A Shuffling Framework for Local Differential Privacy

Author(s)
Institution(s)

ABSTRACT

LDP deployments are vulnerable to inference attacks as an adversary can link the noisy responses to their identity and subsequently, auxiliary information using the *order* of the data. An alternative model, shuffle DP, prevents this by shuffling the noisy responses uniformly at random. However, this limits the data learnability – only symmetric functions (input order agnostic) can be learned. In this paper, we strike a balance and propose a generalized shuffling framework that interpolates between the two deployment models. We show that systematic shuffling of the noisy responses can thwart specific inference attacks while retaining some meaningful data learnability. To this end, we propose a novel privacy guarantee, d_σ-privacy, that captures the privacy of the order of a data sequence. d_σ-privacy allows tuning the granularity at which the ordinal information is maintained, which formalizes the degree the resistance to inference attacks trading it off with data learnability. Additionally, we propose a novel shuffling mechanism that can achieve d_σ-privacy and demonstrate the practicality of our mechanism via evaluation on real-world datasets.

1 INTRODUCTION

Differential Privacy (DP) and its local variant (LDP) are the most commonly accepted notions of data privacy. LDP has the significant advantage of not requiring a trusted centralized aggregator, and has become a popular model for commercial deployments, such as those of Microsoft [15], Apple [29], and Google [22]. Its formal guarantee asserts that an adversary cannot infer the value of an individual's private input by observing the noisy output. However in practice, a vast amount of *public auxiliary information*, such as address, social media connections, court records, property records [3], income and birth dates [4], is available for every individual. An adversary, with access to such auxiliary information, *can* learn about an individual's private data from several *other* participants' noisy responses. We illustrate this as follows.

Problem. An analyst runs a medical survey in Alice's community to investigate how the prevalence of a highly contagious disease changes from neighborhood to neighborhood. Community members report a binary value indicating whether they have the disease.

Next, consider the following two data reporting strategies.

Strategy 1. Each data owner passes their data through an appropriate randomizer (that flips the input bit with some probability) in their local devices and reports the noisy output to the untrusted data analyst.

Strategy 2. The noisy responses from the local devices of each of the data owners are collected by an intermediary trusted shuffler which dissociates the device IDs (metadata) from the responses and uniformly randomly shuffles them before sending them to the analyst.

Strategy 1 corresponds to the standard LDP deployment model (for example, Apple and Microsoft's deployments). Here *the order of the noisy responses is informative of the identity of the data owners* – the noisy response at index 1 corresponds to the first data owner and so on. Thus, the noisy responses can be directly linked with its associated device ID and subsequently, auxiliary information. For instance, an adversary[1] may know the home addresses of the participants and use this to identify the responses of all the individuals from Alice's household. Being highly infectious, all or most of them will have the same true value (0 or 1). So, the adversary can reliably infer Alice's value by taking a simple majority vote of her and her household's noisy responses. Note that this does not violate the LDP guarantee since the inputs are appropriately randomized when observed in isolation. We call such threats *inference attacks* – recovering an individual's private input using all or a subset of other participants' noisy responses. It is well known that protecting against inference attacks, that rely on underlying data correlations, is beyond the purview of DP [34, 36, 38, 46].

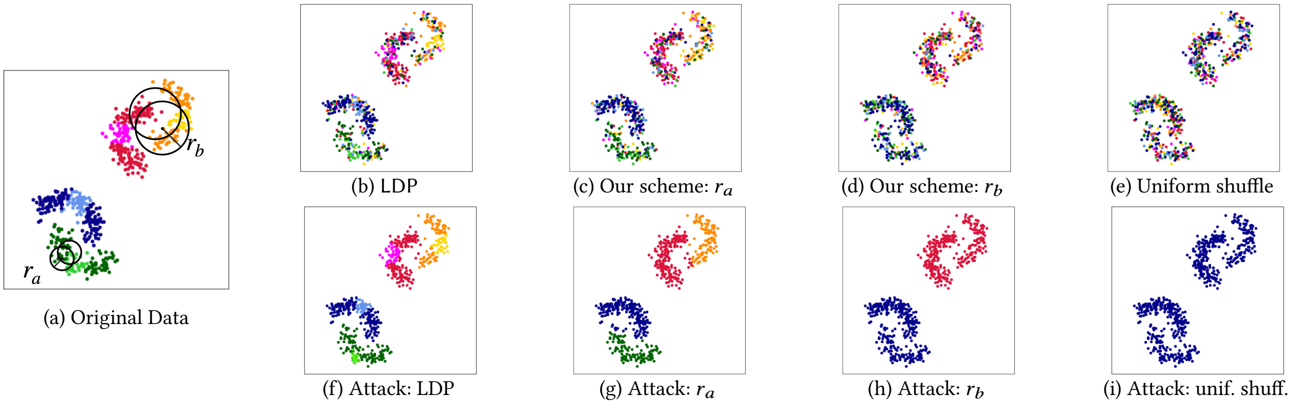
Strategy 2 corresponds to the recently introduced shuffle DP model, such as Google's Prochlo. Here, the noisy responses are completely anonymized – the adversary cannot identify which LDP responses correspond to Alice and her household. Under such a model, only information that is completely order agnostic (i.e., symmetric functions that can be derived from just the *bag* of the values, such as aggregate statistics) can be extracted. Consequently, the analyst also fails to accomplish their original goal as all the underlying data correlation is destroyed.

Thus, we see that the two models of deployment for LDP present a trade-off between vulnerability to inference attacks and scope of data learnability. In fact, as demonstrated by Kifer et. al [35], it is impossible to defend against *all* inference attacks while simultaneously maintaining utility for learning. In the extreme case that the adversary knows *everyone* in Alice's community has the same true value (but not which one), no mechanism can prevent revelation of Alice's datapoint short of destroying all utility of the dataset. This

[1] The analyst and the adversary could be same, we refer to them separately for the ease of understanding.

CCS'21, November 2021, Seoul, South Korea

ACM ISBN 978-x-xxxx-xxxx-x/YY/MM…$15.00
https://doi.org/10.1145/nnnnnnn.nnnnnnn



(a) Original Data

(b) LDP

(c) Our scheme: r_a

(d) Our scheme: r_b

(e) Uniform shuffle

(f) Attack: LDP

(g) Attack: r_a

(h) Attack: r_b

(i) Attack: unif. shuff.

Figure 1: Demonstration of how our proposed scheme thwarts inference attacks at different granularities. Fig. 1a depicts the original sensitive data (such as income bracket) with eight color-coded labels. The position of the points represents public information (such as home address) used to correlate them. There are three levels of granularity: warm vs. cool clusters, blue vs. green and red vs. orange crescents, and light vs. dark within each crescent. Fig. 1b depicts $\epsilon = 2.55$ LDP. Fig. 1c and 1d correspond to our scheme, each with $\alpha = 1$ (privacy parameter, Def. 2.3). The former uses a smaller distance threshold (r_1, used to delineate the granularity of grouping) that mostly shuffles in each crescent. The latter uses a larger distance threshold (r_2) that shuffles within each cluster. Figures in the bottom row demonstrate an inference attack (uses Gaussian process correlation) on all four cases. We see that LDP reveals almost the entire dataset (Fig. 1f) while uniform shuffling prevents all classification (1i). However, the granularity can be controlled with our scheme (Figs. 1g, 1h).

then begs the question: ***Can we formally suppress specific inference attacks targeting each data owner while maintaining some meaningful utility of the private data?*** Referring back to our example, can we thwart attacks inferring Alice's data using specifically her households' responses and still allow the medical analyst to learn its target trends? Can we offer this to every data owner participating?

In this paper, we strike a balance and we propose a generalized shuffle framework for deployment that can interpolate between the two extremes. Specifically, we guarantee each data owner that their data is shuffled together with a carefully chosen group of other data owners. Revisiting our example, consider uniformly shuffling the responses from Alice's household and her immediate neighbors. Now an adversary cannot use her household's responses to predict her value any better than they could with a random sample of responses from this group. In the same way that LDP prevents reconstruction of her datapoint using specifically *her* noisy response, this scheme prevents reconstruction of her datapoint using specifically *her households'* responses. Our scheme can offer this to each data owner, even when their groups are arbitrarily intersecting. We can formally protect each data owner from inference attacks using specifically their household, while still learning how disease prevalence changes across the neighborhoods of Alice's community.

This work offers two key contributions to the machine learning privacy literature:

- **Novel privacy guarantee.** We propose a novel privacy definition, d_σ-privacy, which guarantees each data owner that their data will be shuffled in with a semantically meaningful group.
- **Novel shuffling framework.** We propose a novel mechanism that shuffles the data systematically and achieves d_σ-privacy. This provides us a generalized shuffle framework for deployment that can interpolate between the no shuffling (LDP) and uniform random shuffling (shuffle model). Our experimental results (Sec. 3) demonstrates its efficacy against realistic inference attacks.

1.1 Related Work

The shuffle model of DP [10, 12, 21] differs from our scheme as follows. These works (1) study DP benefits of shuffling where we study the inferential privacy benefits and (2) only study uniformly random shuffling where ours generalizes this to configurable, non-uniform shuffling.

A steady line of work has studied inferential privacy [14, 18, 28, 32, 35, 46]. Our work departs from those in that we focus on *local* inferential privacy and do so via the new angle of shuffling.

Older works such as k-anonymity [44], l-diversity [39], Anatomy [48], and others [13, 16, 45, 47, 49] have studied the privacy risk of non-sensitive auxiliary information, or 'quasi identifiers'. These works (1) focus on the setting of dataset release, where we focus on dataset collection and (2) do not offer each data owner formal inferential guarantees, whereas this work does.

2 DATA PRIVACY AND SHUFFLING

In this section, we present d_σ-privacy and a shuffling mechanism capable of achieving the d_σ-privacy guarantee.

2.1 Problem Setting

In our problem setting (Fig. 2), we have n data owners $\text{DO}_i, i \in [n]$ each with a private input $x_i \in \mathcal{X}$. The data owners first randomize their inputs via a ϵ-LDP mechanism to generate $y_i = \mathcal{M}(x_i)$. We consider an informed adversary with public auxiliary information $\mathbf{t} = \langle t_1, \cdots, t_n \rangle, t_i \in \mathcal{T}$ about each individual. Additionally, just like in the shuffle model, we have a trusted shuffler. It mediates upon the noisy responses $\mathbf{y} = \langle y_1, \cdots, y_n \rangle$ and systematically shuffles them based on $\mathbf{t}$ (since $\mathbf{t}$ is public, it is also accessible to the shuffler) to obtain the final output sequence $\mathbf{z} = \mathcal{A}(\mathbf{y})$ which is sent to the untrusted data analyst. Next, we formally discuss the notion of order and its implications in our setting.

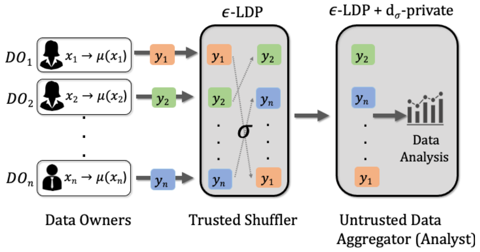

ϵ-LDP
ϵ-LDP + d_σ-private
DO_1
$x_1 \rightarrow \mu(x_1)$
y_1
DO_2
$x_2 \rightarrow \mu(x_2)$
y_2
DO_n
$x_n \rightarrow \mu(x_n)$
y_n
σ
Data Analysis
Data Owners
Trusted Shuffler
Untrusted Data Aggregator (Analyst)

Figure 2: Trust model (similar to shuffle model)

Definition 2.1. (Order) The order of a sequence $\mathbf{x} = \langle x_1, \cdots, x_n \rangle$ refers to the indices of its set of values $\{x_i\}$ and is represented by permutations from S_n.

When the noisy response sequence $\mathbf{y} = \langle y_1, \cdots, y_n \rangle$ is represented by the identity permutation $\sigma_I = (1\ 2\ \cdots\ n)$, the value at index 1 corresponds to DO_1 and so on. Standard LDP releases the identity permutation w.p. 1. The output of the shuffler, $\mathbf{z}$, is some permutation of the sequence $\mathbf{y}$, i.e.,

$$\mathbf{z} = \sigma(\mathbf{y}) = \langle y_{\sigma(1)}, \cdots, y_{\sigma(n)} \rangle$$

where σ is determined via $\mathcal{A}(\cdot)$. For example, for $\sigma = (4\ 5\ 2\ 3\ 1)$, we have $\mathbf{z} = \langle y_4, y_5, y_2, y_3, y_1 \rangle$ which means that the value at index 1 (DO_1) now corresponds to that of DO_4 and so on.

2.2 Definition of d_σ-privacy

Inferential risk captures the threat of an adversary inferring DO_i's private x_i using all or a subset of other data owners' released y_j's. Since we cannot prevent all such attacks and maintain utility, our aim is to formally limit *which data owners* can be leveraged in inferring DO_i's private x_i. To make this precise, each DO_i is assigned a corresponding group, $G_i \subseteq [n]$, of data owners. Each G_i consists of all those DO_js who are similar to DO_i w.r.t auxiliary information t_i, t_j according to some distance metric $d : \mathcal{T} \times \mathcal{T} \rightarrow \mathbb{R}$. Here, we define 'similar' as being under a threshold $r \in \mathbb{R}$:

$$G_i = \{j \in [n] \big| d(t_i, t_j) \leq r\}, \forall i \in [n] \tag{1}$$

$$\mathcal{G} = \{G_1, \cdots, G_n\} \tag{2}$$

For example, $d(\cdot)$ can be Euclidean distance if $\mathcal{T}$ corresponds to geographical locations, thwarting inference attacks using one's immediate neighbors. If $\mathcal{T}$ represents a social media connectivity graph, $d(\cdot)$ can measure the path length between two nodes, thwarting inference attacks using specifically one's friends. By (non-uniformly) shuffling within each group $G_i \in \mathcal{G}$, we prevent an adversary from learning whether a set of k LDP values from G_i correspond to one subset within G_i or another. Ultimately, We maintain indistinguishability between *neighboring permutations*:

Definition 2.2. (Neighboring Permutations) Given a group assignment $\mathcal{G}$, two permutations $\sigma, \sigma' \in \mathrm{S}_n$ are defined to be neighboring w.r.t. a group $G_i \in \mathcal{G}$ (denoted as $\sigma \approx_{G_i} \sigma'$) if

$$\sigma(j) = \sigma'(j)\ \forall j \notin G_i \tag{3}$$

We denote the set of all neighboring permutations as

$$\mathrm{N}_{\mathcal{G}} = \{(\sigma, \sigma') | \sigma \approx_{G_i} \sigma', \forall G_i \in \mathcal{G}\} \tag{4}$$

Now, we formally define d_σ-privacy as follows.

Definition 2.3 (d_σ-privacy)**.** For a given group assignment $\mathcal{G}$ on a set of n entities and a privacy parameter $\alpha \in \mathbb{R}_{\geq 0}$, a randomized mechanism $\mathcal{A} : \mathcal{Y}^n \mapsto \mathcal{V}$ is $(\alpha, \mathcal{G})$-d_σ private if for all $\mathbf{y} \in \mathcal{Y}^n$ and neighboring permutations $\sigma, \sigma' \in \mathrm{N}_{\mathcal{G}}$ and any subset of output $O \subseteq \mathcal{V}$, we have

$$\Pr\big[\mathcal{A}\big(\sigma(\mathbf{y})\big) \in O\big] \leq e^\alpha \cdot \Pr\big[\mathcal{A}\big(\sigma'(\mathbf{y})\big) \in O\big] \tag{5}$$

$\sigma(\mathbf{y})$ and $\sigma'(\mathbf{y})$ are defined to be *neighboring sequences*.

d_σ-privacy states that two neighboring permutations of a data sequences are (almost) equally likely to generate the same output.

An important property of d_σ-privacy is that post-processing computations on the output of a d_σ-private algorithm does not degrade privacy. Additionally, when applied multiple times, the privacy guarantee degrades gracefully. Both the properties are analogous to that of DP and are detailed in App. 5.2.

Privacy Implications. d_σ-privacy offers an inferential guarantee. Regardless of an adversary's prior knowledge of dependence $\Pr_{\mathcal{P}}[x_1, x_2, \ldots, x_n]$, they can't use i's group to make inferences on x_i. Formally, once an adversary knows the (1) set of values $\{y_{G_i}\}$ in i's group, and (2) the sequence of values $\mathbf{y}_{\overline{G_i}}$ outside i's group, they can't learn much about the true x_i:

$$\left| \log \frac{\Pr_{\mathcal{P}}[\mathbf{z} | x_i = a, \{y_{G_i}\}, \mathbf{y}_{\overline{G_i}}]}{\Pr_{\mathcal{P}}[\mathbf{z} | x_i = b, \{y_{G_i}\}, \mathbf{y}_{\overline{G_i}}]} \right|$$

$$= \left| \log \frac{\Pr_{\mathcal{P}}[x_i = a | \mathbf{z}, \{y_{G_i}\}, \mathbf{y}_{\overline{G_i}}]}{\Pr_{\mathcal{P}}[x_i = b | \mathbf{z}, \{y_{G_i}\}, \mathbf{y}_{\overline{G_i}}]} - \log \frac{\Pr_{\mathcal{P}}[x_i = a]}{\Pr_{\mathcal{P}}[x_i = b]} \right| \leq \alpha$$

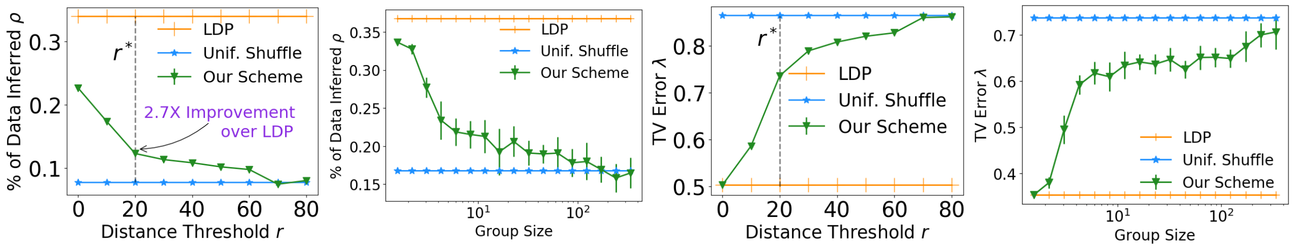
2.3 d_σ-private Shuffling Mechanism

We now describe our novel shuffling mechanism that can achieve d_σ-privacy. In a nutshell, our mechanism samples a permutation from a distribution over permutations known as the Mallows model, a popular probabilistic model for permutations [40]. The mode of the distribution is given by the reference permutation σ_0 – the probability of a permutation increases as we move 'closer' to σ_0 as measured by rank distance metrics, such as the Kendall's tau distance (Def. 5.1). The dispersion parameter θ controls how fast this increase happens.

Definition 2.4. For a dispersion parameter θ, a reference permutation $\sigma_o \in \mathrm{S}_n$, and a rank distance measure $\mathfrak{d} : \mathrm{S}_n \times \mathrm{S}_n \mapsto \mathbb{R}$, $\mathbb{P}_{\Theta,\mathfrak{d}}(\sigma : \sigma_0) = \frac{1}{\psi(\theta,\mathfrak{d})} e^{-\theta \mathfrak{d}(\sigma,\sigma_0)}$ is the Mallows model where $\psi(\theta, \mathfrak{d}) = \sum_{\sigma \in \mathrm{S}_n} e^{-\theta \mathfrak{d}(\sigma,\sigma_0)}$ is a normalization term and $\sigma \in \mathrm{S}_n$.

We can characterize the d_σ-privacy guarantee of our mechanism much as we do the DP guarantee of classical mechanisms: with variance and sensitivity. Intuitively, a larger dispersion parameter $\theta \in \mathbb{R}$ (Def. 2.4) reduces randomness over permutations, increasing utility and increasing (worsening) the privacy parameter α. The most we can increase θ for a given α guarantee depends on the sensitivity of the rank distance measure $\mathfrak{d}(\cdot)$ over all neighboring permutations $N_{\mathcal{G}}$. Formally, we define the sensitivity as

$$\Delta(\sigma_0 : \mathfrak{d}, \mathcal{G}) = \max_{(\sigma,\sigma') \in N_{\mathcal{G}}} \left| \mathfrak{d}(\sigma_0\sigma, \sigma_0) - \mathfrak{d}(\sigma_0\sigma', \sigma_0) \right|,$$



(a) PUDF: Attack

(b) Twitch: Attack

(c) PUDF:Learnability

(d) Twitch: Learnability

Figure 3: Our scheme interpolates between standard LDP (orange line) and uniform shuffling (blue line) in both privacy and data learnability. All plots increase group size along x-axis (except (d)). (a) ⟶ (b): The fraction of participants vulnerable to an inferential attack. (c) ⟶ (d): The accuracy of a calibration model trained on **z** predicting the distribution of LDP outputs at any point $t \in \mathcal{T}$, such as the distribution of medical insurance types used specifically in the Houston area (not possible when uniformly shuffling across Texas).

the maximum change in distance $\mathfrak{d}(\cdot)$ from the reference permutation σ_0 for any pair of neighboring permutations $(\sigma, \sigma') \in N_{\mathcal{G}}$ after applying σ_0 to them, $(\sigma_0\sigma, \sigma_0\sigma')$. The privacy parameter of the mechanism is then proportional to its sensitivity $\alpha = \theta \cdot \Delta(\sigma_0 : \mathfrak{d}, \mathcal{G})$.

The sensitivity of a rank distance measure $\mathfrak{d}(\cdot)$ and reference permutation σ_0 is an increasing function of the *width* parameter, which measures how 'spread apart' the members of G_i are in σ_0:

$$\omega^{\sigma}_{G_i} = \max_{(j,k) \in G_i \times G_i} \left|\sigma^{-1}(j) - \sigma^{-1}(k)\right|, i \in [n]$$
$$\omega^{\sigma}_{\mathcal{G}} = \max_{G_i \in \mathcal{G}} \omega^{\sigma}_{G_i}$$

For Kendall's τ distance $\mathfrak{d}(\cdot)$, the sensitivity is given by $\Delta(\sigma_0 : \mathfrak{d}, \mathcal{G}) = \frac{\omega^{\sigma}_{\mathcal{G}}(\omega^{\sigma}_{\mathcal{G}}+1)}{2}$. If a reference permutation clusters the members of each group closely together (low width) the groups are more likely to permute within themselves. This has two benefits. First, if a group is likely to shuffle within itself, it will have better (η, δ)-preservation (see App. 5.10 for demonstration). Second, since a low-width group is very likely to shuffle its members, we can achieve a lower α for the same dispersion parameter, θ.

Unfortunately, minimizing $\omega^{\sigma}_{\mathcal{G}}$ is an NP-hard problem (Thm. 5.4 in App. 5.5). We instead estimate the optimal σ_0 using a heuristic approach based on a graph breadth first search (details in App. 5.6).

Theorem 2.1. *The Mallows Mechanism is $(\alpha, \mathcal{G})$-d_σ private for $\alpha = \theta \cdot \Delta(\sigma_0 : \mathfrak{d}, \mathcal{G})$ (proof in App. 5.8) .*

3 EVALUATION

Our experiments aim to answer the following two questions: **Q1.** Does the Alg. 1 mechanism protect against realistic inference attacks? **Q2.** How well can Alg. 1 tune a model's ability to learn trends within the shuffled data i.e. tune *data learnability*?

We are not aware of any prior works that provide comparable local inferential privacy. Hence, we baseline our mechanism with the two extremes: standard LDP and uniform random shuffling. See App 6 for further experiments and more detail of experimental methods.

For concreteness, we detail our procedure with the PUDF dataset [2] (license), which includes $n \approx 29$k psychiatric patient records in Texas. Each data owner's sensitive value x_i is their medical payment method, which is reflective of socioeconomic class (such as medicaid or charity). Public side information $t \in \mathcal{T}$ is the hospital's geolocation. Analysts collect such information to better understand how payment methods (and consequently payment amounts) vary from town to town [20]. Uniform shuffling across Texas precludes such analyses. Standard LDP risks inference attacks, since patients attending hospitals in the same neighborhood have similar socioeconomic standing and use similar payment methods, allowing an adversary to correlate their LDP y_i's. Where PUDF uses geographical proximity for $\mathcal{T}$, the Twitch dataset uses proximity in a social network, and the x_i's indicate use of profanity $\mathcal{X} = \{0, 1\}$.

To trade these off inferential threat and learnability, we apply our mechanism with $\alpha = 4$ and check 1) what fraction of the data owners are vulnerable to a simple inferential adversary and 2) how well a calibration model can predict the local distribution of sensitive x_i's near each $t_i \in \mathcal{T}$. The x-axis steadily increases group size either by increasing r or directly by increasing the width $\omega^{\sigma}_{\mathcal{G}}$ accommodated (Twitch). For our attacks Fig. 3a, 3b, a nearest neighbor adversary visits each DO$_i$, picks 25 other data owners close in $d(t_i, \cdot)$, and takes a majority vote of their reported z_j values to predict x_i. For PUDF, this means finding 25 other individuals who visited hospitals near DO$_i$'s, and using their z_j's to predict x_i. We report the fraction of vulnerable data owners, ρ: the subset of data owners for whom this attack is successful over 90% of our LDP trials – although they randomize with LDP, there is a ≥ 90% chance that a simple inference attack can recover their true value. For our learnability test, Fig. 3c, 3d, we train a calibration model to predict the distribution of x_i's near each $t_i \in \mathbf{t}$. For PUDF, this means predicting the distribution of insurance types used near e.g. $t_i =$ Houston.

We observe that our framework effectively interpolates between uniform shuffling and standard LDP for both practical inference attacks and learning tasks.

4 CONCLUSION

We propose a generalized shuffling framework that interpolates between standard LDP and uniform random shuffling. Our new privacy definition, d_σ-privacy, casts new light on the inferential privacy benefits of shuffling.

REFERENCES

[1] [n.d.]. Derangement. https://en.wikipedia.org/wiki/Derangement.

[2] [n.d.]. Hospital Discharge Data Public Use Data File. https://www.dshs.state.tx.us/THCIC/Hospitals/Download.shtm.

[3] [n.d.]. Which Parts of Your Personal Data Are Considered "Public Record"? https://www.checkcriminalrecord.com/which-parts-of-your-personal-data-are-considered-public-record/.

[4] 2019. Supreme Court: State employee birthdates are public record. https://apnews.com/article/c1ff652f271947b2884dfe1216a11bc2/.

[5] Simonetti N. Vazacopoulos A. Balas, E. 2008. Job shop scheduling with setup times, deadlines and precedence constraints. *J Sched* 11 (2008), 253–262. https://doi.org/10.1007/s10951-008-0067-7

[6] Victor Balcer and Albert Cheu. 2020. Separating Local Shuffled Differential Privacy via Histograms. In *ITC*.

[7] Borja Balle, James Bell, Adrià Gascón, and Kobbi Nissim. 2019. The Privacy Blanket of the Shuffle Model. In *Advances in Cryptology – CRYPTO 2019*, Alexandra Boldyreva and Daniele Micciancio (Eds.). Springer International Publishing, Cham, 638–667.

[8] R. Bassily, A. Groce, J. Katz, and A. Smith. 2013. Coupled-Worlds Privacy: Exploiting Adversarial Uncertainty in Statistical Data Privacy. In *2013 IEEE 54th Annual Symposium on Foundations of Computer Science*. 439–448. https://doi.org/10.1109/FOCS.2013.54

[9] Raghav Bhaskar, Abhishek Bhowmick, Vipul Goyal, Srivatsan Laxman, and Abhradeep Thakurta. 2011. Noiseless Database Privacy. In *Advances in Cryptology – ASIACRYPT 2011*, Dong Hoon Lee and Xiaoyun Wang (Eds.). Springer Berlin Heidelberg, Berlin, Heidelberg, 215–232.

[10] Andrea Bittau, Úlfar Erlingsson, Petros Maniatis, Ilya Mironov, Ananth Raghunathan, David Lie, Mitch Rudominer, Ushasree Kode, Julien Tinnes, and Bernhard Seefeld. 2017. Prochlo: Strong Privacy for Analytics in the Crowd. In *Proceedings of the 26th Symposium on Operating Systems Principles* (Shanghai, China) *(SOSP '17)*. Association for Computing Machinery, New York, NY, USA, 441–459. https://doi.org/10.1145/3132747.3132769

[11] Rui Chen, Benjamin C. Fung, Philip S. Yu, and Bipin C. Desai. 2014. Correlated Network Data Publication via Differential Privacy. *The VLDB Journal* 23, 4 (Aug. 2014), 653–676. https://doi.org/10.1007/s00778-013-0344-8

[12] Albert Cheu, Adam Smith, Jonathan Ullman, David Zeber, and Maxim Zhilyaev. 2019. Distributed Differential Privacy via Shuffling. In *Advances in Cryptology – EUROCRYPT 2019*, Yuval Ishai and Vincent Rijmen (Eds.). Springer International Publishing, Cham, 375–403.

[13] Krzysztof M Choromanski, Tony Jebara, and Kui Tang. 2013. Adaptive Anonymity via b-Matching. *Advances in Neural Information Processing Systems* 26 (2013), 3192–3200.

[14] Tore Dalenius. 1977. Towards a methodology for statistical disclosure control. *Statistik Tidskrift* 15 (1977), 429–444.

[15] Bolin Ding, Janardhan Kulkarni, and Sergey Yekhanin. 2017. Collecting Telemetry Data Privately. In *Advances in Neural Information Processing Systems 30*, I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett (Eds.). Curran Associates, Inc., 3571–3580. http://papers.nips.cc/paper/6948-collecting-telemetry-data-privately.pdf

[16] Katerina Doka, Mingqiang Xue, Dimitrios Tsoumakos, and Panagiotis Karras. 2015. k-Anonymization by freeform generalization. In *Proceedings of the 10th ACM Symposium on Information, Computer and Communications Security*. 519–530.

[17] Dheeru Dua and Casey Graff. 2017. UCI Machine Learning Repository. http://archive.ics.uci.edu/ml

[18] Cynthia Dwork and Moni Naor. 2010. On the Difficulties of Disclosure Prevention in Statistical Databases or The Case for Differential Privacy. *Journal of Privacy and Confidentiality* 2 (January 2010), 93–107. https://www.microsoft.com/en-us/research/publication/on-the-difficulties-of-disclosure-prevention-in-statistical-databases-or-the-case-for-differential-privacy/

[19] Cynthia Dwork and Aaron Roth. 2014. The Algorithmic Foundations of Differential Privacy. *Found. Trends Theor. Comput. Sci.* (Aug. 2014), 211–407.

[20] Gary Claxton Eric Lopez. 2020. Comparing Private Payer and Medicare Payment Rates for Select Inpatient Hospital Services. https://www.kff.org/medicare/issue-brief/comparing-private-payer-and-medicare-payment-rates-for-select-inpatient-hospital-services/

[21] Úlfar Erlingsson, Vitaly Feldman, Ilya Mironov, Ananth Raghunathan, Kunal Talwar, and Abhradeep Thakurta. 2019. Amplification by Shuffling: From Local to Central Differential Privacy via Anonymity. In *Proceedings of the Thirtieth Annual ACM-SIAM Symposium on Discrete Algorithms* (San Diego, California) *(SODA '19)*. Society for Industrial and Applied Mathematics, USA, 2468–2479.

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

[23] Vitaly Feldman, Audra McMillan, and Kunal Talwar. 2020. Hiding Among the Clones: A Simple and Nearly Optimal Analysis of Privacy Amplification by Shuffling. arXiv:2012.12803 [cs.LG]

[24] Jerome H Friedman. 2001. Greedy function approximation: a gradient boosting machine. *Annals of statistics* (2001), 1189–1232.

[25] Johannes Gehrke, Michael Hay, Edward Lui, and Rafael Pass. 2012. Crowd-Blending Privacy. In *Advances in Cryptology – CRYPTO 2012*, Reihaneh Safavi-Naini and Ran Canetti (Eds.). Springer Berlin Heidelberg, Berlin, Heidelberg, 479–496.

[26] Johannes Gehrke, Edward Lui, and Rafael Pass. 2011. Towards Privacy for Social Networks: A Zero-Knowledge Based Definition of Privacy. In *Proceedings of the 8th Conference on Theory of Cryptography* (Providence, RI) *(TCC'11)*. Springer-Verlag, Berlin, Heidelberg, 432–449.

[27] Joseph Geumlek and Kamalika Chaudhuri. 2019. Profile-based Privacy for Locally Private Computations. In *IEEE International Symposium on Information Theory, ISIT 2019, Paris, France, July 7-12, 2019*. IEEE, 537–541. https://doi.org/10.1109/ISIT.2019.8849549

[28] Arpita Ghosh and Robert Kleinberg. 2016. Inferential Privacy Guarantees for Differentially Private Mechanisms. *CoRR* abs/1603.01508 (2016). arXiv:1603.01508 http://arxiv.org/abs/1603.01508

[29] Andy Greenberg. 2016. Apple's 'Differential Privacy' Is About Collecting Your Data—But Not *Your* Data. *Wired* (Jun 13 2016).

[30] Krzysztof Grining and Marek Klonowski. 2017. Towards Extending Noiseless Privacy: Dependent Data and More Practical Approach. In *Proceedings of the 2017 ACM on Asia Conference on Computer and Communications Security* (Abu Dhabi, United Arab Emirates) *(ASIA CCS '17)*. Association for Computing Machinery, New York, NY, USA, 546–560. https://doi.org/10.1145/3052973.3052992

[31] Xi He, Ashwin Machanavajjhala, and Bolin Ding. 2014. Blowfish Privacy: Tuning Privacy-Utility Trade-Offs Using Policies. In *Proceedings of the 2014 ACM SIGMOD International Conference on Management of Data* (Snowbird, Utah, USA) *(SIGMOD '14)*. Association for Computing Machinery, New York, NY, USA, 1447–1458. https://doi.org/10.1145/2588555.2588581

[32] S. Kasiviswanathan and A. Smith. 2014. On the 'Semantics' of Differential Privacy: A Bayesian Formulation. *J. Priv. Confidentiality* 6 (2014).

[33] Yusuke Kawamoto and Takao Murakami. 2018. Differentially Private Obfuscation Mechanisms for Hiding Probability Distributions. *CoRR* abs/1812.00939 (2018). arXiv:1812.00939 http://arxiv.org/abs/1812.00939

[34] Daniel Kifer. 2009. Attacks on privacy and deFinetti's theorem. In *Proceedings of the 2009 ACM SIGMOD International Conference on Management of data (SIGMOD '09)*. Association for Computing Machinery, Providence, Rhode Island, USA, 127–138. https://doi.org/10.1145/1559845.1559861

[35] Daniel Kifer and Ashwin Machanavajjhala. 2011. No Free Lunch in Data Privacy. In *Proceedings of the 2011 ACM SIGMOD International Conference on Management of Data* (Athens, Greece) *(SIGMOD '11)*. Association for Computing Machinery, New York, NY, USA, 193–204. https://doi.org/10.1145/1989323.1989345

[36] Daniel Kifer and Ashwin Machanavajjhala. 2014. Pufferfish: A Framework for Mathematical Privacy Definitions. *ACM Trans. Database Syst.* 39, 1, Article 3 (Jan. 2014), 36 pages. https://doi.org/10.1145/2514689

[37] Katrina Ligett, Charlotte Peale, and Omer Reingold. 2020. Bounded-Leakage Differential Privacy. In *1st Symposium on Foundations of Responsible Computing (FORC 2020) (Leibniz International Proceedings in Informatics (LIPIcs), Vol. 156)*, Aaron Roth (Ed.). Schloss Dagstuhl–Leibniz-Zentrum für Informatik, Dagstuhl, Germany, 10:1–10:20. https://doi.org/10.4230/LIPIcs.FORC.2020.10

[38] Changchang Liu, Supriyo Chakraborty, and Prateek Mittal. 2016. Dependence Makes You Vulnberable: Differential Privacy Under Dependent Tuples.. In *NDSS*. The Internet Society. http://dblp.uni-trier.de/db/conf/ndss/ndss2016.html#LiuMC16

[39] Ashwin Machanavajjhala, Daniel Kifer, Johannes Gehrke, and Muthuramakrishnan Venkitasubramaniam. 2007. L-diversity: Privacy beyond k-anonymity. *ACM Transactions on Knowledge Discovery from Data* 1, 1 (March 2007), 3–es. https://doi.org/10.1145/1217299.1217302

[40] C. L. MALLOWS. 1957. NON-NULL RANKING MODELS. I. *Biometrika* 44, 1-2 (06 1957), 114–130. https://doi.org/10.1093/biomet/44.1-2.114 arXiv:https://academic.oup.com/biomet/article-pdf/44/1-2/114/752590/44-1-2-114.pdf

[41] Alexandru Niculescu-Mizil and Rich Caruana. 2005. Predicting good probabilities with supervised learning. In *Proceedings of the 22nd international conference on Machine learning*. 625–632.

[42] Benedek Rozemberczki, Carl Allen, and Rik Sarkar. 2019. Multi-scale attributed node embedding. *arXiv preprint arXiv:1909.13021* (2019). http://snap.stanford.edu/data/twitch-social-networks.html

[43] Shuang Song, Yizhen Wang, and Kamalika Chaudhuri. 2017. Pufferfish Privacy Mechanisms for Correlated Data. In *Proceedings of the 2017 ACM International Conference on Management of Data* (Chicago, Illinois, USA) *(SIGMOD '17)*. Association for Computing Machinery, New York, NY, USA, 1291–1306. https://doi.org/10.1145/3035918.3064025

[44] Latanya Sweeney. 2002. Achieving k-anonymity privacy protection using generalization and suppression. *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems* 10, 5 (Oct. 2002), 571–588. https://doi.org/10.1142/S021848850200165X

[45] Tamir Tassa, Arnon Mazza, and Aristides Gionis. 2012. k-Concealment: An Alternative Model of k-Type Anonymity. *Trans. Data Priv.* 5, 1 (2012), 189–222.

[46] M. C. Tschantz, S. Sen, and A. Datta. 2020. SoK: Differential Privacy as a Causal Property. In *2020 IEEE Symposium on Security and Privacy (SP)*. 354–371. https://doi.org/10.1109/SP40000.2020.00012

[47] Wai Kit Wong, Nikos Mamoulis, and David Wai Lok Cheung. 2010. Non-homogeneous generalization in privacy preserving data publishing. In *Proceedings of the 2010 ACM SIGMOD International Conference on Management of data*. 747–758.

[48] Xiaokui Xiao and Yufei Tao. 2006. Anatomy: Privacy and Correlation Preserving Publication. (Jan. 2006).

[49] Mingqiang Xue, Panagiotis Karras, Chedy Raïssi, Jaideep Vaidya, and Kian-Lee Tan. 2012. Anonymizing set-valued data by nonreciprocal recoding. In *Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining*. 1050–1058.

[50] Bin Yang, Issei Sato, and Hiroshi Nakagawa. 2015. Bayesian Differential Privacy on Correlated Data. In *Proceedings of the 2015 ACM SIGMOD International Conference on Management of Data* (Melbourne, Victoria, Australia) *(SIGMOD '15)*. Association for Computing Machinery, New York, NY, USA, 747–762. https://doi.org/10.1145/2723372.2747643

[51] Wanrong Zhang, Olga Ohrimenko, and Rachel Cummings. 2020. Attribute Privacy: Framework and Mechanisms. arXiv:2009.04013 [cs.CR]

[52] T. Zhu, P. Xiong, G. Li, and W. Zhou. 2015. Correlated Differential Privacy: Hiding Information in Non-IID Data Set. *IEEE Transactions on Information Forensics and Security* 10, 2 (2015), 229–242. https://doi.org/10.1109/TIFS.2014.2368363

5 APPENDIX

5.1 Background Cntd.

Here we define two rank distance measures

Definition 5.1 (Kendall's τ Distance)**.** For any two permutations, $\sigma, \pi \in \mathrm{S}_n$, the Kendall's τ distance $\delta_\tau(\sigma, \pi)$ counts the number of pairwise disagreements between σ and π, i.e., the number of item pairs that have a relative order in one permutation and a different order in the other. Formally,

$$\delta_\tau(\sigma,\pi) = \Big| \big\{(i,j) : i < j, \big[\sigma(i) > \sigma(j) \wedge \pi(i) < \pi(j)\big] \\ \vee \big[\sigma(i) < \sigma(j) \wedge \pi(i) > \pi(j)\big]\big\} \Big| \tag{6}$$

For example, if $\sigma = (1\,2\,3\,4\,5\,6\,7\,8\,9\,10)$ and $\pi = (1\,2\,3\,\underline{6}\,5\,\underline{4}\,7\,8\,9\,10)$, then $\delta_\tau(\sigma, \pi) = 3$.

Next, Hamming distance measure is defined as follows.

Definition 5.2 (Hamming Distance)**.** For any two permutations, $\sigma, \pi \in \mathrm{S}_n$, the Hamming distance $\delta_H(\sigma, \pi)$ counts the number of positions in which the two permutations disagree. Formally,

$$\delta_H(\sigma,\pi) = \Big|\big\{i \in [n] : \sigma(i) \neq \pi(i)\big\}\Big|$$

Repeating the above example, if $\sigma = (1\ 2\ 3\ 4\ 5\ 6\ 7\ 8\,9\,10)$ and $\pi = (1\,2\,3\,\underline{6}\,5\,\underline{4}\,7\,8\,9\,10)$, then $\delta_H(\sigma, \pi) = 2$.

5.2 Additional Properties of d_σ-privacy

Lemma 5.1 (Convexity). *Let $\mathcal{A}_1, \dots \mathcal{A}_k : \mathcal{Y}^n \mapsto \mathcal{V}$ be a collection of k $(\alpha, \mathcal{G})$-$d_\sigma$private mechanisms for a given group assignment $\mathcal{G}$ on a set of n entities. Let $\mathcal{A} : \mathcal{Y}^n \mapsto \mathcal{V}$ be a convex combination of these k mechanisms, where the probability of releasing the output of mechanism $\mathcal{A}_i$ is p_i, and $\sum_{i=1}^{k} p_i = 1$. $\mathcal{A}$ is also $(\alpha, \mathcal{G})$-$d_\sigma$private w.r.t. $\mathcal{G}$.*

Proof. For any $(\sigma, \sigma') \in \mathrm{N}_{\mathcal{G}}$ and $\mathbf{y} \in \mathcal{Y}$:

$$\begin{aligned}\Pr[\mathcal{A}(\sigma(\mathbf{y})) \in O] &= \sum_{i=1}^{k} p_i \Pr[\mathcal{A}_i(\sigma(\mathbf{y})) \in O] \\ &\leq e^{\alpha} \sum_{i=1}^{k} p_i \Pr[\mathcal{A}_i(\sigma'(\mathbf{y})) \in O] \\ &= \Pr[\mathcal{A}(\sigma'(\mathbf{y})) \in O]\end{aligned}$$

□

Theorem 5.1 (Post-processing)**.** *Let $\mathcal{A} : \mathcal{Y}^n \mapsto \mathcal{V}$ be $(\alpha, \mathcal{G})$-$d_\sigma$private for a given group assignment $\mathcal{G}$ on a set of n entities. Let $f : \mathcal{V} \mapsto \mathcal{V}'$ be an arbitrary randomized mapping. Then $f \circ \mathcal{A} : \mathcal{Y}^n \mapsto \mathcal{V}'$ is also $(\alpha, \mathcal{G})$-$d_\sigma$private.*

Proof. Let $g : \mathcal{V} \to \mathcal{V}'$ be a deterministic, measurable function. For any output event $\mathcal{Z} \subset \mathcal{V}'$, let $\mathcal{W}$ be its preimage:
$\mathcal{W} = \{v \in \mathcal{V} | g(v) \in \mathcal{Z}\}$. Then, for any $(\sigma, \sigma') \in \mathrm{N}_{\mathcal{G}}$,

$$\begin{aligned}\Pr\Big[g\Big(\mathcal{A}(\sigma(\mathbf{y}))\Big) \in \mathcal{Z}\Big] &= \Pr\Big[\mathcal{A}(\sigma(\mathbf{y})) \in \mathcal{W}\Big] \\ &\leq e^{\alpha} \cdot \Pr\Big[\mathcal{A}(\sigma'(\mathbf{y})) \in \mathcal{W}\Big] \\ &= e^{\alpha} \cdot \Pr\Big[g\Big(\mathcal{A}(\sigma'(\mathbf{y}))\Big) \in \mathcal{Z}\Big]\end{aligned}$$

This concludes our proof because any randomized mapping can be decomposed into a convex combination of measurable, deterministic functions [19], and as Lemma 5.1 shows, a convex combination of $(\alpha, \mathcal{G})$-$d_\sigma$private mechanisms is also $(\alpha, \mathcal{G})$-$d_\sigma$private. □

Theorem 5.2 (Sequential Composition)**.** *If $\mathcal{A}_1$ and $\mathcal{A}_2$ are $(\alpha_1, \mathcal{G})$- and $(\alpha_2, \mathcal{G})$-$d_\sigma$private mechanisms, respectively, that use independent randomness, then releasing the outputs $\big(\mathcal{A}_1(\mathbf{y}), \mathcal{A}_2(\mathbf{y})\big)$ satisfies $(\alpha_1 + \alpha_2, \mathcal{G})$-$d_\sigma$privacy.*

Proof. We have that $\mathcal{A}_1 : \mathcal{Y}^n \to \mathcal{V}'$ and $\mathcal{A}_1 : \mathcal{Y}^n \to \mathcal{V}''$ each satisfy d_σ-privacy for different α values. Let $\mathcal{A} : \mathcal{Y}^n \to (\mathcal{V}' \times \mathcal{V}'')$ output $\big(\mathcal{A}_1(\mathbf{y}), \mathcal{A}_2(\mathbf{y})\big)$. Then, we may write any event $\mathcal{Z} \in (\mathcal{V}' \times \mathcal{V}'')$ as $\mathcal{Z}' \times \mathcal{Z}''$, where $\mathcal{Z}' \in \mathcal{V}'$ and $\mathcal{Z}'' \in \mathcal{V}''$. We have for any $(\sigma, \sigma') \in \mathrm{N}_{\mathcal{G}}$,

$$\begin{aligned}\Pr\big[\mathcal{A}(\sigma(\mathbf{y})) \in \mathcal{Z}\big] &= \Pr\big[\big(\mathcal{A}_1(\sigma(\mathbf{y})), \mathcal{A}_2(\sigma(\mathbf{y}))\big) \in \mathcal{Z}\big] \\ &= \Pr\big[\{\mathcal{A}_1(\sigma(\mathbf{y})) \in \mathcal{Z}'\} \cap \{\mathcal{A}_2(\sigma(\mathbf{y})) \in \mathcal{Z}''\}\big] \\ &= \Pr\big[\{\mathcal{A}_1(\sigma(\mathbf{y})) \in \mathcal{Z}'\}\big]\Pr\big[\{\mathcal{A}_2(\sigma(\mathbf{y})) \in \mathcal{Z}''\}\big] \\ &\leq e^{\alpha_1+\alpha_2}\Pr\big[\{\mathcal{A}_1(\sigma'(\mathbf{y})) \in \mathcal{Z}'\}\big]\Pr\big[\{\mathcal{A}_2(\sigma'(\mathbf{y})) \in \mathcal{Z}''\}\big] \\ &= e^{\alpha_1+\alpha_2} \cdot \Pr\big[\mathcal{A}(\sigma'(\mathbf{y})) \in \mathcal{Z}\big]\end{aligned}$$

□

Proof of Lemma 5.2

Lemma 5.2. *An ϵ-LDP mechanism is $(k\epsilon, \mathcal{G})$-d_σ private for any group assignment $\mathcal{G}$ such that $k \geq \max_{G_i \in \mathcal{G}} |G_i|$*

Proof. This follows from k-group privacy [19]. $\mathbf{y}$ are ε-LDP outputs $\mathcal{A}_{\mathrm{LDP}}(\mathbf{x})$ from input sequence $\mathbf{x}$. For any $\sigma \approx_{G_i} \sigma'$, we know by definition that $\sigma(j) = \sigma'(j)$ for all $j \notin G_i$. As such, the permuted sequences $\sigma(\mathbf{x})_j = \sigma'(\mathbf{x})_j$ for all $j \notin G_i$, and differ in at most $|G_i|$ entries. In other words,

$$\delta_H\big(\sigma(\mathbf{x}), \sigma'(\mathbf{x})\big) \leq |G_i|$$

Using this fact, we have from the k-group property of LDP that

$$\Pr\big[\mathcal{A}_{\mathrm{LDP}}(\sigma(\mathbf{x})) \in O\big] \leq e^{|G_i|\epsilon}\Pr\big[\mathcal{A}_{\mathrm{LDP}}(\sigma'(\mathbf{x})) \in O\big]$$

and thus if $k \geq \max_{G_i \in \mathcal{G}} |G_i|$,

$$\Pr\big[\mathcal{A}_{\mathrm{LDP}}(\sigma(\mathbf{x})) \in O\big] \leq e^{k\epsilon}\Pr\big[\mathcal{A}_{\mathrm{LDP}}(\sigma'(\mathbf{x})) \in O\big]$$

for all $(\sigma, \sigma') \in \mathrm{N}_{\mathcal{G}}$. □

5.3 Proof for Thm. 5.3

Theorem 5.3. *For a given group assignment $\mathcal{G}$ on a set of n data owners, if a shuffling mechanism $\mathcal{A} : \mathcal{Y}^n \mapsto \mathcal{Y}^n$ is $(\alpha, \mathcal{G})$-$d_\sigma$private, then for each data owner DO_i, $i \in [n]$,*

$$\mathbb{L}^{\sigma}_{\mathcal{P}}(\mathbf{x}) = \max_{\substack{i \in [n] \\ a,b \in \mathcal{X}}} \left| \log \frac{\Pr_{\mathcal{P}}[\mathbf{z} | x_i = a, \{y_{G_i}\}, \mathbf{y}_{\overline{G_i}}]}{\Pr_{\mathcal{P}}[\mathbf{z} | x_i = b, \{y_{G_i}\}, \mathbf{y}_{\overline{G_i}}]} \right| \leq \alpha$$

for a prior distribution $\mathcal{P}$, where $\mathbf{z} = \mathcal{A}(\mathbf{y})$ and $\mathbf{y}_{\overline{G_i}}$ is the noisy sequence for all data owners outside G_i .

Proof.

$$
\begin{aligned}
&\frac{\Pr_{\mathcal{P}}[\mathbf{z}|x_i = a, \{y_{G_i}\}, \mathbf{y}_{\overline{G_i}}]}{\Pr_{\mathcal{P}}[\mathbf{z}|x_i = b, \{y_{G_i}\}, \mathbf{y}_{\overline{G_i}}]}\\
&= \frac{\int \Pr_{\mathcal{P}}[\mathbf{y}|x_i = a, \{y_{G_i}\}, \mathbf{y}_{\overline{G_i}}] \Pr_{\mathcal{A}}[\mathbf{z}|\mathbf{y}]d\mathbf{y}}{\int \Pr_{\mathcal{P}}[\mathbf{y}|x_i = b, \{y_{G_i}\}, \mathbf{y}_{\overline{G_i}}] \Pr_{\mathcal{A}}[\mathbf{z}|\mathbf{y}]d\mathbf{y}}\\
&= \frac{\sum_{\sigma \in \mathrm{S}_m} \Pr_{\mathcal{P}}[\sigma(\mathbf{y}^*_{G_i})|x_i = a, \mathbf{y}_{\overline{G_i}}] \Pr_{\mathcal{A}}[\mathbf{z}|\sigma(\mathbf{y}^*_{G_i}), \mathbf{y}_{\overline{G_i}}]}{\sum_{\sigma \in \mathrm{S}_m} \Pr_{\mathcal{P}}[\sigma(\mathbf{y}^*_{G_i})|x_i = b, \mathbf{y}_{\overline{G_i}}] \Pr_{\mathcal{A}}[\mathbf{z}|\sigma(\mathbf{y}^*_{G_i}), \mathbf{y}_{\overline{G_i}}]}\\
&\leq \max_{\{\sigma,\sigma' \in \mathrm{S}_m\}} \frac{\Pr_{\mathcal{A}}[\mathbf{z}|\sigma(\mathbf{y}^*_{G_i}), \mathbf{y}_{\overline{G_i}}]}{\Pr_{\mathcal{A}}[\mathbf{z}|\sigma'(\mathbf{y}^*_{G_i}), \mathbf{y}_{\overline{G_i}}]}\\
&\leq \max_{\{\sigma,\sigma' \in \mathrm{N}_{G_i}\}} \frac{\Pr_{\mathcal{A}}[\mathbf{z}|\sigma(\mathbf{y})]}{\Pr_{\mathcal{A}}[\mathbf{z}|\sigma'(\mathbf{y})]}\\
&\leq e^{\alpha}
\end{aligned}
$$

The second line simply marginalizes out the full noisy sequence $\mathbf{y}$. The third line reduces this to a sum over permutations of of $\mathbf{y}_{G_i}$, where $m = |G_i|$ and $\mathbf{y}^*_{G_i}$ is any fixed permutation of values $\{y_{G_i}\}$. This is possible since we are given the values outside the group, $\mathbf{y}_{\overline{G_i}}$, and the unordered set of values inside the group, $\{y_{G_i}\}$.

The fourth line uses the fact that the numerator and denominator are both convex combinations of $\Pr_{\mathcal{A}}[\mathbf{z}|\sigma(\mathbf{y}^*_{G_i}), \mathbf{y}_{\overline{G_i}}]$ over all $\sigma \in \mathrm{S}_m$.

The fifth line uses the fact that for any $\mathbf{y}_{\overline{G_i}}$,

$$(\sigma(\mathbf{y}^*_{G_i}), \mathbf{y}_{\overline{G_i}}) \approx_{G_i} (\sigma'(\mathbf{y}^*_{G_i}), \mathbf{y}_{\overline{G_i}}) \ .$$

This allows a further upper bound over all neighboring sequences w.r.t. G_i, and thus over any permutation of $\mathbf{y}_{\overline{G_i}}$, as long as it is the same in the numerator and denominator. □

5.4 Discussion on Properties of Mallows Mechanism

Property 1. *For group assignment $\mathcal{G}$, a mechanism $\mathcal{A}(\cdot)$ that shuffles according to a permutation sampled from the Mallows model $\mathbb{P}_{\theta,\mathfrak{d}}(\cdot)$, satisfies $(\alpha, \mathcal{G})$-$d_\sigma$privacy where*

$$
\begin{aligned}
\Delta(\sigma_0 : \mathfrak{d}, \mathcal{G}) = \max_{(\sigma,\sigma') \in N_{\mathcal{G}}} |\mathfrak{d}(\sigma_0\sigma, \sigma_0) - \mathfrak{d}(\sigma_0\sigma', \sigma_0)|\\
\alpha = \theta \cdot \Delta(\sigma_0 : \mathfrak{d}, \mathcal{G})
\end{aligned}
$$

We refer to $\Delta(\sigma_0 : \mathfrak{d}, \mathcal{G})$ as the sensitivity of the rank-distance measure $\mathfrak{d}(\cdot)$

Proof. Consider two permutations of the initial sequence $\mathbf{y}$, $\sigma_1(\mathbf{y}), \sigma_2(\mathbf{y})$ that are neighboring w.r.t. some group $G_i \in \mathcal{G}$, $\sigma_1 \approx_{G_i} \sigma_2$. Additionally consider any fixed released shuffled sequence $\mathbf{z}$. Let Σ_1, Σ_2 be the set of permutations that turn $\sigma_1(\mathbf{y}), \sigma_2(\mathbf{y})$ into $\mathbf{z}$, respectively:

$$
\begin{aligned}
\Sigma_1 &= \{\sigma \in \mathrm{S}_n : \sigma\sigma_1(\mathbf{y}) = \mathbf{z}\}\\
\Sigma_2 &= \{\sigma \in \mathrm{S}_n : \sigma\sigma_2(\mathbf{y}) = \mathbf{z}\} \quad .
\end{aligned}
$$

In the case that $\{y\}$ consists entirely of unique values, Σ_1, Σ_2 will each contain exactly one permutation, since only one permutation can map $\sigma_i(\mathbf{y})$ to $\mathbf{z}$.

Lemma 5.3. *For each permutation $\sigma'_1 \in \Sigma_1$ there exists a permutation in $\sigma'_2 \in \Sigma_2$ such that*

$$\sigma'_1 \approx_{G_i} \sigma'_2 \quad .$$

Proof follows from the fact that — since only the elements $j \in G_i$ differ in $\sigma_1(\mathbf{y})$ and $\sigma_2(\mathbf{y})$ — only those elements need to differ to achieve the same output permutation. In other words, we may define σ'_1, σ'_2 at all inputs $i \notin G_i$ identically, and then define all inputs $i \in G_i$ differently as needed. As such, they are neighboring w.r.t. G_i.

Recalling that Alg. 1 applies σ_0^{-1} to the sampled permutation, we must sample $\sigma_0\sigma'_1$ (for some $\sigma'_1 \in \Sigma_1$) for the mechanism to produce $\mathbf{z}$ from $\sigma_1(\mathbf{y})$. Formally, since $\sigma'_1\sigma_1(\mathbf{y}) = \mathbf{z}$ we must sample $\sigma_0\sigma'_1$ to get $\mathbf{z}$ since we are going to apply σ_0^{-1} to the sampled permutation.

$$
\begin{aligned}
\Pr\left[\mathcal{A}(\sigma_1(\mathbf{y})) = \mathbf{z}\right] = \mathbb{P}_{\theta,\mathfrak{d}}\left(\sigma_0\sigma', \sigma' \in \Sigma_1 : \sigma_0\right)\\
\Pr\left[\mathcal{A}(\sigma_2(\mathbf{y})) = \mathbf{z}\right] = \mathbb{P}_{\theta,\mathfrak{d}}\left(\sigma_0\sigma', \sigma' \in \Sigma_2 : \sigma_0\right)
\end{aligned}
$$

Taking the log odds, we have

$$
\begin{aligned}
\frac{\mathbb{P}_{\theta,\mathfrak{d}}\left(\sigma_0\sigma', \sigma' \in \Sigma_1 : \sigma_0\right)}{\mathbb{P}_{\theta,\mathfrak{d}}\left(\sigma_0\sigma', \sigma' \in \Sigma_2 : \sigma_0\right)} &= \frac{\sum_{\sigma' \in \Sigma_1} \mathbb{P}_{\Theta,\mathfrak{d}}(\sigma_0\sigma' : \sigma_0)}{\sum_{\sigma' \in \Sigma_2} \mathbb{P}_{\Theta,\mathfrak{d}}(\sigma_0\sigma' : \sigma_0)}\\
&= \frac{\sum_{\sigma' \in \Sigma_1} e^{-\theta\mathfrak{d}(\sigma_0\sigma', \sigma_0)}}{\sum_{\sigma' \in \Sigma_2} e^{-\theta\mathfrak{d}(\sigma_0\sigma', \sigma_0)}}\\
&\leq \frac{e^{-\theta\mathfrak{d}(\sigma_0\sigma_a, \sigma_0)}}{e^{-\theta\mathfrak{d}(\sigma_0\sigma_b, \sigma_0)}}\\
&\leq e^{\theta|\mathfrak{d}(\sigma_0\sigma_a, \sigma_0) - \mathfrak{d}(\sigma_0\sigma_b, \sigma_0)|}\\
&\leq e^{\theta\Delta}
\end{aligned}
$$

Therefore, setting $\alpha = \Delta$, we achieve $(\alpha, \mathcal{G})$-$d_\sigma$privacy. □

Property 2. *The sensitivity of a rank-distance is an increasing function of the width $\omega^{\sigma_0}_{\mathcal{G}}$. For instance, for Kendall's τ distance $\mathfrak{d}_\tau(\cdot)$, we have $\Delta(\sigma_0 : \mathfrak{d}_\tau, \mathcal{G}) = \frac{\omega^{\sigma_0}_{\mathcal{G}}(\omega^{\sigma_0}_{\mathcal{G}}+1)}{2}$.*

To show the sensitivity of Kendall's τ, we make use of its triangle inequality.

Proof. Recall from the proof of the previous property that the expression $\mathfrak{d}(\sigma, \sigma_0) = \mathfrak{d}(\sigma_0\sigma, \sigma_0)$, where $\mathfrak{d}$ is the actual rank distance measure e.g. Kendall's τ. As such, we require that

$$\left|\mathfrak{d}(\sigma_0\sigma_a, \sigma_0) - \mathfrak{d}(\sigma_0\sigma_b, \sigma_0)\right| \leq \frac{\omega^{\sigma_0}_{\mathcal{G}}(\omega^{\sigma_0}_{\mathcal{G}} + 1)}{2}$$

for any pair of permutations $(\sigma_a, \sigma_b) \in N_{\mathcal{G}}$.

For any group $G_i \in \mathcal{G}$, let $W_i \subseteq n$ represent the smallest contiguous subsequence of indices in σ_0 that contains all of G_i.

For instance, if $\sigma_0 = [2, 4, 6, 8, 1, 3, 5, 7]$ and $G_i = \{2, 6, 8\}$, then $W_i = \{2, 4, 6, 8\}$. Then the group width width is $\omega_i = |W_i| - 1 = 3$. Now consider two permutations neighboring w.r.t. G_i, $\sigma_a \approx_{G_i} \sigma_b$, so only the elements of G_i are shuffled between them. We want to bound

$$\left|\mathfrak{d}(\sigma_0\sigma_a, \sigma_0) - \mathfrak{d}(\sigma_0\sigma_b, \sigma_0)\right|$$

For this, we use a pair of triangle inequalities:

$$\mathfrak{d}(\sigma_0\sigma_a, \sigma_0\sigma_b) \geq \mathfrak{d}(\sigma_0\sigma_a, \sigma_0) - \mathfrak{d}(\sigma_0\sigma_b, \sigma_0) \quad \&$$
$$\mathfrak{d}(\sigma_0\sigma_a, \sigma_0\sigma_b) \geq \mathfrak{d}(\sigma_0\sigma_b, \sigma_0) - \mathfrak{d}(\sigma_0\sigma_a, \sigma_0)$$

so,

$$\left|\mathfrak{d}(\sigma_0\sigma_a, \sigma_0) - \mathfrak{d}(\sigma_0\sigma_b, \sigma_0)\right| \leq \mathfrak{d}(\sigma_0\sigma_a, \sigma_0\sigma_b)$$

Since $\sigma_0\sigma_a$ and $\sigma_0\sigma_b$ only differ in the contiguous subset W_i, the largest number of discordant pairs between them is given by the maximum Kendall's τ distance between two permutations of size $\omega_i + 1$:

$$|\mathfrak{d}(\sigma_0\sigma_a, \sigma_0\sigma_b)| \leq \frac{\omega_i(\omega_i + 1)}{2}$$

Since $\omega_{\mathcal{G}}^{\sigma_0} \geq \omega_i$ for all $G_i \in \mathcal{G}$, we have that

$$\Delta(\sigma_0 : \mathfrak{d}, \mathcal{G}) \leq \frac{\omega_{\mathcal{G}}^{\sigma_0}(\omega_{\mathcal{G}}^{\sigma_0} + 1)}{2}$$

□

5.5 Hardness of Computing The Optimum Reference Permutation

Theorem 5.4. *The problem of finding the optimum reference permutation, i.e.,* $\sigma_0 = \arg\min_{\sigma \in \mathrm{S}_n} \omega_{\mathcal{G}}^{\sigma}$ *is NP-hard.*

Proof. We start with the formal representation of the problem as follows.

Optimum Reference Permutation Problem. Given n subsets $\mathcal{G} = \{G_i \in 2^{[n]}, i \in [n]\}$, find the permutation $\sigma_0 = \arg\min_{\sigma \in \mathrm{S}_n} \omega_{\mathcal{G}}^{\sigma}$.

Now, consider the following job-shop scheduling problem.

Job Shop Scheduling. There is one job J with n operations $o_i, i \in [n]$ and n machines such that o_i needs to run on machine M_i. Additionally, each machine has a sequence dependent processing time p_i. Let S be the sequence till There are n subsets $S_i \subseteq [n]$, each corresponding to a set of operations that need to occur in contiguous machines, else the processing times incur penalty as follows. Let p_i denote the processing time for the machine running the i-th operation scheduled. Let $\mathbb{S}_i$ be the prefix sequence with i schedulings. For instance, if the final scheduling is 1 3 4 5 9 8 10 6 7 2 then $\mathbb{S}_4 = 1345$. Additionally, let $P_{\mathbb{S}_i}^{j}$ be the shortest subsequence such of $\mathbb{S}_i$ such that it contains all the elements in $S_j \cap \{\mathbb{S}_i\}$. For example for $S_1 = \{3, 5, 7\}$, $P_{\mathbb{S}_4}^{1} = 345$.

$$p_i = \max_{i \in [n]} (|P_{\mathbb{S}_i}^{j}| - |S_j \cap \{\mathbb{S}_i\}|) \tag{7}$$

The objective is to find a scheduling for J such that it minimizes the makespan, i.e., the completion time of the job. Note that $p_n = \max_i p_i$, hence the problem reduces to minimizing p_n.

Lemma 5.4. *The aforementioned job shop scheduling problem with sequence-dependent processing time is NP-hard.*

Proof. Consider the following instantiation of the sequence-dependent job shop scheduling problem where the processing time is given by p_i=$p_{i-1} + w_{kl}, p_1 = 0$ where $\mathbb{S}_i[i-1] = k$, $\mathbb{S}_i[i] = l$ and $w_{ij}, j \in S_i$ represents some associated weight. This problem is equivalent to the travelling salesman problem (TSP) [5] and is therefore, NP-hard. Thus, our aforementioned job shop scheduling problem is also clearly NP-hard. □

Reduction: Let the n subsets S_i correspond to the groups in $\mathcal{G}$. Clearly, minimizing $\omega_{\mathcal{G}}^{\sigma}$ minimizes p_n. Hence, the optimal reference permutation gives the solution to the scheduling problem as well.

□

5.6 Description of Shuffling Mechanism

Algorithm 1: d_σ-private Shuffling Mech.

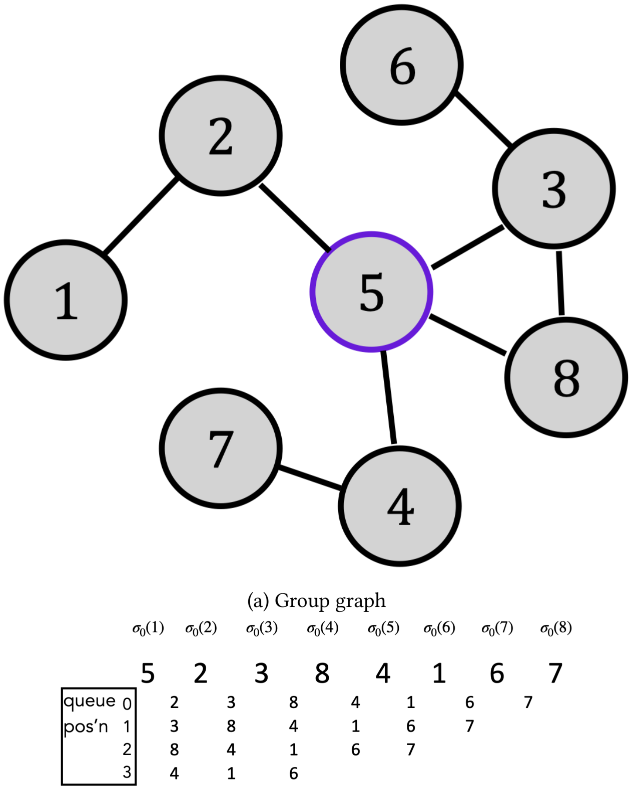
```
Input: LDP sequence y = $\langle y_1, \cdots, y_n \rangle$;
        Public aux. info. $\mathbf{t} = \langle t_1, \cdots t_n \rangle$;
        Dist. threshold $r$; Priv. param. $\alpha$;
Output: z - Shuffled output sequence;
1  $\mathcal{G}$ = ComputeGroupAssignment $(\mathbf{t}, r)$;
2  Construct graph $\mathbb{G}$ with
       a) vertices $V = \{1, 2, \cdots, n\}$
:      b) edges $E = \{(i, j) : j \in G_i, G_i \in \mathcal{G}\}$
3  $root = \arg\max_{i \in [n]} |G_i|$;
4  $\sigma_0$ = BFS($\mathbb{G}, root$);
5  $\Delta$= ComputeSensitivity$(\sigma_0, \mathcal{G})$
6  $\theta = \alpha/\Delta$;
7  $\hat{\sigma} \sim \mathbb{P}_{\theta,\mathfrak{d}}(\sigma_0)$ ;
8  $\sigma^* = \sigma_0^{-1}\hat{\sigma}$;
9  $\mathbf{z} = \langle y_{\sigma^*(1)}, \cdots y_{\sigma^*(n)} \rangle$;
10 Return z;
```

Alg. 1 above proceeds as follows. We first compute the group assignment, $\mathcal{G}$, based on the public auxiliary information and desired distance threshold r. Then we construct σ_0 with a breadth first search (BFS) graph traversal.

We translate $\mathcal{G}$ into an undirected graph (V, E), where the vertices are indices $V = [n]$ and two indices i, j are connected by an edge if they are both in some group (Step 2). Next, σ_0 is computed via a breadth first search traversal (Step 4) – if the k-th node in the traversal is i, then $\sigma_0(k) = i$. The rationale is that neighbors of i (members of G_i) would be traversed in close succession. Hence, a neighboring node j is likely to be traversed at some step h near k which means $|\sigma_0^{-1}(i) - \sigma_0^{-1}(j)| = |h - k|$ would be small (resulting in low width). Additionally, starting from the node with the highest degree (Steps 3-4) which corresponds to the largest group in $\mathcal{G}$ (lower bound for $\omega_{\mathcal{G}}^{\sigma}$ for any σ) helps to curtail the maximum width in σ_0

This is followed by the computation of the dispersion parameter, θ, for our Mallows model (Steps 5-6). Next, we sample a permutation from the Mallows model (Step 7) $\hat{\sigma} \sim \mathbb{P}_\theta(\sigma : \sigma_0)$ and we apply the inverse reference permutation to it, $\sigma^* = \sigma_0^{-1}\hat{\sigma}$ to obtain the desired permutation for shuffling. Recall that $\hat{\sigma}$ is (most likely) close to σ_0, which is unrelated to the original order of the data. σ_0^{-1} therefore brings σ^* back to a shuffled version of the original sequence (identity permutation σ_I). Note that since Alg. 1 is publicly known, the adversary/analyst knows σ_0. Hence, even in the absence of this step from our algorithm, the adversary/analyst could perform this anyway. Finally, we permute $\mathbf{y}$ according to σ^* and output the result $\mathbf{z} = \hat{\sigma}(\mathbf{y})$ (Steps 9-10).

5.7 Illustration of Alg. 1



(a) Group graph

	$\sigma_0(1)$	$\sigma_0(2)$	$\sigma_0(3)$	$\sigma_0(4)$	$\sigma_0(5)$	$\sigma_0(6)$	$\sigma_0(7)$	$\sigma_0(8)$
	5	2	3	8	4	1	6	7
queue pos'n 0	2	3	8	4	1	6	7	
1	3	8	4	1	6	7		
2	8	4	1	6	7			
3	4	1	6					

(b) BFS reference permutation σ_0

Figure 4: Illustration of Alg. 10

We now provide a small-scale step-by-step example of how Alg. 10 operates.

Fig. 4a is an example of a grouping $\mathcal{G}$ on a dataset of $n = 8$ elements. The group of DO$_i$ includes i and its neighbors. For instance, $G_8 = \{8, 3, 5\}$. To build a reference permutation, Alg. 10 starts at the index with the largest group, $i = 5$ (highlighted in purple), with $G_5 = \{5, 2, 3, 8, 4\}$. As shown in Figure 4b, the σ_0 is then constructed by following a BFS traversal from $i = 5$. Each $j \in G_5$ is visited, queuing up the neighbors of each $j \in G_5$ that haven't been visited along the way, and so on. The algorithm completes after the entire graph has been visited.

The goal is to produce a reference permutation in which the width of each group in the reference permutation ω_i is small. In this case, the width of the largest group G_5 is as small as it can be $\omega_5 = 5 - 1 = 4$. However, the width of $G_4 = \{4, 5, 7\}$ is the maximum possible since $\sigma^{-1}(5) = 1$ and $\sigma^{-1}(7) = 8$, so $\omega_4 = 7$. This is difficult to avoid when the maximum group size is large as compared to the full dataset size n. Realistically, we expect n to be significantly larger, leading to relatively smaller groups.

With the reference permutation in place, we compute the sensitivity:

$$\begin{aligned}\Delta(\sigma_0 : \delta, \mathcal{G}) &= \frac{\omega_4(\omega_4 + 1)}{2} \\ &= 28\end{aligned}$$

Which lets us set $\theta = \frac{\alpha}{28}$ for any given α privacy value. To reiterate, lower θ results in more randomness in the mechanism.

We then sample the permutation $\hat{\sigma} = \mathbb{P}_{\theta,\delta}(\sigma_0)$. Suppose

$$\hat{\sigma} = [3\ 2\ 5\ 4\ 8\ 1\ 7\ 6]$$

Then, the released $\mathbf{z}$ is given as

$$\begin{aligned}\mathbf{z} = \sigma^* &= \sigma^{-1}\hat{\sigma}(\mathbf{y}) \\ &= [y_1\ y_2\ y_5\ y_8\ y_3\ y_7\ y_6\ y_4]\end{aligned}$$

One can think of the above operation as follows. What was 5 in the reference permutation ($\sigma_0(1) = 5$) is 3 in the sampled permutation ($\hat{\sigma}(1) = 3$). So, index 5 corresponding to DO$_5$ now holds DO$_3$'s noisy data y_3. As such, we shuffle mostly between members of the same group, and minimally between groups.

5.8 Proof of Thm. 2.1

Theorem 2.1 *Alg. 1 is $(\alpha, \mathcal{G})$-d_σ private.*

Proof. The proof follows from Prop. 1. Having computed the sensitivity of the reference permutation σ_0, Δ, and set $\theta = \alpha/\Delta$, we are guaranteed by Property 1 that shuffling according to the permutation $\hat{\sigma}$ guarantees $(\alpha, \mathcal{G})$-$d_\sigma$privacy.

In the algorithm, we permute by $\sigma_0^{-1}\hat{\sigma}$. Since this is equivalent to first permuting by $\hat{\sigma}$ and then permuting by σ_0^{-1}, this too guarantees $(\alpha, \mathcal{G})$-$d_\sigma$privacy by the immunity to post-processing property (Thm. 5.1). □

.

5.9 Proof of Thm. 5.5

Theorem 5.5. *Theorem Alg. 1 satisfies $(\alpha', \mathcal{G}')$-$d_\sigma$privacy for any group assignment $\mathcal{G}'$ where $\alpha' = \alpha\frac{\Delta(\sigma_0:\delta,\mathcal{G}')}{\Delta(\sigma_0:\delta,\mathcal{G})}$*

Proof. Recall from Property 1 that we satisfy $(\alpha, \mathcal{G})$ d_σ-privacy by setting $\theta = \alpha/\Delta(\sigma_0 : \delta, \mathcal{G})$. Given alternative grouping $\mathcal{G}'$ with sensitivity $\Delta(\sigma_0 : \delta, \mathcal{G}')$, this same mechanism provides

$$\begin{aligned}\alpha' &= \frac{\theta}{\Delta(\sigma_0 : \delta, \mathcal{G}')} \\ &= \frac{\alpha/\Delta(\sigma_0 : \delta, \mathcal{G})}{\Delta(\sigma_0 : \delta, \mathcal{G}')} \\ &= \alpha\frac{\Delta(\sigma_0 : \delta, \mathcal{G}')}{\Delta(\sigma_0 : \delta, \mathcal{G})}\end{aligned}$$

□

5.10 Formal Utility Analysis of Alg. 1

Theorem 5.6. *For a given set $S \subset [n]$ and Hamming distance metric, $\delta_H(\cdot)$, Alg. 10 is (η, δ)-preserving for $\delta = \frac{1}{\psi(\theta, \delta_H)}\sum_{h=2k+1}^{n}(e^{-\theta\cdot h}\cdot c_h)$ where $k = \lceil(1-\eta)\cdot|S|\rceil$ and c_h is the number of permutations* *with hamming distance h from the reference permutation that do not preserve η% of S and is given by*

$$c_h = \sum_{j=k+1}^{\max(l_s,\lfloor h/2 \rfloor)} \binom{l_s}{j} \cdot \binom{n-l_s}{j} \cdot \left[\sum_{i=0}^{\min(l_s-j,h-2j)} \binom{l_s-j}{i} \right.$$

$$\left. \cdot \binom{i+j}{j} \cdot f(i,j) \cdot \binom{n-l_s-j}{h-2j-i} \cdot f(h-2j-i,j)! \right]$$

$$f(i,0) = !i, f(0,q) = q!$$

$$f(i,j) = \sum_{q=0}^{\min(i,j)} \left[\binom{i}{q} \cdot \binom{j}{j-q} \cdot j! \cdot f(i-q,q) \right]$$

$$l_s = |S|, k = (1-\eta) \cdot l_s, !n = \lfloor \frac{n!}{e} + \frac{1}{2} \rfloor$$

Proof. Let $l_s = |S|$ denote the size of the set S and $k = \lceil (1-\eta) \cdot l_S \rceil$ denote the maximum number of correct values that can be missing from S. Now, for a given permutation $\sigma \in \mathrm{S}_n$, let h denote its Hamming distance from the reference permutation σ_0, i.e, $h = \delta_H(\sigma, \sigma_0)$. This means that σ and σ_0 differ in h indices. Now, h can be analysed in the the following two cases,

Case I. $h \leq 2k+1$

For $(1-\eta)$ fraction of indices to be removed from S, we need at least $k+1$ indices from S to be replaced by $k+1$ values from outside S. This is clearly not possible for $h \leq 2k+1$. Hence, here $c_h = 0$.

Case II. $h > 2k$

For the following analysis we consider we treat the permutations as strings (multi-digit numbers are treated as a single string character). Now, Let $\mathbb{S}_{\sigma_0}$ denote the non-contiguous substring of σ_0 such that it consists of all the elements of S, i.e.,

$$|\mathbb{S}| = l_S \tag{8}$$

$$\forall i \in [l_S], \mathbb{S}_{\sigma_0}[i] \in S \tag{9}$$

Let $\mathbb{S}_\sigma$ denote the substring corresponding to the positions occupied by $\mathbb{S}_{\sigma_0}$ in σ. Formally,

$$|\mathbb{S}_\sigma| = l_S \tag{10}$$

$$\forall i \in [l_S], \mathbb{S}_{\sigma_0}[i] = \sigma(\sigma_0^{-1}(\mathbb{S}_{\sigma_0}[i])) \tag{11}$$

For example, for $\sigma_0 = (1\ 2\ 3\ 5\ 4\ 7\ 8\ 10\ 9\ 6)$, $\sigma = (1\ 3\ 2\ 7\ 8\ 5\ 4\ 6\ 10\ 9)$ and $S = \{2,4,5,8\}$, we have $\mathbb{S}_{\sigma_0} = 2548$ and $S_\sigma = 3784$ where $h = \delta_H(\sigma, \sigma_0) = 9$. Let $\{\mathbb{S}_\sigma\}$ denote the set of the elements of string $\mathbb{S}_\sigma$. Let A be the set of characters in $\mathbb{S}_\sigma$ such that they do not belong to S, i.e, $A = \{\mathbb{S}_\sigma[i] | \mathbb{S}_\sigma[i] \notin S, i \in [l_S]\}$. Let B be the set of characters in $\mathbb{S}_\sigma$ that belong to S but differ from $\mathbb{S}_{\sigma_0}$ in position, i.e., $B = \{\mathbb{S}_\sigma[i] | \mathbb{S}_\sigma[i] \in S, \mathbb{S}_\sigma[i] \neq \mathbb{S}_{\sigma_0}[i], i \in [l_S]\}$. Additionally, let $C = S - \{\mathbb{S}_\sigma\}$. For instance, in the above example, $A = \{3,7\}, B = \{4,8\}, C = \{2,5\}$. Now consider an initial arrangement of $p+m$ distinct objects that are subdivided into two types – p objects of Type A and m objects of Type B. Let $f(p,m)$ denote the number of permutations of these $p+m$ objects such that the m Type B objects can occupy any position but no object of Type A can occupy its original position. For example, for $f(p,0)$ this becomes the number of derangements [1] denoted as $!p = \lfloor \frac{p!}{e} + \frac{1}{2} \rfloor$. Therefore, $f(|B|,|A|)$ denotes the number of permutations of $\mathbb{S}_\sigma$ such that $\delta_H(\mathbb{S}_{\sigma_0}, \mathbb{S}_\sigma) = |A| + |B|$. This is because if elements of B are allowed to occupy their original position then this will reduce the Hamming distance.

Now, let $\bar{\mathbb{S}}_\sigma$ $(\bar{\mathbb{S}}_{\sigma_0})$ denote the substring left out after extracting from $\mathbb{S}_\sigma$ $(\mathbb{S}_{\sigma_0})$ from σ (σ_0). For example, $\bar{\mathbb{S}}_\sigma = 1256109$ and $\bar{\mathbb{S}}_{\sigma_0} = 1371096$ in the above example. Let D be the set of elements outside of S and A that occupy different positions in $\bar{\mathbb{S}}_\sigma$ and $\bar{\mathbb{S}}_{\sigma_0}$ (thereby contributing to the hamming distance), i.e., $D = \{\bar{\mathbb{S}}_{\sigma_0[i]} | \bar{\mathbb{S}}_{\sigma_0[i]} \notin S, \bar{\mathbb{S}}_{\sigma_0[i]} \neq \bar{\mathbb{S}}_{\sigma[i]}, i \in [n-l_S]\}$. For instance, in the above example $D = \{9,6,10\}$. Hence, $h = \delta_H(\sigma,\sigma_0) = |A| + |B| + |C| + |D|$ and clearly $f(|D|,|C|)$ represents the number of permutations of $\bar{\mathbb{S}}_\sigma$ such that $\delta_H(\bar{\mathbb{S}}_\sigma, \bar{\mathbb{S}}_{\sigma_0}) = |C| + |D|$. Finally, we have

$$c_h = \sum_{j=k+1}^{\max(l_s,\lfloor h/2 \rfloor)} \underbrace{\binom{l_s}{j}}_{\text{\# ways of selecting set } C} \cdot \underbrace{\binom{n-l_s}{j}}_{\text{\# ways of selecting set } A} \cdot \left[\right.$$

$$\sum_{i=0}^{\min(l_s-j,h-2j)} \underbrace{\binom{l_s-j}{i}}_{\text{\# ways of selecting set } B} \cdot f(i,j)$$

$$\left. \cdot \underbrace{\binom{n-l_s-j}{h-2j-i}}_{\text{\# ways of selecting set } D} \cdot f(h-2j-i,j) \right]$$

Now, for $f(i,j)$ let E be the set of original positions of Type A that are occupied by Type B objects in the resulting permutation. Additionally, let F be the set of the original positions of Type B objects that are still occupied by some Type B object. Clearly, Type B objects can occupy these $|E|+|F| = m$ in any way they like. However, the type A objects can only result in $f(p-q,q)$ permutations. Therefore, $f(p,m)$ is given by the following recursive function

$$f(p,0) = !p$$

$$f(0,m) = m!$$

$$f(p,m) = \sum_{q=0}^{\min p,m} \left(\underbrace{\binom{p}{q}}_{\text{\# ways of selecting set } E} \cdot \underbrace{\binom{m}{m-q}}_{\text{\# ways of selecting set } F} \right.$$

$$\left. \cdot m! \cdot f(p-q,q) \right)$$

Thus, the total probability of failure is given by

$$\delta = \frac{1}{\psi(\theta, \delta_H)} \sum_{h=2k+2}^{n} (e^{-\theta \cdot h} \cdot c_h) \tag{12}$$

□

6 EVALUATION

6.1 Additional Experimental Results/Explanation

The previous sections describe how our shuffling framework interpolates between standard LDP and uniform random shuffling. We now experimentally evaluate this asking the following two questions –

Q1. Does the Alg. 1 mechanism protect against realistic inference attacks?

Q2. How well can Alg. 1 tune a model's ability to learn trends within the shuffled data i.e. tune *data learnability*?

We evaluate on four datasets. We are not aware of any prior work that provides comparable local inferential privacy. Hence, we baseline our mechanism with the two extremes: standard LDP and uniform random shuffling. For concreteness, we detail our procedure with the *PUDF* dataset [2] (license), which comprises $n \approx 29$k psychiatric patient records from Texas. Each data owner's sensitive value x_i is their medical payment method, which is reflective of socioeconomic class (such as medicaid or charity). Public auxiliary information $t \in \mathcal{T}$ is the hospital's geolocation. Such information is used for understanding how payment methods (and payment amounts) vary from town to town for insurances in practice [20]. Uniform shuffling across Texas precludes such analyses. Standard LDP risks inference attacks, since patients attending hospitals in the same neighborhood have similar socioeconomic standing and use similar payment methods, allowing an adversary to correlate their noisy y_i's. To trade these off, we apply Alg. 1 with $d(\cdot)$ being distance (km) between hospitals, $\alpha = 4$ and Kendall's τ rank distance measure for permutations.

Our inference attack predicts DO$_i$'s x_i by taking a majority vote of the z_j values of the 25 data owners within r^* of t_i and who are most similar to DO$_i$ w.r.t some additional privileged auxiliary information $t_j^p \in \mathcal{T}_p$. For PUDF, this includes the 25 data owners who attended hospitals that are within r^* km of DO$_i$'s hospital, and are most similar in payment amount t_j^p. Using an $\epsilon = 2.5$ randomized response mechanism, we resample the LDP sequence $\mathbf{y}$ 50 times, and apply Alg. 1's chosen permutation to each, producing 50 $\mathbf{z}$'s. We then mount the majority vote attack on each x_i for each $\mathbf{z}$. If the attack on a given x_i is successful across $\geq 90\%$ of these LDP trials, we mark that data owner as vulnerable – although they randomize with LDP, there is a $\geq 90\%$ chance that a simple inference attack can recover their true value. We record the fraction of vulnerable data owners as ρ. We report 1-standard deviation error bars over 10 trials.

Additionally, we evaluate *data learnability* – how well the underlying statistics of the dataset are preserved across $\mathcal{T}$. For *PUDF*, this means training a model on the shuffled $\mathbf{z}$ to predict the distribution of payment methods used near, for instance, t_i = Houston for DO$_i$. For this, we train a calibrated model, : $\mathcal{T} \rightarrow \mathcal{D}_x$, on the shuffled outputs where $\mathcal{D}_x$ is the set of all distributions on the domain of sensitive attributes $\mathcal{X}$. We implement as a gradient boosted decision tree (GBDT) model [24] calibrated with Platt scaling [41]. For each location t_i, we treat the empirical distribution of x_i values within r^* as the ground truth distribution at t_i, denoted by $\mathcal{E}(t_i) \in \mathcal{D}_x$. Then, for each t_i, we measure the Total Variation error between the predicted and ground truth distributions $\text{TV}\big(\mathcal{E}(t_i),_r (t_i)\big)$. We then report $\lambda(r)$ – the average TV error for distributions predicted at each $t_i \in \mathbf{t}$ normalized by the TV error of naively guessing the uniform distribution at each t_i. With standard LDP, this task can be performed relatively well at the risk of inference attacks. With uniformly shuffled data, it is impossible to make geographically localized predictions unless the distribution of payment methods is identical in every Texas locale.

We additionally perform the above experiments on the following three datasets

- *Adult* [17]. This dataset is derived from the 1994 Census and has $\approx$ 33K records. Whether DO$_i$'s annual income is $\geq$ 50k is considered private, $\mathcal{X} = \{\geq 50k, < 50k\}$. $\mathcal{T} = [17, 90]$ is age and $\mathcal{T_P}$ is the individual's marriage status.
- *Twitch* [42]. This dataset, gathered from the *Twitch* social media platform, includes a graph of $\approx 9K$ edges (mutual friendships) along with node features. The user's history of explicit language is private $\mathcal{X} = \{0, 1\}$. $\mathcal{T}$ is a user's mutual friendships, i.e. t_i is the i'th row of the graph's adjacency matrix. We do not have any $\mathcal{T_P}$ here, and select the 25 nearest neighbors randomly.
- *Syn.* This is a synthetic dataset of size $20K$ which can be classified at three granularities – 8-way, 4-way and 2-way. The eight color labels are private $\mathcal{X} = [8]$; the 2D-positions are public $\mathcal{T} = \mathbb{R}^2$. For learnability, we measure the accuracy of 8-way, 4-way and 2-way GBDT models trained on $\mathbf{z}$ on an equal sized test set at each r.

Experimental Results.

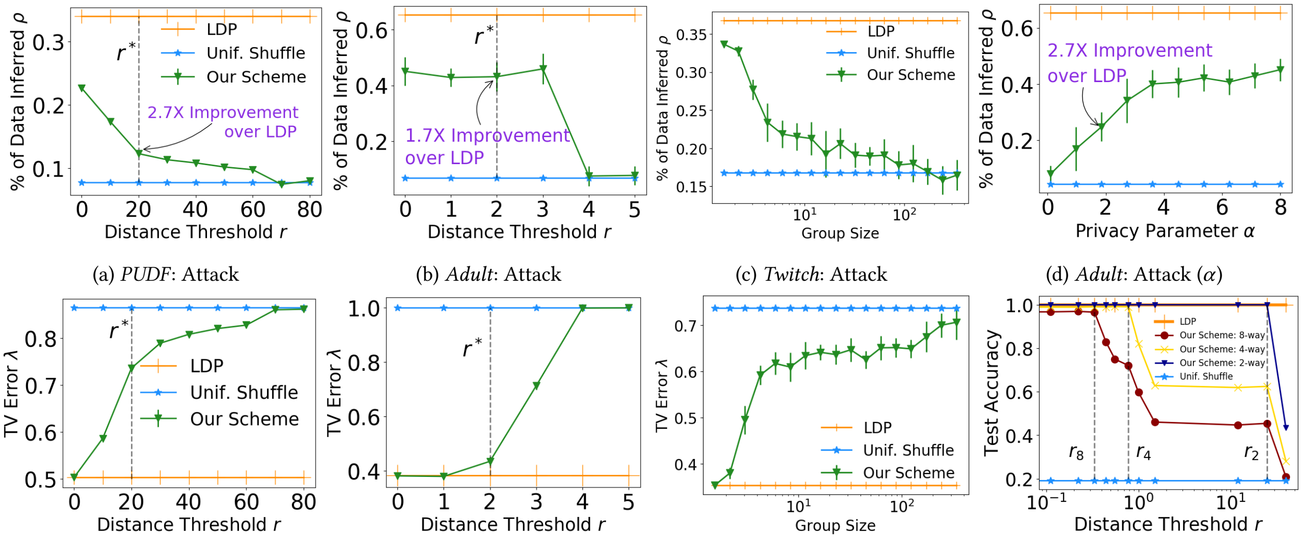
Q1. Our formal guarantee on the inferential privacy loss (Thm. 5.3) is described w.r.t to a 'strong' adversary (with access to $\{y_{G_i}\}, \mathrm{y}_{\overline{G_i}}$). Here, we test how well does our proposed scheme (Alg. 1) protect against inference attacks on real-world datasets without any such assumptions. Additionally, to make our attack more realistic, the adversary has access to extra privileged auxiliary information $\mathcal{T_P}$ which is *not used* by Alg. 10. Fig. 5a→ 5c show that our scheme significantly reduces the attack efficacy. For instance, ρ is reduced by 2.7X at the attack distance threshold r^* for *PUDF*. Additionally, ρ for our scheme varies from that of LDP[2] (minimum privacy) to uniform shuffle (maximum privacy) with increasing r (equivalently group size as in Fig. 5c) thereby spanning the entire privacy spectrum. As expected, ρ decreases with decreasing privacy parameter α (Fig. 5d).

Q2. Fig.5e → 5g show that λ varies from that of LDP (maximum learnability) to that of uniform shuffle (minimum learnability) with increasing r (equivalently, group size), thereby providing tunability. Interestingly, for *Adult* our scheme reduces ρ by 1.7X at the same λ as that of LDP for $r = 1$ (Fig. 5f). Fig. 5h shows that the distance threshold r defines the granularity at which the data can be classified. LDP allows 8-way classification while uniform shuffling allows none. The granularity of classification can be tuned by our scheme – r_8, r_4 and r_2 mark the thresholds for 8-way, 4-way and 2-way classifications, respectively.

6.1.1 *Evaluation of (η, δ)-preservation.*

In this section, we evaluate the characteristics of the (η, δ)-preservation for Kendall's τ distance $\mathfrak{d}_\tau(\cdot, \cdot)$.

[2] Our scheme gives lower ρ than LDP at $r = 0$ because the resulting groups are non-singletons. For instance, for PUDF, G_i includes all individuals with the same zipcode as DO$_i$.



(a) *PUDF*: Attack

(b) *Adult*: Attack

(c) *Twitch*: Attack

(d) *Adult*: Attack (α)

(e) *PUDF*: Learnability

(f) *Adult*: Learnability

(g) *Twitch*: Learnability

(h) *Syn*: Learnability

Figure 5: Our scheme interpolates between standard LDP (orange line) and uniform shuffling (blue line) in both privacy and data learnability. All plots increase group size along x-axis (except (d)). (a) → (c): The fraction of participants vulnerable to an inferential attack. (d): Attack success with varying α for a fixed r. (e) → (g): The accuracy of a calibration model trained on $\mathbf{z}$ predicting the distribution of LDP outputs at any point $t \in \mathcal{T}$, such as the distribution of medical insurance types used specifically in the Houston area (not possible when uniformly shuffling across Texas). (h): Test accuracy of a classifier trained on $\mathbf{z}$ for a synthetic dataset based on the crescents synthetic distribution.

Each sweep of Fig. 6 fixes $\delta = 0.01$, and observes η. We consider a dataset of size $n = 10K$ and a subset S of size l_S corresponding to the indices in the middle of the reference permutation σ_0 (the actual value of the reference permutation is not significant for measuring preservation). For the rest of the discussion, we denote the width of a permutation by ω for notational brevity. For each value of the independent axis, we generate 50 trials of the permutation σ from a Mallows model with the appropriate θ (given the ω and α parameters). We then report the largest η (fraction of subset preserved) that at least 99% of trials satisfy.

In Fig. 6a, we see that preservation is highest for higher α and increases gradually with declining width ω and increasing subset size l_s.

Fig. 6b demonstrates that preservation declines with increasing width. Δ increases quadratically with width ω for $\mathfrak{d}_\tau$, resulting in declining θ and increasing randomness. We also see that larger subset sizes result in a more gradual decline in η. This is due to the fact that the worst-case preservation (uniform random shuffling) is better for larger subsets. i.e. we cannot do worse than 80% preservation for a subset that is 80% of indices.

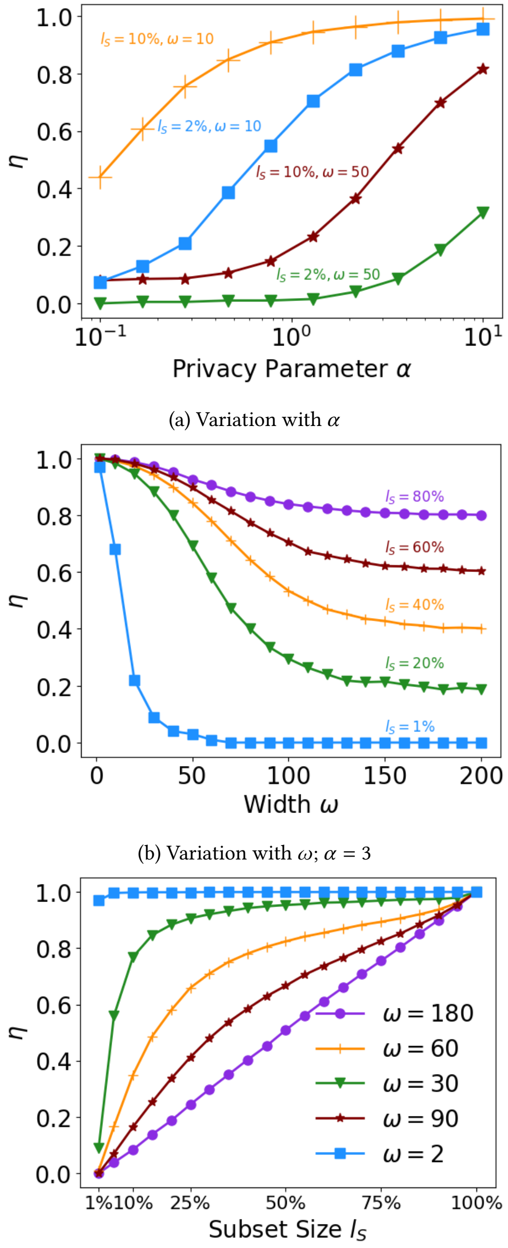
Finally, Fig. 6c demonstrates how preservation grows rapidly with increasing subset size. For large widths, we are nearly uniformly randomly permuting, so preservation will equal the size of the subset relative to the dataset size. For smaller widths, we see that preservation offers diminishing returns as we grow subset size past some critical l_s. For $\omega = 30$, we see that subset sizes much larger than a quarter of the dataset gain little in preservation.

6.2 Related Work

In this section, we discuss the relevant existing work.

The anonymization of noisy responses to improve differential privacy was first proposed by Bittau et al. [10] who proposed a principled system architecture for shuffling. This model was formally studied later in [12, 21]. Erlingsson et al. [21] showed that for arbitrary ϵ-LDP randomizers, random shuffling results in privacy amplification. Cheu et al. [12] formally defined the shuffle DP model and analyzed the privacy guarantees of the binary randomized response in this model. The shuffle DP model differs from our approach in two ways. First, it focuses completely on the DP guarantee. The privacy amplification is manifested in the from of a lower ϵ (roughly a factor of $\sqrt{n}$) when viewed in an alternative DP model known as the central DP model. [6, 7, 10, 12, 21, 23]. However, our result caters to local inferential privacy. Second, the shuffle model involves an uniform random shuffling of the entire dataset. In contrast, our approach the granularity at which the data is shuffled is tunable which delineates a threshold for the learnability of the data.

A steady line of work has sudied the inferential privacy setting [14, 18, 28, 32, 35, 46]. Kifer et al. [35] formally studied privacy degradation in the face of data correlations and later proposed a privacy framework, Pufferfish [31, 36, 43], for analyzing inferential privacy. Subsequently, several other privacy definitions have also been proposed for the inferential privacy setting [8, 11, 38, 50, 52]. For instance, Gehrke et al. proposed a zero-knowledge privacy [25, 26] which is based on simulation semantics. Bhaskar et al. proposed noiseless privacy [9, 30] by restricting the set of prior distributions that the adversary may have access to. A recent work by Zhang et al. proposes attribute privacy [51] which focuses on the sensitive properties of a whole dataset. In another recent work, Ligett et al. study a relaxation of DP that accounts for mechanisms that leak some additional, bounded information about the database [37]. Some early work in local inferential privacy include profile-based privacy [27] by Gehmke et al. where the problem setting comes with a graph of data generating distributions, whose edges encode sensitive pairs of distributions that should be made indistinguishable. In another work by Kawamoto et al., the authors propose distribution privacy [33] – local differential privacy for probability distributions. The major difference between our work and prior research is that we provide local inferential privacy through a new angle – data shuffling.



(a) Variation with α

(b) Variation with ω; $\alpha = 3$

(c) Variation with l_S; $\alpha = 3$

Figure 6: (η, δ)-Preservation Analysis