Continual Counting with Gradual Privacy Expiration

Joel Daniel Andersson
Basic Algorithms Research Copenhagen
University of Copenhagen
jda@di.ku.dk

Monika Henzinger
Institute of Science and Technology Austria
Klosterneuburg, Austria
monika.henzinger@ist.ac.at

Rasmus Pagh
Basic Algorithms Research Copenhagen
University of Copenhagen
pagh@di.ku.dk

Teresa Anna Steiner
University of Southern Denmark
steiner@imada.sdu.dk

Jalaj Upadhyay
Rutgers University
jalaj.upadhyay@rutgers.edu

Abstract

Differential privacy with gradual expiration models the setting where data items arrive in a stream and at a given time t the privacy loss guaranteed for a data item seen at time $(t-d)$ is $\varepsilon g(d)$, where g is a monotonically non-decreasing function. We study the fundamental *continual (binary) counting* problem where each data item consists of a bit, and the algorithm needs to output at each time step the sum of all the bits streamed so far. For a stream of length T and privacy *without* expiration continual counting is possible with maximum (over all time steps) additive error $O(\log^2(T)/\varepsilon)$ and the best known lower bound is $\Omega(\log(T)/\varepsilon)$; closing this gap is a challenging open problem.

We show that the situation is very different for privacy with gradual expiration by giving upper and lower bounds for a large set of expiration functions g. Specifically, our algorithm achieves an additive error of $O(\log(T)/\varepsilon)$ for a large set of privacy expiration functions. We also give a lower bound that shows that if C is the additive error of any ε-DP algorithm for this problem, then the product of C and the privacy expiration function after $2C$ steps must be $\Omega(\log(T)/\varepsilon)$. Our algorithm matches this lower bound as its additive error is $O(\log(T)/\varepsilon)$, even when $g(2C) = O(1)$.

Our empirical evaluation shows that we achieve a slowly growing privacy loss with significantly smaller empirical privacy loss for large values of d than a natural baseline algorithm.

1 Introduction

Differential privacy under continual observation [8, 16] has seen a renewed interest recently [2, 3, 15, 21, 25, 26, 27, 24] due to its application in private learning [10, 11, 13, 14, 34] and statistics [6, 7, 19, 22, 30, 37, 40, 41, 20, 32]. In this model, the curator gets the database in the form of a stream and is required to output a given statistic continually. Chan et al. [8] and Dwork et al. [16] introduced the *binary (tree) mechanism* which allows us to estimate the running count of a binary stream of length T with additive error $O(\log^2(T)/\varepsilon)$ under ε-differential privacy.

The traditional definition of continual observation considers every single entry in the stream equally important for analysis and has equal confidentiality. However, in many applications of continual observation, the data becomes less sensitive with time. For example, consider the case where the

38th Conference on Neural Information Processing Systems (NeurIPS 2024).

https://doi.org/10.52202/079017-0299

stream tracks visits to a certain location or website: it being visited a minute ago may constitute more sensitive information than if it was visited a week ago. To capture such scenarios, Bolot et al. [4] defined *privacy with expiration*, where the privacy of a streamed data item decreases as a function of the time elapsed since it was streamed. However, known algorithms for privacy with expiration work only in the setting when we expect no privacy after a certain time has elapsed [4, Section 6].

This lack of algorithms for privacy with expiration influences some real-world design choices [1]. In particular, real-world deployments either allocate every user a "privacy budget" that is diminished every time their data is used, such that their data should not be used once the privacy budget reaches zero, or they account for privacy loss over the time. However, since the data can still be useful, another common approach in these deployments use the heuristic of "refreshing the privacy budget", i.e., the privacy budget is reset to a positive default value after a prescribed time period, irrespective of how much privacy budget has been used so far. This, for example, was pointed out by Tang et al. [38] in Apple's first large-scale deployment. However, refreshing the privacy budget is very problematic as the privacy loss is, in the worst case, multiplied by the number of refreshes, for example, if the old data is reused (i.e., the privacy expiration is *linear*).

In this paper, we study continual counting with gradual privacy expiration, generalizing the result in Bolot et al. [4]. Our main contributions are algorithms with the following assets:

- *Improve accuracy.* We achieve an additive error of $O(\log(T)/\varepsilon)$ for a large class of privacy expiration functions and show that this is optimal in a particular sense. This is in contrast to continual counting without expiration, where there is a gap of a $\log T$ factor [16]. Our work generalizes the $\Omega(\log(T)/\varepsilon)$ lower bound for continual counting to a wide class of privacy expiration functions and shows that for any additive error C, the product of C and the privacy expiration function after $2C$ steps must be $\Omega(\log T)$. We match this lower bound as our additive error is $O(\log(T)/\varepsilon)$, even when the expiration function after $2C$ steps is a constant.

- *Scale well.* Our algorithms work for unbounded streams, run in amortized $O(1)$ time per update, $\log(T)$ space, and offer different trade-offs than conventional continual counting algorithms. In allowing for a growing privacy loss, we show that polylogarithmic privacy expiration is sufficient for optimal additive error, and parameterize the algorithm by the speed of the privacy expiration; as expected, faster privacy expiration yields a smaller error.

We supplement these theoretical guarantees with empirical evaluations.

Related Works. Before presenting our contributions in detail, we give a brief overview of the most relevant related work. Since Chan et al. [8] and Dwork et al. [16], several algorithms have been proposed for privately estimating *prefix-sum under continual observation*, i.e., given a stream of inputs $x_1, x_2, \ldots$ from some domain $\mathcal{X}$, output $y_t = \sum_{i \le t} x_i$ for all $t \ge 1$. *Continual binary counting* is a special case of prefix sum when $\mathcal{X} = \{0, 1\}$ and x_t is provided at time t.

When the input is given as a stream, earlier works improved on the basic binary mechanism under (i) distributional assumptions on data [35], (ii) structural assumptions on data [36], and (iii) that the importance of data (both with respect to utility and sensitivity) decreases with elapsed time [4], or (iv) by enforcing certain conditions on the behavior of the output [9]. In recent work, Fichtenberger et al. [21] gave algorithms to improve the worst-case non-asymptotic guarantees under continual observation using the *matrix mechanism* [33] and Denisov et al. [14] used similar approach to provide empirical results that minimize the mean-squared error. Subsequently, Henzinger et al. [25] showed that the algorithm in Fichtenberger et al. [21] achieves almost optimal mean-squared error.

These earlier works are in the traditional definition of privacy under continual observation, i.e., they consider data privacy to be constant throughout the stream. The only exception is the work of Bolot et al. [4], which defined differentially private continual release with privacy expiration parameterized by a monotonically non-decreasing function g and gave an algorithm for the special case that the data loses all its confidentially after a prescribed time. Our work is in this privacy model. There is another line of work motivated by applications in private learning that studies privacy-preserving prefix sum without restricting access to the data points (such as allowing multiple passes) [13, 31] and providing privacy-preserving estimates under various privacy models like shuffling [12]. Since we focus on continual observation, we do not compare our results with this line of work.

The work whose techniques are the most related to ours is the algorithm in Dwork et al. [16] for continual counting satisfying pan-privacy [17]. Roughly speaking, an algorithm is *pan-private* if it is resilient against intruders who can observe snapshots of the internal states of the algorithms.

1.1 Our Contributions

We start by first formally stating the problem. As mentioned above, the focus of this work is privacy with expiration given as Definition 3 in Bolot et al.[4]:

Definition 1.1. *Let $g : \mathbb{N} \to \mathbb{R}_{\geq 0}$ be a non-decreasing function*[1]*. Let $\mathcal{A}$ be a randomized online algorithm that takes as an input a stream $x_1, x_2, \ldots$ and at every time step t outputs $\mathcal{A}(x_1, \ldots, x_t)$. $\mathcal{A}$ satisfies* ε-differential privacy with expiration (function) g *if for all $\tau \geq 1$, for all measurable $S \subseteq \text{range}(\mathcal{A})^*$* [2]*, all possible inputs $x_1, \ldots, x_\tau$, all $j \leq \tau$ and all x'_j with $|x_j - x'_j| \in [0, 1]$*

$$\Pr[(\mathcal{A}(x_1, \ldots, x_j, \ldots, x_t))_{t=1}^{\tau} \in S] \quad \leq e^{g(\tau-j)\varepsilon} \Pr[(\mathcal{A}(x_1, \ldots, x'_j, \ldots, x_t))_{t=1}^{\tau} \in S],$$

where the probability is over the coin throws of $\mathcal{A}$. We refer to $g(\tau - j) \cdot \varepsilon$ as the privacy loss.

Letting T be the length of the input stream, the best known bound on the ℓ_∞-error for continual counting under ε-differential privacy is $O(\log^2(T)/\varepsilon)$, achieved by the algorithms in [16, 8]. Alternatively, the analysis of [8] can be used to show that running this algorithm with $\varepsilon' = \varepsilon \log(T)$ achieves ε-differential privacy with expiration function $g(d) = \log(T)$ for all $d = 1, \ldots, T$, and error $O(\log(T)/\varepsilon)$. Our main contribution is to show that better trade-offs are possible: In particular, we can achieve the same error with a *strictly smaller function* g, i.e. we can get an $O(\log(T)/\varepsilon)$ bound on the ℓ_∞-error with an expiration function of $g(d) \approx \log d$. More generally, our algorithm provides a trade-off between privacy loss and both ℓ_∞-error and expected ℓ_2^2-error for all expiration functions $f(d)$ that satisfy (roughly) $f(d) \geq 1 + \log^\lambda(d)$ for any $\lambda > 0$. The exact expiration function g is stated below in Theorem 1.2. It also includes a parameter B that allows the privacy loss to be "shifted" by B time steps, i.e., there is no privacy loss in the first B time steps. If the length T of the stream is unknown, then B is a constant. If T is given to the algorithm, then B can be a function of T.

By Definition 1.1, any algorithm satisfying differential privacy with expiration g also fulfills differential privacy with any expiration function that is pointwise at least as large as g. Specifically, for two functions f and g defined on the same domain $\mathcal{D}$, we say $f \succeq g$ if $f(x) \geq g(x)$ for all $x \in \mathcal{D}$. We are now ready to state our main theorem:

Theorem 1.2. *Let $\lambda \in \mathbb{R}_{>0} \backslash \{\frac{3}{2}\}$ be a constant, and let parameters $\varepsilon \in \mathbb{R}_{>0}$ and $B \in \mathbb{N}$ be given. There exists an algorithm $\mathcal{A}$ that approximates prefix sums of a (potentially unbounded) input sequence $x_1, x_2, \ldots$ with $x_i \in [0, 1]$ satisfying ε-differential privacy with any expiration function f such that $f \succeq g$, where*

$$g(d) = \begin{cases} 0 & \text{for } d < B \\ O(1 + \log^\lambda(d - B + 1)) & \text{for } d \geq B \end{cases}$$

Considering all releases up to and including input t, the algorithm $\mathcal{A}$ uses $O(B + \log t)$ space and $O(1)$ amortized time per input/output pair and has the following error guarantees at each individual time step t for $\beta > 0$,

- $\mathbb{E}_{\mathcal{A}}\left[(\mathcal{A}(x) - \sum_{i=1}^{t} x_i)^2\right] = O(B^2 + \log^{3-2\lambda}(t)/\varepsilon^2)$,
- $|\mathcal{A}(x) - \sum_{i=1}^{t} x_i| = O(B + \log^q(t)\sqrt{\log(1/\beta)}/\varepsilon)$ *with probability* $1 - \beta$ *where* $q = \max(1/2, 3/2 - \lambda)$.

The case when $\lambda \in \{0, 3/2\}$ is covered in Appendix C.3. Note that choosing $\lambda > 3/2$ implies a constant expected squared error at each time step if $B = O(1)$. Parameter λ controls the trade-off between the *asymptotic* growth of the expiration function and the error, while ε controls the trade-off between *initial* privacy (after B time steps which is $\varepsilon \cdot g(B)$) and the error, which is inversely proportional to ε. Also, for releasing T outputs we have the following corollary.

[1] We believe g being *non-increasing* is a typo in Bolot et al. [4].

[2] "$*$" is the Kleene operator: given a set Σ, Σ^* is the (infinite) set of all finite sequences of elements in Σ.

https://doi.org/10.52202/079017-0299

Corollary 1.3. *The algorithm $\mathcal{A}$ with $B = O(\log(T)/\varepsilon)$ and $\lambda \geq 1$ incurs a maximum (over all time steps) additive ℓ_∞-error of $O(\log(T)/\varepsilon)$ when releasing T outputs with probability $1 - 1/T^c$, for constant $c > 0$, and achieves privacy with expiration function g as in Theorem 1.2.*

In Section 6, we provide empirical evidence to show that we achieve a significantly smaller empirical privacy loss than a natural baseline algorithm. Finally, we complement our upper bound with the following lower bound shown in Appendix C.4.

Theorem 1.4. *Let $\mathcal{A}$ be an algorithm for binary counting for streams of length T which satisfies ε-differential privacy with expiration h. Let C be an integer such that $\mathcal{A}$ incurs a maximum additive error of at most $C < T/2$ over T time steps with a probability of at least $2/3$. Then*

$$2C \cdot h(2C-1) \geq \frac{\log(T/(6C))}{2\varepsilon} \ .$$

Note that Theorem 1.4 gives a lower bound for $h(j)$ for a specific j, namely $j = 2C - 1$, and as h is non-decreasing by Definition 1.1, the lower bound also holds for all $h(j')$ with $j' \geq j$.

Note that Theorem 1.4 shows that our algorithm in Corollary 1.3 achieves a tight error bound for the expiration functions $\lambda \geq 1$ and $B = O(\log(T)/\varepsilon)$. Assume $\mathcal{A}'$ is an algorithm that approximates prefix sums in the continual setting and which satisfies differential privacy with expiration function h and maximum error $C \leq B/2 + 1$ at all time steps with probability at least 2/3 for an even B. When run on a binary input sequence, $\mathcal{A}'$ solves the binary counting problem. Thus, by Theorem 1.4, we have that $2C \cdot h(B+1) \geq 2C \cdot h(2C-1) \geq \frac{\log(T/(6C))}{2\varepsilon} \geq \frac{\log(T/(3B+6))}{2\varepsilon} = \Omega(\frac{\log T}{\varepsilon})$.

Now consider the algorithm $\mathcal{A}$ given in Corollary 1.3 and note that, by definition of the expiration function g, $g(B+1) = O(1)$ and that Corollary 1.3 shows that $C = O(\log(T)/\varepsilon)$. This is tight as Theorem 1.4 shows that for such an expiration function $C = \Omega(\log(T)/\varepsilon)$.

1.2 Technical Overview

Central to our work is the event-level pan-private algorithm for continual counting by Dwork et al. [16]. Similarly to the binary tree algorithm of Dwork et al. [8], a noise variable z_I is assigned to every *dyadic interval* I (see Section 3.2 for a formal definition) contained in $[0, T-1]$. Let this set of dyadic intervals be called $\mathcal{I}$. In the version of the binary tree algorithm of Chan et al. [8], the noise added to the sum of the values so far (i.e. the non-private output) at any time step $1 \leq t \leq T$ is equal to $\sum_{I \in D_{[0,t-1]}} z_I$, where $D_{[0,t-1]} \subseteq \mathcal{I}$ is the dyadic interval decomposition of the interval $[0, t-1]$. The pan-private algorithm adds different noise to the output: it adds at time t the *noises for all intervals containing* $t-1$, i.e., $\sum_{I \in \{I \in \mathcal{I} \,:\, t-1 \in I\}} z_I$. This pan-private way of adding noise helps us bound the privacy loss under expiration. For two neighboring streams differing at time step j, we can get the same output at $\tau \geq j$ by shifting the values of the noises of a set of disjoint intervals covering $[j, \tau]$ each by at most 1. Using that $|D_{[j,\tau]}| = O(\log(\tau - j + 1))$, we show that the algorithm satisfies a logarithmic privacy expiration.

In our algorithm, we make four changes to the above construction (i.e., the pan-private construction in Dwork et al. [16]): **(i)** We do not initialize the counter with noise separately from that introduced by the intervals. **(ii)** We split the privacy budget unevenly across the levels of the dyadic interval set instead of uniformly allocating it. This allows us to control the asymptotic growth of the expiration function, and the error. This change, however, requires a more careful privacy analysis. **(iii)** At time t we add noise identified by intervals containing t, not $t-1$. While this is a subtle difference, it allows us to exclude intervals starting at 0 from $\mathcal{I}$, leading to our algorithm running on *unbounded* streams with utility that depends on the current time step t. Said differently, our algorithm does not need to know the stream length in advance. This is in contrast to Dwork et al. [16] where the construction requires an upper bound T on the length of the stream so that the utility guarantee at each step is fixed and a function of T. **(iv)** We allow for a delay of B, meaning we output 0 for the initial B steps. This gives perfect privacy for the first B steps, and, since each element of the stream is in $[0, 1]$, the delayed start leads only to an additive error of $O(B)$.

2 Preliminaries

Let $\mathbb{N}_{>0}$ denote the set $\{1, 2, \cdots\}$ and $\mathbb{R}_{\geq 0}$ the set of non-negative real numbers. We use the symbol g to denote the function that defines the privacy expiration, i.e., $g : \mathbb{N} \to \mathbb{R}_{\geq 0}$. We fix the symbol

$\mathcal{A}(x)$ to denote the randomized streaming algorithm that, given an input $x = x_1, x_2, \ldots$ as a stream, provides ε-differential privacy with expiration g. All algorithms *lazily draw noise*, meaning that a "noise" random variable is only drawn when first used and is re-used and *not* re-drawn when referenced again. For a random variable, Z, we use $\mathsf{support}(Z)$ to denote its support. For a sequence of random variables $Z = Z_1, Z_2, \ldots$, we use $\mathrm{support}(Z)$ to denote $\mathrm{support}(Z_1) \times \mathrm{support}(Z_2) \times \ldots$.

Helpful lemmas. We now collect some helpful lemmas shown formally in Appendix C. To show privacy with expiration in the following sections, we repeatedly use the following observation: a similar lemma has been used to show the standard definition of differential privacy, e.g., in the proof of Theorem 2.1 of Dwork et al. [18]. Informally, it says that if there exists a map q between random choices made by algorithm $\mathcal{A}$ such that for any input x and fixed sequence z of random choices, the map returns a sequence $q(z)$ such that (1) the output $\mathcal{A}(x, z)$ equals the output $\mathcal{A}(x', q(z))$ and (2) the probability of picking z is similar to the probability of picking $q(z)$, then $\mathcal{A}$ is private. The notion of "similar probability" is adapted to the definition of differential privacy with expiration and depends on the function g. All the results in this section are shown formally in Appendix C.1:

Fact 2.1. *Consider an algorithm $\mathcal{A}$ that uses a sequence of random variables $Z = Z_1, Z_2, \ldots$ as the only source of randomness. We can model $\mathcal{A} : \chi \times \mathrm{support}(Z)$ as a (deterministic) function of its actual input from the universe χ and the sequence of its random variables Z. Suppose that for all $\tau \in \mathbb{N}_{>0}$, $j \leq \tau$ and all neighboring pairs of input streams $x = x_1, \ldots, x_j, \ldots, x_\tau$ and $x' = x_1, \ldots, x'_j, \ldots, x_\tau$, there exists a function $q : \mathrm{support}(Z) \to \mathrm{support}(Z)$ such that $\mathcal{A}(x; z) = \mathcal{A}(x'; q(z))$ and*

$$\Pr_{z \sim Z}[z \in \mathcal{N}] \leq e^{\varepsilon g(\tau - j)} \Pr_{z' \sim Z}[z' \in q(\mathcal{N})] \quad \textit{for all} \quad \mathcal{N} \subseteq \mathrm{support}(Z).$$

Then $\mathcal{A}$ satisfies ε-differential privacy with expiration g.

Lemma 2.2. *Let $Z = Z_1, Z_2, \ldots, Z_k$ be a sequence of independent Laplace random variables, such that $Z_i \sim \mathrm{Lap}(b_i)$ for $b_i > 0$, for all $i \in [k]$. Let q be a bijection $q : \mathrm{support}(Z) \to \mathrm{support}(Z)$ of the following form: For all $\Delta := (\Delta_1 \quad \Delta_2 \quad \cdots \quad \Delta_k) \in \mathbb{R}^k$, and for all $z \in \mathrm{support}(Z)$, we have $q(z) = z + \Delta \in \mathrm{support}(Z)$. Then for all $\mathcal{N} \subseteq \mathrm{support}(Z)$ we have*

$$\Pr_{z \in Z}[z \in \mathcal{N}] \leq e^s \Pr_{z \in Z}[z \in q(\mathcal{N})], \quad \textit{where} \quad s = \sum_{i=1}^{k} \frac{|\Delta_i|}{b_i}.$$

3 Warmup

As a warm-up, we give two simple algorithms for two obvious choices of the expiration function: the linear expiration function $g(d) = d$ and the logarithmic expiration function $g(d) = 2\log(d + 1) + 2$.

3.1 A Simple Algorithm with Linear Privacy Expiration

First, we consider a simple algorithm which gives ε-differential privacy with expiration $g : \mathbb{N} \to \mathbb{R}_{\geq 0}$, where $g(d) = d$. The maximum error of this algorithm over T time steps is bounded by $O(\varepsilon^{-1} \log(T/\beta))$, with probability at least $1 - \beta$. The algorithm $\mathcal{A}_{\mathsf{simple}}$ is given in Algorithm 1. It adds fresh Laplace noise to any output sum. Note that this is the same algorithm as the Simple Counting Mechanism I from Chan et al. [8]. However, we show that for the weaker notion of differential privacy with linear expiration, Laplace noise with *constant* scale suffices, even though the sensitivity of $\mathcal{A}_{\mathsf{simple}}$ running on a stream of length T is T. To prove this, we show that for two neighbouring streams differing at time step j, we obtain the same output by "shifting" the values of the Laplace noises for all outputs after step j by at most 1. We defer the proof of the following lemma to Appendix C.2.

Lemma 3.1. *The algorithm $\mathcal{A}_{\mathsf{simple}}$, given in Algorithm 1, is ε-differentially private with expiration g, where $g : \mathbb{N} \to \mathbb{N}$ is the identity function $g(d) = d$ for all $d \in \mathbb{N}$. It incurs a maximum additive error of $O(\varepsilon^{-1} \log(T/\beta))$ over all T time steps simultaneously with probability at least $1 - \beta$.*

https://doi.org/10.52202/079017-0299

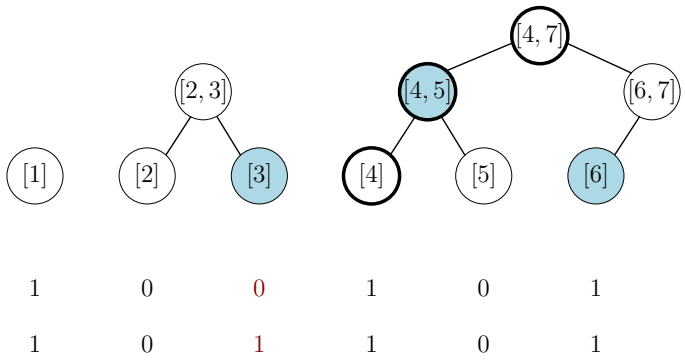


Figure 1: An example of the noise structure for Algorithm 2 and Algorithm 3 for $B = 0$ on two neighbouring streams x and y differing in position 3. The nodes correspond to the dyadic intervals. The filled nodes mark the intervals I for which the noise Z_I is shifted by one between x and y to get the same outputs for $\tau = 6$. The fat nodes mark the intervals I corresponding to the Z_I which are used in the computation of the fourth prefix sum s_4.

Algorithm 1 $\mathcal{A}_{\mathsf{simple}}$: Continual counting under linear gradual privacy expiration

Input: A stream $x_1, x_2, \cdots$, privacy parameter ε
Lazily[3]draw $Z_{t-1} \sim \mathrm{Lap}\left(\frac{1}{\varepsilon}\right)$
At time $t = 1$, output 0
for $t = 2$ **to** ∞ **do**
    At time t, output $\sum_{i=1}^{t-1} x_i + Z_{t-1}$
end for

Algorithm 2 $\mathcal{A}_{\mathsf{log}}$: Continual counting under logarithmic gradual privacy expiration

Input: A stream $x_1, x_2, \cdots$, privacy param ε
Let $\mathcal{I}$ be the dyadic interval set on $[1, \infty)$
$\forall I \in \mathcal{I}$, lazily draw i.i.d. $Z_I \sim \mathrm{Lap}\left(\frac{1}{\varepsilon}\right)$
Set $\mathcal{I}_t = \{I \in \mathcal{I} : t \in I\}$
for $t = 1$ **to** ∞ **do**
    At time t, output $\sum_{i=1}^{t} x_i + \sum_{I \in \mathcal{I}_t} Z_I$
end for

3.2 A Binary-Tree-Based Algorithm with Logarithmic Privacy Expiration

Next, we show how an algorithm similar to the binary tree algorithm [16] gives ε-differential privacy with expiration $g : \mathbb{N} \to \mathbb{R}_{\geq 0}$, where $g(d) = 2\log(d+1) + 2$. This result can also be derived from Theorem 1.2 by setting $\lambda = 1$ and $B = 0$. As in the case when $g(d) = d$, the maximum error of this algorithm over T time steps is again bounded by $O(\varepsilon^{-1}\log(T/\beta))$, with probability at least $1 - \beta$. Similarly to the binary tree algorithm, we define a noise variable for every node in the tree, but we do this in the terminology of *dyadic intervals*. We consider the *dyadic interval set* $\mathcal{I}$ on $[1, \infty)$ (formally defined shortly), associate a noise variable z_I with each interval $I \in \mathcal{I}$, and at time step t add noise z_I for each $I \in \mathcal{I}$ that contains t. This is similar to the construction in Dwork et al. [16], with the exception that they instead consider the dyadic interval set on $[0, T-1]$, add noise z_I at time t if $t - 1 \in I$, and initialize their counter with noise from the same distribution. Our choice of $\mathcal{I}$ allows the algorithm to run on unbounded streams, and leads to adding up $1 + \lfloor \log(t) \rfloor$ noise terms at step t rather than $1 + \lfloor \log(T) \rfloor$. For privacy, we will argue that if two streams differ at time j, then we get the same outputs up to time $\tau \geq j$ by considering a subset of disjoint intervals in $\mathcal{I}$ covering $[j, \tau]$, and shifting the associated Laplace random variables appropriately. In the following, we describe this idea in detail. We start by describing the dyadic interval decomposition of an interval.

Dyadic interval decomposition. For every non-negative integer ℓ, we divide $[1, \infty)$ into disjoint intervals of length 2^ℓ: $\mathcal{I}^\ell = \{[k \cdot 2^\ell, (k+1) \cdot 2^\ell - 1], k \in \mathbb{N}_{>0}\}$. We call $\mathcal{I} = \bigcup_{\ell=0}^{\infty} \mathcal{I}^\ell$ the *dyadic interval set* on $[1, \infty)$, and $\mathcal{I}^\ell$ the ℓ-th *level* of the dyadic interval set. We show the following two facts in Appendix C.

Fact 3.2. *Let $\mathcal{I}$ be the dyadic interval set on $[1, \infty)$. For any interval $[a, b]$, $1 \leq a \leq b$, there exists a set of intervals $D_{[a,b]} \subseteq \mathcal{I}$, referred to as the* dyadic interval decomposition *of $[a, b]$, such that (i) the sets in $D_{[a,b]}$ are disjoint; (ii) $\bigcup_{I \in D_{[a,b]}} I = [a, b]$; and (iii) $D_{[a,b]}$ contains at most 2 intervals per level, and the highest level ℓ of an interval satisfies $\ell \leq \log(b - a + 1)$*

[3]By "lazily", we mean we draw a random variable the first time it is used by the algorithm.

Fact 3.3. *Let $\mathcal{I}$ be the dyadic interval set on $[1,\infty)$, and for $t \in \mathbb{N}_{>0}$ define $\mathcal{I}_t = \{I \in \mathcal{I} : t \in I\}$ as the* intersection *of t with $\mathcal{I}$. Then $|\mathcal{I}_t| = \lfloor \log t \rfloor + 1$.*

Lemma 3.4. *The algorithm $\mathcal{A}_{\mathsf{log}}$ given in Algorithm 2 satisfies ε-differential privacy with expiration g, where $g : \mathbb{N} \to \mathbb{R}$ is defined as $g(x) = 2\log(x+1)+2$. It incurs a maximum additive error of $O(\varepsilon^{-1}\log(T/\beta))$ over all T time steps simultaneously with probability at least $1-\beta$.*

Privacy. We use Fact 2.1 and Lemma 2.2 to argue privacy of $\mathcal{A}_{\mathsf{log}}$: Let x and x' differ at time j. Note that the prefix sums fulfill the following properties: (i) $\sum_{i=1}^{t} x_i = \sum_{i=1}^{t} x'_i$ for all $t < j$ and (ii) $\sum_{i=1}^{t} x'_i = \sum_{i=1}^{t} x_i + y$ for all $t \geq j$, where $y = x'_j - x'_j \in [-1,1]$.

In the following, we refer to the output of the algorithm run on input x and with values of the random variables z as $\mathcal{A}_{\mathsf{log}}(x;z)$. Let $\tau \geq j$ be given, and consider $S \subseteq \mathrm{range}(\mathcal{A}_{\mathsf{log}})^*$. Let $Z = (Z_I)_{I\in\mathcal{I}}$ be the sequence of Laplace random variables used by the algorithm. For any fixed output sequence $s \in S$, let $z = (z_I)_{I\in\mathcal{I}}$ be a sequence of values that the Laplace random variables need to assume to get output sequence s for input x. That is $\sum_{i=1}^{t} x_i + \sum_{I\in\mathcal{I}_t} z_I = s_t$ for $t \geq 1$. Let $D_{[j,\tau]}$ be the decomposition of $[j,\tau]$ as defined in Fact 3.2. We define a bijection q satisfying the properties of Fact 2.1 as follows: $q(z) = z' = (z'_I)_{I\in\mathcal{I}}$ such that

$$z'_I = \begin{cases} z'_I = z_I & \forall I \notin D_{[j,\tau]} \\ z'_I = z_I + y & \forall I \in D_{[j,\tau]} \end{cases}.$$

We show the two properties needed to apply Fact 2.1:

(1) Note that $(\mathcal{A}_{\mathsf{log}}(x;z))_t = \sum_{i=1}^{t} x_i + \sum_{I\in\mathcal{I}_t} z_I$. For $t < j$, we have $t \notin [j,\tau]$ and therefore $t \notin I$ for any $I \in D_{[j,\tau]}$. Therefore, we have

$$\sum_{i=1}^{t} x_i + \sum_{I\in\mathcal{I}_t} z_I = \sum_{i=1}^{t} x'_i + \sum_{I\in\mathcal{I}_t} z_I = \sum_{i=1}^{t} x'_i + \sum_{I\in\mathcal{I}_t} z'_I.$$

For $j \leq t \leq \tau$, we have that t is contained in exactly one $I \in D_{[j,\tau]}$. Thus, $z'_I = z_I + y$ for exactly one $I \in \mathcal{I}_t$, and $z'_{I'} = z_{I'}$ for all $I' \in \mathcal{I}_t \backslash \{I\}$. Further, since $t \geq j$, we have that $\sum_{i=1}^{t} x_i = \sum_{i=1}^{t} x'_i - y$. Together, this shows the first property of Fact 2.1 as

$$\sum_{i=1}^{t} x_i + \sum_{I\in\mathcal{I}_k} z_I = \sum_{i=1}^{t} x'_i - y + \sum_{I\in\mathcal{I}_t} z'_I + y = \sum_{i=1}^{t} x'_i + \sum_{I\in\mathcal{I}_t} z'_I,$$

(2) By Fact 3.2, $|D_{[j,\tau]}| \leq 2(\log(\tau - j + 1) + 1)$. Thus, by Lemma 2.2 for any $\mathcal{N} \in \mathrm{support}(Z)$,

$$\Pr_{z\in Z}[z \in \mathcal{N}] \leq e^{\sum_{I\in D_{[j,\tau]}} |y|\varepsilon} \Pr_{z\in Z}[z \in q(\mathcal{N})] \leq e^{2(\log(\tau-j+1)+1)\varepsilon} \Pr_{z\in Z}[z \in q(\mathcal{N})],$$

so the second property of Fact 2.1 is fulfilled with $g(x) = 2\log(x+1)+2$. By Fact 2.1, we have differential privacy with privacy expiration $g(x) = 2\log(x+1)+2$.

Accuracy. To show accuracy at step t, let $Y_t = \sum_{I\in\mathcal{I}_t} Z_I$, i.e. the noise added at time step t. By Fact 3.3 we add $k = |\mathcal{I}_t| = \lfloor \log t \rfloor + 1$ Laplace noises with scale $\frac{1}{\varepsilon}$. Let $M_{t,\beta} = \max\left\{\sqrt{\lfloor \log t \rfloor + 1}, \sqrt{\ln(2/\beta)}\right\}$. By Corollary B.4, we have that $\Pr\left[|Y_t| > \frac{2}{\varepsilon}\sqrt{2\ln(2/\beta)}M_{t,\beta}\right] \leq \beta$, for any $\beta < 1$. Setting $\beta = \beta'/T$, it follows that with probability at least $1-\beta'$, $|Y_t| = O(\varepsilon^{-1}\log(T/\beta'))$ for all time steps $t \leq T$ simultaneously.

4 Proof of Theorem 1.2 and Corollary 1.3

Section 3.2 shows that we can obtain an error smaller than the binary mechanism by using differential privacy with a logarithmic expiration function. Here we show a general trade-off between the expiration function's growth and the error's growth. Two techniques are needed to show the theorem:

Delay. All outputs are delayed for B steps. That is, in the first B steps, the mechanism outputs 0, thereafter, it outputs a private approximation of $\sum_{i=1}^{j-B} x_i$. Delay introduces an extra additive error of up to B, but ensures perfect privacy for the first B time steps after receiving an input.

https://doi.org/10.52202/079017-0299

Algorithm 3 Continual counting, gradual privacy expiration

Input: A stream $x_1, x_2 \cdots$, privacy parameter ε,
parameters $B \in \mathbb{N}$, $\lambda \in \mathbb{R}_{>0} \backslash \{3/2\}$
Let $\mathcal{I}$ be the dyadic interval set on $[1, \infty)$ and $\mathcal{I}_t = \{I \in \mathcal{I} : t - B \in I\}$
$\forall \ell$, $I \in \mathcal{I}^\ell$, lazily draw i.i.d. $Z_I \sim \mathrm{Lap}\left(\frac{(1+\ell)^{1-\lambda}}{\varepsilon}\right)$
for $t = 1$ **to** B **do**
    At time t, output 0
end for
for $t = B + 1$ **to** ∞ **do**
    At time t, output $s_{t-B} = \sum_{i=1}^{t-B} x_i + \sum_{I \in \mathcal{I}_t} Z_I$
end for

Budgeting across levels. The privacy budget is split unevenly across levels of the dyadic interval set in order to control the asymptotic growth of the expiration function. Specifically, the budget at level ℓ is chosen to be proportional to $(\ell + 1)^{\lambda-1}$. The case $\lambda = 1$ corresponds to the even distribution used in the construction of Section 3.2.

Our algorithm is shown as Algorithm 3. In the following, we refer to the output of the algorithm run on input x and with values of the random variables z as $\mathcal{A}(x; z)$.

Privacy. We now show that $\mathcal{A}$ satisfies Definition 1.1. For x and x' that differ (only) at time $j \leq \tau$, the prefix sums fulfill the following properties:

- $\sum_{i=1}^{t} x_i = \sum_{i=1}^{t} x'_i$ for all $t < j$ and
- $\sum_{i=1}^{t} x'_i = \sum_{i=1}^{t} x_i + y$ for all $j \leq t \leq \tau$, where $y = x'_j - x_j \in [-1, 1]$.

Let j and τ be defined as in Definition 1.1. Due to the delay, if $\tau' = \tau - B < j$ (corresponding to $d < B$) the privacy claim is immediate since the output distributions of $\mathcal{A}$ up to step τ are identical on the two inputs. Otherwise, for $\tau' = \tau - B \geq j$, i.e., for $d \geq B$, we wish to use Fact 2.1 and Lemma 2.2. Let $Z = (Z_I)_{I \in \mathcal{I}}$ be the sequence of Laplace random variables used by $\mathcal{A}$. For input x consider a fixed length-τ output sequence consisting of B zeros followed by $s_1, \ldots, s_{\tau'}$. Let $z = (z_I)_{I \in \mathcal{I}}$ be a sequence of values for the Laplace random variables in order to produce this output sequence with input x. That is, $s_t = \sum_{i=1}^{t} x_i + \sum_{I \in \mathcal{I}_t} z_I$ for $t \geq 1$. Let $D_{[j,\tau']}$ be the decomposition of $[j, \tau']$ as defined in Fact 3.2. We define a bijection q satisfying the properties of Fact 2.1 as follows: $q(z) = z' = (z'_I)_{I \in \mathcal{I}}$ such that

$$z'_I = \begin{cases} z'_I = z_I & \forall I \notin D_{[j,\tau']} \\ z'_I = z_I + y & \forall I \in D_{[j,\tau']} \end{cases}.$$

We show the two properties needed to apply Fact 2.1:

Lemma 4.1. *The function $q(z)$ satisfies $\mathcal{A}(x; z) = \mathcal{A}(x'; q(z))$ and for $g(d) = O(1 + \frac{1}{\lambda} \log^{\lambda}(d - B + 1))$ we have $\Pr_{z \sim Z}[z \in \mathcal{N}] \leq e^{\varepsilon g(\tau - j)} \Pr[z' \in \mathcal{N}]$ for all $\mathcal{N} \subseteq \mathrm{support}(Z)$.*

Space and time. Algorithm 3 can update the sums $\sum_{i=1}^{t-B} x_i$ and $\sum_{I \in \mathcal{I}_t} Z_I$ in each time step t using the following idea: The B most recent inputs are kept in a buffer to allow calculation of prefix sums with delay and also the $\lfloor \log(t) \rfloor + 1$ random variables of those values that were added to the most recent output. At a given step, each random value that is no longer used is subtracted from the most recent output, and each new random value is added. An amortization argument as in the analysis of the number of bit flips in a binary counter yields the $O(1)$ amortized bound.

Accuracy. To show the accuracy guarantee, we need to account for the error due to delay as well as the noise required for privacy. It is easy to see that the delay causes an error of at most B, since the sum of any B inputs is bounded by B. Thus, for both error bounds it remains to account for the error due to noise. At every time step t after B, the output is the delayed prefix sum plus a sum of Laplace distributed noise terms as indicated by $\mathcal{I}_{t-B}$ with parameters $b_\ell = (\ell + 1)^{1-\lambda}/\varepsilon$, where

$\ell = 0, \ldots, \lfloor \log(t-B) \rfloor$. To bound the variance of the noise, $2\sum_\ell b_\ell^2$ we compute:

$$\sum_{\ell=0}^{\lfloor \log(t-B) \rfloor} b_\ell^2 = \frac{1}{\varepsilon^2} \sum_{\ell=0}^{\lfloor \log(t-B) \rfloor} (\ell+1)^{2(1-\lambda)} \leq \frac{1}{\varepsilon^2} \left(1 + \int_1^{\log(t)+2} x^{2(1-\lambda)} \mathrm{d}x \right)$$
$$\leq \frac{1}{\varepsilon^2} \left(1 + \left[\frac{1}{3-2\lambda} x^{3-2\lambda} \right]_{x=1}^{x=\log(t)+2} \right) = O\left(\frac{1+\log(t)^{3-2\lambda}}{\varepsilon^2} \right) .$$

This calculation assumes $\lambda \neq 3/2$, proving the statement on the squared error in Theorem 1.2. For the high probability bound we invoke Lemma B.3 with $b_M = \max_\ell(b_\ell) = \max(1, \log(2t)^{1-\lambda})/\varepsilon) = O((1+\log(t)^{1-\lambda})/\varepsilon)$. For $\nu = \sqrt{\sum_\ell b_\ell^2} + b_M\sqrt{\log t} = O(b_M\sqrt{\log t})$, applying Lemma B.3 says that the error from the noise is $O(\nu\sqrt{\log(1/\beta)}) = O(\sqrt{\log(1/\beta)}\log(t)^{\max(0.5,1.5-\lambda)})$ with probability $1-\beta$, proving Theorem 1.2.

Proof of Corollary 1.3. For releasing T outputs, choosing $\beta = 1/T^{c+1}$, $c > 0$ being a constant, and using a union bound over all outputs gives a bound on the maximum noise equal to $O(\log(T)^{\max(1,2-\lambda)}/\varepsilon)$ with probability $1 - 1/T^c$, proving Corollary 1.3.

5 Lower Bound on the Privacy Decay

The lower bound follows from a careful packing argument. The proof is deferred to Appendix C.4.

Theorem 5.1. *Let $\mathcal{A}$ be an algorithm for binary counting for streams of length T which satisfies ε-differential privacy with expiration g. Let C be an integer and assume that the additive error of $\mathcal{A}$ is bounded by $C < T/2$ at all time steps with a probability of at least 2/3. Then*

$$\sum_{j=0}^{2C-1} g(j) \geq \log(T/6C)/\varepsilon$$

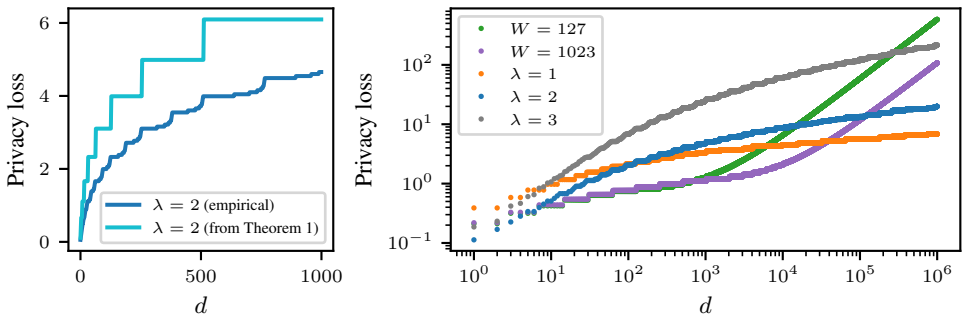
The lower bound extends to mechanisms running on unbounded streams. By Definition 1.1, $g(j)$ is non-decreasing in j. This immediately gives Theorem 1.4.

6 Empirical Evaluation

We empirically evaluated (i) how the privacy loss increases as the elapsed time increases for Algorithm 3, (ii) how tightly the corresponding theoretical expiration function g of Theorem 1.2 bounds this privacy loss, and (iii) how this privacy loss compares to the privacy loss of a realistic baseline. As different algorithms have different parameters that can affect privacy loss, we use the following approach to perform a fair comparison: In the design of ε-differentially private algorithms the error of different algorithms is frequently measured with the same value of the privacy loss parameter ε. Here, we turn this approach around: *We compare the privacy loss (as a function of elapsed time) of different algorithms whose privacy parameter ε is chosen to achieve the same error.*

We empirically compute the privacy loss for Algorithm 3 by considering the exact dyadic decompositions used for the privacy argument (see Appendix A for details). As a baseline to compare against, we break the input stream into intervals of length W, run the 'standard' binary mechanism $\mathcal{A}_B$ of Chan et al. [8] with a privacy parameter ε_{cur} on the current interval, and compute the sum of all prior intervals with a different privacy guarantee ε_{past}. As for both algorithms that we evaluate it is straightforward to compute the *mean-squared error* (MSE) for all outputs on a stream of length T, while the corresponding maximum absolute error can only be observed empirically, we fix the MSE for $T = d_{max} + 1$, where d_{max} is the greatest d (on the x-axis) shown in each plot. We normalize each plot to achieve the same MSE over the first T outputs, across all algorithms and parameter choices. For all runs of Algorithm 3 we used $B = 0$, as for larger values of B, the primary effect would be to shift the privacy loss curve to the right. We picked the MSE to be 1000 for all plots as it leads to small values of the empirical privacy loss. As for both algorithms, the privacy loss *does not* depend on the input data; we used an all-zero input stream, a standard approach in the industry (see, for example, Thakurta's [39] plenary talk at USENIX, 2017).

https://doi.org/10.52202/079017-0299



(a) Alg. 3 vs. g in Theorem 1.2.

(b) Alg. 3 and baseline for various λ, resp., W.

Figure 2: Plots on the privacy loss for our Algorithm 3 and a baseline algorithm.

Figure 2(a) shows that g from Theorem 1.2 is a good approximation of the empirical privacy loss, and that both exhibit the same polylogarithmic growth. Figure 2(b) shows that for large enough d our algorithm has lower privacy loss than the baseline algorithm. See more details in the appendix.

7 Conclusion

In this work, we give the first algorithm for the continual counting problem for privacy with expiration for a wide range of expiration functions and characterize for which expiration functions it is possible to get an ℓ_∞-error of $O(\log(T)/\varepsilon)$. We also give a general lower bound for any such algorithms and show that ours is tight for certain expiration functions. It would be interesting to study this model further, e.g., with slower-growing expiration functions, and also algorithms for other problems in continual observation, such as maintaining histograms and frequency-based statics over changing data. Specifically, it would be an interesting direction for future research to study problems in this model, where a polynomial error gap between the batch model and the continual release model is known to exist (for example max sum [29] and counting distinct elements [28, 23]), and to see if this model allows for new trade-offs. Further, one of the main applications of continual counting algorithms is in privacy-preserving federated learning algorithms, specifically in stochastic gradient descent (see e.g. [14, 30, 34]). It would be interesting to explore how our algorithm can be deployed in this setting.

Though the concept of privacy expiration has not been defined for approximate differential privacy, it is natural to wonder if there exist analogous results in this setting, which, in general, allows better privacy-utility trade-offs. We note that for ρ-zero-concentrated differential privacy [5] there is a natural analog of Definition 1.1 for which it seems possible to prove results analogous to those shown here for pure differential privacy. In that context it would also be interesting to see whether the matrix mechanism can be used to improve the constants in the error, similar to [21, 24].

8 Acknowledgements

Monika Henzinger: This project has received funding from the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation programme (Grant agreement No. 101019564) and the Austrian Science Fund (FWF) grant DOI 10.55776/Z422, grant DOI 10.55776/I5982, and grant DOI 10.55776/P33775 with additional funding from the netidee SCIENCE Stiftung, 2020–2024.

Joel Daniel Andersson and Rasmus Pagh are affiliated with Basic Algorithms Research Copenhagen (BARC), supported by the VILLUM Foundation grant 16582, and are also supported by Providentia, a Data Science Distinguished Investigator grant from Novo Nordisk Fonden.

Teresa Anna Steiner is supported by a research grant (VIL51463) from VILLUM FONDEN. This work was done while Teresa Anna Steiner was a Postdoc at the Technical University of Denmark.

Jalaj Upadhyay's research was funded by the Rutgers Decanal Grant no. 302918 and an unrestricted gift from Google.

References

[1] Kareem Amin, Alex Kulesza, and Sergei Vassilvitskii. Practical considerations for differential privacy. *arXiv preprint arXiv:2408.07614*, 2024.

[2] Joel Daniel Andersson and Rasmus Pagh. A smooth binary mechanism for efficient private continual observation. In *Neural Information Processing Letters*, 2023.

[3] Joel Daniel Andersson, Rasmus Pagh, and Sahel Torkamani. Count on your elders: Laplace vs Gaussian noise. *arXiv preprint arXiv:2408.07021*, 2024.

[4] Jean Bolot, Nadia Fawaz, Shanmugavelayutham Muthukrishnan, Aleksandar Nikolov, and Nina Taft. Private decayed predicate sums on streams. In *Proceedings of the 16th International Conference on Database Theory*, pages 284–295, 2013.

[5] Mark Bun and Thomas Steinke. Concentrated differential privacy: Simplifications, extensions, and lower bounds. In Martin Hirt and Adam D. Smith, editors, *Theory of Cryptography - 14th International Conference, TCC 2016-B, Beijing, China, October 31 - November 3, 2016, Proceedings, Part I*, volume 9985 of *Lecture Notes in Computer Science*, pages 635–658, 2016.

[6] Adrian Rivera Cardoso and Ryan Rogers. Differentially private histograms under continual observation: Streaming selection into the unknown. In *International Conference on Artificial Intelligence and Statistics*, pages 2397–2419. PMLR, 2022.

[7] T-H Hubert Chan, Mingfei Li, Elaine Shi, and Wenchang Xu. Differentially private continual monitoring of heavy hitters from distributed streams. In *International Symposium on Privacy Enhancing Technologies Symposium*, pages 140–159. Springer, 2012.

[8] T.-H. Hubert Chan, Elaine Shi, and Dawn Song. Private and continual release of statistics. *ACM Trans. Inf. Syst. Secur.*, 14(3):26:1–26:24, 2011.

[9] Yan Chen, Ashwin Machanavajjhala, Michael Hay, and Gerome Miklau. Pegasus: Data-adaptive differentially private stream processing. In *Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security*, pages 1375–1388, 2017.

[10] Christopher A Choquette-Choo, Krishnamurthy Dvijotham, Krishna Pillutla, Arun Ganesh, Thomas Steinke, and Abhradeep Thakurta. Correlated noise provably beats independent noise for differentially private learning. In *International Conference on Learning Representation*, 2023.

[11] Christopher A Choquette-Choo, Arun Ganesh, Saminul Haque, Thomas Steinke, and Abhradeep Thakurta. Near exact privacy amplification for matrix mechanisms. *arXiv preprint arXiv:2410.06266*, 2024.

[12] Christopher A Choquette-Choo, Arun Ganesh, Ryan McKenna, H Brendan McMahan, John Rush, Abhradeep Guha Thakurta, and Zheng Xu. (amplified) banded matrix factorization: A unified approach to private training. *Advances in Neural Information Processing Systems*, 36, 2024.

[13] Christopher A. Choquette-Choo, H. Brendan McMahan, Keith Rush, and Abhradeep Thakurta. Multi-epoch matrix factorization mechanisms for private machine learning. In *Proceedings of the 40th International Conference on Machine Learning*, ICML'23, 2023.

[14] Sergey Denisov, H Brendan McMahan, John Rush, Adam Smith, and Abhradeep Guha Thakurta. Improved differential privacy for sgd via optimal private linear operators on adaptive streams. *Advances in Neural Information Processing Systems*, 35:5910–5924, 2022.

[15] Krishnamurthy (Dj) Dvijotham, H. Brendan McMahan, Krishna Pillutla, Thomas Steinke, and Abhradeep Thakurta. Efficient and near-optimal noise generation for streaming differential privacy. In *Foundations of Computer Science*, 2024.

[16] Cynthia Dwork, Moni Naor, Toniann Pitassi, and Guy N. Rothblum. Differential privacy under continual observation. In *Proceedings of the 42nd ACM Symposium on Theory of Computing*, pages 715–724, 2010.

 https://doi.org/10.52202/079017-0299

[17] Cynthia Dwork, Moni Naor, Toniann Pitassi, Guy N Rothblum, and Sergey Yekhanin. Pan-private streaming algorithms. In *ics*, pages 66–80, 2010.

[18] Cynthia Dwork, Moni Naor, Omer Reingold, and Guy N Rothblum. Pure differential privacy for rectangle queries via private partitions. In *International Conference on the Theory and Application of Cryptology and Information Security*, pages 735–751. Springer, 2015.

[19] Alessandro Epasto, Jieming Mao, Andres Munoz Medina, Vahab Mirrokni, Sergei Vassilvitskii, and Peilin Zhong. Differentially Private Continual Releases of Streaming Frequency Moment Estimations. In *14th Innovations in Theoretical Computer Science Conference (ITCS 2023)*, volume 251, pages 48:1–48:24, 2023.

[20] Hendrik Fichtenberger, Monika Henzinger, and Lara Ost. Differentially private algorithms for graphs under continual observation. In *29th Annual European Symposium on Algorithms, ESA 2021, September 6-8, 2021, Lisbon, Portugal (Virtual Conference)*, 2021.

[21] Hendrik Fichtenberger, Monika Henzinger, and Jalaj Upadhyay. Constant matters: Fine-grained error bound on differentially private continual observation. In *International Conference on Machine Learning, ICML*, volume 202 of *Proceedings of Machine Learning Research*, pages 10072–10092. PMLR, 2023.

[22] Monika Henzinger, A. R. Sricharan, and Teresa Anna Steiner. Differentially private data structures under continual observation for histograms and related queries. *CoRR*, abs/2302.11341, 2023.

[23] Monika Henzinger, A. R. Sricharan, and Teresa Anna Steiner. Private counting of distinct elements in the turnstile model and extensions. In *Proc. APPROX/RANDOM 2024*, pages 40:1–40:21, 2024.

[24] Monika Henzinger and Jalaj Upadhyay. Improved differentially private continual observation using group algebra. In *Proceedings of the 2025 Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*. SIAM, 2025.

[25] Monika Henzinger, Jalaj Upadhyay, and Sarvagya Upadhyay. Almost tight error bounds on differentially private continual counting. In *Proceedings of the 2023 Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 5003–5039. SIAM, 2023.

[26] Monika Henzinger, Jalaj Upadhyay, and Sarvagya Upadhyay. A unifying framework for differentially private sums under continual observation. In *Proceedings of the 2024 Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 995–1018. SIAM, 2024.

[27] James Honaker. Efficient use of differentially private binary trees. *Theory and Practice of Differential Privacy (TPDP 2015), London, UK*, 2:26–27, 2015.

[28] Palak Jain, Iden Kalemaj, Sofya Raskhodnikova, Satchit Sivakumar, and Adam D. Smith. Counting distinct elements in the turnstile model with differential privacy under continual observation. In *Proc. 37th NeurIPS*, 2023.

[29] Palak Jain, Sofya Raskhodnikova, Satchit Sivakumar, and Adam Smith. The price of differential privacy under continual observation. In *Proceedings of the 40th International Conference on Machine Learning*, volume 202 of *Proceedings of Machine Learning Research*, pages 14654–14678. PMLR, 23–29 Jul 2023.

[30] Peter Kairouz, Brendan McMahan, Shuang Song, Om Thakkar, Abhradeep Thakurta, and Zheng Xu. Practical and private (deep) learning without sampling or shuffling. In *International Conference on Machine Learning*, pages 5213–5225. PMLR, 2021.

[31] Anastasiia Koloskova, Ryan McKenna, Zachary Charles, John Rush, and H Brendan McMahan. Gradient descent with linearly correlated noise: Theory and applications to differential privacy. *Advances in Neural Information Processing Systems*, 36, 2023.

[32] Max Dupré la Tour, Monika Henzinger, and David Saulpic. Making old things new: A unified algorithm for differentially private clustering. In *Forty-first International Conference on Machine Learning, ICML 2024, Vienna, Austria, July 21-27, 2024*. OpenReview.net, 2024.

[33] Chao Li, Michael Hay, Vibhor Rastogi, Gerome Miklau, and Andrew McGregor. Optimizing linear counting queries under differential privacy. In *Proceedings of the twenty-ninth ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems*, pages 123–134. ACM, 2010.

[34] Brendan McMahan and Abhradeep Thakurta. Federated learning with formal differential privacy guarantees. *Google AI Blog*, 2022.

[35] Victor Perrier, Hassan Jameel Asghar, and Dali Kaafar. Private continual release of real-valued data streams. In *Network and Distributed Systems Security (NDSS) Symposium*, 2019.

[36] Vibhor Rastogi and Suman Nath. Differentially private aggregation of distributed time-series with transformation and encryption. In *Proceedings of SIGMOD International Conference on Management of data*, pages 735–746, 2010.

[37] Adam Smith, Abhradeep Thakurta, and Jalaj Upadhyay. Is interaction necessary for distributed private learning? In *IEEE Symposium on Security and Privacy*, pages 58–77. IEEE, 2017.

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

[39] Abhradeep Thakurta. Differential privacy: From theory to deployment, `https://www.youtube.com/watch?v=Nvy-TspgZMs&t=2320s&ab_channel=USENIX`, 2017.

[40] Jalaj Upadhyay. Sublinear space private algorithms under the sliding window model. In *International Conference on Machine Learning*, pages 6363–6372, 2019.

[41] Jalaj Upadhyay, Sarvagya Upadhyay, and Raman Arora. Differentially private analysis on graph streams. In *International Conference on Artificial Intelligence and Statistics*, pages 1171–1179. PMLR, 2021.

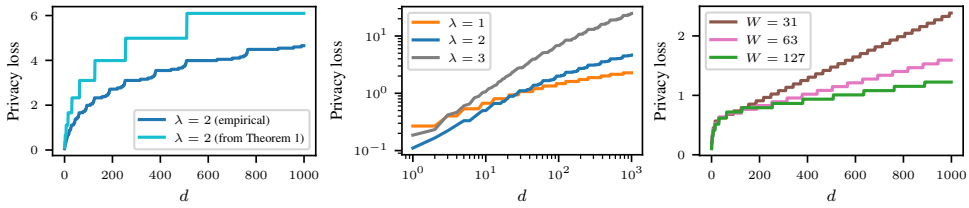
 https://doi.org/10.52202/079017-0299

A Empirical Evaluation

In this section, we empirically review (i) how the privacy loss increases as the elapsed time increases for Algorithm 3, (ii) how tightly the corresponding theoretical expiration function g of Theorem 1.2 bounds this privacy loss, and (iii) how this privacy loss compares to the privacy loss of a realistic baseline. Note that different algorithms have different parameters that can affect privacy loss; thus, it is not clear how to perform a fair comparison at first. We use the following approach: In the design of ε-differentially private algorithms the error is frequently bound as a function of the privacy loss, i.e., the error of different algorithms is measured with the same value of the privacy loss parameter ε. Here, we turn this approach around: *We compare the privacy loss (as a function of elapsed time) of different algorithms whose privacy parameter ε is chosen to achieve the same error.* As for both algorithms that we evaluate it is straightforward to compute the *mean-squared error* (MSE) for all outputs on a stream of length T, while the corresponding maximum absolute error can only be observed empirically, we fix the MSE for $T = d_{max} + 1$, where d_{max} is the greatest d shown in each plot. We normalize each plot to achieve the same MSE over the first T outputs, across all algorithms and parameter choices. For all runs of Algorithm 3 we used $B = 0$, as for larger but practical values of B, the primary effect would be to shift the privacy loss curve to the right. We picked the MSE to be 1000 for all plots as it leads to small values of the empirical privacy loss. Any other choice would simply rescale the values of the y-axes.

Note that for both algorithms the privacy loss *does not* depend on the input data and, thus, we used an all-zero input stream, a standard approach in the industry (see, for example, [39]'s plenary talk at USENIX, 2017). We also emphasize that all computations producing the plots shown are deterministic, and so there is no need for error bars.

Empirical privacy expiration for Algorithm 3. For our evaluation we empirically compute the privacy loss for Algorithm 3 by considering the exact dyadic decompositions used for the privacy argument. More specifically, to compute the empirical privacy expiration function g (for $d \geq B$) of Algorithm 3, we use the proof of Theorem 1.2 to reason that for $\mathcal{N} \in \text{support}(Z)$, $\Pr_{z \in Z}[z \in \mathcal{N}]$ is bounded by eq. (1). There we bound the exponent, i.e. the privacy loss, as a function of the size of the interval $d - B + 1 = \tau' - j + 1$, and this yields the g stated. However, this might not be exact, as the decomposition of an interval of size $d - B + 1$ does not necessarily involve using 2 intervals per level up to $\ell^* = \lfloor \log(d - B + 1) \rfloor$. Instead, for a given $d \geq B$ and λ, we can for all $\tau' \geq j$, where $\tau' - j + 1 = d - B + 1$, compute the exact associated dyadic decomposition $D_{[j,\tau']}$ and the resulting privacy loss. For each value of d taking the maximum over these losses over all $\tau' \leq t$ gives the worst-case privacy loss at time $t + d$ for an input streamed at time t.



(a) Alg. 3 vs. g in Theorem 1.2. (b) Algorithm 3 for multiple λ. (c) Baseline for multiple W.

Figure 3: Worst-case privacy loss computed empirically for a data item streamed d steps earlier.

Figure 3(a) demonstrates how the privacy loss compares to what is predicted by Theorem 1.2. The main observation is that g from Theorem 1.2 is a good approximation of the empirical privacy loss, and that both exhibit the same polylogarithmic growth.

Figure 3(b) plots the empirically computed privacy loss for different choices of λ. It demonstrates that the choice of λ equates to a trade-off between short-term and long-term privacy. The larger λ the higher the privacy loss for large values of d, which is to be expected. It also shows that a smaller λ provides greater privacy loss early on.

A baseline. As a baseline to compare against, we break the input stream into intervals of length W, run the 'standard' binary mechanism $\mathcal{A}_B$ of [8] with a privacy parameter ε_{cur} on the current interval, and compute the sum of all prior intervals with a different privacy guarantee ε_{past}. More exactly, the baseline outputs $\mathcal{A}_B(\varepsilon_{cur}, t)$ for all time steps t in the first round, i.e., $t \in [1, W]$. For

https://doi.org/10.52202/079017-0299

each later round $r > 1$, it does the following: (1) Compute $c_r = z_{past} + \sum_{t=1}^{(r-1)\cdot W} x_t$ where $z_{past} \sim \text{Lap}(1/\varepsilon_{past})$. (2) For $t \in [(r-1)\cdot W + 1, r \cdot W]$, output $c_r + \mathcal{A}_B(\varepsilon_{cur}, t)$

Intuitively, this models what is currently done in the software industry: Each user is given a "privacy budget" for a fixed time interval with the guarantee that their data is no longer used whenever the budget reaches 0 *within the current interval*. However, at the beginning of the next interval, the privacy budget is being reset. Thus the total privacy loss is the sum of the privacy losses in all intervals.

What fraction of the privacy budget is spent on the binary mechanism (ε_{cur}) and what is spent on releasing the prefix of the past (ε_{past}) can be chosen in multiple ways. For our experiments, we choose a fixed fraction of $\varepsilon_{past}/\varepsilon_{cur} = 0.1$, implying that we release sums from past rounds with a stronger privacy guarantee compared to what is used in the binary tree. Other choices and their implication are discussed in Appendix A.1.

Figure 3(c) shows the privacy loss of the baseline for a selection of round sizes. The main feature to underline is that, after W time steps, the privacy expiration becomes linear in the number of rounds (and therefore in d). This is a direct consequence of privacy composition: An input x_t impacts the outputs of $\mathcal{A}_B$ in the round $r = \lceil t/W \rceil$ it participates in, and then is subsequently released in each future round $r' > r$ as part of $c_{r'}$.

Comparing Algorithm 3 to the baseline. Algorithm 3 is compared to the baseline in Figure 4. Qualitatively the results align with our expectations. For large enough $d > W$, the baseline enters the region where the degradation in privacy is dominated by the number of rounds that a given input participates in, yielding linear expiration. Notably, the baseline generally achieves a lower privacy loss for smaller d compared to our method. This is largely decided by how $\varepsilon_{past}/\varepsilon_{cur}$ is picked. Choosing a smaller fraction would lower the slope in the linear regime at the expense of the early privacy loss incurred from $\mathcal{A}_B$. By contrast, Algorithm 3 exhibits a comparable initial privacy loss that grows more slowly – never reaching linear privacy loss – as predicted by our theorem.

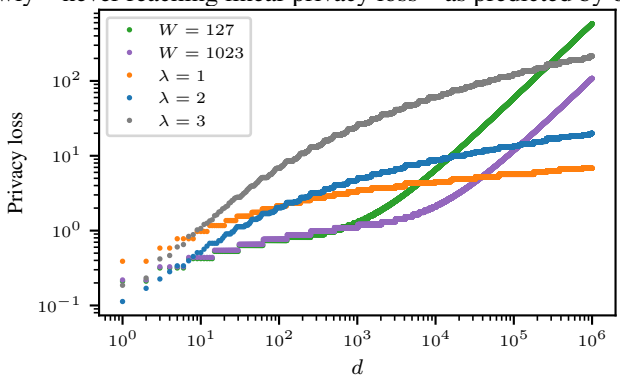


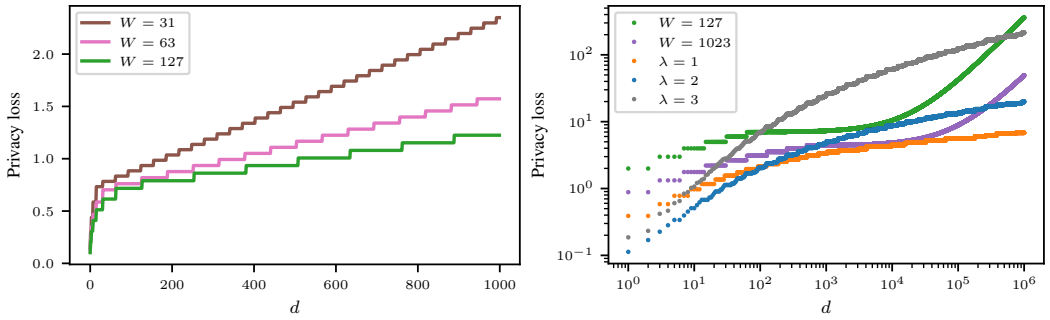
Figure 4: Worst-case privacy loss for a data item streamed d steps earlier, shown for Algorithm 3 (with $\lambda = 1, 2, 3$) versus the baseline ($W = 127$ and $W = 1023$).

A.1 Technical Details

Choosing $\varepsilon_{past}/\varepsilon_{cur}$ for the baseline For our experiments above, we choose a fraction $\varepsilon_{past}/\varepsilon_{cur} = 0.1$ for our baseline, but, as we state, other choices are possible. In particular, one appealing choice is the fraction that minimizes the privacy loss at $d = T - 1$, making the baseline as privacy preserving as possible for the last d shown in Figures 3(c) and 4. Note that the privacy loss for the largest value of d will be, roughly, $\varepsilon_{cur} + \varepsilon_{past} \cdot (N - 1)$, where N is the total number of rounds, corresponding to the privacy loss for an input participating in the first round. We can compute the fraction $\varepsilon_{past}/\varepsilon_{cur}$ that minimizes this privacy loss under the constraint of having a fix mean-squared error, yielding a solution that is a function of the round length W and input stream length T.

The Figures 3(c) and 4 from Section 6 are shown below as Figures 5(a) and 5(b) where we instead of using $\varepsilon_{past}/\varepsilon_{cur} = 0.1$, we use the fraction minimizing the maximum privacy loss. Note that Figure 5(a) is almost identical to Figure 3(c). This is due to the fact that for this choice of W and T, the optimal fraction is close 0.1. In the case of Figure 5(b) however, there is a notable difference. The minimizing fraction is here considerably smaller, resulting in a tangible reduction of the final privacy

https://doi.org/10.52202/079017-0299



(a) Baseline for multiple W.

(b) Algorithm 3. vs. baseline

Figure 5: Worst-case privacy loss computed empirically for a data item streamed d steps earlier. Figure 5(a) is a re-computation of Figure 3(c) where the ratio $\varepsilon_{past}/\varepsilon_{cur}$ is set to minimize the maximum privacy loss, yielding a ratio of 0.069 for $W = 31$, 0.08 for $W = 63$ and 0.095 for $W = 127$. Figure 5(b) is a re-computation of Figure 4 where the ratio $\varepsilon_{past}/\varepsilon_{cur}$ is set to minimize the maximum privacy loss, yielding a ratio of 0.0064 for $W = 127$ and 0.010 for $W = 1023$.

loss, but at the expense of the privacy loss early on. This is in line with expectations: spending more of the privacy budget on releasing sums from past rounds implies spending less on releasing the binary tree in each round, which dominates the privacy loss for small d.

While the fraction $\varepsilon_{past}/\varepsilon_{cur}$ has an impact, running the baseline for a fix round length W will always result in linear privacy expiration for a great enough stream length T, where ε_{past} affects the slope.

A.2 Exact Parameters for the Experiments

In all the plots, we choose the privacy parameter(s) to achieve a mean-squared error of 1000 over the stream length T in each figure shown, where $T = 10^3$ for Figure 3(a), 3(b) and 3(c), and $T = 10^6$ in Figure 4. The corresponding privacy parameters are listed in Table 1, ε refers to the privacy parameter used by Algorithm 3 and ε_{cur}, ε_{past} are the ones used by the baseline.

Table 1: Table over the privacy parameters used in each of the plots.

Figure	Algorithm	ε	ε_{cur}	ε_{past}
3(a)	$\lambda = 2$	0.05542	-	-
3(b)	$\lambda = 1$	0.1341	-	-
	$\lambda = 2$	0.05542	-	-
	$\lambda = 3$	0.04651	-	-
3(c)	$W = 31$	-	0.5678	0.05678
	$W = 63$	-	0.6372	0.06372
	$W = 127$	-	0.7197	0.07197
4	$W = 127$	-	0.7387	0.07387
	$W = 1023$	-	1.096	0.1096
	$\lambda = 1$	0.1947	-	-
	$\lambda = 2$	0.05645	-	-
	$\lambda = 3$	0.04652	-	-
5(a)	$W = 31$	-	0.7328	0.05048
	$W = 63$	-	0.7031	0.05796
	$W = 127$	-	0.7170	0.07252
5(b)	$W = 127$	-	6.973	0.04488
	$W = 1023$	-	4.413	0.04589
	$\lambda = 1$	0.1947	-	-
	$\lambda = 2$	0.05645	-	-
	$\lambda = 3$	0.04652	-	-

https://doi.org/10.52202/079017-0299

B Background on Random Variables

Definition B.1 (Laplace Distribution)**.** *The* Laplace distribution *centered at* 0 *with scale* b *is the distribution with probability density function*

$$f_{\mathrm{Lap}}(b)(x) = \frac{1}{2b}\exp\left(\frac{-|x|}{b}\right).$$

We use $X \sim \mathrm{Lap}(b)$ *or just* $\mathrm{Lap}(b)$ *to denote a random variable* X *distributed according to* $f_{\mathrm{Lap}}(b)(x)$*.*

Fact B.2 (Laplace Tailbound)**.** *If* $X \sim \mathrm{Lap}(b)$*, then*

$$\Pr[|X| > t \cdot b] \le e^{-t}.$$

Lemma B.3 (Measure Concentration Lemma in [8])**.** *Let* $Y_1, \ldots, Y_k$ *be independent variables with* $Y_i \sim \mathrm{Lap}(b_i)$ *for all* $i \in [k]$*. Denote* $b_M = \max_{i\in[k]} b_i$ *and* $Y = \sum_{i=1}^k Y_i$*. Let* $0 < \beta < 1$ *and* $\nu > \max\left\{\sqrt{\sum_{i=1}^k b_i^2}, b_M\sqrt{\ln(2/\beta)}\right\}$*. Then,*

$$\Pr\left[|Y| > \nu\sqrt{8\ln(2/\beta)}\right] \le \beta.$$

Corollary B.4. *Let* $Y_1, \ldots, Y_k$ *be independent variables with distribution* $\mathrm{Lap}(b)$ *and let* $Y = \sum_{i=1}^k Y_i$*. Let* $0 < \beta < 1$*. Then*

$$\Pr\left[|Y| > 2b\sqrt{2\ln(2/\beta)}\max\left\{\sqrt{k}, \sqrt{\ln(2/\beta)}\right\}\right] \le \beta.$$

Proof. Apply Lemma B.3 to $b_1 = \cdots = b_k = b$. □

C Omitted proofs

C.1 Omitted proofs from Section 2

Proof of Fact 2.1. Given x, x' and any set of output sequences set S, let $\mathcal{N}_S$ be the set of all choices of random variables z such that $\mathcal{A}(x; z) \in S$. Note that for all $z \in \mathcal{N}_S$, $\mathcal{A}(x'; q(z)) \in S$, i.e., for all $z' \in q(\mathcal{N}_S)$, $\mathcal{A}(x'; z') \in S$. Then

$$\begin{aligned}\Pr_{z\sim Z}[\mathcal{A}(x; z) \in S] &= \Pr_{z\sim Z}[z \in \mathcal{N}_S] \le e^{\varepsilon g(\tau - j)} \Pr_{z'\sim Z}[z' \in q(\mathcal{N}_S)] \\ &\le e^{\varepsilon g(\tau-j)} \Pr_{z'\sim Z}[\mathcal{A}(x'; z') \in S].\end{aligned}$$

This completes the proof of Theorem 2.1. □

Proof of Lemma 2.2. Let $f_{\mathrm{Lap}}(b)$ denote the density function of $\mathrm{Lap}(b)$. For all $x, y \in \mathbb{R}$:

$$f_{\mathrm{Lap}}(b)(x) = \frac{1}{2b}e^{-|x|/b} \le \frac{1}{2b}e^{(|y|-|x+y|)/b} = e^{|y|/b}\frac{1}{2b}e^{(-|x+y|)/b} = e^{|y|/b}f_{\mathrm{Lap}}(b)(x+y),$$

where the inequality follows from the triangle inequality $|x+y| \le |x| + |y|$. In the following integral, let $z' = q(z)$. Let s be as defined in the lemma statement. Then we have

$$\begin{aligned}\Pr_{z\in Z}[z \in \mathcal{N}] &= \int_{z\in\mathcal{N}} \prod_{i=1}^k f_{\mathrm{Lap}}(b_i)(z_i)dz \le \int_{z\in\mathcal{N}} \prod_{i=1}^k e^{|z_i'-z_i|/b_i} f_{\mathrm{Lap}}(b_i)(z_i')dz \\ &= e^s \int_{z\in\mathcal{N}} \prod_{i=1}^k f_{\mathrm{Lap}}(b_i)(z_i')dz = e^s \int_{z\in q(\mathcal{N})} \prod_{i=1}^k f_{\mathrm{Lap}}(b_i)(z_i)dz \\ &= e^s \Pr_{z\sim Z}[z \in q(\mathcal{N})]\end{aligned}$$

This completes the proof of Theorem 2.2. □

https://doi.org/10.52202/079017-0299

C.2 Omitted proofs from Section 3

Proof of Lemma 3.1. **Privacy.** We use Fact 2.1 and Lemma 2.2 to argue the privacy of our simple algorithm, $\mathcal{A}_{\mathsf{simple}}$: Let x and x' differ at time j. Note that the prefix sums fulfill the following properties:

- $\sum_{i=1}^{t} x_i = \sum_{i=1}^{t} x'_i$ for all $t < j$ and
- $\sum_{i=1}^{t} x'_i = \sum_{i=1}^{t} x_i + y$ for all $t \geq j$, where $y = x'_j - x_j \in [-1, 1]$.

Let $S \subseteq \mathrm{range}(\mathcal{A}_{\mathsf{simple}})^*$. For any $\tau \geq j$, we consider the output distribution of length-τ prefixes. There are two cases: For $j = \tau$ and any $t \leq \tau$, the output on x is given by $\sum_{i=1}^{t-1} x_i + Z_{t-1}$ with $t - 1 < j$, thus we get equal output distributions on x and x'. That is,

$$\Pr[\mathcal{A}(x_1, \ldots, x_t)_{t=1}^{\tau} \in S] = \Pr[\mathcal{A}(x'_1, \ldots, x'_t)_{t=1}^{\tau} \in S] = e^{g(0)\varepsilon} \Pr[\mathcal{A}(x'_1, \ldots, x'_t)_{t=1}^{\tau} \in S].$$

For $j < \tau$, let $Z = Z_1, \ldots, Z_{\tau-1}$ be the sequence of Laplace random variables used by the algorithm. For any fixed output sequence $s \in S$, note that the algorithm guarantees that $s_1 = 0$ and let $z = z_1, z_2, \ldots, z_{\tau-1}$ be the values that the Laplace random variables need to assume to get output sequence s with input x. That is, $\sum_{i=1}^{t} x_i + z_t = s_{t+1}$ for $t \geq 1$. We define a bijection q satisfying the properties of Fact 2.1 as follows. Define $q(z) = z'$ such that

$$z'_i = \begin{cases} z_i & i \leq j \\ z_i + y & i > j \end{cases}.$$

This gives $\sum_{i=1}^{t} x'_i + z'_t = s_{t+1}$. All Laplace noises have the same distribution $\mathrm{Lap}(1/\varepsilon)$. By Lemma 2.2, we have for any $\mathcal{N} \in \mathrm{support}(Z)$,

$$\Pr_{z \in Z}[z \in \mathcal{N}] \leq e^{(\tau - j)\varepsilon} \Pr_{z \in Z}[z \in q(\mathcal{N})].$$

Thus, q fulfills the properties of Fact 2.1, and therefore our algorithm satisfies ε-differential privacy with expiration $g(t) = t$.

Accuracy. The error at time step t is given by $x_t + Z_{t-1}$. By the Laplace tail bound (Fact B.2), we have that

$$\Pr_{z_i \sim \mathrm{Lap}(1/\varepsilon)}[|z_i| > \varepsilon^{-1} \log(T/\beta)] \leq \beta/T$$

for any $i \in [T - 1]$.

Therefore, by a union bound, $|Z_i| \leq \log(T/\beta)$ simultaneously for all $i \in [T - 1]$ with probability at least $1 - \beta$. This implies that, with probability at least $1 - \beta$, the maximum error over the entire stream is bounded by $\varepsilon^{-1} \log(T/\beta)$. □

Proof sketch of Fact 3.2. For $a = b$, the claim is immediate and so we prove it for $a < b$. First, consider the case where $a = 2^\ell$ and $b \leq 2 \cdot 2^\ell - 1$. We show that in this case, there exists a set $D_{[a,b]}$ with the properties above, only it contains at most *one* interval per level. If $b = 2 \cdot 2^\ell - 1$, then $D_{[a,b]} = \{[a, b]\}$. Else, we have $b - a + 1 < 2^\ell$, thus there exists a binary representation $(q_0, \ldots, q_{\ell-1}) \in \{0, 1\}^\ell$ such that $b - a + 1 = q_0 + 2q_1 + 2^2 q_2 + \cdots + 2^{\ell-1} q_{\ell-1}$. Note that for any $j \leq \ell$, there is an interval in $\mathcal{I}$ of level j starting at $a = 2^\ell$. We now show how to construct the set $D_{[a,b]}$. In the first step, choose the largest j such that $q_j = 1$ and add the interval $[2^\ell, 2^\ell + 2^j - 1]$ to $D_{[a,b]}$. Then, set $q_j = 0$. Next, assume we already cover the interval $[a, s]$ for some $b \geq s > a$. We then again pick the largest remaining j such that $q_j = 1$ and add the interval $[s, s + 2^j - 1]$ to $D_{[a,b]}$. By an inductive argument, this interval will be in $\mathcal{I}$, since s is the ending position of an interval of a higher level in $\mathcal{I}$. We do this until no more q_j with $q_j = 1$ remains, at which point we have covered $[a, b]$. The same argument can be repeated for the case where $b = 2^\ell - 1$ and $a \geq 1$. Now, for the general case, pick $\ell = \lfloor \log(b) \rfloor$ and $c = 2^\ell - 1$ and define $D_{[a,b]} = D_{[a,c]} \cup D_{[c+1,b]}$. □

Proof sketch of Fact 3.3. First note that in our definition of $\mathcal{I}$, we exclude all intervals starting at 0. For a given t, let $\ell \in \mathbb{N}$ be the greatest interval level satisfying $2^\ell - 1 < t$. It follows that for every $j > \ell$, $t \in [0, 2^j - 1] \notin \mathcal{I}$. Analogously, for every $j \leq \ell$ we have $t \notin [0, 2^j - 1]$, and therefore there must exist a unique $I \in \mathcal{I}^j$ such that $t \in I$. Taken together, we have that $|\mathcal{I}_t| = \ell + 1$ where $\ell = \lfloor \log t \rfloor$. □

C.3 Omitted Proofs from Section 4

In this section, we give the detail proof for privacy analysis for all $\lambda \in \mathbb{R}_{\geq 0}$ and accuracy analysis for $\lambda = 3/2$, which is not covered in Section 4.

C.3.1 Privacy proof for all $\lambda \in \mathbb{R}_{\geq 0}$

Proof of Lemma 4.1. We prove a slightly more general result; i.e., for all $\lambda \in \mathbb{R}_{\geq 0}$, we have the following bound on the privacy expiration function:

$$g(d) \leq \begin{cases} 0 & \text{for } d < B \\ 2\left(1 + \left\lceil \frac{1}{\lambda}\left((\log(d - B + 1) + 1)^\lambda - 1\right)\right\rceil\right) & \text{for } B \leq d \text{ and } \lambda \neq 0 \\ 2\left(1 + \log\log(d - B + 1)\right) & \text{for } B \leq d \text{ and } \lambda = 0 \end{cases}$$

We wish to use Theorem 2.1. For that, we first show that both the requirements in Theorem 2.1 are satisfied.

1. By definition, $\mathcal{A}(x_1 x_2 \dots x_t; z) = \sum_{i=1}^t x_i + \sum_{I \in \mathcal{I}_t} z_I$. For $t < j$, we have $t \notin [j, \tau']$ and therefore $t \notin I$ for all $I \in D_{[j,\tau']}$. It follows that for all $I \in \mathcal{I}_t$ it holds that $I \notin D_{[j,\tau']}$. Therefore, we have

$$\sum_{i=1}^t x_i + \sum_{I \in \mathcal{I}_t} z_I = \sum_{i=1}^t x'_i + \sum_{I \in \mathcal{I}_t} z_I = \sum_{i=1}^t x'_i + \sum_{I \in \mathcal{I}_t} z'_I.$$

   For $t \geq j$, we have that t is contained in exactly one $I \in D_{[j,\tau']}$.Thus, $z'_I = z_I + y$ for exactly one $I \in \mathcal{I}_t$, and $z'_{I'} = z_{I'}$ for all $I' \in \mathcal{I}_t \backslash \{I\}$. Further, since $t \geq j$, we have that $\sum_{i=1}^t x_i = \sum_{i=1}^t x'_i - y$. Together, we have

$$\sum_{i=1}^t x_i + \sum_{I \in \mathcal{I}_k} z_I = \sum_{i=1}^t x'_i - y + \sum_{I \in \mathcal{I}_t} z'_I + y = \sum_{i=1}^t x'_i + \sum_{I \in \mathcal{I}_t} z'_I,$$

   so the first property of Fact 2.1 is fulfilled.

2. By Fact 3.2, $D_{[j,\tau']}$ consists of at most two intervals from each of $\mathcal{I}^0, \dots, \mathcal{I}^{\ell^*}$ where $\ell^* = \lfloor \log(\tau' - j + 1) \rfloor$. Thus, by Lemma 2.2 we have for any $\mathcal{N} \in \text{support}(Z)$, then

$$\begin{aligned} \Pr_{z \in Z}[z \in \mathcal{N}] &\leq \exp\left(\sum_{\ell=0}^{\log T} \sum_{I \in D_{[j,\tau']} \cap \mathcal{I}^\ell} \frac{|y|\varepsilon}{(1+\ell)^{1-\lambda}}\right) \Pr_{z \in Z}[z \in q(\mathcal{N})] \\ &\leq \exp\left(2\varepsilon \sum_{\ell=0}^{\ell^*} (1+\ell)^{\lambda-1}\right) \Pr_{z \in Z}[z \in q(\mathcal{N})] \ . \end{aligned} \tag{1}$$

https://doi.org/10.52202/079017-0299

Using Fact 2.1 with $d = \tau - j = B + \tau' - j$ we conclude differential privacy with privacy expiration $g(d) = 2\sum_{\ell=0}^{\lfloor \log(d-B+1)\rfloor}(1+\ell)^{\lambda-1}$. Next we bound:

$$\begin{aligned} g(d) &= 2 \sum_{\ell=0}^{\lfloor \log(d-B+1)\rfloor} (1+\ell)^{\lambda-1} \\ &\leq 2\left(1 + \int_{1}^{\log(d-B+1)+1} x^{\lambda-1}\,\mathsf{d}x\right) \\ &= 2\left(1 + \left[\frac{1}{\lambda}x^{\lambda}\right]_{x=1}^{x=\log(d-B+1)+1}\right) \\ &= O(1+\log^{\lambda}(d-B+1)) \ . \end{aligned}$$

Note that we use the assumption $\lambda > 0$.

In the case $\lambda = 0$ it instead holds that $g(d) = O(\log\log(d-B+1))$ by setting the value of $\lambda = 0$ in the first inequality and then using the bound on the Harmonic sum.

This is not a point of discontinuity as the $O(\cdot)$ notation would imply. We treated λ as a constant and derived the last equation in the above. If instead, we treat λ as a parameter, we have the following

$$g(d) \leq 2\left(1 + \underbrace{\left[\frac{1}{\lambda}\left((\log(d-B+1)+1)^{\lambda}-1\right)\right]}_{f(\lambda)}\right)$$

We can now evaluate the $f(\lambda)$ as $\lambda \to 0$ as follows: Since $f(\lambda)$ as an indeterminate form and noting that it satisfies the condition required to apply the L'Hôpital rule. Therefore, an application of L'Hôpital rule gives us

$$\begin{aligned} \lim_{\lambda\to 0} f(\lambda) &= \lim_{\lambda\to 0} \frac{\frac{\mathsf{d}}{\mathsf{d}\lambda}((\log(d-B+1)+1)^{\lambda}-1)}{\frac{\mathsf{d}}{\mathsf{d}\lambda}\lambda} \\ &= \lim_{\lambda\to 0} \log(\log(d-B+1))\log^{\lambda}(d-B+1) \\ &= \log(\log(d-B+1)). \end{aligned}$$

That is, we have the desired bound on the privacy expiration. This completes the proof of Theorem 4.1.

□

C.3.2 Accuracy proof when $\lambda = 3/2$

Recall that in the main text, we stated that for $\lambda = 3/2$, we achieve

$$\sum_{\ell=0}^{\lfloor \log(t-B)\rfloor} b_{\ell}^2 \leq \sum_{\ell=0}^{\log t} b_{\ell}^2 = O\left(\frac{\log\log(t)}{\varepsilon^2}\right).$$

This follows from straightforward computation by setting $\lambda = 3/2$:

$$\sum_{\ell=0}^{\log t} b_{\ell}^2 = \sum_{\ell=0}^{\log t} \frac{(\ell+1)^{2(1-\lambda)}}{\varepsilon^2} = \sum_{\ell=0}^{\log t} \frac{1}{\varepsilon^2(\ell+1)} = O\left(\frac{\log\log(t)}{\varepsilon^2}\right).$$

While it seems like a point of discontinuity based on what we showed in Section 4, we now show that it is not the case. Recall the following computation in Section 4

$$\begin{aligned}\sum_{\ell=0}^{\log t} b_\ell^2 &= \sum_{\ell=0}^{\log t} \frac{(\ell+1)^{2(1-\lambda)}}{\varepsilon^2} \\ &\le \frac{1}{\varepsilon^2}\left(1 + \int_1^{\log(t)+2} x^{2(1-\lambda)}\mathsf{d}x\right) \\ &\le \frac{1}{\varepsilon^2}\left(1 + \left[\frac{1}{3-2\lambda}x^{3-2\lambda}\right]_{x=1}^{x=\log(t)+2}\right) \quad .\end{aligned}$$

Note that we did not include the last step. Now, setting the limits, we get

$$\sum_{\ell=0}^{\log t} b_\ell^2 \le \frac{1}{\varepsilon^2}\left(1 + \frac{(\log(t)+2)^{3-2\lambda}-1}{3-2\lambda}\right)$$

Just as in the case of the privacy proof, we can compute the limit by applying L'Hôpital rule to get the desired expression.

C.4 Omitted Proofs from Section 5

Proof of Theorem 5.1. We define $x^{(0)} = 0^T$ to be the 0 stream of length T. Let $T' \le T$ such that $2C$ divides T', and $T - T' < 2C$. Since $T \ge 2C$ we have $T' \ge 2C$ and $T' \ge T/2$. We define $T'/(2C)$ data sets $x^{(1)}, \ldots, x^{(T'/(2C))}$ as follows:

$$x_t^{(i)} := \begin{cases} 1 & t \in [2C(i-1)+1, 2Ci] \\ 0 & \text{otherwise} \end{cases} .$$

For $i \in [T'/(2C)]$, let S_i be the set of all output sequences $a_1, \ldots, a_T \in \text{range}(\mathcal{A})$ satisfying $a_{2Ci} > C$, and $a_{2Ck} < C$ for all $k < i$. Note that $S_i \cap S_j = \emptyset$ for $i \ne j$ where $i, j \in [T'/(2C)]$. Further, since by assumption $\mathcal{A}$ has error at most C at all times steps with probability at least $2/3$, we have that

$$\Pr[\mathcal{A}(x_1^{(i)}, \ldots, x_t^{(i)})_{t=1}^{2Ci} \in S_i] \ge 2/3.$$

Recall that $x^{(0)}$ and $x^{(i)}$differ in exactly the positions $t \in [2C(i-1)+1, 2Ci]$. By the assumption that $\mathcal{A}$ satisfies ε-differential privacy with expiration g,

$$\begin{aligned}2/3 &\le \Pr[\mathcal{A}(x_1^{(i)}, \ldots, x_t^{(i)})_{t=1}^{2Ci} \in S_i] \\ &\le e^{\sum_{j=0}^{2C-1} g(j)\varepsilon} \Pr[\mathcal{A}(x_1^{(0)}, \ldots, x_t^{(0)})_{t=1}^{2Ci} \in S_i] \\ &= e^{\sum_{j=0}^{2C-1} g(j)\varepsilon} \Pr[\mathcal{A}(x_1^{(0)}, \ldots, x_t^{(0)})_{t=1}^{T'} \in S_i]\end{aligned}$$

Now, since the S_i's are disjoint, we have

$$\begin{aligned}1 &\ge \sum_{i=1}^{T'/(2C)} \Pr[\mathcal{A}(x_1^{(0)}, \ldots, x_t^{(0)})_{t=1}^{T'} \in S_i] \\ &\ge \sum_{i=1}^{T'/(2C)} (2/3)e^{-\sum_{j=0}^{2C-1} g(j)\varepsilon} \\ &= (T'/(2C))(2/3)e^{-\sum_{j=0}^{2C-1} g(j)\varepsilon}.\end{aligned}$$

It follows that

$$\sum_{j=0}^{2C-1} g(j)\varepsilon \ge \log(2T'/(6C)) \ge \log(T/6C)$$

completing the proof of Theorem 5.1. □

https://doi.org/10.52202/079017-0299

NeurIPS Paper Checklist

1. **Claims**

   Question: Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope?

   Answer: [Yes]

   Justification: All theoretical results claimed are backed up by Theorems 1.2 and 1.4, both of which have complete proofs, and our claims about empirical results are backed up by experiments, which can be reproduced using the code in the supplementary material. We also believe we accurately contextualize our contribution in the existing literature.

   Guidelines:

   - The answer NA means that the abstract and introduction do not include the claims made in the paper.
   - The abstract and/or introduction should clearly state the claims made, including the contributions made in the paper and important assumptions and limitations. A No or NA answer to this question will not be perceived well by the reviewers.
   - The claims made should match theoretical and experimental results, and reflect how much the results can be expected to generalize to other settings.
   - It is fine to include aspirational goals as motivation as long as it is clear that these goals are not attained by the paper.

2. **Limitations**

   Question: Does the paper discuss the limitations of the work performed by the authors?

   Answer: [Yes]

   Justification: Yes, we believe we do. For our theoretical results, the clearest limitation is that it only holds for pure DP, not for approximate DP. We note that our techniques should extend to zero-concentrated DP in Section 7. The other practical limitation is likely to be whether a framework that allows for a privacy loss that increases with time is of practical interest. We argue for its relevance in Section 1, by pointing out that (i) it exists in practice, and, (ii) actively taking privacy expiration into account allows for better trade-offs between privacy and error (our Theorem 1.2) than what is achieved in practice.

   Guidelines:

   - The answer NA means that the paper has no limitation while the answer No means that the paper has limitations, but those are not discussed in the paper.
   - The authors are encouraged to create a separate "Limitations" section in their paper.
   - The paper should point out any strong assumptions and how robust the results are to violations of these assumptions (e.g., independence assumptions, noiseless settings, model well-specification, asymptotic approximations only holding locally). The authors should reflect on how these assumptions might be violated in practice and what the implications would be.
   - The authors should reflect on the scope of the claims made, e.g., if the approach was only tested on a few datasets or with a few runs. In general, empirical results often depend on implicit assumptions, which should be articulated.
   - The authors should reflect on the factors that influence the performance of the approach. For example, a facial recognition algorithm may perform poorly when image resolution is low or images are taken in low lighting. Or a speech-to-text system might not be used reliably to provide closed captions for online lectures because it fails to handle technical jargon.
   - The authors should discuss the computational efficiency of the proposed algorithms and how they scale with dataset size.
   - If applicable, the authors should discuss possible limitations of their approach to address problems of privacy and fairness.
   - While the authors might fear that complete honesty about limitations might be used by reviewers as grounds for rejection, a worse outcome might be that reviewers discover limitations that aren't acknowledged in the paper. The authors should use their best

judgment and recognize that individual actions in favor of transparency play an important role in developing norms that preserve the integrity of the community. Reviewers will be specifically instructed to not penalize honesty concerning limitations.

3. **Theory Assumptions and Proofs**

   Question: For each theoretical result, does the paper provide the full set of assumptions and a complete (and correct) proof?

   Answer: [Yes]

   Justification: All our theoretical results are provided with the full set of assumptions and are backed up by proofs, which, if not in the main part of the paper, can be found in Appendix C.

   Guidelines:

   - The answer NA means that the paper does not include theoretical results.
   - All the theorems, formulas, and proofs in the paper should be numbered and cross-referenced.
   - All assumptions should be clearly stated or referenced in the statement of any theorems.
   - The proofs can either appear in the main paper or the supplemental material, but if they appear in the supplemental material, the authors are encouraged to provide a short proof sketch to provide intuition.
   - Inversely, any informal proof provided in the core of the paper should be complemented by formal proofs provided in appendix or supplemental material.
   - Theorems and Lemmas that the proof relies upon should be properly referenced.

4. **Experimental Result Reproducibility**

   Question: Does the paper fully disclose all the information needed to reproduce the main experimental results of the paper to the extent that it affects the main claims and/or conclusions of the paper (regardless of whether the code and data are provided or not)?

   Answer: [Yes]

   Justification: The full experimental setup, together with the results, is given in Section 6.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - If the paper includes experiments, a No answer to this question will not be perceived well by the reviewers: Making the paper reproducible is important, regardless of whether the code and data are provided or not.
   - If the contribution is a dataset and/or model, the authors should describe the steps taken to make their results reproducible or verifiable.
   - Depending on the contribution, reproducibility can be accomplished in various ways. For example, if the contribution is a novel architecture, describing the architecture fully might suffice, or if the contribution is a specific model and empirical evaluation, it may be necessary to either make it possible for others to replicate the model with the same dataset, or provide access to the model. In general. releasing code and data is often one good way to accomplish this, but reproducibility can also be provided via detailed instructions for how to replicate the results, access to a hosted model (e.g., in the case of a large language model), releasing of a model checkpoint, or other means that are appropriate to the research performed.
   - While NeurIPS does not require releasing code, the conference does require all submissions to provide some reasonable avenue for reproducibility, which may depend on the nature of the contribution. For example
     (a) If the contribution is primarily a new algorithm, the paper should make it clear how to reproduce that algorithm.
     (b) If the contribution is primarily a new model architecture, the paper should describe the architecture clearly and fully.
     (c) If the contribution is a new model (e.g., a large language model), then there should either be a way to access this model for reproducing the results or a way to reproduce the model (e.g., with an open-source dataset or instructions for how to construct the dataset).

https://doi.org/10.52202/079017-0299

(d) We recognize that reproducibility may be tricky in some cases, in which case authors are welcome to describe the particular way they provide for reproducibility. In the case of closed-source models, it may be that access to the model is limited in some way (e.g., to registered users), but it should be possible for other researchers to have some path to reproducing or verifying the results.

5. **Open access to data and code**

   Question: Does the paper provide open access to the data and code, with sufficient instructions to faithfully reproduce the main experimental results, as described in supplemental material?

   Answer: [Yes]

   Justification: The code for reproducing the result is provided as a zip file containing documentation and Python code. In particular, executing the included Jupyter Notebook will reproduce our figures.

   Guidelines:

   - The answer NA means that paper does not include experiments requiring code.
   - Please see the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - While we encourage the release of code and data, we understand that this might not be possible, so "No" is an acceptable answer. Papers cannot be rejected simply for not including code, unless this is central to the contribution (e.g., for a new open-source benchmark).
   - The instructions should contain the exact command and environment needed to run to reproduce the results. See the NeurIPS code and data submission guidelines (`https://nips.cc/public/guides/CodeSubmissionPolicy`) for more details.
   - The authors should provide instructions on data access and preparation, including how to access the raw data, preprocessed data, intermediate data, and generated data, etc.
   - The authors should provide scripts to reproduce all experimental results for the new proposed method and baselines. If only a subset of experiments are reproducible, they should state which ones are omitted from the script and why.
   - At submission time, to preserve anonymity, the authors should release anonymized versions (if applicable).
   - Providing as much information as possible in supplemental material (appended to the paper) is recommended, but including URLs to data and code is permitted.

6. **Experimental Setting/Details**

   Question: Does the paper specify all the training and test details (e.g., data splits, hyperparameters, how they were chosen, type of optimizer, etc.) necessary to understand the results?

   Answer: [Yes]

   Justification: The details provided in Section 6 should be sufficient to understand our empirical results, which involve no training data.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The experimental setting should be presented in the core of the paper to a level of detail that is necessary to appreciate the results and make sense of them.
   - The full details can be provided either with the code, in appendix, or as supplemental material.

7. **Experiment Statistical Significance**

   Question: Does the paper report error bars suitably and correctly defined or other appropriate information about the statistical significance of the experiments?

   Answer: [Yes]

   Justification: Our experiments involves exact computations of bounds on the privacy loss for different algorithms. These computations are exact and involve no randomness, so no error bars are necessary, and this information is stated in Section 6.

Guidelines:

- The answer NA means that the paper does not include experiments.
- The authors should answer "Yes" if the results are accompanied by error bars, confidence intervals, or statistical significance tests, at least for the experiments that support the main claims of the paper.
- The factors of variability that the error bars are capturing should be clearly stated (for example, train/test split, initialization, random drawing of some parameter, or overall run with given experimental conditions).
- The method for calculating the error bars should be explained (closed form formula, call to a library function, bootstrap, etc.)
- The assumptions made should be given (e.g., Normally distributed errors).
- It should be clear whether the error bar is the standard deviation or the standard error of the mean.
- It is OK to report 1-sigma error bars, but one should state it. The authors should preferably report a 2-sigma error bar than state that they have a 96% CI, if the hypothesis of Normality of errors is not verified.
- For asymmetric distributions, the authors should be careful not to show in tables or figures symmetric error bars that would yield results that are out of range (e.g. negative error rates).
- If error bars are reported in tables or plots, The authors should explain in the text how they were calculated and reference the corresponding figures or tables in the text.

8. **Experiments Compute Resources**

   Question: For each experiment, does the paper provide sufficient information on the computer resources (type of compute workers, memory, time of execution) needed to reproduce the experiments?

   Answer: [No]

   Justification: Our empirical results plots bounds on privacy loss for different algorithms, but it does not run any of them. As the resources needed to produce the figures is not related to actually running the algorithms considered, these resources are not of interest nor reported.

   Guidelines:

   - The answer NA means that the paper does not include experiments.
   - The paper should indicate the type of compute workers CPU or GPU, internal cluster, or cloud provider, including relevant memory and storage.
   - The paper should provide the amount of compute required for each of the individual experimental runs as well as estimate the total compute.
   - The paper should disclose whether the full research project required more compute than the experiments reported in the paper (e.g., preliminary or failed experiments that didn't make it into the paper).

9. **Code Of Ethics**

   Question: Does the research conducted in the paper conform, in every respect, with the NeurIPS Code of Ethics `https://neurips.cc/public/EthicsGuidelines`?

   Answer: [Yes]

   Justification: Yes, our research involves no data and we otherwise conform to every other aspect in the Code of Ethics.

   Guidelines:

   - The answer NA means that the authors have not reviewed the NeurIPS Code of Ethics.
   - If the authors answer No, they should explain the special circumstances that require a deviation from the Code of Ethics.
   - The authors should make sure to preserve anonymity (e.g., if there is a special consideration due to laws or regulations in their jurisdiction).

10. **Broader Impacts**

https://doi.org/10.52202/079017-0299

Question: Does the paper discuss both potential positive societal impacts and negative societal impacts of the work performed?

Answer: [NA]

Justification: Our work has the potential to expand the applications of differential privacy, the societal impact of which depends on the particular use case.

Guidelines:

- The answer NA means that there is no societal impact of the work performed.
- If the authors answer NA or No, they should explain why their work has no societal impact or why the paper does not address societal impact.
- Examples of negative societal impacts include potential malicious or unintended uses (e.g., disinformation, generating fake profiles, surveillance), fairness considerations (e.g., deployment of technologies that could make decisions that unfairly impact specific groups), privacy considerations, and security considerations.
- The conference expects that many papers will be foundational research and not tied to particular applications, let alone deployments. However, if there is a direct path to any negative applications, the authors should point it out. For example, it is legitimate to point out that an improvement in the quality of generative models could be used to generate deepfakes for disinformation. On the other hand, it is not needed to point out that a generic algorithm for optimizing neural networks could enable people to train models that generate Deepfakes faster.
- The authors should consider possible harms that could arise when the technology is being used as intended and functioning correctly, harms that could arise when the technology is being used as intended but gives incorrect results, and harms following from (intentional or unintentional) misuse of the technology.
- If there are negative societal impacts, the authors could also discuss possible mitigation strategies (e.g., gated release of models, providing defenses in addition to attacks, mechanisms for monitoring misuse, mechanisms to monitor how a system learns from feedback over time, improving the efficiency and accessibility of ML).

11. **Safeguards**

Question: Does the paper describe safeguards that have been put in place for responsible release of data or models that have a high risk for misuse (e.g., pretrained language models, image generators, or scraped datasets)?

Answer: [NA]

Justification: Not relevant to our research.

Guidelines:

- The answer NA means that the paper poses no such risks.
- Released models that have a high risk for misuse or dual-use should be released with necessary safeguards to allow for controlled use of the model, for example by requiring that users adhere to usage guidelines or restrictions to access the model or implementing safety filters.
- Datasets that have been scraped from the Internet could pose safety risks. The authors should describe how they avoided releasing unsafe images.
- We recognize that providing effective safeguards is challenging, and many papers do not require this, but we encourage authors to take this into account and make a best faith effort.

12. **Licenses for existing assets**

Question: Are the creators or original owners of assets (e.g., code, data, models), used in the paper, properly credited and are the license and terms of use explicitly mentioned and properly respected?

Answer: [NA]

Justification: We use a standard open-source Python environment for our experiments.

Guidelines:

- The answer NA means that the paper does not use existing assets.
- The authors should cite the original paper that produced the code package or dataset.
- The authors should state which version of the asset is used and, if possible, include a URL.
- The name of the license (e.g., CC-BY 4.0) should be included for each asset.
- For scraped data from a particular source (e.g., website), the copyright and terms of service of that source should be provided.
- If assets are released, the license, copyright information, and terms of use in the package should be provided. For popular datasets, `paperswithcode.com/datasets` has curated licenses for some datasets. Their licensing guide can help determine the license of a dataset.
- For existing datasets that are re-packaged, both the original license and the license of the derived asset (if it has changed) should be provided.
- If this information is not available online, the authors are encouraged to reach out to the asset's creators.

13. **New Assets**

    Question: Are new assets introduced in the paper well documented and is the documentation provided alongside the assets?

    Answer: [NA]

    Justification: We do not release any new assets.

    Guidelines:

    - The answer NA means that the paper does not release new assets.
    - Researchers should communicate the details of the dataset/code/model as part of their submissions via structured templates. This includes details about training, license, limitations, etc.
    - The paper should discuss whether and how consent was obtained from people whose asset is used.
    - At submission time, remember to anonymize your assets (if applicable). You can either create an anonymized URL or include an anonymized zip file.

14. **Crowdsourcing and Research with Human Subjects**

    Question: For crowdsourcing experiments and research with human subjects, does the paper include the full text of instructions given to participants and screenshots, if applicable, as well as details about compensation (if any)?

    Answer: [NA]

    Justification: No crowdsourcing nor research with human subjects are involved in our paper.

    Guidelines:

    - The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
    - Including this information in the supplemental material is fine, but if the main contribution of the paper involves human subjects, then as much detail as possible should be included in the main paper.
    - According to the NeurIPS Code of Ethics, workers involved in data collection, curation, or other labor should be paid at least the minimum wage in the country of the data collector.

15. **Institutional Review Board (IRB) Approvals or Equivalent for Research with Human Subjects**

    Question: Does the paper describe potential risks incurred by study participants, whether such risks were disclosed to the subjects, and whether Institutional Review Board (IRB) approvals (or an equivalent approval/review based on the requirements of your country or institution) were obtained?

    Answer: [NA]

    Justification: No crowdsourcing nor research with human subjects are involved in our paper.

https://doi.org/10.52202/079017-0299

Guidelines:

- The answer NA means that the paper does not involve crowdsourcing nor research with human subjects.
- Depending on the country in which research is conducted, IRB approval (or equivalent) may be required for any human subjects research. If you obtained IRB approval, you should clearly state this in the paper.
- We recognize that the procedures for this may vary significantly between institutions and locations, and we expect authors to adhere to the NeurIPS Code of Ethics and the guidelines for their institution.
- For initial submissions, do not include any information that would break anonymity (if applicable), such as the institution conducting the review.