Applying the Shuffle Model of Differential Privacy to Vector Aggregation

Mary Scott[1], Graham Cormode[1] and Carsten Maple[2]

[1]Department of Computer Science, University of Warwick, Coventry, CV4 7AL, UK
[2]WMG, University of Warwick, Coventry, CV4 7AL, UK

Abstract
In this work we introduce a new protocol for vector aggregation in the context of the Shuffle Model, a recent model within Differential Privacy (DP). It sits between the Centralized Model, which prioritizes the level of accuracy over the secrecy of the data, and the Local Model, for which an improvement in trust is counteracted by a much higher noise requirement. The Shuffle Model was developed to provide a good balance between these two models through the addition of a shuffling step, which unbinds the users from their data whilst maintaining a moderate noise requirement. We provide a single message protocol for the summation of real vectors in the Shuffle Model, using advanced composition results. Our contribution provides a mechanism to enable private aggregation and analysis across more sophisticated structures such as matrices and higher-dimensional tensors, both of which are reliant on the functionality of the vector case.

Keywords
Differential privacy (DP), single-message shuffle model, local randomizer, randomized response, mean squared error (MSE).

1. Introduction

Differential Privacy (DP) [1] is a strong, mathematical definition of privacy that guarantees a measurable level of confidentiality for any data subject in the dataset to which it is applied. In this way, useful collective information can be learned about a population, whilst simultaneously protecting the personal information of each data subject.

In particular, DP guarantees that the impact on any particular individual as a result of analysis on a dataset is the same, whether or not the individual is included in the dataset. This guarantee is quantified by a parameter ε, which represents good privacy if it is small. However, finding an algorithm that achieves DP often requires a trade-off between privacy and accuracy, as a smaller ε sacrifices accuracy for better privacy, and vice versa. DP enables data analyses such as the statistical analysis of the salaries of a population. This allows useful collective information to be studied, as long as ε is adjusted appropriately to satisfy the definition of DP.

In this work we focus on protocols in the *Single-Message Shuffle Model* [2], a one-time data collection model where each of n users is permitted to submit a single message. We have chosen to apply the Single-Message Shuffle Model to the problem of *vector aggregation*, as there are links to Federated Learning and Secure Aggregation.

There are many practical applications of the Single-Message Shuffle Model in Federated Learning, where multiple users collaboratively solve a Machine Learning problem, the results of which simultaneously improves the model for the next round [3]. The updates generated by the users after each round are high-dimensional vectors, so this data type will prove useful in applications such as training a Deep Neural Network to predict the next word that a user types [4]. Additionally, aggregation is closely related to Secure Aggregation, which can be used to compute the outputs of Machine Learning problems such as the one above [5].

Our contribution is a protocol in the Single-Message Shuffle Model for the private summation of vector-valued messages, extending an existing result from Balle *et al.* [2] by permitting the n users to each submit a vector of real numbers instead of a scalar. The resulting estimator is unbiased and has normalized mean squared error (MSE) $O_{\varepsilon,\delta}(d^{8/3}n^{-5/3})$, where d is the dimension of each vector.

This vector summation protocol above can be extended to produce a similar protocol for the linearization of matrices. It is important to use matrix reduction to ensure that the constituent vectors are linearly independent. This problem can be extended further to higher-dimensional tensors, which are useful for the representation of multi-dimensional data in Neural Networks.

2. Related Work

The earliest attempts at protecting the privacy of users in a dataset focused on simple ways of suppressing or generalising the data. Examples include k-anonymity [6], l-diversity [7] and t-closeness [8]. However, such attempts

BICOD21: British International Conference on Databases, March 28, 2022, London, UK
mary.p.scott@warwick.ac.uk (M. Scott); g.cormode@warwick.ac.uk (G. Cormode); cm@warwick.ac.uk (C. Maple)
0000-0003-0799-5840 (M. Scott); 0000-0002-0698-0922 (G. Cormode); 0000-0002-4715-212X (C. Maple)

have been shown to be insufficient, as proved by numerous examples [9].

This harmful leakage of sensitive information can be easily prevented through the use of DP, as this mathematically guarantees that the chance of a *linkage attack* on an individual in the dataset is almost identical to that on an individual not in the dataset.

Ever since DP was first conceptualized in 2006 by Dwork *et al.* [1], the majority of research in the field has focused on two opposing models. In the Centralized Model, users submit their sensitive personal information directly to a *trusted* central data collector, who adds *random noise* to the raw data to provide DP, before assembling and analyzing the aggregated results.

In the Local Model, DP is guaranteed when each user applies a *local randomizer* to add random noise to their data before it is submitted. The Local Model differs from the Centralized Model in that the central entity does not see the users' raw data at any point, and therefore does not have to be trusted. However, the level of noise required per user for the same privacy guarantee is much higher, which limits the usage of Local Differential Privacy (LDP) to major companies such as Google [10], Apple [11] and Microsoft [12].

Neither of these two extensively studied models can provide a good balance between the trust of the central entity and the level of noise required to guarantee DP. Hence, in recent years researchers have tried to create intermediate models that reap the benefits of both.

In 2017, Bittau *et al.* [13] introduced the Encode, Shuffle, Analyze (ESA) model, which provides a general framework for the addition of a *shuffling step* in a private protocol. After the data from each user is encoded, it is randomly permuted to unbind each user from their data before analysis takes place. In 2019, Cheu *et al.* [14] formalized the Shuffle Model as a special case of the ESA model, which connects this additional shuffling step to the Local Model. In the Shuffle Model, the local randomizer applies a randomized mechanism on a per-element basis, potentially replacing a truthful value with another randomly selected domain element. The role of these independent reports is to create what we call a *privacy blanket*, which masks the outputs which are reported truthfully.

As well as the result on the private summation of scalar-valued messages in the Single-Message Shuffle Model that we will be using [2], Balle *et al.* have published two more recent works that solve related problems. The first paper [15] improved the distributed n-party summation protocol from Ishai *et al.* [16] in the context of the Single-Message Shuffle Model to require $O(1 + \pi/\log n)$ scalar-valued messages, instead of a logarithmic dependency of $O(\log n + \pi)$, to achieve statistical security $2^{-\pi}$. The second paper [17] introduced two new protocols for the private summation of scalar-valued messages in the Multi-Message Shuffle Model, an extension of the Single-Message Shuffle Model that permits each of the n users to submit more than one message, using several independent shufflers to securely compute the sum. In this work, Balle *et al.* contributed a recursive construction based on the protocol from [2], as well as an alternative mechanism which implements a discretized distributed noise addition technique using the result from Ishai *et al.* [16].

Also relevant to our research is the work of Ghazi *et al.* [18], which explored the related problems of private frequency estimation and selection in a similar context, drawing comparisons between the errors achieved in the Single-Message and Multi-Message Shuffle Models. A similar team of authors produced a follow-up paper [19] describing a more efficient protocol for private summation in the Single-Message Shuffle Model, using the 'invisibility cloak' technique to facilitate the addition of zero-sum noise without coordination between the users.

3. Preliminaries

We consider randomized mechanisms [9] $\mathcal{M}$, $\mathcal{R}$ under domains $\mathbb{X}$, $\mathbb{Y}$, and apply them to input datasets $\vec{D}, \vec{D}'$ to generate (vector-valued) messages $\vec{x}_i, \vec{x}_i'$. We write $[k] = \{1, \ldots, k\}$ and $\mathbb{N}$ for the set of natural numbers.

3.1. Models of Differential Privacy

The essence of Differential Privacy (DP) is the requirement that the contribution $\vec{x}_i$ of a user i to a dataset $\vec{D} = (\vec{x}_1, \ldots, \vec{x}_n)$ does not have much effect on the outcome of the mechanism applied to that dataset.

In the *centralized* model of DP, random noise is only introduced after the users' inputs are gathered by a (trusted) aggregator. Consider a dataset $\vec{D}'$ that differs from $\vec{D}$ only in the contribution of a single user, denoted $\vec{D} \simeq \vec{D}'$. Also let $\varepsilon \geq 0$ and $\delta \in (0, 1)$. We say that a randomized mechanism $\mathcal{M} : \mathbb{X}^n \to \mathbb{Y}$ is (ε, δ)-differentially private if $\forall \vec{D} \simeq \vec{D}', \forall E \subseteq \mathbb{Y}$:

$$\Pr[\mathcal{M}(\vec{D}) \in E] \leq e^{\varepsilon} \cdot \Pr[\mathcal{M}(\vec{D}') \in E] + \delta \, [9].$$

In this definition, we assume that the trusted aggregator obtains the raw data from all users and introduces the necessary perturbations.

In the *local* model of DP, each user i independently uses randomness on their input $\vec{x}_i \in \mathbb{X}$ by using a *local randomizer* $\mathcal{R} : \mathbb{X} \to \mathbb{Y}$ to obtain a perturbed result $\mathcal{R}(\vec{x}_i)$. We say that the local randomizer is (ε, δ)-differentially private if $\forall \vec{D}, \vec{D}', \forall E \subseteq \mathbb{Y}$:

$$\Pr[\mathcal{R}(\vec{x}_i) \in E] \leq e^{\varepsilon} \cdot \Pr[\mathcal{R}(\vec{x}_i') \in E] + \delta \, [2],$$

where $\vec{x}_i' \in \mathbb{X}$ is some other valid input vector that i could hold. The Local Model guarantees that any observer will not have access to the raw data from any of the users. That is, it removes the requirement for trust. The price is that this requires a higher level of noise per user to achieve the same privacy guarantee.

3.2. Single-Message Shuffle Model

The Single-Message Shuffle Model sits in between the Centralized and Local Models of DP [2]. Let a protocol $\mathscr{P}$ in the Single-Message Shuffle Model be of the form $\mathscr{P} = (\mathscr{R}, \mathscr{A})$, where $\mathscr{R} : \mathbb{X} \to \mathbb{Y}$ is the *local randomizer*, and $\mathscr{A} : \mathbb{Y}^n \to \mathbb{Z}$ is the *analyzer* of $\mathscr{P}$. Overall, $\mathscr{P}$ implements a mechanism $\mathscr{P} : \mathbb{X}^n \to \mathbb{Z}$ as follows. Each user i independently applies the local randomizer to their message $\vec{x}_i$ to obtain a message $\vec{y}_i = \mathscr{R}(\vec{x}_i)$. Subsequently, the messages $(\vec{y}_1, \dots, \vec{y}_n)$ are randomly permuted by a trusted *shuffler* $\mathcal{S} : \mathbb{Y}^n \to \mathbb{Y}^n$. The random permutation $\mathcal{S}(\vec{y}_1, \dots, \vec{y}_n)$ is submitted to an untrusted data collector, who applies the analyzer $\mathscr{A}$ to obtain an output for the mechanism. In summary, the output of $\mathscr{P}(\vec{x}_1, \dots, \vec{x}_n)$ is given by:

$$\mathscr{A} \circ \mathcal{S} \circ \mathscr{R}^n(\vec{x}) = \mathscr{A}(\mathcal{S}(\mathscr{R}(\vec{x}_1), \dots, \mathscr{R}(\vec{x}_n))).$$

Note that the data collector observing the shuffled messages $\mathcal{S}(\vec{y}_1, \dots, \vec{y}_n)$ obtains no information about which user generated each of the messages. Therefore, the privacy of $\mathscr{P}$ relies on the indistinguishability between the shuffles $\mathcal{S} \circ \mathscr{R}^n(\vec{D})$ and $\mathcal{S} \circ \mathscr{R}^n(\vec{D}')$ for datasets $\vec{D} \simeq \vec{D}'$. The analyzer can represent the shuffled messages as a *histogram*, which counts the number of occurrences of the possible outputs of $\mathbb{Y}$.

3.3. Measuring Accuracy

In Section 4 we use the *mean squared error* to compare the overall output of a private summation protocol in the Single-Message Shuffle Model with the original dataset. The MSE is used to measure the average squared difference in the comparison between a fixed input $f(\vec{D})$ to the randomized protocol $\mathscr{P}$, and its output $\mathscr{P}(\vec{D})$. In this context, $\mathrm{MSE}(\mathscr{P}, \vec{D}) = \mathsf{E}\left[(\mathscr{P}(\vec{D}) - f(\vec{D}))^2\right]$, where the expectation is taken over the randomness of $\mathscr{P}$. Note when $\mathsf{E}[\mathscr{P}(\vec{D})] = f(\vec{D})$, MSE is equivalent to variance, i.e.:

$$\mathrm{MSE}(\mathscr{P}, \vec{D}) = \mathsf{E}\left[(\mathscr{P}(\vec{D}) - \mathsf{E}[\mathscr{P}(\vec{D})])^2\right] = \mathrm{Var}[\mathscr{P}(\vec{D})].$$

4. Vector Sum in the Shuffle Model

In this section we introduce our protocol for vector summation in the Shuffle Model and tune its parameters to optimize accuracy.

Algorithm 1: Local Randomizer $\mathscr{R}^{PH}_{\gamma,k,n}$

```
Public Parameters:
  $\gamma \in [0, 1]$, domain size $k$, and number of
  parties $n$
Input: $x_i \in [k]$
Output: $y_i \in [k]$
Sample $b \leftarrow \texttt{Ber}(\gamma)$
if $b = 0$ then let $y_i \leftarrow x_i$
else sample $y_i \leftarrow \texttt{Unif}([k])$
return $y_i$
```

4.1. Basic Randomizer

First, we describe a basic local randomizer applied by each user i to an input $x_i \in [k]$. The output of this protocol is a (private) histogram of shuffled messages over the domain $[k]$.

The Local Randomizer $\mathscr{R}^{PH}_{\gamma,k,n}$, shown in Algorithm 1, applies a generalized *randomized response* mechanism that returns the true message x_i with probability $1 - \gamma$ and a uniformly random message with probability γ. Such a basic randomizer is used by Balle *et al.* [2] in the Single-Message Shuffle Model for scalar-valued messages, as well as in several other previous works in the Local Model [20, 21, 22]. In Section 4.3, we find an appropriate γ to optimize the proportion of random messages that are submitted, and therefore guarantee DP.

We now describe how the presence of these random messages can form a 'privacy blanket' to protect against a *difference attack* on a particular user. Suppose we apply Algorithm 1 to the messages from all n users. Note that a subset B of approximately γn of these users returned a uniformly random message, while the remaining users returned their true message. Following Balle *et al.* [2], the analyzer can represent the messages sent by users in B by a histogram Y_1 of uniformly random messages, and can form a histogram Y_2 of truthful messages from users not in B. As these subsets are mutually exclusive and collectively exhaustive, the information represented by the analyzer is equivalent to the histogram $Y = Y_1 \cup Y_2$.

Consider two neighbouring datasets, each consisting of n messages from n users, that differ only on the input from the nth user. To simplify the discussion and subsequent proof, we temporarily omit the action of the shuffler. By the post-processing property of DP, this can be reintroduced later on without adversely affecting the privacy guarantees. To achieve DP we need to find an appropriate γ such that when Algorithm 1 is applied, the change in Y is appropriately bounded. As the knowledge of either the set B or the messages from the first $n - 1$ users does not affect DP, we can assume that the analyzer knows both of these details. This lets the analyzer remove all of the truthful messages associated with the first $n-1$ users from Y.

If the nth user is in B, this means their submission is independent of their input, so we trivially satisfy DP. Otherwise, the (curious) analyzer knows that the nth user has submitted their true message x_n. The analyzer can remove all of the truthful messages associated with the first $n-1$ users from Y, and obtain $Y_1 \cup \{x_n\}$. The subsequent privacy analysis will argue that this does not reveal x_n if γ is set so that Y_1, the histogram of random messages, appropriately 'hides' x_n.

4.2. Private Summation of Vectors

Here, we extend the protocol from Section 4.1 to address the problem of computing the sum of n real vectors, each of the form $\vec{x}_i = (x_i^{(1)}, \dots, x_i^{(d)}) \in [0,1]^d$, in the Single-Message Shuffle Model. Specifically, we analyze the utility of a protocol $\mathcal{P}_{d,k,n,t} = (\mathcal{R}_{d,k,n,t}, \mathcal{A}_{d,k,t})$ for this purpose, by using the MSE from Section 3.3 as the accuracy measure. In the scalar case, each user applies the protocol to their entire input [2]. Moving to the vector case, we allow each user to independently sample a set of $1 \le t \le d$ coordinates from their vector to report. Our analysis allows us to optimize the parameter t.

Hence, the first step of the Local Randomizer $\mathcal{R}_{d,k,n,t}$, presented in Algorithm 2, is to uniformly sample t coordinates $(\alpha_1, \dots, \alpha_t) \in [d]$ (without replacement) from each vector $\vec{x}_i$. To compute a differentially private approximation of $\sum_i \vec{x}_i$, we fix a quantization level k. Then we randomly round each $x_i^{(\alpha_j)}$ to obtain $\bar{x}_i^{(\alpha_j)}$ as either $\lfloor x_i^{(\alpha_j)} k \rfloor$ or $\lceil x_i^{(\alpha_j)} k \rceil$. Next, we apply the randomized response mechanism from Algorithm 1 to each $\bar{x}_i^{(\alpha_j)}$, which sets each output $y_i^{(\alpha_j)}$ independently to be equal to $\bar{x}_i^{(\alpha_j)}$ with probability $1-\gamma$, or a random value in $\{0,1,\dots,k\}$ with probability γ. Each $y_i^{(\alpha_j)}$ will contribute to a histogram of the form $(y_1^{(\alpha_j)}, \dots, y_n^{(\alpha_j)})$ as in Section 4.1.

The Analyzer $\mathcal{A}_{d,k,t}$, shown in Algorithm 3, aggregates the histograms to approximate $\sum_i \vec{x}_i$ by post-processing the vectors coordinate-wise. More precisely, the analyzer sets each output $y_i^{(\alpha_j)}$ to $y_i^{(l)}$, where the new label l is from its corresponding input $x_i^{(l)}$ of the original d-dimensional vector $\vec{x}_i$. For all inputs $x_i^{(l)}$ that were not sampled, we set $y_i^{(l)} = 0$. Subsequently, the analyzer aggregates the sets of outputs from all users corresponding to each of those l coordinates in turn, so that a d-dimensional vector is formed. Finally, a standard debiasing step is applied to this vector to remove the scaling and rounding applied to each submission. DeBias returns an unbiased estimator, $\vec{z}$, which calculates an estimate of the true sum of the vectors by subtracting the expected uniform noise from the randomized sum of the vectors.

Algorithm 2: Local Randomizer $\mathcal{R}_{d,k,n,t}$

```
Public Parameters: $k$, $t$, dimension $d$, and
  number of parties $n$
Input: $\vec{x}_i = (x_i^{(1)}, \dots, x_i^{(d)}) \in [0,1]^d$
Output: $\vec{y}_i = (y_i^{(\alpha_1)}, \dots, y_i^{(\alpha_t)}) \in \{0,1,\dots,k\}^t$

Sample $(\alpha_1, \dots, \alpha_t) \leftarrow$ Unif($[d]$)
Let $\bar{x}_i^{(\alpha_j)} \leftarrow \lfloor x_i^{(\alpha_j)} k \rfloor +$ Ber($x_i^{(\alpha_j)} k - \lfloor x_i^{(\alpha_j)} k \rfloor$)
  ▷ $\bar{x}_i^{(\alpha_j)}$: encoding of $x_i^{(\alpha_j)}$ with precision $k$
  ▷ $y_i^{(\alpha_j)}$: apply Algorithm 1 to each $\bar{x}_i^{(\alpha_j)}$
return $\vec{y}_i = (y_i^{(\alpha_1)}, \dots, y_i^{(\alpha_t)})$
```

Algorithm 3: Analyzer $\mathcal{A}_{d,k,t}$

```
Public Parameters: $k$, $t$, and dimension $d$
Input: Multiset $\{\vec{y}_i\}_{i \in [n]}$, with
    $(y_i^{(\alpha_1)}, \dots, y_i^{(\alpha_t)}) \in \{0,1,\dots,k\}^t$
Output: $\vec{z} = (z^{(1)}, \dots, z^{(d)}) \in [0,1]^d$

Let $y_i^{(l)} \leftarrow y_i^{(\alpha_j)}$
  ▷ $y_i^{(\alpha_j)}$: submission corresponding to $x_i^{(l)}$

Let $(\hat{z}^{(1)}, \dots, \hat{z}^{(d)}) \leftarrow (\frac{1}{k}\sum_i y_i^{(1)}, \dots, \frac{1}{k}\sum_i y_i^{(d)})$
Let
  $(z^{(1)}, \dots, z^{(d)}) \leftarrow$ (DeBias($\hat{z}^{(1)}$), … ,DeBias($\hat{z}^{(d)}$))
  ▷ DeBias($\hat{z}^{(l)}$) $= (\hat{z}^{(l)} - \frac{\gamma}{2} \cdot |y_i^{(l)}|)/(1-\gamma)$
return $\vec{z} = (z^{(1)}, \dots, z^{(d)})$
```

4.3. Privacy Analysis of Algorithms

In this section, we will find an appropriate γ that ensures that the mechanism described in Algorithms 2 and 3 satisfies (ε, δ)-DP for vector-valued messages in the Single-Message Shuffle Model. To achieve this, we prove the following theorem, where we initially assume $\varepsilon < 1$ to simplify our computations. At the end of this section, we discuss how to cover the additional case $1 \le \varepsilon < 6$ to suit our experimental study.

Theorem 4.1. *The shuffled mechanism $\mathcal{M} = \mathcal{S} \circ \mathcal{R}_{d,k,n,t}$ is (ε, δ)-DP for any $d, k, n \in \mathbb{N}$, $\{t \in \mathbb{N} \mid t \in [d]\}$, $\varepsilon < 6$ and $\delta \in (0,1]$ such that:*

$$\gamma = \begin{cases} \frac{56dk\log(1/\delta)\log(2t/\delta)}{(n-1)\varepsilon^2}, & \text{when } \varepsilon < 1 \\ \frac{2016dk\log(1/\delta)\log(2t/\delta)}{(n-1)\varepsilon^2}, & \text{when } 1 \le \varepsilon < 6. \end{cases}$$

Proof. Let $\vec{D} = (\vec{x}_1, \dots, \vec{x}_n)$ and $\vec{D}' = (\vec{x}_1, \dots, \vec{x}_n')$ be the two neighbouring datasets differing only in the input of the nth user, as used in Section 4.1. Here each vector-valued message $\vec{x}_i$ is of the form $(x_i^{(1)}, \dots, x_i^{(d)})$. Recall from Section 4.1 that we assume that the analyzer can see the users in B (i.e., the subset of users that returned a uniformly random message), as well as the inputs from the first $n-1$ users.

We now introduce the *vector view* $\mathrm{VView}_{\mathcal{M}}(\vec{D})$ as the collection of information that the analyzer is able to see after the mechanism $\mathcal{M}$ is applied to all vector-valued messages in the dataset $\vec{D}$. $\mathrm{VView}_{\mathcal{M}}(\vec{D})$ is defined as the tuple $(\vec{Y}, \vec{D}_\cap, \vec{b})$, where $\vec{Y}$ is the multiset containing the outputs $\{\vec{y}_1, \ldots, \vec{y}_n\}$ of the mechanism $\mathcal{M}(\vec{D})$, $\vec{D}_\cap$ is the vector containing the inputs $(\vec{x}_1, \ldots, \vec{x}_{n-1})$ from the first $n-1$ users, and $\vec{b}$ contains binary vectors $(\vec{b}_1, \ldots, \vec{b}_n)$ which indicate for which coordinates each user reports truthful information. This vector view can be projected to t overlapping *scalar views* by applying Algorithm 2 only to the j^{th} uniformly sampled coordinate $\alpha_j \in [d]$ from each user, where $j \in [t]$. The j^{th} scalar view $\mathrm{View}_{\mathcal{M}}^{(\alpha_j)}(\vec{D})$ of $\mathrm{VView}_{\mathcal{M}}(\vec{D})$ is defined as the tuple $(\vec{Y}^{(\alpha_j)}, \vec{D}_\cap^{(\alpha_j)}, \vec{b}^{(\alpha_j)})$, where:

$$\begin{aligned} \vec{Y}^{(\alpha_j)} &= \mathcal{M}(\vec{D}^{(\alpha_j)}) = \{y_1^{(\alpha_j)}, \ldots, y_n^{(\alpha_j)}\}, \\ \vec{D}_\cap^{(\alpha_j)} &= (x_1^{(\alpha_j)}, \ldots, x_{n-1}^{(\alpha_j)}) \\ \text{and} \quad \vec{b}^{(\alpha_j)} &= (b_1^{(\alpha_j)}, \ldots, b_n^{(\alpha_j)}) \end{aligned}$$

are the analogous definitions of $\vec{Y}$, $\vec{D}_\cap$ and $\vec{b}$, but containing only the information referring to the j^{th} uniformly sampled coordinate of each vector-valued message.

The following *advanced composition* results will be used in our setting to get a tight upper bound:

Theorem 4.2 (Dwork *et al.* [9]). *For all $\varepsilon', \delta', \delta \geq 0$, the class of (ε', δ')-differentially private mechanisms satisfies $(\varepsilon, r\delta' + \delta)$-differential privacy under r-fold adaptive composition for:*

$$\varepsilon = \sqrt{2r\log(1/\delta)}\varepsilon' + r\varepsilon'\left(e^{\varepsilon'} - 1\right).$$

Corollary 4.3. *Given target privacy parameters $0 < \varepsilon < 1$ and $\delta > 0$, to ensure $(\varepsilon, r\delta' + \delta)$ cumulative privacy loss over r mechanisms, it suffices that each mechanism is (ε', δ')-DP, where:*

$$\varepsilon' = \frac{\varepsilon}{2\sqrt{2r\log(1/\delta)}}.$$

To show that $\mathrm{VView}_{\mathcal{M}}(\vec{D})$ satisfies (ε, δ)-DP it suffices to prove that:

$$\Pr_{\widetilde{\mathrm{V}} \sim \mathrm{VView}_{\mathcal{M}}(\vec{D})}\left[\frac{\Pr[\mathrm{VView}_{\mathcal{M}}(\vec{D}) = \widetilde{\mathrm{V}}]}{\Pr[\mathrm{VView}_{\mathcal{M}}(\vec{D}') = \widetilde{\mathrm{V}}]} \geq e^{\varepsilon}\right] \leq \delta. \quad (1)$$

By considering this vector view as a union of overlapping scalar views, and letting $r = t$ in Corollary 4.3, it is sufficient to derive (1) from:

$$\Pr_{\mathrm{V}_{\alpha_j} \sim \mathrm{View}_{\mathcal{M}}^{(\alpha_j)}(\vec{D})}\left[\frac{\Pr[\mathrm{View}_{\mathcal{M}}^{(\alpha_j)}(\vec{D}) = \mathrm{V}_{\alpha_j}]}{\Pr[\mathrm{View}_{\mathcal{M}}^{(\alpha_j)}(\vec{D}') = \mathrm{V}_{\alpha_j}]} \geq e^{\varepsilon'}\right] \leq \delta', \quad (2)$$

where $\widetilde{\mathrm{V}} = \bigcup_{\alpha_j} \mathrm{V}_{\alpha_j}$, $\varepsilon' = \frac{\varepsilon}{2\sqrt{2t\log(1/\delta)}}$ and $\delta' = \frac{\delta}{t}$.

Lemma 4.4. *Condition (2) implies condition (1).*

Proof. We can express $\mathrm{VView}_{\mathcal{M}}(\vec{D})$ as the composition of the t scalar views $\mathrm{View}_{\mathcal{M}}^{(\alpha_1)}, \ldots, \mathrm{View}_{\mathcal{M}}^{(\alpha_t)}$, as:

$$\begin{aligned} &\Pr[\mathrm{VView}_{\mathcal{M}}(\vec{D}) = \widetilde{\mathrm{V}}] \\ &\quad = \Pr[\mathrm{View}_{\mathcal{M}}^{(\alpha_1)}(\vec{D}) = \mathrm{V}_{\alpha_1} \wedge \cdots \wedge \mathrm{View}_{\mathcal{M}}^{(\alpha_t)}(\vec{D}) = \mathrm{V}_{\alpha_t}] \\ &\quad = \Pr[\mathrm{View}_{\mathcal{M}}^{(\alpha_1)}(\vec{D}) = \mathrm{V}_{\alpha_1}] \cdot \cdots \cdot \Pr[\mathrm{View}_{\mathcal{M}}^{(\alpha_t)}(\vec{D}) = \mathrm{V}_{\alpha_t}]. \end{aligned}$$

Our desired result is immediate by applying Corollary 4.3, which states that the use of t overlapping (ε', δ')-DP mechanisms, when taken together, is (ε, δ)-DP. This applies in our setting, since we have assumed that $\mathrm{VView}_{\mathcal{M}}(\vec{D})$ satisfies the requirements of (ε, δ)-DP, and that each of the t overlapping scalar views is formed identically but for a different uniformly sampled coordinate of the vector-valued messages. □

To complete the proof of Theorem 4.1 for $\varepsilon < 1$, it remains to show that for a uniformly sampled coordinate $\alpha_j \in [d]$, $\mathrm{View}_{\mathcal{M}}^{(\alpha_j)}(\vec{D})$ satisfies (ε', δ')-DP.

Lemma 4.5. *Condition (2) holds.*

Proof. See Appendix A. □

We now show that the above proof can be adjusted to cover the additional case $1 \leq \varepsilon < 6$. This will be sufficient to complete the proof of our main Theorem 4.1.

First, we scale the setting of ε' by a multiple of 6 in Corollary 4.3 so that the advanced composition property holds for all $1 \leq \varepsilon < 6$. We now insert $\varepsilon' = \frac{\varepsilon}{12\sqrt{2r\log(1/\delta)}}$ into the proof of Theorem 4.1, resulting in a change of constant from 56 to 2016. □

4.4. Accuracy Bounds for Shuffled Vector Sum

We now formulate an upper bound for the MSE of our protocol, and then identify the value(s) of t that minimize this upper bound.

First, note that encoding the coordinate $x_i^{(\alpha_j)}$ as $\bar{x}_i^{(\alpha_j)} = \lfloor x_i^{(\alpha_j)} k \rfloor + \mathtt{Ber}(x_i^{(\alpha_j)} k - \lfloor x_i^{(\alpha_j)} k \rfloor)$ in Algorithm 2 ensures that $\mathbb{E}[\bar{x}_i^{(\alpha_j)}/k] = \mathbb{E}[x_i^{(\alpha_j)}]$. This means that our protocol is unbiased. For any unbiased random variable X with

$a < X < b$ then $\mathrm{Var}[X] \le (b-a)^2/4$, and so the MSE per coordinate due to the fixed-point approximation of the true vector in $\mathscr{R}_{d,k,n,t}$ is at most $\frac{1}{4k^2}$. Meanwhile, the MSE when $\mathscr{R}_{d,k,n,t}$ submits a random vector is at most $\frac{1}{2}$ per coordinate.

We now use the unbiasedness of our protocol to obtain a result for estimating the squared error between the estimated average vector and the true average vector. When calculating the MSE, each coordinate location is used with expectation n/d. Therefore, we define the *normalized* MSE, or $\widehat{\mathrm{MSE}}$, as the normalization of the MSE by a factor of $(n/d)^2$.

Theorem 4.6. *For any $d, n \in \mathbb{N}$, $\{t \in \mathbb{N} \mid t \in [d]\}$, $\varepsilon < 6$ and $\delta \in (0,1]$, there exists a parameter k such that $\mathscr{P}_{d,k,n,t}$ is (ε,δ)-DP and*

$$\widehat{\mathrm{MSE}}(\mathscr{P}_{d,k,n,t}) = \begin{cases} \frac{2td^{8/3}(14\log(1/\delta)\log(2t/\delta))^{2/3}}{(1-\gamma)^2 n^{5/3}\varepsilon^{4/3}}, \\ \quad \text{when } \varepsilon < 1 \\ \frac{8td^{8/3}(63\log(1/\delta)\log(2t/\delta))^{2/3}}{(1-\gamma)^2 n^{5/3}\varepsilon^{4/3}}, \\ \quad \text{when } 1 \le \varepsilon < 6, \end{cases}$$

where $\widehat{\mathrm{MSE}}$ denotes the squared error between the estimated average vector and the true average vector.

Proof. We consider the $\sum_{l=1}^{d} \mathtt{DeBias}(\hat{z}^{(l)})$ of $\mathscr{P}_{d,k,n,t}$ compared to the corresponding input $\sum_{j=1}^{t}\sum_{i=1}^{n} x_i^{(\alpha_j)}$ over the dataset $\vec{D}$. We use the bounds on the variance of the randomized response mechanism from Theorem 4.6 to give us an upper bound for this comparison.

$$\begin{aligned} \mathrm{MSE}(\mathscr{P}_{d,k,n,t}) &= \sup_{\vec{D}} \mathsf{E}\left[\left(\sum_{l=1}^{d} \mathtt{DeBias}(\hat{z}^{(l)}) - \sum_{j=1}^{t}\sum_{i=1}^{n} x_i^{(\alpha_j)}\right)^2\right] \\ &= \sup_{\vec{D}} \mathsf{E}\left[\left(\sum_{j=1}^{t}\sum_{i=1}^{n}\left(\mathtt{DeBias}(y_i^{(\alpha_j)}/k) - x_i^{(\alpha_j)}\right)\right)^2\right] \\ &= \sup_{\vec{D}} \sum_{j=1}^{t}\sum_{i=1}^{n} \mathsf{E}\left[\left(\mathtt{DeBias}(y_i^{(\alpha_j)}/k) - x_i^{(\alpha_j)}\right)^2\right] \\ &= \sup_{\vec{D}} \sum_{j=1}^{t}\sum_{i=1}^{n} \mathrm{Var}\left[\mathtt{DeBias}(y_i^{(\alpha_j)}/k)\right] \\ &= \frac{tn}{(1-\gamma)^2} \sup_{x_1^{(\alpha_1)}} \mathrm{Var}[y_1^{(\alpha_1)}/k] \le \frac{tn}{(1-\gamma)^2}\left(\frac{1-\gamma}{4k^2} + \frac{\gamma}{2}\right) \\ &\le \frac{tn}{(1-\gamma)^2}\left(\frac{1}{4k^2} + \frac{A_\varepsilon dk\log(1/\delta)\log(2t/\delta)}{(n-1)\varepsilon^2}\right), \end{aligned}$$

where $A_\varepsilon = 28$ when $\varepsilon < 1$, and $A_\varepsilon = 1008$ when $1 \le \varepsilon < 6$. In other words, A_ε is equal to half the constant term in the expression of γ stated in Theorem 4.1. The choice $k = \frac{(n-1)\varepsilon^2}{4A_\varepsilon d\log(1/\delta)\log(2t/\delta)}$ minimizes the bracketed sum above and the bounds in the statement of the theorem follow. □

To obtain the error between the estimated average vector and the true average vector, we simply take the square root of the result obtained in Theorem 4.6.

Corollary 4.7. *For every statistical query $q : \mathcal{X} \mapsto [0,1]^d$, $d, n \in \mathbb{N}$, $\{t \in \mathbb{N} \mid t \in [d]\}$, $\varepsilon < 6$ and $\delta \in (0,1]$, there is an (ε,δ)-DP n-party unbiased protocol for estimating $\frac{d}{n}\sum_i q(\vec{x}_i)$ in the Single-Message Shuffle Model with standard deviation*

$$\hat{\sigma}(\mathscr{P}_{d,k,n,t}) = \begin{cases} \frac{(2t)^{1/2}d^{4/3}(14\log(1/\delta)\log(2t/\delta))^{1/3}}{(1-\gamma)n^{5/6}\varepsilon^{2/3}}, \\ \quad \text{when } \varepsilon < 1 \\ \frac{(8t)^{1/2}d^{4/3}(63\log(1/\delta)\log(2t/\delta))^{1/3}}{(1-\gamma)n^{5/6}\varepsilon^{2/3}}, \\ \quad \text{when } 1 \le \varepsilon < 6, \end{cases}$$

where $\hat{\sigma}$ denotes the error between the estimated average vector and the true average vector.

To summarize, we have produced an unbiased protocol for the computation of the sum of n real vectors in the Single-Message Shuffle Model with normalized MSE $O_{\varepsilon,\delta}(d^{8/3}tn^{-5/3})$, using advanced composition results from Dwork *et al.* [9]. Minimizing this bound as a function of t leads us to choose $t = 1$, but any choice of t that is small and not dependent on d produces a bound of the same order. In our experimental study, we determine that the best choice of t in practice is indeed $t = 1$.

4.5. Improved bounds for t=1

We observe that in the optimal case in which $t = 1$, we can tighten the bounds further, as we do not need to invoke the advanced composition results when each user samples only a single coordinate. This changes the value of γ by a factor of $O(\log(1/\delta))$, which propagates through to the expression for the MSE. That is, we can more simply set $\varepsilon' = \varepsilon$ and $\delta' = \delta$ in the proof of Theorem 4.1. When $\varepsilon < 1$, the computation is straightforward, with $c \ge \frac{14}{\varepsilon'^2}\log(2t/\delta)$ being chosen as before. However, when $1 \le \varepsilon < 6$, a tighter $c \ge \frac{80}{\varepsilon'^2}\log(2t/\delta)$ must be selected, as the condition $\varepsilon' < 1$ no longer holds.

Using $\varepsilon' < 6$, we have:

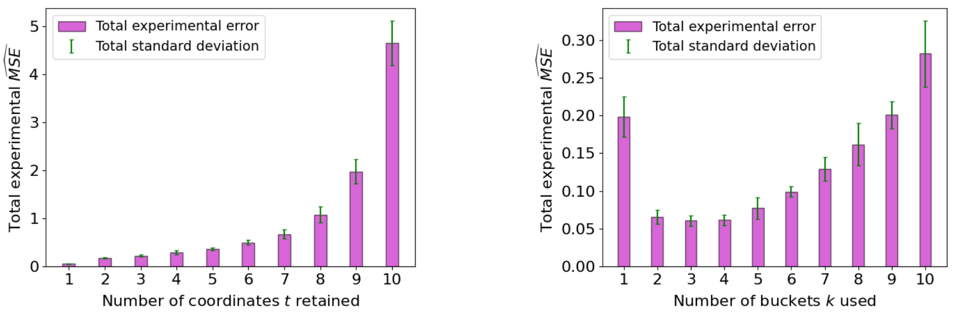
$$(1 - \exp(-\varepsilon'/2)) \ge \left(1 - \exp\left(-\frac{2}{3\sqrt{15}}\right)\right)\varepsilon' \ge \frac{\varepsilon'}{2\sqrt{10}}.$$

Thus, we have:

$$\begin{aligned} \Pr\left[\frac{\mathsf{N}_\theta}{\mathsf{N}_\phi} \ge e^{\varepsilon'}\right] &\le \exp\Big(-\frac{c}{3}(\varepsilon'/2)^2\Big) + \exp\Big(-\frac{c}{2}\Big(\frac{\varepsilon'}{2\sqrt{10}}\Big)^2\Big) \\ &\le 2\exp\left(-\frac{80}{2\varepsilon'^2}\frac{\varepsilon'^2}{40}\log(2t/\delta)\right) \le \delta/t, \end{aligned}$$

which yields:

$$\gamma = \begin{cases} \max\left\{\frac{14dk\log(2/\delta)}{(n-1)\varepsilon^2}, \frac{27dk}{(n-1)\varepsilon}\right\}, & \text{when } \varepsilon < 1 \\ \max\left\{\frac{80dk\log(2/\delta)}{(n-1)\varepsilon^2}, \frac{36dk}{11(n-1)\varepsilon}\right\}, & \text{when } 1 \le \varepsilon < 6. \end{cases}$$



(a) Experimental error by number of coordinates t retained

(b) Experimental error by number of buckets k used

Figure 1: Bar charts confirming that the choices $t = 1$ (a) and $k = 3$ (b) minimize the total experimental $\widehat{\text{MSE}}$ for the ECG Heartbeat Categorization Dataset.

Note that the above expression for γ in the case $\varepsilon < 1$ coincides with the result obtained by Balle *et al.* in the scalar case [2]. Putting this expression for γ in the proof of Theorem 4.6, with the choice

$$k = \begin{cases} \min\Big\{\Big(\frac{n\varepsilon^2}{28d\log(2/\delta)}\Big)^{1/3}, \Big(\frac{n\varepsilon}{54d}\Big)^{1/3}\Big\}, & \text{when } \varepsilon < 1 \\ \min\Big\{\Big(\frac{n\varepsilon^2}{160d\log(2/\delta)}\Big)^{1/3}, \Big(\frac{11n\varepsilon}{72d}\Big)^{1/3}\Big\}, & \text{when } 1 \le \varepsilon < 6, \end{cases}$$

causes the upper bound on the normalized MSE to reduce to:

$$\widehat{\text{MSE}} = \begin{cases} \max\Big\{\frac{98^{1/3}d^{8/3}\log^{2/3}(2/\delta)}{(1-\gamma)^2 n^{5/3}\varepsilon^{4/3}}, \frac{18d^{8/3}}{(1-\gamma)^2 n^{5/3}(4\varepsilon)^{2/3}}\Big\}, \\ \quad \text{when } \varepsilon < 1 \\ \max\Big\{\frac{2d^{8/3}(20\log(2/\delta))^{2/3}}{(1-\gamma)^2 n^{5/3}\varepsilon^{4/3}}, \frac{2(9^{2/3})d^{8/3}}{(1-\gamma)^2 n^{5/3}(11\varepsilon)^{2/3}}\Big\}, \\ \quad \text{when } 1 \le \varepsilon < 6. \end{cases}$$

By updating Corollary 4.7 in the same way, we can conclude that for the optimal choice $t = 1$, the normalized standard deviation of our unbiased protocol can be further tightened to:

$$\hat{\sigma} = \begin{cases} \max\Big\{\frac{98^{1/6}d^{4/3}\log^{1/3}(2/\delta)}{(1-\gamma)n^{5/6}\varepsilon^{2/3}}, \frac{18^{1/2}d^{4/3}}{(1-\gamma)n^{5/6}(4\varepsilon)^{1/3}}\Big\}, \\ \quad \text{when } \varepsilon < 1 \\ \max\Big\{\frac{2^{1/2}d^{4/3}(20\log(2/\delta))^{1/3}}{(1-\gamma)n^{5/6}\varepsilon^{2/3}}, \frac{2^{1/2}9^{1/3}d^{4/3}}{(1-\gamma)n^{5/6}(11\varepsilon)^{1/3}}\Big\}, \\ \quad \text{when } 1 \le \varepsilon < 6. \end{cases}$$

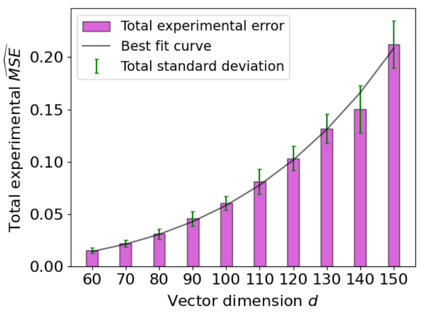
5. Experimental Evaluation

In this section we present and compare the bounds generated by applying Algorithms 2 and 3 to an ECG Heartbeat Categorization Dataset in Python, available at https://www.kaggle.com/shayanfazeli/heartbeat. We analyse the effect of changing one key parameter at a time, whilst the others remain the same. Our default settings are vector dimension $d = 100$, rounding parameter $k = 3$, number of users $n = 50000$, number of coordinates to sample $t = 1$, and differential privacy parameters $\varepsilon = 0.95$ and $\delta = 0.5$. The ranges of all parameters have been adjusted to best display the dependencies, whilst simultaneously ensuring that the parameter γ of the randomized response mechanism is always within its permitted range of $[0, 1]$. The Python code is available at https://github.com/mary-python/dft/blob/master/shuffle.

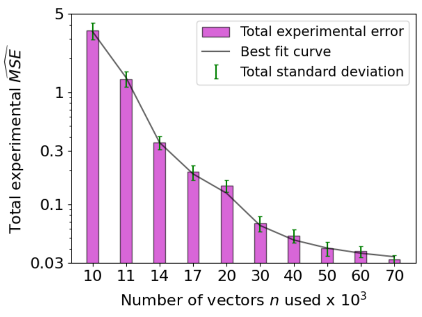
We first confirm that the choice of $t = 1$ is optimal, as predicted by the results of Section 4.5. Indeed, Fig. 1 (a) shows that the total experimental $\widehat{\text{MSE}}$ for the ECG Heartbeat Categorization Dataset is significantly smaller when $t = 1$, compared to any other small value of t.

Similarly, Fig. 1 (b) suggests that the total experimental $\widehat{\text{MSE}}$ is lowest when $k = 3$, which is sufficiently close to the choice of k selected in the proof of Theorem 4.6, with all other default parameter values substituted in. Observe that the absolute value of the observed MSE is below 0.3 in this case, meaning that the vector is reconstructed to a high degree of accuracy, sufficient for many applications.

Next, we verify the bounds of $d^{8/3}$ and $n^{-5/3}$ from Theorem 4.6. Fig. 2 (a) is plotted with a best fit curve with equation a multiple of $d^{8/3}$, exactly as desired. Unsurprisingly, the MSE increases as d goes up according to this superlinear dependence. Meanwhile, Fig. 2 (b) fits a curve dependent on $n^{-7/6}$, sufficiently close to the required result. We see the benefit of increasing n: as n increases by a factor of 10 across the plot, the error decreases by more than two orders of magnitude. In Fig. 3, we verify the dependency $\varepsilon^{-4/3}$ in the two ranges $\varepsilon < 1$ and $1 \le \varepsilon < 6$. The behavior for $\varepsilon < 1$ is quite smooth,

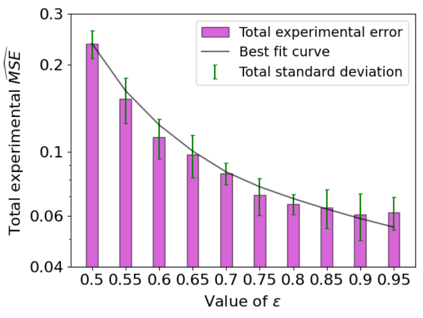


(a) Experimental error by vector dimension d

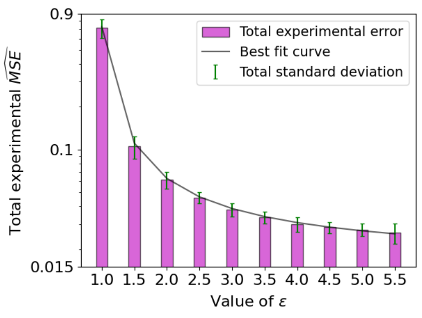


(b) Experimental error by number of vectors n used

Figure 2: Bar charts with best fit curves confirming the dependencies $d^{8/3}$ (a) and $n^{-5/3}$ (b) from Theorem 4.6.



(a) Experimental error by value of ε where $\varepsilon < 1$



(b) Experimental error by value of ε where $1 \leq \varepsilon < 6$

Figure 3: Bar charts with best fit curves confirming the dependency $\varepsilon^{-4/3}$ from Theorem 4.6 in the two ranges $\varepsilon < 1$ (a) and $1 \leq \varepsilon < 6$ (b).

but becomes more variable for larger ε values.

In conclusion, these experiments confirm that picking $t = 1$ and $k = 3$ serves to minimize the error. The lines of best fit confirm the dependencies on the other parameters from Section 4 for d, ε and n, by implementing and applying Algorithms 2 and 3 to an ECG Heartbeat Categorization Dataset in Python. The experiments demonstrate that the MSE observed in practice is sufficiently small to allow effective reconstruction of average vectors for a suitably large cohort of users.

6. Conclusion

Our results extend a result from Balle *et al.* [2] for scalar sums to provide a protocol $\mathscr{P}_{d,k,n,t}$ in the Single-Message Shuffle Model for the private summation of vector-valued messages $(\vec{x}_1, \ldots, \vec{x}_n) \in ([0,1]^d)^n$. It is not surprising that the normalized MSE of the resulting estimator has a dependence on $n^{-5/3}$, as this was the case for scalars, but the addition of a new dimension d introduces a new dependency for the bound, as well as the possibility of sampling t coordinates from each d-dimensional vector. For this extension, we formally defined the *vector view* as the knowledge of the analyzer upon receiving the randomized vectors, and expressed it as a union of overlapping scalar views. Through the use of advanced composition results from Dwork *et al.* [9], we showed that the estimator now has normalized MSE $O_{\varepsilon,\delta}(d^{8/3}tn^{-5/3})$ which can be further improved to $O_{\varepsilon,\delta}(d^{8/3}n^{-5/3})$ by setting $t = 1$.

Our contribution has provided a stepping stone between the summation of the scalar case discussed by Balle *et al.* [2] and the linearization of more sophisticated structures such as matrices and higher-dimensional tensors, both of which are reliant on the functionality of the vector case. As mentioned in Section 2, there is potential for further exploration in the Multi-Message Shuffle Model to gain additional privacy, echoing the follow-up paper of Balle *et al.* [17].

A. Proof of Lemma 4.5

Lemma 4.5. *Condition* (2) *holds.*

Proof. The way in which we split the vector view (i.e., to consider a single uniformly sampled coordinate of each vector-valued message in turn), means that we can apply a proof that is analogous to the scalar-valued case [2]. We work through the key steps needed.

Recall from Section 4.1 that the case where the nth user submits a uniformly random message independent of their input satisfies DP trivially. Otherwise, the nth user submits their true message, and we assume that analyzer removes from $\vec{Y}^{(\alpha_j)}$ any truthful messages associated with the first $n-1$ users. Denote $n_l^{(\alpha_j)}$ to be the count of jth coordinates remaining with a particular value $l \in [k]$. If $\vec{x}_n^{(\alpha_j)} = \theta$ and $\vec{x}_n'^{(\alpha_j)} = \phi$, we obtain the relationship

$$\frac{\Pr[\mathrm{View}_{\mathcal{M}}^{(\alpha_j)}(\vec{D}) = V_{\alpha_j}]}{\Pr[\mathrm{View}_{\mathcal{M}}^{(\alpha_j)}(\vec{D}') = V_{\alpha_j}]} = \frac{n_\theta^{(\alpha_j)}}{n_\phi^{(\alpha_j)}}.$$

We observe that the counts $n_\theta^{(\alpha_j)}$ and $n_\phi^{(\alpha_j)}$ follow the binomial distributions $\mathsf{N}_\theta \sim \mathtt{Bin}\left(s, \frac{\gamma}{k}\right) + 1$ and $\mathsf{N}_\phi \sim \mathtt{Bin}\left(s, \frac{\gamma}{k}\right)$ respectively, where s denotes the number of times that the coordinate j is sampled. In expectation, $s = (n-1)t/d$, and below we will show that it is close to its expectation:

$$\Pr_{\mathsf{V}_{\alpha_j} \sim \mathrm{View}_{\mathcal{M}}^{(\alpha_j)}(\vec{D})}\left[\frac{\Pr[\mathrm{View}_{\mathcal{M}}^{(\alpha_j)}(\vec{D}) = \mathsf{V}_{\alpha_j}]}{\Pr[\mathrm{View}_{\mathcal{M}}^{(\alpha_j)}(\vec{D}') = \mathsf{V}_{\alpha_j}]} \geq e^{\varepsilon'}\right]$$
$$= \Pr\left[\frac{\mathsf{N}_\theta}{\mathsf{N}_\phi} \geq e^{\varepsilon'}\right].$$

We define $c := \mathsf{E}[\mathsf{N}_\phi] = \frac{\gamma}{k} \cdot s$ and split this into the union of two events, $N_\theta \geq ce^{\varepsilon'/2}$ and $N_\phi \leq ce^{-\varepsilon'/2}$. Applying a Chernoff bound gives:

$$\Pr\left[\frac{\mathsf{N}_\theta}{\mathsf{N}_\phi} \geq e^{\varepsilon'}\right] \leq \exp\left(-\frac{c}{3}\left(e^{\varepsilon'/2} - 1 - \frac{1}{c}\right)^2\right)$$
$$+ \exp\left(-\frac{c}{2}\left(1 - e^{-\varepsilon'/2}\right)^2\right).$$

We will choose $c \geq \frac{14}{\varepsilon'^2}\log(2t/\delta)$ so that we have:

$$\exp(\varepsilon'/2) - 1 - \frac{1}{c} \geq \frac{\varepsilon'}{2} + \frac{\varepsilon'^2}{8} - \frac{\varepsilon'^2}{14\log(2t/\delta)} \geq \frac{\varepsilon'}{2}.$$

Using $\varepsilon' < 1$, we have:

$$(1 - \exp(-\varepsilon'/2)) \geq (1 - \exp(-1/2))\varepsilon' \geq \frac{\varepsilon'}{\sqrt{7}}.$$

Thus we have:

$$\Pr\left[\frac{\mathsf{N}_\theta}{\mathsf{N}_\phi} \geq e^{\varepsilon'}\right] \leq \exp\left(-\frac{c}{3}(\varepsilon'/2)^2\right) + \exp\left(-\frac{c}{2}(\varepsilon'/\sqrt{7})^2\right)$$
$$\leq 2\exp\left(-\frac{14}{2\varepsilon'^2}\frac{\varepsilon'^2}{7}\log(2t/\delta)\right) \leq \delta/t.$$

We now apply another Chernoff bound to show that $s \leq 2\mathsf{E}[s]$, which can be used to give a bound on γ. The following calculation proves that $\Pr[s \geq 2\mathsf{E}(s)] \leq \exp(-\mathsf{E}(s)/3)$, using $\mathsf{E}(s) = (n-1)t/d$:

$$\Pr[s \geq 2\mathsf{E}(s)] \leq \exp\left(-\frac{n-1}{3}t/d\right) \leq \exp\left(-\frac{n}{3}\right) < \delta/3t,$$

for all reasonable values of δ.

Substituting these bounds on s and c into $\gamma s/k = c$ along with $\varepsilon' = \frac{\varepsilon}{2\sqrt{2t\log(1/\delta)}}$ gives:

$$\gamma \geq \frac{112kt\log(1/\delta)\log(2t/\delta)}{s\varepsilon^2} \geq \frac{56dk\log(1/\delta)\log(2t/\delta)}{(n-1)\varepsilon^2}.$$

□

References

[1] C. Dwork, Differential privacy, in: Proceedings of the 33rd International Colloquium on Automata, Languages and Programming (ICALP), Springer, Cham, 2006, pp. 1–12.

[2] B. Balle, J. Bell, A. Gascón, K. Nissim, The privacy blanket of the shuffle model, in: Annual International Cryptology Conference, Springer, Cham, 2019, pp. 638–667.

[3] B. McMahan, E. Moore, D. Ramage, S. Hampson, B. A. y Arcas, Communication-efficient learning of deep networks from decentralized data, in: Artificial Intelligence and Statistics Conference, PMLR, New York City, 2017, pp. 1273–1282.

[4] M. Abadi, A. Chu, I. Goodfellow, Deep learning with differential privacy, in: Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security, ACM, New York City, 2016, pp. 308–318.

[5] K. Bonawitz, V. Ivanov, B. Kreuter, A. Marcedone, B. McMahan, S. Patel, D. Ramage, A. Segal, K. Seth, Practical secure aggregation for privacy-preserving machine learning, in: Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security, ACM, New York City, 2017, pp. 1175–1191.

[6] L. Sweeney, k-anonymity: A model for protecting privacy, International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems 10 (2002) 557–570.

[7] A. Machanavajjhala, D. Kifer, J. Gehrke, M. Venkitasubramaniam, l-diversity: Privacy beyond k-anonymity, in: ACM Transactions on Knowledge Discovery from Data (TKDD), ACM, New York City, 2007, pp. 3–es.

[8] N. Li, T. Li, S. Venkatasubramanian, t-closeness: Privacy beyond k-anonymity and l-diversity, in: 2007 IEEE 23rd International Conference on Data Engineering, IEEE, New York City, 2007, pp. 106–115.

[9] C. Dwork, A. Roth, The algorithmic foundations of differential privacy, Foundations and Trends in Theoretical Computer Science 9 (2014) 211–407.

[10] Ú. Erlingsson, V. Pihur, A. Korolova, RAPPOR: Randomized aggregatable privacy-preserving ordinal response, in: Proceedings of the 2014 ACM SIGSAC Conference on Computer and Communications Security, ACM, New York City, 2014, pp. 1054–1067.

[11] A. D. P. Team, Learning with privacy at scale, 2017.

[12] B. Ding, J. Kulkarni, S. Yekhanin, Collecting telemetry data privately, in: Advances in Neural Information Processing Systems, ACM, New York City, 2017, pp. 3571–3580.

[13] A. Bittau, Ú. Erlingsson, P. Maniatis, I. Mironov, A. Raghunathan, D. Lie, M. Rudominer, U. Kode, J. Tinnes, B. Seefeld, PROCHLO: Strong privacy for analytics in the crowd, in: Proceedings of the 26th Symposium on Operating Systems Principles, ACM, New York City, 2017, pp. 441–459.

[14] A. Cheu, A. Smith, J. Ullman, D. Zeber, M. Zhilyaev, Distributed differential privacy via shuffling, in: Annual International Conference on the Theory and Applications of Cryptographic Techniques, Springer, Cham, 2019, pp. 375–403.

[15] B. Balle, J. Bell, A. Gascón, K. Nissim, Improved summation from shuffling, arXiv preprint arXiv:1909.11225, 2019.

[16] Y. Ishai, E. Kushilevitz, R. Ostrovsky, A. Sahai, Cryptography from anonymity, in: 47th Annual IEEE Symposium on Foundations of Computer Science, IEEE, New York City, 2006, pp. 239–248.

[17] B. Balle, J. Bell, A. Gascón, K. Nissim, Private summation in the multi-message shuffle model, arXiv preprint arXiv:2002.00817, 2020.

[18] B. Ghazi, N. Golowich, R. Kumar, R. Pagh, A. Velingker, On the power of multiple anonymous messages, in: International Association for Cryptologic Research, ePrint Archive, Santa Barbara, 2019, p. 1382.

[19] B. Ghazi, P. Manurangsi, R. Pagh, A. Velingker, Private aggregation from fewer anonymous messages, in: Advances in Cryptology—EUROCRYPT 2020, Springer, Cham, 2020, pp. 798–827.

[20] P. Kairouz, S. Oh, P. Viswanath, Extremal mechanisms for local differential privacy, The Journal of Machine Learning Research 17 (2016) 492–542.

[21] P. Kairouz, K. Bonawitz, D. Ramage, Discrete distribution estimation under local privacy, in: Proceedings of the 33rd International Conference on Machine Learning, volume 48, ACM, New York City, 2016, pp. 2436–2444.

[22] A. Bhowmick, J. Duchi, J. Freudiger, G. Kapoor, R. Rogers, Protection against reconstruction and its applications in private federated learning, arXiv preprint arXiv:1812.00984, 2018.