{E}_{p_1,p_2,\ldots,p_K\sim\mathcal{T}}\left[(\alpha_l - \alpha_{K-l})\right]\cdot\gamma(l) \tag{B.60}$$

Eq. B.60 is the optimization objective we use in the experiments. We see the optimization objective is linear in γ.

Note in the general setting, $\mathcal{L}(\mathcal{D}) \sim \text{PoissonBinomial}(p_1, p_2, \ldots, p_K)$, where recall $\mathcal{L}(\mathcal{D})$ is the sum of observed outcomes on dataset $\mathcal{D}$, and hence, $\alpha_l = \Pr[\mathcal{L}(\mathcal{D}) = l]$ is the pmf of the Poisson Binomial distribution at $l \in \{0, 1, \ldots, K\}$.

B.3.2 Practical Approximation of the Objective

Since the optimization objective in Eq. B.59 requires taking an expectation over $p_1, \ldots, p_K$, and this invovles integrating over K variables, which can be slow in practice, we propose the following approximation to efficiently compute the objective. We start with a simple idea to compute the objective, by sampling p_i's from $[0, 1]$ and take an empirical average of the objective value over all subsampled sets of $p_1, \ldots, p_K$ as the approximation of the expectation in Section B.3.2. However, we found this approach is less numerically stable. We then propose the second approach to approximate the objective in Section B.3.2, which approximates the integration over p_i's using the rectangular rule instead of directly approximating the objective value. We use the second approximation approach in our experiments and empirically demonstrates its effectiveness. **Note approximating the optimization objective does not affect the privacy guarantee.**

Approximation via Direct Sampling of p_i's

One straightforward way of efficiently computing an approximation to the optimization objective is as follows:

Algorithm 9 Straightforward Approximation of the Optimization Objective

Input: # mechanisms $K \in \mathbb{N}$, # iterations $T \in \mathbb{N}$, noise function $\gamma : \{0, 1, \ldots, K\} \to [0, 1]$
for $t = 1, 2, \ldots, T$ **do**
  Sample $\hat{p}_1, \hat{p}_2, \ldots, \hat{p}_K \sim \mathcal{T}$
  $\widehat{\mathcal{L}} \leftarrow \text{PoissonBinomail}(\hat{p}_1, \ldots, \hat{p}_K)$
  $\hat{\alpha}_l \leftarrow \Pr[\widehat{\mathcal{L}} = l], \forall l \{0, \ldots, K\}$
  $g_t \leftarrow -\frac{1}{2} \sum_{l=\frac{K+1}{2}}^{K} (\hat{\alpha}_l - \hat{\alpha}_{K-l}) \cdot \gamma(l)$
end for
Return $\frac{1}{T} \sum_{t=1}^{T} g_t$

However, we found this approximation is not very numerically stable even for $T = 10000$ in the experiments and so we propose to adopt the second approximation as follows.

Approximating the Integration Over p_i's

Consider the following surrogate objective:

$$-\frac{1}{2} \sum_{l=\frac{K+1}{2}}^{K} \int_{0.5}^{1} \int_{0.5}^{1} \cdots \int_{0.5}^{1} (\alpha_l - \alpha_{K-l}) dp_1 dp_2 \ldots dp_K \cdot \gamma(l) \tag{B.61}$$

where we approximate the integration instead of directly approximating the objective value. The approximation of the integration is based on the rectangular rule and that the Poisson Binomial distribution is invariant to the order of its probability parameters.

First, we discretize the integration over p_i's: pick $\tau = 50$ points representing probabilities between $[0.5, 1)$ with equal distance $\theta = \frac{0.5}{\tau}$. Denote this set of points as $\mathcal{W}$. We pick only $\tau = 50$ samples to ensure the distance between each sample, i.e., θ, is not too small; or this can cause numerical instability. For each $l \in$

$\{\frac{K+1}{2}, \frac{K+1}{2}+1, \ldots, K\}$, we want to compute an approximated coefficient for $\gamma(l)$ as follows:

$$\int_{0.5}^{1}\int_{0.5}^{1}\cdots\int_{0.5}^{1}(\alpha_l - \alpha_{K-l})dp_1dp_2\ldots dp_K \approx \sum_{p_1\in\mathcal{W}}\sum_{p_2\in\mathcal{W}}\cdots\sum_{p_K\in\mathcal{W}}(\alpha_l - \alpha_{K-l})$$

which approximates integration over a K-dimensional grid $\mathcal{W}^K$.

The idea is then to sample points from this K-dimensional grid $\mathcal{W}^K$ and compute an empirical mean of the integration based on the sample probabilities for $p_1, \ldots, p_K$ from $\mathcal{W}^K$ as the approximation of the integration in the objective.

Let $(s_1, s_2, \ldots, s_K)$ be randomly sampled probability values from $\mathcal{W}^K$ and we want to compute $(\alpha_l - \alpha_{K-l})$ for all l based on $(p_1, \ldots, p_K) = (s_1, \ldots, s_K)$. To apply the rectangular rule, since the grid of probabilities is K-dimensional, the weight of $(\alpha_l - \alpha_{K-l})$ in the approximate integration is θ^K. Furthermore, observe that α_l is the pmf at l from a Poison Binomial distribution in our case, and PoissonBinomial$(p_1, \ldots, p_K) \overset{dist.}{\sim}$ PoissonBinomial$(\pi(p_1, \ldots, p_K))$, where π denotes a permutation of $p_1, \ldots, p_K$ and $\overset{dist.}{\sim}$ denotes "the same distribution". Hence, with a single probability sample $(s_1, \ldots, s_K)$, we can indeed compute $\alpha_l - \alpha_{K-l}$ for each l at $K!$ points from the grid $\mathcal{W}^K$, since they all have the same value. Therefore, we should set the weight of $\alpha_l - \alpha_{K-l}$ in the approximate integration as $w = \theta^K \cdot K!$. Furthermore, since the order of $(p_1, \ldots, p_K)$ does not affect the objective value, there is a total of (τ choose K with replacement) $= \binom{\tau+K-1}{K} := P$ different points in the grid $\mathcal{W}^K$.

In summary, the integration based approximation of the objective proceeds as in Algorithm 10.

B.3.3 Reducing # Constraints from ∞ to a Polynomial Set

Lemma B.3.1 (Restatement of Lemma 3.6.1)**.** *Consider using DaRRM (Algorithm 1) to solve Problem 3.1.1 and let f be the privacy cost objective as defined in Lemma 3.4.4. Given an arbitrary noise function γ, let the worst case probabilities be*

$$(p_1^*, \ldots, p_K^*, p_1'^*, \ldots, p_K'^*) = \underset{\{(p_i, p_i')\}_{i=1}^K}{\operatorname{argmax}} f(p_1, \ldots, p_K, p_1', \ldots, p_K'; \gamma)$$

Algorithm 10 Integration Based Approximation of the Optimization Objective

Input: # mechanisms $K \in \mathbb{N}$, # iterations $T = 10000 \in \mathbb{N}$, noise function $\gamma : \{0, 1, \ldots, K\} \to [0, 1]$, $\tau = 50$: # samples between $[0.5, 1)$ to form the set $\mathcal{W}$
$\theta \leftarrow 0.5/\tau$ distance between samples
$w \leftarrow \theta^K \cdot K!$
$P \leftarrow \binom{\tau+K-1}{K}$
for $t = 1, 2, \ldots, T$ **do**
  Sample probabilities $(s_1, s_2, \ldots, s_K) \sim \mathcal{W}^K$
  $\widehat{\mathcal{L}} \sim \text{PoissonBinomial}(s_1, s_2, \ldots, s_K)$
  $\hat{\alpha}_l \leftarrow \Pr[\widehat{\mathcal{L}} = l], \forall l \in \{0, 1, \ldots, K\}$
  $g_t \leftarrow -\frac{1}{2} \sum_{l=\frac{K+1}{2}}^{K} w \cdot (\hat{\alpha}_l - \hat{\alpha}_{K-l}) \cdot \gamma(l)$
end for
Return $\frac{P}{N} \sum_{t=1}^{T} g_t$

Then, each pair $(p_i^*, p_i'^*), \forall i \in [K]$ *satisfies*

$$(p_i^*, p_i'^*) \in \{(0,0), (1,1), (0,\Delta), (\Delta,0), (1-\Delta,1), (1,1-\Delta), (\frac{e^\epsilon+\Delta}{e^\epsilon+1}, \frac{1-\Delta}{e^\epsilon+1}), (\frac{1-\Delta}{e^\epsilon+1}, \frac{e^\epsilon+\Delta}{e^\epsilon+1})\}$$

Furthermore, when $\delta > 0$*, there exists a finite vector set* $\mathcal{P}$ *of size* $O(K^7)$ *such that if* $\beta = \max_{\{(p_i, p_i')\}_{i=1}^K \in \mathcal{P}} f(p_1, \ldots, p_K, p_1', \ldots, p_K'; \gamma)$*, then* $f(p_1^*, \ldots, p_K^*, p_1'^*, \ldots, p_K'^*; \gamma) \leq \beta$*. When* $\delta = 0$*, the size of* $\mathcal{P}$ *can be reduced to* $O(K^3)$*.*

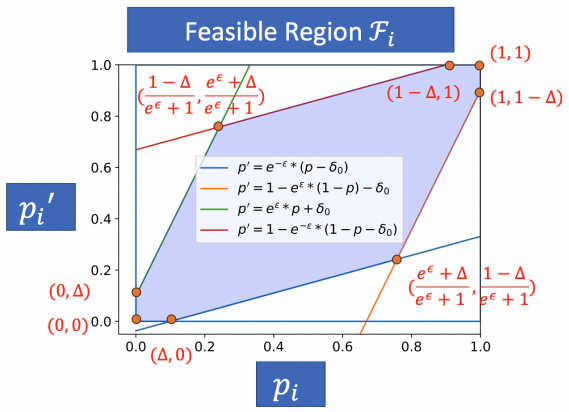

Figure B.3: An illustration of the feasible region $\mathcal{F}_i$.

Proof. **Part I: Reducing # privacy constraints from ∞ to exponentially many.**

Consider (p_i, p_i') for an arbitrary $i \in [K]$ and fixing $(p_j, p_j'), \forall j \neq i$. Given any noise function γ, recall the privacy cost objective $f(p_1, \ldots, p_K, p_1', \ldots, p_K'; \gamma)$ (see Lemma 3.4.4), is

$$f(p_1, \ldots, p_K, p_1', \ldots, p_K'; \gamma) = \sum_{l=0}^{\frac{K-1}{2}} (e^{m\epsilon}\alpha_l' - \alpha_l) \cdot \gamma(l) + \sum_{l=\frac{K+1}{2}}^{K} (\alpha_l - e^{m\epsilon}\alpha_l') \cdot \gamma(l)$$

and the privacy constraints are of the form

$$f(p_1, \ldots, p_K, p_1', \ldots, p_K'; \gamma) \leq e^{m\epsilon} - 1 + 2\delta$$

where recall that $\alpha_l = \Pr[\mathcal{L}(\mathcal{D}) = l]$ is a function of $\{p_i\}_{i=1}^K$ and $\alpha_l' = \Pr[\mathcal{L}(\mathcal{D}') = l]$ is a function of $\{p_i'\}_{i=1}^K$, $\forall l \in \{0, 1, \ldots, K\}$ and $\mathcal{L}(\mathcal{D})$, $\mathcal{L}(\mathcal{D}')$ are the sum of observed outcomes on neighboring datasets $\mathcal{D}$ and $\mathcal{D}'$. By Lemma 3.4.4, γ needs to make the above privacy constraint hold for all possible $\{(p_i, p_i')\}_{i=1}^K$ to make DaRRM_γ $(m\epsilon, \delta)$-differentially private. This is equivalent to saying, γ needs to ensure $\max_{\{(p_i, p_i')\}_{i=1}^K} f(p_1, \ldots, p_K, p_1', \ldots, p_K'; \gamma) \leq e^{m\epsilon} - 1 + 2\delta$.

Notice that the sum of observed outcomes follows a Poisson Binomial distribution, i.e., $\mathcal{L}(\mathcal{D}) \sim \text{PoissonBinomial}(p_1, \ldots, p_K)$ and $\mathcal{L}(\mathcal{D}') \sim \text{PoissonBinomial}(p_1', \ldots, p_K')$. Hence, by the pmf of the Poisson Binomial distribution[1], the privacy cost objective f is linear in each p_i and p_i', fixing all (p_j, p_j'), $\forall j \neq i$. Since each mechanism M_i is (ϵ, Δ)-differentially private, by definition, (p_i, p_i') satisfies all of the following:

$$\begin{aligned} &p_i \leq e^\epsilon p_i' + \Delta, \quad p_i' \leq e^\epsilon p + \Delta \\ &1 - p_i \leq e^\epsilon (1 - p_i') + \Delta, \quad 1 - p_i' \leq e^\epsilon (1 - p_i) + \Delta \end{aligned}$$

That is, (p_i, p_i') lies in a feasible region $\mathcal{F}_i$ (see Figure B.3). Note the constraints on (p_i, p_i'), that is, the boundaries of $\mathcal{F}_i$, are linear in p_i and p_i'. And so the optimization problem $(p_i^*, p_i'^*) = \text{argmax}_{(p_i, p_i')} f(p_1, \ldots, p_K, p_1', \ldots, p_K'; \gamma)$, which finds the worst

[1]See, e.g. https://en.wikipedia.org/wiki/Poisson_binomial_distribution, for the pmf of Poisson Binomial distribution.

case probabilities in (p_i, p_i'), is a Linear Programming (LP) problem in (p_i, p_i') for $i \in [K]$. This implies $(p_i^*, p_i'^*)$ has to be on one of the eight corners of $\mathcal{F}_i$ — that is $(p_i^*, p_i'^*) \in \{(0,0), (1,1), (0,\Delta), (\Delta,0), (1-\Delta,1), (1,1-\Delta), (\frac{e^\epsilon+\Delta}{e^\epsilon+1}, \frac{1-\Delta}{e^\epsilon+1}), (\frac{1-\Delta}{e^\epsilon+1}, \frac{e^\epsilon+\Delta}{e^\epsilon+1})\} := \mathcal{C}$. Since all (p_i, p_i') and (p_j, p_j'), for $i \neq j$, are independent, we can search for the worst case probabilities by searching for $(p_i^*, p_i'^*) \in \mathcal{C}$, instead of searching for $(p_i, p_i') \in \mathcal{F}_i, \forall i \in [K]$. Therefore, the infinitely many privacy constraints are now reduced to only 8^K to optimize for the best γ function that maximizes the utility of DaRRM_γ, while ensuring the output is $m\epsilon$-differentially private.

Part II: Reducing # privacy constraints from exponentially many to a polynomial set.

To further reduce the number of privacy constraints in optimization, observe that the Poisson Binomial distribution is invariant under the permutation of its parameters. That is, PoissonBinomial$(p_1, \ldots, p_K) \overset{dist.}{\sim}$ PoissonBinomial$(\pi(p_1, \ldots, p_K))$, for some permutation π and $\overset{dist.}{\sim}$ means "follows the same distribution". Similarly, PoissonBinomial$(p_1', \ldots, p_K') \overset{dist.}{\sim}$ PoissonBinomial$(\pi(p_1', \ldots, p_K'))$.

The above observation implies if we have one privacy constraint $f(p_1 = v_1, \ldots, p_K = v_K, p_1' = v_1', \ldots, p_K' = v_K'; \gamma) \leq e^{m\epsilon} - 1 + 2\delta$, for some $\{(v_i, v_i')\}_{i=1}^K \in \mathcal{C}^K$, then any privacy constraint $f(p_1 = s_1, \ldots, p_K = s_K, p_1' = s_1', \ldots, p_K' = s_K'; \gamma) \leq e^{m\epsilon} - 1 + 2\delta$, where $(s_1, \ldots, s_K) = \pi_1(v_1, \ldots, v_K)$, $(s_1', \ldots, s_K') = \pi(v_1', \ldots, v_K')$, for permutations π_1 and π_2, is redundant.

Therefore, there is a vector set $\mathcal{P}$, where each probability vector $(p_1, \ldots, p_K, p_1', \ldots, p_K')$ in $\mathcal{P}$ is constructed by setting $(p_1, p_1'), (p_2, p_2'), \ldots, (p_K, p_K') = (v_1, v_2, \ldots, v_K)$, where $v_i \in \mathcal{C}, \forall i \in [K]$, such that vectors constructed by $(p_1, p_1'), (p_2, p_2'), \ldots, (p_K, p_K') = \pi(v_1, v_2, \ldots, v_K)$ is not in $\mathcal{P}$. Note $|\mathcal{P}|$ = (8 chooses K with replacement) = $\binom{K+8-1}{K} = O(K^7)$. If we can restrict our search for the worst case probabilities to this set $\mathcal{P}$ — that is, solving for $\beta := \max_{\{(p_i,p_i')\}_{i=1}^K \in \mathcal{P}} f(p_1, \ldots, p_K, p_1', \ldots, p_K'; \gamma)$, then $f(p_1^*, \ldots, p_K^*, p_1'^*, \ldots, p_K'^*; \gamma) \leq \beta$. This implies we only need $O(K^7)$ privacy constraints to optimize for the best noise function γ in DaRRM, while making sure DaRRM_γ is $m\epsilon$-differentially private.

Note if $\Delta = 0$, i.e., the mechanism M_i's are pure differentially private, the feasible region $\mathcal{F}_i$ in which (p_i, p_i') lies has only 4 corners instead of 8. This implies $(p_i^*, p_i'^*) \in \mathcal{C} = \{(0,0), (1,1), (\frac{e^\epsilon}{e^\epsilon+1}, \frac{1}{e^\epsilon+1}), (\frac{1}{e^\epsilon+1}, \frac{e^\epsilon}{e^\epsilon+1})\}$. Hence, in this case, $|\mathcal{P}|$ = (4 choose K with replacement) = $\binom{K+4-1}{K} = O(K^3)$, which implies we only need $O(K^3)$

privacy constraints to optimize for the best noise function γ in DaRRM.

□

B.4 More Experiment Results

Comparison Using General Composition

The general composition (Theorem 3.3.3) indicates less total privacy loss than simple composition (Theorem 3.3.2) when the number of folds, m, is large, or when the failure probability δ is large. To enable meaningful comparison against general composition, we consider a larger K and a larger failure probability δ.

Consider $K = 35, \epsilon = 0.1, \Delta = 10^{-5}$. By general composition, if one outputs the majority of M subsampled mechanisms for some $M < K$, the majority output is $(\epsilon_{opt}, \delta_{opt})$-differentially private, where

$$\epsilon_{opt} = \min\left\{M\epsilon, \frac{(e^\epsilon - 1)\epsilon M}{e^\epsilon + 1} + \epsilon\sqrt{2M\log(e + \frac{\sqrt{M\epsilon^2}}{\delta'})}, \frac{(e^\epsilon - 1)\epsilon M}{e^\epsilon + 1} + \epsilon\sqrt{2M\log(\frac{1}{\delta'})}\right\},$$
$$\delta_{opt} = 1 - (1-\delta)^M(1-\delta')$$

for some $\delta' \geq 0$. We set this as the privacy guarantee of all majority ensembling algorithms. That is, if we want the majority output to be $(m\epsilon, \delta)$-differentially private, we set

$$m = \frac{\epsilon_{opt}}{\epsilon} = \min\left\{M, \frac{(e^\epsilon - 1)M}{e^\epsilon + 1} + \sqrt{2M\log(e + \frac{\sqrt{M\epsilon^2}}{\delta'})}, \frac{(e^\epsilon - 1)M}{e^\epsilon + 1} + \sqrt{2M\log(\frac{1}{\delta'})}\right\}$$

and $\delta = 1 - (1-\delta)^M(1-\delta')$ accordingly. The parameters τ and λ to compute p_{const} in RR (see Section B.1.1) are set to be

$$\tau = \min\left\{K, \frac{(e^\epsilon - 1)K}{e^\epsilon + 1} + \sqrt{2K\log(e + \frac{\sqrt{K\epsilon^2}}{\delta'})}, \frac{(e^\epsilon - 1)K}{e^\epsilon + 1} + \sqrt{2K\log(\frac{1}{\delta'})}\right\}$$

and $\lambda = 1 - (1-\delta)^K(1-\delta')$.

In the experiments, we consider $M = \{10, 13, 15, 20\}$ and $\delta' = 0.1$; and γ_{opt} is

computed using a uniform prior $\mathcal{T}$.

All values of the parameters of the private ensembling algorithms we use in the experiment are listed in Table B.1. The results are presented in Figure B.4.

# Subsampled mechanisms	M	10	13	15	20
Privacy allowance	m	6.4521	7.5742	8.2708	9.8823
Parameter of constant γ	τ	14.0328	14.0328	14.0328	14.0328
Parameter of constant γ	λ	0.1003	0.1003	0.1003	0.1003
Overall privacy loss	$m\epsilon$	0.6452	0.7574	0.8271	0.9882
Overall failure probability	δ	0.1001	0.1001	0.1001	0.1002

Table B.1: All parameter values. Note that all the private ensembling algorithms we compare in the experiment is required to be $(m\epsilon, \delta)$-differentially private. Here, $K = 35, \epsilon = 0.1, \Delta = 10^{-5}$ and $\delta' = 0.1$.

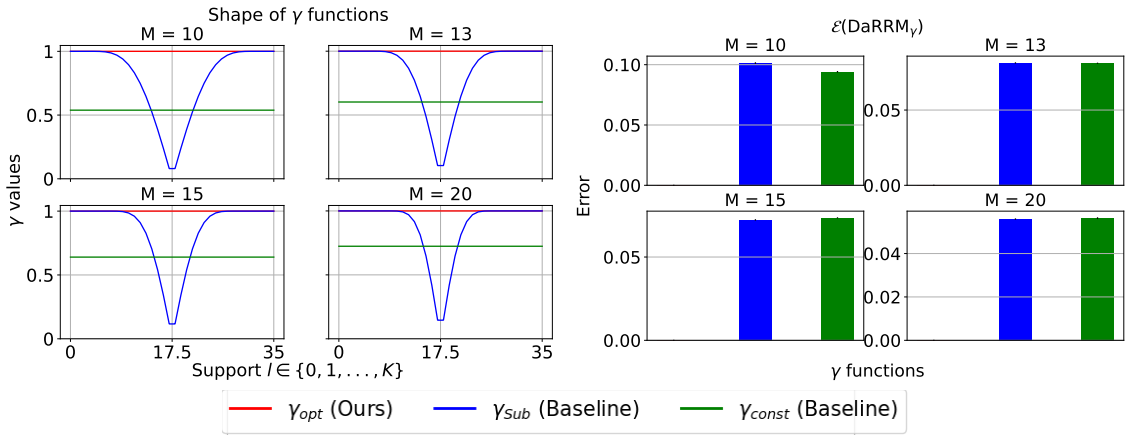

Figure B.4: Plots of the shape and $\mathcal{E}(\mathsf{DaRRM}_\gamma)$ of different γ functions: the optimized γ_{Sub}, and the baselines γ_{Sub} (corresponding to subsampling) and γ_{const} (corresponding to RR). Here, $K = 35, M \in \{10, 13, 15, 20\}$, $\Delta = 10^{-5}$, $\epsilon = 0.1$, $\delta' = 0.1$.

Comparison in Pure Differential Privacy Settings

Consider the pure differential privacy setting, where $\Delta = \delta = 0$. Note in this setting, it is known that simple composition is tight.

To compute an optimized γ_{opt} in DaRRM, since we have shown the number of constraints is $O(K^3)$ if $\Delta = \delta = 0$ (see Lemma 3.6.1), we can set K to be larger. Here, we present results for $K \in \{11, 101\}$ and $\epsilon = 0.1$.

Again, we compare the shape of different γ and the corresponding $\mathcal{E}(\mathsf{DaRRM}_\gamma)$ under those γ functions, fixing the total privacy loss to be $m\epsilon$. γ_{opt} is computed using a uniform prior $\mathcal{T}$.

Since the subsampling mechanism from Section 3.5 with privacy amplification applies to this setting, we compare four different γ noise functions here:

1. γ_{opt} (Ours): optimized γ function using our optimization framework
2. γ_{Sub} (Baseline): the γ function that corresponds to outputting the majority of m out K subsampled mechanisms
3. γ_{DSub} (Baseline): the γ function that corresponds to outputting $2m-1$ subsampled mechanisms from Theorem 3.5.1, aka., Double Subsampling (DSub)
4. γ_{const} (Baseline): the constant γ function that corresponds to the classical Randomized Response (RR) algorithm

Setting 1. $K = 11$, $m \in \{1, 3, 5, 7, 9, 11\}$. The results are presented in Figure B.5.

Setting 2. $K = 101, m \in \{10, 20, 30, 40, 60, 80\}$. The results are presented in Figure B.6.

Comparison Using Different Prior Distributions

When optimizing γ that maximizes the utility in DaRRM, recall that the objective takes an expectation over p_i's for $p_i \sim \mathcal{T}$, where $\mathcal{T}$ is some distribution and $p_i = \Pr[M_i(\mathcal{D}) = 1]$. The previous experiments assume we do not have access to any prior knowledge about p_i's and hence $\mathcal{T}$ is the uniform distribution, i.e., Uniform($[0, 1]$). However, when one has knowledge about the mechanisms, one can set a proper prior $\mathcal{T}$ to further maximize the utility of DaRRM.

In this section, let $\mathcal{T}_U$ denote Uniform($[0, 1]$) and we present results considering a different prior distribution, which we call $\mathcal{T}_P$, as follows. Suppose our prior belief is that each mechanism M_i has a clear tendency towards voting 0 or 1, i.e., p_i is far from 0.5. Let $\mathcal{T}_P$ be Uniform($[0, 0.3] \cup [0.7, 1]$).

To optimize γ under $\mathcal{T}_P$, we change the approximate optimization objective in

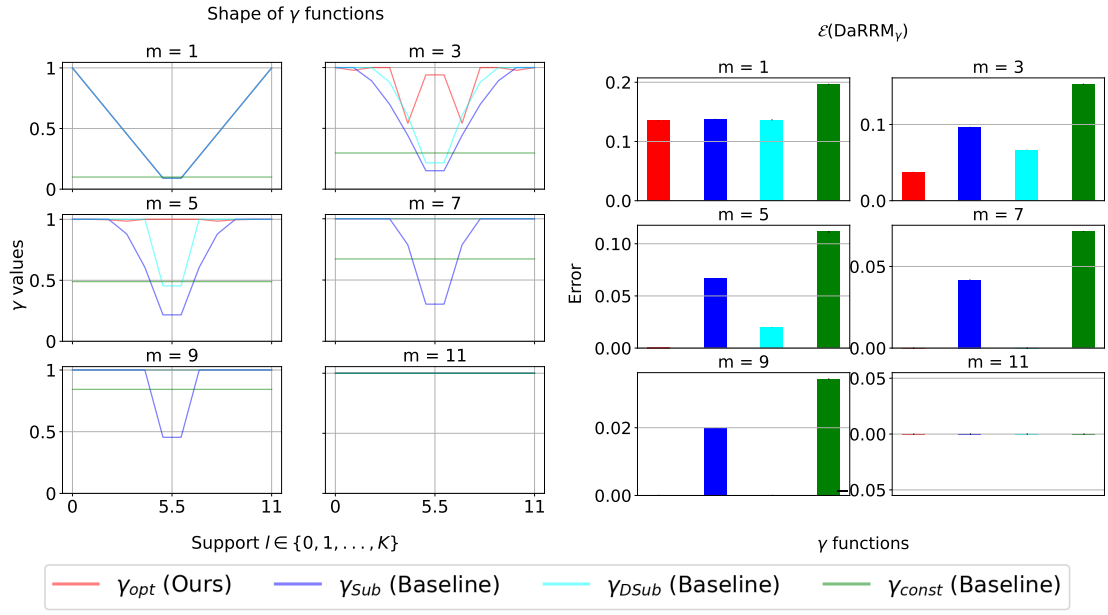

Figure B.5: Plots of shape and $\mathcal{E}(\mathsf{DaRRM}_\gamma)$ of different γ functions: the optimized γ_{Opt}, the baselines γ_{Sub} and γ_{DSub} (Theorem 3.5.1), and the constant γ_{const} (corresponding to RR). Here, $K = 11, m \in \{1, 3, 5, 7, 9, 11\}$, $\epsilon = 0.1$ and $\delta = \Delta = 0$. Note when $m \in \{7, 9\}$, the cyan line (γ_{DSub}) and the red line (γ_{opt}) overlap. When $m = 11$, all lines overlap. Observe that when $m \geq \frac{K+1}{2}$, that is, $m \in \{7, 9, 11\}$ in this case, the above plots suggest both γ_{opt} and γ_{DSub} achieve the minimum error at 0. This is consistent with our theory.

Eq. B.61, which optimizes γ under $\mathcal{T}_U$, to be the following,

$$-\frac{1}{2}\sum_{l=\frac{K+1}{2}}^{K}\int_{0.7}^{1}\int_{0.7}^{1}\cdots\int_{0.7}^{1}(\alpha_l - \alpha_{K-l})dp_1dp_2\ldots dp_K \cdot \gamma(l) \tag{B.62}$$

Setting. $K = 11, m \in \{3, 5\}$, $\epsilon = 0.1$, $\delta = \Delta = 0$.

We compare the shape and $\mathcal{E}(\mathsf{DaRRM}_\gamma)$ of different γ functions:

1. γ_{opt-U} denote the γ function optimized under $p_i \sim \mathcal{T}_U$
2. γ_{opt-P} denote the γ function optimized under $p_i \sim \mathcal{T}_P$
3. γ_{Sub}, corresponding to the subsampling baseline
4. γ_{const}, corresponding to the RR baseline

Note when we compute the error, we take the expectation w.r.t. the actual p_i

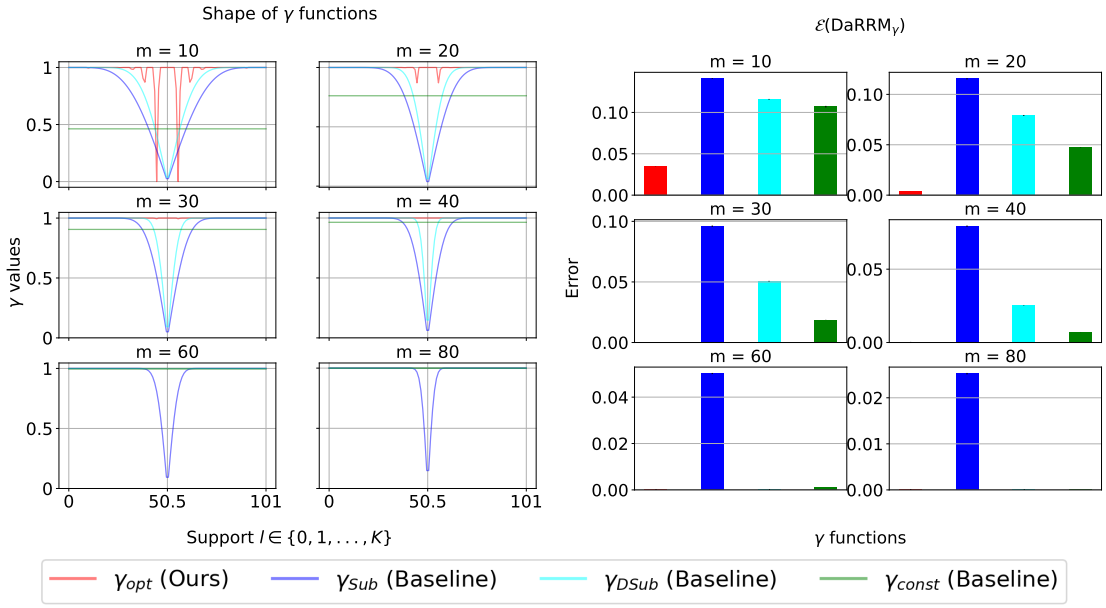

Figure B.6: Plots of shape and $\mathcal{E}(\mathsf{DaRRM}_\gamma)$ of different γ functions: the optimized γ_{Opt}, the baselines γ_{Sub} and γ_{DSub} (Theorem 3.5.1), and the constant γ_{const} (corresponding to RR). Here, $K = 101, m \in \{10, 20, 30, 40, 60, 80\}$, $\epsilon = 0.1$ and $\delta = \Delta = 0$.

distributions, regardless of the prior used to optimize γ. In the experiments, we consider three different actual p_i distributions:"

1. "Actual: Uniform($[0, 1]$)": $p_i \sim \mathcal{T}_U, \forall i \in [K]$
2. "Actual: $p_i = 0.5$": $p_i = 0.5, \forall i \in [K]$

   This setting implies the mechanisms do not have a clear majority
3. "Actual: Uniform($[0, 0.1]$)": $p_i \sim \text{Uniform}([0, 0.1]), \forall i \in [K]$

   This setting implies the mechanisms have a clear majority (i.e., 0)

Since our prior $\mathcal{T}_P$ is closer to Uniform($[0, 0.1]$) (i.e., there is a clear majority), we would expect $\mathcal{E}(\mathsf{DaRRM}_{\gamma_{opt-P}})$ to be the lowest when $p_i \sim \text{Uniform}[0, 0.1]$, but to be higher than $\mathcal{E}(\mathsf{DaRRM}_{\gamma_{opt-U}})$ when $p_i \sim \text{Uniform}([0, 1])$ or $p_i = 0.5$. The results are presented in Figure B.7.

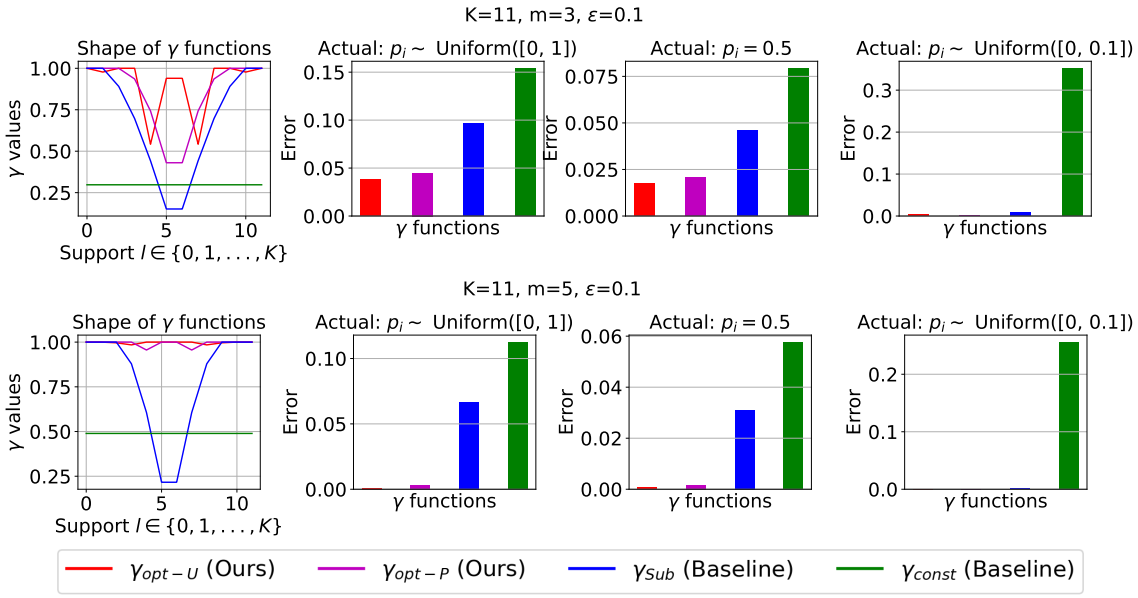

Figure B.7: Comparison of the shape and $\mathcal{E}(\mathsf{DaRRM}_\gamma)$ of different γ functions: 1) γ optimized under prior $\mathcal{T}_U$, 2) γ optimized under prior $\mathcal{T}_P$, 3) γ_{Sub} (corresponding to the subsampling baseline) and 4) γ_{const} (corresponding to the RR baseline). Here, $K = 11, m \in \{3, 5\}, \epsilon = 0.1$. Observe that if the prior $\mathcal{T}_P$ used in optimizing γ is closer to the actual distribution of p_i's, there is additional utility gain (i.e., decreased error); otherwise, we slightly suffer a utility loss (i.e., increased error), compared to optimize γ under the $\mathcal{T}_U$ prior. Furthermore, regardless of the choice of the prior distribution $\mathcal{T}$ in optimizing γ, DaRRM_γ with an optimized γ achieves a lower error compared to the the baselines.

B.4.1 Private Semi-Supervised Knowledge Transfer

More Details about the Baseline GNMax [95]

The GNMax aggregation mechanism for majority ensembling of *non-private* teachers proceeds as follows (Section 4.1 of [95]): on input x,

$$M_\sigma(x) = \underset{i}{\operatorname{argmax}}\{n_i(x) + \mathcal{N}(0, \sigma^2)\}$$

where $n_i(x)$ is # teachers who vote for class i.

How to set σ in GNMax?

Section 4.1 of [95] states the GNMax mechanism is $(\lambda, \lambda/\sigma^2)$-Renyi differentially

private (RDP), for all $\lambda \geq 1$. RDP bounds can be converted to DP bounds as follows:

Theorem B.4.1 (RDP to DP (Theorem 5 of [95])). *If a mechanism M guarantees (λ, ϵ)-RDP, then M guarantees $(\epsilon + \frac{\log 1/\delta}{\lambda - 1}, \delta)$-differential privacy for $\delta \in (0, 1)$.*

Therefore, GNMax with parameter σ^2 guarantees $(\frac{\lambda}{\sigma^2} + \frac{\log 1/\delta}{\lambda - 1}, \delta)$-differential privacy, $\forall \lambda \geq 1$. Given m, ϵ, Δ, we want to choose λ and σ^2 here so that the output of GNMax is $(m\epsilon, m\Delta)$-differentially private. Here, $\delta = m\Delta$.

We first obtain a valid range of λ. Since $m\epsilon \geq 0$, $\frac{\lambda}{\sigma^2} + \frac{\log 1/\delta}{\lambda - 1} \geq 0$ and so $\lambda \geq \frac{\log 1/\delta}{m\epsilon} + 1 := \lambda_{min}$. And $\sigma^2 = \frac{\lambda}{m\epsilon - \frac{\log 1/\delta}{\lambda - 1}}$. Since the smaller σ^2 is, the higher the utility, we perform a grid search over $\lambda \in [\lambda_{min}, 500]$, with discretized λ values of equal distance 0.5, to find the minimum σ^2_{min}. For the $(m\epsilon, m\Delta)$ values used in the experiments, we observe σ^2 decreases first and then increases as λ increases, as shown in Figure B.8. The λ and σ_{min} values in the RDP bound of Gaussian noise to compute the privacy loss of GNMax's output we use in the experiments are presented in Table B.2.

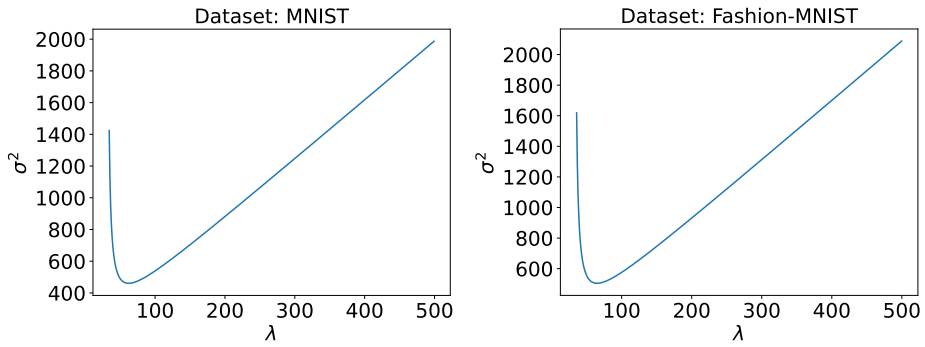

Figure B.8: Plots of λ vs. σ^2 in the Gaussian RDP privacy bound. The goal is to choose a λ value that minimizes σ^2. It is not hard to see the value of σ^2 decreases at first and then increases as λ increases.

	Privacy Loss Per Query $(m\epsilon, m\Delta)$	λ	σ_{min}
MNIST	$(0.2676, 0.0003)$	34.31	21.46
Fashion-MNIST	$(0.2556, 0.0003)$	35.74	22.46

Table B.2: Parameters of the RDP bound of Gaussian noise to compute the privacy loss of GNMax's output.

A Note on the Data-dependent Privacy Loss Bound

[95] gives a potentially tighter data-dependent bound on the privacy loss using GNMax to output the majority of non-private teacherss votes. We give a clean pseudo-code on computing the data-dependent privacy loss bound in Algorithm 11, based on the lemmas and theorems in [95]. Given privacy parameters σ, λ and the teacher votes per class $\{n_i\}_{i=1}^{C}$ for C classes, the data-dependent bound can be empirically evaluated and compared against the Gaussian privacy loss bound. The smaller one is the final privacy loss. We empirically find that the condition of the data-dependent bound (line 8 in Algorithm 11) is not satisfied when K and the number of classes C are small, e.g., $K = 11, C = 2$ as in our case, even if all teachers agree on the same output. And so in the experiments, we can only apply the Gaussian privacy loss bound (line 14).

Algorithm 11 Compute Tighter Privacy Loss

1: Input: Std. of Gaussian noise σ, Privacy parameter λ, # teachers K, # classes C, # votes per class $\{n_i\}_{i=1}^{C}$
2: $\mathcal{B} \leftarrow \{\}$ bound candidates
3: **for** $i = 1, 2, \ldots, K$ **do**
4: $\quad q^{(i)} \leftarrow \frac{1}{2}\sum_{i \neq i^*} \text{erfc}(\frac{n_{i^*} - n_i}{2\sigma})$
5: $\quad \mu_2^{(i)} \leftarrow \sigma \cdot \sqrt{\log 1/q^{(i)}}$, $\mu_1^{(i)} \leftarrow \mu_2^{(i)} + 1$
6: $\quad \epsilon_1^{(i)} \leftarrow \frac{\mu_1^{(i)}}{\sigma^2}$, $\epsilon_2^{(i)} \leftarrow \frac{\mu_2^{(i)}}{\sigma^2}$
7: $\quad q_{ub}^{(i)} \leftarrow \exp((\mu_2^{(i)} - 1)\epsilon_2^{(i)})/(\frac{\mu_1^{(i)}}{\mu_1^{(i)}-1} \cdot \frac{\mu_2^{(i)}}{\mu_2^{(i)}-1})^{\mu_2^{(i)}}$
8: $\quad$ **if** $q^{(i)} < 1$ and $\mu_1^{(i)} \geq \lambda$ and $\mu_2 > 1$ and $q^{(i)} \leq q_{ub}^{(i)}$ **then**
9: $\quad\quad A^{(i)} \leftarrow (1 - q^{(i)})/(1 - q^{(i)} \cdot \exp(\epsilon_2^{(i)})^{\frac{\mu_2^{(i)}-1}{\mu_2^{(i)}}})$
10: $\quad\quad B^{(i)} \leftarrow \exp(\epsilon_1^{(i)})/(q^{(i)})^{\frac{1}{\mu_1^{(i)}-1}}$
11: $\quad\quad \text{DataDependentBound} \leftarrow \frac{1}{\lambda-1} \cdot \left((1 - q^{(i)}) \cdot (A^{(i)})^{\lambda-1} + q^{(i)} \cdot (B^{(i)})^{\lambda-1}\right)$
12: $\quad\quad \mathcal{B} \leftarrow \mathcal{B} \cup \text{DataDependentBound}$
13: $\quad$ **else**
14: $\quad\quad \text{GaussianBound} \leftarrow \frac{\lambda}{\sigma^2}$
15: $\quad\quad \mathcal{B} \leftarrow \mathcal{B} \cup \text{GaussianBound}$
16: $\quad$ **end if**
17: **end for**
18: Return $\min \mathcal{B}$

Additional Results for Private Semi-Supervised Knowledge Transfer

m = 1. The privacy parameters and the results are presented in Table B.3 and B.4, respectively.

Dataset	# Queries	Privacy loss per query $(\epsilon_{query}, \delta_{query})$	Total privacy loss over Q queries $(\epsilon_{total}, \delta_{total})$
MNIST	$Q = 20$	$(0.0892, 0.0001)$	$(1.704, 0.002)$
	$Q = 50$		$(2.837, 0.005)$
	$Q = 100$		$(4.202, 0.010)$
Fashion MNIST	$Q = 20$	$(0.0852, 0.0001)$	$(1.620, 0.002)$
	$Q = 50$		$(2.695, 0.005)$
	$Q = 100$		$(3.988, 0.010)$

Table B.3: The privacy loss per query to the teachers and the total privacy loss over Q queries. Note the total privacy loss is computed by general composition, where we set $\delta' = 0.0001$.

Dataset	MNIST			Dataset	Fashion-MNIST		
# Queries	GNMax (Baseline)	DaRRM$_{\gamma_{Sub}}$ (Baseline)	DaRRM$_{\gamma_{opt}}$ (Ours)	# Queries	GNMax (Baseline)	DaRRM$_{\gamma_{Sub}}$ (Baseline)	DaRRM$_{\gamma_{opt}}$ (Ours)
$Q = 20$	0.54 (0.11)	0.68 (0.07)	**0.74 (0.08)**	$Q = 20$	0.56 (0.10)	**0.92 (0.05)**	0.89 (0.06)
$Q = 50$	0.51 (0.07)	**0.67 (0.05)**	0.66 (0.05)	$Q = 50$	0.52 (0.05)	0.89 (0.04)	**0.92 (0.03)**
$Q = 100$	0.57 (0.03)	**0.71 (0.03)**	0.69 (0.04)	$Q = 100$	0.56 (0.04)	0.89 (0.04)	**0.91 (0.04)**

Table B.4: Accuracy of the predicted labels of Q query samples on datasets MNIST (on the left) and Fashion-MNIST (on the right). We report the mean and one std. in parentheses over 10 random draws of the query samples from the test dataset. Note each prediction on the query sample is $(\epsilon_{total}, \delta_{total})$-differentially private. Note in this case where $m = 1$, by Lemma 3.4.2, subsampling achieves the optimal error/utility. Hence, there is not much difference in terms of accuracy between DaRRM$_{\gamma_{Sub}}$ and DaRRM$_{\gamma_{opt}}$ as expected.

m = 5. The privacy parameters and the results are presented in Table B.5 and B.6, respectively.

Dataset	# Queries	Privacy loss per query $(\epsilon_{query}, \delta_{query})$	Total privacy loss over Q queries $(\epsilon_{total}, \delta_{total})$
MNIST	$Q = 20$	$(0.4460, 0.0005)$	$(8.920, 0.010)$
	$Q = 50$		$(18.428, 0.025)$
	$Q = 100$		$(28.926, 0.049)$
Fashion MNIST	$Q = 20$	$(0.4260, 0.0005)$	$(8.520, 0.010)$
	$Q = 50$		$(17.398, 0.025)$
	$Q = 100$		$(27.223, 0.049)$

Table B.5: The privacy loss per query to the teachers and the total privacy loss over Q queries. Note the total privacy loss is computed by general composition, where we set $\delta' = 0.0001$.

Dataset	MNIST			Dataset	Fashion-MNIST		
# Queries	GNMax (Baseline)	DaRRM$_{\gamma_{Sub}}$ (Baseline)	DaRRM$_{\gamma_{opt}}$ (Ours)	# Queries	GNMax (Baseline)	DaRRM$_{\gamma_{Sub}}$ (Baseline)	DaRRM$_{\gamma_{opt}}$ (Ours)
$Q = 20$	0.73 (0.11)	0.76 (0.09)	**0.84 (0.07)**	$Q = 20$	0.72 (0.10)	0.96 (0.04)	**0.97 (0.04)**
$Q = 50$	0.75 (0.07)	0.82 (0.04)	**0.83 (0.04)**	$Q = 50$	0.72 (0.08)	0.96 (0.02)	**0.97 (0.02)**
$Q = 100$	0.72 (0.04)	0.79 (0.05)	**0.83 (0.03)**	$Q = 100$	0.72 (0.06)	**0.97 (0.01)**	**0.97 (0.01)**

Table B.6: Accuracy of the predicted labels of Q query samples on datasets MNIST (on the left) and Fashion-MNIST (on the right). We report the mean and one std. in parentheses over 10 random draws of the query samples from the test dataset. Note each prediction on the query sample is $(\epsilon_{total}, \delta_{total})$-differentially private. With the same per query privacy loss (and hence the same total privacy loss over Q samples), DaRRM$_{\gamma_{opt}}$ achieves the highest accuracy compared to the other two baselines.

m = 7. The privacy parameters and the results are presented in Table B.7 and B.8, respectively.

Dataset	# Queries	Privacy loss per query $(\epsilon_{query}, \delta_{query})$	Total privacy loss over Q queries $(\epsilon_{total}, \delta_{total})$
MNIST	$Q = 20$	$(0.6244, 0.0007)$	$(12.488, 0.014)$
	$Q = 50$		$(28.392, 0.035)$
	$Q = 100$		$(45.683, 0.068)$
Fashion MNIST	$Q = 20$	$(0.5964, 0.0007)$	$(11.928, 0.014)$
	$Q = 50$		$(26.738, 0.035)$
	$Q = 100$		$(42.873, 0.068)$

Table B.7: The privacy loss per query to the teachers and the total privacy loss over Q queries. Note the total privacy loss is computed by general composition, where we set $\delta' = 0.0001$.

Dataset	MNIST		
# Queries	GNMax (Baseline)	DaRRM$_{\gamma_{Sub}}$ (Baseline)	DaRRM$_{\gamma_{opt}}$ (Ours)
$Q = 20$	0.79 (0.07)	0.80 (0.09)	**0.85 (0.08)**
$Q = 50$	0.80 (0.05)	0.82 (0.05)	**0.85 (0.04)**
$Q = 100$	0.80 (0.04)	0.80 (0.04)	**0.83 (0.03)**

Dataset	Fashion-MNIST		
# Queries	GNMax (Baseline)	DaRRM$_{\gamma_{Sub}}$ (Baseline)	DaRRM$_{\gamma_{opt}}$ (Ours)
$Q = 20$	0.79 (0.07)	0.95 (0.04)	**0.96 (0.04)**
$Q = 50$	0.79 (0.05)	0.96 (0.03)	**0.97 (0.03)**
$Q = 100$	0.79 (0.03)	**0.96 (0.02)**	**0.96 (0.02)**

Table B.8: Accuracy of the predicted labels of Q query samples on datasets MNIST (on the left) and Fashion-MNIST (on the right). We report the mean and one std. in parentheses over 10 random draws of the query samples from the test dataset. Note each prediction on the query sample is $(\epsilon_{total}, \delta_{total})$-differentially private. With the same per query privacy loss (and hence the same total privacy loss over Q samples), DaRRM$_{\gamma_{opt}}$ achieves the highest accuracy compared to the other two baselines.

Appendix C

Differentially Private Shuffled Gradient Methods

C.1 Proof of Theorem 4.4.7

Notation. Before presenting the proof, we summarize key notations to improve readability. Recall that we denote the Bregman divergence induced by a real-valued convex function $g(\mathbf{x}) : \mathbb{R}^d \to \mathbb{R} \cup \{+\infty\}$ as $B_g(\mathbf{x}, \mathbf{y}) = g(\mathbf{x}) - g(\mathbf{y}) - \langle \nabla g(\mathbf{y}), \mathbf{x} - \mathbf{y} \rangle, \forall \mathbf{x}, \mathbf{y} \in \mathbb{R}^d$. We also denote the domain of $g(\mathbf{x})$ by $\mathrm{dom}(g)$.

1. Number of epochs: $K \geq 2$
2. $\mathbb{E}_A[\cdot]$ denotes taking the expectation w.r.t. variable A. When the context is clear, A is omitted.
3. The target objective function:

$$G(\mathbf{x}) = G(\mathbf{x}; \mathcal{D}) = F(\mathbf{x}; \mathcal{D}) + \psi(\mathbf{x}) \tag{C.1}$$

$$\text{where } \mathcal{D} = \{\mathbf{d}_1, \ldots, \mathbf{d}_n\}, \quad F(\mathbf{x}) := F(\mathbf{x}; \mathcal{D}) = \frac{1}{n} \sum_{i=1}^{n} f(\mathbf{x}; \mathbf{d}_i) = \frac{1}{n} \sum_{i=1}^{n} f_i(\mathbf{x})$$

4. The optimum: $\mathbf{x}^* = \mathrm{argmin}_{\mathbf{x} \in \mathbb{R}^d} G(\mathbf{x})$.

   Note that we always care about the convergence of the target objective function, i.e., $\mathbb{E}[G(\mathbf{x}; \mathcal{D})] - \mathbb{E}[G(\mathbf{x}^*; \mathcal{D})]$

5. Optimization uncertainty: $\sigma_{any}^2 = \frac{1}{n}\sum_{i=1}^n \|\nabla f_i(\mathbf{x}^*)\|^2$.
6. Objective function used in the k-th epoch, under permutation $\pi^{(k)} \in \Pi_n$, for $k \in [K]$:

$$G^{(k)}(\mathbf{x}) = G(\mathbf{x}; \mathcal{D}^{(k)} \cup \mathcal{P}^{(k)}) = F(\mathbf{x}; \mathcal{D}^{(k)} \cup \mathcal{P}^{(k)}) + \psi(\mathbf{x}) \tag{C.2}$$

where

- $\mathcal{D}^{(k)} \in \{\emptyset\} \cup \{\{\mathbf{d}^{(k)}_{\pi^{(k)}_1}, \ldots, \mathbf{d}^{(k)}_{\pi^{(k)}_{n_d}}\} : 1 \leq n_d^{(k)} \leq n\}$ is the private dataset used in epoch k, generated by first permuting $\mathcal{D}$ and then taking the first $n_d^{(k)}$ samples
- $\mathcal{P}^{(k)} \in \{\emptyset\} \cup \{\{\mathbf{p}^{(k)}_1, \ldots, \mathbf{p}^{(k)}_{n-n_d^{(k)}}\}\}$, $\mathcal{P}^{(k)} \subseteq \mathcal{P}$, is the public dataset used in epoch k

and

$$\begin{aligned} F^{(k)}(\mathbf{x}) = F(\mathbf{x}; \mathcal{D}^{(k)} \cup \mathcal{P}^{(k)}) &= \frac{1}{n}\Big(\sum_{i=1}^{n_d^{(k)}} f(\mathbf{x}; \mathbf{d}_{\pi_i^{(k)}}) + \sum_{i=n_d^{(k)}+1}^{n} f(\mathbf{x}; \mathbf{p}^{(k)}_{i-n_d^{(k)}})\Big) \\ &= \frac{1}{n}\Big(\sum_{i=1}^{n_d^{(k)}} f_{\pi_i^{(k)}}(\mathbf{x}) + \sum_{i=n_d^{(k)}+1}^{n} f^{(k,pub)}_{i-n_d^{(k)}}(\mathbf{x})\Big) \end{aligned}$$

7. The objective difference for epoch $k \in [K]$:

$$H^{(k)}(\mathbf{x}) = G(\mathbf{x}; \mathcal{D}) - G(\mathbf{x}; \mathcal{D}^{(k)} \cup \mathcal{P}^{(k)})$$

8. Smoothness:

   Assumption C.1.1 (Smoothness (Re-statement of Assumption 4.4.2))**.** *$f(\mathbf{x}; \mathbf{d}_i)$ is L_i-smooth, $\forall i \in [n]$ and $\mathbf{d}_i \in \mathcal{D}$. $f(\mathbf{x}; \mathbf{p}_j^{(k)})$ is $\widetilde{L}_j^{(k)}$-smooth, $\forall j \in [n - n_d^{(k)}]$, $\mathbf{p}_j^{(k)} \in \mathcal{P}$ and $\forall k \in [K]$.*

9. The average smoothness constant
   (a) of the target objective: $L = \frac{1}{n}\sum_{i=1}^n L_i$.
   (b) of the objective used in the k-th epoch: $\widehat{L}^{(k)} = \frac{1}{n}\Big(\sum_{i=1}^{n_d^{(k)}} L_{\pi_i^{(k)}} + \sum_{j=1}^{n-n_d^{(k)}} \widetilde{L}_j^{(k)}\Big)$.

10. The maximum smoothness constant
    (a) of the target objective: $L^* = \max_{i\in[n]}\{L_i\}$.
    (b) of the objective used in the k-th epoch: $\widehat{L}^{(k)*} = \max\{\{L_{\pi_i^{(k)}}\}_{i=1}^{n_d^{(k)}} \cup \{\widetilde{L}_i^{(k)}\}_{i=1}^{n-n_d^{(k)}}\}$
11. The maximum average smoothness constant: $\bar{L}^* = \max\{L, \max_{k\in[K]} \widehat{L}^{(k)}\}$.
12. Lipschitzness (only needed for privacy analysis):
    Assumption C.1.2 (Lipschitz Continuity (Re-statement of Assumption 4.4.3))**.** *A convex function $f : \mathbb{R}^d \to \mathbb{R}$ is G-Lipschitz if $\|\nabla f(\mathbf{x})\| \leq G$. $f(\mathbf{x}, \mathbf{d})$ is G-Lipschitz, $\forall \mathbf{d} \in \mathcal{D}$; $f(\mathbf{x}; \mathbf{p})$ is $\widetilde{G}$-Lipschitz, for all $\mathbf{p} \in \mathcal{P}$.*
13. The maximum Lipschitz parameter: $G^* = \max\{G, \widetilde{G}\}$
14. Gaussian noise applied to the gradient at the i-th step in epoch k, for $i \in [n], k \in [K]$: $\rho_i^{(k)} \sim \mathcal{N}(0, (\sigma^{(k)})^2 \mathbb{I}_d)$

Roadmap. We begin by presenting useful lemmas used in the convergence proof in section C.1.1. After that, we show the one epoch convergence section C.1.2 and the expected one epoch convergence, taking into account the randomness due to data shuffling and noise injection, in section C.1.3. Finally, we give show the convergence bound across K epochs in section C.1.4.

C.1.1 Useful Lemmas

Lemma C.1.3 (Stein's Lemma)**.** *For a zero-mean isotropic Gaussian random variable $\rho \sim \mathcal{N}(0, \sigma^2 \mathbb{I}_d)$, and a differentiable function $h : \mathbb{R}^d \to \mathbb{R}^d$, the following holds:*

$$\mathbb{E}\left[\langle \rho, h(\rho)\rangle\right] = \sigma^2 \mathbb{E}\left[tr(\nabla_\rho h(\rho))\right]$$

where $\nabla h(\rho)$ is the Jacobian matrix of $h(\rho)$ and $tr(\cdot)$ denotes the trace operator.

Lemma C.1.4 (Lemma 3.6 of [76])**.** *Given a convex and differentiable function $g(\mathbf{x}) : \mathbb{R}^d \to \mathbb{R}$ satisfying $\|\nabla g(\mathbf{x}) - \nabla g(\mathbf{y})\| \leq L\|\mathbf{x} - \mathbf{y}\|$, $\forall \mathbf{x}, \mathbf{y} \in \mathbb{R}^d$ for some $L > 0$, then $\forall \mathbf{x}, \mathbf{y} \in \mathbb{R}^d$,*

$$\frac{\|\nabla g(\mathbf{x}) - \nabla g(\mathbf{y})\|^2}{2L} \leq B_g(\mathbf{x}, \mathbf{y}) \leq \frac{L}{2}\|\mathbf{x} - \mathbf{y}\|^2$$

Lemma C.1.5 (Lemma E.1 of [76])**.** *Under Assumption C.1.1, for any permutation*

π *of* $[n]$,

$$\frac{1}{n}\sum_{i=2}^{n} L_i \left\| \sum_{j=1}^{i-1} \nabla f_j(\mathbf{x}^*) \right\|^2 \leq n^2 L \sigma_{any}^2,$$

where $L = \frac{1}{n}\sum_{i=1}^{n} L_i$.

Lemma C.1.6 (Extension of Lemma 6.2 of [76])**.** *Given two sequences of reals:* $d^{(1)}, d^{(2)}, \ldots, d^{(K)}, d^{(K+1)}$ *and* $e^{(1)}, e^{(2)}, \ldots, e^{(K)}$*, suppose there exist positive constants* a, b, c *satisfying*

$$d^{(k+1)} \leq \frac{a}{k} + b(1 + \log k) + c\sum_{l=2}^{k} \frac{d^{(l)}}{k-l+2} + \sum_{l=1}^{k} \frac{e^{(l)}}{k-l+1}, \quad \forall k \in [K] \tag{C.3}$$

then the following inequality holds

$$d^{(k+1)} \leq \left(\frac{a}{k} + b(1 + \log k) + M\right) \sum_{i=0}^{k-1} (2c(1 + \log k))^i \tag{C.4}$$

where $M := \max_{k\in[K]} \sum_{l=1}^{k} \frac{e^{(l)}}{k-l+1}$.

Proof. We use induction to show Eq. C.4.

Base Case: for $k = 1$, by Eq. C.3 and the definition of M, $d^{(2)} \leq a + b + e^{(1)} \leq a + b + M$, which also satisfies Eq. C.4.

Induction Hypothesis: suppose Eq. C.4 holds for 1 to $k-1$ (where $2 \leq k \leq K$), i.e.,

$$d^{(l)} \leq \left(\frac{a}{l-1} + b(1 + \log(l-1)) + M\right) \sum_{i=0}^{l-2} (2c(1 + \log(l-1)))^i$$

which implies

$$d^{(l)} \leq \left(\frac{a}{l-1} + b(1 + \log k) + M\right) \sum_{i=0}^{l-2} (2c(1 + \log k))^i$$

Now for $d^{(k+1)}$, by Eq. C.3,

$$\begin{aligned} d^{(k+1)} &\le \frac{a}{k} + b(1+\log k) + c\sum_{l=2}^{k}\frac{d^{(l)}}{k-l+2} + \sum_{l=1}^{k}\frac{e^{(l)}}{k-l+1} \\ &\le \frac{a}{k} + b(1+\log k) + c\sum_{l=2}^{k}\frac{d^{(l)}}{k-l+2} + M \\ &\le \frac{a}{k} + ac\sum_{l=2}^{k}\sum_{i=0}^{i-2}\frac{(2c(1+\log k))^i}{(k-l+2)(l-1)} \\ &\quad + \Big(b(1+\log k) + M\Big)\Big(1 + c\sum_{l=2}^{k}\sum_{i=0}^{l-2}\frac{(2c(1+\log k))^i}{k-l+2}\Big) \end{aligned} \tag{C.5}$$

Note that

$$\begin{aligned} c\sum_{l=2}^{k}\sum_{i=0}^{i-2}\frac{(2c(1+\log k))^i}{(k-l+2)(l-1)} &= c\sum_{i=0}^{k-2}(2c(1+\log k))^i\Big(\sum_{l=2+i}^{k}\frac{1}{(k-l+2)(l-1)}\Big) \\ &= \frac{c}{k+1}\sum_{i=0}^{k-2}(2c(1+\log k))^i\Big(\sum_{l=2+i}^{k}\frac{1}{k-l+2} + \frac{1}{l-1}\Big) \\ &\le \frac{c}{k+1}\sum_{i=0}^{k-2}(2c(1+\log k))^i\sum_{l=1}^{k}\frac{2}{l} \\ &\le \frac{\sum_{i=0}^{k-2}(2c(1+\log k))^{i+1}}{k+1} \\ &\le \frac{\sum_{i=0}^{k-1}(2c(1+\log k))^{i}}{k+1} \\ &\le \frac{\sum_{i=1}^{k-1}(2c(1+\log k))^{i}}{k} \end{aligned} \tag{C.6}$$

and

$$\begin{aligned} c\sum_{l=2}^{k}\sum_{i=0}^{l-2}\frac{(2c(1+\log k))^i}{k-l+2} &= c\sum_{i=0}^{k-2}(2c(1+\log k))^i\sum_{l=2+i}^{k}\frac{1}{k-l+2} \\ &\le c\sum_{i=0}^{k-2}(2c(1+\log k))^i\sum_{l=1}^{k}\frac{1}{l} \end{aligned}$$

$$\leq c(1+\log k)\sum_{i=0}^{k-2}(2c(1+\log k))^i \leq \sum_{i=0}^{k-2}(2c(1+\log k))^{i+1}$$
$$\leq \sum_{i=1}^{k-1}(2c(1+\log k))^i \tag{C.7}$$

Combining Eq. C.5, Eq. C.6 and Eq. C.7,

$$d^{(k+1)} \leq \frac{a}{k} + \frac{a}{k}\sum_{i=1}^{k-1}(2c(1+\log k))^i + \Big(b(1+\log k)+M\Big)\Big(1+\sum_{i=1}^{k-1}(2c(1+\log k))^i\Big)$$
$$= \Big(\frac{a}{k} + b(1+\log k) + M\Big)\sum_{i=0}^{k-1}(2c(1+\log k))^i$$

which finishes the induction.

□

C.1.2 One Epoch Convergence

The following lemma is a generalization of Lemma D.1 of [76] from two dimensions: allowing the usage of surrogate objectives and adding additional noise for privacy preservation.

Lemma C.1.7. *Under Assumptions 4.4.1 and 4.4.4, for any epoch $k \in [K]$, permutation $\pi^{(k)}$ and $\mathbf{z} \in \mathbb{R}^d$, Algorithm 3 guarantees*

$$G(\mathbf{x}_1^{(k+1)}) - G(\mathbf{z}) \leq H^{(k)}(\mathbf{x}_1^{(k+1)}) - H^{(k)}(\mathbf{z}) + \frac{\|\mathbf{z}-\mathbf{x}_1^{(k)}\|^2}{2n\eta}$$
$$- (\frac{1}{2n\eta} + \frac{\mu_\psi}{2})\|\mathbf{z}-\mathbf{x}_1^{(k+1)}\|^2 - \frac{1}{2n\eta}\|\mathbf{x}_1^{(k+1)}-\mathbf{x}_1^{(k)}\|^2$$
$$+ \frac{1}{n}\Big(\sum_{i=1}^{n_d^{(k)}}\Big(B_{f_{\pi_i^{(k)}}}(\mathbf{x}_1^{(k+1)},\mathbf{x}_i^{(k)}) - B_{f_{\pi_i^{(k)}}}(\mathbf{z},\mathbf{x}_i^{(k)})\Big)$$
$$+ \sum_{i=n_d^{(k)}+1}^{n}\Big(B_{f^{(k,pub)}_{i-n_d^{(k)}}}(\mathbf{x}_1^{(k+1)},\mathbf{x}_i^{(k)}) - B_{f^{(k,pub)}_{i-n_d^{(k)}}}(\mathbf{z},\mathbf{x}_i^{(k)})\Big)\Big) + \frac{1}{n}\sum_{i=1}^{n}\langle -\rho_i^{(k)}, \mathbf{x}_1^{(k+1)}-\mathbf{z}\rangle.$$

Proof of Lemma C.1.7. It suffices to only consider $\mathbf{x} \in \mathrm{dom}(\psi)$.

Let $\mathbf{g}^{(k)} = \sum_{i=1}^{n_d^{(k)}} \left(\nabla f_{\pi_i^{(k)}}(\mathbf{x}_i^{(k)}) + \rho_i^{(k)}\right) + \sum_{i=n_d^{(k)}+1}^{n} \left(\nabla f_{i-n_d^{(k)}}^{(k,pub)}(\mathbf{x}_i^{(k)}) + \rho_i^{(k)}\right)$.

According to the update rule in 3, $\mathbf{x}_{n+1}^{(k)} = \mathbf{x}_1^{(k)} - \eta \cdot \mathbf{g}^{(k)}$. Observe that

$$\begin{aligned}
\mathbf{x}_1^{(k+1)} &= \operatorname*{argmin}_{\mathbf{x}\in\mathbb{R}^d} \left\{ n\psi(\mathbf{x}) + \frac{\|\mathbf{x} - \mathbf{x}_{n+1}^{(k)}\|^2}{2\eta} \right\} = \operatorname*{argmin}_{\mathbf{x}\in\mathbb{R}^d} \left\{ n\psi(\mathbf{x}) + \frac{\|\mathbf{x} - \mathbf{x}_1^{(k)} + \eta \cdot \mathbf{g}^{(k)}\|^2}{2\eta} \right\} \\
&= \operatorname*{argmin}_{\mathbf{x}\in\mathbb{R}^d} \left\{ n\psi(\mathbf{x}) + \frac{\|\mathbf{x} - \mathbf{x}_1^{(k)}\|^2 + \eta^2\|\mathbf{g}^{(k)}\|^2 + 2\langle \mathbf{x} - \mathbf{x}_1^{(k)}, \eta\mathbf{g}^{(k)}\rangle}{2\eta} \right\} \\
&= \operatorname*{argmin}_{\mathbf{x}\in\mathbb{R}^d} \left\{ n\psi(\mathbf{x}) + \frac{\|\mathbf{x} - \mathbf{x}_1^{(k)}\|^2}{2\eta} + \langle \mathbf{x} - \mathbf{x}_1^{(k)}, \mathbf{g}^{(k)}\rangle \right\}
\end{aligned}$$

By the first-order optimality condition, there exists some vector $\nabla\psi(\mathbf{x}_1^{(k+1)})$ in the subgradient of $\psi(\mathbf{x}_1^{(k+1)})$ such that

$$n\nabla\psi(\mathbf{x}_1^{(k+1)}) + \mathbf{g}^{(k)} + \frac{\mathbf{x}_1^{(k+1)} - \mathbf{x}_1^{(k)}}{\eta} = \mathbf{0} \iff \mathbf{g}^{(k)} = -n\nabla\psi(\mathbf{x}_1^{(k+1)}) + \frac{\mathbf{x}_1^{(k)} - \mathbf{x}_1^{(k+1)}}{\eta}$$

Therefore, for $\mathbf{z} \in \mathrm{dom}(\psi)$,

$$\begin{aligned}
&\langle \mathbf{g}^{(k)}, \mathbf{x}_1^{(k+1)} - \mathbf{z}\rangle \\
&= n\langle \nabla\psi(\mathbf{x}_1^{(k+1)}), \mathbf{z} - \mathbf{x}_1^{(k+1)}\rangle + \frac{1}{\eta}\langle \mathbf{x}_1^{(k)} - \mathbf{x}_1^{(k+1)}, \mathbf{x}_1^{(k+1)} - \mathbf{z}\rangle \\
&\overset{(a)}{\leq} n\left(\psi(\mathbf{z}) - \psi(\mathbf{x}_1^{(k+1)}) - \frac{\mu_\psi}{2}\|\mathbf{z} - \mathbf{x}_1^{(k+1)}\|^2\right) + \frac{1}{\eta}\langle \mathbf{x}_1^{(k)} - \mathbf{x}_1^{(k+1)}, \mathbf{x}_1^{(k+1)} - \mathbf{z}\rangle \\
&= n\left(\psi(\mathbf{z}) - \psi(\mathbf{x}_1^{(k+1)}) - \frac{\mu_\psi}{2}\|\mathbf{z} - \mathbf{x}_1^{(k+1)}\|^2\right) \\
&\quad + \frac{1}{2\eta}\left(\|\mathbf{z} - \mathbf{x}_1^{(k)}\|^2 - \|\mathbf{z} - \mathbf{x}_1^{(k+1)}\|^2 - \|\mathbf{x}_1^{(k+1)} - \mathbf{x}_1^{(k)}\|^2\right) \\
&= n\left(\psi(\mathbf{z}) - \psi(\mathbf{x}_1^{(k+1)})\right) + \frac{\|\mathbf{z} - \mathbf{x}_1^{(k)}\|^2}{2\eta} \\
&\quad - (\frac{1}{2\eta} + \frac{n\mu_\psi}{2})\|\mathbf{z} - \mathbf{x}_1^{(k+1)}\|^2 - \frac{1}{2\eta}\|\mathbf{x}_1^{(k+1)} - \mathbf{x}_1^{(k)}\|^2
\end{aligned} \tag{C.8}$$

where (a) is by Assumption 4.4.4 on the μ_ψ-strong convexity of ψ.

By the definition of $\mathbf{g}^{(k)}$,

$$\langle \mathbf{g}^{(k)}, \mathbf{x}_1^{(k+1)} - \mathbf{z}\rangle$$

$$
\begin{aligned}
&= \langle \sum_{i=1}^{n_d^{(k)}} \Big(\nabla f_{\pi_i^{(k)}}(\mathbf{x}_i^{(k)}) + \rho_i^{(k)} \Big) + \sum_{i=n_d^{(k)}+1}^{n} \Big(\nabla f_{i-n_d^{(k)}}^{(k,pub)}(\mathbf{x}_i^{(k)}) + \rho_i^{(k)} \Big), \mathbf{x}_1^{(k+1)} - \mathbf{z} \rangle \\
&= \sum_{i=1}^{n_d^{(k)}} \langle \nabla f_{\pi_i^{(k)}}(\mathbf{x}_i^{(k)}), \mathbf{x}_1^{(k+1)} - \mathbf{z} \rangle + \sum_{i=n_d^{(k)}+1}^{n} \langle \nabla f_{i-n_d^{(k)}}^{(k,pub)}(\mathbf{x}_i^{(k)}), \mathbf{x}_1^{(k+1)} - \mathbf{z} \rangle \qquad \text{(C.9)} \\
&\quad + \sum_{i=1}^{n} \langle \rho_i^{(k)}, \mathbf{x}_1^{(k+1)} - \mathbf{z} \rangle
\end{aligned}
$$

Since for $i \leq n_d^{(k)}$,

$$
\begin{aligned}
B_{f_{\pi_i^{(k)}}}(\mathbf{x}_1^{(k+1)}, \mathbf{x}_i^{(k)}) &= f_{\pi_i^{(k)}}(\mathbf{x}_1^{(k+1)}) - f_{\pi_i^{(k)}}(\mathbf{x}_i^{(k)}) - \langle \nabla f_{\pi_i^{(k)}}(\mathbf{x}_i^{(k)}), \mathbf{x}_1^{(k+1)} - \mathbf{x}_i^{(k)} \rangle \\
B_{f_{\pi_i^{(k)}}}(\mathbf{z}, \mathbf{x}_i^{(k)}) &= f_{\pi_i^{(k)}}(\mathbf{z}) - f_{\pi_i^{(k)}}(\mathbf{x}_i^{(k)}) - \langle \nabla f_{\pi_i^{(k)}}(\mathbf{x}_i^{(k)}), \mathbf{z} - \mathbf{x}_i^{(k)} \rangle
\end{aligned}
$$

and for $n_d^{(k)} < i \leq n$,

$$
\begin{aligned}
B_{f_{i-n_d^{(k)}}^{(k,pub)}}(\mathbf{x}_1^{(k+1)}, \mathbf{x}_i^{(k)}) &= f_{i-n_d^{(k)}}^{(k,pub)}(\mathbf{x}_1^{(k+1)}) - f_{i-n_d^{(k)}}^{(k,pub)}(\mathbf{x}_i^{(k)}) - \langle \nabla f_{i-n_d^{(k)}}^{(k,pub)}(\mathbf{x}_i^{(k)}), \mathbf{x}_1^{(k+1)} - \mathbf{x}_i^{(k)} \rangle \\
B_{f_{i-n_d^{(k)}}^{(k,pub)}}(\mathbf{z}, \mathbf{x}_i^{(k)}) &= f_{i-n_d^{(k)}}^{(k,pub)}(\mathbf{z}) - f_{i-n_d^{(k)}}^{(k,pub)}(\mathbf{x}_i^{(k)}) - \langle \nabla f_{i-n_d^{(k)}}^{(k,pub)}(\mathbf{x}_i^{(k)}), \mathbf{z} - \mathbf{x}_i^{(k)} \rangle
\end{aligned}
$$

there is for $i \leq n_d^{(k)}$,

$$
\begin{aligned}
&\sum_{i=1}^{n_d^{(k)}} \langle \nabla f_{\pi_i^{(k)}}(\mathbf{x}_i^{(k)}), \mathbf{x}_1^{(k+1)} - \mathbf{z} \rangle \qquad \text{(C.10)} \\
&= \sum_{i=1}^{n_d^{(k)}} \Big(f_{\pi_i^{(k)}}(\mathbf{x}_1^{(k+1)}) - f_{\pi_i^{(k)}}(\mathbf{z}) - B_{f_{\pi_i^{(k)}}}(\mathbf{x}_1^{(k+1)}, \mathbf{x}_i^{(k)}) + B_{f_{\pi_i^{(k)}}}(\mathbf{z}, \mathbf{x}_i^{(k)}) \Big)
\end{aligned}
$$

and for $n_d^{(k)} < i \leq n$,

$$
\sum_{i=n_d^{(k)}+1}^{n} \langle \nabla f_{i-n_d^{(k)}}^{(k,pub)}(\mathbf{x}_i^{(k)}), \mathbf{x}_1^{(k+1)} - \mathbf{z} \rangle \qquad \text{(C.11)}
$$

$$= \sum_{i=n_d^{(k)}+1}^{n} \left(f_{i-n_d^{(k)}}^{(k,pub)}(\mathbf{x}_1^{(k+1)}) - f_{i-n_d^{(k)}}^{(k,pub)}(\mathbf{z}) - B_{f_{i-n_d^{(k)}}^{(k,pub)}}(\mathbf{x}_1^{(k+1)}, \mathbf{x}_i^{(k)}) + B_{f_{i-n_d^{(k)}}^{(k,pub)}}(\mathbf{z}, \mathbf{x}_i^{(k)}) \right)$$

Therefore, summing up Eq. C.10 and Eq. C.11, we have

$$\begin{aligned}
&\sum_{i=1}^{n_d^{(k)}} \langle \nabla f_{\pi_i^{(k)}}(\mathbf{x}_i^{(k)}), \mathbf{x}_1^{(k+1)} - \mathbf{z} \rangle + \sum_{i=n_d^{(k)}+1}^{n} \langle \nabla f_{i-n_d^{(k)}}^{(k,pub)}(\mathbf{x}_i^{(k)}), \mathbf{x}_1^{(k+1)} - \mathbf{z} \rangle \\
&= \sum_{i=1}^{n_d^{(k)}} \left(f_{\pi_i^{(k)}}(\mathbf{x}_1^{(k+1)}) - f_{\pi_i^{(k)}}(\mathbf{z}) \right) + \sum_{i=n_d^{(k)}+1}^{n} \left(f_{i-n_d^{(k)}}^{(k,pub)}(\mathbf{x}_1^{(k+1)}) - f_{i-n_d^{(k)}}^{(k,pub)}(\mathbf{z}) \right) \\
&\quad - \sum_{i=1}^{n_d^{(k)}} \left(B_{f_{\pi_i^{(k)}}}(\mathbf{x}_1^{(k+1)}, \mathbf{x}_i^{(k)}) - B_{f_{\pi_i^{(k)}}}(\mathbf{z}, \mathbf{x}_i^{(k)}) \right) \\
&\quad - \sum_{i=n_d^{(k)}+1}^{n} \left(B_{f_{i-n_d^{(k)}}^{(k,pub)}}(\mathbf{x}_1^{(k+1)}, \mathbf{x}_i^{(k)}) - B_{f_{i-n_d^{(k)}}^{(k,pub)}}(\mathbf{z}, \mathbf{x}_i^{(k)}) \right) \\
&= nF^{(k)}(\mathbf{x}_1^{(k+1)}) - nF^{(k)}(\mathbf{z}) - \sum_{i=1}^{n_d^{(k)}} \left(B_{f_{\pi_i^{(k)}}^{(k,priv)}}(\mathbf{x}_1^{(k+1)}, \mathbf{x}_i^{(k)}) - B_{f_{\pi_i^{(k)}}^{(k,priv)}}(\mathbf{z}, \mathbf{x}_i^{(k)}) \right) \\
&\quad - \sum_{i=n_d^{(k)}+1}^{n} \left(B_{f_{i-n_d^{(k)}}^{(k,pub)}}(\mathbf{x}_1^{(k+1)}, \mathbf{x}_i^{(k)}) - B_{f_{i-n_d^{(k)}}^{(k,pub)}}(\mathbf{z}, \mathbf{x}_i^{(k)}) \right)
\end{aligned} \tag{C.12}$$

Hence, plugging Eq. C.12 back to Eq. C.9, there is

$$\begin{aligned}
\langle \mathbf{g}^{(k)}, \mathbf{x}_1^{(k+1)} - \mathbf{z} \rangle &= nF^{(k)}(\mathbf{x}_1^{(k+1)}) - nF^{(k)}(\mathbf{z}) + \sum_{i=1}^{n} \langle \rho_i^{(k)}, \mathbf{x}_1^{(k+1)} - \mathbf{z} \rangle \\
&\quad - \sum_{i=1}^{n_d^{(k)}} \left(B_{f_{\pi_i^{(k)}}}(\mathbf{x}_1^{(k+1)}, \mathbf{x}_i^{(k)}) - B_{f_{\pi_i^{(k)}}}(\mathbf{z}, \mathbf{x}_i^{(k)}) \right) \\
&\quad - \sum_{i=n_d^{(k)}+1}^{n} \left(B_{f_{i-n_d^{(k)}}^{(k,pub)}}(\mathbf{x}_1^{(k+1)}, \mathbf{x}_i^{(k)}) - B_{f_{i-n_d^{(k)}}^{(k,pub)}}(\mathbf{z}, \mathbf{x}_i^{(k)}) \right)
\end{aligned} \tag{C.13}$$

Recall that $G(\mathbf{x}) = F(\mathbf{x}; \mathcal{D}) + \psi(\mathbf{x})$ is the target objective (see Eq. C.1) and

$G^{(k)}(\mathbf{x}) = F^{(k)}(\mathbf{x}; \mathcal{D}^{(k)} \cup \mathcal{P}^{(k)}) + \psi(\mathbf{x})$ (see Eq. C.2) is the objective used in the k-th epoch during optimization for $k \in [K]$.

Now, by Eq. C.8 and Eq. C.13, after rearranging

$$
\begin{aligned}
G^{(k)}(\mathbf{x}_1^{(k+1)}) - G^{(k)}(\mathbf{z}) \leq{}& \frac{\|\mathbf{z} - \mathbf{x}_1^{(k)}\|^2}{2n\eta} - (\frac{1}{2n\eta} + \frac{\mu_\psi}{2})\|\mathbf{z} - \mathbf{x}_1^{(k+1)}\|^2 - \frac{1}{2n\eta}\|\mathbf{x}_1^{(k+1)} - \mathbf{x}_1^{(k)}\|^2 \\
&+ \frac{1}{n}\sum_{i=1}^{n_d^{(k)}} \left(B_{f_{\pi_i^{(k)}}}(\mathbf{x}_1^{(k+1)}, \mathbf{x}_i^{(k)}) - B_{f_{\pi_i^{(k)}}}(\mathbf{z}, \mathbf{x}_i^{(k)}) \right) \\
&+ \frac{1}{n}\sum_{i=n_d^{(k)}+1}^{n} \left(B_{f^{(k,pub)}_{i-n_d^{(k)}}}(\mathbf{x}_1^{(k+1)}, \mathbf{x}_i^{(k)}) - B_{f^{(k,pub)}_{i-n_d^{(k)}}}(\mathbf{z}, \mathbf{x}_i^{(k)}) \right) + \frac{1}{n}\sum_{i=1}^{n} \langle -\rho_i^{(k)}, \mathbf{x}_1^{(k+1)} - \mathbf{z} \rangle
\end{aligned}
$$

And following the above, for any $\mathbf{z} \in \mathbb{R}^d$ and $s \in [K]$,

$$
\begin{aligned}
&G(\mathbf{x}_1^{(k+1)}) - G(\mathbf{z}) \\
&= \left(G^{(k)}(\mathbf{x}_1^{(k+1)}) - G^{(k)}(\mathbf{z}) \right) + \left(G(\mathbf{x}_1^{(k+1)}) - G^{(k)}(\mathbf{x}_1^{(k+1)}) \right) - \left(G(\mathbf{z}) - G^{(k)}(\mathbf{z}) \right) \\
&\leq H^{(k)}(\mathbf{x}_1^{(k+1)}) - H^{(k)}(\mathbf{z}) + \frac{\|\mathbf{z} - \mathbf{x}_1^{(k)}\|^2}{2n\eta} - (\frac{1}{2n\eta} + \frac{\mu_\psi}{2})\|\mathbf{z} - \mathbf{x}_1^{(k+1)}\|^2 \\
&\quad - \frac{1}{2n\eta}\|\mathbf{x}_1^{(k+1)} - \mathbf{x}_1^{(k)}\|^2 + \frac{1}{n}\sum_{i=1}^{n_d^{(k)}} \left(B_{f_{\pi_i^{(k)}}}(\mathbf{x}_1^{(k+1)}, \mathbf{x}_i^{(k)}) - B_{f_{\pi_i^{(k)}}}(\mathbf{z}, \mathbf{x}_i^{(k)}) \right) \\
&\quad + \frac{1}{n}\sum_{i=n_d^{(k)}+1}^{n} \left(B_{f^{(k,pub)}_{i-n_d^{(k)}}}(\mathbf{x}_1^{(k+1)}, \mathbf{x}_i^{(k)}) - B_{f^{(k,pub)}_{i-n_d^{(k)}}}(\mathbf{z}, \mathbf{x}_i^{(k)}) \right) + \frac{1}{n}\sum_{i=1}^{n} \langle -\rho_i^{(k)}, \mathbf{x}_1^{(k+1)} - \mathbf{z} \rangle
\end{aligned}
$$

□

The following lemma is a generalization of Lemma D.2 of [76] from two dimensions: allowing the usage of surrogate objectives and adding additional noise for privacy preservation.

Lemma C.1.8. *Under Assumptions 4.4.1, 4.4.6 and C.1.1, for any epoch $k \in [K]$, permutation $\pi^{(k)}$ and $\mathbf{z} \in \mathbb{R}^d$, if the learning rate $\eta \leq \frac{1}{n\sqrt{10\widehat{L}^{(k)}\widehat{L}^{(k)*}}}$, Algorithm 3*

guarantees

$$
\begin{aligned}
&\frac{1}{n}\Big(\sum_{i=1}^{n_d^{(k)}}\Big(B_{f_{\pi_i^{(k)}}}(\mathbf{x}_1^{(k+1)},\mathbf{x}_i^{(k)})-B_{f_{\pi_i^{(k)}}}(\mathbf{z},\mathbf{x}_i^{(k)})\Big)\\
&\quad+\sum_{i=n_d^{(k)}+1}^{n}\Big(B_{f^{(k,pub)}_{i-n_d^{(k)}}}(\mathbf{x}_1^{(k+1)},\mathbf{x}_i^{(k)})-B_{f^{(k,pub)}_{i-n_d^{(k)}}}(\mathbf{z},\mathbf{x}_i^{(k)})\Big)\Big)\\
&\le \widehat{L}^{(k)}\|\mathbf{x}_1^{(k+1)}-\mathbf{x}_1^{(k)}\|^2+10\eta^2n^2\widehat{L}^{(k)}LB_F(\mathbf{z},\mathbf{x}^*)\\
&\quad+5\eta^2\frac{1}{n}\Big(\sum_{i=2}^{n_d^{(k)}}L_{\pi_i^{(k)}}\Big\|\sum_{j=1}^{i-1}\nabla f_{\pi_j^{(k)}}(\mathbf{x}^*)\Big\|^2+\sum_{i=n_d^{(k)}+1}^{n}\widetilde{L}^{(k)}_{i-n_d^{(k)}}\Big\|\sum_{j=1}^{i-1}\nabla f_{\pi_j^{(k)}}(\mathbf{x}^*)\Big\|^2\Big)\\
&\quad+5\eta^2\frac{1}{n}\Big(\sum_{i=2}^{n_d^{(k)}}L_{\pi_i^{(k)}}\Big\|\sum_{j=1}^{i-1}\rho_j^{(k)}\Big\|^2+\sum_{i=n_d^{(k)}+1}^{n}\widetilde{L}^{(k)}_{i-n_d^{(k)}}\Big\|\sum_{j=1}^{i-1}\rho_j^{(k)}\Big\|^2\Big)\\
&\quad+5\eta^2L^{(k)*}\frac{1}{n}\sum_{i=n_d^{(k)}+1}^{n}\Big\|\sum_{j=n_d^{(k)}+1}^{i-1}\nabla f^{(k,pub)}_{i-n_d^{(k)}}(\mathbf{z})-\sum_{j=n_d^{(k)}+1}^{i-1}\nabla f_{\pi_j^{(k)}}(\mathbf{z})\Big\|^2
\end{aligned}
$$

Proof of Lemma C.1.8. By Lemma C.1.4, for $i\le n_d^{(k)}$, and permutation $\pi_i^{(k)}\in\Pi_n$,

$$
\begin{aligned}
&B_{f_{\pi_i^{(k)}}}(\mathbf{x}_1^{(k+1)},\mathbf{x}_i^{(k)})\le\frac{L_{\pi_i^{(k)}}}{2}\|\mathbf{x}_1^{(k+1)}-\mathbf{x}_i^{(k)}\|^2\le L_{\pi_i^{(k)}}\Big(\|\mathbf{x}_1^{(k+1)}-\mathbf{x}_1^{(k)}\|^2+\|\mathbf{x}_i^{(k)}-\mathbf{x}_1^{(k)}\|^2\Big)\\
&B_{f_{\pi_i^{(k)}}}(\mathbf{z},\mathbf{x}_i^{(k)})\ge\frac{\Big\|\nabla f_{\pi_i^{(k)}}(\mathbf{x}_i^{(k)})-\nabla f_{\pi_i^{(k)}}(\mathbf{z})\Big\|^2}{2L_{\pi_i^{(k)}}}
\end{aligned}
$$

and for $n_d^{(k)}<i\le n$,

$$
\begin{aligned}
&B_{f^{(k,pub)}_{i-n_d^{(k)}}}(\mathbf{x}_1^{(k+1)},\mathbf{x}_i^{(k)})\le\frac{\widetilde{L}^{(k)}_{i-n_d^{(k)}}}{2}\|\mathbf{x}_1^{(k+1)}-\mathbf{x}_i^{(k)}\|^2\le\widetilde{L}^{(k)}_{i-n_d^{(k)}}\Big(\|\mathbf{x}_1^{(k+1)}-\mathbf{x}_1^{(k)}\|^2+\|\mathbf{x}_i^{(k)}-\mathbf{x}_1^{(k)}\|^2\Big)\\
&B_{f^{(k,pub)}_{i-n_d^{(k)}}}(\mathbf{z},\mathbf{x}_i^{(k)})\ge\frac{\Big\|\nabla f^{(k,pub)}_{i-n_d^{(k)}}(\mathbf{x}_i^{(k)})-\nabla f^{(k,pub)}_{i-n_d^{(k)}}(\mathbf{z})\Big\|^2}{2\widetilde{L}^{(k)}_{i-n_d^{(k)}}}
\end{aligned}
$$

Therefore,

$$
\begin{aligned}
&\frac{1}{n}\Big(\sum_{i=1}^{n_d^{(k)}}\Big(B_{f_{\pi_i^{(k)}}}(\mathbf{x}_1^{(k+1)},\mathbf{x}_i^{(k)})-B_{f_{\pi_i^{(k)}}}(\mathbf{z},\mathbf{x}_i^{(k)})\Big)\\
&\quad+\sum_{i=n_d^{(k)}+1}^{n}\Big(B_{f^{(k,pub)}_{i-n_d^{(k)}}}(\mathbf{x}_1^{(k+1)},\mathbf{x}_i^{(k)})-B_{f^{(k,pub)}_{i-n_d^{(k)}}}(\mathbf{z},\mathbf{x}_i^{(k)})\Big)\Big)\\
&\le\frac{1}{n}\sum_{i=1}^{n_d^{(k)}}\Big(L_{\pi_i^{(k)}}\Big(\|\mathbf{x}_1^{(k+1)}-\mathbf{x}_1^{(k)}\|^2+\|\mathbf{x}_i^{(k)}-\mathbf{x}_1^{(k)}\|^2\Big)-\frac{\Big\|\nabla f_{\pi_i^{(k)}}(\mathbf{x}_i^{(k)})-\nabla f_{\pi_i^{(k)}}(\mathbf{z})\Big\|^2}{2L_{\pi_i^{(k)}}}\Big)\\
&\quad+\frac{1}{n}\sum_{i=n_d^{(k)}+1}^{n}\Big(\widetilde{L}^{(k)}_{i-n_d^{(k)}}\Big(\|\mathbf{x}_1^{(k+1)}-\mathbf{x}_1^{(k)}\|^2+\|\mathbf{x}_i^{(k)}-\mathbf{x}_1^{(k)}\|^2\Big)\\
&\quad-\frac{\Big\|\nabla f^{(k,pub)}_{i-n_d^{(k)}}(\mathbf{x}_i^{(k)})-\nabla f^{(k,pub)}_{i-n_d^{(k)}}(\mathbf{z})\Big\|^2}{2\widetilde{L}^{(k)}_{i-n_d^{(k)}}}\Big)\\
&=\widehat{L}^{(k)}\|\mathbf{x}_1^{(k+1)}-\mathbf{x}_1^{(k)}\|^2+\frac{1}{n}\Big(\underbrace{\sum_{i=1}^{n_d^{(k)}}L_{\pi_i^{(k)}}\|\mathbf{x}_i^{(k)}-\mathbf{x}_1^{(k)}\|^2}_{:=I_1}+\underbrace{\sum_{i=n_d^{(k)}+1}^{n}\widetilde{L}^{(k)}_{i-n_d^{(k)}}\|\mathbf{x}_i^{(k)}-\mathbf{x}_1^{(k)}\|^2}_{:=I_2}\Big)
\end{aligned}
\tag{C.14}
$$

$$
-\frac{1}{n}\Big(\sum_{i=1}^{n_d^{(k)}}\frac{\Big\|\nabla f_{\pi_i^{(k)}}(\mathbf{x}_i^{(k)})-\nabla f_{\pi_i^{(k)}}(\mathbf{z})\Big\|^2}{2L_{\pi_i^{(k)}}}+\sum_{i=n_d^{(k)}+1}^{n}\frac{\Big\|\nabla f^{(k,pub)}_{i-n_d^{(k)}}(\mathbf{x}_i^{(k)})-\nabla f^{(k,pub)}_{i-n_d^{(k)}}(\mathbf{z})\Big\|^2}{2\widetilde{L}^{(k)}_{i-n_d^{(k)}}}\Big)
$$

where recall that $\widehat{L}^{(k)}=\frac{1}{n}\sum_{i=1}^{n}\Big(\sum_{i=1}^{n_d^{(k)}}L_{\pi_i^{(k)}}+\sum_{j=1}^{n-n_d^{(k)}}\widetilde{L}_j^{(k)}\Big)$. Now we bound terms $I_1\triangleq\sum_{i=1}^{n_d^{(k)}}L_{\pi_i^{(k)}}\|\mathbf{x}_i^{(k)}-\mathbf{x}_1^{(k)}\|^2$ (in Part I) and $I_2\triangleq\sum_{i=n_d^{(k)}+1}^{n}\widetilde{L}^{(k)}_{i-n_d^{(k)}}\|\mathbf{x}_i^{(k)}-\mathbf{x}_1^{(k)}\|^2$ (in Part II) as follows:

Part I: For $i\le n_d^{(k)}$,

$$
I_1\triangleq\sum_{i=1}^{n_d^{(k)}}L_{\pi_i^{(k)}}\|\mathbf{x}_i^{(k)}-\mathbf{x}_1^{(k)}\|^2=\sum_{i=2}^{n_d^{(k)}}L_{\pi_i^{(k)}}\|\mathbf{x}_i^{(k)}-\mathbf{x}_1^{(k)}\|^2 \tag{C.15}
$$

$$
\begin{aligned}
&= \sum_{i=2}^{n_d^{(k)}} L_{\pi_i^{(k)}} \eta^2 \Big\| \sum_{j=1}^{i-1} (\nabla f_{\pi_j^{(k)}}(\mathbf{x}_j^{(k)}) + \rho_j^{(k)}) \Big\|^2 && \text{(From the update in 3)} \\
&= \eta^2 \sum_{i=2}^{n_d^{(k)}} L_{\pi_i^{(k)}} \Big\| \sum_{j=1}^{i-1} \Big(\nabla f_{\pi_j^{(k)}}(\mathbf{x}_j^{(k)}) - \nabla f_{\pi_j^{(k)}}(\mathbf{z}) + \nabla f_{\pi_j^{(k)}}(\mathbf{z}) - \nabla f_{\pi_j^{(k)}}(\mathbf{x}^*) + \nabla f_{\pi_j^{(k)}}(\mathbf{x}^*) + \rho_j^{(k)} \Big) \Big\|^2 \\
&\leq \eta^2 \sum_{i=2}^{n_d^{(k)}} L_{\pi_i^{(k)}} \Big(4 \Big\| \sum_{j=1}^{i-1} \Big(\nabla f_{\pi_j^{(k)}}(\mathbf{x}_j^{(k)}) - \nabla f_{\pi_j^{(k)}}(\mathbf{z}) \Big) \Big\|^2 && \text{(C.16)} \\
&\quad + 4 \Big\| \sum_{j=1}^{i-1} \Big(\nabla f_{\pi_j^{(k)}}(\mathbf{z}) - \nabla f_{\pi_j^{(k)}}(\mathbf{x}^*) \Big) \Big\|^2 + 4 \Big\| \sum_{j=1}^{i-1} \nabla f_{\pi_j^{(k)}}(\mathbf{x}^*) \Big\|^2 + 4 \Big\| \sum_{j=1}^{i-1} \rho_j^{(k)} \Big\|^2 \Big)
\end{aligned}
$$

We proceed by bounding the first two terms in Eq. C.16 separately. First,

$$
\begin{aligned}
&\sum_{i=2}^{n_d^{(k)}} L_{\pi_i^{(k)}} \Big\| \sum_{j=1}^{i-1} \Big(\nabla f_{\pi_j^{(k)}}(\mathbf{x}_j^{(k)}) - \nabla f_{\pi_j^{(k)}}(\mathbf{z}) \Big) \Big\|^2 \\
&\leq \sum_{i=2}^{n_d^{(k)}} L_{\pi_i^{(k)}} (i-1) \sum_{j=1}^{i-1} \Big\| \nabla f_{\pi_j^{(k)}}(\mathbf{x}_j^{(k)}) - \nabla f_{\pi_j^{(k)}}(\mathbf{z}) \Big\|^2 \\
&= \sum_{j=1}^{n_d^{(k)}-1} \Big(\sum_{i=j+1}^{n_d^{(k)}} L_{\pi_i^{(k)}} (i-1) \Big) \Big\| \nabla f_{\pi_j^{(k)}}(\mathbf{x}_j^{(k)}) - \nabla f_{\pi_j^{(k)}}(\mathbf{z}) \Big\|^2 \Big) \\
&\leq \sum_{j=1}^{n_d^{(k)}-1} n \Big(\sum_{i=1}^{n_d^{(k)}} L_{\pi_i^{(k)}} \Big) \Big\| \nabla f_{\pi_j^{(k)}}(\mathbf{x}_j^{(k)}) - \nabla f_{\pi_j^{(k)}}(\mathbf{z}) \Big\|^2 \\
&\leq n \Big(\sum_{i=1}^{n_d^{(k)}} L_{\pi_i^{(k)}} \Big) \sum_{j=1}^{n_d^{(k)}} \Big\| \nabla f_{\pi_j^{(k)}}(\mathbf{x}_j^{(k)}) - \nabla f_{\pi_j^{(k)}}(\mathbf{z}) \Big\|^2 && \text{(C.17)}
\end{aligned}
$$

Next,

$$
\begin{aligned}
&\sum_{i=2}^{n_d^{(k)}} L_{\pi_i^{(k)}} \Big\| \sum_{j=1}^{i-1} \Big(\nabla f_{\pi_j^{(k)}}(\mathbf{z}) - \nabla f_{\pi_j^{(k)}}(\mathbf{x}^*) \Big) \Big\|^2 \\
&\overset{(a)}{\leq} \sum_{i=2}^{n_d^{(k)}} L_{\pi_i^{(k)}} 2 \Big(\sum_{j=1}^{i-1} L_{\pi_j^{(k)}} \Big) \Big(\sum_{l=1}^{i-1} B_{f_{\pi_l^{(k)}}}(\mathbf{z}, \mathbf{x}^*) \Big) \leq 2nL \sum_{i=1}^{n_d^{(k)}} L_{\pi_i^{(k)}} \Big(\sum_{l=1}^{i-1} B_{f_{\pi_l^{(k)}}}(\mathbf{z}, \mathbf{x}^*) \Big)
\end{aligned}
$$

$$\overset{(b)}{\leq} 2nL\sum_{i=1}^{n_d^{(k)}} L_{\pi_i^{(k)}}\Big(\sum_{l=1}^{n} B_{f_{\pi_l}^{(k)}}(\mathbf{z},\mathbf{x}^*)\Big) = 2n^2LB_F(\mathbf{z},\mathbf{x}^*)\cdot\sum_{i=1}^{n_d^{(k)}} L_{\pi_i^{(k)}} \tag{C.18}$$

where (a) is by Lemma C.1.4 and (b) is due to $B_{f_i^{(k)}}(\mathbf{z},\mathbf{x}^*)\geq 0, \forall \mathbf{z}\in\mathbb{R}^d, i\in[n]$.

Plugging Eq. C.17 and Eq. C.18 back to Eq. C.16, there is

$$\begin{aligned} I_1 &\triangleq \sum_{i=1}^{n_d^{(k)}} L_{\pi_i^{(k)}}\|\mathbf{x}_i^{(k)}-\mathbf{x}_1^{(k)}\|^2 \\ &\leq 4\eta^2 n(\sum_{i=1}^{n_d^{(k)}} L_{\pi_i^{(k)}})\sum_{j=1}^{n_d^{(k)}}\Big\|\nabla f_{\pi_j^{(k)}}(\mathbf{x}_j^{(k)})-\nabla f_{\pi_j^{(k)}}(\mathbf{z})\Big\|^2 \\ &\quad + 8\eta^2n^2LB_F(\mathbf{z},\mathbf{x}^*)\cdot\sum_{i=1}^{n_d^{(k)}} L_{\pi_i^{(k)}} + 4\eta^2\sum_{i=2}^{n_d^{(k)}} L_{\pi_i^{(k)}}\Big\|\sum_{j=1}^{i-1}\nabla f_{\pi_j^{(k)}}(\mathbf{x}^*)\Big\|^2 \\ &\quad + 4\eta^2\sum_{i=2}^{n_d^{(k)}} L_{\pi_i^{(k)}}\Big\|\sum_{j=1}^{i-1}\rho_j^{(k)}\Big\|^2 \end{aligned} \tag{C.19}$$

Part II: Similarly, for $n_d^{(k)} < i \leq n$,

$$\begin{aligned} I_2 &\triangleq \sum_{i=n_d^{(k)}+1}^{n}\widetilde{L}_{i-n_d^{(k)}}^{(k)}\|\mathbf{x}_i^{(k)}-\mathbf{x}_1^{(k)}\|^2 \\ &= \sum_{i=n_d^{(k)}+1}^{n}\widetilde{L}_{i-n_d^{(k)}}^{(k)}\eta^2\Big\|\sum_{j=1}^{n_d^{(k)}}(\nabla f_{\pi_j^{(k)}}(\mathbf{x}_j^{(k)})+\rho_j^{(k)}) + \sum_{j=n_d^{(k)}+1}^{i-1}(\nabla f_{j-n_d^{(k)}}^{(k,pub)}(\mathbf{x}_j^{(k)})+\rho_j^{(k)})\Big\|^2 \\ &\qquad\qquad \text{(From the update of Algorithm 3)} \\ &= \eta^2\sum_{i=n_d^{(k)}+1}^{n}\widetilde{L}_{i-n_d^{(k)}}^{(k)}\Big\|\sum_{j=1}^{n_d^{(k)}}\nabla f_{\pi_j^{(k)}}(\mathbf{x}_j^{(k)}) - \sum_{j=1}^{n_d^{(k)}}\nabla f_{\pi_j^{(k)}}(\mathbf{z}) + \sum_{j=1}^{n_d^{(k)}}\nabla f_{\pi_j^{(k)}}(\mathbf{z}) \\ &\quad - \sum_{j=1}^{n_d^{(k)}}\nabla f_{\pi_j^{(k)}}(\mathbf{x}^*) + \sum_{j=1}^{n_d^{(k)}}\nabla f_{\pi_j^{(k)}}(\mathbf{x}^*) + \sum_{j=n_d^{(k)}+1}^{i-1}\nabla f_{j-n_d^{(k)}}^{(k,pub)}(\mathbf{x}_j^{(k)}) \end{aligned}$$

$$
\begin{aligned}
&- \sum_{j=n_d^{(k)}+1}^{i-1} \nabla f_{j-n_d^{(k)}}^{(k,pub)}(\mathbf{z}) + \sum_{j=n_d^{(k)}+1}^{i-1} \nabla f_{j-n_d^{(k)}}^{(k,pub)}(\mathbf{z}) - \sum_{j=n_d^{(k)}+1}^{i-1} \nabla f_{\pi_j^{(k)}}(\mathbf{z}) \\
&+ \sum_{j=n_d^{(k)}+1}^{i-1} \nabla f_{\pi_j^{(k)}}(\mathbf{z}) - \sum_{j=n_d^{(k)}+1}^{i-1} \nabla f_{\pi_j^{(k)}}(\mathbf{x}^*) + \sum_{j=n_d^{(k)}+1}^{i-1} \nabla f_{\pi_j^{(k)}}(\mathbf{x}^*) + \sum_{j=1}^{i-1} \rho_j^{(k)} \Big\|^2 \\
&= \eta^2 \sum_{i=n_d^{(k)}+1}^{n} \widetilde{L}_{i-n_d^{(k)}}^{(k)} \Big\| \Big(\sum_{j=1}^{n_d^{(k)}} \nabla f_{\pi_j^{(k)}}(\mathbf{x}_j^{(k)}) + \sum_{j=n_d^{(k)}+1}^{i-1} \nabla f_{j-n_d^{(k)}}^{(k,pub)}(\mathbf{x}_j^{(k)}) \\
&- \sum_{j=1}^{n_d^{(k)}} \nabla f_{\pi_j^{(k)}}(\mathbf{z}) - \sum_{j=n_d^{(k)}+1}^{n} \nabla f_{j-n_d^{(k)}}^{(k,pub)}(\mathbf{z}) \Big) \\
&+ \Big(\sum_{j=n_d^{(k)}+1}^{i-1} \nabla f_{j-n_d^{(k)}}^{(k,pub)}(\mathbf{z}) - \sum_{j=n_d^{(k)}+1}^{i-1} \nabla f_{\pi_j^{(k)}}(\mathbf{z}) \Big) \\
&+ \Big(\sum_{j=1}^{i-1} \nabla f_{\pi_j^{(k)}}(\mathbf{z}) - \sum_{j=1}^{i-1} \nabla f_{\pi_j^{(k)}}(\mathbf{x}^*) \Big) + \sum_{j=1}^{i-1} \nabla f_{\pi_j^{(k)}}(\mathbf{x}^*) + \sum_{j=1}^{i-1} \rho_j^{(k)} \Big\|^2 \\
&\leq \eta^2 \sum_{i=n_d^{(k)}+1}^{n} \widetilde{L}_{i-n_d^{(k)}}^{(k)} \Big(5 \Big\| \sum_{j=1}^{n_d^{(k)}} \nabla f_{\pi_j^{(k)}}(\mathbf{x}_j^{(k)}) + \sum_{j=n_d^{(k)}+1}^{i-1} \nabla f_{j-n_d^{(k)}}^{(k,pub)}(\mathbf{x}_j^{(k)}) \\
&- \sum_{j=1}^{n_d^{(k)}} \nabla f_{\pi_j^{(k)}}(\mathbf{z}) - \sum_{j=n_d^{(k)}+1}^{i-1} \nabla f_{j-n_d^{(k)}}^{(k,pub)}(\mathbf{z}) \Big\|^2 \\
&+ 5 \Big\| \sum_{j=n_d^{(k)}+1}^{i-1} \nabla f_{j-n_d^{(k)}}^{(k,pub)}(\mathbf{z}) - \sum_{j=n_d^{(k)}+1}^{i-1} \nabla f_{\pi_j^{(k)}}(\mathbf{z}) \Big\|^2 \\
&+ 5 \Big\| \sum_{j=1}^{i-1} \nabla f_{\pi_j^{(k)}}(\mathbf{z}) - \sum_{j=1}^{i-1} \nabla f_{\pi_j^{(k)}}(\mathbf{x}^*) \Big\|^2 + 5 \Big\| \sum_{j=1}^{i-1} \nabla f_{\pi_j^{(k)}}(\mathbf{x}^*) \Big\|^2 + 5 \Big\| \sum_{j=1}^{i-1} \rho_j^{(k)} \Big\|^2 \Big)
\end{aligned}
\tag{C.20}
$$

We proceed by bounding the first three terms in Eq. C.20 separately. First,

$$
\sum_{i=n_d^{(k)}+1}^{n} \widetilde{L}_{i-n_d^{(k)}}^{(k)} \Big\| \sum_{j=1}^{n_d^{(k)}} \nabla f_{\pi_j^{(k)}}(\mathbf{x}_j^{(k)}) + \sum_{j=n_d^{(k)}+1}^{i-1} \nabla f_{j-n_d^{(k)}}^{(k,pub)}(\mathbf{x}_j^{(k)})
$$

$$
\begin{aligned}
&- \sum_{j=1}^{n_d^{(k)}} \nabla f_{\pi_j^{(k)}}(\mathbf{z}) - \sum_{j=n_d^{(k)}+1}^{i-1} \nabla f^{(k,pub)}_{j-n_d^{(k)}}(\mathbf{z}) \Big\|^2 \\
&\leq \sum_{i=n_d^{(k)}+1}^{n} \widetilde{L}^{(k)}_{i-n_d^{(k)}}(i-1) \Big(\sum_{j=1}^{n_d^{(k)}} \Big\| \nabla f_{\pi_j^{(k)}}(\mathbf{x}_j^{(k)}) - \nabla f_{\pi_j^{(k)}}(\mathbf{z}) \Big\|^2 \\
&\quad + \sum_{j=n_d^{(k)}+1}^{i-1} \Big\| \nabla f^{(k,pub)}_{j-n_d^{(k)}}(\mathbf{x}_j^{(k)}) - \nabla f^{(k,pub)}_{j-n_d^{(k)}}(\mathbf{z}) \Big\|^2 \Big) \\
&\leq \sum_{j=1}^{n_d^{(k)}} \Big(\sum_{i=n_d^{(k)}+1}^{n} \widetilde{L}^{(k)}_{i-n_d^{(k)}}(i-1) \Big) \Big\| \nabla f_{\pi_j^{(k)}}(\mathbf{x}_j^{(k)}) - \nabla f_{\pi_j^{(k)}}(\mathbf{z}) \Big\|^2 \\
&\quad + \sum_{j=n_d^{(k)}+1}^{n-1} \Big(\sum_{i=n_d^{(k)}+1}^{n} \widetilde{L}^{(k)}_{i-n_d^{(k)}}(i-1) \Big) \Big\| \nabla f^{(k,pub)}_{j-n_d^{(k)}}(\mathbf{x}_j^{(k)}) - \nabla f^{(k,pub)}_{j-n_d^{(k)}}(\mathbf{z}) \Big\|^2 \\
&\leq \sum_{j=1}^{n_d^{(k)}} n \Big(\sum_{i=n_d^{(k)}+1}^{n} \widetilde{L}^{(k)}_{i-n_d^{(k)}} \Big) \Big\| \nabla f_{\pi_j^{(k)}}(\mathbf{x}_j^{(k)}) - \nabla f_{\pi_j^{(k)}}(\mathbf{z}) \Big\|^2 \qquad \text{(C.21)} \\
&\quad + \sum_{j=n_d^{(k)}+1}^{n} n \Big(\sum_{i=n_d^{(k)}+1}^{n} \widetilde{L}^{(k)}_{i-n_d^{(k)}} \Big) \Big\| \nabla f^{(k,pub)}_{j-n_d^{(k)}}(\mathbf{x}_j^{(k)}) - \nabla f^{(k,pub)}_{j-n_d^{(k)}}(\mathbf{z}) \Big\|^2
\end{aligned}
$$

Next,

$$
\begin{aligned}
&\sum_{i=n_d^{(k)}+1}^{n} \widetilde{L}^{(k)}_{i-n_d^{(k)}} \Big\| \sum_{j=n_d^{(k)}+1}^{i-1} \nabla f^{(k,pub)}_{i-n_d^{(k)}}(\mathbf{z}) - \sum_{j=n_d^{(k)}+1}^{i-1} \nabla f_{\pi_j^{(k)}}(\mathbf{z}) \Big\|^2 \qquad \text{(C.22)} \\
&\leq L^{(k)*} \sum_{i=n_d^{(k)}+1}^{n} \Big\| \sum_{j=n_d^{(k)}+1}^{i-1} \nabla f^{(k,pub)}_{i-n_d^{(k)}}(\mathbf{z}) - \sum_{j=n_d^{(k)}+1}^{i-1} \nabla f_{\pi_j^{(k)}}(\mathbf{z}) \Big\|^2
\end{aligned}
$$

Moreover,

$$
\sum_{i=n_d^{(k)}+1}^{n} \widetilde{L}^{(k)}_{i-n_d^{(k)}} \Big\| \sum_{j=1}^{i-1} \nabla f_{\pi_j^{(k)}}(\mathbf{z}) - \sum_{j=1}^{i-1} \nabla f_{\pi_j^{(k)}}(\mathbf{x}^*) \Big\|^2
$$

$$\overset{(a)}{\leq} \sum_{i=n_d^{(k)}+1}^{n} \widetilde{L}^{(k)}_{i-n_d^{(k)}} \cdot 2\Big(\sum_{j=1}^{i-1} L_{\pi_j^{(k)}}\Big)\Big(\sum_{j=1}^{i-1} B_{\pi_j^{(k)}}(\mathbf{z},\mathbf{x}^*)\Big)$$

$$\overset{(b)}{\leq} \sum_{i=n_d^{(k)}+1}^{n} \widetilde{L}^{(k)}_{i-n_d^{(k)}} \cdot 2\Big(\sum_{j=1}^{i-1} L_{\pi_j^{(k)}}\Big)\Big(\sum_{j=1}^{n} B_{\pi_j^{(k)}}(\mathbf{z},\mathbf{x}^*)\Big)$$

$$\leq 2\sum_{i=n_d^{(k)}+1}^{n} \widetilde{L}^{(k)}_{i-n_d^{(k)}} \cdot nL \cdot \Big(\sum_{j=1}^{n} B_j(\mathbf{z},\mathbf{x}^*)\Big) \leq 2n^2 LB_F(\mathbf{z},\mathbf{x}^*) \cdot \sum_{i=n_d^{(k)}+1}^{n} \widetilde{L}^{(k)}_{i-n_d^{(k)}} \quad \text{(C.23)}$$

where (a) is by Lemma C.1.4 and (b) is due to $B_{f_i}(\mathbf{z},\mathbf{x}^*) \geq 0, \forall \mathbf{z} \in \mathbb{R}^d, i \in [n]$. Plugging Eq. C.21, Eq. C.22 and Eq. C.23 back to Eq. C.20, there is

$$\begin{aligned}
I_2 &= \sum_{i=n_d^{(k)}+1}^{n} \widetilde{L}^{(k)}_{i-n_d^{(k)}} \|\mathbf{x}_i^{(k)} - \mathbf{x}_1^{(k)}\|^2 \qquad \text{(C.24)}\\
&\leq 5\eta^2 n\Big(\sum_{i=n_d^{(k)}+1}^{n} \widetilde{L}^{(k)}_{i-n_d^{(k)}}\Big)\sum_{j=1}^{n_d^{(k)}} \Big\|\nabla f_{\pi_j^{(k)}}(\mathbf{x}_j^{(k)}) - \nabla f_{\pi_j^{(k)}}(\mathbf{z})\Big\|^2\\
&\quad + 5\eta^2 n\Big(\sum_{i=n_d^{(k)}+1}^{n} \widetilde{L}^{(k)}_{i-n_d^{(k)}}\Big)\sum_{j=n_d^{(k)}+1}^{n} \Big\|\nabla f^{(k,pub)}_{j-n_d^{(k)}}(\mathbf{x}_j^{(k)}) - \nabla f^{(k,pub)}_{j-n_d^{(k)}}(\mathbf{z})\Big\|^2\\
&\quad + 5\eta^2 L^{(k)*}\sum_{i=n_d^{(k)}+1}^{n} \Big\| \sum_{j=n_d^{(k)}+1}^{i-1} \nabla f^{(k,pub)}_{i-n_d^{(k)}}(\mathbf{z}) - \sum_{j=n_d^{(k)}+1}^{i-1} \nabla f_{\pi_j^{(k)}}(\mathbf{z})\Big\|^2\\
&\quad + 10\eta^2 n^2 LB_F(\mathbf{z},\mathbf{x}^*)\sum_{i=n_d^{(k)}+1}^{n} \widetilde{L}^{(k)}_{i-n_d^{(k)}}\\
&\quad + 5\eta^2\sum_{i=n_d^{(k)}+1}^{n} \widetilde{L}^{(k)}_{i-n_d^{(k)}}\Big\|\sum_{j=1}^{i-1}\nabla f_{\pi_j^{(k)}}(\mathbf{x}^*)\Big\|^2 + 5\eta^2\sum_{i=n_d^{(k)}+1}^{n} \widetilde{L}^{(k)}_{i-n_d^{(k)}}\Big\|\sum_{j=1}^{i-1}\rho_j^{(k)}\Big\|^2
\end{aligned}$$

Combining Eq. C.19 and Eq. C.24, there is

$$\frac{1}{n}(I_1+I_2) = \frac{1}{n}\Big(\sum_{i=1}^{n_d^{(k)}} L_{\pi_i^{(k)}}\|\mathbf{x}_i^{(k)} - \mathbf{x}_1^{(k)}\|^2 + \sum_{i=n_d^{(k)}+1}^{n} \widetilde{L}^{(k)}_{i-n_d^{(k)}}\|\mathbf{x}_i^{(k)} - \mathbf{x}_1^{(k)}\|^2\Big) \qquad \text{(C.25)}$$

$$
\begin{aligned}
&\leq 5\eta^2 n\widehat{L}^{(k)} \sum_{j=1}^{n_d^{(k)}} \left\|\nabla f_{\pi_j^{(k)}}(\mathbf{x}_j^{(k)}) - \nabla f_{\pi_j^{(k)}}(\mathbf{z})\right\|^2 \\
&\quad + 5\eta^2 n\widehat{L}^{(k)} \sum_{j=n_d^{(k)}+1}^{n} \left\|\nabla f^{(k,pub)}_{j-n_d^{(k)}}(\mathbf{x}_j^{(k)}) - \nabla f^{(k,pub)}_{j-n_d^{(k)}}(\mathbf{z})\right\|^2 + 10\eta^2 n^2 \widehat{L}^{(k)} L B_F(\mathbf{z}, \mathbf{x}^*) \\
&\quad + 5\eta^2 \frac{1}{n}\Big(\sum_{i=2}^{n_d^{(k)}} L_{\pi_i^{(k)}} \Big\| \sum_{j=1}^{i-1} \nabla f_{\pi_j^{(k)}}(\mathbf{x}^*)\Big\|^2 + \sum_{i=n_d^{(k)}+1}^{n} \widetilde{L}^{(k)}_{i-n_d^{(k)}} \Big\| \sum_{j=1}^{i-1} \nabla f_{\pi_j^{(k)}}(\mathbf{x}^*)\Big\|^2 \Big) \\
&\quad + 5\eta^2 \frac{1}{n}\Big(\sum_{i=2}^{n_d^{(k)}} L_{\pi_i^{(k)}} \Big\| \sum_{j=1}^{i-1} \rho_j^{(k)}\Big\|^2 + \sum_{i=n_d^{(k)}+1}^{n} \widetilde{L}^{(k)}_{i-n_d^{(k)}} \Big\| \sum_{j=1}^{i-1} \rho_j^{(k)}\Big\|^2 \Big) \\
&\quad + 5\eta^2 L^{(k)*} \frac{1}{n} \sum_{i=n_d^{(k)}+1}^{n} \Big\| \sum_{j=n_d^{(k)}+1}^{i-1} \nabla f^{(k,pub)}_{i-n_d^{(k)}}(\mathbf{z}) - \sum_{j=n_d^{(k)}+1}^{i-1} \nabla f_{\pi_j^{(k)}}(\mathbf{z})\Big\|^2
\end{aligned}
$$

Hence, plugging Eq. C.25 back to Eq. C.14, there is

$$
\begin{aligned}
&\frac{1}{n}\Big(\sum_{i=1}^{n_d^{(k)}} \Big(B_{f_{\pi_i^{(k)}}}(\mathbf{x}_1^{(k+1)}, \mathbf{x}_i^{(k)}) - B_{f_{\pi_i^{(k)}}}(\mathbf{z}, \mathbf{x}_i^{(k)})\Big) \\
&\quad + \sum_{i=n_d^{(k)}+1}^{n} \Big(B_{f^{(k,pub)}_{i-n_d^{(k)}}}(\mathbf{x}_1^{(k+1)}, \mathbf{x}_i^{(k)}) - B_{f^{(k,pub)}_{i-n_d^{(k)}}}(\mathbf{z}, \mathbf{x}_i^{(k)})\Big)\Big) \\
&= \widehat{L}^{(k)} \|\mathbf{x}_1^{(k+1)} - \mathbf{x}_1^{(k)}\|^2 + 5\eta^2 n\widehat{L}^{(k)} \sum_{j=1}^{n_d^{(k)}} \left\|\nabla f_{\pi_j^{(k)}}(\mathbf{x}_j^{(k)}) - \nabla f_{\pi_j^{(k)}}(\mathbf{z})\right\|^2 \\
&\quad + 5\eta^2 n\widehat{L}^{(k)} \sum_{j=n_d^{(k)}+1}^{n} \left\|\nabla f^{(k,pub)}_{j-n_d^{(k)}}(\mathbf{x}_j^{(k)}) - \nabla f^{(k,pub)}_{j-n_d^{(k)}}(\mathbf{z})\right\|^2 + 10\eta^2 n^2 \widehat{L}^{(k)} L B_F(\mathbf{z}, \mathbf{x}^*) \\
&\quad + 5\eta^2 \frac{1}{n}\Big(\sum_{i=2}^{n_d^{(k)}} L_{\pi_i^{(k)}} \Big\| \sum_{j=1}^{i-1} \nabla f_{\pi_j^{(k)}}(\mathbf{x}^*)\Big\|^2 + \sum_{i=n_d^{(k)}+1}^{n} \widetilde{L}^{(k)}_{i-n_d^{(k)}} \Big\| \sum_{j=1}^{i-1} \nabla f_{\pi_j^{(k)}}(\mathbf{x}^*)\Big\|^2 \Big) \\
&\quad + 5\eta^2 \frac{1}{n}\Big(\sum_{i=2}^{n_d^{(k)}} L_{\pi_i^{(k)}} \Big\| \sum_{j=1}^{i-1} \rho_j^{(k)}\Big\|^2 + \sum_{i=n_d^{(k)}+1}^{n} \widetilde{L}^{(k)}_{i-n_d^{(k)}} \Big\| \sum_{j=1}^{i-1} \rho_j^{(k)}\Big\|^2 \Big)
\end{aligned}
$$

$$+ 5\eta^2 L^{(k)*} \frac{1}{n} \sum_{i=n_d^{(k)}+1}^{n} \Big\| \sum_{j=n_d^{(k)}+1}^{i-1} \nabla f_{i-n_d^{(k)}}^{(k,pub)}(\mathbf{z}) - \sum_{j=n_d^{(k)}+1}^{i-1} \nabla f_{\pi_j^{(k)}}(\mathbf{z}) \Big\|^2$$

$$- \frac{1}{n} \Big(\sum_{i=1}^{n_d^{(k)}} \frac{\Big\| \nabla f_{\pi_i^{(k)}}(\mathbf{x}_i^{(k)}) - \nabla f_{\pi_i^{(k)}}(\mathbf{z}) \Big\|^2}{2L_{\pi_i^{(k)}}} + \sum_{i=n_d^{(k)}+1}^{n} \frac{\Big\| \nabla f_{i-n_d^{(k)}}^{(k,pub)}(\mathbf{x}_i^{(k)}) - \nabla f_{i-n_d^{(k)}}^{(k,pub)}(\mathbf{z}) \Big\|^2}{2\widetilde{L}_{i-n_d^{(k)}}^{(k)}} \Big)$$

If one sets the learning rate η such that

$$5\eta^2 n \widehat{L}^{(k)} \leq \frac{1}{n} \cdot \frac{1}{2\widehat{L}^{(k)*}}, \quad \Rightarrow \eta \leq \frac{1}{n\sqrt{10\widehat{L}^{(k)}\widehat{L}^{(k)*}}}$$

then there is

$$\begin{aligned}
&\frac{1}{n} \Big(\sum_{i=1}^{n_d^{(k)}} \Big(B_{f_{\pi_i^{(k)}}}(\mathbf{x}_1^{(k+1)}, \mathbf{x}_i^{(k)}) - B_{f_{\pi_i^{(k)}}}(\mathbf{z}, \mathbf{x}_i^{(k)}) \Big) \\
&\quad + \sum_{i=n_d^{(k)}+1}^{n} \Big(B_{f_{i-n_d^{(k)}}^{(k,pub)}}(\mathbf{x}_1^{(k+1)}, \mathbf{x}_i^{(k)}) - B_{f_{i-n_d^{(k)}}^{(k,pub)}}(\mathbf{z}, \mathbf{x}_i^{(k)}) \Big) \Big) \\
&\leq \widehat{L}^{(k)} \|\mathbf{x}_1^{(k+1)} - \mathbf{x}_1^{(k)}\|^2 + 10\eta^2 n^2 \widehat{L}^{(k)} L B_F(\mathbf{z}, \mathbf{x}^*) \\
&\quad + 5\eta^2 \frac{1}{n} \Big(\sum_{i=2}^{n_d^{(k)}} L_{\pi_i^{(k)}} \Big\| \sum_{j=1}^{i-1} \nabla f_{\pi_j^{(k)}}(\mathbf{x}^*) \Big\|^2 + \sum_{i=n_d^{(k)}+1}^{n} \widetilde{L}_{i-n_d^{(k)}}^{(k)} \Big\| \sum_{j=1}^{i-1} \nabla f_{\pi_j^{(k)}}(\mathbf{x}^*) \Big\|^2 \Big) \\
&\quad + 5\eta^2 \frac{1}{n} \Big(\sum_{i=2}^{n_d^{(k)}} L_{\pi_i^{(k)}} \Big\| \sum_{j=1}^{i-1} \rho_j^{(k)} \Big\|^2 + \sum_{i=n_d^{(k)}+1}^{n} \widetilde{L}_{i-n_d^{(k)}}^{(k)} \Big\| \sum_{j=1}^{i-1} \rho_j^{(k)} \Big\|^2 \Big) \\
&\quad + 5\eta^2 L^{(k)*} \frac{1}{n} \sum_{i=n_d^{(k)}+1}^{n} \Big\| \sum_{j=n_d^{(k)}+1}^{i-1} \nabla f_{i-n_d^{(k)}}^{(k,pub)}(\mathbf{z}) - \sum_{j=n_d^{(k)}+1}^{i-1} \nabla f_{\pi_j^{(k)}}(\mathbf{z}) \Big\|^2
\end{aligned}$$

□

Lemma C.1.9 (One Epoch Convergence)**.** *Under Assumptions 4.4.1, 4.4.4, 4.4.6, C.1.1 and 4.4.5, for any epoch $k \in [K]$, $\beta > 0$, and $\forall \mathbf{z} \in \mathbb{R}^d$, if $\eta \leq \frac{1}{n\sqrt{10\widehat{L}^{(k)}\widehat{L}^{(k)*}}}$, Algo-*

rithm 3 guarantees

$$
\begin{aligned}
G(\mathbf{x}_1^{(k+1)}) - G(\mathbf{z}) &\leq \frac{1}{2n\eta}(\|\mathbf{z}-\mathbf{x}_1^{(k)}\|^2 - \|\mathbf{z}-\mathbf{x}_1^{(k+1)}\|^2) \\
&+ (\frac{L_H^{(k)}+\beta}{2} - \frac{\mu_\psi}{2})\|\mathbf{z}-\mathbf{x}_1^{(k+1)}\|^2 + 10\eta^2 n^2 \widehat{L}^{(k)} L B_F(\mathbf{z},\mathbf{x}^*) \\
&+ \underbrace{5\eta^2 \frac{1}{n}\Big(\sum_{i=2}^{n_d^{(k)}} L_{\pi_i^{(k)}} \Big\|\sum_{j=1}^{i-1}\nabla f_{\pi_j^{(k)}}(\mathbf{x}^*)\Big\|^2 + \sum_{i=n_d^{(k)}+1}^{n} \widetilde{L}^{(k)}_{i-n_d^{(k)}} \Big\|\sum_{j=1}^{i-1}\nabla f_{\pi_j^{(k)}}(\mathbf{x}^*)\Big\|^2\Big)}_{\textit{Optimization Uncertainty}} \\
&+ \underbrace{\frac{1}{n}\sum_{i=1}^{n}\langle -\rho_i^{(k)}, \mathbf{x}_1^{(k+1)}-\mathbf{z}\rangle + 5\eta^2\frac{1}{n}\Big(\sum_{i=2}^{n_d^{(k)}} L_{\pi_i^{(k)}} \Big\|\sum_{j=1}^{i-1}\rho_j^{(k)}\Big\|^2 + \sum_{i=n_d^{(k)}+1}^{n} \widetilde{L}^{(k)}_{i-n_d^{(k)}} \Big\|\sum_{j=1}^{i-1}\rho_j^{(k)}\Big\|^2\Big)}_{\textit{Injected Noise}} \\
&+ \underbrace{\frac{1}{2n^2\beta}(C_n^{(k)})^2}_{\textit{Non-vanishing Dissimilarity}} + \underbrace{5\eta^2 L^{(k)*}\frac{1}{n}\sum_{i=n_d^{(k)}+1}^{n}\Big\|\sum_{j=n_d^{(k)}+1}^{i-1}\nabla f_{i-n_d^{(k)}}^{(k,pub)}(\mathbf{z}) - \sum_{j=n_d^{(k)}+1}^{i-1}\nabla f_{\pi_j^{(k)}}(\mathbf{z})\Big\|^2}_{\textit{Vanishing Dissimilarity}}
\end{aligned} \tag{C.26}
$$

Proof of Lemma C.1.9. By Lemma C.1.7 and Lemma C.1.8, for any $k \in [K]$ and $\forall \mathbf{z} \in \mathbb{R}^d$, if $\eta \leq \frac{1}{n\sqrt{10\widehat{L}^{(k)}\widehat{L}^{(k)*}}}$,

$$
\begin{aligned}
G(\mathbf{x}_1^{(k+1)}) - G(\mathbf{z}) &\leq H^{(k)}(\mathbf{x}_1^{(k+1)}) - H^{(k)}(\mathbf{z}) \\
&+ \frac{\|\mathbf{z}-\mathbf{x}_1^{(k)}\|^2}{2n\eta} - (\frac{1}{2n\eta}+\frac{\mu_\psi}{2})\|\mathbf{z}-\mathbf{x}_1^{(k+1)}\|^2 - \frac{1}{2n\eta}\|\mathbf{x}_1^{(k+1)}-\mathbf{x}_1^{(k)}\|^2 \\
&+ \frac{1}{n}\sum_{i=1}^{n}\langle -\rho_i^{(k)}, \mathbf{x}_1^{(k+1)}-\mathbf{z}\rangle + \widehat{L}^{(k)}\|\mathbf{x}_1^{(k+1)}-\mathbf{x}_1^{(k)}\|^2 + 10\eta^2 n^2 \widehat{L}^{(k)} L B_F(\mathbf{z},\mathbf{x}^*) \\
&+ 5\eta^2\frac{1}{n}\Big(\sum_{i=2}^{n_d^{(k)}} L_{\pi_i^{(k)}} \Big\|\sum_{j=1}^{i-1}\nabla f_{\pi_j^{(k)}}(\mathbf{x}^*)\Big\|^2 + \sum_{i=n_d^{(k)}+1}^{n} \widetilde{L}^{(k)}_{i-n_d^{(k)}} \Big\|\sum_{j=1}^{i-1}\nabla f_{\pi_j^{(k)}}(\mathbf{x}^*)\Big\|^2\Big) \\
&+ 5\eta^2\frac{1}{n}\Big(\sum_{i=2}^{n_d^{(k)}} L_{\pi_i^{(k)}} \Big\|\sum_{j=1}^{i-1}\rho_j^{(k)}\Big\|^2 + \sum_{i=n_d^{(k)}+1}^{n} \widetilde{L}^{(k)}_{i-n_d^{(k)}} \Big\|\sum_{j=1}^{i-1}\rho_j^{(k)}\Big\|^2\Big)
\end{aligned} \tag{C.27}
$$

$$+ 5\eta^2 L^{(k)*} \frac{1}{n} \sum_{i=n_d^{(k)}+1}^{n} \Big\| \sum_{j=n_d^{(k)}+1}^{i-1} \nabla f_{i-n_d^{(k)}}^{(k,pub)}(\mathbf{z}) - \sum_{j=n_d^{(k)}+1}^{i-1} \nabla f_{\pi_j^{(k)}}(\mathbf{z}) \Big\|^2$$

Since $\eta \leq \frac{1}{n\sqrt{10\widehat{L}^{(k)}\widehat{L}^{(k)*}}}$, there is $\widehat{L}^{(k)} \leq \sqrt{\widehat{L}^{(k)}\widehat{L}^{(k)*}} \leq \frac{1}{\sqrt{10}n\eta} \leq \frac{1}{2n\eta}$, and so $(\widehat{L}^{(k)} - \frac{1}{2n\eta})\|\mathbf{x}_1^{(k+1)} - \mathbf{x}_1^{(k)}\|^2 \leq 0$.

For any $\beta > 0$ and any $\mathbf{z} \in \mathbb{R}^d$,

$$\begin{aligned}
&H^{(k)}(\mathbf{x}_1^{(k+1)}) - H^{(k)}(\mathbf{z}) \\
&= H^{(k)}(\mathbf{x}_1^{(k+1)}) - H^{(k)}(\mathbf{z}) - \langle \nabla H^{(k)}(\mathbf{z}), \mathbf{x}_1^{(k+1)} - \mathbf{z} \rangle + \langle \nabla H^{(k)}(\mathbf{z}), \mathbf{x}_1^{(k+1)} - \mathbf{z} \rangle \\
&\overset{(a)}{\leq} \frac{L_H^{(k)}}{2} \|\mathbf{x}_1^{(k+1)} - \mathbf{z}\|^2 + \frac{1}{2n^2\beta}(C_n^{(k)})^2 + \frac{\beta}{2}\|\mathbf{x}_1^{(k+1)} - \mathbf{z}\|^2 \\
&= \frac{L_H^{(k)} + \beta}{2} \|\mathbf{x}_1^{(k+1)} - \mathbf{z}\|^2 + \frac{1}{2n^2\beta}(C_n^{(k)})^2
\end{aligned} \tag{C.28}$$

where (a) is by Assumption 4.4.5, 4.4.6, Lemma C.1.4 and Young's inequality.

We comment that if $H^{(k)} = 0$ for epoch s, a tighter bound $H^{(k)}(\mathbf{x}_1^{(k+1)}) - H^{(k)}(\mathbf{z}) = 0$ holds, as Young's inequality is not tight in this case. Consequently, one can set $\beta = 0$.

Finally, plugging Eq.C.28 back to Eq.C.27 yields the inequality (C.26) stated in the lemma.

□

C.1.3 Expected One Epoch Convergence

There are two sources of randomness involved in each epoch: 1) the shuffling operator in optimization, and 2) injected Gaussian noise to perturb the gradient for privacy preservation. 1) can be bounded using Lemma C.1.5. To bound 2), in this section, we show upper bounds on the expectation of the additional error term due to noise injection and the noise variance in Lemma C.1.10 and Lemma C.1.11. We then give an expected one epoch convergence bound, where the expectation is taken over the two sources of randomness, in Lemma C.1.12.

Lemma C.1.10 (Additional Error)**.** *For any epoch $s \in [K]$ and $\forall \mathbf{z} \in \mathbb{R}^d$, consider the injected noise $\rho_i^{(k)} \sim \mathcal{N}(0, (\sigma^{(k)})^2 \mathbb{I}_d)$, $\forall i \in [n]$, if the regularization function ψ is*

twice differentiable and $\mathbf{z}$ *is independent of* $\rho_i^{(k)}, \forall i \in [n]$*, then the error caused by noise injection in epoch* k *is*

$$\mathbb{E}\left[\frac{1}{n}\sum_{i=1}^{n}\langle \rho_i^{(k)}, \mathbf{x}_1^{(k+1)} - \mathbf{z}\rangle\right] \leq (\sigma^{(k)})^2 nd\eta^2 \widehat{L}^{(k)*} \tag{C.29}$$

where the expectation is taken w.r.t. the injected noise $\{\rho_i^{(k)}\}_{i=1}^n$*.*

Proof of Lemma C.1.10. First, note that if $\mathbf{z}$ is independent of $\rho_i^{(k)}$, $\forall i \in [n]$, there is $\mathbb{E}\left[\langle \frac{1}{n}\sum_{i=1}^n \rho_i^{(k)}, \mathbf{x}_1^{(k+1)} - \mathbf{z}\rangle\right] = \frac{1}{n}\sum_{i=1}^n \mathbb{E}\left[\langle \rho_i^{(k)}, \mathbf{x}_1^{(k+1)} - \mathbf{z}\rangle\right] = \frac{1}{n}\sum_{i=1}^n \mathbb{E}\left[\langle \rho_i^{(k)}, \mathbf{x}_1^{(k+1)}\rangle\right]$.

Recall that the update rule in Algorithm 3 in epoch $k \in [K]$ is

$$\begin{aligned}
\mathbf{x}_{i+1}^{(k)} &= \mathbf{x}_i^{(k)} - \eta\Big(\nabla f_{\pi_i^{(k)}} + \rho_i^{(k)}\Big), \quad \forall i \in [n_d^{(k)}] \\
\mathbf{x}_{i+1}^{(k)} &= \mathbf{x}_i^{(k)} - \eta\Big(\nabla f_{i-n_d^{(k)}}^{(k,pub)} + \rho_i^{(k)}\Big), \quad \forall n_d^{(k)} < i \leq n \\
\mathbf{x}_1^{(k+1)} &= \operatorname*{argmin}_{\mathbf{x}\in\mathbb{R}^d} n\psi(\mathbf{x}) + \frac{\|\mathbf{x} - \mathbf{x}_{n+1}^{(k)}\|^2}{2\eta}
\end{aligned} \tag{C.30}$$

Since ψ is twice differentiable, by Stein's Lemma (Lemma C.1.3), for any $i \in [n]$, conditional on $\rho_j^{(k)}, \forall j \neq i$,

$$\mathbb{E}_{\rho_i^{(k)}}\left[\langle \rho_i^{(k)}, \mathbf{x}_1^{(k+1)}\rangle \mid \{\rho_j^{(k)}\}_{j\neq i}\right] = (\sigma^{(k)})^2 \cdot \mathbb{E}_{\rho_i^{(k)}}\left[\operatorname{tr}\left(\frac{\partial \mathbf{x}_1^{(k+1)}}{\partial \rho_i^{(k)}}\right) \mid \{\rho_j^{(k)}\}_{j\neq i}\right] \tag{C.31}$$

We proceed by computing $\frac{\partial \mathbf{x}_1^{(k+1)}}{\partial \rho_i^{(k)}}$. By the optimality condition of $\mathbf{x}_1^{(k+1)}$ as in Eq. C.30,

$$\begin{aligned}
n\nabla\psi(\mathbf{x}_1^{(k+1)}) + \frac{1}{\eta}\cdot(\mathbf{x}_1^{(k+1)} - \mathbf{x}_{n+1}^{(k)}) &= \mathbf{0} \\
n\eta\nabla\psi(\mathbf{x}_1^{(k+1)}) + \mathbf{x}_1^{(k+1)} - \mathbf{x}_{n+1}^{(k)} &= \mathbf{0}
\end{aligned}$$

And using implicit differentiation of the above optimality condition,

$$n\eta\frac{\partial \nabla\psi(\mathbf{x}_1^{(k+1)})}{\partial \rho_i^{(k)}} + \frac{\partial \mathbf{x}_1^{(k+1)}}{\partial \rho_i^{(k)}} - \frac{\partial \mathbf{x}_{n+1}^{(k)}}{\partial \rho_i^{(k)}} = \mathbf{0}$$

$$
\begin{aligned}
&n\eta\nabla^2\psi(\mathbf{x}_1^{(k+1)})\frac{\partial \mathbf{x}_1^{(k+1)}}{\partial \rho_i^{(k)}} + \frac{\partial \mathbf{x}_1^{(k+1)}}{\partial \rho_i^{(k)}} - \frac{\partial \mathbf{x}_{n+1}^{(k)}}{\partial \rho_i^{(k)}} = \mathbf{0} \\
&\Big(n\eta\nabla^2\psi(\mathbf{x}_1^{(k+1)}) + \mathbb{I}_d\Big)\frac{\partial \mathbf{x}_1^{(k+1)}}{\partial \rho_i^{(k)}} = \frac{\partial \mathbf{x}_{n+1}^{(k)}}{\partial \rho_i^{(k)}} \\
&\frac{\partial \mathbf{x}_1^{(k+1)}}{\partial \rho_i^{(k)}} = \Big(\eta n\nabla^2\psi(\mathbf{x}_1^{(k+1)}) + \mathbb{I}_d\Big)^{-1}\frac{\partial \mathbf{x}_{n+1}^{(k)}}{\partial \rho_i^{(k)}}
\end{aligned} \tag{C.32}
$$

where $\nabla^2\psi(\mathbf{x}_1^{(k+1)})$ is the Hessian of ψ evaluated at $\mathbf{x}_1^{(k+1)}$.

We proceed by computing $\frac{\partial \mathbf{x}_{n+1}^{(k)}}{\partial \rho_i^{(k)}}$. Note that $\rho_i^{(k)}$ directly affects the update of $\mathbf{x}_{i+1}^{(k)}$ and indirectly affects the subsequent updates of $\mathbf{x}_j^{(k)}$ for all $j > i+1$. Hence, we decompose $\mathbf{x}_{n+1}^{(k)}$ as follows: for $i \le n_d^{(k)}$,

$$
\begin{aligned}
\mathbf{x}_{n+1}^{(k)} = &\underbrace{\mathbf{x}_1 - \eta\sum_{j=1}^{i-1}\Big(\nabla f_{\pi_j^{(k)}}(\mathbf{x}_j^{(k)}) + \rho_j^{(k)}\Big) - \nabla f_{\pi_i^{(k)}}(\mathbf{x}_i^{(k)})}_{\text{Independent of } \rho_i^{(k)}} \\
&- \underbrace{\rho_i^{(k)}}_{\text{Direct dependency}} \underbrace{- \eta\sum_{j=i+1}^{n_d^{(k)}}\Big(\nabla f_{\pi_j^{(k)}}(\mathbf{x}_j^{(k)}) + \rho_j^{(k)}\Big) - \eta\sum_{j=n_d^{(k)}+1}^{n}\Big(\nabla f_{j-n_d^{(k)}}^{(k,pub)}(\mathbf{x}_j^{(k)}) + \rho_j^{(k)}\Big)}_{\text{Implicit dependency on } \rho_i^{(k)} \text{ through } \mathbf{x}_j^{(k)}\text{'s}}
\end{aligned}
$$

and for $n_d^{(k)} < i \le n$,

$$
\begin{aligned}
\mathbf{x}_{n+1}^{(k)} = &\underbrace{\mathbf{x}_1 - \eta\sum_{j=1}^{n_d^{(k)}}\Big(\nabla f_{\pi_j^{(k)}}(\mathbf{x}_j^{(k)}) + \rho_j^{(k)}\Big) - \eta\sum_{j=n_d^{(k)}+1}^{i-1}\Big(\nabla f_{j-n_d^{(k)}}^{(k,pub)}(\mathbf{x}_j^{(k)}) + \rho_j^{(k)}\Big) - \nabla f_i^{(k,pub)}(\mathbf{x}_i^{(k)})}_{\text{Independent of } \rho_i^{(k)} \text{ through } \mathbf{x}_j^{(k)}\text{'s}} \\
&- \underbrace{\rho_i^{(k)}}_{\text{Direct dependency}} \underbrace{- \eta\sum_{j=i+1}^{n}\Big(\nabla f_{j-n_d^{(k)}}^{(k,pub)}(\mathbf{x}_j^{(k)}) + \rho_j^{(k)}\Big)}_{\text{Implicit dependency on } \rho_i^{(k)}}
\end{aligned}
$$

And so for $i \leq n_d^{(k)}$,

$$\begin{aligned}\frac{\partial \mathbf{x}_{n+1}^{(k)}}{\partial \rho_i^{(k)}} &= -\eta \mathbb{I}_d - \eta \sum_{j=i+1}^{n_d^{(k)}} \frac{\partial \nabla f_{\pi_j^{(k)}}(\mathbf{x}_j^{(k)})}{\partial \rho_i^{(k)}} - \eta \sum_{j=n_d^{(k)}+1}^{n} \frac{\partial \nabla f_{j-n_d^{(k)}}^{(k,pub)}(\mathbf{x}_j^{(k)})}{\partial \rho_i^{(k)}} \\ &= -\eta \mathbb{I}_d - \eta \sum_{j=i+1}^{n_d^{(k)}} \nabla^2 f_{\pi_j^{(k)}}(\mathbf{x}_j^{(k)}) \frac{\partial \mathbf{x}_j^{(k)}}{\partial \rho_i^{(k)}} - \eta \sum_{j=n_d^{(k)}+1}^{n} \nabla^2 f_{j-n_d^{(k)}}^{(k,pub)}(\mathbf{x}_j^{(k)}) \frac{\partial \mathbf{x}_j^{(k)}}{\partial \rho_i^{(k)}} \end{aligned} \tag{C.33}$$

and for $n_d^{(k)} < i \leq n$,

$$\frac{\mathbf{x}_{n+1}^{(k)}}{\partial \rho_i^{(k)}} = -\eta \mathbb{I}_d - \eta \sum_{j=i+1}^{n} \frac{\partial \nabla f_{j-n_d^{(k)}}^{(k,pub)}(\mathbf{x}_j^{(k)})}{\partial \rho_i^{(k)}} = -\eta \mathbb{I}_d - \eta \sum_{j=i+1}^{n} \nabla^2 f_{j-n_d^{(k)}}^{(k,pub)}(\mathbf{x}_j^{(k)}) \frac{\partial \mathbf{x}_j^{(k)}}{\partial \rho_i^{(k)}} \tag{C.34}$$

We now compute $\frac{\partial \mathbf{x}_j^{(k)}}{\partial \rho_i^{(k)}}$ for $j > i$. First, note that by the update rule,

$$\frac{\partial \mathbf{x}_{i+1}^{(k)}}{\partial \rho_i^{(k)}} = -\eta \mathbb{I}_d \tag{C.35}$$

and for any $i < j \leq n$,

$$\begin{aligned}\frac{\partial \mathbf{x}_{j+1}^{(k)}}{\partial \rho_i^{(k)}} &= \begin{cases} \frac{\partial}{\partial \rho_i^{(k)}} \left(\mathbf{x}_j^{(k)} - \eta \left(\nabla f_{\pi_j^{(k)}}(\mathbf{x}_j^{(k)}) + \rho_j^{(k)} \right) \right) & \text{if } j \leq n_d^{(k)} \\ \frac{\partial}{\partial \rho_i^{(k)}} \left(\mathbf{x}_{j-n_d^{(k)}}^{(k,pub)}(\mathbf{x}_j^{(k)}) + \rho_j^{(k)} \right) & \text{Otherwise} \end{cases} \\ \Rightarrow \frac{\partial \mathbf{x}_{j+1}^{(k)}}{\partial \rho_i^{(k)}} &= \begin{cases} \left(\mathbb{I}_d - \eta \nabla^2 f_{\pi_j^{(k)}}(\mathbf{x}_j^{(k)}) \right) \frac{\partial \mathbf{x}_j^{(k)}}{\partial \rho_i^{(k)}} & \text{if } j \leq n_d^{(k)} \\ \left(\mathbb{I}_d - \eta \nabla^2 f_{j-n_d^{(k)}}^{(k,pub)}(\mathbf{x}_j^{(k)}) \right) \frac{\partial \mathbf{x}_j^{(k)}}{\partial \rho_i^{(k)}} & \text{Otherwise} \end{cases} \end{aligned}$$

Therefore, by the above recursion, for any $i < j \leq n_d^{(k)}$,

$$\frac{\partial \mathbf{x}_{j+1}^{(k)}}{\partial \rho_i^{(k)}} = -\eta \cdot \prod_{l=i+1}^{j} \left(\mathbb{I}_d - \eta \nabla^2 f_{\pi_l^{(k)}}(\mathbf{x}_l^{(k)}) \right) \tag{C.36}$$

and similarity, for any $n_d^{(k)} < j \leq n$,

$$\frac{\partial \mathbf{x}_{j+1}^{(k)}}{\partial \rho_i^{(k)}} = -\eta \cdot \Big(\prod_{l=i+1}^{n_d^{(k)}} \Big(\mathbb{I}_d - \eta \nabla^2 f_{\pi_l^{(k)}}(\mathbf{x}_l^{(k)}) \Big) \Big) \cdot \Big(\prod_{l=n_d^{(k)}+1}^{j} \Big(\mathbb{I}_d - \eta \nabla^2 f_{l-n_d^{(k)}}^{(k,pub)}(\mathbf{x}_l^{(k)}) \Big) \Big) \tag{C.37}$$

Therefore, plugging Eq. C.35, Eq. C.36 and Eq. C.37 back to Eq. C.33 or Eq. C.34, there is, for $i \leq n_d^{(k)}$,

$$\begin{aligned}
\frac{\partial \mathbf{x}_{n+1}^{(k)}}{\partial \rho_i^{(k)}} &= -\eta \mathbb{I}_d + \eta^2 \nabla^2 f_{\pi_{i+1}^{(k)}}(\mathbf{x}_{i+1}^{(k)}) + \eta^2 \sum_{j=i+2}^{n_d^{(k)}} \nabla^2 f_{\pi_j^{(k)}}(\mathbf{x}_j^{(k)}) \prod_{l=i+1}^{j-1} \Big(\mathbb{I}_d - \eta \nabla^2 f_{\pi_l^{(k)}}(\mathbf{x}_l^{(k)}) \Big) \\
&\quad + \eta^2 \nabla^2 f_1^{(k,pub)}(\mathbf{x}_{n_d^{(k)}+1}^{(k)}) \prod_{l=i+1}^{n_d^{(k)}} \Big(\mathbb{I}_d - \eta \nabla^2 f_{\pi_l^{(k)}}(\mathbf{x}_l^{(k)}) \Big) \\
&\quad + \eta^2 \sum_{j=n_d^{(k)}+2}^{n} \nabla^2 f_{j-n_d^{(k)}}^{(k,pub)}(\mathbf{x}_j^{(k)}) \Big(\prod_{l=i+1}^{n_d^{(k)}} \Big(\mathbb{I}_d - \eta \nabla^2 f_{\pi_l^{(k)}}(\mathbf{x}_l^{(k)}) \Big) \Big) \\
&\quad \cdot \Big(\prod_{l=n_d^{(k)}+1}^{j-1} \Big(\mathbb{I}_d - \eta \nabla f_{l-n_d^{(k)}}^{(k,pub)}(\mathbf{x}_l^{(k)}) \Big) \Big)
\end{aligned}$$

and for $n_d^{(k)} < i \leq n$,

$$\begin{aligned}
\frac{\mathbf{x}_{n+1}^{(k)}}{\partial \rho_i^{(k)}} &= -\eta \mathbb{I}_d + \eta^2 \nabla^2 f_{i+1-n_d^{(k)}}^{(k,pub)}(\mathbf{x}_{i+1}^{(k)}) + \eta^2 \sum_{j=i+2}^{n} \nabla^2 f_{j-n_d^{(k)}}^{(k,pub)}(\mathbf{x}_j^{(k)}) \\
&\qquad \cdot \Big(\prod_{i+1}^{n_d^{(k)}} \Big(\mathbb{I}_d - \eta \nabla^2 f_{\pi_k^{(k)}}(\mathbf{x}_k^{(k)}) \Big) \Big) \cdot \Big(\prod_{l=n_d^{(k)}+1}^{j-1} \Big(\mathbb{I}_d - \eta \nabla f_{l-n_d^{(k)}}^{(k,pub)}(\mathbf{x}_l^{(k)}) \Big) \Big)
\end{aligned}$$

By Assumption 4.4.1 and 4.4.2, $\|\nabla^2 f_i(\mathbf{x})\|_{op} \leq \widehat{L}^{(k)*}$ and $\|\nabla^2 f_j^{(k,pub)}(\mathbf{x})\|_{op} \leq \widehat{L}^{(k)*}$, $\forall i \in [n_d^{(k)}], j \in [n - n_d^{(k)}]$ and $\forall \mathbf{x} \in \mathbb{R}^d$, where $\|\cdot\|_{op}$ denotes the matrix operator norm and recall that $\widehat{L}^{(k)*} = \max\{\{L_{\pi_i^{(k)}}\}_{i=1}^{n_d^{(k)}} \cup \{\widetilde{L}_i^{(k)}\}_{i=1}^{n-n_d^{(k)}}\}$ is the maximum smoothness parameter in epoch k.

And so if $\eta \leq \frac{1}{\widehat{L}^{(k)*}}$, $\forall i \in [n]$,

$$\left\| \frac{\mathbf{x}_{n+1}^{(k)}}{\partial \rho_i^{(k)}} \right\|_{op} \leq n\eta^2 \widehat{L}^{(k)*} \tag{C.38}$$

Moreover, by Assumption 4.4.4, $\lambda_{min}(\nabla^2 \psi(\mathbf{x})) \geq 0$, $\forall \mathbf{x} \in \mathbb{R}^d$, where λ_{min} denotes the minimum eigenvalue. And so

$$\left\| \left(\mathbb{I}_d + \eta n \nabla^2 \psi(\mathbf{x}_1^{(k+1)}) \right)^{-1} \right\|_{op} \leq 1$$

Hence, by Eq. C.32,

$$\left\| \frac{\partial \mathbf{x}_1^{(k)}}{\partial \rho_i^{(k)}} \right\|_{op} = \left\| \left(\eta n \nabla^2 \psi(\mathbf{x}_1^{(k+1)}) + \mathbb{I}_d \right)^{-1} \frac{\partial \mathbf{x}_{n+1}^{(k)}}{\partial \rho_i^{(k)}} \right\|_{op} \leq n\eta^2 \widehat{L}^{(k)*}$$

Since for some symmetric real matrix $\boldsymbol{A}$, $\mathrm{tr}(\boldsymbol{A}) \leq d\|\boldsymbol{A}\|_{op}$,

$$\mathrm{tr}\left(\frac{\partial \mathbf{x}_1^{(k+1)}}{\partial \rho_i^{(k)}}\right) \leq nd\eta^2 \widehat{L}^{(k)*}$$

Hence, by Eq. C.31, for any $i \in [n]$, conditional on $\rho_j^{(k)}, \forall j \neq i$,

$$\begin{aligned} \mathbb{E}_{\rho_i^{(k)}} \left[\langle \rho_i^{(k)}, \mathbf{x}_1^{(k+1)} \rangle \mid \{\rho_j^{(k)}\}_{j \neq i} \right] &= (\sigma^{(k)})^2 \cdot \mathbb{E}_{\rho_i^{(k)}} \left[\mathrm{tr}\left(\frac{\partial \mathbf{x}_1^{(k+1)}}{\partial \rho_i^{(k)}}\right) \mid \{\rho_j^{(k)}\}_{j \neq i} \right] \\ &\leq (\sigma^{(k)})^2 nd\eta^2 \widehat{L}^{(k)*} \end{aligned}$$

and by law of total expectation,

$$\mathbb{E}\left[\langle \rho_i^{(k)}, \mathbf{x}_1^{(k+1)} \rangle \right] = \mathbb{E}\left[\mathbb{E}_{\rho_i^{(k)}} \left[\langle \rho_i^{(k)}, \mathbf{x}_1^{(k+1)} \rangle \mid \{\rho_j\}_{j \neq i} \right] \right] \leq (\sigma^{(k)})^2 nd\eta^2 \widehat{L}^{(k)*}$$

and so

$$\mathbb{E}\left[\frac{1}{n} \sum_{i=1}^{n} \langle \rho_i^{(k)}, \mathbf{x}_1^{(k+1)} - \mathbf{z} \rangle \right] \leq (\sigma^{(k)})^2 nd\eta^2 \widehat{L}^{(k)*}$$

□

Lemma C.1.11 (Noise Variance). *For any epoch $k \in [K]$ and $\forall \mathbf{z} \in \mathbb{R}^d$, consider the injected noise $\rho_i^{(k)} \sim \mathcal{N}(0, (\sigma^{(k)})^2 \mathbb{I}_d)$, $\forall i \in [n]$, the variance caused by noise injection in epoch k is, $\forall i \in [n]$,*

$$\mathbb{E}\left[\left\|\sum_{j=1}^{i} \rho_j^{(k)}\right\|^2\right] \leq id(\sigma^{(k)})^2$$

where the expectation is taken w.r.t. the injected noise $\{\rho_i^{(k)}\}_{i=1}^n$.

Proof of Lemma C.1.11. First, note that $\rho_i^{(k)}$ and $\rho_j^{(k)}$, i.e., the noise injected at step i and step j in epoch k, are independent, for any $i \neq j$. Thus,

$$\begin{aligned}
\mathbb{E}\left[\left\|\sum_{j=1}^{i} \rho_j^{(k)}\right\|^2\right] &= \sum_{j=1}^{i} \mathbb{E}\left[\left\|\rho_j^{(k)}\right\|^2\right] + 2\sum_{j=1}^{i}\sum_{k=j+1}^{i} \mathbb{E}\left[\langle \rho_i^{(k)}, \rho_j^{(k)} \rangle\right] \\
&= \sum_{j=1}^{i} \mathbb{E}\left[\left\|\rho_j^{(k)}\right\|^2\right] \\
&\leq id(\sigma^{(k)})^2
\end{aligned}$$

□

Lemma C.1.12 (Expected One Epoch Convergence). *Under Assumptions 4.4.1, 4.4.4, 4.4.6, C.1.1 and 4.4.5, for any epoch $k \in [K]$, $\beta > 0$ and $\forall \mathbf{z} \in \mathbb{R}^d$, if $\eta \leq \frac{1}{n\sqrt{10\widehat{L}^{(k)}\widehat{L}^{(k)*}}}$ and $\mathbf{z}$ is independent of $\rho_i^{(k)}$, $\forall i \in [n]$, Algorithm 3 guarantees*

$$\begin{aligned}
\mathbb{E}\left[G(\mathbf{x}_1^{(k+1)})\right] - \mathbb{E}\left[G(\mathbf{z})\right] &\leq \frac{1}{2n\eta}\left(\mathbb{E}\left[\|\mathbf{z} - \mathbf{x}_1^{(k)}\|^2\right] - \mathbb{E}\left[\|\mathbf{z} - \mathbf{x}_1^{(k+1)}\|^2\right]\right) \qquad \text{(C.39)} \\
&+ \left(\frac{L_H^{(k)} + \beta}{2} - \frac{\mu_\psi}{2}\right)\mathbb{E}\left[\|\mathbf{z} - \mathbf{x}_1^{(k+1)}\|^2\right] + 10\eta^2 n^2 \widehat{L}^{(k)} L \mathbb{E}\left[B_F(\mathbf{z}, \mathbf{x}^*)\right] \\
&+ \underbrace{5\eta^2 n^2 \widehat{L}^{(k)} \sigma_{any}^2}_{\text{Optimization Uncertainty}} + \underbrace{\frac{1}{2n^2\beta}(C_n^{(k)})^2}_{\text{Non-vanishing Dissimilarity}} + \underbrace{5\eta^2 \widehat{L}^{(k)*} \frac{1}{n} \sum_{i=n_d^{(k)}+1}^{n-1} (C_i^{(k)})^2}_{\text{Vanishing Dissimilarity}}
\end{aligned}$$

$$+\underbrace{6\eta^2 nd(\sigma^{(k)})^2\widehat{L}^{(k)*}}_{\textit{Injected Noise}}$$

where the expectation is taken w.r.t. both the injected noise within epoch k, i.e., $\{\rho_i^{(k)}\}_{i=1}^n$, and the shuffling operator $\pi^{(k)}$.

Proof of C.1.12. By Assumption 4.4.6,

$$\begin{aligned}&\frac{1}{n}\sum_{i=n_d^{(k)}+1}^{n}\mathbb{E}_{\pi^{(k)}}\Big\|\sum_{j=n_d^{(k)}+1}^{i-1}\nabla f^{(k,pub)}_{i-n_d^{(k)}}(\mathbf{z})-\sum_{j=n_d^{(k)}+1}^{i-1}\nabla f_{\pi_j^{(k)}}(\mathbf{z})\Big\|^2 \\ &\quad\le\frac{1}{n}\sum_{i=n_d^{(k)}+1}^{n-1}(C_i^{(k)})^2\end{aligned} \tag{C.40}$$

and by Lemma C.1.5, for any permutation $\pi^{(k)}\in\Pi_n$, there is

$$\begin{aligned}&\mathbb{E}_{\pi^{(k)}}\Big[\frac{1}{n}\Big(\sum_{i=2}^{n_d^{(k)}}L_{\pi_i^{(k)}}\Big\|\sum_{j=1}^{i-1}\nabla f_{\pi_j^{(k)}}(\mathbf{x}^*)\Big\|^2+\sum_{i=n_d^{(k)}+1}^{n}\widetilde{L}^{(k)}_{i-n_d^{(k)}}\Big\|\sum_{j=1}^{i-1}\nabla f_{\pi_j^{(k)}}(\mathbf{x}^*)\Big\|^2\Big)\Big] \\ &\le n^2\widehat{L}^{(k)}\sigma_{any}^2\end{aligned} \tag{C.41}$$

where the expectation is taken w.r.t. the shuffling operator $\pi^{(k)}$.

Moreover, by Lemma C.1.11,

$$\begin{aligned}&\mathbb{E}_{\pi^{(k)}}\Big[\frac{1}{n}\Big(\sum_{i=2}^{n_d^{(k)}}L_{\pi_i^{(k)}}\Big\|\sum_{j=1}^{i-1}\rho_j^{(k)}\Big\|^2+\sum_{i=n_d^{(k)}+1}^{n}\widetilde{L}^{(k)}_{i-n_d^{(k)}}\Big\|\sum_{j=1}^{i-1}\rho_j^{(k)}\Big\|^2\Big)\Big] \\ &\le\frac{1}{n}\Big(\sum_{i=2}^{n_d^{(k)}}L_{\pi_i^{(k)}}(i-1)d(\sigma^{(k)})^2+\sum_{i=n_d^{(k)}+1}^{n}\widetilde{L}^{(k)}_{i-n_d^{(k)}}(i-1)d(\sigma^{(k)})^2\Big) \\ &\le\frac{1}{n}nd(\sigma^{(k)})^2\Big(\sum_{i=2}^{n_d^{(k)}}L_{\pi_i^{(k)}}+\sum_{i=n_d^{(k)}+1}^{n}\widetilde{L}^{(k)}_{i-n_d^{(k)}}\Big)\le nd(\sigma^{(k)})^2\widehat{L}^{(k)}\end{aligned} \tag{C.42}$$

where the expectation is taken w.r.t. the injected noise $\{\rho_j^{(k)}\}_{j=1}^n$.

Following Eq. C.40, C.41, C.42, and Lemma C.1.9, C.1.10, for any $\mathbf{z}\in\mathbb{R}^d$,

$$\mathbb{E}\left[G(\mathbf{x}_1^{(k+1)})\right] - \mathbb{E}\left[G(\mathbf{z})\right] \tag{C.43}$$
$$\leq \frac{1}{2n\eta}\Big(\mathbb{E}\left[\|\mathbf{z}-\mathbf{x}_1^{(k)}\|^2\right] - \mathbb{E}\left[\|\mathbf{z}-\mathbf{x}_1^{(k+1)}\|^2\right]\Big) + \Big(\frac{L_H^{(k)}+\beta}{2} - \frac{\mu_\psi}{2}\Big)\mathbb{E}\left[\|\mathbf{z}-\mathbf{x}_1^{(k+1)}\|^2\right]$$
$$+ 10\eta^2 n^2 \widehat{L}^{(k)} L \mathbb{E}\left[B_F(\mathbf{z},\mathbf{x}^*)\right] + \frac{1}{2n^2\beta}(C_n^{(k)})^2 + 5\eta^2\widehat{L}^{(k)*}\frac{1}{n}\sum_{i=n_d^{(k)}+1}^{n-1}(C_i^{(k)})^2$$
$$+ 5\eta^2 n^2 \widehat{L}^{(k)}\sigma_{any}^2 + \eta^2(\sigma^{(k)})^2 nd\widehat{L}^{(k)*} + 5\eta^2 nd(\sigma^{(k)})^2\widehat{L}^{(k)}$$

where the expectation is taken w.r.t. both the injected noise within epoch k, i.e., $\{\rho_i^{(k)}\}_{i=1}^n$, and the shuffling operator $\pi^{(k)}$. Combining the last two terms and note that $\widehat{L}^{(k)} \leq \widehat{L}^{(k)*}$ yields the inequality (C.39) in the above lemma statement.

□

C.1.4 Convergence Across K Epochs

Now that we have the expected convergence rate for one epoch, we follow a similar approach as in [76], first showing the convergence for any arbitrary number of epochs $s \in [K]$ by picking proper $\mathbf{z}$ points as the virtual sequence and using a weighted telescoping sum in Lemma C.1.13 and then showing the convergence for K epochs in Theorem C.1.14.

Lemma C.1.13 (Convergence Across Arbitrary Epochs)**.** *Under Assumptions 4.4.1, 4.4.4, 4.4.6, C.1.1 and 4.4.5, for any number of epochs $s \in [K]$ and $\beta > 0$, if $\mu_\psi \geq L_H^{(k)} + \beta$, $\forall k \in [s]$, and $\eta \leq \frac{1}{n\sqrt{10\max_{k\in[s]}(\widehat{L}^{(k)}\widehat{L}^{(k)*})}}$, Algorithm 3 guarantees*

$$\mathbb{E}\left[G(\mathbf{x}_1^{(s+1)})\right] - G(\mathbf{x}^*) \leq \frac{1}{2\eta n s}\|\mathbf{x}^* - \mathbf{x}_1^{(1)}\|^2$$
$$+ 10\eta^2 n^2 L \max_{k\in[s]}\widehat{L}^{(k)}\sum_{k=2}^{s}\frac{1}{s+2-k}\mathbb{E}\left[B_F(\mathbf{x}_1^{(k)},\mathbf{x}^*)\right]$$
$$+ 5\eta^2 n^2\sigma_{any}^2\sum_{k=1}^{s}\frac{\widehat{L}^{(k)}}{s+1-k} + \frac{1}{2n^2\beta}\sum_{k=1}^{s}\frac{(C_n^{(k)})^2}{s+1-k}$$

$$+ 5\eta^2 \sum_{k=1}^{s} \frac{\widehat{L}^{(k)*} \frac{1}{n} \sum_{i=n_d^{(k)}+1}^{n-1} (C_i^{(k)})^2}{s+1-k} + 6\eta^2 nd \sum_{k=1}^{s} \frac{(\sigma^{(k)})^2 \widehat{L}^{(k)*}}{s+1-k}$$

Proof of Lemma C.1.13. Fix an arbitrary number of epochs $s \in [K]$.

If $\mu_\psi \geq L_H^{(k)} + \beta$, $\forall k \in [s]$, $\eta \leq \frac{1}{n\sqrt{10 \max_{k\in[s]} (\widehat{L}^{(k)} \widehat{L}^{(k)*})}}$, and $\mathbf{z}$ independent of $\{\rho_i^{(k)}\}_{i=1}^n$, then by Lemma C.1.12, for any $k \in [s]$,

$$\begin{aligned} &\mathbb{E}\left[G(\mathbf{x}_1^{(k+1)})\right] - \mathbb{E}\left[G(\mathbf{z})\right] \\ &\leq \frac{1}{2n\eta}\Big(\mathbb{E}\left[\|\mathbf{z} - \mathbf{x}_1^{(k)}\|^2\right] - \mathbb{E}\left[\|\mathbf{z} - \mathbf{x}_1^{(k+1)}\|^2\right]\Big) + 10\eta^2 n^2 \widehat{L}^{(k)} L \mathbb{E}\left[B_F(\mathbf{z}, \mathbf{x}^*)\right] \\ &\quad + 5\eta^2 n^2 \widehat{L}^{(k)} \sigma_{any}^2 + \frac{1}{2n^2\beta}(C_n^{(k)})^2 + 5\eta^2 \widehat{L}^{(k)*} \frac{1}{n} \sum_{i=n_d^{(k)}+1}^{n-1} (C_i^{(k)})^2 + 6\eta^2 nd (\sigma^{(k)})^2 \widehat{L}^{(k)*} \end{aligned} \tag{C.44}$$

Define the non-decreasing sequence

$$v_k = \frac{1}{s+1-k}, \forall k \in [s], \quad v_0 = v_1 = \frac{1}{s}$$

and the auxiliary points

$$\mathbf{z}^{(0)} = \mathbf{x}^*, \quad \mathbf{z}^{(k)} = \Big(1 - \frac{v_{k-1}}{v_k}\Big)\mathbf{x}_1^{(k)} + \frac{v_{k-1}}{v_k}\mathbf{z}^{(k-1)}, \forall k \in [s] \tag{C.45}$$

Equivalently, $\mathbf{z}^{(k)}$ can be re-written as

$$\mathbf{z}^{(k)} = \frac{v_0}{v_k}\mathbf{x}^* + \sum_{l=1}^{k} \frac{v_l - v_{l-1}}{v_k} \mathbf{x}_1^{(l)}, \forall k \in [0] \cup [s] \tag{C.46}$$

Note that for an epoch k, $\mathbf{z}^{(k)}$ only depends on $\mathbf{x}^*$ and $\mathbf{x}_1^{(l)}$, for $l \leq k$, and hence by the update rule, $\mathbf{z}^{(k)}$ is independent of $\rho_i^{(k)}$, $\forall i \in [n]$. And so for any $k \in [s]$, we can choose $\mathbf{z} = \mathbf{z}^{(k)}$ in Eq. C.44 and this leads to

$$\begin{aligned} &\mathbb{E}\left[G(\mathbf{x}_1^{(k+1)})\right] - \mathbb{E}\left[G(\mathbf{z})\right] \\ &\leq \frac{1}{2n\eta}\Big(\mathbb{E}\left[\|\mathbf{z}^{(k)} - \mathbf{x}_1^{(k)}\|^2\right] - \mathbb{E}\left[\|\mathbf{z}^{(k)} - \mathbf{x}_1^{(k+1)}\|^2\right]\Big) + 10\eta^2 n^2 \widehat{L}^{(k)} L \mathbb{E}\left[B_F(\mathbf{z}^{(k)}, \mathbf{x}^*)\right] \end{aligned} \tag{C.47}$$

$$+ 5\eta^2 n^2 \widehat{L}^{(k)} \sigma_{any}^2 + \frac{1}{2n^2\beta}(C_n^{(k)})^2 + 5\eta^2 \widehat{L}^{(k)*} \frac{1}{n} \sum_{i=n_d^{(k)}+1}^{n-1} (C_i^{(k)})^2 + 6\eta^2 nd(\sigma^{(k)})^2 \widehat{L}^{(k)*}$$

Note that by Eq. C.45,

$$\|\mathbf{z}^{(k)} - \mathbf{x}_1^{(k)}\|^2 = \frac{v_{k-1}^2}{v_k^2}\|\mathbf{z}^{(k-1)} - \mathbf{x}_1^{(k)}\|^2 \leq \frac{v_{k-1}}{v_k}\|\mathbf{z}^{(k-1)} - \mathbf{x}_1^{(k)}\|^2 \tag{C.48}$$

where the last inequality is due to $v_{k-1} \leq v_k$. Hence, following Eq. C.47,

$$\begin{aligned} & v_k \cdot \left(\mathbb{E}\left[G(\mathbf{x}_1^{(k+1)})\right] - \mathbb{E}\left[G(\mathbf{z})\right]\right) \\ & \leq \frac{1}{2n\eta}\left(\mathbb{E}\left[v_{k-1}\|\mathbf{z}^{(k-1)} - \mathbf{x}_1^{(k)}\|^2\right] - v_k \mathbb{E}\left[\|\mathbf{z}^{(k)} - \mathbf{x}_1^{(k+1)}\|^2\right]\right) \\ & \quad + 10 v_k \eta^2 n^2 \widehat{L}^{(k)} L \mathbb{E}\left[B_F(\mathbf{z}^{(k)}, \mathbf{x}^*)\right] + 5 v_k \eta^2 n^2 \widehat{L}^{(k)} \sigma_{any}^2 \\ & \quad + v_k \frac{1}{2n^2\beta}(C_n^{(k)})^2 + 5 v_k \eta^2 \widehat{L}^{(k)*} \frac{1}{n} \sum_{i=n_d^{(k)}+1}^{n-1} (C_i^{(k)})^2 + 6 v_k \eta^2 nd(\sigma^{(k)})^2 \widehat{L}^{(k)*} \end{aligned} \tag{C.49}$$

Summing Eq. C.49 from $k = 1$ to s to obtain

$$\begin{aligned} & \sum_{k=1}^{s} v_k \cdot \left(\mathbb{E}\left[G(\mathbf{x}_1^{(k+1)})\right] - \mathbb{E}\left[G(\mathbf{z})\right]\right) \\ & \leq \frac{1}{2n\eta}\left(\mathbb{E}\left[v_0\|\mathbf{z}^{(0)} - \mathbf{z}_1^{(1)}\|^2\right] - v_s \mathbb{E}\left[\|\mathbf{z}^{(s)} - \mathbf{x}_1^{(s+1)}\|^2\right]\right) \\ & \quad + 10\eta^2 n^2 L \sum_{k=1}^{s} \widehat{L}^{(k)} v_k \mathbb{E}\left[B_F(\mathbf{z}^{(k)}, \mathbf{x}^*)\right] + 5\eta^2 n^2 \sigma_{any}^2 \sum_{k=1}^{s} v_k \widehat{L}^{(k)} + \frac{1}{2n^2\beta}\sum_{k=1}^{s} v_k (C_n^{(k)})^2 \\ & \quad + 5\eta^2 \sum_{k=1}^{s} v_k \widehat{L}^{(k)*} \frac{1}{n} \sum_{i=n_d^{(k)}+1}^{n-1} (C_i^{(k)})^2 + 6\eta^2 nd \sum_{k=1}^{s} v_k (\sigma^{(k)})^2 \widehat{L}^{(k)*} \end{aligned} \tag{C.50}$$

Note since $\|\mathbf{z}^{(s)} - \mathbf{x}_1^{(s+1)}\|^2 \geq 0$ and $\mathbf{z}^{(0)} = \mathbf{x}^*, v_0 = \frac{1}{s}$,

$$\frac{1}{2n\eta}\left(v_0 \mathbb{E}\left[\|\mathbf{z}^{(0)} - \mathbf{x}_1^{(1)}\|^2\right] - v_s \mathbb{E}\left[\|\mathbf{z}^{(s)} - \mathbf{x}_1^{(s+1)}\|^2\right]\right) \leq \frac{1}{2\eta n s}\|\mathbf{x}^* - \mathbf{x}_1^{(1)}\|^2 \tag{C.51}$$

We first bound the L.H.S. of Eq. C.50. By Assumption 4.4.1 and 4.4.4, the

objective function $G(\mathbf{x}) = F(\mathbf{x}) + \psi(\mathbf{x})$ is convex, and hence, by Eq. C.46, there is

$$G(\mathbf{z}^{(k)}) \leq \frac{v_0}{v_k} G(\mathbf{x}^*) + \sum_{l=1}^{k} \frac{v_l - v_{l-1}}{v_k} G(\mathbf{x}_1^{(l)}) = G(\mathbf{x}^*) + \sum_{l=1}^{k} \frac{v_l - v_{l-1}}{v_k} \Big(G(\mathbf{x}_1^{(l)}) - G(\mathbf{x}^*) \Big)$$

which implies

$$\begin{aligned}
&\sum_{k=1}^{s} v_k \Big(G(\mathbf{x}_1^{(k+1)}) - G(\mathbf{z}^{(k)}) \Big) \\
&\geq \sum_{k=1}^{s} \Big(v_k \Big(G(\mathbf{x}_1^{(k+1)}) - G(\mathbf{x}^*) \Big) - \sum_{l=1}^{k} (v_l - v_{l-1}) \Big(G(\mathbf{x}_1^{(l)}) - G(\mathbf{x}^*) \Big) \Big) \\
&\geq \sum_{k=1}^{s} v_k \Big(G(\mathbf{x}_1^{(k+1)}) - G(\mathbf{x}^*) \Big) - \sum_{k=1}^{s} \sum_{l=1}^{k} (v_l - v_{l-1}) \Big(G(\mathbf{x}_1^{(l)}) - G(\mathbf{x}^*) \Big) \\
&= \sum_{k=1}^{s} v_k \Big(G(\mathbf{x}_1^{(k+1)}) - G(\mathbf{x}^*) \Big) - \sum_{l=1}^{s} (s+1-l)(v_l - v_{l-1}) \Big(G(\mathbf{x}_1^{(l)}) - G(\mathbf{x}^*) \Big) \\
&= v_s \Big(G(\mathbf{x}_1^{(s+1)}) - G(\mathbf{x}^*) \Big) + \sum_{k=1}^{s-1} \frac{1}{s+1-k} \Big(G(\mathbf{x}_1^{(k+1)}) - G(\mathbf{x}^*) \Big) \\
&\quad - s(v_1 - v_0) \Big(G(\mathbf{x}_1^{(1)}) - G(\mathbf{x}^*) \Big) \\
&\quad - \sum_{l=2}^{s} (s+1-l) \Big(\frac{1}{s+1-l} - \frac{1}{s+2-l} \Big) \Big(G(\mathbf{x}_1^{(l)}) - G(\mathbf{x}^*) \Big) \\
&= v_s \Big(G(\mathbf{x}_1^{(s+1)}) - G(\mathbf{x}^*) \Big) + \sum_{k=2}^{s} \frac{1}{s+2-k} \Big(G(\mathbf{x}_1^{(k)}) - G(\mathbf{x}^*) \Big) \\
&\quad - s(v_1 - v_0) \Big(G(\mathbf{x}_1^{(1)}) - G(\mathbf{x}^*) \Big) - \sum_{l=2}^{s} \frac{1}{s+2-l} \Big(G(\mathbf{x}_1^{(l)}) - G(\mathbf{x}^*) \Big) \\
&= v_s \Big(G(\mathbf{x}_1^{(s+1)}) - G(\mathbf{x}^*) \Big) - s(v_1 - v_0) \Big(G(\mathbf{x}_1^{(1)}) - G(\mathbf{x}^*) \Big)
\end{aligned}$$

Note that $v_1 = v_0$ and $v_s = 1$ by definition, and so taking expectation of both sides,

$$\sum_{k=1}^{s} v_k \Big(\mathbb{E} \left[G(\mathbf{x}_1^{(k+1)}) \right] - \mathbb{E} \left[G(\mathbf{z}^{(k)}) \right] \Big) \geq \mathbb{E} \left[G(\mathbf{x}_1^{(s+1)}) \right] - G(\mathbf{x}^*) \tag{C.52}$$

We now bound the term involving $\mathbb{E}\left[B_F(\mathbf{z}^{(k)}, \mathbf{x}^*) \right]$ in the R.H.S. of Eq. C.50. By

the convexity of $B_F(\cdot, \mathbf{x}^*)$ fixing the second argument (due to F being convex), and Eq. C.46,

$$B_F(\mathbf{z}^{(k)}, \mathbf{x}^*) \leq \frac{v_0}{v_k} B_F(\mathbf{x}^*, \mathbf{x}^*) + \sum_{l=1}^{k} \frac{v_l - v_{l-1}}{v_k} B_F(\mathbf{x}_1^{(l)}, \mathbf{x}^*) = \sum_{l=1}^{k} \frac{v_l - v_{l-1}}{v_k} B_F(\mathbf{x}_1^{(l)}, \mathbf{x}^*)$$

which implies

$$\begin{aligned}
&10\eta^2 n^2 L \sum_{k=1}^{s} v_k \widehat{L}^{(k)} \mathbb{E}\left[B_F(\mathbf{z}^{(k)}, \mathbf{x}^*)\right] \\
&\leq 10\eta^2 n^2 L \max_{k\in[s]} \widehat{L}^{(k)} \sum_{k=1}^{s} \sum_{l=1}^{k} (v_l - v_{l-1}) \mathbb{E}\left[B_F(\mathbf{x}_1^{(l)}, \mathbf{x}^*)\right] \\
&= 10\eta^2 n^2 L \max_{k\in[s]} \widehat{L}^{(k)} \sum_{l=1}^{s} (s+1-l)(v_l - v_{l-1}) \mathbb{E}\left[B_F(\mathbf{x}_1^{(l)}, \mathbf{x}^*)\right] \\
&= 10\eta^2 n^2 L \max_{k\in[s]} \widehat{L}^{(k)} \sum_{l=2}^{s} (s+1-l)\left(\frac{1}{s+1-l} - \frac{1}{s+2-l}\right) \mathbb{E}\left[B_F(\mathbf{x}_1^{(l)}, \mathbf{x}^*)\right] \\
&= 10\eta^2 n^2 L \max_{k\in[s]} \widehat{L}^{(k)} \sum_{l=2}^{s} \frac{1}{s+2-l} \mathbb{E}\left[B_F(\mathbf{x}_1^{(l)}, \mathbf{x}^*)\right] \\
&= 10\eta^2 n^2 L \max_{k\in[s]} \widehat{L}^{(k)} \sum_{l=2}^{s} v_{l-1} \mathbb{E}\left[B_F(\mathbf{x}_1^{(l)}, \mathbf{x}^*)\right] \qquad \text{(C.53)}
\end{aligned}$$

Therefore, plugging Eq. C.51, Eq. C.52 and Eq. C.53 back to Eq. C.50, there is, for any fixed epoch $s \in [K]$

$$\begin{aligned}
&\mathbb{E}\left[G(\mathbf{x}_1^{(s+1)})\right] - G(\mathbf{x}^*) \\
&\leq \frac{1}{2\eta n s} \|\mathbf{x}^* - \mathbf{x}_1^{(1)}\|^2 + 10\eta^2 n^2 L \max_{k\in[s]} \widehat{L}^{(k)} \sum_{k=2}^{s} \frac{1}{s+2-k} \mathbb{E}\left[B_F(\mathbf{x}_1^{(k)}, \mathbf{x}^*)\right] \\
&\quad + 5\eta^2 n^2 \sigma_{any}^2 \sum_{k=1}^{s} \frac{\widehat{L}^{(k)}}{s+1-k} + \frac{1}{2n^2\beta} \sum_{k=1}^{s} \frac{(C_n^{(k)})^2}{s+1-k} \\
&\quad + 5\eta^2 \sum_{k=1}^{s} \frac{\widehat{L}^{(k)*} \frac{1}{n} \sum_{i=n_d^{(k)}+1}^{n-1} (C_i^{(k)})^2}{s+1-k} + 6\eta^2 n d \sum_{k=1}^{s} \frac{(\sigma^{(k)})^2 \widehat{L}^{(k)*}}{s+1-k}
\end{aligned}$$

□

Theorem C.1.14 (Convergence of Generalized Shuffled Gradient Framework (Re-statement of Theorem 4.4.7)). *Under Assumptions 4.4.1, 4.4.4, 4.4.6, C.1.1 and 4.4.5, for $\beta > 0$, if $\mu_\psi \geq L_H^{(k)} + \beta$, $\forall k \in [K]$, and $\eta \leq \frac{1}{2n\sqrt{10\bar{L}^* \max_{k\in[K]} \widehat{L}^{(k)*}(1+\log K)}}$, where $\bar{L}^* = \max\{L, \max_{k\in[K]} \widehat{L}^{(k)}\}$, Algorithm 3 guarantees*

$$\mathbb{E}\left[G(\mathbf{x}_1^{(K+1)})\right] - G(\mathbf{x}^*) \leq \underbrace{\frac{\|\mathbf{x}_1^{(1)} - \mathbf{x}^*\|^2}{\eta n K}}_{\textit{Initialization}} + \underbrace{10\eta^2 n^2 \sigma_{any}^2 (1+\log K) \max_{k\in[K]} \widehat{L}^{(k)}}_{\textit{Optimization Uncertainty}} + 2M \tag{C.54}$$

where

$$M = \max_{s\in[K]} \Bigg(\underbrace{\frac{1}{2n^2\beta} \sum_{k=1}^{s} \frac{(C_n^{(k)})^2}{s+1-k}}_{\textit{Non-vanishing Dissimilarity}} + \underbrace{5\eta^2 \sum_{k=1}^{s} \frac{\widehat{L}^{(k)*} \frac{1}{n} \sum_{i=n_d^{(k)}+1}^{n-1} (C_i^{(k)})^2}{s+1-k}}_{\textit{Vanishing Dissimilarity}} + \underbrace{6\eta^2 nd \sum_{k=1}^{s} \frac{(\sigma^{(k)})^2 \widehat{L}^{(k)*}}{s+1-k}}_{\textit{Injected Noise}} \Bigg)$$

and the expectation is taken w.r.t. the injected noise $\{\rho_i^{(k)}\}$ and the order of samples $\pi^{(k)}$, $\forall i \in [n], k \in [K]$.

Proof of Theorem C.1.14. Taking the learning rate $\eta \leq \frac{1}{2n\sqrt{20\bar{L}^* \max_{k\in[K]} \widehat{L}^{(k)*}(1+\log K)}}$, where $\bar{L}^* = \max\{L, \max_{k\in[K]} \widehat{L}^{(k)}\}$, i.e., the max average smoothness parameters, satisfies the condition of Lemma C.1.13, and so for any number of epochs $s \in [K]$,

$$\begin{aligned}
\mathbb{E}\left[G(\mathbf{x}_1^{(s+1)})\right] - G(\mathbf{x}^*) &\leq \frac{1}{2\eta n s} \|\mathbf{x}^* - \mathbf{x}_1^{(1)}\|^2 \\
&+ 10\eta^2 n^2 L \max_{k\in[s]} \widehat{L}^{(k)} \sum_{k=2}^{s} \frac{1}{s+2-k} \mathbb{E}\left[B_F(\mathbf{x}_1^{(k)}, \mathbf{x}^*)\right] \\
&+ 5\eta^2 n^2 \sigma_{any}^2 \sum_{k=1}^{s} \frac{\widehat{L}^{(k)}}{s+1-k} + \frac{1}{2n^2\beta} \sum_{k=1}^{s} \frac{(C_n^{(k)})^2}{s+1-k} \\
&+ 5\eta^2 \sum_{k=1}^{s} \frac{\widehat{L}^{(k)*} \frac{1}{n} \sum_{i=n_d^{(k)}+1}^{n-1} (C_i^{(k)})^2}{s+1-k} + 6\eta^2 nd \sum_{k=1}^{s} \frac{(\sigma^{(k)})^2 \widehat{L}^{(k)*}}{s+1-k}
\end{aligned}$$

Note that $\sum_{k=1}^{s} v_k = \sum_{k=1}^{s} \frac{1}{s+1-k} = \sum_{k=1}^{s} \frac{1}{k} \leq 1 + \log s$, and so

$$5\eta^2 n^2 \sigma_{any}^2 \sum_{k=1}^{s} \frac{\widehat{L}^{(k)}}{s+1-k} \leq \eta^2 n^2 \sigma_{any}^2 (1+\log s) \max_{k\in[s]} \widehat{L}^{(k)}$$

Hence,

$$\begin{aligned}
&\mathbb{E}\left[G(\mathbf{x}_1^{(s+1)})\right] - G(\mathbf{x}^*) \\
&\leq \frac{1}{2\eta n s}\|\mathbf{x}^* - \mathbf{x}_1^{(1)}\|^2 + 10\eta^2 n^2 L \max_{k\in[s]} \widehat{L}^{(k)} \sum_{k=2}^{s} \frac{1}{s+2-k} \mathbb{E}\left[B_F(\mathbf{x}_1^{(k)}, \mathbf{x}^*)\right] \\
&\quad + 5\eta^2 n^2 \sigma_{any}^2 (1+\log s) \max_{k\in[s]} \widehat{L}^{(k)} + \frac{1}{2n^2\beta} \sum_{k=1}^{s} \frac{(C_n^{(k)})^2}{s+1-k} \\
&\quad + 5\eta^2 \sum_{k=1}^{s} \frac{\widehat{L}^{(k)*} \frac{1}{n} \sum_{i=n_d^{(k)}+1}^{n-1} (C_i^{(k)})^2}{s+1-k} + 6\eta^2 nd \sum_{k=1}^{s} \frac{(\sigma^{(k)})^2 \widehat{L}^{(k)*}}{s+1-k}
\end{aligned}$$

By the optimality condition, $\nabla G(\mathbf{x}^*) = \nabla F(\mathbf{x}^*) + \nabla \psi(\mathbf{x}^*) = \mathbf{0}$. Thus, for any $s \in [K]$,

$$\begin{aligned}
\mathbb{E}\left[G(\mathbf{x}_1^{(s+1)})\right] - G(\mathbf{x}^*) &\geq \mathbb{E}\left[G(\mathbf{x}_1^{(s+1)})\right] - \mathbb{E}\left[G(\mathbf{x}^*)\right] - \mathbb{E}\left[\langle \nabla F(\mathbf{x}^*) + \nabla \psi(\mathbf{x}^*), \mathbf{x}_1^{(s+1)} - \mathbf{x}^* \rangle\right] \\
&= \mathbb{E}\left[B_F(\mathbf{x}_1^{(s+1)}, \mathbf{x}^*)\right] + \mathbb{E}\left[B_\psi(\mathbf{x}_1^{(s+1)}, \mathbf{x}^*)\right] \geq \mathbb{E}\left[B_F(\mathbf{x}_1^{(s+1)}, \mathbf{x}^*)\right]
\end{aligned}$$

which implies that for any $s \in [K]$,

$$\begin{aligned}
\mathbb{E}\left[B_F(\mathbf{x}_1^{(s+1)}), \mathbf{x}^*)\right] &\leq \frac{1}{2\eta n s}\|\mathbf{x}^* - \mathbf{x}_1^{(1)}\|^2 + 10\eta^2 n^2 L \max_{k\in[s]} \widehat{L}^{(k)} \sum_{k=2}^{s} \frac{1}{s+2-k} \mathbb{E}\left[B_F(\mathbf{x}_1^{(k)}, \mathbf{x}^*)\right] \\
&\quad + 5\eta^2 n^2 \sigma_{any}^2 (1+\log s) \max_{k\in[s]} \widehat{L}^{(k)} + \frac{1}{2n^2\beta} \sum_{k=1}^{s} \frac{(C_n^{(k)})^2}{s+1-k} \\
&\quad + 5\eta^2 \sum_{k=1}^{s} \frac{\widehat{L}^{(k)*} \frac{1}{n} \sum_{i=n_d^{(k)}+1}^{n-1} (C_i^{(k)})^2}{s+1-k} + 6\eta^2 nd \sum_{k=1}^{s} \frac{(\sigma^{(k)})^2 \widehat{L}^{(k)*}}{s+1-k}
\end{aligned}$$

Note that $\max_{k\in[s]} \widehat{L}^{(k)} \leq \max_{k\in[K]} \widehat{L}^{(k)}$.

Now we apply Lemma C.1.6 with

- $d^{(s+1)} = \begin{cases} \mathbb{E}\left[B_F(\mathbf{x}_1^{(s+1)}, \mathbf{x}^*)\right] & s \in [K-1] \\ \mathbb{E}\left[F(\mathbf{x}_1^{(K+1)})\right] - \mathbb{E}\left[F(\mathbf{x}^*)\right] & s = K \end{cases}$
- $e^{(k)} = \frac{1}{2n^2\beta}(C_n^{(k)})^2 + 5\eta^2\widehat{L}^{(k)*}\frac{1}{n}\sum_{i=1}^{n-1}(C_i^{(k)})^2 + 6\eta^2 nd(\sigma^{(k)})^2\widehat{L}^{(k)*}, \forall k \in [K]$
- $a = \frac{1}{2\eta n}\|\mathbf{x}_1^{(1)} - \mathbf{x}^*\|^2$
- $b = 5\eta^2 n^2 \sigma_{any}^2(1+\log s)\max_{k\in[K]}\widehat{L}^{(k)}$
- $c = 10\eta^2 n^2 L \max_{k\in[K]}\widehat{L}^{(k)}$

to obtain

$$\begin{aligned} &\mathbb{E}\left[G(\mathbf{x}_1^{(K+1)})\right] - G(\mathbf{x}^*) \\ &\leq \Big(\frac{1}{2\eta nK}\|\mathbf{x}^* - \mathbf{x}_1^{(1)}\|^2 + 5\eta^2 n^2\sigma_{any}^2(1+\log K)\max_{k\in[K]}\widehat{L}^{(k)} + M\Big) \\ &\cdot \sum_{i=0}^{K-1}\Big(20\eta^2 n^2 L\max_{k\in[K]}\widehat{L}^{(k)}(1+\log K)\Big)^i \end{aligned}$$

where

$$M = \max_{s\in[K]}\Big(\frac{1}{2n^2\beta}\sum_{k=1}^{s}\frac{(C_n^{(k)})^2}{s+1-k} + 5\eta^2\sum_{k=1}^{s}\frac{\widehat{L}^{(k)*}\frac{1}{n}\sum_{i=n_d^{(k)}+1}^{n-1}(C_i^{(k)})^2}{s+1-k} + 6\eta^2 nd\sum_{k=1}^{s}\frac{(\sigma^{(k)})^2\widehat{L}^{(k)*}}{s+1-k}\Big)$$

By setting $\eta \leq \frac{1}{2n\sqrt{10\bar{L}^*\max_{k\in[K]}\widehat{L}^{(k)*}(1+\log K)}}$, where $\bar{L}^* = \max\{L, \max_{k\in[K]}\widehat{L}^{(k)}\}$, there is

$$\begin{aligned} \sum_{i=0}^{K-1}\Big(20\eta^2 n^2 L\max_{k\in[K]}\widehat{L}^{(k)}(1+\log K)\Big)^i &\leq \sum_{i=0}^{K-1}\Big(\frac{n^2 L\max_{k\in[K]}\widehat{L}^{(k)}(1+\log K)}{2n^2\bar{L}^*\max_{k\in[K]}\widehat{L}^{(k)*}(1+\log K)}\Big)^i \\ &\leq \sum_{i=0}^{\infty}\frac{1}{2^i} = 2 \end{aligned}$$

Therefore,

$$\mathbb{E}\left[G(\mathbf{x}_1^{(K+1)})\right] - G(\mathbf{x}^*) \leq \frac{\|\mathbf{x}_1^{(1)} - \mathbf{x}^*\|^2}{\eta nK} + 10\eta^2 n^2\sigma_{any}^2(1+\log K)\max_{k\in[K]}\widehat{L}^{(k)} + 2M$$

□

C.2 Private Shuffled Gradient Methods

C.2.1 Privacy Analysis

Remark on the Privacy Loss of Random Reshuffling (RR). For Incremental Gradient Methods (IG) and Shuffle Once (SO), it is not hard to see that the worst-case privacy loss bound presented above is tight. One might argue that, in the case of Random Reshuffling (RR), where the permutation $\pi_i^{(k)}$ is re-generated at the beginning of each epoch, leading to a reshuffling of the sample order in $\mathcal{D}$, this additional randomness could amplify privacy, thereby reducing the privacy loss. However, we argue that this potential improvement is limited to a constant level. Deriving a significantly smaller privacy loss bound in the RR setting – such as one that scales proportionally to $1/n$ – is unlikely without additional assumptions.

We use the following lemma to derive a tighter bound on the privacy loss of RR per epoch, taking into account the randomness introduced by shuffling:

Lemma C.2.1 (Joint convexity of scaled exponentiation of Rényi divergence, Lemma 4.1 of [143])**.** *Let $\mu_1, \ldots, \mu_m$ and $\nu_1, \ldots, \nu_m$ be distributions over $\mathbb{R}^d$. Then, for any RDP order $\alpha \geq 1$, and any coefficients $p_1, \ldots, p_m \geq 0$ that satisfy $p_1 + \cdots + p_m = 1$, the following inequality holds*

$$e^{(\alpha-1)D_\alpha(\sum_{j=1}^m p_j\mu_j \,\|\, \sum_{j=1}^m p_j\nu_j)} \leq \sum_{j=1}^m p_j \cdot e^{(\alpha-1)D_\alpha(\mu_j \,\|\, \nu_j)}$$

From the previous proof and the PABI bound (Theorem 4.3.9), we observe that the privacy loss for a single epoch is primarily determined by the index j such that $\pi_j^{(k)} = t$, where t is the index of the sample at which the two neighboring datasets $\mathcal{D}$ and $\mathcal{D}'$ differ ($\mathbf{d}_t \in \mathcal{D}$ and $\mathbf{d}'_t \in \mathcal{D}'$). Since shuffling ensures that j can take any value in $\{1, 2, \ldots, n\}$ with equal probability, j is a random variable uniformly distributed over $[n]$.

We apply Lemma C.2.1 by instantiating the distributions μ_i as the CNI's on $\mathcal{D}$ with $j = i$ for value $i \in [n]$ and similarly, the distributions ν_i as the CNI's on $\mathcal{D}'$ with $j = i$ for value $i \in [n]$. It easy to see that $p_i = \frac{1}{n}$. Hence, privacy loss of RR in epoch

k, $\epsilon^{(k)}_{\text{per-epoch}}$, is given by

$$\epsilon^{(k)}_{\text{per-epoch}} = \frac{1}{\alpha-1}\log e^{(\alpha-1)\cdot D_\alpha(\mathbf{x}^{(k)}_{n+1} \,\|\, \mathbf{x}^{(k)}_{n+1})}$$

$$\leq \frac{1}{\alpha-1}\log\left(\frac{1}{n}\sum_{j=1}^{n} e^{(\alpha-1)\cdot\frac{2\alpha G^2}{\sigma^2(n-j+1)}}\right) = \frac{1}{\alpha-1}\log\left(\underbrace{\frac{1}{n}\sum_{j=1}^{n} e^{(\alpha-1)\cdot\frac{2\alpha G^2}{\sigma^2\cdot j}}}_{:=S_n}\right)$$

In shuffled gradient methods, the dataset size n is finite and usually small to allow for $K \geq 2$ epochs over the dataset to ensure convergence. Therefore, we cannot asymptotically approximate the bound by treating $n \to \infty$. When n is small, the term S_n in bound is dominated by $e^{(\alpha-1)\frac{2\alpha G^2}{\sigma^2}}$ and $\frac{1}{n}$, leading to the approximation $S_n \approx \frac{1}{n}e^{(\alpha-1)\frac{2\alpha G^2}{\sigma^2}}$. Consequently, the upper bound on $\epsilon^{(k)}_{\text{per-epoch}}$ becomes $\frac{1}{\alpha-1}\log S_n \approx \frac{2\alpha G^2}{\sigma^2} - \log n$. This indicates the privacy loss bound for random reshuffling (RR) is nearly identical to that of IG and SO. As a result, the shuffling operation provides only a marginal improvement in privacy loss in this case.

Similar privacy loss bounds occur in the PABI-based privacy analysis of (impractical) variants of SGD and one can of course apply strong assumptions on $D_\alpha(\mathbf{x}^{(k)}_{n+1} \,\|\, (\mathbf{x}^{(k)}_{n+1})')$ to reduce the above upper bound. See, for example, Lemma 25 of the seminal work on PABI [39].

C.3 Additional Experiment Results

C.3.1 Variants of *DP-ShuffleG*

In the main paper, we present results using Random Reshuffling (RR). Here, we show more results using the other two variants of shuffled gradient methods, Incremental Gradient (IG) and Shuffle Once (SO), on datasets `CreditCard` and `MNIST-69`.

Again, we replace "ShuffleG" in each algortihm's name with "IG" or "SO". This results in the following algorithms for comparison:

1. IG-based: *Interleaved-IG*, *Priv-Pub-IG*, *Pub-Priv-IG* and *DP-IG*
2. SO-based: *Interleaved-SO*, *Priv-Pub-SO*, *Pub-Priv-SO* and *DP-SO*

We also include the baseline *Public Only* which uses public samples ($\mathcal{P}$) only.

Here, we fix $p = 0.5$ and the privacy parameters are $\epsilon \in \{5, 10\}$ and $\delta = 10^{-6}$.

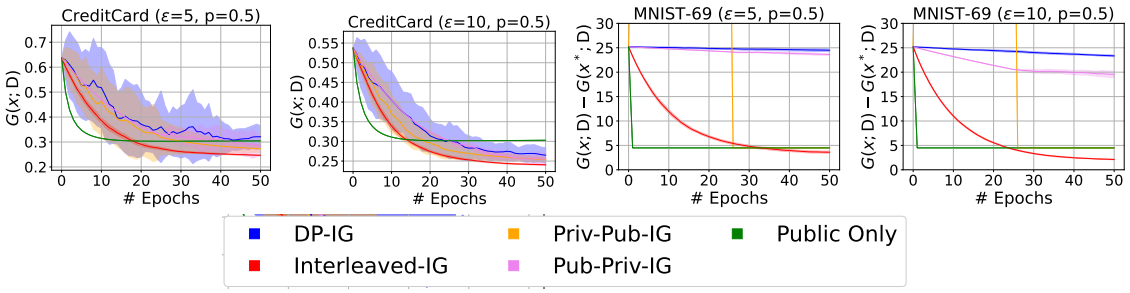

Figure C.1: Results of comparing IG-based algorithms on two datasets.

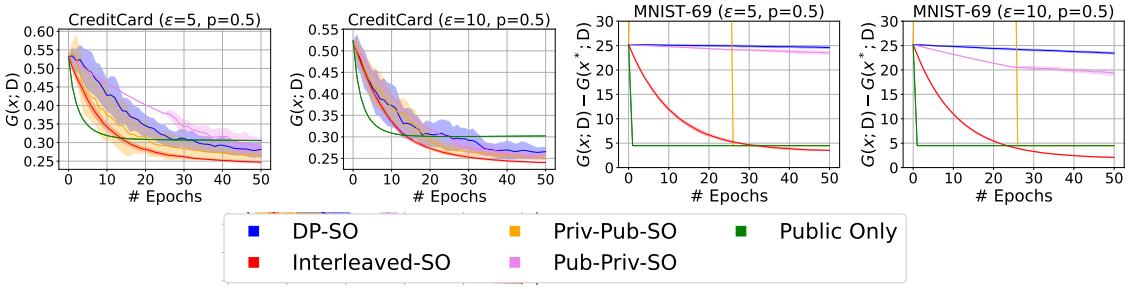

Figure C.2: Results of comparing SO-based algorithms on two datasets.

C.3.2 Varying Fraction p of Private Samples

In this setting, we vary the fraction of private samples p used in algorithms that leverage public data. Here, present results with $p \in \{0.25, 0.75\}$ on datasets `CreditCard` and `MNIST-69`.

We use RR in each algorithm. The privacy parameters are $\epsilon \in \{5, 10\}$ and $\delta = 10^{-6}$.

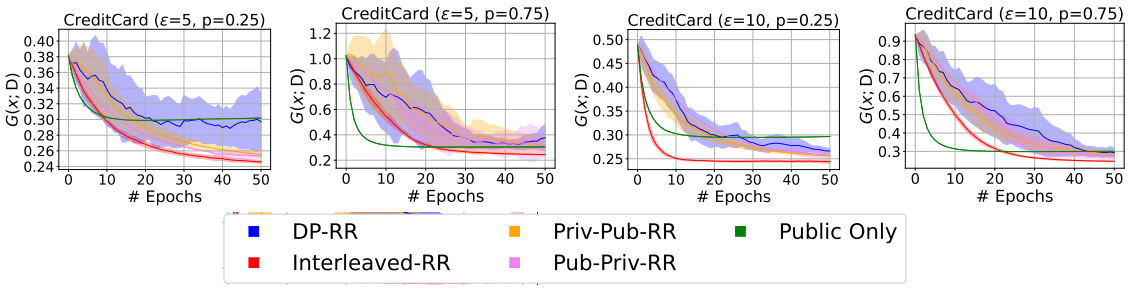

Figure C.3: Results of using different fractions of private samples for $p \in \{0.25, 0.75\}$ on dataset `CreditCard`.

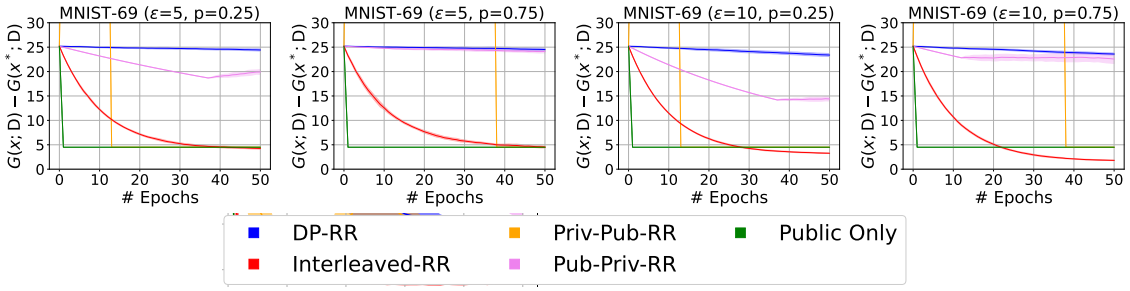

Figure C.4: Results of using different fractions of private samples for $p \in \{0.25, 0.75\}$ on dataset `MNIST-69`.

Bibliography

[1] Martin Abadi, Andy Chu, Ian Goodfellow, H. Brendan McMahan, Ilya Mironov, Kunal Talwar, and Li Zhang. Deep learning with differential privacy. In *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*, CCS'16. ACM, October 2016. doi: 10.1145/2976749.2978318. URL http://dx.doi.org/10.1145/2976749.2978318.

[2] Martin Abadi, Andy Chu, Ian Goodfellow, H Brendan McMahan, Ilya Mironov, Kunal Talwar, and Li Zhang. Deep learning with differential privacy. In *Proceedings of the 2016 ACM SIGSAC conference on computer and communications security*, pages 308–318, 2016.

[3] Nir Ailon and Bernard Chazelle. Approximate nearest neighbors and the fast johnson-lindenstrauss transform. In *Proceedings of the Thirty-Eighth Annual ACM Symposium on Theory of Computing*, STOC '06, page 557–563, New York, NY, USA, 2006. Association for Computing Machinery. ISBN 1595931341. doi: 10.1145/1132516.1132597. URL https://doi.org/10.1145/1132516.1132597.

[4] 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.

[5] Dan Alistarh, Torsten Hoefler, Mikael Johansson, Nikola Konstantinov, Sarit Khirirat, and Cédric Renggli. The convergence of sparsified gradient methods. *Advances in Neural Information Processing Systems*, 31, 2018.

[6] Jason Altschuler and Kunal Talwar. Privacy of noisy stochastic gradient descent: More iterations without more privacy loss. In Alice H. Oh, Alekh Agarwal, Danielle Belgrave, and Kyunghyun Cho, editors, *Advances in Neural Information Processing Systems*, 2022. URL https://openreview.net/forum?id=pDUYkwrx__w.

[7] Peter Arbenz, Walter Gander, and Gene H. Golub. Restricted rank modification of the symmetric eigenvalue problem: Theoretical considerations. *Linear Algebra and its Applications*, 104:75–95, 1988. ISSN 0024-3795. doi: https:

//doi.org/10.1016/0024-3795(88)90309-6. URL https://www.sciencedirect.com/science/article/pii/0024379588903096.

[8] Li Bai, Haibo Hu, Qingqing Ye, Haoyang Li, Leixia Wang, and Jianliang Xu. Membership inference attacks and defenses in federated learning: A survey. *ACM Comput. Surv.*, 57(4), December 2024. ISSN 0360-0300. doi: 10.1145/3704633. URL https://doi.org/10.1145/3704633.

[9] Oleg Balabanov, Matthias Beaupère, Laura Grigori, and Victor Lederer. Block subsampled randomized Hadamard transform for low-rank approximation on distributed architectures. working paper or preprint, October 2022. URL https://inria.hal.science/hal-03828607.

[10] Maria-Florina F Balcan, Steven Ehrlich, and Yingyu Liang. Distributed k-means and k-median clustering on general topologies. *Advances in neural information processing systems*, 26, 2013.

[11] Leighton Pate Barnes, Huseyin A Inan, Berivan Isik, and Ayfer Özgür. rtop-k: A statistical estimation approach to distributed sgd. *IEEE Journal on Selected Areas in Information Theory*, 1(3):897–907, 2020.

[12] Raef Bassily, Adam Smith, and Abhradeep Thakurta. Differentially private empirical risk minimization: Efficient algorithms and tight error bounds, 2014.

[13] Raef Bassily, Adam Smith, and Abhradeep Thakurta. Private empirical risk minimization: Efficient algorithms and tight error bounds. In *Proceedings of the 2014 IEEE 55th Annual Symposium on Foundations of Computer Science*, FOCS '14, page 464–473, USA, 2014. IEEE Computer Society. ISBN 9781479965175. doi: 10.1109/FOCS.2014.56. URL https://doi.org/10.1109/FOCS.2014.56.

[14] Raef Bassily, Om Thakkar, and Abhradeep Thakurta. Model-agnostic private learning via stability. *arXiv preprint arXiv:1803.05101*, 2018.

[15] Raef Bassily, Albert Cheu, Shay Moran, Aleksandar Nikolov, Jonathan Ullman, and Steven Wu. Private query release assisted by public data. 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 695–703. PMLR, 13–18 Jul 2020. URL https://proceedings.mlr.press/v119/bassily20a.html.

[16] Raef Bassily, Mehryar Mohri, and Ananda Theertha Suresh. Principled approaches for private adaptation from a public source. In Francisco Ruiz, Jennifer Dy, and Jan-Willem van de Meent, editors, *Proceedings of The 26th International Conference on Artificial Intelligence and Statistics*, volume 206 of *Proceedings of Machine Learning Research*, pages 8405–8432. PMLR, 25–27 Apr 2023. URL https://proceedings.mlr.press/v206/bassily23a.html.

[17] Debraj Basu, Deepesh Data, Can Karakus, and Suhas Diggavi. Qsparse-local-

sgd: Distributed sgd with quantization, sparsification and local computations. *Advances in Neural Information Processing Systems*, 32, 2019.

[18] 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.

[19] Alex Bie, Gautam Kamath, and Vikrant Singhal. Private estimation with public data. In Alice H. Oh, Alekh Agarwal, Danielle Belgrave, and Kyunghyun Cho, editors, *Advances in Neural Information Processing Systems*, 2022. URL https://openreview.net/forum?id=YpyGV_i8Z_J.

[20] Adam Block, Mark Bun, Rathin Desai, Abhishek Shetty, and Steven Wu. Oracle-efficient differentially private learning with public data. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024. URL https://openreview.net/forum?id=BAjjINf0Oh.

[21] Lucas Bourtoule, Varun Chandrasekaran, Christopher A Choquette-Choo, Hengrui Jia, Adelin Travers, Baiwu Zhang, David Lie, and Nicolas Papernot. Machine unlearning. In *2021 IEEE Symposium on Security and Privacy (SP)*, pages 141–159. IEEE, 2021.

[22] Christos Boutsidis and Alex Gittens. Improved matrix algorithms via the subsampled randomized hadamard transform, 2013.

[23] 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.

[24] Zhiqi Bu, Yu-Xiang Wang, Sheng Zha, and George Karypis. Differentially private bias-term only fine-tuning of foundation models, 2023. URL https://openreview.net/forum?id=zoTUH3Fjup.

[25] Wei-Ning Chen, Peter Kairouz, and Ayfer Ozgur. Breaking the communication-privacy-accuracy trilemma. *Advances in Neural Information Processing Systems*, 33:3312–3324, 2020.

[26] Lynn Chua, Badih Ghazi, Pritish Kamath, Ravi Kumar, Pasin Manurangsi, Amer Sinha, and Chiyuan Zhang. How private are DP-SGD implementations? 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 8904–8918. PMLR, 21–27 Jul 2024. URL https://proceedings.mlr.press/v235/chua24a.html.

[27] Lynn Chua, Badih Ghazi, Pritish Kamath, Ravi Kumar, Pasin Manurangsi, Amer Sinha, and Chiyuan Zhang. Scalable DP-SGD: Shuffling vs. poisson

subsampling. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024. URL https://openreview.net/forum?id=6gMnj9oc6d.

[28] Graham Cormode, Somesh Jha, Tejas Kulkarni, Ninghui Li, Divesh Srivastava, and Tianhao Wang. Privacy at scale: Local differential privacy in practice. In *Proceedings of the 2018 International Conference on Management of Data*, pages 1655–1658, 2018.

[29] Peter Davies, Vijaykrishna Gurunathan, Niusha Moshrefi, Saleh Ashkboos, and Dan Alistarh. New bounds for distributed mean estimation and variance reduction, 2021.

[30] Jinshuo Dong, David Durfee, and Ryan Rogers. Optimal differential privacy composition for exponential mechanisms. In *International Conference on Machine Learning*, pages 2597–2606. PMLR, 2020.

[31] Iain Dove. Applying differential privacy protection to ons mortality data, pilot study. Technical report, Office for National Statistics, 2021. URL https://www.ons.gov.uk/peoplepopulationandcommunity/birthsdeathsandmarriages/deaths/methodologies/applyingdifferentialprivacyprotectiontoonsmortalitydatapilotstudy. Technical report.

[32] David Durfee and Ryan M Rogers. Practical differentially private top-k selection with pay-what-you-get composition. *Advances in Neural Information Processing Systems*, 32, 2019.

[33] Cynthia Dwork and Vitaly Feldman. Privacy-preserving prediction. In *Conference On Learning Theory*, pages 1693–1702. PMLR, 2018.

[34] Cynthia Dwork and Jing Lei. Differential privacy and robust statistics. In *Proceedings of the forty-first annual ACM symposium on Theory of computing*, pages 371–380, 2009.

[35] Cynthia Dwork, Frank McSherry, Kobbi Nissim, and Adam Smith. Calibrating noise to sensitivity in private data analysis. In *Theory of cryptography conference*, pages 265–284. Springer, 2006.

[36] Cynthia Dwork, Aaron Roth, et al. The algorithmic foundations of differential privacy. *Found. Trends Theor. Comput. Sci.*, 9(3-4):211–407, 2014.

[37] Úlfar Erlingsson, Vasyl Pihur, and Aleksandra Korolova. Rappor: Randomized aggregatable privacy-preserving ordinal response. In *Proceedings of the 2014 ACM SIGSAC Conference on Computer and Communications Security*, CCS '14, page 1054–1067, New York, NY, USA, 2014. Association for Computing Machinery. ISBN 9781450329576. doi: 10.1145/2660267.2660348. URL https://doi.org/10.1145/2660267.2660348.

[38] Úlfar Erlingsson, Vitaly Feldman, Ilya Mironov, Ananth Raghunathan, Kunal Talwar, and Abhradeep Thakurta. Amplification by shuffling: From local to central differential privacy via anonymity. In *Proceedings of the Thirtieth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 2468–2479. SIAM, 2019.

[39] Vitaly Feldman, Ilya Mironov, Kunal Talwar, and Abhradeep Thakurta. Privacy amplification by iteration. In *2018 IEEE 59th Annual Symposium on Foundations of Computer Science (FOCS)*. IEEE, October 2018. doi: 10.1109/focs.2018.00056. URL http://dx.doi.org/10.1109/FOCS.2018.00056.

[40] Vitaly Feldman, Ilya Mironov, Kunal Talwar, and Abhradeep Thakurta. Privacy amplification by iteration. In *2018 IEEE 59th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 521–532. IEEE, 2018.

[41] Vitaly Feldman, Tomer Koren, and Kunal Talwar. Private stochastic convex optimization: optimal rates in linear time. In *Proceedings of the 52nd Annual ACM SIGACT Symposium on Theory of Computing*, STOC 2020, page 439–449, New York, NY, USA, 2020. Association for Computing Machinery. ISBN 9781450369794. doi: 10.1145/3357713.3384335. URL https://doi.org/10.1145/3357713.3384335.

[42] Tiantian Feng, Raghuveer Peri, and Shrikanth Narayanan. User-level differential privacy against attribute inference attack of speech emotion recognition on federated learning. In *Proceedings of Interspeech 2022*. ISCA, September 2022. doi: 10.21437/Interspeech.2022-10060. URL http://dx.doi.org/10.21437/Interspeech.2022-10060.

[43] R. Gallager. Low-density parity-check codes. *IRE Transactions on Information Theory*, 8(1):21–28, 1962. doi: 10.1109/TIT.1962.1057683.

[44] Venkata Gandikota, Daniel Kane, Raj Kumar Maity, and Arya Mazumdar. vqsgd: Vector quantized stochastic gradient descent. In *International Conference on Artificial Intelligence and Statistics*, pages 2197–2205. PMLR, 2021.

[45] Ankit Garg, Tengyu Ma, and Huy Nguyen. On communication cost of distributed statistical estimation and dimensionality. *Advances in Neural Information Processing Systems*, 27, 2014.

[46] Quan Geng and Pramod Viswanath. The optimal noise-adding mechanism in differential privacy. *IEEE Transactions on Information Theory*, 62(2):925–951, 2015.

[47] Gene H. Golub. Some modified matrix eigenvalue problems. *SIAM Review*, 15(2):318–334, 1973. ISSN 00361445. URL http://www.jstor.org/stable/2028604.

[48] Gene H Golub and Charles F Van Loan. *Matrix computations*. JHU press, 2013.

[49] Ming Gu and Stanley C. Eisenstat. A stable and efficient algorithm for the rank-one modification of the symmetric eigenproblem. *SIAM Journal on Matrix Analysis and Applications*, 15(4):1266–1276, 1994. doi: 10.1137/S089547989223924X. URL https://doi.org/10.1137/S089547989223924X.

[50] John A Gubner. Distributed estimation and quantization. *IEEE Transactions on Information Theory*, 39(4):1456–1459, 1993.

[51] Farzin Haddadpour, Belhal Karimi, Ping Li, and Xiaoyun Li. Fedsketch: Communication-efficient and private federated learning via sketching. *arXiv preprint arXiv:2008.04975*, 2020.

[52] Mostafa Haghir Chehreghani. Subsampled randomized hadamard transform for regression of dynamic graphs. In *Proceedings of the 29th ACM International Conference on Information & Knowledge Management*, CIKM '20, page 2045–2048, New York, NY, USA, 2020. Association for Computing Machinery. ISBN 9781450368599. doi: 10.1145/3340531.3412158. URL https://doi.org/10.1145/3340531.3412158.

[53] Andrew Hard, Kanishka Rao, Rajiv Mathews, Swaroop Ramaswamy, Françoise Beaufays, Sean Augenstein, Hubert Eichner, Chloé Kiddon, and Daniel Ramage. Federated learning for mobile keyboard prediction, 2019. URL https://arxiv.org/abs/1811.03604.

[54] Naoise Holohan, Douglas J. Leith, and Oliver Mason. Optimal differentially private mechanisms for randomised response. *IEEE Transactions on Information Forensics and Security*, 12(11):2726–2735, November 2017. ISSN 1556-6021. doi: 10.1109/tifs.2017.2718487. URL http://dx.doi.org/10.1109/TIFS.2017.2718487.

[55] Samuel Horváth and Peter Richtarik. A better alternative to error feedback for communication-efficient distributed learning. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=vYVI1CHPaQg.

[56] Nikita Ivkin, Daniel Rothchild, Enayat Ullah, Ion Stoica, Raman Arora, et al. Communication-efficient distributed sgd with sketching. *Advances in Neural Information Processing Systems*, 32, 2019.

[57] Divyansh Jhunjhunwala, Ankur Mallick, Advait Harshal Gadhikar, Swanand Kadhe, and Gauri Joshi. Leveraging spatial and temporal correlations in sparsified mean estimation. In A. Beygelzimer, Y. Dauphin, P. Liang, and J. Wortman Vaughan, editors, *Advances in Neural Information Processing Systems*, 2021. URL https://openreview.net/forum?id=BKeJmkspvc.

[58] Junjie Jia and Wanyong Qiu. Research on an ensemble classification algorithm based on differential privacy. *IEEE Access*, 8:93499–93513, 2020. doi: 10.1109/

ACCESS.2020.2995058.

[59] Madhura Joshi, Ankit Pal, and Malaikannan Sankarasubbu. Federated learning for healthcare domain - pipeline, applications and challenges. *ACM Trans. Comput. Healthcare*, 3(4), November 2022. doi: 10.1145/3533708. URL https://doi.org/10.1145/3533708.

[60] Peter Kairouz, Sewoong Oh, and Pramod Viswanath. The composition theorem for differential privacy. In *International conference on machine learning*, pages 1376–1385. PMLR, 2015.

[61] Peter Kairouz, Monica Ribero Diaz, Keith Rush, and Abhradeep Thakurta. (nearly) dimension independent private erm with adagrad rates
via publicly estimated subspaces. In Mikhail Belkin and Samory Kpotufe, editors, *Proceedings of Thirty Fourth Conference on Learning Theory*, volume 134 of *Proceedings of Machine Learning Research*, pages 2717–2746. PMLR, 15–19 Aug 2021. URL https://proceedings.mlr.press/v134/kairouz21a.html.

[62] Peter Kairouz, Brendan McMahan, Shuang Song, Om Thakkar, Abhradeep Thakurta, and Zheng Xu. Practical and private (deep) learning without sampling or shuffling, 2021. URL https://arxiv.org/abs/2103.00039.

[63] 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.

[64] Jared Kaplan, Sam McCandlish, Tom Henighan, Tom B. Brown, Benjamin Chess, Rewon Child, Scott Gray, Alec Radford, Jeffrey Wu, and Dario Amodei. Scaling laws for neural language models, 2020. URL https://arxiv.org/abs/2001.08361.

[65] 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.

[66] Jakub Konečnỳ and Peter Richtárik. Randomized distributed mean estimation: Accuracy vs. communication. *Frontiers in Applied Mathematics and Statistics*, 4:62, 2018.

[67] Jakub Konečnỳ, H Brendan McMahan, Daniel Ramage, and Peter Richtárik. Federated optimization: Distributed machine learning for on-device intelligence. *arXiv preprint arXiv:1610.02527*, 2016.

[68] Jakub Konečnỳ, H Brendan McMahan, Felix X Yu, Peter Richtárik, Ananda Theertha Suresh, and Dave Bacon. Federated learning: Strategies for improving communication efficiency. *arXiv preprint arXiv:1610.05492*, 2016.

[69] Jonathan Lacotte and Mert Pilanci. Optimal randomized first-order methods for least-squares problems, 2020.

[70] Jonathan Lacotte, Sifan Liu, Edgar Dobriban, and Mert Pilanci. Optimal iterative sketching with the subsampled randomized hadamard transform, 2020.

[71] Zijian Lei and Liang Lan. Improved subsampled randomized hadamard transform for linear svm. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 4519–4526, 2020.

[72] Ming Li, Pengcheng Xu, Junjie Hu, Zeyu Tang, and Guang Yang. From challenges and pitfalls to recommendations and opportunities: Implementing federated learning in healthcare, 2025. URL https://arxiv.org/abs/2409.09727.

[73] Kai Liang and Youlong Wu. Improved communication efficiency for distributed mean estimation with side information. In *2021 IEEE International Symposium on Information Theory (ISIT)*, pages 3185–3190. IEEE, 2021.

[74] Jingcheng Liu and Kunal Talwar. Private selection from private candidates. In *Proceedings of the 51st Annual ACM SIGACT Symposium on Theory of Computing*, STOC 2019, page 298–309, New York, NY, USA, 2019. Association for Computing Machinery. ISBN 9781450367059. doi: 10.1145/3313276.3316377. URL https://doi.org/10.1145/3313276.3316377.

[75] Zhongfeng Liu, Yun Li, and Wei Ji. Differential private ensemble feature selection. In *2018 International Joint Conference on Neural Networks (IJCNN)*, pages 1–6, 2018. doi: 10.1109/IJCNN.2018.8489308.

[76] Zijian Liu and Zhengyuan Zhou. On the last-iterate convergence of shuffling gradient methods. In *Proceedings of the 41st International Conference on Machine Learning*, ICML'24. JMLR.org, 2025.

[77] Yichao Lu, Paramveer Dhillon, Dean P Foster, and Lyle Ungar. Faster ridge regression via the subsampled randomized hadamard transform. In C.J. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K.Q. Weinberger, editors, *Advances in Neural Information Processing Systems*, volume 26. Curran Associates, Inc., 2013. URL https://proceedings.neurips.cc/paper_files/paper/2013/file/621bf66ddb7c962aa0d22ac97d69b793-Paper.pdf.

[78] Prathamesh Mayekar, Ananda Theertha Suresh, and Himanshu Tyagi. Wyner-ziv estimators: Efficient distributed mean estimation with side-information. In *International Conference on Artificial Intelligence and Statistics*, pages 3502–3510. PMLR, 2021.

[79] 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.

[80] Gregory T. Minton and Eric Price. Improved concentration bounds for count-sketch, 2013.

[81] Fatemehsadat Mireshghallah, Mohammadkazem Taram, Prakash Ramrakhyani, Ali Jalali, Dean Tullsen, and Hadi Esmaeilzadeh. Shredder: Learning noise distributions to protect inference privacy. In *Proceedings of the Twenty-Fifth International Conference on Architectural Support for Programming Languages and Operating Systems*, pages 3–18, 2020.

[82] Ilya Mironov. Rényi differential privacy. In *2017 IEEE 30th Computer Security Foundations Symposium (CSF)*. IEEE, August 2017. doi: 10.1109/csf.2017.11. URL http://dx.doi.org/10.1109/CSF.2017.11.

[83] Konstantin Mishchenko, Ahmed Khaled, and Peter Richtárik. Proximal and federated random reshuffling, 2021. URL https://arxiv.org/abs/2102.06704.

[84] Konstantin Mishchenko, Ahmed Khaled, and Peter Richtárik. Random reshuffling: Simple analysis with vast improvements, 2021. URL https://arxiv.org/abs/2006.05988.

[85] Moni Naor, Kobbi Nissim, Uri Stemmer, and Chao Yan. Private everlasting prediction. *arXiv preprint arXiv:2305.09579*, 2023.

[86] Mohammad Naseri, Jamie Hayes, and Emiliano De Cristofaro. Local and central differential privacy for robustness and privacy in federated learning. *arXiv preprint arXiv:2009.03561*, 2020.

[87] Mohammad Naseri, Jamie Hayes, and Emiliano De Cristofaro. Local and central differential privacy for robustness and privacy in federated learning, 2022. URL https://arxiv.org/abs/2009.03561.

[88] Kobbi Nissim, Sofya Raskhodnikova, and Adam Smith. Smooth sensitivity and sampling in private data analysis. In *Proceedings of the thirty-ninth annual ACM symposium on Theory of computing*, pages 75–84, 2007.

[89] Matthew Nokleby and Waheed U Bajwa. Stochastic optimization from distributed streaming data in rate-limited networks. *IEEE transactions on signal and information processing over networks*, 5(1):152–167, 2018.

[90] Ahmed El Ouadrhiri and Ahmed Abdelhadi. Differential privacy for deep and federated learning: A survey. *IEEE Access*, 10:22359–22380, 2022. doi: 10.1109/ACCESS.2022.3151670.

[91] Emre Ozfatura, Kerem Ozfatura, and Deniz Gündüz. Time-correlated sparsification for communication-efficient federated learning. In *2021 IEEE International Symposium on Information Theory (ISIT)*, pages 461–466. IEEE, 2021.

[92] Balaji Palanisamy, Chao Li, and Prashant Krishnamurthy. Group Differential

Privacy-Preserving Disclosure of Multi-level Association Graphs . In *2017 IEEE 37th International Conference on Distributed Computing Systems (ICDCS)*, pages 2587–2588, Los Alamitos, CA, USA, June 2017. IEEE Computer Society. doi: 10.1109/ICDCS.2017.223. URL https://doi.ieeecomputersociety.org/10.1109/ICDCS.2017.223.

[93] Nicolas Papernot and Thomas Steinke. Hyperparameter tuning with renyi differential privacy. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=-70L8lpp9DF.

[94] Nicolas Papernot, Martin Abadi, Ulfar Erlingsson, Ian Goodfellow, and Kunal Talwar. Semi-supervised knowledge transfer for deep learning from private training data. In *Proceedings of the International Conference on Learning Representations*, 2017. URL https://arxiv.org/abs/1610.05755.

[95] Nicolas Papernot, Shuang Song, Ilya Mironov, Ananth Raghunathan, Kunal Talwar, and Úlfar Erlingsson. Scalable private learning with pate. *arXiv preprint arXiv:1802.08908*, 2018.

[96] Natalia Ponomareva, Hussein Hazimeh, Alex Kurakin, Zheng Xu, Carson Denison, H. Brendan McMahan, Sergei Vassilvitskii, Steve Chien, and Abhradeep Guha Thakurta. How to dp-fy ml: A practical guide to machine learning with differential privacy. *Journal of Artificial Intelligence Research*, 77:1113–1201, July 2023. ISSN 1076-9757. doi: 10.1613/jair.1.14649. URL http://dx.doi.org/10.1613/jair.1.14649.

[97] Carey Radebaugh and Ulfar Erlingsson. Introducing tensorflow privacy: Learning with differential privacy for training data, 2019. URL blog.tensorflow.org.

[98] Mahfuzzur Rahman, Mahima Rabbi, Annajiat Alim Rasel, and Md Tanzim Reza. A design and implementation of bangla next word predictor based on personalized federated learning leveraging model agnostic meta learning and semantic analysis. In *2022 IEEE 13th Annual Information Technology, Electronics and Mobile Communication Conference (IEMCON)*, pages 0384–0390, 2022. doi: 10.1109/IEMCON56893.2022.9946584.

[99] Swaroop Ramaswamy, Om Thakkar, Rajiv Mathews, Galen Andrew, H. Brendan McMahan, and Françoise Beaufays. Training production language models without memorizing user data, 2020. URL https://arxiv.org/abs/2009.10031.

[100] Amirhossein Reisizadeh, Aryan Mokhtari, Hamed Hassani, Ali Jadbabaie, and Ramtin Pedarsani. Fedpaq: A communication-efficient federated learning method with periodic averaging and quantization. In *International Conference on Artificial Intelligence and Statistics*, pages 2021–2031. PMLR, 2020.

[101] Nicola Rieke, Jonny Hancox, Wenqi Li, Fausto Milletarì, Holger R. Roth, Shadi Albarqouni, Spyridon Bakas, Mathieu N. Galtier, Bennett A. Landman, Klaus Maier-Hein, Sébastien Ourselin, Micah Sheller, Ronald M. Summers, Andrew Trask, Daguang Xu, Maximilian Baust, and M. Jorge Cardoso. The future of digital health with federated learning. *npj Digital Medicine*, 3(1): 119, 2020. ISSN 2398-6352. doi: 10.1038/s41746-020-00323-1. URL https://doi.org/10.1038/s41746-020-00323-1.

[102] Daniel Rothchild, Ashwinee Panda, Enayat Ullah, Nikita Ivkin, Ion Stoica, Vladimir Braverman, Joseph Gonzalez, and Raman Arora. Fetchsgd: Communication-efficient federated learning with sketching. In *International Conference on Machine Learning*, pages 8253–8265. PMLR, 2020.

[103] Omer Sagi and Lior Rokach. Ensemble learning: A survey. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, 8(4):e1249, 2018.

[104] Felix Sattler, Simon Wiedemann, Klaus-Robert Müller, and Wojciech Samek. Robust and communication-efficient federated learning from non-iid data. *IEEE transactions on neural networks and learning systems*, 31(9):3400–3413, 2019.

[105] Ohad Shamir and Tong Zhang. Stochastic gradient descent for non-smooth optimization: Convergence results and optimal averaging schemes. In Sanjoy Dasgupta and David McAllester, editors, *Proceedings of the 30th International Conference on Machine Learning*, volume 28 of *Proceedings of Machine Learning Research*, pages 71–79, Atlanta, Georgia, USA, 17–19 Jun 2013. PMLR. URL https://proceedings.mlr.press/v28/shamir13.html.

[106] Shaohuai Shi, Xiaowen Chu, Ka Chun Cheung, and Simon See. Understanding top-k sparsification in distributed deep learning. *arXiv preprint arXiv:1911.08772*, 2019.

[107] Nir Shlezinger, Mingzhe Chen, Yonina C Eldar, H Vincent Poor, and Shuguang Cui. Uveqfed: Universal vector quantization for federated learning. *IEEE Transactions on Signal Processing*, 69:500–514, 2020.

[108] Reza Shokri, Marco Stronati, Congzheng Song, and Vitaly Shmatikov. Membership Inference Attacks Against Machine Learning Models . In *2017 IEEE Symposium on Security and Privacy (SP)*, pages 3–18, Los Alamitos, CA, USA, May 2017. IEEE Computer Society. doi: 10.1109/SP.2017.41. URL https://doi.ieeecomputersociety.org/10.1109/SP.2017.41.

[109] Amin Shokrollahi. Fountain codes. *Iee Proceedings-communications - IEE PROC-COMMUN*, 152, 01 2005.

[110] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

[111] Michal Staňo, Ladislav Hluchý, Martin Bobák, Peter Krammer, and Viet

Tran. Federated learning methods for analytics of big and sensitive distributed data and survey. In *2023 IEEE 17th International Symposium on Applied Computational Intelligence and Informatics (SACI)*, pages 000705–000710, 2023. doi: 10.1109/SACI58269.2023.10158622.

[112] Uri Stemmer. Private truly-everlasting robust-prediction. *arXiv preprint arXiv:2401.04311*, 2024.

[113] Sebastian U Stich, Jean-Baptiste Cordonnier, and Martin Jaggi. Sparsified sgd with memory. *Advances in Neural Information Processing Systems*, 31, 2018.

[114] Joel Stremmel and Arjun Singh. Pretraining federated text models for next word prediction, 2020. URL https://arxiv.org/abs/2005.04828.

[115] Ananda Theertha Suresh, X Yu Felix, Sanjiv Kumar, and H Brendan McMahan. Distributed mean estimation with limited communication. In *International conference on machine learning*, pages 3329–3337. PMLR, 2017.

[116] Ananda Theertha Suresh, Ziteng Sun, Jae Ro, and Felix Yu. Correlated quantization for distributed mean estimation and optimization. In *International Conference on Machine Learning*, pages 20856–20876. PMLR, 2022.

[117] Anshuman Suri, Pallika Kanani, Virendra J. Marathe, and Daniel W. Peterson. Subject membership inference attacks in federated learning, 2023. URL https://arxiv.org/abs/2206.03317.

[118] Qi Tan, Qi Li, Yi Zhao, Zhuotao Liu, Xiaobing Guo, and Ke Xu. Defending against data reconstruction attacks in federated learning: an information theory approach. In *Proceedings of the 33rd USENIX Conference on Security Symposium*, SEC '24, USA, 2024. USENIX Association. ISBN 978-1-939133-44-1.

[119] Sumanth Tatineni. Federated learning for privacy- preserving data analysis: Applications and challenges. *INTERNATIONAL JOURNAL OF COMPUTER ENGINEERING & TECHNOLOGY*, 9:270–277, 12 2018.

[120] Dan Teng, Xiaowei Zhang, Li Cheng, and Delin Chu. Least squares approximation via sparse subsampled randomized hadamard transform. *IEEE Transactions on Big Data*, 8(2):446–457, 2022. doi: 10.1109/TBDATA.2020.2972887.

[121] Abhradeep Guha Thakurta and Adam Smith. Differentially private feature selection via stability arguments, and the robustness of the lasso. In *Conference on Learning Theory*, pages 819–850. PMLR, 2013.

[122] Rohit Tripathy, Ilias Bilionis, and Marcial Gonzalez. Gaussian processes with built-in dimensionality reduction: Applications to high-dimensional uncertainty propagation. *Journal of Computational Physics*, 321:191–223, 2016.

[123] Joel A. Tropp. Improved analysis of the subsampled randomized hadamard

transform, 2011.

[124] Enayat Ullah, Michael Menart, Raef Bassily, Cristóbal A Guzmán, and Raman Arora. Public-data assisted private stochastic optimization: Power and limitations. In *The Thirty-eighth Annual Conference on Neural Information Processing Systems*, 2024. URL https://openreview.net/forum?id=j14wStqZni.

[125] Salil Vadhan. *The Complexity of Differential Privacy*, pages 347–450. Springer International Publishing, Cham, 2017. doi: 10.1007/978-3-319-57048-8_7. URL https://doi.org/10.1007/978-3-319-57048-8_7.

[126] Laurens van der Maaten and Awni Hannun. The trade-offs of private prediction, 2020.

[127] Shay Vargaftik, Ran Ben-Basat, Amit Portnoy, Gal Mendelson, Yaniv Ben-Itzhak, and Michael Mitzenmacher. Drive: One-bit distributed mean estimation. *Advances in Neural Information Processing Systems*, 34:362–377, 2021.

[128] Shay Vargaftik, Ran Ben Basat, Amit Portnoy, Gal Mendelson, Yaniv Ben-Itzhak, and Michael Mitzenmacher. Eden: Communication-efficient and robust distributed mean estimation for federated learning, 2022.

[129] Shay Vargaftik, Ran Ben Basat, Amit Portnoy, Gal Mendelson, Yaniv Ben Itzhak, and Michael Mitzenmacher. Eden: Communication-efficient and robust distributed mean estimation for federated learning. In *International Conference on Machine Learning*, pages 21984–22014. PMLR, 2022.

[130] Jianyu Wang, Zachary Charles, Zheng Xu, Gauri Joshi, H Brendan McMahan, Maruan Al-Shedivat, Galen Andrew, Salman Avestimehr, Katharine Daly, Deepesh Data, et al. A field guide to federated optimization. *arXiv preprint arXiv:2107.06917*, 2021.

[131] Jianyu Wang, Qinghua Liu, Hao Liang, Gauri Joshi, and H. Vincent Poor. A novel framework for the analysis and design of heterogeneous federated learning. *IEEE Transactions on Signal Processing*, 69:5234–5249, 2021. doi: 10.1109/TSP.2021.3106104.

[132] Jun Wang and Zhi-Hua Zhou. Differentially private learning with small public data. *Proceedings of the AAAI Conference on Artificial Intelligence*, 34(04): 6219–6226, Apr. 2020. doi: 10.1609/aaai.v34i04.6088. URL https://ojs.aaai.org/index.php/AAAI/article/view/6088.

[133] Shiqiang Wang, Tiffany Tuor, Theodoros Salonidis, Kin K. Leung, Christian Makaya, Ting He, and Kevin Chan. Adaptive federated learning in resource constrained edge computing systems. *IEEE Journal on Selected Areas in Communications*, 37(6):1205–1221, 2019. doi: 10.1109/JSAC.2019.2904348.

[134] Zhibo Wang, Mengkai Song, Zhifei Zhang, Yang Song, Qian Wang, and Hairong

Qi. Beyond inferring class representatives: User-level privacy leakage from federated learning. In *IEEE INFOCOM 2019 - IEEE Conference on Computer Communications*, page 2512–2520. IEEE Press, 2019. doi: 10.1109/INFOCOM.2019.8737416. URL https://doi.org/10.1109/INFOCOM.2019.8737416.

[135] Jianqiao Wangni, Jialei Wang, Ji Liu, and Tong Zhang. Gradient sparsification for communication-efficient distributed optimization. *Advances in Neural Information Processing Systems*, 31, 2018.

[136] Kang Wei, Jun Li, Ming Ding, Chuan Ma, Howard H. Yang, Farhad Farokhi, Shi Jin, Tony Q. S. Quek, and H. Vincent Poor. Federated learning with differential privacy: Algorithms and performance analysis. *IEEE Transactions on Information Forensics and Security*, 15:3454–3469, 2020. doi: 10.1109/TIFS.2020.2988575.

[137] Kilian Q Weinberger, Fei Sha, and Lawrence K Saul. Learning a kernel matrix for nonlinear dimensionality reduction. In *Proceedings of the twenty-first international conference on Machine learning*, page 106, 2004.

[138] Blake Woodworth, Konstantin Mishchenko, and Francis Bach. Two losses are better than one: faster optimization using a cheaper proxy. In *Proceedings of the 40th International Conference on Machine Learning*, ICML'23. JMLR.org, 2023.

[139] Ming Xiang and Lili Su. β-stochastic sign SGD: A byzantine resilient and differentially private gradient compressor for federated learning, 2023. URL https://openreview.net/forum?id=oVPqFCI1g7q.

[140] Tao Xiang, Yang Li, Xiaoguo Li, Shigang Zhong, and Shui Yu. Collaborative ensemble learning under differential privacy. *Web Intelligence*, 16:73–87, 03 2018. doi: 10.3233/WEB-180374.

[141] Han Xiao, Kashif Rasul, and Roland Vollgraf. Fashion-mnist: a novel image dataset for benchmarking machine learning algorithms. *arXiv preprint arXiv:1708.07747*, 2017.

[142] Jirui Yang, Peng Chen, Zhihui Lu, Qiang Duan, and Yubing Bao. Uifv: Data reconstruction attack in vertical federated learning, 2025. URL https://arxiv.org/abs/2406.12588.

[143] Jiayuan Ye and Reza Shokri. Differentially private learning needs hidden state (or much faster convergence). In Alice H. Oh, Alekh Agarwal, Danielle Belgrave, and Kyunghyun Cho, editors, *Advances in Neural Information Processing Systems*, 2022. URL https://openreview.net/forum?id=ipAz7H8pPnI.

[144] Ashkan Yousefpour, Igor Shilov, Alexandre Sablayrolles, Davide Testuggine, Karthik Prasad, Mani Malek, John Nguyen, Sayan Ghosh, Akash Bharadwaj, Jessica Zhao, Graham Cormode, and Ilya Mironov. Opacus: User-friendly

differential privacy library in pytorch, 2022. URL https://arxiv.org/abs/2109.12298.

[145] Da Yu, Saurabh Naik, Arturs Backurs, Sivakanth Gopi, Huseyin A Inan, Gautam Kamath, Janardhan Kulkarni, Yin Tat Lee, Andre Manoel, Lukas Wutschitz, Sergey Yekhanin, and Huishuai Zhang. Differentially private fine-tuning of language models. In *International Conference on Learning Representations*, 2022. URL https://openreview.net/forum?id=Q42f0dfjECO.

[146] Amirreza Zamani, Tobias J. Oechtering, and Mikael Skoglund. On the privacy-utility trade-off with and without direct access to the private data. *IEEE Transactions on Information Theory*, 70(3):2177–2200, 2024. doi: 10.1109/TIT.2023.3326070.

[147] Ying-Ying Zhang, Teng-Zhong Rong, and Man-Man Li. Expectation identity for the binomial distribution and its application in the calculations of high-order binomial moments. *Communications in Statistics - Theory and Methods*, 48(22):5467–5476, 2019. doi: 10.1080/03610926.2018.1435818. URL https://doi.org/10.1080/03610926.2018.1435818.

[148] Yuchen Zhang, John Duchi, Michael I Jordan, and Martin J Wainwright. Information-theoretic lower bounds for distributed statistical estimation with communication constraints. *Advances in Neural Information Processing Systems*, 26, 2013.

[149] Hao Zhong and Kaifeng Bu. Privacy-utility trade-off, 2022. URL https://arxiv.org/abs/2204.12057.

[150] Mingxun Zhou, Tianhao Wang, T-H. Hubert Chan, Giulia Fanti, and Elaine Shi. Locally differentially private sparse vector aggregation. In *2022 IEEE Symposium on Security and Privacy (SP)*, pages 422–439, 2022. doi: 10.1109/SP46214.2022.9833635.

[151] Yingxue Zhou, Steven Wu, and Arindam Banerjee. Bypassing the ambient dimension: Private {sgd} with gradient subspace identification. In *International Conference on Learning Representations*, 2021. URL https://openreview.net/forum?id=7dpmlkBuJFC.

[152] Gongxi Zhu, Donghao Li, Hanlin Gu, Yuan Yao, Lixin Fan, and Yuxing Han. Fedmia: An effective membership inference attack exploiting "all for one" principle in federated learning, 2025. URL https://arxiv.org/abs/2402.06289.

[153] Ligeng Zhu, Zhijian Liu, and Song Han. Deep leakage from gradients. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019. URL https://proceedings.neurips.cc/paper_

files/paper/2019/file/60a6c4002cc7b29142def8871531281a-Paper.pdf.

[154] Yuqing Zhu, Xuandong Zhao, Chuan Guo, and Yu-Xiang Wang. " private prediction strikes back!"private kernelized nearest neighbors with individual renyi filter. *arXiv preprint arXiv:2306.07381*, 2023.

[155] J. Ziegler, B. Pfitzner, H. Schulz, A. Saalbach, and B. Arnrich. Defending against reconstruction attacks through differentially private federated learning for classification of heterogeneous chest x-ray data. *Sensors*, 22(14):5195, 2022. doi: 10.3390/s22145195. URL https://doi.org/10.3390/s22145195.