Per-User Histograms In the Shuffle Model

1 Introduction

Estimating a histogram is a fundamental problem with applications in many domains including data analytics, machine learning, and recommender systems. In its most common version, the problem concerns with identifying a *single* histogram representing counts over a domain of items across *all* users. For instance, the counts of products bought, movies watched, or websites browsed by all the users. Importantly, in this query each user either interacts with an item or not, thus having a binary vector of size equal to the size of the item domain. Hence, the typical histogram estimation is an aggregate query that is useful to study common patterns across a user population. In contrast, we are interested in the problem of *per-user histogram* i.e., we seek to estimate a separate histogram for *every* individual user.

Per-user histograms are useful for providing personalized services to the users since the distribution of counts across different item classes for each user can represent their personal preferences and interests. An analyst can perform several queries on top of the histograms, such as computing the top-k heavy hitters, estimation of the parameters of the distribution. For instance, Google has recently released a new framework called Topics top (2023; 2022) for collecting the most popular topics, pertaining to the visited websites, from users' browsing history. These topics are revealed to the content providers, such as the advertisers, to show personalized content to the users when they surf the internet. Here, the topics are item classes and each website is an item.

However, the histograms might encode personal information about the users and are hence, sensitive. Therefore, our aim is to support this computation while providing rigorous privacy guarantees under the differential privacy (DP) framework. Specifically, we focus on the shuffle DP model which is an extension of the local DP model where a trusted shuffler uniformly permutes the noisy responses from the users before releasing them. The shuffling of the responses results in privacy amplification Erlingsson et al. (2019); Cheu et al. (2019b); Balle et al. (2019); Cheu et al. (2019a); Feldman et al. (2022). In other words, this improves the utility.

In this paper, we ask the question: *“Can the shuffle model of DP result in privacy amplification for the per-user histogram query?"* We investigate the above question and provide a privacy amplification result and utility analysis. To the best of our knowledge, this is the first instance of analyzing the implications of shuffling in the context of a non-aggregate query.

2 Preliminary

In this section, we present the necessary background.

Notations. $[n]$ denotes the set $\{1, \cdots, n\}$. Let $H \in \mathbb{N}^d$ represent a histogram of counts over a domain of size d, $H[i] \in \mathbb{N}$ denote the count of the i-th bin and $s_H = \sum_{i=1}^{d} H[i]$ denote the total count of the histogram. Additionally, $\bar{H}$ represents the histogram after normalization, i.e., $\bar{H}[i] = \frac{H[i]}{s_H}, \forall i \in [n]$. A permutation of a set S is a bijection $S \mapsto S$. The set of permutations of $[n], n \in \mathbb{N}$ forms a symmetric group S_n. We use $\{\cdot\}$ to represent a set whereas $\langle \cdot \rangle$ represents an ordered sequence. As a shorthand, we use $\sigma(x)$ to denote applying permutation $\sigma \in \mathrm{S}_n$ to a data sequence $x = \langle x_1, \cdots, x_n \rangle$ of length n. Additionally, $\sigma(i), i \in [n], \sigma \in \mathrm{S}_n$ denotes the value at index i in σ and σ^{-1} denotes its inverse.

Definition 1 (c-Neighboring Histograms)**.** *Two histograms $H, H' \in \mathbb{N}^d \times \mathbb{N}^d$ are defined to be c-neighboring histograms where $c \in [0, 1]$ if*

1. *both the histograms have the same total count, i.e., $s_H = s_{H'}$, and*
2. *the histograms differ from each other in by at most $\lfloor c \cdot s_H \rfloor$ counts, i.e., $|H - H'|_1 \leq \lfloor c \cdot s_H \rfloor$*

For example from the above definition, for a given histogram H, any histogram H' with the same total count that differs from H by at most 5% of the total count is defined to be its 0.05-neighbor.

Definition 2 ((ϵ, c)-Histogram Local Differential Privacy (hLDP))**.** *A randomized mechanism $\mathcal{A} : [0, 1]^d \mapsto \mathcal{Y}$ satisfies (ϵ, c)-LDP if for any two c-neighboring histograms $(H, H') \in \mathbb{N}^d \times \mathbb{N}^d$ we have*

$$\Pr[\mathcal{A}(H) = y] \leq e^{\epsilon} \Pr[\mathcal{A}(H') = y] \tag{1}$$

(ϵ, c)-hLDP thus implies that an adversary cannot distinguish between c-neighboring histograms. Note that we define c to be a fraction of the histogram's total count rather than a concrete integral value for a more uniform privacy semantics. Consider two histograms H_1 and H_2 with total count 5 and 100, respectively. Now consider two “neighboring" histograms H_1' and H_2' such that $s_{H_1} = s_{H_1'}, |H_1 - H_1'| = 5$ and $s_{H_1} = s_{H_1'}, |H_1 - H_1'| = 5$, respectively. Clearly, H_1 and H_1' could be radically different from each other whereas H_2 and H_2' would be quite similarly distributed. This introduces a disparity in the privacy semantics based on the total count of the histograms. On the other hand, a difference of 5% for both the histograms implies a more equitable privacy semantics.

Next, we define a central DP version of the above privacy as follows.

Definition 3 ((ϵ, δ, c) - Histogram Differential Privacy (hDP))**.** *Let $\mathcal{H} = (H_1, \cdots, H_n)$ and $\mathcal{H}' = (H_1, \cdots, H_{i-1}, H_i', H_{i+1} \cdots, H_n)$ be two sets of n such that (H_i, H_i') are c-neighboring. A randomized mechanism*

$\mathcal{A} : (\mathbb{N}^d \times \cdots \times \mathbb{N}^d) \mapsto \mathcal{O}$ *is* (ϵ, δ, c)*-hDP, if for any* $o \in \mathcal{O}$ *we have:*

$$\Pr[\mathcal{A}(\mathcal{H}) = o] \leq e^{\epsilon}\Pr[\mathcal{A}(\mathcal{H}') = o] + \delta \tag{2}$$

Compared to the standard central DPcase, here we are dealing with a "dataset" of histograms – each user contributes an individual histogram (which is analogous to a "record"). The adversary now cannot distinguish if one of the "records" (histogram) is replaced by a neighboring one (c-neighboring histogram), thereby limiting the leakage about about an individual user contribution.

3 Proposed Scheme

Problem Setting. We consider a setting with n users $\mathcal{U}_i, i \in [n]$ and an untrusted server, S. Each user $\mathcal{U}_i$ holds a private histogram of counts $H_i \in \mathbb{N}^d$ defined over a domain of size d. The server S is interested estimating the histogram H_i for each user. Additionally, just like in the shuffle model, we have a trusted shuffler who mediates upon the noisy responses of each of the users before releasing it to the untrusted server.

Algorithm 1

```
    Input: $\epsilon$ - Privacy Parameter; $H_i$ - Histogram of user $\mathcal{U}_i$
    Output: $\hat{\mathcal{H}} - \langle \hat{H}_1, \cdots, \hat{H}_n \rangle$ An ordered sequence of noisy histograms such that $\hat{H}_i$ is treated as the response from user $\mathcal{U}_i$
    Users
 1: For $i \in [n]$
 2:     For $j \in [d]$        ▷ Construct two copies of the histogram with privacy budgets $\epsilon_1$ and $\epsilon - \epsilon_1$
 3:         $W_i[j] = \bar{H}_i[j] + \eta_1, \eta_1 \sim Lap(\frac{2c}{\epsilon_1})$
 4:         $T_i[j] = \bar{H}_i[j] + \eta_2, \eta_2 \sim Lap(\frac{2c}{\epsilon-\epsilon_1})$
 5:         $R_i[j] = \frac{\epsilon_1^2 T_i[j] + (\epsilon-\epsilon_1)^2 W_i[j]}{\epsilon_1^2 + (\epsilon-\epsilon_1)^2}$        ▷ Construct a better estimate using the two copies
 6:     End For
 7:     User $\mathcal{U}_i$ sends the noisy histograms $(W_i, R_i)$ to the shuffler
 8: End For
    Shuffler
 9: Cluster the users into $g$ groups $\mathcal{G} = \{G_1, \cdots, G_g\}$ based on a similarity metric on the noisy histograms $\mathcal{W} = \{W_1, \cdots, W_n\}$
10: For $j \in [g]$
11:     Let $G_j = \langle g_{11}, \cdots, g_{1|G_1|} \rangle$ be the sequence of users in $G_i$ arranged in order of their indices
12:     $\sigma_j \in_R S_{|G_j|}$        ▷ Sample a random permutation for shuffling group $G_i$
13:     For $l \in [|G_j|]$
14:         $\hat{H}_{g_{jl}} = \bar{R}_{g_{j\sigma_j(l)}}$        ▷ $l$-th user of group $G_j$ is assigned the value of $\sigma_j(l)$-th user after shuffling
15:     End For
16: End For
17: Sends $\hat{\mathcal{H}} = \langle \hat{H}_1, \cdots, \hat{H}_n \rangle$ to the untrusted server
```

3.1 Protocol Description

The naive solution is to release the histogram noised via the Laplace mechanism. In this paper, we look at improving the utility by considering an intermediate shuffler. Typically, the shuffler applies a uniform permutation to the noisy responses of the users. This anonymizes the user responses, resulting in privacy amplification by roughly a factor of $\sqrt{n}$. Consequently, the utility is improved over a pure LDP mechanism. When the final query is an *aggregate*, the order in which the individual users are tagged with the noisy responses after the shuffling is inconsequential for utility. However, recall in our setting we require a *separate* measure for each user. Hence, the utility has to be measured w.r.t every user $\mathcal{U}_i$ which depends on the specific (noisy) histogram that is assigned to $\mathcal{U}_i$. Consequently, the standard shuffling paradigm (of uniform random permutation) is not amenable to our setting – a user can be assigned a histogram with a completely different distribution. For instance, assume that the server wants to estimate the top-k values from the individual histograms – then a user can be assigned a histogram with a completely different set of top-k values which completely destroys their utility.

Key Idea. Instead of uniformly shuffling all the users, we partition the users into several groups and then uniformly shuffle them *within* the groups. Specifically, the groups are constructed by clustering the noisy histograms based on a similarity (distance) metric. This ensures that even after shuffling, the users will be assigned a (noisy) histogram that is similar to their original histogram. However, the challenge here is that the construction of the groups itself is now data-dependent thereby leaking privacy. In this paper, we provide an analysis of the resulting privacy amplification and utility (Theorem 2 and Theorem 3).

Protocol Details. Our protocol is outlined in Algorithm 1 and described as follows. Each user $\mathcal{U}_i$ computes two copies of the histograms, W_i and T_i with privacy budgets ϵ_1 and $\epsilon - \epsilon_1$, respectively (Steps 3-4). Next, user $\mathcal{U}_i$ combines them via weighted average[1] to obtain a better estimate R (Step 5) and releases two copies $(W_i, T_i)$to the

[1] The choice of our weights minimizes the variance over all unbiased estimators.

shuffler (Step 7). The shuffler uses the first copies of the histograms $\mathcal{W} = \{W_1, \cdots, W_n\}$ to cluster the users into groups based on a similarity metric. Next, it shuffles the noisy responses of the users of every group $G_j, j \in [g]$ according to a random permutation $\sigma_j \in S_{|G_j|}$. Note that the second copy of the histogram $\mathcal{R} = \{R_1, \cdots, R_n\}$ is used for this step (Steps 10-16). In other words, user $\mathcal{U}_{g_{jl}}$, i.e., the l-th user (when arranged in order of the indices – Step 11) of group G_j is assigned the noisy histogram $R_{g_{j\sigma_j(l)}}$ of the $\sigma_j(l)$-th user $\mathcal{U}_{g_{j\sigma_j(l)}}$ (Step 14). Finally, the shuffler releases the ordered sequence $\hat{\mathcal{H}}$ to the untrusted server.

Illustrative Example. We illustrate our protocol with the following example. Consider a set of 10 users who are partitioned into three groups $\mathcal{G} = \{G_1 = \langle \mathcal{U}_2, \mathcal{U}_5, \mathcal{U}_7 \rangle, G_2 = \langle \mathcal{U}_1, \mathcal{U}_3, \mathcal{U}_6, \mathcal{U}_9 \rangle, G_3 = \langle \mathcal{U}_4, \mathcal{U}_8, \mathcal{U}_{10} \rangle\}$ based on the first copies of the histograms $\mathcal{W}$. Let the three random permutations picked by the shuffler be $\sigma_1 = (3, 1, 2), \sigma_2 = (3, 4, 2, 1)$ and $\sigma_3 = (2, 1, 3)$. For group G_1, it means that the 1st user $\mathcal{U}_2$ is going to be assigned the value of the 3rd (since $\sigma_1(1) = 3$) user $\mathcal{U}_7$, i.e., $\hat{H}_2 = \bar{R}_7$. Following suit for all other users, we obtain the final ordered sequence $\hat{\mathcal{H}} = \langle \bar{R}_6, \bar{R}_7, \bar{R}_9, \bar{R}_8, \bar{R}_2, \bar{R}_3, \bar{R}_5, \bar{R}_4, \bar{R}_1, \bar{R}_{10} \rangle$

Design Choices. Here we discuss our rationale behind our design choices in Algorithm 1.

Two copies of histogram. It is *necessary* to use two separate copies of the data, one for the (data-dependent) clustering by the shuffler and the other for the actual output release to the server (after shuffling) for obtaining privacy amplification in our setting. Intuitively, the reason is that the knowledge of the groups itself leaks privacy. We formalize this as follows. Let us consider a generic protocol $\mathcal{A} : \{[0, 1]^d\}^n : \mathcal{Y}^n$ (detailed in Algorithm 2 in Appendix) that uses a single copy of the (noisy) data for both the clustering and the final output release as follows. First each user $\mathcal{U}_i$ releases a single noisy response $Z_i = \mathcal{R}_i(H_i)$ where randomizer $\mathcal{R}_i : [0, 1]^d \mapsto \mathcal{Y}$ satisfies (ϵ, c)-hLDP. Next, the shuffler partitions the users into groups $\mathcal{G} = \{G_1, \cdots, G_g\}$ where $\mathcal{G}$ is a non-trivial function of the noisy responses $\mathcal{G} = f(Z_1, \cdots, Z_n)$ and releases the shuffled responses $\mathcal{Z}$ such that the responses are uniformly randomly shuffled within each group.

Theorem 1. *There exists a function $f(\cdot)$ such that $\mathcal{A}' : \{[0, 1]^d\}^n \mapsto \mathcal{Y}^n$ cannot satisfy (ϵ', δ, c)-hDP for any $\epsilon' < \epsilon$ for all $\delta \geq 0$.*

Choice of clustering algorithm. The goal of the clustering algorithm is to cluster similar histograms together. As such, any standard clustering algorithm can be used. For instance, DBSCAN Ester et al. (1996) with distance metric, such as Jensen-Shannon divergence (after re-normalizing the noisy histograms), could be good choice. In case the original data distribution was well separated to begin with, k-means clustering Hartigan & Wong (1979) could be used as well.

Privacy budget allocation. In case the user has a prior over the data distribution, they can optimize the privacy budget allocation for ϵ_1. We provide a formal analysis in the following section (Theorem 4).

3.2 Privacy and Utility Analysis

We assume that the groups $\mathcal{G}$ are known to the server. Although this is not explicitly evident from Algorithm 1, we assume a worst-case server with this extra information. This is because the server could post-process the shuffled histograms and cluster them again to get an estimate of the original groups (this would violate privacy amplification, see Appendix B for more details). Next, we present a privacy amplification by shuffling result. An important point to note is that the resulting privacy amplification is *not uniform* across all the users. In other words, the amplification factor depends on the individual users.

Theorem 2 (Per-User Privacy Amplification Theorem)**.** *Let $G_{\mathcal{U}_i} = \{g_1, \ldots, g_{|G_{\mathcal{U}_i}|}\}$ be the group containing user $\mathcal{U}_i$, i.e., $\mathcal{U}_i \in G_{\mathcal{U}_i}$. For any subset $Q \subseteq G_{\mathcal{U}_i}$, define $r(Q) = \max_{u,v \in [|G_{\mathcal{U}_i}|]} \|H_u - H_v\|_1$. Algorithm 1 satisfies (ϵ', δ, c)-hDP for user $\mathcal{U}_i$ where*

$$\epsilon' = \begin{cases} \mathcal{O}\Big(\epsilon_1 + (\epsilon - \epsilon_1) \min\Big\{1, \min_{Q \subseteq \mathcal{P}(G_{\mathcal{U}_i})} (1 + \frac{r(Q)}{c}) \sqrt{\frac{\log \frac{1}{\delta}}{|Q|}}\Big\}\Big), & \textit{if } (\epsilon - \epsilon_1) \leq 1 \\ \mathcal{O}\Big(\epsilon_1 + \min\Big\{\epsilon - \epsilon_1, \min_{Q \subseteq \mathcal{P}(G_{\mathcal{U}_i})} \sqrt{e^{(\epsilon - \epsilon_1)(1 + \frac{r(Q)}{c})} \frac{\log \frac{1}{\delta}}{|Q|}}\Big\}\Big), & \textit{if } (\epsilon - \epsilon_1) > 1 \end{cases} \tag{3}$$

where $\epsilon - \epsilon_1 \leq \log\left(\frac{|G_{\mathcal{U}_i}|}{16 \log(\frac{2}{\delta})}\right)$ and $\mathcal{P}(G_{\mathcal{U}_i})$ is the power set of $G_{\mathcal{U}_i}$.

From the above theorem, we observe that we get privacy amplification only for $(\epsilon - \epsilon_1)$ portion of the privacy budget – ϵ_1 is used constructing the groups (Step 9) and results in no amplification (corollary of Theorem 1, see Appendix B and C for details). W.l.o.g. in the above theorem, let us consider $Q = G_{\mathcal{U}_i}$ for the simplicity of exposition. We observe that the amplification depends on (1) the number of members in the group, $|G_{\mathcal{U}_i}|$, and (2) $r(G_{\mathcal{U}_i})$, the maximum ℓ_1 distance between any pair of histograms belonging to $G_{\mathcal{U}_i}$ – "diameter" of the cluster given by $G_{\mathcal{U}_i}$. This implies that users who are clustered in a larger group (larger $|G_{\mathcal{U}_i}|$) and are positioned near the center of cluster (smaller $r(G_{\mathcal{U}_i})$) would enjoy a higher amplification factor. The dependence on the first term is intuitive – larger the crowd to hide among, better is the privacy – and is along the lines of prior work on amplification by shuffling. The intuition of the second condition is as follows – users who are closer to the center of a cluster will get assigned to the same cluster even after randomization with high probability, while users closer to the boundary might get assigned to different clusters due to the introduced noise. Thus, the second condition is tied to the *stability* of the clustering algorithm.

Next, we present our utility theorem.

Theorem 3 (Per-User Utility Theorem)**.** *Let $G_{\mathcal{U}_i} = \{g_1, \ldots, g_{|G_{\mathcal{U}_i}|}\}$ be the group containing user $\mathcal{U}_i$, i.e., $\mathcal{U}_i \in G_{\mathcal{U}_i}$. The expected utility of user $\mathcal{U}_i \in G$ is given as follows:*

$$\mathbb{E}\big[\|\hat{H}_{\mathcal{U}_i} - \bar{H}_{\mathcal{U}_i}\|_1\big] \leq \frac{1}{|G_{\mathcal{U}_i}|} \sum_{j=1}^{|G_{\mathcal{U}_i}|} \|H_{\mathcal{U}_i} - H_{\mathcal{U}_{g_j}}\|_1 + \frac{2dc \log \frac{n}{\delta}}{\sqrt{\epsilon_1^2 + (\epsilon - \epsilon_1)^2}} \tag{4}$$

The first term accounts for the error obtained from replacing the value of $\mathcal{U}_i$ with that of a random user from the group $G_{\mathcal{U}_i}$ and this depends on how well the clustering algorithm works. The second term is the error introduced due to the Laplace mechanism (for noising the histograms). Intuitively, our algorithm is suited for input data distributions where (1) the clusters are well-separated since this implies that we would be able to form the groups accurately even with a small privacy budget ϵ_1 – this is a desirable condition because we get amplification only for the $(\epsilon - \epsilon_1)$ portion of the budget, and (2) the individual clusters are dense, i.e., have small diameters – this means that any of the histograms corresponding to the users in $G_{\mathcal{U}_i}$ is a good estimate for $\mathcal{U}_i$ which reduces the first term in Eq. 4.

In light of the above discussion, we derive the following optimal privacy budget allocation for ϵ_1 for a user assuming some prior on the data distribution.

Theorem 4 (Optimum Budget Allocation)**.** *For $\mathcal{U}_i$, suppose there are at least n_i other users $\{\mathcal{U}_j\}$ such that*

- *$\|H_{\mathcal{U}_i} - H_{\mathcal{U}_j}\|_1 \leq cr_i$, i.e., r_i is the "diameter" of $\mathcal{U}_i$'s cluster (group),*
- *all other users satisfy $\|H_{\mathcal{U}_i} - H_{\mathcal{U}_j}\|_1 > cR_i$ for some $r_i \leq 2R_i$, i.e., R_i the minimum separation from the other clusters.*

Then, using $\epsilon_1 = \frac{4d \log \frac{n}{\delta}}{R_i}$, $\mathcal{U}_i$ attains (ϵ', δ, c)-hDP privacy guarantee in expectation where

$$\mathbb{E}[\epsilon'] = \begin{cases} \mathcal{O}\big(\epsilon_1 + (\epsilon - \epsilon_1)(1 + r_i)\sqrt{\frac{\log \frac{1}{\delta}}{n_i}}\big), & \text{if } (\epsilon - \epsilon_1) \leq 1 \\ \mathcal{O}\big(\epsilon_1 + \sqrt{e^{(\epsilon - \epsilon_1)(1 + r_i)} \frac{\log \frac{1}{\delta}}{n_i}}\big), & \text{if } (\epsilon - \epsilon_1) > 1 \end{cases} \tag{5}$$

The resulting expected utility is

$$\mathbb{E}[\|\hat{H}_{\mathcal{U}_i} - \bar{H}_{\mathcal{U}_i}\|_1] \leq 2r_i c + \frac{2dc \log \frac{n}{\delta}}{\sqrt{\epsilon_1^2 + (\epsilon - \epsilon_1)^2}} \tag{6}$$

Additionally, for $\epsilon - \epsilon_1 \leq 1$, the chosen value of ϵ_1 is optimal, i.e., it maximizes both the expected privacy guarantee and utility.

Assuming a prior on the diameter of their cluster (r_i) and the well-separatedness $(R_i; r_i \leq 2R_i)$ of all the clusters, by setting $\epsilon_1 = \frac{4d \log \frac{n}{\delta}}{R_i}$ user $\mathcal{U}_i$ can maximize both the expected privacy guarantee (Eq. 5) and utility (Eq. 6).

4 Discussion and Future Work

Revealing amplified privacy parameter. An interesting observation here is that first the computation of the amplified privacy budget is data-dependent and hence, revealing its value leaks privacy. Additionally, no party (users, shuffler, server) in the setup has access to private data of all the users and hence, none of them can compute the amplified budget. We argue this might be acceptable in practice as follows. As long as the users were satisfied with releasing their data with the initial privacy budget ϵ, it might be okay for the users to not know the actual value of the privacy parameter (after amplification) since after the shuffling it can only get better. Nevertheless, an interesting direction to explore is releasing an estimate of the amplified privacy budget by the shuffler (computed using the noisy responses from the users).

Per-user amplification. As observed in Theorem 2, the privacy amplification is user-dependent. Additionally, lower the diameter of the clusters $(r(Q))$ better is the amplification. One way to improve the amplification is by trading-off the number of users enjoying amplification for the actual amplification factor. Specifically, the shuffler can exclusively feed the diameter of the clusters as a parameter in the clustering algorithm – only clusters with smaller diameters are identified and rest of the points (users) are treated as outliers (analogously, these users are in singleton groups with just themselves). Hence, it is possible in our scheme that some of the users, who have very different histograms, do not enjoy any privacy amplification. This makes intuitive sense as it is harder to provide privacy for "outliers' without adversely affecting utility.

Differentially private clustering Currently, we perform the clustering as a post-processing operation on the first copy of the noisy histograms $\mathcal{W} = \{W_1, \cdots, W_n\}$. However, another alternative is to consume the budget ϵ_1 for an algorithm directly tailored for clustering Chang et al. (2021)– this could potentially drive down the first term in Eq. 4.

Implementation of the shuffler. Note that the shuffler in our setup has to perform additional work (clustering) as compared to that of the typical shuffle model (just uniform random shuffle across all users). One possible solution is to implement it via trusted execution environments (TEE) as suggested by prior work, such as just Google's Prochlo Bittau et al. (2017) and Meehan et al. (2022).

Prior work on group shuffle. Prior work Meehan et al. (2022); Abouei & Canonne (2021) has also explored group shuffle. Our work differs from them in the following ways. First, Abouei & Canonne (2021) addresses the problem of releasing aggregate queries but with multiple shufflers – the users are partitioned into several groups (construction of the groups is random and not data-dependent like ours), each group having its own local shuffler. Second, Meehan et al. (2022) considers an inferential privacy framework where the groups are formed based on some public auxiliary information.

References

Get to know the new topics api for privacy sandbox. https://blog.google/products/chrome/get-know-new-topics-api-privacy-sandbox/, 2022.

Topics api overview. https://developer.chrome.com/docs/privacy-sandbox/topics/overview/, 2023.

Amir Mohammad Abouei and Clement Louis Canonne. Differential privacy via group shuffling. In *NeurIPS 2021 Workshop Privacy in Machine Learning*, 2021. URL https://openreview.net/forum?id=j7WJc7-VKZM.

Shimaa Ahmed, Amrita Roy Chowdhury, Kassem Fawaz, and Parmesh Ramanathan. Preech: A system for Privacy-Preserving speech transcription. In *29th USENIX Security Symposium (USENIX Security 20)*, pp. 2703–2720. USENIX Association, August 2020. ISBN 978-1-939133-17-5. URL https://www.usenix.org/conference/usenixsecurity20/presentation/ahmed-shimaa.

Mitali Bafna and Jonathan Ullman. The price of selection in differential privacy. In *Conference on Learning Theory*, 2017.

Borja Balle, James Bell, Adrià Gascón, and Kobbi Nissim. The privacy blanket of the shuffle model. In Alexandra Boldyreva and Daniele Micciancio (eds.), *Advances in Cryptology – CRYPTO 2019*, pp. 638–667, Cham, 2019. Springer International Publishing. ISBN 978-3-030-26951-7.

Andrea Bittau, Úlfar Erlingsson, Petros Maniatis, Ilya Mironov, Ananth Raghunathan, David Lie, Mitch Rudominer, Ushasree Kode, Julien Tinnes, and Bernhard Seefeld. Prochlo: Strong privacy for analytics in the crowd. In *Proceedings of the 26th Symposium on Operating Systems Principles*, SOSP '17, pp. 441–459, New York, NY, USA, 2017. ACM. ISBN 978-1-4503-5085-3. doi: 10.1145/3132747.3132769. URL http://doi.acm.org/10.1145/3132747.3132769.

Alisa Chang, Badih Ghazi, Ravi Kumar, and Pasin Manurangsi. Locally private k-means in one round, 2021.

Albert Cheu. Differential privacy in the shuffle model: A survey of separations, 2022.

Albert Cheu, Adam Smith, Jonathan Ullman, David Zeber, and Maxim Zhilyaev. Distributed differential privacy via shuffling. In Yuval Ishai and Vincent Rijmen (eds.), *Advances in Cryptology – EUROCRYPT 2019*, pp. 375–403, Cham, 2019a. Springer International Publishing. ISBN 978-3-030-17653-2.

Albert Cheu, Adam Smith, Jonathan Ullman, David Zeber, and Maxim Zhilyaev. Distributed differential privacy via shuffling. In Yuval Ishai and Vincent Rijmen (eds.), *Advances in Cryptology – EUROCRYPT 2019*, pp. 375–403, Cham, 2019b. Springer International Publishing. ISBN 978-3-030-17653-2.

Zeyu Ding, Yuxin Wang, Danfeng Zhang, and Daniel Kifer. Free gap information from the differentially private sparse vector and noisy max mechanisms. *arXiv preprint arXiv:1904.12773*, 2019.

John C. Duchi, Michael I. Jordan, and Martin J. Wainwright. Local privacy and statistical minimax rates. In *2013 IEEE 54th Annual Symposium on Foundations of Computer Science*, pp. 429–438, 2013. doi: 10.1109/FOCS.2013.53.

David Durfee and Ryan M Rogers. Practical differentially private top-k selection with pay-what-you-get composition. *Advances in Neural Information Processing Systems*, 2019.

Cynthia Dwork, Aaron Roth, et al. The algorithmic foundations of differential privacy. *Foundations and Trends® in Theoretical Computer Science*, 2014.

Cynthia Dwork, Weijie J Su, and Li Zhang. Differentially private false discovery rate control. *arXiv preprint arXiv:1807.04209*, 2018.

Úlfar Erlingsson, Vasyl Pihur, and Aleksandra Korolova. Rappor: Randomized aggregatable privacy-preserving ordinal response. In *CCS*, 2014.

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*, SODA '19, pp. 2468–2479, USA, 2019. Society for Industrial and Applied Mathematics.

Martin Ester, Hans-Peter Kriegel, Jörg Sander, and Xiaowei Xu. A density-based algorithm for discovering clusters in large spatial databases with noise. In *Proceedings of the Second International Conference on Knowledge Discovery and Data Mining*, KDD'96, pp. 226–231. AAAI Press, 1996.

Vitaly Feldman, Audra McMillan, and Kunal Talwar. Hiding among the clones: A simple and nearly optimal analysis of privacy amplification by shuffling. In *2021 IEEE 62nd Annual Symposium on Foundations of Computer Science (FOCS)*, pp. 954–964, 2022. doi: 10.1109/FOCS52979.2021.00096.

Jennifer Gillenwater, Matthew Joseph, Andres Munoz, and Monica Ribero Diaz. A joint exponential mechanism for differentially private top-k. In *International Conference on Machine Learning*. PMLR, 2022.

John A Hartigan and Manchek A Wong. Algorithm as 136: A k-means clustering algorithm. *Journal of the royal statistical society. series c (applied statistics)*, 28(1):100–108, 1979.

Ryan McKenna and Daniel R Sheldon. Permute-and-flip: A new mechanism for differentially private selection. *Advances in Neural Information Processing Systems*, 2020.

Casey Meehan, Amrita Roy Chowdhury, Kamalika Chaudhuri, and Somesh Jha. Privacy implications of shuffling. In *International Conference on Learning Representations*, 2022. URL `https://openreview.net/forum?id=5i2f-aR6B8H`.

Gang Qiao, Weijie Su, and Li Zhang. Oneshot differentially private top-k selection. In *International Conference on Machine Learning*. PMLR, 2021.

Zhan Qin, Yin Yang, Ting Yu, Issa Khalil, Xiaokui Xiao, and Kui Ren. Heavy hitter estimation over set-valued data with local differential privacy. In *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*, 2016.

Thomas Steinke and Jonathan Ullman. Between pure and approximate differential privacy. *arXiv preprint arXiv:1501.06095*, 2015.

Thomas Steinke and Jonathan Ullman. Tight lower bounds for differentially private selection. In *2017 IEEE 58th Annual Symposium on Foundations of Computer Science (FOCS)*, 2017.

Tianhao Wang, Jeremiah Blocki, Ninghui Li, and Somesh Jha. Locally differentially private protocols for frequency estimation. In *26th USENIX Security Symposium (USENIX Security 17)*, pp. 729–745, Vancouver, BC, August 2017. USENIX Association. ISBN 978-1-931971-40-9. URL `https://www.usenix.org/conference/usenixsecurity17/technical-sessions/presentation/wang-tianhao`.

A Related Work

Estimating aggregate histograms privately is a fundamental problem that has been studied extensively in the context of differential privacy Duchi et al. (2013); Erlingsson et al. (2014); Wang et al. (2017). A very common related task is differentially-private heavy-hitters Qiao et al. (2021); Gillenwater et al. (2022); Durfee & Rogers (2019); McKenna & Sheldon (2020); Ding et al. (2019); Steinke & Ullman (2015); Bafna & Ullman (2017); Steinke & Ullman (2017); Dwork et al. (2018); Qin et al. (2016). In this problem, each user has a binary vector that represents their item interactions from an item domain. The target is to find the most popular items across all users. Different mechanisms have been proposed in the central DP and the local DP settings Dwork et al. (2014).

In the central DP setting, iterative peeling mechanisms have been proposed where either the report noisy-max mechanism or exponential mechanism is applied iteratively to pick the top-K items (say with most frequency) Dwork et al. (2014; 2018). These mechanisms may be inefficient when the item domain is large. To address this issue, one-shot Laplace and exponential mechanisms have also been proposed Qiao et al. (2021); Durfee & Rogers (2019); McKenna & Sheldon (2020). These mechanisms have been designed for both pure DP ($\epsilon-$DP) and approximate DP ((ϵ, δ)) settings although providing utility analysis for the approximate DP setting can be challenging Qiao et al. (2021). More works have therefore analyzed the lower bounds on utility provided by these mechanisms Bafna & Ullman (2017); Steinke & Ullman (2017).

In the local DP setting, most of the mechanisms designed in the central DP setting cannot be applied as-is since there is no global view of all users' data. In the local DP setting, randomized response mechanism and iteratively improving the randomized-response mechanism are the typical solutions Erlingsson et al. (2014); Qin et al. (2016). To improve the utility of these mechanisms the shuffle model has been proposed. In the shuffle model, an intermediate shuffler is trusted to shuffle the noisy records before aggregating to delink the user from their record. This provides privacy guarantees similar to the central DP setting, therefore, a higher utility. The principled system architecture for shuffling was first proposed by Bittau et al. Bittau et al. (2017). This model was formally studied later in Erlingsson et al. (2019); Cheu et al. (2019b). Erlingsson et al. Erlingsson et al. (2019) showed that for arbitrary ϵ-LDPrandomizers, random shuffling results in privacy amplification. Cheu et al. Cheu et al. (2019b) formally defined the shuffle DPmodel and analyzed the privacy guarantees of the binary randomized response in this model. See Cheu (2022) for a survey on the existing literature on shuffle DP.

The problem that we propose in this work is very different from the aforementioned one. We aim to estimate histograms per user. Conceptually, our motivation stems from creating personalized user profiles rather than computing aggregate statistics across all users. To address the new technical challenges imposed by our problem setup, we design a new mechanism which combines the traditional Laplace and shuffle mechanisms in a novel way—clustering the users based on their noisy responses and shuffling the users within clusters. We believe that our mechanism by itself could be interesting for other differentially private queries in the local DP setting. Our privacy definition of (ϵ, c)-hLDP for histograms is inspired by previous work Ahmed et al. (2020).

B Proof of Theorem 1

Proof. We start the proof by considering a slight modification of the protocol – $\mathcal{A}''$ is identical to the $\mathcal{A}'$ except that now it explicitly releases the groups $\mathcal{G}$ along side the shuffled outputs $\hat{\mathcal{Z}}$.

Lemma 5. *$\mathcal{A}''$ is (ϵ, c)-hLDP.*

Proof. Note that the groups $\mathcal{G}$ are constructed by post-processing $\langle Z_1, \cdots, Z_n\rangle$ *before* shuffling. Hence, this computation is still (ϵ, c)-hLDP which makes the full release $(\mathcal{G}, \hat{\mathcal{Z}})$ is (ϵ, c)-hLDP as well. □

Now, coming back to $\mathcal{A}'$ note that the untrusted server could run the same function $f(\cdot)$ on the shuffled responses $\hat{\mathcal{Z}}$. This could result in the exact same groups $\mathcal{G}$ (for instance - if $f(\cdot)$ is clustering algorithm the server would be able to construct back the exact same groups since only all the labels (user ids) are shuffled within each group). Hence, by lemma 5 this could result in an extra privacy leakage resulting in a (ϵ, c)-hLDP guarantee. This means we cannot have a (ϵ', δ, c)-hDP guarantee where $\epsilon' < \epsilon$ since hLDP is a stronger privacy guarantee than hDP. □

Algorithm 2

```
Input: $\epsilon$ - Privacy Parameter; $H_i$ - Histogram of user $\mathcal{U}_i$
Output: $\hat{\mathcal{H}} - \langle \hat{H}_1, \cdots, \hat{H}_n\rangle$ An ordered sequence of noisy histograms such that $\hat{H}_i$ is treated as the response from user $\mathcal{U}_i$
Users
User $\mathcal{U}_i$ sends the noisy response $\mathcal{R}_i(H_i) = Z_i$ where randomizer $\mathcal{R}_i : [0,1]^d \mapsto \mathcal{Y}$ satisfies $(\epsilon, c)$-hLDP to the shuffler
Shuffler
Cluster the users into $g$ groups $\mathcal{G} = \{G_1, \cdots, G_g\}$ based on some non-trivial function over the noisy responses $f(Z_1, \cdots, Z_n) = \mathcal{G}$
For $j \in [g]$
    Let $G_j = \langle g_{11}, \cdots, g_{1|G_1|}\rangle$ be the sequence of users in $G_i$ arranged in order of their indices
    $\sigma_j \in_R S_{|G_j|}$        ▷ Sample a random permutation for shuffling group $G_i$
    For $l \in [|G_j|]$
        $\hat{Z}_{g_{jl}} = Z_{g_{j\sigma_j(l)}}$        ▷ $l$-th user of group $G_j$ is assigned the value of $\sigma_j(l)$-th user after shuffling
    End For
End For
Sends $\hat{\mathcal{Z}} = \langle \hat{Z}_1, \cdots, \hat{Z}_n\rangle$ to the untrusted server
```

C Proof of Theorem 2

Our proof will use the following privacy amplification by shuffling result of Feldman et al. (2022).

Theorem 6. *Suppose $\mathcal{A}(\cdot)$ satisfies ϵ-local differential privacy where $\epsilon \leq \log\left(\frac{n}{16\log(\frac{2}{\delta})}\right)$. Define $\mathcal{A}(\mathcal{H}) = \langle \mathcal{A}(H_1), \ldots, \mathcal{A}(H_n)\rangle$. Then, for a random permutation σ, the release $\sigma \circ \mathcal{A}(\mathcal{H})$ satisfies (ϵ', δ)-DP, where*

$$\epsilon' = \log\left(1 + \frac{e^\epsilon - 1}{e^\epsilon + 1}\left(\frac{8\sqrt{e^\epsilon \log(4/\delta)}}{\sqrt{n}} + \frac{8e^\epsilon}{q}\right)\right).$$

If $\epsilon \leq 1$, then $\epsilon' = \mathcal{O}(\epsilon\sqrt{\frac{\log\frac{1}{\delta}}{n}})$, $\epsilon' = \mathcal{O}(\sqrt{e^\epsilon \frac{\log\frac{1}{\delta}}{n}})$ otherwise.

Proof. Let $\mathcal{H} = (H_1, \ldots, H_i, \ldots, H_n)$ and $\mathcal{H}' = (H_1, \ldots, H_i', \ldots, H_n)$ where (H_i, H_i') denote two c-neighboring histograms.

We will analyze the privacy guarantee of releasing $\{W_j : j \in [n]\}$ and $\{T_j : j \in [n]\}$, separately, and then obtain a user-dependent privacy guarantee by composition. It is easy to see the first release, $\{W_j : j \in [n]\}$, satisfies $(\epsilon_1, 0, c)$-hDP as it is the Laplace mechanism.

Let $Q \subseteq G_{\mathcal{U}_i}$. To analyze the second release, first let $\mathcal{A}_Q(\mathcal{H}) = \{\mathcal{A}_g(H_g) : g \in Q\}$ denote the outputs $\mathcal{A}_i(H_i)$ indexed on the users in Q. The adversary sees the output $\sigma(\mathcal{A}(\cdot))$, where σ is a permutation on $[n]$. Observe this output has the same distribution as $\sigma(\langle \mu(\mathcal{A}_Q(\cdot)), \mathcal{A}_{[n]\setminus Q}(\cdot)\rangle)$, where μ is a random permutation on Q. Since $\mathcal{H}, \mathcal{H}'$ are equal on indices outside of Q, when given either $\mathcal{H}$ or $\mathcal{H}'$ as an argument, we have that $\sigma(\mathcal{A}(\cdot))$ is a post-processing of $\mu(\mathcal{A}_Q(\cdot))$.

By definition, it holds that $\|H_u - H_v\|_1 \leq r(Q)$ for all $u, v \in Q$, and by the triangle inequality it holds that $\|H_u' - H_v\|_1 \leq r(Q) + c$ and $\|H_u - H_v'\|_1 \leq r(Q) + c$ for all $u, v \in Q$ where H_u' means drawn from $\mathcal{H}'$.

Thus, an $((1 + \frac{r(Q)}{c})\epsilon, 0)$-local differential privacy guarantee holds for every pair $\mathcal{A}(H_u'), \mathcal{A}(H_v)$ and $\mathcal{A}(H_u), \mathcal{A}(H_v')$, when $u, v \in Q$. The result follows directly from using privacy amplification by shuffling Theorem 6. □

D Proof of Theorem 3

Proof. Each R_i is equal to $H_i + a\vec{\eta}_1 + b\vec{\eta}_2$, where $a = \frac{\epsilon_1^2}{\epsilon_1^2 + \epsilon_2^2}$ and $b = 1 - a$. We observe that $\hat{H}_i$ is a uniformly random element from the set

$$\{H_i + a\vec{\eta}_{1,i} + b\vec{\eta}_{2,i} : i \in G_{\mathcal{U}_i}\},$$

where $\vec{\eta}_{1,i}, \vec{\eta}_{2,i}$ are d-dimensional vectors with each element drawn from $Lap(\frac{2c}{\epsilon_1})$, $Lap(\frac{2c}{\epsilon_2})$, respectively. Thus, each element of that $a\vec{\eta}_{1,i} + b\vec{\eta}_{2,i}$ is sub-exponential with variance $4c^2(\frac{a^2}{\epsilon_1^2} + \frac{b^2}{\epsilon_2^2}) = \frac{4c^2}{\epsilon_1^2+\epsilon_2^2}$. Thus, with probability at least $1-\delta$, we have $\|a\vec{\eta}_{1,i} + b\vec{\eta}_{2,i}\|_1 \leq \frac{2cd\log\frac{1}{\delta}}{\sqrt{\epsilon_1^2+\epsilon_2^2}}$. Thus,

$$
\begin{aligned}
\mathbb{E}[\|\hat{H}_i - H_i\|_1] &= \frac{1}{|G_i|}\sum_{j\in G_i}\|R_j - H_i\|_1 \\
&= \frac{1}{|G_i|}\sum_{j\in G_i}\|H_j + a\vec{\eta}_1 + b\vec{\eta}_2 - H_i\|_1 \\
&\leq \frac{1}{|G_i|}\sum_{j\in G_i}\|H_j - H_i\| + \frac{1}{|G_i|}\sum_{j\in G_i}\|a\vec{\eta}_1 + b\vec{\eta}_2\|_1 \\
&\leq \frac{1}{|G_i|}\sum_{j\in G_i}\|H_j - H_i\| + \frac{2cd\log\frac{1}{\delta}}{\sqrt{\epsilon_1^2+\epsilon_2^2}}.
\end{aligned}
$$

□

E Proof of Theorem 4

Proof. To begin, let ϵ_1 be indeterminate, whose value will be decided later. Given the prior and applying Theorem 6, each user will have a (ϵ', δ, c)-hLDP guarantee where

$$
\epsilon' = \epsilon_1 + \begin{cases} \sqrt{\exp((\epsilon-\epsilon_1)(1+r_i))}^{\frac{\log\frac{1}{\delta}}{\sqrt{n_i}}} & \epsilon - \epsilon_1 \geq 1 \\ (\epsilon-\epsilon_1)\sqrt{(1+r_i)}^{\frac{\log\frac{1}{\delta}}{\sqrt{n_i}}} & \epsilon - \epsilon_1 < 1 \end{cases}.
$$

.

Let Q denote the group of n_i users within distance cr_i of user i. Each Laplace noise vector $\vec{\eta}_{1,i}$ added to $\bar{H}_i$ has magnitude $\frac{2cd\log\frac{n}{\delta}}{\epsilon_1}$. Thus, in order to unambiguously identify Q, it is necessary that $\frac{1}{2}cR_i \geq \frac{2cd\log\frac{n}{\delta}}{\epsilon_1}$, meaning $\epsilon_1 \geq \frac{4d\log\frac{n}{\delta}}{R_i}$.

By Theorem 3, user i will have expected utility at most $2r_ic + \frac{2cd\log\frac{n}{\delta}}{\sqrt{\epsilon_1^2+(\epsilon-\epsilon_1)^2}}$. In the case $\epsilon - \epsilon_1 \geq 1$, we plug in $\epsilon = \frac{4d\log\frac{n}{\delta}}{R_i}$ to obtain the desired privacy guarantee. Otherwise, observe that $\epsilon_1 + (\epsilon-\epsilon_1)\sqrt{(1+r_i)}^{\frac{\log\frac{1}{\delta}}{\sqrt{n_i}}}$ is an increasing function of ϵ_1, and thus the optimal utility guarantee occurs when $\epsilon = \frac{4d\log\frac{n}{\delta}}{R_i}$. □