Frequency Estimation of Evolving Data Under Local Differential Privacy

Héber H. Arcolezi*
Inria and École Polytechnique (IPP)
heber.hwang-arcolezi@inria.fr

Carlos Pinzón*
Inria and École Polytechnique (IPP)
carlos.pinzon@inria.fr

Catuscia Palamidessi
Inria and École Polytechnique (IPP)
catuscia@lix.polytechnique.fr

Sébastien Gambs
Université du Québec à Montréal, UQAM
gambs.sebastien@uqam.ca

ABSTRACT

Collecting and analyzing evolving longitudinal data has become a common practice. One possible approach to protect the users' privacy in this context is to use local differential privacy (LDP) protocols, which ensure the privacy protection of all users even in the case of a breach or data misuse. Existing LDP data collection protocols such as Google's RAPPOR [23] and Microsoft's *d*BitFlipPM [13] can have longitudinal privacy linear to the domain size k, which is excessive for large domains, such as Internet domains. To solve this issue, in this paper we introduce a new LDP data collection protocol for longitudinal frequency monitoring named LOngitudinal LOcal HAshing (LOLOHA) with formal privacy guarantees. In addition, the privacy-utility trade-off of our protocol is only linear with respect to a reduced domain size $2 \leq g \ll k$. LOLOHA combines a domain reduction approach via local hashing with double randomization to minimize the privacy leakage incurred by data updates. As demonstrated by our theoretical analysis as well as our experimental evaluation, LOLOHA achieves a utility competitive to current state-of-the-art protocols, while substantially minimizing the longitudinal privacy budget consumption by up to k/g orders of magnitude.

1 INTRODUCTION

Estimating histograms of evolving categorical data is a fundamental task in data analysis and data mining that requires collecting and processing data in a continuous manner. A typical instance of such a problem is the online monitoring performed on software applications [10], for example for error reporting [25], to find commonly typed emojis [37], as well as to measure the users' system usage statistics [13]. However, the data collected can contain sensitive information such as location, health information, preferred webpage, etc. Thus, the direct collection and storage of users' raw data on a centralized server should be avoided to preserve their privacy. To address this issue, recent works have proposed several mechanisms satisfying Differential Privacy (DP) [18–20] in the distributed setting in which an individual can directly randomize her own profile locally, referred to as Local DP (LDP) [16, 17, 30].

One of the strengths of LDP is its simple trust model: since each user perturbs her data locally, user privacy is protected even if the server is malicious. For instance, some big tech companies have chosen to operate some of their applications in the local model, reporting the implementation of LDP protocols to collect statistics on well-known systems such as Google Chrome browser [23], Apple iOS/macOS [37], and Windows 10 operating system [13]).

Existing LDP protocols for frequency estimation typically focus on one-time computation [2, 8, 9, 12, 24, 28, 29, 40]. However, considering both evolving data and the continuous monitoring together, pose a significant challenge under LDP guarantees. For instance, the naïve solution in which an LDP computation is repeated, will quickly increase the privacy loss leading to large values of ϵ due to the sequential composition theorem in DP [20]. To tackle this issue, most state-of-the-art solutions relies on **memoization** [4, 5, 13, 21, 23].

Initially proposed by Erlingsson, Pihur, and Korolova [23], the memoization-based RAPPOR protocol allows a user to memorize randomized versions of their true data and consistently reuse it when the same true value occurs. In addition, to improve privacy (*e.g.*, minimize data change detection and/or tracking), the RAPPOR [23] protocol applies a second round of sanitization to the memoized value. However, the longitudinal privacy protection of RAPPOR only works if the underlying true value never or rarely changes (or changes in an uncorrelated fashion), which is unrealistic for evolving data (*e.g.*, the number of seconds an application is used) as the privacy loss is proportional to the number of data changes, *i.e.*, the domain size k in the worst-case.

To address this issue, Ding, Kulkarni, and Yekhanin [13] have proposed a new LDP protocol named *d*BitFlipPM that improved memoization by mapping several values to the same randomized value. More precisely, *d*BitFlipPM partitions the original values into $b \leq k$ buckets (*e.g.*, with equal widths), which allows close values to be mapped to the same bucket. Afterwards, each user only samples $d \leq b$ buckets to minimize the number of bits to be randomized. Note that these two steps contributes to the information loss. Another limitation of *d*BitFlipPM is the possibility of detecting data changes [42] on the fly since the true value will fall in a different bucket, there will be a higher probability of changing the randomization of the d bits. Even if this only indicates that the user's value has changed, not what it was or is [13, 23], there are still some privacy implications with respect to the type of inference an adversary can perform, especially if there are correlation patterns to be exploited [33, 36]. Finally, *d*BitFlipPM's privacy loss can still be proportional to the number of bucket changes, *i.e.*, the new domain size b in the worst case.

A different line of work has taken into account the infrequent data changes on the user side, hereafter referred to as **data change-based** [22, 27, 35, 42]. For instance, Joseph *et al.* [27] have proposed a new LDP protocol THRESH for monitoring statistics (*e.g.*, frequency) based on two sub-routines: voting and estimation, which requires splitting the privacy budget. The main idea of THRESH is to update through voting the global estimate only when it becomes sufficiently inaccurate. However, privacy budget

* These are co-first authors that contributed equally to this work.

splitting under LDP guarantees is sub-optimal [3–5, 21, 34, 39, 40], which negatively impacts the data utility. Moreover, the authors in [22, 35] proposed the sanitization and report of data changes for frequency monitoring by assuming a limited number of data changes and longitudinal Boolean data, though it can be extend to larger domain. This leads to an accuracy that decays linearly (or sub-linearly) in the number of data changes. Finally, in a recent work, Xue *et al.* [42] have proposed a new LDP protocol DDRM (Dynamic Difference Report Mechanism) based on difference trees. However, DDRM assumes that the user's private sequence exhibit continuity (*i.e.*, do not fluctuate significantly) and was mainly designed for longitudinal Boolean data. Besides, DDRM requires a privacy budget allocation scheme that depends on the number of data collections as well as to split the privacy budget when extending to a larger domain (*i.e.*, sub-optimal).

Main contributions. In this paper, we address the limitations of memoization-based protocols [5, 13, 21, 23] without imposing any restriction on the number of data changes and/or on the number of data collections as in data change-based protocols [22, 27, 35, 42]. More precisely, we propose a novel LDP protocol with formal privacy guarantees for longitudinal frequency estimation of evolving counter (or categorical) data.

Our protocol, hereafter named LOngitudinal LOcal HAshing (LOLOHA), combines a domain reduction approach through local hashing [9, 40] with the memoization solution of RAPPOR using two rounds of sanitization [5, 23]. The main strength of LOLOHA is that the longitudinal privacy-utility trade-off is linear only on the new (reduced) domain size g, in which $2 \leq g \ll k$ is a tunable hyper-parameter. This way, the worst-case longitudinal privacy loss of LOLOHA has a significant k/g or b/g decrease factor in comparison with RAPPOR and dBitFlipPM, respectively.

Indeed, LOLOHA can be tuned for strong longitudinal privacy by selecting $g = 2$ (BiLOLOHA protocol). To maximize LOLOHA's utility, we also find the optimal g value (OLOLOHA protocol). Experimental evaluations demonstrate the effectiveness of LOLOHA with respect to the quality of frequency estimates, in addition to substantially minimizing the longitudinal privacy loss.

We also show why LDP is generally impossible to achieve when data is longitudinal, which motivates a definition of privacy that better suits the longitudinal scenario. This is in opposition with the common and mathematically equivalent path in the literature of claiming a protocol to be LDP but assuming that the evolving data is uncorrelated or constant in time, which we believe not be realistic in real-life deployments.

In summary, the main contributions of this paper are threefold:

- We propose the LOLOHA protocol for longitudinal frequency monitoring under LDP guarantees.
- We prove the longitudinal privacy and accuracy guarantees of LOLOHA through theoretical analysis and compare it to existing protocols.
- We show the performance of LOLOHA numerically and experimentally, using both real-world and synthetic datasets.

Outline. The remainder of this paper is organized as follows. First, in Section 2, we provide the problem definition and review LDP and existing longitudinal LDP protocols. Next, we present and analyze our LOLOHA protocols in Section 3. In Section 4, we give a theoretical comparison of LOLOHA and state-of-the-art LDP protocols before presenting and interpreting the experimental results in Section 5. Finally, in Section 6, we review related work before concluding with future perspectives in Section 7.

2 PRELIMINARIES

In this section, we present the problem considered and we review the LDP privacy model and relevant protocols.

Notation. For denoting sets, we will use italic uppercase letters V, U, etc, and we write $[1..n] = \{1, \ldots, n\}$. For a vector $\mathbf{x}$ (bold lowercase letters), $\mathbf{x}_i$ represents the value of its i-th coordinate. Finally, we denote randomized protocols as $\mathcal{M}$.

2.1 Problem Statement

We consider the situation in which a server collects data from a distributed group of users while requiring the protection of privacy for each user, through LDP. The server collects sanitized data over time from each member of the group with respect to a fixed discrete random variable (*e.g.*, daily usage of a mobile application). Its objective is to estimate the true frequencies, or histograms, of the random variable as well as its evolution over time. We aim to provide the server with an optimized combination of two algorithms: one for the users, who must sanitize locally their data before sending it, and another for the server, which wants to aggregate data and perform the estimation accurately.

Formally, there are n users $U = \{u_1, \ldots, u_n\}$ and a random variable taking values in a set V of size k with true frequencies $\{f(v)\}_{v \in V}$, which may vary over time. Each user $u \in U$, holds a private sequence of values $\mathbf{v}^{(u)} = \left[v_1^{(u)}, v_2^{(u)}, \ldots, v_\tau^{(u)}\right]$, in which $v_t^{(u)}$ represents the discrete value $v \in V$ of user u at time step $t \in [1..\tau]$. At each time step t, upon collecting the sanitized values of all n users, the server will estimate a k-bins histogram $\{\hat{f}(v)\}_{v \in V}$ in a way that minimizes the *Mean Squared Error* (MSE) with respect to $\{f(v)\}_{v \in V}$. For all the algorithms presented hereafter, the estimation $\hat{f}(v)$ is unbiased (*i.e.*, $\mathbb{E}(\hat{f}(v)) = f(v)$). As a consequence, the MSE is equivalent to the variance as:

$$\text{MSE} = \frac{1}{|V|} \sum_{v \in V} \mathbb{E}\left[\left(\hat{f}(v) - f(v)\right)^2\right] = \frac{1}{|V|} \sum_{v \in V} \mathbb{V}[\hat{f}(v)].$$

2.2 Local Differential Privacy

Privacy model. In this paper, we use LDP (Local Differential Privacy) [16, 17, 30] as the privacy model considered, which is formally defined as follows.

Definition 2.1 (ϵ-Local Differential Privacy). A randomized algorithm $\mathcal{M}$ satisfies ϵ-local-differential-privacy (ϵ-LDP), where $\epsilon > 0$, if for any pair of input values $v_1, v_2 \in \mathit{Domain}(\mathcal{M})$ and any possible output x' of $\mathcal{M}$:

$$\Pr[\mathcal{M}(v_1) = x'] \leq e^{\epsilon} \cdot \Pr[\mathcal{M}(v_2) = x'].$$

In essence, LDP guarantees that it is unlikely for the data aggregator to reconstruct the input data. The privacy loss ϵ controls the privacy-utility trade-off for which lower values of ϵ result in tighter privacy protection. Similar to central DP, LDP also has several fundamental properties, such as robustness to post-processing and composition [20].

Proposition 2.2 (Post-Processing [20]). *If $\mathcal{M}$ is ϵ-LDP, then $f(\mathcal{M})$ is also ϵ-LDP for any function f.*

Proposition 2.3 (Sequential Composition [20]). *Let $\mathcal{M}_t$ be ϵ_t-LDP mechanism, for $t \in [\tau]$. Then, the sequence of outputs $[\mathcal{M}_1(v), \ldots, \mathcal{M}_\tau(v)]$ is $\sum_{t=1}^{\tau} \epsilon_t$-LDP. Moreover, if $\mathcal{M}$ is an ϵ-LDP mechanism and $\mathbf{v}$ is a finite sequence of k values, then the sequence of outputs $[\mathcal{M}(v_1), \ldots, \mathcal{M}(v_k)]$ is $k\epsilon$-LDP.*

2.3 LDP Frequency Estimation Protocols

In this section, we review five state-of-the-art LDP frequency estimation protocols, which are often used as building blocks for more complex tasks (*e.g.*, heavy hitter estimation [8, 9], machine learning [32], and private frequency monitoring [5, 13, 23]).

2.3.1 Generalized Randomized Response (GRR). The GRR [28, 29] protocol generalizes the Randomized Response (RR) technique proposed by Warner [41] for $k \geq 2$ while satisfying LDP.

Fix a parameter $\epsilon > 0$ and let $p \coloneqq \frac{e^\epsilon}{e^\epsilon + k - 1} \in (0, 1)$ in which $k = |V|$. For each $v \in V$, let $\eta_{\neq v} \in V$ be a uniform (*i.e.*, exogenous noise) random variable over $V \setminus \{v\}$. We let $\mathcal{M}_{\text{GRR}} : V \to V$ be the random variable given by:

$$\mathcal{M}_{\text{GRR}}(v; \epsilon) \coloneqq \begin{cases} v, & \text{w.p. } p \\ \eta_{\neq v}, & \text{w.p. } 1 - p. \end{cases}$$

This protocol satisfies ϵ-LDP, because $\frac{p}{q} = e^\epsilon$ [28], in which $q \coloneqq {}^{(1-p)}/_{(k-1)}$ determines the probability of the response being any fixed noise value different of v. To estimate the normalized frequency of $v \in V$, one counts how many times v is reported, expressed as $C(v)$, and then computes:

$$\hat{f}(v) = \frac{C(v) - nq}{n(p - q)}, \tag{1}$$

in which n is the total number of users. In [40], it was proven that Eq. (1) is an unbiased estimator (*i.e.*, $\mathbb{E}(\hat{f}(v)) = f(v)$).

2.3.2 Local Hashing (LH). LH protocols [40] can handle a large domain size k by first using hash functions to map an input value to a smaller domain of size $g \geq 2$ (typically $g \ll k$), and then applying GRR to the hashed value.

Fix $\epsilon > 0$ and let $\mathcal{M}_{\text{GRR}} : [1..g] \to [1..g]$ be the GRR mechanism with parameter ϵ and assuming the input-output domain to be $[1..g]$ instead of V, so that the size is g instead of k. In local hashing, each user selects at random a hashing function H from a family of universal hash functions, and reports the pair $\langle \text{H}, \mathcal{M}_{\text{GRR}}(x; \epsilon) \rangle$, in which $x = \text{H}(v)$.

The hash values will remain unchanged with probability $p = \frac{e^\epsilon}{e^\epsilon + g - 1}$ and switch to any different fixed value in $[1..g]$ with probability $q = \frac{1}{e^\epsilon + g - 1}$. This means that for each hash value $x \in [1..g]$, it holds that:

$$\Pr[\mathcal{M}_{\text{GRR}}(\text{H}(v); \epsilon) = x] = \begin{cases} p, & \text{if } x = \text{H}(v) \\ q, & \text{otherwise.} \end{cases}$$

Let $\langle \text{H}^u, x^u \rangle$ be the report from user $u \in U$. The server can obtain the unbiased estimation of $v \in V$, with Eq. (1) by setting $q = \frac{1}{g}$ and $C(v) = |\{u \in U \mid \text{H}^u(v) = x^u\}|$ [40].

The authors in [40] describe two LH protocols that differ on how g is selected: (1) Binary LH (BLH) that selects $g = 2$ and (2) Optimal LH (OLH) that selects $g = \lfloor e^\epsilon + 1 \rceil$ (rounded to closest integer).

2.3.3 Unary Encoding (UE). UE protocols interpret the user's input $v \in V$, as a one-hot k-dimensional vector. More precisely, $\mathbf{x} = \text{UE}(v)$ is a binary vector with only the bit at the position corresponding to v set to 1 and the other bits set to 0. The perturbation function of UE protocols randomizes the bits from $\mathbf{x}$ independently with probabilities:

$$\forall i \in [k]: \quad \Pr[\mathbf{x}'_i = 1] = \begin{cases} p, & \text{if } \mathbf{x}_i = 1, \\ q, & \text{if } \mathbf{x}_i = 0. \end{cases} \tag{2}$$

Afterwards, the client sends $\mathbf{x}'$ to the server. The authors in [40] describe two UE protocols that depend on the parameters p and q in Eq. (2): (1) Symmetric UE (SUE) [23], which selects $p = \frac{e^{\epsilon/2}}{e^{\epsilon/2}+1}$ and $q = \frac{1}{e^{\epsilon/2}+1}$ such that $p + q = 1$, and (2) Optimal UE (OUE), which selects $p = \frac{1}{2}$ and $q = \frac{1}{e^\epsilon+1}$.

The estimation method used in Eq. (1) applies equally to both UE protocols, in which $C(v)$ represents the number of times the bit corresponding to v has been reported. Last, both SUE and OUE protocols satisfy ϵ-LDP for $\epsilon = ln\left(\frac{p(1-q)}{(1-p)q}\right)$ [40].

2.4 Existing Longitudinal LDP Frequency Estimation Protocols

For privately monitoring the frequency of values of a population, the simplest way is that each user adds independent fresh noise to v in each data collection $t \in [1..\tau]$ following one of the LDP protocols described in the previous section. However, this solution is vulnerable to "averaging attacks" in which an adversary can estimate the true value from observing multiple randomized versions of it. To avoid this averaging attack, the memoization approach [23] was designed to enable longitudinal collections through memorizing a randomized version of the true value v and consistently reusing it [4, 13] or reusing it as the input to a second round of sanitization (*i.e.*, chaining two LDP protocols) [5, 21, 23]. The next four subsections describe state-of-the-art memoization-based protocols.

2.4.1 RAPPOR Protocol. The utility-oriented version of RAPPOR [23] is based on the SUE protocol, which encodes the user's input $v \in V$ as a k-dimensional bit-vector and randomizes each bit independently. More specifically, for each value $v \in V$, the user encodes $\mathbf{x} = \text{UE}(v)$ and randomizes $\mathbf{x}$ as follows:

Step 1. Permanent RR (PRR): Memoize $\mathbf{x}'$ such that:

$$\forall i \in [k]: \quad \Pr[\mathbf{x}'_i = 1] = \begin{cases} p_1 = \frac{e^{\epsilon_\infty/2}}{e^{\epsilon_\infty/2}+1}, & \text{if } \mathbf{x}_i = 1, \\ q_1 = \frac{1}{e^{\epsilon_\infty/2}+1}, & \text{if } \mathbf{x}_i = 0, \end{cases}$$

in which p_1 and q_1 control the level of longitudinal ϵ_∞-LDP for $\epsilon_\infty = ln\left(\frac{p_1(1-q_1)}{(1-p_1)q_1}\right)$ [23]. This step is carried out only once for each value $v \in V$ that the user has. Thus, the value $\mathbf{x}'$ shall be reused as the basis for all future reports of v.

Step 2. Instantaneous RR (IRR): Generate $\mathbf{x}''$ such that:

$$\forall i \in [k]: \quad \Pr[\mathbf{x}''_i = 1] = \begin{cases} p_2, & \text{if } \mathbf{x}'_i = 1, \\ q_2, & \text{if } \mathbf{x}'_i = 0. \end{cases}$$

This second step is carried out each time $t \in [1..\tau]$ a user report the value v. RAPPOR's deployment selected $p_2 = 0.75$ and $q_2 = 0.25$ [23, 40] (*i.e.*, also symmetric). The RAPPOR protocol that chains two SUE protocols is referred to as L-SUE in [5, 6]. We provide the calculation of parameters p_2 and q_2 in the repository [1]. Note that ϵ_∞ corresponds to an upper bound for each value v as $t \to \infty$. The privacy guarantees of the IRR step degrade according to the number of reports $t \in [1..\tau]$ [21, 23].

With two rounds of sanitization, each consisting of an LDP protocol parametrized with p, q, the unbiased estimator in Eq. (1) is now extended to [5, 23]:

$$\hat{f}_L(v) = \frac{\frac{C(v) - nq_2}{(p_2 - q_2)} - nq_1}{n(p_1 - q_1)} = \frac{C(v) - nq_1(p_2 - q_2) - nq_2}{n(p_1 - q_1)(p_2 - q_2)}, \tag{3}$$

in which p_1 and q_1 are the parameters of the LDP protocol used in the first step while p_2 and q_2 are the parameters of the LDP protocol used in the second step.

In [5], it was proven that Eq. (3) is an unbiased estimator (*i.e.*, $\mathbb{E}(\hat{f}_L(v)) = f(v)$) and that for any value $v \in V$, the variance $\mathbb{V}$ of the estimator $\hat{f}_L(v)$ in Eq. (3) is:

$$\mathbb{V}[\hat{f}_L(v)] = \frac{\gamma(1-\gamma)}{n(p_1-q_1)^2(p_2-q_2)^2}, \text{ where}$$
$$\gamma = f(v)\,(2p_1p_2 - 2p_1q_2 + 2q_2 - 1) + p_2q_1 + q_2(1-q_1). \tag{4}$$

In this paper, we will use the *approximate variance* $\mathbb{V}^*$, in which $f(v) = 0$ in Eq. (4), which gives:

$$\mathbb{V}^*\left[\hat{f}_L(v)\right] = \frac{(p_2q_1 - q_2\,(q_1-1))\,(-p_2q_1 + q_2\,(q_1-1) + 1)}{n(p_1-q_1)^2(p_2-q_2)^2}. \tag{5}$$

Therefore, one can obtain the RAPPOR approximate variance $\mathbb{V}^*[\hat{f}_{\text{RAPPOR}}(v)]$ by replacing the resulting p_1, q_1, p_2, q_2 parameters into Eq. (5).

2.4.2 Optimized Longitudinal UE Protocol. The authors in [5] analyzed all four combinations between OUE and SUE in both PRR and IRR steps. The optimized protocol named L-OSUE chains the OUE protocol (PRR step) and the SUE protocol (IRR step). Thus, for each value $v \in V$, the user encodes $\mathbf{x} = \mathsf{UE}(v)$ and randomizes $\mathbf{x}$ as follows:

Step 1. PRR: Memoize $\mathbf{x}'$ such that:

$$\forall i \in [k]: \quad \Pr[\mathbf{x}'_i = 1] = \begin{cases} p_1 = \frac{1}{2}, & \text{if } \mathbf{x}_i = 1, \\ q_1 = \frac{1}{e^{\epsilon_\infty}+1}, & \text{if } \mathbf{x}_i = 0, \end{cases}$$

in which p_1 and q_1 control the level of longitudinal ϵ_∞-LDP as $e^{\epsilon_\infty} = \frac{p_1(1-q_1)}{q_1(q-p_1)}$ [5, 23]. The value $\mathbf{x}'$ shall be reused as the basis for all future reports when the real value is v.

Step 2. IRR: Generate $\mathbf{x}''$ such that:

$$\forall i \in [k]: \quad \Pr[\mathbf{x}''_i = 1] = \begin{cases} p_2, & \text{if } \mathbf{x}'_i = 1, \\ q_2 = 1 - p_2, & \text{if } \mathbf{x}'_i = 0. \end{cases}$$

in which $p_2 = \frac{e^{\epsilon_\infty}e^{\epsilon_1}-1}{e^{\epsilon_\infty}-e^{\epsilon_1}+e^{\epsilon_\infty+\epsilon_1}-1}$ and $\mathbf{x}''$ is the report to be sent to the server. Let $p_s = \Pr[\mathbf{x}''_i = 1|\mathbf{x}_i = 1] = p_1p_2 + (1-p_1)q_2$ and $q_s = \Pr[\mathbf{x}''_i = 1|\mathbf{x}_i = 0] = q_1p_2 + (1-q_1)q_2$. For the first report, the L-OSUE protocol satisfies ϵ_1-LDP as $e^{\epsilon_1} = \frac{p_s(1-q_s)}{(1-p_s)q_s}$ [5, 23].

Similar to RAPPOR, the estimated frequency $\hat{f}_{\text{L-OSUE}}(v)$ that a value $v \in V$ occurs, can be computed using Eq. (3). One can also obtain the L-OSUE approximate variance $\mathbb{V}^*[\hat{f}_{\text{L-OSUE}}(v)]$ by replacing the resulting p_1, q_1, p_2, q_2 parameters into Eq. (5).

2.4.3 Longitudinal GRR (L-GRR). The L-GRR [5] protocol chains GRR in both PRR and IRR steps. Therefore, for each value $v \in V$, the user randomizes v as follows:

Step 1. PRR: Memoize x' such that:

$$x' = \begin{cases} v, & \text{w.p. } p_1 = \frac{e^{\epsilon_\infty}}{e^{\epsilon_\infty}+k-1}, \\ \tilde{v} \in V \setminus \{v\}, & \text{w.p. } q_1 = \frac{1-p_1}{k-1}, \end{cases}$$

in which p_1 and q_1 control the level of longitudinal ϵ_∞-LDP as $e^{\epsilon_\infty} = \frac{p_1}{q_1}$ [5, 28]. The value x' shall be reused as the basis for all future reports on the real value v.

Step 2. IRR: Generate a report x'' such that:

$$x'' = \begin{cases} x', & \text{w.p. } p_2, \\ \tilde{x} \in V \setminus \{x'\}, & \text{w.p. } q_2 = \frac{1-p_2}{k-1}, \end{cases}$$

in which $p_2 = \frac{e^{\epsilon_\infty+\epsilon_1}-1}{-ke^{\epsilon_1}+(k-1)e^{\epsilon_\infty}+e^{\epsilon_1}+e^{\epsilon_1+\epsilon_\infty}-1}$ and x'' is the report to be sent to the server. Let $p_s = \Pr[x'' = v|v] = p_1p_2 + q_1q_2$ and $q_s = \Pr[x'' = v|\tilde{v} \in V \setminus \{v\}] = p_1q_2 + q_1p_2$. For the first report, the L-GRR protocol satisfies ϵ_1-LDP since $e^{\epsilon_1} = \frac{p_s}{q_s}$ [5].

The estimated frequency $\hat{f}_{\text{L-GRR}}(v)$ that a value v occurs can also be obtained using Eq. (3). Besides, one can compute the L-GRR approximate variance $\mathbb{V}^*[\hat{f}_{\text{L-GRR}}(v)]$ by replacing the resulting p_1, q_1, p_2, q_2 parameters into Eq. (5).

2.4.4 dBitFlipPM Protocol. The *d*BitFlipPM [13] protocol was proposed to improve the memoization solution of RAPPOR [23] by mapping several true values to the same noisy response at the cost of losing information due to generalization. This is done by first partitioning the original domain V into b buckets (*i.e.*, new domain size $2 \le b \le k$) using a function $\mathsf{bucket}: V \to [1..b]$, such that close values will fall into the same bucket. Next, each user randomly draws d bucket numbers without replacement from $[1..b]$, denoted by $j_1, j_2, \ldots, j_d$, and fixes them for all future data collections. Then, for each $v \in V$, the user sends a sanitized vector $\mathbf{x}' = \left[(j_1, x_{j_1}), \ldots, (j_d, x_{j_d})\right]$ parameterized with the privacy guarantee ϵ_∞ as follows:

$$\forall l \in [1..d]: \Pr[x_{j_l} = 1] = \begin{cases} p = \frac{e^{\epsilon_\infty/2}}{e^{\epsilon_\infty/2}+1}, & \text{if } \mathsf{bucket}(v) = j_l \\ q = \frac{1}{e^{\epsilon_\infty/2}+1}, & \text{if } \mathsf{bucket}(v) \neq j_l \end{cases}.$$

In other words, users inform the server which bits are sampled as well as their perturbed values, but the server does not receive any information about the remaining $b-d$ bits. The server can estimate the number of times each bucket in $[1..b]$ has been reported with Eq. (1) by replacing n with $\frac{nd}{b}$ as each user only sampled d bits among b buckets.

In contrast to RAPPOR, there is no second round of sanitization, which means the user runs *d*BitFlipPM with ϵ_∞-LDP for all b buckets, with randomization applied to the d fixed bits $j_1, j_2, \ldots, j_d$ and memoizes the response. This approach adds uncertainty to the real value because multiple (close) values will be mapped to the same bucket. The highest protection is given when $d = 1$ [13], which will minimize the chances (to some extent) of detecting high data changes.

3 LOLOHA

In this section, we introduce our LOLOHA (Longitudinal Local Hashing) protocol for frequency monitoring throughout time under LDP constraints, and we analyze its utility and privacy.

The privacy analysis of longitudinal protocols requires special treatment because, since they are stateful, they cannot be modeled as mechanisms mapping values into values, but rather sequences into sequences. This makes the LDP constraint too strong in the long term as shown in the following theorem.

Theorem 3.1. *(LDP cannot be satisfied when $\tau \to \infty$) Consider a randomized longitudinal mechanism $\vec{\mathcal{M}}: [1..n]^\tau \to [1..m]^\tau$ mapping an input sequence $X_1, \ldots, X_\tau$ to an output sequence $Y_1, \ldots, Y_\tau$, for some positive integer τ. For the sake of utility of each reported value Y_t (0-LDP means total detriment of utility), assume some negligible but positive fixed $\alpha > 0$ such that the mechanism for*

generating Y_t from X_t and the history $X_1, Y_1, \ldots, X_{t-1}, Y_{t-1}$ is not α-LDP. If $\tau \geq \epsilon/\alpha$ then $\vec{\mathcal{M}}$ is not ϵ-LDP.

Proof. Let $y_1 = \arg\max_y \frac{\max_x P(X_1=x|Y_1=y)}{\min_x P(X_1=x|Y_1=y)}$, and call x_1^+ and x_1^- to the values that respectively maximize and minimize $P(X_1 = x|Y_1 = y_1)$. By the minimal utility assumption, $\frac{p(x_1^+|y_1)}{p(x_1^-|y_1)} > e^{\alpha}$.

Let $y_2 = \arg\max_y \frac{\max_x P(X_2=x|Y_2=y,Y_1=y_1,X_1=x_1)}{\min_x P(X_2=x|Y_2=y,Y_1=y_1,X_1=x_1)}$, and call x_2^+ and x_2^- to the values that respectively maximize and minimize $P(X_2 = x|Y_2 = y, Y_1 = y_1, X_1 = x_1)$. Since the output values of the mechanism are reported one by one in temporal order, we have $p(x_1, x_2|y_1, y_2) = p(x_1|y_1)p(x_2|y_2, y_1, x_1)$, hence by the minimal utility assumption and the first step, $\frac{p(x_1^+,x_2^+|y_1,y_2)}{p(x_1^-,x_2^-|y_1,y_2)} = \frac{p(x_1^+|y_1)p(x_2^+|y_2,y_1,x_1^+)}{p(x_1^-|y_1)p(x_2^-|y_2,y_1,x_1^-)} > e^{2\alpha}$.

Repeating this process inductively yields three sequences y_i, x_i^+ and x_i^- of length τ such that $\frac{p(x_1^+,\ldots,x_\tau^+|y_1,\ldots,y_\tau)}{p(x_1^-,\ldots,x_\tau^-|y_1,\ldots,y_\tau)} > e^{\tau\alpha}$.

This makes it impossible for the mechanism to be ϵ-LDP for any $\tau \geq \epsilon/\alpha$. □

For instance, assume that a user has a secret sequence $\mathbf{v} = [1, 1, 1, 3, 1, 2, 1, 1, 3]$ ($\tau = 9$ time steps), and reports $\vec{\mathcal{M}}(\mathbf{v}) := [\mathcal{M}(v_1), \ldots, \mathcal{M}(v_9)]$, in which $\mathcal{M}$ is the memoization mechanism $(1 \mapsto 2; 2 \mapsto 2; 3 \mapsto 3)$ that reuses the sanitized report. The server receives $[2, 2, 2, 3, 2, 2, 2, 2, 3]$, hence some time-related patterns in the sequence are exposed, but the memoization protects the uncertainty about the user actual values. As the sequence size grows, the vectorized memoization mechanism $\vec{\mathcal{M}}$ that processes temporal data continues to protect the values indefinitely, but fails to satisfy LDP. For this reason, we introduce the following relaxed definition of privacy for longitudinal mechanisms.

Definition 3.2 (Longitudinal LDP). For a longitudinal memoizing mechanism $\mathcal{M} : A^\tau \to B^\tau$, in which $A = [1..k]$, let $\mathcal{M}^\star$ denote a mechanism that takes as input a permutation x of A and outputs $\mathcal{M}^\star(x) := x''$ by shuffling the k entries of x, yielding x', and letting $x_i'' := \mathcal{M}(x_i')$ for each $i = 1..k$, sequentially. $\mathcal{M}$ is said to be ϵ-LDP on the users' values iff $\mathcal{M}^\star$ is ϵ-LDP.

Definition 3.2 discards all information contained in time correlation by shuffling the input and aggregates the total privacy loss after all input values have been memoized. Moreover, Definition 3.2 corresponds to the total privacy budget that will be consumed for sanitizing all the values of the user.

Previous influential works, such as RAPPOR [23] and *d*BitFlipPM [13], handle the negative consequences of Theorem 3.1 implicitly by assuming that the data values (or buckets) never change or change in an uncorrelated manner. We consider the former to be unrealistic and the latter is insufficient to guarantee LDP, though it makes users indistinguishable. In this paper, we privileged Definition 3.2 over extreme assumptions on the data to be able to explain at least what is actually being protected by the mechanism when the assumptions do not hold. Hence, we present long term guarantees in terms of LDP on the users' values, but also, single-report LDP guarantees, as done in the literature, which are equivalent to LDP assuming constant values.

3.1 Overview of LOLOHA

LOLOHA is inspired by the strengths of RAPPOR [23] (double sanitization to minimize data change detection) and *d*BitFlipPM [13] (several values are mapped to the same randomized value) protocols. More precisely, LOLOHA is based on LH for the PRR step to satisfy ϵ_∞-LDP (upper bound), which significantly reduces the domain size. Thus, the user will uniformly choose at random a universal hash function H that maps the original domain $V \to [1..g]$, with $g \geq 2$ typically much smaller than $k = |V|$. Indeed, given a general (universal) family of hash functions $\mathscr{H}$, each input value $v \in V$ is hashed into a value in $[1..g]$ by hash function $\mathrm{H} \in \mathscr{H}$, and the universal property requires:

$$\forall v_1, v_2 \in V, v_1 \neq v_2 : \quad \Pr_{\mathrm{H}\in\mathscr{H}}[\mathrm{H}(v_1) = \mathrm{H}(v_2)] \leq \frac{1}{g}.$$

In other words, approximately k/g values $v \in V$ can be mapped to the same hashed value $\mathrm{H}(v)$ in $[1..g]$ due to collision. After the hashing step, to satisfy ϵ_∞-LDP, the user invokes the GRR protocol to the hashed value $x = \mathrm{H}(v)$ and memoizes the response $x' = \mathcal{M}_{\mathrm{GRR}}(x; \epsilon_\infty)$. Then, the value x' will be reused as the basis for all future reports on the **hashed value** x, which supports all values in set $X_H = \{v \in V \mid \mathrm{H}(v) = x\}$. The intuition is that the user only leaks ϵ_∞ for each hashed value $x \in [1..g]$ as they support all values $v \in V$ that collide to $x = \mathrm{H}(v)$. Notice that instead of memoization, users could also pre-compute the mapping for each input value. These two methods would be equivalent in terms of the functionality provided.

Moreover, in contrast with the *d*BitFlipPM protocol in which only close values are mapped to the same bucket, any two values in V can collide with probability at most $1/g$. Therefore, even if the user's value changes periodically, correlated or in a abrupt manner, there will still be uncertainty on the actual value v. However, with only this PRR step, it would be possible to detect some of the data changes due to the randomization of a different hash value. Therefore, LOLOHA also requires the user to apply a second round of sanitization (*i.e.*, IRR step) to the memoized values x' with the GRR protocol such that the first report satisfies ϵ_1-LDP, for some chosen positive $\epsilon_1 < \epsilon_\infty$.

3.2 Client-Side of LOLOHA

Algorithm 1 displays the pseudocode of LOLOHA on the client-side, which receives as input: the true sequence of values $\mathbf{v} = [v_1, v_2, \ldots, v_\tau]$ of the user that is running the code, a universal family $\mathscr{H}$ of hash functions $\mathrm{H} : V \to [1..g]$, and the constants $\epsilon_1, \epsilon_\infty$, with $0 < \epsilon_1 < \epsilon_\infty$, that represent respectively the leakage of the first report and the maximal longitudinal leakage.

Algorithm 1 Client-Side of LOLOHA.

Input: User longitudinal values $[v_1, v_2, \ldots, v_\tau]$, family $\mathscr{H}$ of hash functions and constants $0 < \epsilon_1 < \epsilon_\infty$.
Output: None. Sends data to server during execution.

$\mathrm{H} \leftarrow_R \mathscr{H}$ ▷ Hash function chosen at random
Send H.
$\epsilon_{\mathrm{IRR}} \leftarrow \ln\left(\frac{e^{\epsilon_\infty+\epsilon_1}-1}{e^{\epsilon_\infty}-e^{\epsilon_1}}\right)$
for each time $t \in [1..\tau]$ **do**:
  $x \leftarrow \mathrm{H}(v_t)$. ▷ Hash step
  if x *is not* memoized **then**:
    $x' \leftarrow \mathcal{M}_{\mathrm{GRR}}(x; \epsilon_\infty)$ over $[1..g]$. ▷ PRR step
    Memoize output x' for x.
  else:
    Get memoized output x' for x.
  end if
  $x_t'' \leftarrow \mathcal{M}_{\mathrm{GRR}}(x'; \epsilon_{\mathrm{IRR}})$ over $[1..g]$. ▷ IRR step
  Send x_t''. ▷ Sanitized data
end for

Privacy analysis. The privacy guarantees of Algorithm 1 are detailed in Theorems 3.3, 3.4 and especially 3.5.

Theorem 3.3. *(Single report LDP of memoization) Let $\mathcal{M} : V \to \mathscr{H} \times [1..g]$ denote the process of applying the hash and PRR steps of LOLOHA to a single element $v \in V$, producing $\mathcal{M}(v) = (\mathrm{H}, x')$. Then $\mathcal{M}$ is ϵ_∞-LDP.*

Proof. The parameters for the PRR step are $p = \frac{e^{\epsilon_\infty}}{e^{\epsilon_\infty}+g-1}$ and $q = \frac{1}{e^{\epsilon_\infty}+g-1}$. For any two possible input values $v_1, v_2 \in V$ and any reported output (H, x'), we have

$$\frac{\Pr\left[(\mathrm{H}, x')|v_1\right]}{\Pr\left[(\mathrm{H}, x')|v_2\right]} \leq \frac{p}{q} = \frac{\frac{e^{\epsilon_\infty}}{e^{\epsilon_\infty}+g-1}}{\frac{1}{e^{\epsilon_\infty}+g-1}} = e^{\epsilon_\infty}.$$

□

Theorem 3.4. *(Single report LDP of LOLOHA) Let $\mathcal{M} : V \to \mathscr{H} \times [1..g]$ denote the process of applying the hash, PRR, and IRR steps of LOLOHA to a single element $v \in V$, producing $\mathcal{M}(v) = (\mathrm{H}, x'')$. Then $\mathcal{M}$ is ϵ_1-LDP.*

Proof. Let (p_1, q_1) denote the parameters for the PRR step and (p_2, q_2), the parameters for the IRR step. That is, $p_1 = \frac{e^{\epsilon_\infty}}{e^{\epsilon_\infty}+g-1}$, $q_1 = \frac{1}{e^{\epsilon_\infty}+g-1}$, $p_2 = \frac{e^{\epsilon_{\mathrm{IRR}}}}{e^{\epsilon_{\mathrm{IRR}}}+g-1}$, and $q_2 = \frac{1}{e^{\epsilon_{\mathrm{IRR}}}+g-1}$. If $x'' \neq \mathrm{H}(v)$, it must have changed during either the PRR or the IRR step, and if $x'' = \mathrm{H}(v)$, either it was not changed during either step or it was changed during both. From this analysis, it can be concluded that for each $y \in [1..g]$, we have

$$\Pr[x'' = y] = \begin{cases} p_1 p_2 + q_1 q_2, & \text{if } y = \mathrm{H}(v), \\ p_1 q_2 + q_1 p_2, & \text{if } y \neq \mathrm{H}(v). \end{cases}$$

Therefore, for any two possible input values $v_1, v_2 \in V$ and any output (H, x''), we have,

$$\frac{\Pr\left[(\mathrm{H}, x'')|v_1\right]}{\Pr\left[(\mathrm{H}, x'')|v_2\right]} \leq \frac{p_1 p_2 + q_1 q_2}{p_1 q_2 + q_1 p_2} = \frac{e^{\epsilon_\infty} \cdot e^{\epsilon_{\mathrm{IRR}}} + 1 \cdot 1}{e^{\epsilon_\infty} \cdot 1 + 1 \cdot e^{\epsilon_{\mathrm{IRR}}}}.$$

Moreover, since $e^{\epsilon_{\mathrm{IRR}}} = \frac{e^{\epsilon_\infty+\epsilon_1}-1}{e^{\epsilon_\infty}-e^{\epsilon_1}}$, then $e^{\epsilon_{\mathrm{IRR}}} e^{\epsilon_\infty} + 1 = e^{\epsilon_1}(e^{\epsilon_{\mathrm{IRR}}} + e^{\epsilon_\infty})$. Hence,

$$\frac{\Pr\left[(\mathrm{H}, x'')|v_1\right]}{\Pr\left[(\mathrm{H}, x'')|v_2\right]} \leq e^{\epsilon_1}.$$

□

Theorem 3.5. *(Privacy protection as $\tau \to \infty$) The client-side of LOLOHA is $g\epsilon_\infty$-LDP on the users' values.*

Proof. The non-vectorized memoization mechanism (hash and PRR steps) of LOLOHA is a function $\mathcal{M} : V \to [1..g]$ that can memorize at most g reports. For each separate individual report, we known that $\mathcal{M}$ satisfies ϵ_∞-LDP (Theorem 3.3). Therefore, by sequential composition of at most g results (Proposition 2.3), $\mathcal{M}$ satisfies $g\epsilon_\infty$-LDP, and LOLOHA satisfies $g\epsilon_\infty$-LDP on the users' values. □

The privacy guarantees of the IRR step (Theorem 3.4) degrade according to the number of reports $t \in [1..\tau]$ [21, 23]. If we let ϵ_t be the privacy guarantee on the users' values of Algorithm 1 for a fixed user using the data in times $[1..t]$, so that $t = 1$ matches exactly ϵ_1 (Theorem 3.4), then we have $\epsilon_1 \leq \epsilon_2 \leq \cdots \leq \epsilon_\tau \leq g\epsilon_\infty$.

Besides, from Algorithm 1, one can remark that instead of leaking a new ϵ_∞ for each $v \in V$, LOLOHA will only leak ϵ_∞ for each hashed value $x \in [1..g]$. Therefore, unlike RAPPOR that has a worst-case guarantee of $k\epsilon_\infty$-LDP on the users' values, the overall privacy guarantee of our LOLOHA solution will grow proportionally to the new domain size $2 \leq g \ll k$, with worst-case longitudinal privacy of $g\epsilon_\infty$-LDP on the users' values.

3.3 Server-Side of LOLOHA

The server-side algorithm of LOLOHA is described in Algorithm 2, which takes the reported values by n users and aggregates them to estimate the frequencies of each $v \in V$ at each point in time.

Algorithm 2 Server-Side of LOLOHA.

Input: Constants $0 < \epsilon_1 < \epsilon_\infty$, and for each user $u \in U$, a hash function $H_u : V \to [1..g]$ and a sequence of hash values $[x_1''^{(u)}, \ldots, x_\tau''^{(u)}]$.

Output: Matrix with estimations $\hat{f}_{\mathrm{LOLOHA}}(v)_t$ for each $v \in V$ at each $t \in [1..\tau]$.

1: Compute parameters:

$\epsilon_{\mathrm{IRR}} \leftarrow \ln\left(\frac{e^{\epsilon_\infty+\epsilon_1}-1}{e^{\epsilon_\infty}-e^{\epsilon_1}}\right)$; $n \leftarrow |U|$

$p_1 \leftarrow \frac{e^{\epsilon_\infty}}{e^{\epsilon_\infty}+g-1}$; $q_1' \leftarrow \frac{1}{g}$

$p_2 \leftarrow \frac{e^{\epsilon_{\mathrm{IRR}}}}{e^{\epsilon_{\mathrm{IRR}}}+g-1}$; $q_2 \leftarrow \frac{1}{e^{\epsilon_{\mathrm{IRR}}}+g-1}$

2: **for** each time $t \in [1..\tau]$ **do**:

3: **for** each $v \in V$ **do**:

4: $C(v) \leftarrow |\{u \in U \mid \mathrm{H}_u(v) = x_t''^{(u)}\}|$

5: $\hat{f}_L(v)_t \leftarrow \frac{C(v)-nq_1'(p_2-q_2)-nq_2}{n(p_1-q_1')(p_2-q_2)}$ ▷ Eq. (3) with q_1'.

6: **end for**

7: **end for**

8: **return** matrix $[\hat{f}_L(v)_t]_{t,v}$

For large n, the estimations of Algorithm 2 are guaranteed to be close to the true population parameters with high probability as explained in Proposition 3.6. Moreover, one can also compute the LOLOHA approximate variance $\mathbb{V}^*[\hat{f}_{\mathrm{LOLOHA}}(v)]$ by replacing the server parameters in Algorithm 2 into Eq. (5).

Proposition 3.6. *(Asymptotic utility guarantee of LOLOHA) Fix any arbitrary $t \in [1..\tau]$. For each $v \in V$, let $f(v)$ be the true population probability of producing the value v at time t, and let $\hat{f}(v) \in [0, 1]$ be the estimation produced by Algorithm 2 for time t. For any $\beta \in (0, 1)$, it holds with probability at least $1 - \beta$ that:*

$$\max_{v \in V} |\hat{f}(v) - f(v)| < \sqrt{\frac{k}{4n\beta(p_1 - q_1')(p_2 - q_2)}}.$$

Proof. Fix $v \in V$, and let Δ be the random variable given by $\Delta := \hat{f}(v) - f(v) \in [-1, 1]$ be a random variable. Since $\hat{f}(v)$ is unbiased, we have $\mathbb{E}[\Delta] = 0$ and $\mathbb{V}[\Delta] = \mathbb{V}[\hat{f}(v)]$. We remark that for any $\delta, \beta \in (0, 1)$, among all random variables Δ' defined in $[-1, 1]$ such that $\mathbb{E}[\Delta']$ and $\Pr[|\Delta'| \geq \delta] = \beta$, the one with minimal variance is the random variable Δ^* that concentrates a mass of $1-\beta$ at $\Delta' = 0$ and two masses of $\beta/2$ at $-\delta$ and δ. This random variable has variance $\mathbb{V}[\Delta^*] = \beta\delta^2$. Hence, for arbitrary $\delta \in (0, 1)$ and letting in particular $\beta := \Pr[|\hat{f}(v) - f(v)| \geq \delta]$, we conclude that $\mathbb{V}[\hat{f}(v)] = \mathbb{V}[|\hat{f}(v) - f(v)|] \geq \mathbb{V}[\Delta^*] = \Pr[|\hat{f}(v) - f(v)| \geq \delta] \cdot \delta^2$. In other words,

$$\Pr[|\hat{f}(v) - f(v)| \geq \delta] \leq \mathbb{V}[\hat{f}(v)]/\delta^2.$$

Now, considering all $v \in V$ simultaneously, we obtain $\Pr[\max_{v \in V} |\hat{f}(v) - f(v)| \geq \delta] \leq \sum_{v \in V} \Pr[|\hat{f}(v) - f(v)| \geq \delta] =$

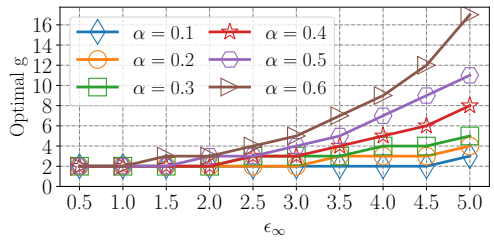


Figure 1: Optimal g selection for our OLOLOHA protocol by varying the longitudinal ϵ_∞ and first report $\epsilon_1 = \alpha\epsilon_\infty$ privacy guarantees, for $\alpha \in \{0.1, 0.2, \ldots, 0.6\}$.

$(1/\delta^2) \sum_{v \in V} \mathbb{V}[\hat{f}(v)]$. By rewriting this equation in terms of confidence, we conclude that with probability at least $1 - \beta$,

$$\max_{v \in V} |\hat{f}(v) - f(v)| < \sqrt{\textstyle\sum_{v \in V} \mathbb{V}[\hat{f}(v)]/\beta}.$$

Lastly, from Eq. (4) it can be concluded that $\mathbb{V}[\hat{f}(v)] \leq 1/4n(p_1 - q_1')(p_2 - q_2)$ because the product $\gamma(1-\gamma)$ is maximal at $\gamma = 1/2$. As a consequence, $\max_{v \in V} |\hat{f}(v) - f(v)| < \sqrt{k/4n\beta(p_1 - q_1')(p_2 - q_2)}$. □

3.4 Selecting and Optimizing Parameter g

Binary LOLOHA (BiLOLOHA). Following Theorem 3.5, the strongest longitudinal privacy protection of LOLOHA is when $g = 2$.

Optimal LOLOHA (OLOLOHA). To maximize the utility of LOLOHA, we find the optimal g value by taking the partial derivative of $\mathbb{V}^*[\hat{f}_{\text{LOLOHA}}(v)]$ with respect to g. Let $\epsilon_1 = \alpha\epsilon_\infty$, for $\alpha \in (0, 1)$. This partial derivative is a function in terms of ϵ_∞ and α, or alternatively, in terms of $a = e^{\epsilon_\infty}$ and $b = e^{\alpha\epsilon_\infty}$, and it is minimized when g equals (*cf.* development in repository [1]):

$$g = 1 + \max\left(1, \left\lfloor \frac{1 - a^2 + \sqrt{a^4 - 14a^2 + 12ab(1-ab) + 12a^3b + 1}}{6(a-b)} \right\rfloor\right), \tag{6}$$

in which $\lfloor . \rceil$ means rounding to the closest integer. Fig. 1 illustrates the optimal g selection with Eq. (6) by varying the longitudinal privacy guarantee $\epsilon_\infty = [0.5, 1, \ldots, 4.5, 5]$ and $\alpha \in \{0.1, 0.2, \ldots, 0.6\}$. From Fig. 1, one can remark that in high privacy regimes (*i.e.*, low ϵ values), the optimal g is binary (*i.e.*, our BiLOLOHA protocol with $g = 2$). As ϵ_∞ or/and $\epsilon_1 = \alpha\epsilon_\infty$ get(s) higher (low privacy regimes), the optimal g is non-binary, which can maximize utility with a cost in the overall longitudinal privacy $g\epsilon_\infty$-LDP on the users' values, for $g > 2$.

4 THEORETICAL COMPARISON

In this section, we compare LOLOHA with the state-of-the-art protocols described in the previous Section 2.4 from a theoretical point of view. Table 1 shows a summary of the main characteristics of these protocols, excluding utility.

For the theoretical utility, numerical analysis is preferred over an analytical one because the formulas of variance and approximate variance are excessively complex. For L-OSUE and dBitFlipPM, the approximate variances are $\frac{4e^{\epsilon_1}}{n(e^{2\epsilon_1} - 2e^{\epsilon_1} + 1)}$ and $\frac{b}{2dn \sinh\left(\frac{\epsilon_\infty}{2}\right)}$ respectively, but for the other protocols, the formulas are provided only in the repository [1] since they are excessively verbose for this document.

Protocol	**Comm.** bits per user per time step	**Server** run-time complexity	**Privacy loss** budget consumption
LOLOHA	$\lceil \log_2 g \rceil$	$n\,k$	$g\,\epsilon_\infty$
L-GRR [5]	$\lceil \log_2 k \rceil$	n	$k\,\epsilon_\infty$
RAPPOR [23]	k	$n\,k$	$k\,\epsilon_\infty$
L-OSUE [5]	k	$n\,k$	$k\,\epsilon_\infty$
dBitFlipPM [13]	d	$n\,b$	$\min(d+1, b)\,\epsilon_\infty$

Table 1: Theoretical comparison of the protocols.

In order to evaluate numerically the approximate variance $\mathbb{V}^*$ of LOLOHA in comparison with state-of-the-art ones [5, 23], for each protocol, we set the longitudinal privacy guarantee ϵ_∞ (upper bound) and the first report privacy guarantee $\epsilon_1 = \alpha\epsilon_\infty$ (lower bound), for $\alpha \in (0, 1)$. This allows to obtain parameters p_1, q_1, p_2, q_2 for each protocol, which are then used to compute their approximate variance with Eq. (5).

Fig. 2 illustrates the numerical values of the approximate variance for our LOLOHA protocols, RAPPOR [23], and L-OSUE [5] with $n = 10000$, $\epsilon_\infty = [0.5, 1, \ldots, 4.5, 5]$, and $\alpha \in \{0.1, 0.2, \ldots, 0.6\}$. From Fig. 2, one can remark that all protocols have similar variance values when $\alpha \leq 0.3$ with only a small difference when ϵ_∞ is high. However, in low privacy regimes, *i.e.*, when ϵ_∞ and α are high, BiLOLOHA is the least performing protocol in terms of utility, accompanied by RAPPOR. Indeed, our OLOLOHA protocol has a very similar utility as the optimized L-OSUE [5] protocol, which indicates a clear connection also found between their one-round versions [40], *i.e.*, OLH and OUE.

Though not included in our analysis, the L-GRR protocol from [5] has shown to be very sensitive to k (a parameter on which its variance depends on), leading to extremely high values that would obfuscate the curves of the other protocols in Fig. 2. However, L-GRR is ideal when k is small, which is the case for instance for binary attributes. Besides, we also did not numerically compare our protocols with dBitFlipPM as it only has a single round of sanitization. A proper comparison with dBitFlipPM would be only considering the PRR step of our LOLOHA protocols. Therefore, by comparing the approximate variances of double randomization protocols, we can conclude that our LOLOHA protocols preserve as much utility as state-of-the-art protocols [5, 23].

Moreover, from Table 1, LOLOHA has less communication cost than L-UE and similar server time computation, which is advantageous for large-scale system deployment to monitor frequency longitudinally. In addition, one clear limitation of RAPPOR, L-OSUE, and L-GRR is that they do not support even small data changes of the user's actual data [13], which requires to invoke the whole algorithm again on the new value. Therefore, following Definition 3.2 and Proposition 2.3, the overall privacy guarantee of RAPPOR, L-OSUE, and L-GRR, for all user's true value $v \in V$ (assuming the user's value will change periodically) will grow proportionally to the number of *data changes*, with worst-case longitudinal privacy of $k\epsilon_\infty$-LDP on the users' values.

On the other hand, with dBitFlipPM, the overall privacy guarantee of dBitFlipPM for all user's true value $v \in V$ (assuming the user's value will change periodically) will grow proportionally to the number *of bits* d or the number of *bucket changes*, with worst-case longitudinal privacy of $\min(d+1, b)\epsilon_\infty$-LDP on the users' values (*cf.* Definition 3.2 and Proposition 2.3). However, there is a loss of information due to both the generalization of

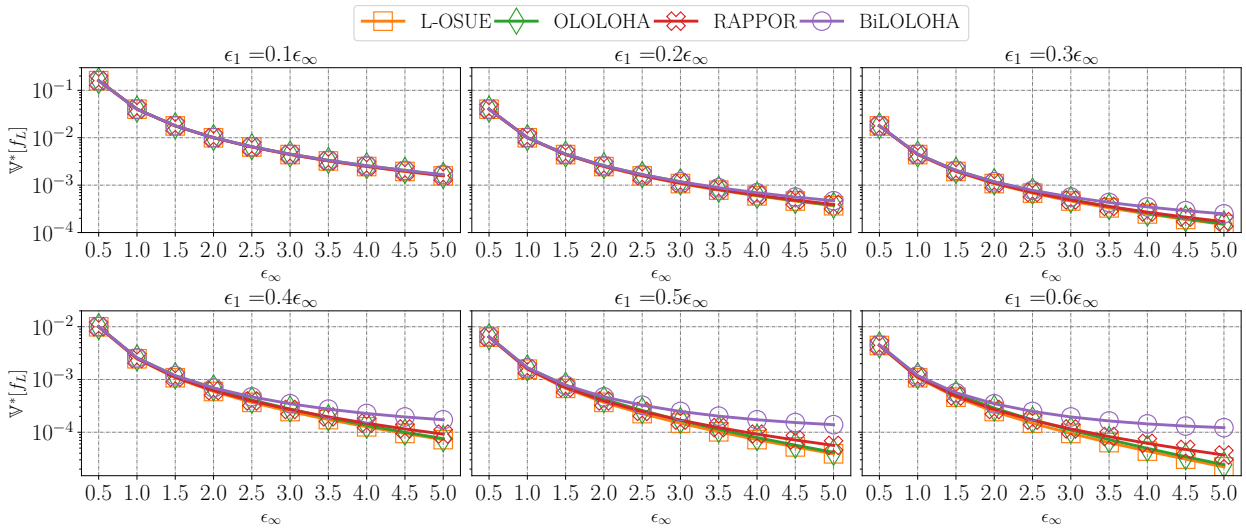


Figure 2: Numerical values of the approximate variance $\mathbb{V}^*[\hat{f}_L(v)]$ in Eq. (5) of our LOLOHA protocols, RAPPOR [23], and L-OSUE [5] varying the longitudinal ϵ_∞ and first report $\epsilon_1 = \alpha\epsilon_\infty$ privacy guarantees, for $\alpha \in \{0.1, 0.2, \ldots, 0.6\}$.

the original domain size k to b buckets and due to sampling only d bits. Besides, the dBitFlipPM protocol is vulnerable to detecting high data changes (*i.e.*, change of real bucket) as there is no second round of sanitization (*i.e.*, IRR step) [42]. This data change detection problem is (to some extent) minimized when d is small.

5 EXPERIMENTAL EVALUATION

In this section, we present the setup of our experiments and the experimental results of our LOLOHA protocols in comparison with the state-of-the-art.

5.1 Setup of Experiments

The main goal of our experiments is to study the effectiveness of our proposed LOLOHA protocols on longitudinal frequency estimates through multiple $\tau > 1$ data collections. In particular, we aim to show that our LOLOHA protocols (i) maintain competitive utility to state-of-the-art memoization-based LDP protocols [5, 13, 23] while (ii) minimize longitudinal privacy loss. With these objectives in mind, we run experiments using both synthetic and real-world datasets.

Environment. All algorithms are implemented in Python 3 with Numpy and Numba libraries. The codes we develop for all experiments are available in the repository [1]. Since LDP algorithms are randomized, we report average results over 20 runs.

Datasets. We use the following real and synthetic datasets.

- **Syn.** To simulate the deployment of [13] to collect data every 6 hours, we generate a synthetic dataset with $k = 360$ (*i.e.*., the number of minutes in 6 hours), $n = 10000$ users, and $\tau = 120$ data collections (*i.e.*, 4x over 30 days). For each user, the value at the first timestamp follows a Uniform distribution. For each subsequent time, a change can occur with probability $p_{ch} = 0.25$, with value following a Uniform distribution too.
- **Adult.** This is a classical dataset from the UCI machine learning repository [15] with $n = 45222$ samples after cleaning. We only selected the "hours-per-week" attribute with $k = 96$. To simulate multiple data collections, we randomly permuted the data $\tau = 260$ times (*i.e.*, 52 weeks over 5-years). Note that the real frequency remains the same but each user has a random private sequence.
- **DB_MT.** This dataset is produced by the `folktables` Python package [14] that provides access to datasets derived from the US Census. We selected the survey year 2018 and the "Montana" state, which results in $n = 10336$ samples. To simulate $\tau = 80$ counter data collections, we selected all person record-replicate weights attributes[1], *i.e.*, PWGTP1, ..., PWGTP80. The total number of unique values among all columns is $k = 1412$.
- **DB_DE.** Similar to DB_MT, we selected the "Delaware" state, which results in $n = 9123$, $\tau = 80$, and $k = 1234$.

Methods evaluated. We consider for evaluation the following longitudinal LDP protocols:

- *RAPPOR.* The utility-oriented protocol from [23] based on SUE (*cf.* Section 2.4.1).
- *L-OSUE.* The optimized L-UE protocol from [5] (*cf.* Section 2.4.2).
- *L-GRR.* The optimized longitudinal protocol from [5] when k is small (*cf.* Section 2.4.3).
- *dBitFlipPM.* The one-round randomization mechanism from [13] with $d \in \{1, b\}$, referred respectively as 1BitFlipPM and bBitFlipPM, in which the former 1BitFlipPM is tuned for privacy and the latter bBitFlipPM for utility (*cf.* Section 2.4.4)
- *LOLOHA.* Our protocols following Algorithm 1, which are BiLOLOHA with $g = 2$ adjusted for privacy and OLOLOHA with g following Eq. (6) tuned for utility.

Privacy metrics. We vary the longitudinal privacy parameter in the range $\epsilon_\infty = [0.5, 1, \ldots, 4.5, 5]$ and $\epsilon_1 = \alpha\epsilon_\infty$, for $\alpha \in \{0.4, 0.5, 0.6\}$, to compare our experimental results with numerical ones from Section 4 (with higher visibility).

Performance metrics. To evaluate our results, we use the MSE averaged by the number of data collection τ, denoted by MSE_{avg}. Thus, for each time $t \in [1..\tau]$, we compute for each value $v \in V$ the estimated frequency $\hat{f}_L(v)_t$ and the real one $f(v)_t$ and calculate their differences before averaging by τ. More formally,

[1] https://www.census.gov/programs-surveys/acs/microdata/documentation.html.

$$MSE_{avg} = \frac{1}{\tau} \sum_{t \in [1..\tau]} \frac{1}{|V|} \sum_{v \in V} \left(f(v)_t - \hat{f}_L(v)_t \right)^2 . \quad (7)$$

We also assess the averaged longitudinal privacy loss for all users, denoted by $\check{\epsilon}_{avg}$. More precisely, after the end of all data collections τ, we compute for each user $u \in U$ their overall longitudinal privacy loss $\check{\epsilon}_{\infty}^{(u)}$ and average by n. For example, RAPPOR (and L-GRR and L-OSUE) leaks a new ϵ_{∞} in each data change with $\check{\epsilon}_{\infty} \leq k\epsilon_{\infty}$, while LOLOHA protocols leak a new ϵ_{∞} in each hash value change with $\check{\epsilon}_{\infty} \leq g\epsilon_{\infty}$. More formally,

$$\check{\epsilon}_{avg} = \frac{1}{n} \sum_{u \in U} \check{\epsilon}_{\infty}^{(u)} . \quad (8)$$

Finally, for the *d*BitFlipPM protocol, we also evaluate the percentage of users in which an attacker can identify **all** (bucket) data change points (*i.e.*, *worst-case* analysis) due to different PRR reports throughout the τ data collections.

5.2 Results

First, we compare the utility performance of our LOLOHA protocols with all four state-of-the-art memoization-based protocols for frequency monitoring under LDP guarantees, namely, RAPPOR [23], L-OSUE [5], L-GRR [5], and *d*BitFlipPM [13], for $d \in \{1, b\}$. Fig. 3 illustrates the MSE_{avg} metric in Eq. (7) for all methods and all Syn, Adult, DB_MT, and DB_DE datasets, by varying the longitudinal ϵ_{∞} and first report $\epsilon_1 = \alpha\epsilon_{\infty}$ privacy guarantees, for $\alpha \in \{0.4, 0.5, 0.6\}$. On the one hand, since $k \leq 360$ for Syn and Adult datasets, when implementing *d*BitFlipPM, we select $b = k$ to estimate the same k-bins histogram as all other methods in Figs. 3a and 3b. On the other hand, we select $b = \lfloor k/4 \rfloor$ bins for both DB_MT ($k = 1412$) and DB_DE ($k = 1234$) datasets, but we did not include the error metric of *d*BitFlipPM in Figs. 3c and 3d as the error is five orders of magnitude higher due to histograms of different sizes ($b < k$).

Fig. 3 shows that the experimental results with all datasets match the numerical results of variance values from Fig. 2 for our LOLOHA protocols, RAPPOR, and L-OSUE. More specifically, our OLOLOHA protocol has similar utility to the optimized L-OSUE protocol, a relationship also find between their one-round versions OLH and OUE in [40]. In high privacy regimes, all four protocols, *i.e.*, RAPPOR, L-OSUE, BiLOLOHA, and OLOLOHA have very similar utility. In low privacy regimes, L-OSUE and OLOLOHA outperforms both RAPPOR and BiLOLOHA. The least performing longitudinal LDP protocols are L-GRR and 1BitFlipPM, the former due to high domain sizes k, as shown in [5], and the latter due to sampling only a single $d = 1$ bit out of b ones. The *b*BitFlipPM protocol outperforms all experimented longitudinal LDP protocols due to having only a single round of sanitization (*i.e.*, the PRR step) and by reporting all $d = b$ bits, which is consistent with [13] (the larger d the greater the utility).

However, increasing the number of bits d the users must report negatively impacts privacy, as each new input value has a high probability of generating a new output value, which will be detected by the server. For instance, for both *d*BitFlipPM protocols, for $d \in \{1, b\}$, Table 2 exhibits the percentage of users in which **all** bucket changes were detected by the server due to different PRR responses throughout τ data collections, for all Syn, Adult, DB_MT, and DB_DE datasets. Remark that when $d = 1$, the protocol is adjusted for privacy, thus being less vulnerable with respect to privacy with only a small percentage (< 1%) of users that the server always detected a different randomized output due to different input values. Besides, one can note that the percentage of attacked users decreases as ϵ_{∞} gets higher when $d = 1$. The intuition is that the probability of randomizing the single bit will be smaller with high ϵ_{∞}, thus generating the same report many times. On the other hand, the *b*BitFlipPM protocol is tuned for utility, which increased the probability of *always* generating a new randomized output due to new input values and, thus leading to 100% of detection for all four datasets. Though we only perform both extreme cases (lower $d = 1$ and upper $d = b$ bounds), one can picture the privacy-utility trade-off of *d*BitFlipPM for other d values in between our results of Fig. 3 and Table 2.

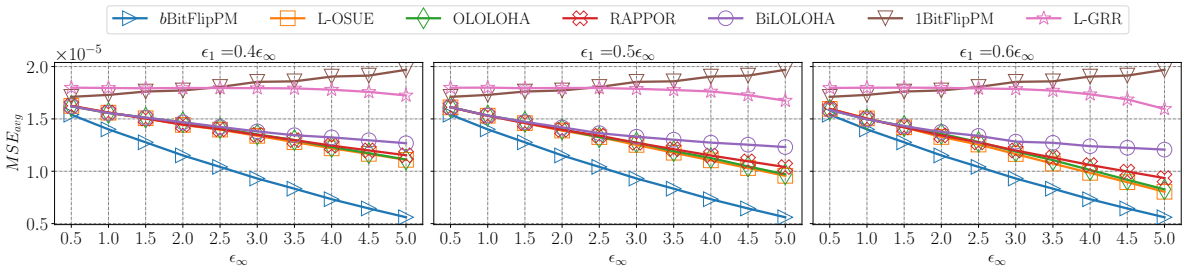
Table 2: Percentage of users in which the server detected all data change points for *d*BitFlipPM, for $d \in \{1, b\}$, and all Syn, Adult, DB_MT, and DB_DE datasets.

ϵ_{∞}	$d = 1$				$d = b$			
	Syn	Adult	DB_MT	DB_DE	Syn	Adult	DB_MT	DB_DE
0.5	0%	0%	0.0048%	0%	100%	100%	100%	100%
1.0	0%	0%	0.0044%	0%	100%	100%	100%	100%
1.5	0%	0%	0.0048%	0%	100%	100%	100%	100%
2.0	0%	0%	0.0039%	0%	100%	100%	100%	100%
2.5	0%	0%	0.0024%	0%	100%	100%	100%	100%
3.0	0%	0%	0.0024%	0%	100%	100%	100%	100%
3.5	0%	0%	0.0024%	0%	100%	100%	100%	100%
4.0	0%	0%	0.0019%	0%	100%	100%	100%	100%
4.5	0%	0%	0.0010%	0%	100%	100%	100%	100%
5.0	0%	0%	0.0010%	0%	100%	99.99%	100%	100%

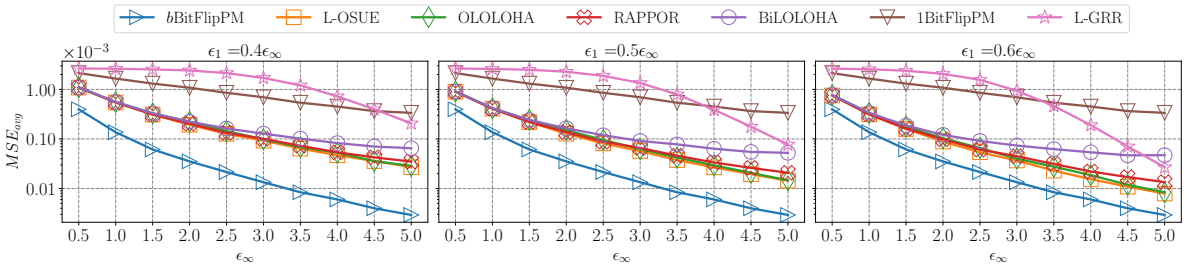
We now analyze the longitudinal privacy guarantees of our LOLOHA protocols in comparison with the state-of-the-art memoization-based LDP protocols. Fig. 4 illustrates the $\check{\epsilon}_{avg}$ metric in Eq. (8) for all methods and all Syn, Adult, DB_MT, and DB_DE datasets, by varying the longitudinal ϵ_{∞} and first report $\epsilon_1 = \alpha\epsilon_{\infty}$ privacy guarantees, for $\alpha \in \{0.4, 0.5, 0.6\}$. Notice that the results of *d*BitFlipPM protocols in Figs. 4a and 4b are with $b = k$ buckets and in Figs. 4c and 4d are with $b = \lfloor k/4 \rfloor$ buckets.

From Fig. 4, one can remark that all four LDP protocols, RAPPOR, L-OSUE, L-GRR, and *b*BitFlipPM (when $b = k$ in Figs. 4a and 4b), have an averaged longitudinal privacy loss linear to the number of data changes the users performed throughout the τ data collections. Fig. 4a presents the smallest $\check{\epsilon}_{avg}$ as both $k = 360$ and the change rate $p_{ch} = 0.25$ are small. However, in a worst-case scenario in which the users change their values significantly or $\tau \to \infty$, the overall privacy loss of RAPPOR, L-OSUE, L-GRR, and *b*BitFlipPM can grow to values as large as $k\epsilon_{\infty}$ for all datasets. Note that in Figs. 4a and 4b, naturally, setting $b = k$ does not benefit from the *d*BitFlipPM advantage for enhancing longitudinal privacy protection by mapping several close values to the same bin, which leads to higher $\check{\epsilon}_{avg}$. In contrast, in Figs. 4c and 4d, the longitudinal privacy loss of *b*BitFlipPM protocols is lower than RAPPOR, L-OSUE, and L-GRR because $b = \lfloor k/4 \rfloor$ buckets, but still significantly higher than our LOLOHA protocols.

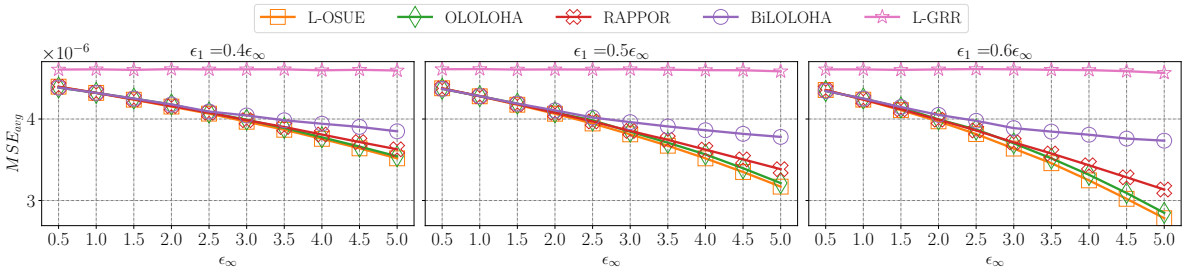
Indeed, the privacy loss of our LOLOHA protocols depends only on the new domain size $g \geq 2$, which is agnostic to k. For this reason, our BiLOLOHA protocol with $g = 2$ leaked about 15 to 25 orders of magnitude less than the state-of-the-art LDP protocols considering the experimented τ values. These are similar results achieved by the 1BitFlipPM protocol, which agrees with the theoretical analysis in Table 1, although BiLOLOHA consistently and considerably outperforms 1BitFlipPM in terms



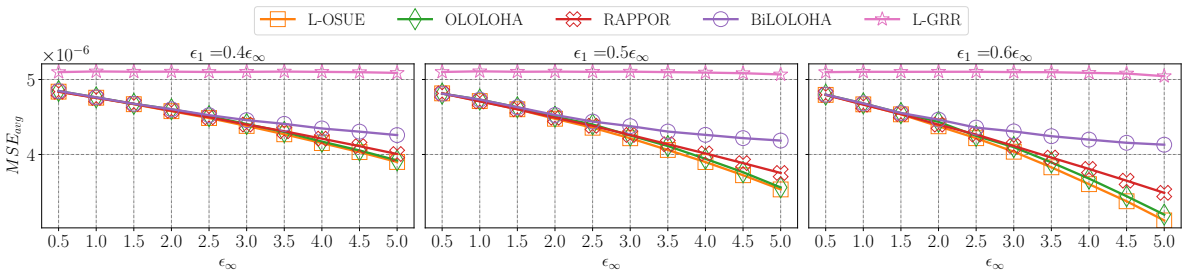
(a) Syn dataset: $k = 360$, $n = 10000$, and $\tau = 120$.



(b) Adult dataset: $k = 96$, $n = 45222$, and $\tau = 260$.



(c) DB_MT dataset: $k = 1412$, $n = 10336$, and $\tau = 80$.



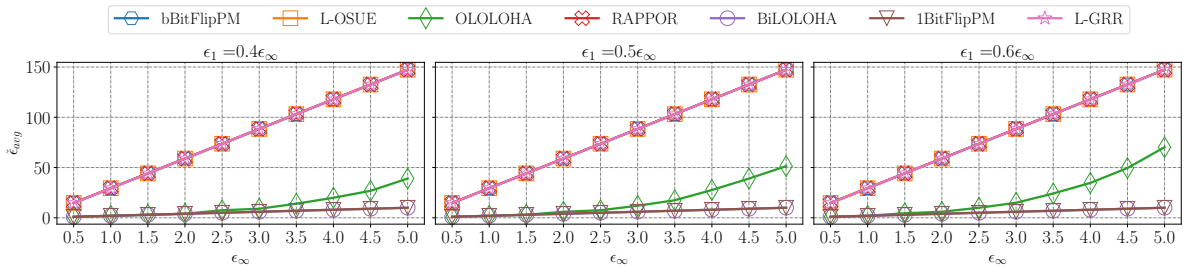
(d) DB_DE dataset: $k = 1234$, $n = 9123$, and $\tau = 80$.

Figure 3: Averaged MSE for τ data collections in Eq. (7) by varying the longitudinal ϵ_∞ and first report $\epsilon_1 = \alpha\epsilon_\infty$ privacy guarantees, for $\alpha \in \{0.4, 0.5, 0.6\}$, on (a) Syn, (b) Adult, (c) DB_MT, and (d) DB_DE datasets. The evaluated methods are: *d*BitFlipPM [13], L-OSUE [5], RAPPOR [23], L-GRR [5], and our LOLOHA protocols.

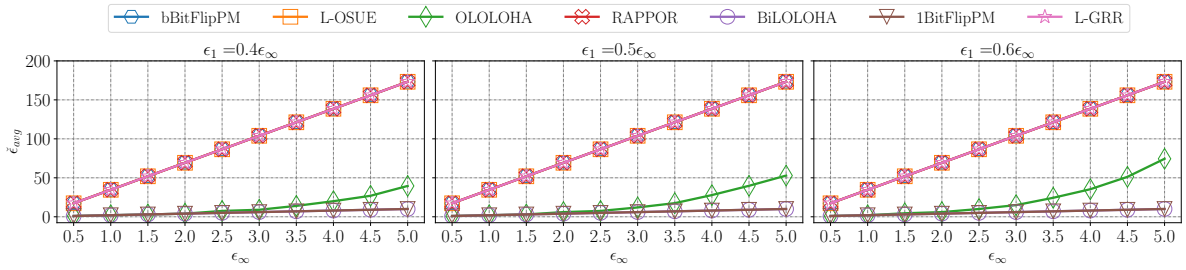
of utility loss (see Fig. 3). Besides, since our OLOLOHA protocol has privacy loss depending on the optimal g value in Eq. (6), which can be $g > 2$ in low privacy regimes, it only resulted in about 2 to 5 order of magnitude less privacy loss than the state-of-the-art LDP protocols, for the experimented τ value. More specifically, when ϵ_∞ is high and $\alpha = 0.6$ (see Fig. 4d), OLOLOHA leaked about 2 orders of magnitude less privacy loss than the *b*BitFlipPM protocol. However, as the number of data collections $\tau \to \infty$, *b*BitFlipPM privacy loss will go to $b\epsilon_\infty$, which is b/g times higher than the one from OLOLOHA with $g\epsilon_\infty$. Besides, in practice, lower values of ϵ_∞ and $\alpha \ll 1$ should be used to ensure strong longitudinal privacy guarantees since the first $\epsilon_1 = \alpha\epsilon_\infty$-LDP report. As shown in Fig. 1, this will mean lower values of g, which will substantially decrease the longitudinal privacy loss of OLOLOHA.

5.3 Discussion

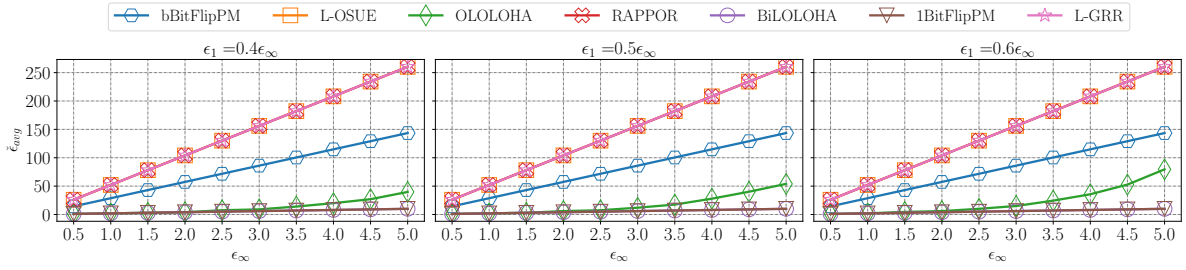
In brief, we have evaluated in our experiments the performance of our LOLOHA protocols in comparison with four state-of-the-art memoization-based LDP protocols [5, 13, 23] for frequency monitoring on different datasets and varying different parameters. We now summarize the main findings that help justify the many claims of our paper.



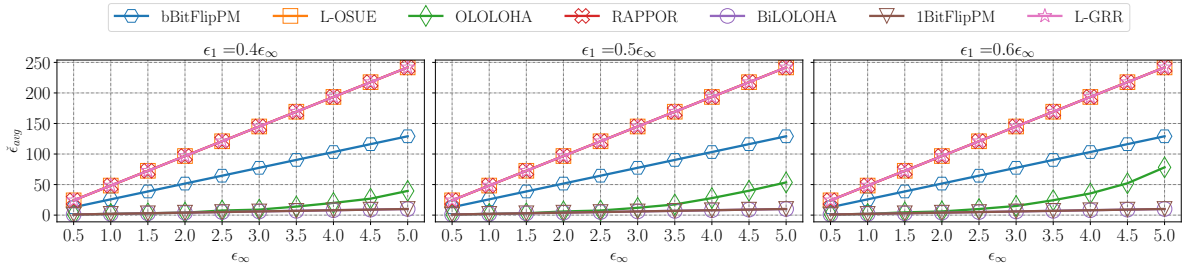
(a) Syn dataset: $k = 360$, $n = 10000$, and $\tau = 120$.



(b) Adult dataset: $k = 96$, $n = 45222$, and $\tau = 260$.



(c) DB_MT dataset: $k = 1412$, $n = 10336$, and $\tau = 80$.



(d) DB_DE dataset: $k = 1234$, $n = 9123$, and $\tau = 80$.

Figure 4: Averaged longitudinal privacy loss for τ data collections in Eq. (8) by varying the longitudinal ϵ_∞ and first report $\epsilon_1 = \alpha\epsilon_\infty$ privacy guarantees, for $\alpha \in \{0.4, 0.5, 0.6\}$, on (a) Syn, (b) Adult, (c) DB_MT, and (d) DB_DE datasets. The evaluated methods are: *d*BitFlipPM [13], L-OSUE [5], RAPPOR [23], L-GRR [5], and our LOLOHA protocols.

More precisely, the conclusions we stated in Section 4 are based on the analytical variances of the LDP protocols. To corroborate these conclusions, our empirical experiments in Section 5.2, which measured the MSE metric, do indeed correspond to the numerical results of the variances.

Furthermore, the main disadvantage of RAPPOR [23] and the two others optimized protocols from [5], (*i.e.*, L-GRR and L-OSUE), is the linear relation on k for the overall longitudinal privacy loss, *i.e.*, $k\epsilon_\infty$, as each data change needs to be memoized. Thus, for the monitoring of large-scale systems (*e.g.*, application usage, calories ingestion, preferred webpage, etc), the overall privacy loss of such protocols will be tremendous, being unrealistic for private frequency monitoring.

Even though the *d*BitFlipPM [13] generalizes the original domain size k to b buckets, there is still a linear relation on the new domain size $b \leq k$ for the overall longitudinal privacy loss, *i.e.*, $b\epsilon_\infty$, as each bucket change needs to be memoized *when the mechanism is tuned for utility*. What is more, this generalization naturally leads to loss of information and one has to carefully choose the bucket numbers/width for the best privacy-utility trade-off. Besides, the privacy-utility trade-off of *d*BitFlipPM also depends on the number of bits $d \leq b$ each user samples. However, even when $d = 1$, which offers the strongest protection [13], in our experiments, the server was still able to detect **all** bucket change of a small portion of users (see Table 2). Hence, as one

adjusts d for utility, *i.e.*, $1 < d \leq b$, the higher the attacker's success rate to detect all user's data changes will be.

The best choice for adequately balancing privacy and utility for frequency monitoring is with our LOLOHA protocols, as the privacy loss is only linear to the new (reduced) domain size $2 \leq g \ll k$. Though we only experiment with $80 \leq \tau \leq 260$ data collections in Section 5.2, in the worst case, this represents a significant k/g decrease factor of privacy loss by our LOLOHA protocols. Intuitively, LOLOHA can be tuned to satisfy the strongest longitudinal privacy protection by selecting $g = 2$ (*i.e.*, our BiLOLOHA protocol). In this setting, there is loss of utility in the encoding step through local hashing since the output is just one bit. For instance, even if this bit is transmitted correctly after the two rounds of sanitization, the server can only obtain one bit of information about the input (*i.e.*, to which half of the input domain the value belongs to [40]). Nevertheless, from the analytic variance analysis in Fig. 2 and empirical experiments in Fig. 3, LOLOHA is optimal with $g = 2$ in high privacy regimes, *i.e.*, low ϵ_∞ values, which is desirable for practical deployments.

As a limitation, users fix their randomly selected hash function $H \in \mathscr{H}$ with our LOLOHA protocols (*cf.* Algorithm 1), which can be regarded as a unique identifier in longitudinal data collection. However, this is a common assumption of the LDP model, which assumes the server already knows the users' identifiers [10, 21, 22, 39], but not their private data. One way to counter this link between the user's randomized report and their identifier is to assume a trusted intermediate, such as a shuffler, that does not collude with the server, *e.g.*, the Shuffle DP model [10, 21, 22], which we let the investigation for future work.

6 RELATED WORK

Differential privacy [18–20] has been increasingly accepted as the current standard for data privacy. The central DP model assumes a trusted curator, which collects the clients' raw data and releases sanitized aggregated data. The LDP model [16, 17, 30] does not rely on collecting raw data anymore, which has a clear connection with the concept of randomized response [41]. In recent years, there have been several studies on the local DP setting, *e.g.*, for frequency estimation of a single [2, 12, 24, 28, 29, 33, 40] and multiple [3, 31, 38] attributes; mean estimation [34, 39], heavy hitter estimation [8, 9], and machine learning [32, 43].

As for locally differentially private monitoring, Erlingsson, Pihur, and Korolova [23] proposed the RAPPOR algorithm for frequency monitoring that is based on the *memoization* solution described in Section 2.4. The recent study of Arcolezi *et al.* [5] generalizes this framework for optimally chaining two LDP protocols, proposing the L-GRR protocol that is optimized for small domain size k and the L-OSUE protocol for higher k (see Figs. 2 and 3). Moreover, Erlingsson *et al.* [21] formalize the privacy guarantees of using two rounds of sanitization under both local and shuffle DP guarantees. Naor and Vexler [33] also formalized the privacy guarantees of chaining two LDP protocol as well as introduced a new Everlasting privacy definition.

An alternative approach for memoization named dBitFlipPM has been proposed by Ding, Kulkarni, and Yekhanin [13], discussed in Section 2.4.4. The dBitFlipPM protocol allows frequent but only *small* changes in the original data since a high change (*i.e.*, a different bucket) can be detected by an attacker (*cf.* Table 2). Although an attacker that is able to identify a data change can still not infer the user's actual data (controlled by ϵ_∞), the overall LDP guarantees can be highly reduced if these changes are correlated [13, 22, 23]. For instance, the authors in [36] performed a detailed analysis of Apple's LDP implementation and examined its longitudinal privacy implications. Naor and Vexler [33] also investigated the trackability of RAPPOR following their new Everlasting privacy definition.

LOLOHA leverages the best of RAPPOR and dBitFlipPM, which can inherently minimize these inference attacks. More precisely, on the one hand, LOLOHA uses LH [40] for domain reduction, which allows many values to collide (universal hashing property) and thus creates uncertainty about the user's actual value. Indeed, LH protocols are the least attackable LDP protocols in the recent studies of Arcolezi *et al.* [7] and Emre Gursoy *et al.* [26] considering a Bayesian adversary. Besides that, LOLOHA also has two rounds of sanitization following RAPPOR's framework, which can improve privacy to minimize data change detection. Finally, another line of work for frequency monitoring under LDP is data change-based [22, 27, 35, 42], motivated by the fact that, generally, users' data changes infrequently. A similar idea was proposed much earlier in the work of Chatzikokolakis, Palamidessi, and Stronati [11], which proposed a predictive mechanism for location-based systems to utilize privacy budget only for new "hard" location points (*i.e.*, with bad predictions). However, these approaches normally impose restrictions on the number of data collections τ and on the number of data changes as their accuracy degrades linearly or sub-linearly with the number of changes in the underlying data distributions, which can limit their applicability and scalability to real-world systems.

7 CONCLUSION AND PERSPECTIVES

In this paper, we study the fundamental problem of monitoring the frequency of evolving data throughout time under LDP guarantees. We proposed a new locally differentially private protocol named LOLOHA, which is built on top of domain reduction to minimize longitudinal privacy loss up to a k/g factor and double randomization to enhance privacy. Through theoretical analysis, we have proven the longitudinal privacy (Theorems 3.3, 3.4, and 3.5) and accuracy guarantees (Proposition 3.6) of our LOLOHA protocols. In addition, through extensive experiments with synthetic and real-world datasets, we have shown that our proposed LOLOHA protocols preserve competitive utility as state-of-the-art LDP protocols [5, 13, 23] by considerably minimizing longitudinal privacy loss (from 2 to 25 orders of magnitude with the experimented τ values). As future work, we intend to identify reasonable conditions of the input data (not constant as in [5, 23]) in which one can satisfy the standard ϵ-LDP definition. Besides, we intend to identify attack-based approaches to longitudinal LDP frequency estimation protocols (*e.g.*, data change detection or correlated data) and to extend the analysis of our LOLOHA protocols to the shuffle DP model. Last, we also aim to integrate LOLOHA to the `multi-freq-ldpy` package [6].

ACKNOWLEDGMENTS

The authors deeply thank the anonymous meta-reviewer and reviewers for their insightful suggestions. The work of Héber H. Arcolezi, Carlos Pinzón, and Catuscia Palamidessi was supported by the European Research Council (ERC) project HYPATIA under the European Union's Horizon 2020 research and innovation programme. Grant agreement n. 835294. Sébastien Gambs is supported by the Canada Research Chair program as well as a Discovery Grant from NSERC.

REFERENCES

[1] [n.d.]. LOLOHA repository. https://github.com/hharcolezi/LOLOHA.

[2] Jayadev Acharya, Ziteng Sun, and Huanyu Zhang. 2019. Hadamard Response: Estimating Distributions Privately, Efficiently, and with Little Communication. In *Proceedings of the Twenty-Second International Conference on Artificial Intelligence and Statistics (Proceedings of Machine Learning Research)*, Kamalika Chaudhuri and Masashi Sugiyama (Eds.), Vol. 89. PMLR, 1120–1129.

[3] Héber H. Arcolezi, Jean-François Couchot, Bechara Al Bouna, and Xiaokui Xiao. 2021. Random Sampling Plus Fake Data: Multidimensional Frequency Estimates With Local Differential Privacy. In *Proceedings of the 30th ACM International Conference on Information & Knowledge Management (CIKM '21)*. Association for Computing Machinery, New York, NY, USA, 47–57. https://doi.org/10.1145/3459637.3482467

[4] Héber H. Arcolezi, Jean-François Couchot, Bechara Al Bouna, and Xiaokui Xiao. 2021. Longitudinal Collection and Analysis of Mobile Phone Data with Local Differential Privacy. In *Privacy and Identity Management*, Michael Friedewald, Stefan Schiffner, and Stephan Krenn (Eds.). Springer International Publishing, Cham, 40–57. https://doi.org/10.1007/978-3-030-72465-8_3

[5] Héber H. Arcolezi, Jean-François Couchot, Bechara Al Bouna, and Xiaokui Xiao. 2022. Improving the utility of locally differentially private protocols for longitudinal and multidimensional frequency estimates. *Digital Communications and Networks* (2022). https://doi.org/10.1016/j.dcan.2022.07.003

[6] Héber H. Arcolezi, Jean-François Couchot, Sébastien Gambs, Catuscia Palamidessi, and Majid Zolfaghari. 2022. Multi-Freq-LDPy: Multiple Frequency Estimation Under Local Differential Privacy in Python. In *Computer Security – ESORICS 2022*, Vijayalakshmi Atluri, Roberto Di Pietro, Christian D. Jensen, and Weizhi Meng (Eds.). Springer Nature Switzerland, Cham, 770–775. https://doi.org/10.1007/978-3-031-17143-7_40

[7] Héber H. Arcolezi, Sébastien Gambs, Jean-François Couchot, and Catuscia Palamidessi. 2022. On the Risks of Collecting Multidimensional Data Under Local Differential Privacy. *arXiv preprint arXiv:2209.01684* (2022).

[8] Raef Bassily, Kobbi Nissim, Uri Stemmer, and Abhradeep Thakurta. 2017. Practical Locally Private Heavy Hitters. In *Proceedings of the 31st International Conference on Neural Information Processing Systems (NIPS'17)*. Curran Associates Inc., Red Hook, NY, USA, 2285–2293.

[9] Raef Bassily and Adam Smith. 2015. Local, Private, Efficient Protocols for Succinct Histograms. In *Proceedings of the Forty-Seventh Annual ACM Symposium on Theory of Computing (STOC '15)*. Association for Computing Machinery, New York, NY, USA, 127–135. https://doi.org/10.1145/2746539.2746632

[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 (SOSP '17)*. Association for Computing Machinery, New York, NY, USA, 441–459. https://doi.org/10.1145/3132747.3132769

[11] Konstantinos Chatzikokolakis, Catuscia Palamidessi, and Marco Stronati. 2014. A Predictive Differentially-Private Mechanism for Mobility Traces. In *Privacy Enhancing Technologies*, Emiliano De Cristofaro and Steven J. Murdoch (Eds.). Springer International Publishing, Cham, 21–41. https://doi.org/10.1007/978-3-319-08506-7_2

[12] Graham Cormode, Samuel Maddock, and Carsten Maple. 2021. Frequency estimation under local differential privacy. *Proceedings of the VLDB Endowment* 14, 11 (July 2021), 2046–2058. https://doi.org/10.14778/3476249.3476261

[13] 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.

[14] Frances Ding, Moritz Hardt, John Miller, and Ludwig Schmidt. 2021. Retiring Adult: New Datasets for Fair Machine Learning. In *Advances in Neural Information Processing Systems*, M. Ranzato, A. Beygelzimer, Y. Dauphin, P.S. Liang, and J. Wortman Vaughan (Eds.), Vol. 34. Curran Associates, Inc., 6478–6490.

[15] Dheeru Dua and Casey Graff. 2017. UCI Machine Learning Repository. http://archive.ics.uci.edu/ml, (accessed January 2023).

[16] John Duchi, Martin J Wainwright, and Michael I Jordan. 2013. Local Privacy and Minimax Bounds: Sharp Rates for Probability Estimation. In *Advances in Neural Information Processing Systems*, C.J. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K.Q. Weinberger (Eds.), Vol. 26. Curran Associates, Inc.

[17] John C. Duchi, Michael I. Jordan, and Martin J. Wainwright. 2013. Local Privacy and Statistical Minimax Rates. In *2013 IEEE 54th Annual Symposium on Foundations of Computer Science*. IEEE, 429–438. https://doi.org/10.1109/focs.2013.53

[18] Cynthia Dwork. 2006. Differential Privacy. In *Automata, Languages and Programming*, Michele Bugliesi, Bart Preneel, Vladimiro Sassone, and Ingo Wegener (Eds.). Springer Berlin Heidelberg, Berlin, Heidelberg, 1–12.

[19] Cynthia Dwork, Frank McSherry, Kobbi Nissim, and Adam Smith. 2006. Calibrating Noise to Sensitivity in Private Data Analysis. In *Theory of Cryptography*. Springer Berlin Heidelberg, 265–284. https://doi.org/10.1007/11681878_14

[20] Cynthia Dwork, Aaron Roth, et al. 2014. The algorithmic foundations of differential privacy. *Foundations and Trends® in Theoretical Computer Science* 9, 3–4 (2014), 211–407.

[21] Úlfar Erlingsson, Vitaly Feldman, Ilya Mironov, Ananth Raghunathan, Shuang Song, Kunal Talwar, and Abhradeep Thakurta. 2020. Encode, shuffle, analyze privacy revisited: Formalizations and empirical evaluation. *arXiv preprint arXiv:2001.03618* (2020).

[22] 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*. SIAM, 2468–2479.

[23] Úlfar Erlingsson, Vasyl Pihur, and Aleksandra Korolova. 2014. RAPPOR: Randomized Aggregatable Privacy-Preserving Ordinal Response. In *Proceedings of the 2014 ACM SIGSAC Conference on Computer and Communications Security*. ACM, New York, NY, USA, 1054–1067. https://doi.org/10.1145/2660267.2660348

[24] Vitaly Feldman, Jelani Nelson, Huy Nguyen, and Kunal Talwar. 2022. Private frequency estimation via projective geometry. In *Proceedings of the 39th International Conference on Machine Learning (Proceedings of Machine Learning Research)*, Kamalika Chaudhuri, Stefanie Jegelka, Le Song, Csaba Szepesvari, Gang Niu, and Sivan Sabato (Eds.), Vol. 162. PMLR, 6418–6433.

[25] Kirk Glerum, Kinshuman Kinshumann, Steve Greenberg, Gabriel Aul, Vince Orgovan, Greg Nichols, David Grant, Gretchen Loihle, and Galen Hunt. 2009. Debugging in the (Very) Large: Ten Years of Implementation and Experience. In *Proceedings of the ACM SIGOPS 22nd Symposium on Operating Systems Principles (SOSP '09)*. Association for Computing Machinery, New York, NY, USA, 103–116. https://doi.org/10.1145/1629575.1629586

[26] M. Emre Gursoy, Ling Liu, Ka-Ho Chow, Stacey Truex, and Wenqi Wei. 2022. An Adversarial Approach to Protocol Analysis and Selection in Local Differential Privacy. *IEEE Transactions on Information Forensics and Security* 17 (2022), 1785–1799. https://doi.org/10.1109/TIFS.2022.3170242

[27] Matthew Joseph, Aaron Roth, Jonathan Ullman, and Bo Waggoner. 2018. Local Differential Privacy for Evolving Data. In *Advances in Neural Information Processing Systems*, S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett (Eds.), Vol. 31. Curran Associates, Inc.

[28] Peter Kairouz, Keith Bonawitz, and Daniel Ramage. 2016. Discrete distribution estimation under local privacy. In *Int. Conf. on Machine Learning*. PMLR, 2436–2444.

[29] Peter Kairouz, Sewoong Oh, and Pramod Viswanath. 2016. Extremal mechanisms for local differential privacy. *The Journal of Machine Learning Research* 17, 1 (2016), 492–542.

[30] Shiva Prasad Kasiviswanathan, Homin K. Lee, Kobbi Nissim, Sofya Raskhodnikova, and Adam Smith. 2008. What Can We Learn Privately?. In *2008 49th Annual IEEE Symposium on Foundations of Computer Science*. IEEE, 531–540. https://doi.org/10.1109/FOCS.2008.27

[31] Gaoyuan Liu, Peng Tang, Chengyu Hu, Chongshi Jin, and Shanqing Guo. 2023. Multi-Dimensional Data Publishing with Local Differential Privacy. In *Proceedings of the 26th International Conference on Extending Database Technology, EDBT 2023, Ioannina, Greece, March 28 - March 31, 2023*. OpenProceedings.org, 183–194. https://doi.org/10.48786/edbt.2023.15

[32] Pathum Chamikara Mahawaga Arachchige, Peter Bertok, Ibrahim Khalil, Dongxi Liu, Seyit Camtepe, and Mohammed Atiquzzaman. 2020. Local Differential Privacy for Deep Learning. *IEEE Internet of Things Journal* 7, 7 (2020), 5827–5842. https://doi.org/10.1109/JIOT.2019.2952146

[33] Moni Naor and Neil Vexler. 2020. Can Two Walk Together: Privacy Enhancing Methods and Preventing Tracking of Users. In *1st Symposium on Foundations of Responsible Computing (FORC 2020) (Leibniz International Proceedings in Informatics (LIPIcs))*, Aaron Roth (Ed.), Vol. 156. Schloss Dagstuhl–Leibniz-Zentrum für Informatik, Dagstuhl, Germany, 4:1–4:20. https://doi.org/10.4230/LIPIcs.FORC.2020.4

[34] Thông T. Nguyên, Xiaokui Xiao, Yin Yang, Siu Cheung Hui, Hyejin Shin, and Junbum Shin. 2016. Collecting and Analyzing Data from Smart Device Users with Local Differential Privacy. *ArXiv* abs/1606.05053 (2016).

[35] Olga Ohrimenko, Anthony Wirth, and Hao Wu. 2022. Randomize the Future: Asymptotically Optimal Locally Private Frequency Estimation Protocol for Longitudinal Data. In *Proceedings of the 41st ACM SIGMOD-SIGACT-SIGAI Symposium on Principles of Database Systems (PODS '22)*. Association for Computing Machinery, New York, NY, USA, 237–249. https://doi.org/10.1145/3517804.3526226

[36] Jun Tang, Aleksandra Korolova, Xiaolong Bai, Xueqiang Wang, and Xiaofeng Wang. 2017. Privacy loss in apple's implementation of differential privacy on macos 10.12. *arXiv preprint arXiv:1709.02753* (2017).

[37] Apple Differential Privacy Team. 2017. Learning with privacy at scale. https://docs-assets.developer.apple.com/ml-research/papers/learning-with-privacy-at-scale.pdf, (accessed January 2023).

[38] Gatha Varma, Ritu Chauhan, and Dhananjay Singh. 2022. Sarve: synthetic data and local differential privacy for private frequency estimation. *Cybersecurity* 5, 26 (2022). https://doi.org/10.1186/s42400-022-00129-6

[39] Ning Wang, Xiaokui Xiao, Yin Yang, Jun Zhao, Siu Cheung Hui, Hyejin Shin, Junbum Shin, and Ge Yu. 2019. Collecting and Analyzing Multidimensional Data with Local Differential Privacy. In *2019 IEEE 35th International Conference on Data Engineering (ICDE)*. IEEE, 638–649. https://doi.org/10.1109/ICDE.2019.00063

[40] Tianhao Wang, Jeremiah Blocki, Ninghui Li, and Somesh Jha. 2017. Locally Differentially Private Protocols for Frequency Estimation. In *26th USENIX Security Symposium (USENIX Security 17)*. USENIX Association, Vancouver, BC, 729–745.

[41] Stanley L. Warner. 1965. Randomized Response: A Survey Technique for Eliminating Evasive Answer Bias. *J. Amer. Statist. Assoc.* 60, 309 (March 1965), 63–69. https://doi.org/10.1080/01621459.1965.10480775

[42] Qiao Xue, Qingqing Ye, Haibo Hu, Youwen Zhu, and Jian Wang. 2022. DDRM: A Continual Frequency Estimation Mechanism with Local Differential Privacy. *IEEE Transactions on Knowledge and Data Engineering* (2022), 1–1. https://doi.org/10.1109/TKDE.2022.3177721

[43] Xingyu Zhou and Jian Tan. 2021. Local Differential Privacy for Bayesian Optimization. In *Proceedings of the AAAI Conference on Artificial Intelligence*, Vol. 35. 11152–11159.