Optimal Pure Differentially Private Sparse Histograms in Deterministic Linear Time

FLORIAN KERSCHBAUM, Cheriton School of Computer Science, University of Waterloo, Canada
STEVEN LEE, Cheriton School of Computer Science, University of Waterloo, Canada
HAO WU, Cheriton School of Computer Science, University of Waterloo, Canada

We present an algorithm that releases a pure differentially private (under the replacement neighboring relation) sparse histogram for n participants over a domain of size $d \gg n$. Our method achieves the optimal ℓ_∞-estimation error and runs in strictly $O(n)$ time in the Word-RAM model, improving upon the previous best deterministic-time bound of $\tilde{O}(n^2)$ and resolving the open problem of breaking this quadratic barrier (Balcer and Vadhan 2019). Moreover, the algorithm admits an efficient circuit implementation, enabling the first near-linear communication and computation cost pure DP histogram MPC protocol with optimal ℓ_∞-estimation error. Central to our algorithm is a novel *private item blanket* technique with target-length padding, which hides differences in the supports of neighboring histograms while remaining efficiently implementable.

CCS Concepts: • **Security and privacy → Privacy-preserving protocols**; **Data anonymization and sanitization**; • **Theory of computation → Design and analysis of algorithms**.

Additional Key Words and Phrases: Differential Privacy, Sparse Histogram

ACM Reference Format:
Florian Kerschbaum, Steven Lee, and Hao Wu. 2026. Optimal Pure Differentially Private Sparse Histograms in Deterministic Linear Time. *Proc. ACM Manag. Data* 4, 2 (PODS), Article 113 (May 2026), 27 pages. https://doi.org/10.1145/3801909

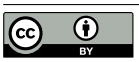
1 Introduction

Differential privacy is a rigorous mathematical framework for protecting individual data in statistical analyses. It ensures that algorithms produce similar output distributions on neighboring datasets—those differing in a single individual's data—making it difficult to infer whether any particular individual is included in the input. This strong privacy guarantee has made differential privacy widely adopted in both theory and practice.

We focus on one of the most fundamental tasks in differential privacy: histogram publishing. In this setting, there are n participants, each holding an element drawn from the domain $[d] = [1\,..\,d]$. The histogram, $\vec{h} \in [0\,..\,n]^d$, is a vector where $\vec{h}[i]$ equals the number of times element i appears among the participants. The goal is to publish a differentially private version $\tilde{h}$ that well approximates $\vec{h}$ while protecting individual participant's data.

One of the earliest and simplest solutions for privatizing a histogram $\vec{h}$ is the Laplace mechanism [21], which adds independent, continuous Laplace noise to each entry of $\vec{h}$. This approach achieves various asymptotically optimal error guarantees [6, 28]. However, implementing the mechanism presents subtle challenges, as it assumes operations on real numbers and requires operating on all

Authors' Contact Information: Florian Kerschbaum, florian.kerschbaum@uwaterloo.ca, Cheriton School of Computer Science, University of Waterloo, Ontario, Canada; Steven Lee, hj44lee@uwaterloo.ca, Cheriton School of Computer Science, University of Waterloo, Ontario, Canada; Hao Wu, hao.wu1@uwaterloo.ca, Cheriton School of Computer Science, University of Waterloo, Ontario, Canada.



ACM 2836-6573/2026/5-ART113
https://doi.org/10.1145/3801909

entries of $\vec{h}$. The former can introduce potential privacy vulnerabilities, while the latter may incur prohibitive computational costs, which we elaborate on below.

Floating-Point Attack. Sampling Laplace noise requires representing real numbers on discrete machines, typically via double-precision floating-point arithmetic. As early as 2012, Mironov [38] observed that due to finite precision and rounding artifacts, certain floating-point values cannot be generated. These artifacts can be exploited to distinguish exactly between neighboring inputs based on the outputs of the Laplace mechanism, thereby completely compromising its privacy guarantees. A fundamental mitigation is to avoid real-number arithmetic altogether by using discrete analogues of Laplace noise—such as the discrete Laplace distribution [25].

Timing Attack. However, avoiding floating-point vulnerabilities introduces a new threat: the timing attack. Since discrete Laplace noise has unbounded support, its sampler can require unbounded memory and has running time only bounded in expectation [12]. Jin et al. [29] showed a positive correlation between the sampled value and the algorithm's running time, allowing adversaries to infer the output from timing information—again violating privacy. This motivates the design of *time-oblivious* algorithms, whose running time leaks no—or only a limited amount of—information about the input. For instance, one possible approach is to enforce strict upper bounds on running time (ideally polynomial in the bit length of the input), allowing all executions to be padded to run for the same duration.

Exponential-Size Domains. Adding noise to all entries of $\vec{h}$ results in a running time proportional to d, which can be exponential in the input size. For example, consider the task of identifying popular URLs of up to 20 characters in length,[1] as discussed in Fanti et al. [22]. This setting corresponds to a domain size of at least $d \geq 10^{36}$. To avoid adding noise to all entries, a natural idea is to publish a sparse histogram $\tilde{h}$ to approximate the original histogram $\vec{h}$, since the latter is also sparse: it has at most n nonzero entries. Some early works along this line, either satisfy only approximate differential privacy [11, 34]—in contrast to the pure differential privacy guaranteed by the Laplace mechanism [2]—or require sampling from complicated binomial distributions [15], for which it remains open whether one can sample efficiently while avoiding timing attacks [3].

Research Problem: Design an efficient algorithm for publishing a pure differentially private histogram that avoids both floating-point and timing attacks.

The Quadratic Time Barrier. Balcer and Vadhan [3] initiated a systematic study of differentially private sparse histograms and proposed a sparsified variant of the Laplace mechanism suited for Word-RAM model. They designed a deterministic-time sampler for a relaxed discrete Laplace distribution supported on a finite domain, as well as an algorithm that reports only the top-n elements with the highest noisy counts, without explicitly adding noise to all entries in $\vec{h}$. Their algorithm achieves asymptotically optimal error among sparse histograms. However, the noise sampler has a deterministic per-sample cost of $\tilde{O}(1/\varepsilon)$, and identifying the top-n elements requires $\tilde{O}(n^2)$ (deterministic) time. The paper concludes with the following open question:

Research Question: Can one design deterministic and linear-time pure differentially private histogram algorithms in the Word-RAM model while preserving privacy and accuracy guarantees?

[1] Valid URL characters include digits (0–9), letters (A–Z, a–z), and a few special characters ("-", ".", "_", "~").
[2] The distinction between pure and approximate differential privacy will be formally explained in Section 2

Algorithm	ℓ_∞-Error	Model	Type	Running Time
Laplace Mechanism [18]	$\Theta(1/\varepsilon \cdot \ln d)$	Real-RAM	Deterministic	$O(d)$
Geometric Mechanism [25]	$\Theta(1/\varepsilon \cdot \ln d)$	Word-RAM	Randomized	Expected $O(d)$
Filter Mechanism [15, 40]	$\Theta(1/\varepsilon \cdot \ln d)$	Real-RAM	Randomized	Expected $O(n)$
PureSparseHistogram [3]	$\Theta(1/\varepsilon \cdot \ln d)$	Word-RAM	Deterministic	$\tilde{O}(n^2)$
Theorem 1.1	$\Theta(1/\varepsilon \cdot \ln d)$	Word-RAM	Deterministic	$O(n)$

Table 1. Comparison of sparse histogram algorithms: error, model, type, and running time. The Real-RAM model assumes unit-cost arithmetic on infinite-precision floating-point numbers, while the Word-RAM model assumes unit-cost arithmetic on standard word-sized values.

1.1 Our Contributions

Our contributions are twofold. First, we provide a positive answer to the open question posed by Balcer and Vadhan [3]. Second, as a byproduct, we develop a general framework of deterministic-time approximate sampler for arbitrary discrete distributions, and instantiate it specifically for generating discrete Laplace noise. We discuss both results separately.

Histogram Construction. Consistent with [3], our construction builds on Word-RAM model, and the privacy guarantee is respect to *replacement neighboring relation*, both formally defined in Section 2.

Theorem 1.1 (Private Sparse Histogram, Informal version of Theorem 4.1). *Let $n, d \in \mathbb{N}_+$ and $\varepsilon \in \mathbb{Q}_+$, with representations that fit in a constant number of machine words. Given a dataset $\mathcal{X}$ of n user-contributed elements from domain $[d]$, there exists an ε-differentially private algorithm that runs deterministically in $O(n)$ time and outputs a histogram $\tilde{h} \in \mathbb{N}^d_{\le n}$ approximating the original histogram $\vec{h}$ of $\mathcal{X}$, such that $\|\tilde{h}\|_0 \in O(n)$ and $\mathbb{E}\left[\|\vec{h} - \tilde{h}\|_\infty\right] \in O\left(\frac{1}{\varepsilon} \cdot \ln d\right)$.*

The expected error bound above is asymptotically optimal, matching the lower bounds established in [3, 6, 28]. Table 1 compares our algorithm with the previous ones. Beyond its applications to private data mining and machine learning, our algorithm also helps *close a longstanding utility gap between the central and distributed differential privacy models for histograms.*

From Central to Distributed DP: Matching Utility. Our algorithm operates in the *central* DP model, where a trusted server collects participants' data to release a private histogram. A commonly studied alternative is the *distributed* DP (DDP) model, where the server is untrusted. The simplest DDP setting is the *local* model, where each participant perturbs their own data before reporting it to the server. Unfortunately, this added privacy comes at a steep cost: local protocols incur substantial noise, and the best known frequency-estimation protocol achieves an expected ℓ_∞ error of $\Theta(\frac{1}{\varepsilon} \cdot \sqrt{n \ln d})$ [5, 10, 45], creating a tight $\Omega(\sqrt{n})$ utility gap compared to the central model.

A significant body of work has attempted to bridge this gap by enabling participants (and, in some settings, multiple servers) to jointly and securely compute a function such that no party learns anything beyond their own input and the final DP output. Initial efforts focused on the *shuffle model* [4, 9, 14, 24], where the joint computation is restricted to randomly permuting locally perturbed messages. Despite these advances, a gap remains: even the best near–linear-time shuffle protocols [24] still incur a polynomial (in $\log d$) degradation in utility compared to the optimal bounds achieved in the central model.

Recent works have further bridged this gap by exploiting the full potential of the DDP model, supporting richer classes of functionalities beyond shuffling via secure multiparty computation (MPC), as demonstrated by systems from Mozilla (Prio) [16], Google [7], and Microsoft [1]. This line of work culminated in a recent (ε, δ)-DP sparse histogram algorithm [7] that achieves optimal

Sampler	Space Usage (words)	Running Time / Sample	TV Distance
Dov et al. [17]	$O\left(\frac{1}{\omega} \cdot \lvert C_{\delta/2}\rvert \cdot \frac{1}{\delta} \cdot \ln \lvert C_{\delta/2}\rvert\right)$	$\frac{1}{\omega} \cdot (\log \lvert C_{\delta/2}\rvert + \log \frac{1}{\delta}) + O(1)$	$\leq \delta$
Theorem 1.3	$O\left(\frac{1}{\omega} \cdot \lvert C_{\delta/2}\rvert \cdot \left(\ln \frac{1}{\delta} + \ln \lvert C_{\delta/2}\rvert\right)\right)$	$\frac{1}{\omega} \cdot (\log \lvert C_{\delta/2}\rvert + \log \frac{1}{\delta}) + O(1)$	$\leq \delta$

Table 2. Comparison of deterministic approximate samplers for a discrete distribution μ. Here, $C_{\delta/2}$ denotes a smallest subset satisfying $\mu(C_{\delta/2}) \geq 1 - \delta/2$, and ω is the word size. Our sampler improves space complexity by reducing its dependence on δ from $1/\delta$ to $\ln(1/\delta)$.

ℓ_∞ error, on par with the best central-model guarantees. *However, a Pure ε-DP counterpart with optimal error remains elusive.* The primary obstacle has been the absence of efficient, time-oblivious Pure DP algorithms, even within the central model. Such algorithms are necessary in the MPC setting because data-dependent execution time or branching patterns can introduce side channels that compromise the privacy of the secret participant inputs. Our algorithm overcomes this barrier: it admits an efficient circuit implementation, enabling secure execution in the MPC setting using standard techniques [8, 26].

Theorem 1.2 (Private Sparse Histogram Circuit, Informal Version). *Let $n, d \in \mathbb{N}_+$ and $\varepsilon \in \mathbb{Q}_+$, with representations that fit in a constant number of machine words. There exists a circuit with $\tilde{O}(n\omega)$ gates (where ω is the machine word size) implementing an ε-DP mechanism that takes a dataset $\mathcal{X}$ of n user-contributed elements from domain $[d]$ as input, and outputs a histogram $\tilde{h} \in \mathbb{N}^d_{\leq n}$ approximating the original histogram $\vec{h}$ of $\mathcal{X}$, such that $\|\tilde{h}\|_0 \in O(n)$ and $\mathbb{E}\left[\|\vec{h} - \tilde{h}\|_\infty\right] \in O(\frac{1}{\varepsilon} \cdot \ln d)$.*

The number of gates scales with $n\omega$ rather than n alone because, in the circuit model, merely reading n input elements from $[d]$ (with each element described using $\log d \in O(\omega)$ bits) requires $O(n\omega)$ gates; hence, $n\omega$ is considered linear in the input size. Additionally, simulating Random Access Memory (RAM) operations of our algorithm in the circuit model introduces a logarithmic overhead in the gate complexity.

Noisy Generation. We develop a general framework for constructing deterministic-time approximate samplers for arbitrary discrete distributions. This framework is of independent interest and applies broadly to differentially private noise-generation tasks. In addition, it strengthens the result of Dov, David, Naor, and Tzalik [17] by achieving comparable running time and accuracy guarantees while using asymptotically less space.

Theorem 1.3 (Deterministic Approximate Sampler, Informal Version). *Let μ be a distribution over a discrete domain, let $\delta \in (0, 1)$, and let $C_{\delta/2}$ denote a smallest subset satisfying $\mu(C_{\delta/2}) \geq 1 - \delta/2$. Then there exists a deterministic-time sampler for μ with the following guarantees:*

- ***Memory:*** $O\left(\frac{1}{\omega} \lvert C_{\delta/2}\rvert \left(\ln \frac{1}{\delta} + \ln \lvert C_{\delta/2}\rvert\right)\right)$, *with ω the Word-RAM word size.*
- ***Sampling time per draw:*** $\frac{1}{\omega}\left(\log \lvert C_{\delta/2}\rvert + \log \frac{1}{\delta}\right) + O(1)$.
- ***Accuracy:*** *The resulting distribution has total variation distance at most δ from μ.*

Comparison to Prior Work. For reference, Dov, David, Naor, and Tzalik [17] (Claim 2.22) achieve the same guarantees for sampling time and accuracy (Items 2 and 3 above). However, their sampler requires space $O\left(\omega^{-1} \lvert C_{\delta/2}\rvert\, \delta^{-1} \ln \lvert C_{\delta/2}\rvert\right)$, which is exponentially worse in its dependence on δ compared to our construction. Table 2 summarizes the comparison.

We now instantiate our general framework to generate a variant of discrete Laplace noise tailored for perturbing elements in the bounded range $[0\,..\,n]$. This instantiation incorporates two key optimizations: (1) we adopt a technique from Balcer and Vadhan [3] (see Section 3) to mix the sampled noise with a uniform distribution over $[0\,..\,n]$, which ensures that the truncated output preserves pure differential privacy; (2) we exploit the memoryless property of the discrete Laplace

distribution to further reduce the space required for sampling. The formal guarantees of this sampler are presented below and serve as a crucial building block for our linear-time private histogram protocol.

Theorem 1.4 (Relaxed Discrete Laplace, Informal version). *Let $n \in \mathbb{N}_+$, $\varepsilon \in \mathbb{Q}_+$ and $\gamma \in \mathbb{Q}_+ \cap (0,1)$, with representations that fit in a constant number of machine words. There exists a randomized algorithm $\mathcal{M}_{\mathbb{PA}\text{pxD}\mathbb{L}\text{ap}\,(n,\varepsilon,\gamma)} : \mathbb{N}_{\le n} \to \mathbb{N}_{\le n}$ with initialization time $\tilde{O}(\frac{1}{\varepsilon})$, memory usage $O\big(\frac{1}{\varepsilon} + \ln\frac{1}{\gamma} + \ln n\big)$, deterministic per-query running time $O(1)$ after initialization. It satisfies the following privacy and utility guarantees:*

- *For each $t \in [n]$, $\Pr\left[\mathcal{M}_{\mathbb{PA}\text{pxD}\mathbb{L}\text{ap}\,(n,\varepsilon,\gamma)}\,(t-1) = i\right] / \Pr\left[\mathcal{M}_{\mathbb{PA}\text{pxD}\mathbb{L}\text{ap}\,(n,\varepsilon,\gamma)}\,(t) = i\right] \in [e^{-\varepsilon}, e^{\varepsilon}]$.*
- *For each $t \in \mathbb{N}_{\le n}$, $\beta > 2\cdot\gamma$, $\Pr\left[|\mathcal{M}_{\mathbb{PA}\text{pxD}\mathbb{L}\text{ap}\,(n,\varepsilon,\gamma)}\,(t) - t| \ge \alpha\right] \le \beta$ for $\alpha \in O(\frac{1}{\varepsilon}\cdot\ln\frac{1}{\beta})$.*

Technical Overview. We provide a high-level overview of our private sparse histogram algorithm and noise sampler. More detailed overviews appear in the respective sections where each algorithm is formally presented and analyzed.

Private Histogram. Our algorithm builds on the traditional Laplace mechanism. To mitigate floating-point and timing attacks, a standard approach is to replace the continuous, unbounded Laplace noise with a discrete and truncated counterpart, as we detail in the subsequent paragraph on noise sampler. However, applying this does not resolve the true computational hurdle: processing an exponentially large domain ($d \gg n$) in strictly linear time.

The Core Bottleneck: Intractable Binomial Sampling. If we explicitly add the noise to all d elements, the runtime becomes an unacceptable $O(d)$. To circumvent this, classical algorithms like the Filter mechanism [15] handle zero-count entries implicitly: they add noise to the non-zero entries, and then sample the number of zero entries that exceed a threshold τ directly from a binomial distribution, $\mathbb{B}\text{in}(d - |\text{supp}(\vec{h})|, q_\tau)$, where q_τ is the probability of such an element exceeding threshold τ.

The real challenge lies in this implicit sampling step. Exact sampling from this complex binomial distribution in deterministic time requires exponential space in the word-RAM model—indeed, merely representing the exact probabilities requires exponentially many bits. If one opts for an approximate sampler (e.g., to within total variation distance δ), the algorithm only guarantees approximate DP. To recover pure DP, one could attempt to apply a "purification" technique [3] (mixing the output with a uniform distribution over the output space). However, because the final output of the histogram can be any subset of $[d]$, successful purification demands an exponentially small approximation error $\delta \in O(1/2^d)$. We do not know of any efficient way to sample the binomial distribution to such strict tolerances.

The Solution: Target-Length Padding and the Privacy Blanket. Our algorithm circumvents the binomial sampling bottleneck by fundamentally redesigning how zero-count entries are handled. Instead of explicitly calculating how many zero-entries exceed the threshold, we introduce a *privacy blanket* based on fixed-size target-length padding.

First, we apply noises strictly to the non-zero elements (using our time-oblivious sampler), retaining only those that exceed the threshold τ. Second, to obscure the exact support of the input dataset, we pad the selected elements with uniformly random elements drawn (without replacement) from the remaining domain $[d]$. We continue padding until the combined set reaches a fixed, deterministic size of $O(n)$.

This fixed-size uniform padding serves as a privacy blanket. If two neighboring datasets differ by a single element, the blanket masks this element: it is mathematically indistinguishable whether this element was retained because its true count pushed it over the threshold τ, or whether its true count was zero and it was merely scooped up by the uniform random padding.

A more detailed overview and formal analysis of the algorithm are provided in Section 4. While the optimal utility and efficient running time follow directly from the tail bounds and efficiency of our noise sampler (Theorem 1.4), the privacy analysis is significantly more delicate. It requires a careful, case-by-case argument to prove that the distribution over the final selected elements remains stable across neighboring datasets.

Noise Sampler. Our general framework for approximately sampling from arbitrary discrete distributions builds directly on the classic Alias method [43], applied to carefully chosen truncated supports and truncated probability representations to achieve a prescribed total variation distance. Despite its simplicity, this yields a deterministic-time sampler and improves upon the recent time-oblivious sampling scheme of Dov, David, Naor, and Tzalik [17], making it worthwhile to document in this context.

Next, we instantiate the framework for the discrete Laplace distribution. Specifically, we reduce sampling from the discrete Laplace to sampling geometric noise, exploiting the fact that the discrete Laplace is almost a two-sided geometric distribution. We further leverage the property that geometric noise has identical conditional distributions over intervals of equal length, allowing decomposition into two smaller geometric components. Finally, following a technique by Balcer and Vadhan [3] (see Section 3), we mix the sampled noise with a uniform distribution over $[0\,..\,n]$ to ensure that the truncated output preserves pure differential privacy.

Organization. The remainder of the paper is organized as follows. Section 2 formally introduces the problem. Section 3 reviews the relevant probability distributions and a key building block used in our algorithms. Due to space constraints, we present only our private sparse histogram algorithm in Section 4 and defer the description of our noise-sampling procedures to the full version of our paper [31]. This reordering does not affect the exposition, as Section 4 is self-contained and can be read independently of the sampler implementation details. Section 5 provides a comprehensive overview of related work. The circuit realization of our sparse histogram algorithm also appears in the full version [31].

2 Problem Description

Notation. Let $\mathbb{N}$ denote the set of natural numbers, $\mathbb{Z}$ the set of integers, $\mathbb{Q}$ the set of rational numbers, and $\mathbb{R}$ the set of real numbers. We use $\mathbb{N}_+$, $\mathbb{Q}_+$, and $\mathbb{R}_+$ to denote the sets of positive integers, positive rationals, and positive reals, respectively. For any $n \in \mathbb{N}$, we write $\mathbb{N}_{<n}$ for the set $\{0, 1, \ldots, n-1\}$, $\mathbb{N}_{\leq n}$ for the set $\{0, 1, \ldots, n\}$, and $[n]$ for the set $\{1, \ldots, n\}$.

Computational Model. We design our algorithms in the standard Word-RAM model, where each machine word consists of ω bits. In this model, basic operations on ω-bit words—such as arithmetic, comparisons, bitwise manipulations, and memory access (read/write)—are assumed to take constant time. We also assume that sampling a uniformly random integer from $\mathbb{N}_{<2^\omega}$ takes constant time. Rational inputs are represented as pairs of integers. We measure running time by counting the number of word-level operations, so the possible values of running time lie in $\mathbb{N}$.

2.1 Differentially Private Histogram

Let $\mathcal{D} \doteq \{1, \ldots, d\}$ be a set of d elements and $\mathcal{U} \doteq \{1, \ldots, n\}$ be a set of n participants. Each participant $u \in \mathcal{U}$ holds an element $x^{(u)} \in \mathcal{D}$. The dataset $\mathcal{X} \doteq \{x^{(1)}, \ldots, x^{(n)}\}$ consists of all participant elements. The *frequency* of an element $i \in \mathcal{D}$, denoted by $\vec{h}[i] \doteq \{u \in \mathcal{U} : x^{(u)} = i\}$, is defined as the number of participants holding the element i. The *histogram*, denoted by $\vec{h} \doteq (\vec{h}[1], \ldots, \vec{h}[d]) \in \mathbb{N}_{\leq n}^d$, is the frequency vector.

The objective is to release a histogram $\tilde{h} \in \mathbb{N}^d$ that accurately approximates $\vec{h}$, while preserving the privacy of each individual participant's data, as formally defined below.

Privacy Guarantee. We require that the output $\tilde{h}$ reveal only a limited amount of information about the input. The privacy of the output is formally captured by the notion of differential privacy.

Neighboring Datasets. Consistent with Balcer and Vadhan [3], we adopt the *replacement neighboring* model[3], where two datasets $\mathcal{X}$ and $\mathcal{X}'$ of equal size are *neighboring*, denoted $\mathcal{X} \sim \mathcal{X}'$, if they differ in exactly one participant's data; that is, there exists a unique $u \in [n]$ such that $x^{(u)} \neq x'^{(u)}$ and $x^{(i)} = x'^{(i)}$ for all $i \neq u$. Accordingly, two histograms $\vec{h}, \vec{h}' \in \mathbb{N}^d$ are called *neighboring* if they are induced by neighboring datasets under one of the above definitions.

Private Algorithm. A histogram releasing algorithm is said to be *differentially private* if its output distributions are similar on all pairs of neighboring inputs. Formally:

Definition 2.1 ((ε, δ)-Indistinguishability). *Let $\varepsilon \in \mathbb{R}_+$, $\delta \in [0, 1]$, and let $(\mathcal{Z}, \mathcal{F})$ be a measurable space. Two probability measures μ and ν on this space are (ε, δ)-indistinguishable if*

$$e^{-\varepsilon} \cdot (\mu(E) - \delta) \leq \nu(E) \leq e^{\varepsilon} \cdot \mu(E) + \delta, \quad \forall E \in \mathcal{F}. \tag{1}$$

Similarly, two random variables are (ε, δ)-indistinguishable if their corresponding distributions are.

Definition 2.2 ((ε, δ)-Private Algorithm [19]). *Given $\varepsilon \in \mathbb{R}_+$ and $\delta \in [0, 1]$, and a measurable space $(\mathcal{Z}, \mathcal{F})$, a randomized algorithm $\mathcal{M} : \mathcal{D}^n \to \mathcal{Z}$ is called (ε, δ)-differentially private (DP), if for every $\mathcal{X}, \mathcal{X}' \in \mathcal{D}^n$ such that $\mathcal{X} \sim \mathcal{X}'$, $\mathcal{M}(\mathcal{X})$ and $\mathcal{M}(\mathcal{X}')$ are (ε, δ)-indistinguishable.*

An algorithm $\mathcal{M}$ is said to be *pure differentially private*, or simply *ε-DP*, if it satisfies $(\varepsilon, 0)$-differential privacy. If instead $\mathcal{M}$ satisfies (ε, δ)-differential privacy for some $\delta > 0$, it is referred to as *approximate differentially private*. Although we present differential privacy in the context of histogram release, the definition applies more generally to any randomized algorithm $\mathcal{M} : \mathcal{Y} \to \mathcal{Z}$, where $\mathcal{Y}$ is an arbitrary input domain endowed with a symmetric neighboring relation $\sim$.

Utility Guarantee. We evaluate utility using the ℓ_∞ error [3].

Definition 2.3 ((α, β)-Simultaneous Accurate Estimator). *Let $\alpha \in \mathbb{R}_+$ and $\beta \in [0, 1]$. A random vector $\vec{X} \doteq (X_1, \ldots, X_m)$ is an (α, β)-simultaneously accurate estimator of a vector $\vec{t} \doteq (t_1, \ldots, t_m) \in \mathbb{R}^m$ if $\Pr\left[\|\vec{X} - \vec{t}\|_\infty \geq \alpha\right] \leq \beta$.*

In addition to (α, β)-simultaneous guarantees for $\tilde{h}$, we also provide bounds on the expected error $\mathbb{E}\left[\|\tilde{h} - \vec{h}\|_\infty\right]$; the latter follows from the former using standard integration techniques (see Section 4 for details).

3 Preliminaries

In this section, we define several probability distributions used throughout the paper, and review a framework of converting approximate DP into pure DP algorithms.

[3]Another commonly used notion is the *add/remove neighboring* model. *However, under this model, no optimal-error private frequency histogram algorithm can run in deterministic time.* A formal definition of the model and further discussion appear in the full version of our paper [31]. *This limitation is precisely why Balcer and Vadhan [3] restrict their study to the replacement model, where the question of breaking the quadratic running-time barrier was originally posed.*

3.1 Probability Distributions

Definition 3.1 (Geometric Distribution). *Given $p \in [0,1)$ and $u \in \mathbb{N} \cup \{\infty\}$, let $\mathbb{G}\mathrm{eo}(p, \mathbb{N}_{\le u})$ be a random variable supported on $\mathbb{N}_{\le u}$ with probability mass function*

$$\Pr[\mathbb{G}\mathrm{eo}(p, \mathbb{N}_{\le u}) = t] = \tfrac{1-p}{1-p^{1+u}} \cdot p^t, \quad \forall t \in \mathbb{N}_{\le u}.$$

When $u = \infty$, we write $\mathbb{G}\mathrm{eo}(p)$ as shorthand for $\mathbb{G}\mathrm{eo}(p, \mathbb{N}_{\le \infty})$.

Definition 3.2 (Discrete Laplace Distribution). *Given $p \in [0,1)$, let $\mathbb{DL}\mathrm{ap}(p)$ denote a random variable following the* discrete Laplace distribution *with probability mass function:*

$$\Pr[\mathbb{DL}\mathrm{ap}(p) = t] = \frac{1-p}{1+p} \cdot p^{|t|}, \quad \forall t \in \mathbb{Z}.$$

3.2 Purification

Balcer and Vadhan [3] showed that if an algorithm $\mathcal{M}' : \mathcal{Y} \to \mathcal{Z}$ with finite output domain is close in total variation distance to some pure DP algorithm $\mathcal{M} : \mathcal{Y} \to \mathcal{Z}$, then one can convert $\mathcal{M}'$ into a pure DP algorithm by mixing its output with the uniform distribution over $\mathcal{Z}$. The purification procedure is given in Algorithm 1: with probability $1-\gamma$, the algorithm outputs $\mathcal{M}'(y)$, and otherwise outputs a uniformly random $Z \in \mathcal{Z}$. The formal guarantee is stated in Proposition 3.1.

Algorithm 1 Purification $\mathcal{M}_{\text{purify}}$

Input: An algorithm $\mathcal{M}' : \mathcal{Y} \to \mathcal{Z}$, where $|\mathcal{Z}|$ is finite; input $y \in \mathcal{Y}$; and $\gamma \in \mathbb{Q}_+ \cap (0,1)$
Output: $Z \in \mathcal{Z}$
1: $B \leftarrow \mathbb{B}\mathrm{ernoulli}(\gamma)$ ▷ $\Pr[B = 1] = \gamma$
2: Output $(1-B) \cdot \mathcal{M}'(y) + B \cdot Z$, where $Z \sim \mathbb{U}\mathrm{niform}(\mathcal{Z})$.

Proposition 3.1 ([3]). *Given an algorithm $\mathcal{M}' : \mathcal{Y} \to \mathcal{Z}$, assume there exists an ε-DP algorithm $\mathcal{M} : \mathcal{Y} \to \mathcal{Z}$ such that $\mathbb{TV}(\mathcal{M}(y), \mathcal{M}'(y)) \le \delta$ for all $y \in \mathcal{Y}$, with parameter $\delta \in [0,1)$. Then Algorithm 1 satisfies:*

- *ε-DP, provided that $\delta \le \frac{e^\varepsilon - 1}{e^\varepsilon + 1} \cdot \frac{\gamma}{1-\gamma} \cdot \frac{1}{|\mathcal{Z}|}$.*
- *Running time $O\left(\frac{1}{\omega} \cdot \log \frac{1}{\gamma} + \frac{1}{\omega} \cdot \log|\mathcal{Z}|\right) + T_{\mathcal{M}'}$, where $T_{\mathcal{M}'}$ is the running time of $\mathcal{M}'$.*

4 Differentially Private Sparse Histograms

In this section, we present our algorithm for releasing sparse histograms under pure differential privacy. The following theorem summarizes our main result.

Theorem 4.1 (Private Tail Padding Histogram). *Given integers $n, d, k, a_\varepsilon, b_\varepsilon, a_\gamma, b_\gamma \in \mathbb{N}_+$ that fit in a constant number of machine words, where $\varepsilon \doteq \frac{a_\varepsilon}{b_\varepsilon} \in \mathbb{Q}_+$ and $\gamma \doteq a_\gamma / b_\gamma \in (1/n^{O(1)}, 1)$, and a dataset $\mathcal{X} \doteq \{x^{(1)}, \ldots, x^{(n)}\} \in [d]^n$, there exists an algorithm, denoted by $\mathcal{M}_{\mathit{HistT}}$, that outputs a histogram $\tilde{h} \in \mathbb{N}^d_{\le n}$ approximating the original histogram $\vec{h}$ of $\mathcal{X}$, satisfying $\|\tilde{h}\|_0 \in O(n)$ and*

- ***Privacy Guarantee:** $\mathcal{M}_{\mathit{HistT}}$ satisfies $\max\left(\frac{3\varepsilon}{2} + \frac{1}{k+1}, 2\varepsilon\right)$-differential privacy.*
- ***Utility Guarantee:** There exists a universal constant $c_\alpha \in \mathbb{R}_+$, such that for each $\beta \ge 2\varepsilon\gamma$, $\alpha \doteq c_\alpha/\varepsilon \cdot \ln(d/\beta)$, the output $\tilde{h}$ is an (α, β)-simultaneous accurate estimator of $\vec{h}$.*
- ***Running Time:** $\mathcal{M}_{\mathit{HistT}}$ runs deterministically in $O(n+k)$ time.*

Remark. The utility guarantee of Theorem 4.1 is an ℓ_∞ bound and is asymptotically optimal, matching the lower bound $\Omega(\min\{\frac{1}{\varepsilon}\ln d, n\})$ of Balcer and Vadhan [3]. The bound provided here is a high-probability one, in contrast to the expected one stated in Theorem 1.1. The latter can be recovered from the former by the standard identity $\mathbb{E}[X] = \int_0^\infty \Pr[X \geq t]\, dt$ for a nonnegative random variable X.

Setting $k = 2/\varepsilon$ yields a privacy guarantee of 2ε-DP and running time $O(n+1/\varepsilon)$, which simplifies to $O(n)$ in the meaningful regime $n \geq 1/\varepsilon$. When $n < 1/\varepsilon$, simply returning the all-zero frequency vector incurs error at most $n < 1/\varepsilon$, matching the lower bound.

Also, in what follows, we assume $d \geq 10n$ to focus on the sparse regime where our algorithm is most relevant. Otherwise, when $d < 10n$, one can simply release the entire histogram $\vec{h}$ after perturbing each count using the algorithm in Theorem 1.4, which yields an algorithm with $O(n)$ running time.

Roadmap. The remainder of this section is organized as follows. We begin by discussing Theorem 4.1 and its connection to the informal version stated in the introduction (Theorem 1.1). Section 4.1 presents the key ideas behind our algorithm, along with its pseudocode. Section 4.2 provides the proof of Theorem 4.1.

A detailed comparison with prior work is deferred to Section 5. We present an MPC-friendly prototype circuit implementation of our algorithm in the full version of our paper [31].

4.1 Overview of $\mathcal{M}_{\textsc{Hist}}$

To motivate our algorithm, we begin by reviewing the Filter algorithm for releasing pure-DP sparse histograms [15], which lacks a known efficient time-oblivious implementation. Given a histogram $\vec{h} \in \mathbb{N}^d_{\leq n}$ with $\|\vec{h}\|_1 = n$, the Filter algorithm produces an output $\tilde{h}$ distributed identically to the output of the following procedure: (1) Add independent $\mathbb{DL}\text{ap}\,(e^{-\varepsilon})$ noise to each entry of $\vec{h}$. (2) Retain only the entries whose noisy values exceed a threshold $\tau \in \Theta\left(\frac{1}{\varepsilon}\cdot\ln d\right)$, chosen so that the expected number of outputs is $O(n)$.

Implementing the above procedure directly takes $O(d)$ time. To avoid this, the Filter algorithm treats non-zero and zero entries separately. Non-zero entries are processed directly as described, taking $O(n)$ time. For zero entries, Cormode et al. [15] observed that after noise addition, each entry exceeds the threshold τ independently with the same probability, denoted by $q_\tau \in O(n/d)$. Thus, the algorithm proceeds as follows:

(1) Sample the number X of zero entries exceeding τ from a binomial distribution $\mathbb{B}\text{in}(d - |\textsc{supp}(\vec{h})|, q_\tau)$ where $\textsc{supp}(\vec{h})$ denotes the support of $\vec{h}$.
(2) Sample a random subset of size X from $[d] \setminus \textsc{supp}(\vec{h})$.
(3) Assign i.i.d. noise to the X entries from distribution $\mathbb{DL}\text{ap}\,(e^{-\varepsilon})$, conditioned on exceeding τ.

Obstacles to a Time-Oblivious Implementation. Though Filter has a simple description, implementing it properly is not straightforward. The first issue is that sampling from $\mathbb{DL}\text{ap}\,(e^{-\varepsilon})$ exactly is vulnerable to timing attacks. To address this, one can replace $\mathbb{DL}\text{ap}\,(e^{-\varepsilon})$ with a noise distribution with similar properties that has deterministic sampling time, such as the one presented in Theorem 1.4. In addition, one can avoid sampling from the conditional distribution in the last step by instead using fresh noises, at the slight cost of a factor-2 degradation in utility.

The real challenge lies in sampling from the distribution $\mathbb{B}\text{in}(d - |\textsc{supp}(\vec{h})|, q_\tau)$. Even writing down the exact probabilities of this binomial distribution requires exponential space. A natural idea is to approximately sample: generate a random variable X such that $\mathbb{TV}(X, \mathbb{B}\text{in}(d - |\textsc{supp}(\vec{h})|, q_\tau)) \leq \delta$ for some small δ. This leads to an algorithm that satisfies approximate DP. To recover pure DP,

Algorithm 2 Private Tail Padding Histogram $\mathcal{M}_{\text{HIST}}$

Input: parameters $d, n, k, a_\varepsilon, b_\varepsilon, a_\gamma, b_\gamma \in \mathbb{N}_+$, s.t., $\varepsilon = a_\varepsilon / b_\varepsilon$ and $\gamma = a_\gamma / b_\gamma$
dataset $\mathcal{X} \doteq \{x^{(1)}, \ldots, x^{(n)}\} \in [d]^n$
Output: histogram $\tilde{h} \in \mathbb{N}^d_{\le n}$

construct histogram $\vec{h} \in \mathbb{N}^d_{\le n}$ based on $\mathcal{X}$
for each $i \in \text{SUPP}(\vec{h})$ **do**
  $\hat{h}[i] \leftarrow \mathcal{M}_{\mathbb{PA}\text{px}\mathbb{DL}\text{ap}(n,\varepsilon,\varepsilon\gamma/d)}(\vec{h}[i])$ ▷ $\hat{h}[i] \approx \text{clamp}[\vec{h}[i] + \mathbb{DL}\text{ap}(e^{-\varepsilon}), 0, n]$
$I_1 \leftarrow \{i \in \text{SUPP}(\vec{h}) : \hat{h}[i] \ge \tau\}$ ▷ $\tau \in \Theta(\frac{\ln d}{\varepsilon})$ as defined in Equation (2)
$I_2 \leftarrow$ a uniform random subset in $\binom{[d]\setminus I_1}{n+k-|I_1|}$
$I \leftarrow I_1 \cup I_2$
for each $i \in I$ **do**
  $\tilde{h}[i] \leftarrow \mathcal{M}_{\mathbb{PA}\text{px}\mathbb{DL}\text{ap}(n,\varepsilon,\varepsilon\gamma/d)}(\vec{h}[i])$ ▷ $\tilde{h}[i] \approx \text{clamp}[\vec{h}[i] + \mathbb{DL}\text{ap}(e^{-\varepsilon}), 0, n]$
return $\tilde{h}$ ▷ where $\tilde{h}[i] = 0$ for all $i \notin I$

one could apply the purification technique (Proposition 3.1), which mixes the final output distribution of FILTER with the uniform distribution over the output space to smooth out the δ gap. However, Proposition 3.1 requires $\delta \in O(1/2^d)$, since the output of FILTER may be any subset of $[d]$. For such a small δ, we do not know how to efficiently sample X satisfying $\mathbb{TV}(X, \mathbb{B}\text{in}(d - |\text{SUPP}(\vec{h})|, q_\tau)) \le \delta$.

Rethinking Zero Entry Sampling for Privacy. The challenge above motivates a modification to how FILTER handles zero entries in $\vec{h}$, in order to avoid sampling from a complex distribution. We revisit the role that the selected zero entries play in ensuring privacy. Consider two neighboring histograms $\vec{h}$ and $\vec{h}'$. If they have the same support, then the selected zero entries have no impact on privacy. However, if their supports differ (in one or two entries under the replacement neighboring model), the sampled zero entries help obscure this difference. Informally, they serve as a *privacy blanket* over the support set. This protection can be decomposed into two parts:

(1) After adding the privacy blanket, the total number of selected entries (including both zero and nonzero entries) should have similar distributions for neighboring histograms.
(2) Conditioned on the total number being the same, the distribution over the selected sets themselves should also be similar across neighbors.

Simplifying Privacy Blanket. Our goal is to design a simpler privacy blanket that satisfies both requirements. The pseudocode for our algorithm is provided in Algorithm 2. The noise sampler $\mathcal{M}_{\mathbb{PA}\text{px}\mathbb{DL}\text{ap}(n,\varepsilon,\varepsilon\gamma/d)}$ used in Algorithm 2 is described in Theorem 1.4. This mechanism produces noise with the same privacy guarantees and nearly identical tail bounds as discrete Laplace noise, but clamped to the range $[0, n]$. For ease of exposition, *it is helpful to conceptually treat these samples as clamped discrete Laplace noise.*

For the first requirement, we leverage the fact that, under the replacement neighboring model, the total count n can be made public. Denote by I_1 the set of selected entries from $\text{SUPP}(\vec{h})$, obtained by adding noise to each entry and retaining those whose noisy counts exceed the threshold τ. We then pad I_1 to a fixed target size $n + k$, for some chosen $k \in O(n)$. This completely avoids sampling the blanket size from the distribution $\mathbb{B}\text{in}(d - |\text{SUPP}(\vec{h})|, q_\tau)$.

For the second requirement, we sample a subset of $n + k - |I_1|$ random entries from the set $[d] \setminus I_1$ (without replacement). This includes not only zero counts, but may also include non-zero ones; i.e., entries in $\text{SUPP}(\vec{h})$ can have a second chance of being selected. To see how this works, consider

one case where $\textsc{supp}(\vec{h}) \cup \{i^*\} = \textsc{supp}(\vec{h}')$ for some $i^* \in [d]$. In this case, the count $\vec{h}[i^*] = 0$ and $\vec{h}'[i^*] = 1$. Further, define I_1' for $\vec{h}'$ analogously to I_1 for $\vec{h}$. Conditioned on $i^* \notin I_1'$ (which happens in most cases), I_1 and I_1' have identical distributions. Padding them with sampled entries from the same distribution preserves the distributional identity of the final selected sets.

We only need to handle the case where $i^* \in I_1'$. In this case, I_1 receives $n+k-|I_1|$ padding entries, whereas I_1' receives $n+k-|I_1'| = n+k-|I_1|-1$ padding entries. The privacy blanket for I_1 is thus larger by one entry than that for I_1'. Conditioned on this additional entry being i^*, the remaining parts of the two privacy blankets are identically distributed. Hence, it suffices to show that the probability of i^* appearing in the privacy blanket is comparable to its probability of appearing in I_1', which we will formally establish in Section 4.2.

Further Discussion and Open Questions on Padding Strategies. Up to this point, our discussion naturally raises several questions regarding the proposed padding strategy. For instance, one might wonder what happens if we sample the $n+k-|I_1|$ padding entries *with replacement* from the set $[d] \setminus I_1$, thereby allowing the final total number of selected entries to be potentially smaller than $n+k$. Another natural question is whether the padding entries could be sampled (with or without replacement) from the larger set $[d] \setminus \textsc{supp}(\vec{h})$ instead of $[d] \setminus I_1$. Do these variations still guarantee pure differential privacy? Do they help avoid re-generating fresh noise for all selected elements in Line 8 of Algorithm 2?

While these modifications might also be rigorously analyzed by suitably adapting the proof techniques presented here, our current analysis does not directly cover them. Our primary goal in this work is to initiate the study of, and provide one workable padding strategy, rather than to offer an exhaustive treatment. We therefore leave these questions open as interesting directions for future research.

4.2 Proof of Theorem 4.1

In this subsection, we begin the formal proof of Theorem 4.1. We show that Algorithm 2 satisfies the theorem, and analyze its privacy, utility guarantees, and running time separately. All missing proofs of the lemmas in this section can be found in Appendix A.

Throughout the proof, we set

$$\tau \doteq \arg\min_{t \in \mathbb{N}} \left\{ \Pr\left[1 + \mathcal{M}_{\mathbb{PA}\text{pxD}\mathbb{L}\text{ap}\,(n,\varepsilon,\varepsilon\gamma/d)}\,(1) \geq t\right] \leq 1/d \right\}, \tag{2}$$

Based on Theorem 1.4, it holds that $\tau \in \Theta(1/\varepsilon \cdot \ln d)$. Further, denote

$$p_\tau \doteq \Pr\left[1 + \mathcal{M}_{\mathbb{PA}\text{pxD}\mathbb{L}\text{ap}\,(n,\varepsilon,\varepsilon\gamma/d)}\,(1) \geq \tau\right]. \tag{3}$$

The definition of τ implies that $p_\tau \leq 1/d$.

4.2.1 **Privacy Guarantee.** Let $\vec{h}' \sim \vec{h}$ be a neighboring histogram. To distinguish the notations, denote I_1', I_2' and I' be the items sampled by Algorithm 2 when the input is $\vec{h}'$. We want to show that,

$$e^{-\max\left(\frac{\varepsilon}{2}+2\cdot p_\tau,\varepsilon\right)} \Pr[I=S] \leq \Pr[I'=S] \leq e^{\max\left(\frac{\varepsilon}{2}+p_\tau\cdot\frac{d-k}{k+1},\varepsilon\right)} \cdot \Pr[I=S],\ \forall S \subseteq [d]. \tag{4}$$

Conditioned on $I = I'$, by the privacy guarantee of $\mathcal{M}_{\mathbb{PA}\text{pxD}\mathbb{L}\text{ap}\,(n,\varepsilon,\varepsilon\gamma/d)}$ (Theorem 1.4), the values $\{\tilde{h}[i] : i \in I\}$ and $\{\tilde{h}'[i] : i \in I'\}$ are $(\varepsilon, 0)$-indistinguishable. By basic composition theorem of DP and that $p_\tau \leq 1/d$, the Algorithm 2 is

$$\max\left(\frac{3\varepsilon}{2} + p_\tau \cdot \frac{d-k}{k+1}, 2\varepsilon\right) \leq \max\left(\frac{3\varepsilon}{2} + \frac{1}{k+1}, 2\varepsilon\right)\text{-DP}$$

It remains to prove Equation (4).

Proving Equation (4). Without loss of generality, we assume that $\vec{h}$ and $\vec{h}'$ differ in two entries indexed by $i^*, j^* \in [d]$. Further, we assume that

$$\vec{h}[i^*] + 1 = \vec{h}'[i^*], \quad \vec{h}[j^*] = \vec{h}'[j^*] + 1, \quad \vec{h}[i] = \vec{h}'[i], \forall i \in [d] \setminus \{i^*, j^*\}.$$

We consider four cases.

Case 1. $\textsc{supp}(\vec{h}) = \textsc{supp}(\vec{h}')$. Then clearly the distributions of I_1 and I_1' are $(\varepsilon, 0)$-indistinguishable, due to the privacy guarantee of $\mathcal{M}_{\mathbb{PA}\text{pxD}\mathbb{L}\text{ap}\,(n,\varepsilon,\varepsilon\gamma/d)}$. Hence so are the ones of I and I' and Equation (4) holds.

Case 2. $\textsc{supp}(\vec{h}) \cup \{i^*\} = \textsc{supp}(\vec{h}')$. In this case, we have $\vec{h}'[i^*] = 1$, $\vec{h}[i^*] = 0$, $\vec{h}[j^*] > \vec{h}'[j^*] > 0$. For convenience, denote μ as the distribution of I, i.e., $\mu(S) = \Pr[I = S]$. Similarly, let μ' be the distribution of I', with $\mu'(S) = \Pr[I' = S]$. Let $\mathcal{E}$ be the event that $i^* \in I_1'$, and let $\mu'_{\mathcal{E}}$ and $\mu'_{\bar{\mathcal{E}}}$ denote the distributions of I' conditioned on $\mathcal{E}$ and $\bar{\mathcal{E}}$, respectively. By the definition of p_τ,

$$\mu' = p_\tau \cdot \mu'_{\mathcal{E}} + (1 - p_\tau) \cdot \mu'_{\bar{\mathcal{E}}}.^4$$

First, to prove the LHS of Equation (4), we invoke Lemma 4.2.

Lemma 4.2. *μ and $\mu'_{\bar{\mathcal{E}}}$ are $(\varepsilon/2, 0)$-indistinguishable.*

Intuitively, the lemma holds because, conditioned on the event $\bar{\mathcal{E}}$ (i.e., $i^* \notin I_1'$), the distributions of I_1' and I_1 become $(\varepsilon/2, 0)$-indistinguishable: the only remaining difference between $\vec{h}$ and $\vec{h}'$ is the single-coordinate change $\vec{h}[j^*] = \vec{h}'[j^*] + 1$, and the privacy guarantee of $\mathcal{M}_{\mathbb{PA}\text{pxD}\mathbb{L}\text{ap}\,(n,\varepsilon,\varepsilon\gamma/d)}$ ensures this. Moreover, conditioned on $I_1' = I_1$, the distributions of I_2' and I_2 coincide. Consequently, $\mu'_{\bar{\mathcal{E}}}$ and μ are $(\varepsilon/2, 0)$-indistinguishable.

By Lemma 4.2,

$$\mu' \geq (1 - p_\tau) \cdot \mu'_{\bar{\mathcal{E}}} \geq (1 - p_\tau) \cdot e^{-\varepsilon/2} \cdot \mu \geq e^{-2p_\tau - \varepsilon/2} \cdot \mu,$$

where we use that $1 - p_\tau \geq e^{-2p_\tau}$ for $p_\tau \in (0, 0.5)$.

Second, to prove the RHS of Equation (4), we use Lemma 4.3.

Lemma 4.3. $\mu \geq \frac{k}{d} \cdot e^{-\varepsilon/2} \cdot \mu'_{\mathcal{E}}$.

Informally, the lemma holds because, the element i^* can be added to I only during the padding phase (Algorithm 2, Lines 5–6), which occurs with probability at least k/d. Conditioned on this event and on $\mathcal{E}$, the distributions of $I_1 \cup \{i^*\}$ and I_1' are $(\varepsilon/2, 0)$-indistinguishable: the only remaining difference between $\vec{h}$ and $\vec{h}'$ is the single-coordinate change $\vec{h}[j^*] = \vec{h}'[j^*] + 1$, and the privacy guarantee of $\mathcal{M}_{\mathbb{PA}\text{pxD}\mathbb{L}\text{ap}\,(n,\varepsilon,\varepsilon\gamma/d)}$ ensures this. Moreover, when $I_1 \cup \{i^*\} = I_1'$ and $i^* \in I_2$, the remaining padding elements in $I_2 \setminus \{i^*\}$ and I_2' have identical distributions.

Based on Lemmas 4.2 and 4.3,

$$\begin{aligned}\mu' &= p_\tau \cdot \mu'_{\mathcal{E}} + (1 - p_\tau) \cdot \mu'_{\bar{\mathcal{E}}} \leq p_\tau \cdot e^{\varepsilon/2} \cdot d/k \cdot \mu + (1 - p_\tau) \cdot e^{\varepsilon/2} \cdot \mu \\ &= e^{\varepsilon/2} \cdot ((1 - p_\tau) + p_\tau \cdot d/k) \cdot \mu = e^{\varepsilon/2} \cdot (1 + p_\tau \cdot (d/k - 1)) \cdot \mu \leq e^{\varepsilon/2 + p_\tau \cdot (d/k - 1)} \cdot \mu,\end{aligned}$$

where we use $1 + p_\tau \cdot (d/k - 1) \leq \exp(p_\tau \cdot (d/k - 1))$.

Case 3. $\textsc{supp}(\vec{h}) = \textsc{supp}(\vec{h}') \cup \{j^*\}$. The discussion follows from the symmetry of Case 2.

[4] Given a $(\mathcal{Z}, \mathcal{F})$ measurable space and two probability measures μ and ν on it, we write $\mu = \nu$ if $\mu(F) = \nu(F)$ for each $F \in \mathcal{F}$. We similarly write $\mu \geq \nu$ when $\mu(F) \geq \nu(F)$ for all $F \in \mathcal{F}$.

Case 4. $\textsc{supp}(\vec{h}) \cup \{i^*\} = \textsc{supp}(\vec{h'}) \cup \{j^*\}$. Let $\mu, \mu', \mu'_{\mathcal{E}}$ and $\mu'_{\bar{\mathcal{E}}}$ be defined as before. Further, let $\mathcal{F}$ be the event that $j^* \in I_1$, and let $\mu_{\mathcal{F}}$ and $\mu_{\bar{\mathcal{F}}}$ denote the distributions of I conditioned on $\mathcal{F}$ and $\bar{\mathcal{F}}$, respectively. By the definition of p_τ,

$$\mu = p_\tau \cdot \mu_{\mathcal{F}} + (1 - p_\tau) \cdot \mu_{\bar{\mathcal{F}}}, \quad \mu' = p_\tau \cdot \mu'_{\mathcal{E}} + (1 - p_\tau) \cdot \mu'_{\bar{\mathcal{E}}}.$$

To prove Equation (4), we need Lemma 4.4.

Lemma 4.4. $\mu_{\bar{\mathcal{F}}} = \mu'_{\bar{\mathcal{E}}}, \mu_{\bar{\mathcal{F}}} \geq \frac{k}{d} \cdot \mu'_{\mathcal{E}}, \mu'_{\bar{\mathcal{E}}} \geq \frac{k}{d} \cdot \mu_{\mathcal{F}}$.

We present an intuitive explanation of the lemma. First, conditioned on $\bar{\mathcal{F}}$ and $\bar{\mathcal{E}}$, the pair I_1 and I'_1, as well as the pair I_2 and I'_2, have identical distributions. Therefore, $\mu_{\bar{\mathcal{F}}} = \mu'_{\bar{\mathcal{E}}}$ holds. Second, the element i^* can be added to I only during the padding phase (Algorithm 2, Lines 5–6), which occurs with probability at least k/d. Conditioned on this event and on $\bar{\mathcal{F}}$ and $\mathcal{E}$, the pair I_1 and I'_1, as well as the pair I_2 and I'_2, have identical distributions. Therefore, $\mu_{\bar{\mathcal{F}}} \geq \frac{k}{d} \cdot \mu'_{\mathcal{E}}$ holds. The last inequality follows from symmetry of the second one.

Based on Lemma 4.4

$$\mu = p_\tau \cdot \mu_{\mathcal{F}} + (1 - p_\tau) \cdot \mu_{\bar{\mathcal{F}}} \leq p_\tau \cdot \tfrac{d}{k} \cdot \mu'_{\bar{\mathcal{E}}} + (1 - p_\tau) \cdot \mu'_{\bar{\mathcal{E}}} \leq \left(1 + p_\tau \cdot \left(\frac{d}{k} - 1\right)\right) \cdot \mu' \leq e^{p_\tau \cdot \frac{d-k}{k}} \mu'.$$

4.2.2 **Utility Guarantee.** Consider any $\beta > 2\varepsilon\gamma$. Set $\gamma' \doteq \varepsilon\gamma/d$ and $\beta' \doteq \beta/d$, so that $\beta' > 2\gamma'$. Recall from Theorem 1.4 that for each $t \in \mathbb{N}_{\leq n}$, $\mathcal{M}_{\mathbb{PA}\text{pxD}\mathbb{L}\text{ap}(n,\varepsilon,\gamma')}(t)$ is an (α, β')-accurate estimator of t, where

$$\alpha \in O\left(\frac{1}{\varepsilon} \ln \frac{1}{\beta'}\right) = O\left(\frac{1}{\varepsilon} \ln \frac{d}{\beta}\right).$$

We consider the accuracy of each coordinate $\tilde{h}[i]$ across three cases.

Case 1 ($i \in I$). : In this case, $\tilde{h}[i]$ is directly constructed from the output of $\mathcal{M}_{\mathbb{PA}\text{pxD}\mathbb{L}\text{ap}(n,\varepsilon,\gamma')}(\vec{h}[i])$, and is thus an (α, β')-accurate estimator of $\vec{h}[i]$.

Case 2 ($i \in [d] \setminus (\textsc{supp}(\vec{h}) \cup I)$). : Here, i is neither in the support of $\vec{h}$ nor in I, so $\tilde{h}[i] = 0 = \vec{h}[i]$, and the error is exactly zero.

Case 3 ($i \in \textsc{supp}(\vec{h}) \setminus I$). : In this case, $\hat{h}[i]$ is an (α, β')-accurate estimator of $\vec{h}[i]$. Moreover, the fact that $i \notin I$ implies that $\hat{h}[i] < \tau$. Hence,

$$|\vec{h}[i] - \tilde{h}[i]| = \vec{h}[i] \leq |\vec{h}[i] - \hat{h}[i]| + \hat{h}[i] = \alpha + \tau \in O\left(\frac{1}{\varepsilon} \cdot \ln \frac{d}{\beta}\right).$$

By a union bound, we conclude that $\tilde{h}$ is an (α, β)-simultaneous accurate estimator of $\vec{h}$.

4.2.3 **Running Time.** We need to analyze the time for the noise generation, the time for sampling the privacy blanket, and the time of the dictionary operations, separately.

Noise Generation. Under the assumption in Theorem 4.1 that the descriptions of n, ε, and γ fit in a constant number of machine words (i.e., $\ln n, \ln \frac{1}{\varepsilon}, \ln \frac{1}{\gamma} \in O(\omega)$, where ω is the word size), Theorem 1.4 implies that the noise sampler $\mathcal{M}_{\mathbb{PA}\text{pxD}\mathbb{L}\text{ap}(n,\varepsilon,\varepsilon\gamma/d)}$ has initialization time $\tilde{O}(1)$ and uses $O\left(\frac{1}{\varepsilon} + \ln \frac{d}{\varepsilon\gamma} + \ln n\right)$ words of memory.

After initialization, each sample from the noise distribution can be generated in $O(1)$ time. Thus, producing $n + k$ samples takes total time $O(n + k)$.

Privacy Blanket Sampling. To sample a uniform random subset from $\binom{[d]\setminus I_1}{n+k-|I_1|}$ (the set of all subsets of size $n+k-|I_1|$ from $[d] \setminus I_1$), we first sample a uniform random subset of size $n+k$ from $[d]$, and then randomly select $n+k-|I_1|$ elements from this subset that do not appear in I_1.

To generate a uniform random subset of size $n+k$ from $[d]$, we employ the following efficient method: sample $m = 4(n+k) \ll d$ elements independently and uniformly from $[d]$, and remove duplicates. If the resulting set contains at least $n+k$ distinct elements, we return the first $n+k$ of them. Otherwise, there are two options:

- *Abort and output a fixed sparse histogram.* Terminate Algorithm 2 early and return a fixed sparse histogram, e.g., one where elements 1 through n each have count 1. Although this may slightly degrade utility, it does not compromise privacy, since sampling m independent elements from $[d]$ is independent of the input dataset. We show that this increases the failure probability of the utility guarantee by at most an additive $\sqrt{n+k} \cdot \exp(-(n+k)/16)$.
- *Repeat the sampling procedure until success.* This approach maintains both correctness and privacy, as the repetition process reveals nothing about the underlying histogram $\vec{h}$, nor about the size or contents of the privacy blanket. However, it deviates from our goal of a deterministic linear-time algorithm, yielding instead only expected linear-time performance.

To bound the failure probability of this sampling procedure, we apply the standard balls-and-bins model [39]: we throw m balls into d bins and analyze the number of non-empty bins.

Let X_i denote the number of balls in bin i, for each $i \in [d]$, so that $\sum_{i\in[d]} X_i = m$. Let $S_X = \sum_{i\in[d]} \mathbb{1}_{[X_i>0]}$ be the number of non-empty bins. We aim to upper bound the probability $\Pr[S_X \le n+k]$.

To analyze this, we apply the Poisson approximation method [39]. Let $Y_1, \ldots, Y_d$ be independent Poisson random variables with mean m/d, i.e., $\Pr[Y_i = t] = e^{-m/d} \cdot \frac{(m/d)^t}{t!}, \quad \forall t \in \mathbb{N}$. Define $S_Y = \sum_{i\in[d]} \mathbb{1}_{[Y_i>0]}$ analogously. The Poisson approximation theorem implies that

$$\Pr[S_X \le n+k] \le e\sqrt{m} \cdot \Pr[S_Y \le n+k].$$

Thus, it suffices to bound $\Pr[S_Y \le n+k]$. Since $\Pr[Y_i > 0] = 1 - e^{-m/d}$, we have $\mathbb{E}[S_Y] = d \cdot (1 - e^{-m/d}) \ge d \cdot \frac{m}{2d} = m/2$, where the inequality uses $1 - e^{-x} \ge x/2$ for $0 \le x \le 1$, applicable whenever $m \le d$.

Via the Poisson Chernoff bound,

$$\Pr[S_Y \le n+k] \le \Pr\left[S_Y \le \frac{1}{2}\mathbb{E}[S_Y]\right] \le e^{-\frac{1}{8}\cdot\mathbb{E}[S_Y]} \le e^{-\frac{m}{16}}.$$

Dictionary Operations. Implementing Algorithm 2 requires standard dictionary operations— inserting a key–value pair, deleting a pair, updating the value associated with a key, and checking whether a key is present. These operations arise throughout the algorithm: during the histogram construction step (Line 1), during the blanket sampling and padding steps (Lines 5 and 6), and when reading or updating values stored in $\vec{h}$, $\hat{h}$, and $\tilde{h}$.

To support these operations, we use the hashing-based dictionary (Backyard Cuckoo Hashing) of Arbitman et al. [2]. Deleting a pair, updating a value, and checking whether a key is present all run in deterministic constant time per operation. Insertions require more care, as they may not always complete in constant time.

The key observation is that in Algorithm 2, there are $O(n+k)$ intersections in total. By Arbitman et al. [2], for any constant c, the dictionary performs all these intersections in total time $O(n+k)$, with failure probability at most $(n+k)^{-c}$.

A natural approach is to enforce a deterministic time bound $T \in O(n+k)$, so that the combined dictionary operations exceed this bound with probability at most $(n+k)^{-c}$. However, this creates a

subtle issue for differential privacy: two neighboring datasets $\mathcal{X}$ and $\mathcal{X}'$ may have different failure probabilities, potentially revealing information about their difference. A careful strategy is therefore required to control the impact of these failures on the overall privacy guarantee.

To handle potential failures, we adopt the following padding procedure, ensuring that neighboring datasets have similar failure probabilities with the hash dictionary:

- Flip a coin with probability $\frac{1}{e^\varepsilon - 1} \cdot (n+k)^{-c}$. If heads, output a uniform random histogram.
- Otherwise, run Algorithm 2. If all dictionary operations combined exceed time T, halt Algorithm 2 and output a uniform random histogram $\tilde{h}$.

It remains to analyze its impact on the privacy guarantee. The output distribution of Algorithm 2 ($\mathcal{M}_{\text{HIST}}$) on input dataset $\mathcal{X}$, denoted $\bar{\mu}$, can be decomposed into two components: the distribution conditioned on the dictionary operations completing within the enforced time bound, denoted μ, and a uniform distribution over all possible outputs, denoted ν. There exist weights $w_1, w_2 \in [0,1]$ such that

$$(1)\ \bar{\mu} = w_1\mu + w_2\nu,\ (2)\ w_1 + w_2 = 1,\ \text{and}\ (3)\ \frac{1}{e^\varepsilon - 1} \cdot (n+k)^{-c} \le w_2 \le \frac{e^\varepsilon}{e^\varepsilon - 1} \cdot (n+k)^{-c}.$$

Similarly, for a neighboring dataset $\mathcal{X}'$, let μ' be the output distribution of $\mathcal{M}_{\text{HIST}}$ when the dictionary operations do not exceed the time bound. There exist weights $w_1', w_2' \in [0,1]$ such that

$$(1)\bar{\mu}' = w_1'\mu' + w_2'\nu,\ (2)w_1' + w_2' = 1,\ \text{and}\ (3)\ \frac{1}{e^\varepsilon - 1} \cdot (n+k)^{-c} \le w_2' \le \frac{e^\varepsilon}{e^\varepsilon - 1} \cdot (n+k)^{-c}.$$

Therefore, $\frac{w_2}{w_2'} \le e^\varepsilon$, and $\frac{w_1}{w_1'} \le \frac{1}{1 - \frac{e^\varepsilon}{e^\varepsilon - 1} \cdot (n+k)^{-c}}$. Further, based on the privacy analysis, μ and μ' are ε-indistinguishable. For each measurable set E in the range of $\mathcal{M}_{\text{HIST}}$, it holds that

$$\frac{\bar{\mu}(E)}{\bar{\mu}'(E)} = \frac{w_1\,\mu(E) + w_2\,\nu(E)}{w_1'\,\mu'(E) + w_2'\,\nu(E)} \le \max\left\{\frac{w_1\,\mu(E)}{w_1'\,\mu'(E)}, \frac{w_2\,\nu(E)}{w_2'\,\nu(E)}\right\} \le \frac{e^\varepsilon}{1 - \frac{e^\varepsilon}{e^\varepsilon - 1} \cdot (n+k)^{-c}} \le e^{\varepsilon + \frac{2}{n^2}}.$$

where the last inequality holds since, for a sufficiently large constant c, the term $\frac{e^\varepsilon}{e^\varepsilon - 1} \cdot (n+k)^{-c} \le \frac{1}{n^2}$, so that $1 - \frac{e^\varepsilon}{e^\varepsilon - 1} \cdot (n+k)^{-c} \ge e^{-2/n^2}$. Finally, under the assumption that $1/\varepsilon \le n$, we have $e^{\varepsilon + \frac{2}{n^2}} \le e^{\varepsilon + \frac{2\varepsilon}{n}}$.

5 Related Work

In this section, we review related work. We begin with possible side-channel attacks and defenses, then discuss noise sampling algorithms for differential privacy, and finally cover differentially private frequency estimation algorithms.

5.1 Side-Channel Attacks

While differentially private mechanisms offer strong theoretical guarantees, practical implementations can introduce side channels—such as floating-point rounding artifacts and timing variability—that can be exploited to compromise privacy.

Floating Point Attack. Mironov [38] was the first to expose the vulnerability of implementing the Laplace mechanism using double-precision floating-point numbers. He observed that certain floating-point values cannot be generated due to the finite precision and rounding effects inherent in floating-point arithmetic, which can make it possible to distinguish between neighboring inputs and thereby break the guarantees of differential privacy. Jin et al. [29] later showed that similar attacks are also possible against the Gaussian mechanism. One solution to these vulnerabilities is to adopt discrete versions of the Laplace and Gaussian mechanisms [12, 25], which avoid the pitfalls of floating-point arithmetic.

Timing Attack. Jin et al. [29] further demonstrated that even discrete mechanisms designed to avoid floating-point issues—such as those by Canonne et al. [12] and Google [27]—can still leak information through timing side channels. These mechanisms typically employ *geometric sampling*, which when directly simulated, repeatedly performs biased coin tosses until the first "head" occurs. This introduces a positive correlation between the sampled value and the algorithm's running time, potentially leaking information.

Time-Oblivious Sampling. Dov, David, Naor, and Tzalik [17] systematically study noise sampling algorithms resilient to timing-attack, and their implications for DP mechanisms. An algorithm is defined as time-oblivious if its output distribution and running time distribution are independent. Their key findings include: 1) a discrete distribution admits a time-oblivious sampling if and only if it has a finite support and a rational probability mass function, for which they provide sampling algorithms with optimal number of unbiased random bits; and 2) such a distribution admits a worst-case time complexity sampling algorithm if the least common multiple of the denominators of its rational probabilities is a power of 2.

As a time-oblivious sampling algorithm can require significantly more random bits than the classical Knuth-Yao sampler [33], Dov et al. [17] introduce a relaxed notion of (ε, δ) time-oblivious sampler. They design both ε-pure and $(0, \delta)$-approximate time-oblivious samplers that use logarithmic number of random bits but exponential space.

DP Time-Oblivious Algorithm. Extending their notion of (ε, δ) time-oblivious sampler, Dov et al. [17] defines DP time-oblivious mechanisms. They show that any pure DP time-oblivious mechanism over infinitely many datasets gives an irrelevant output with some constant probability. Nevertheless, they demonstrate that a pure DP mechanism can be transformed into a time-oblivious with nearly the same privacy and utility guarantees, although the efficiency of this transformation remains an open problem due to its reliance on manipulating countably many probability values.

A similar definition was proposed by Ratliff and Vadhan [41] under the name (ε, δ)-*Joint Output/-Timing Privacy*. Their formulation is more comprehensive, as it explicitly models the computational environment as part of the algorithm's input, thereby capturing its influence on execution time. They also introduce a general framework that composes timing-stable programs with randomized delays to enforce timing privacy. However, this framework guarantees only approximate timing differential privacy.

In follow-up work, Ratliff and Vadhan [42] present a new framework for converting pure, time-oblivious differentially private algorithms from the *upper-bounded setting* (see the discussion following Definition 2.2) to the *unbounded setting*, under the add/remove neighboring relation and in the RAM model.

5.2 Noise Generation

This subsection reviews methods for generating random noise used in differentially private algorithms, with a focus on Bernoulli, discrete Laplace, and discrete Gaussian distributions. Many of these methods were developed in the MPC setting, and when simulated in the central model, they can prevent timing attacks.

Dwork et al. [20] presented a distributed protocol for generating shares of approximate Gaussian or approximate geometric random variables, secure against malicious participants. In particular, they proved that each bit in the binary representation of a geometric random variable (over an interval whose length is a power of 2) can be generated independently according to distinct Bernoulli distributions, and they provided closed-form formulas for these distributions.

Thus, the problem of efficiently generating geometric random variables reduces to that of generating Bernoulli random variables. Dwork et al. [20] further studied efficient protocols for

generating Bernoulli random variables $\mathbb{B}\text{ernoulli}(p)$ for some $p \in (0,1)$. They first observed that such a random variable can be generated using at most 2 unbiased random bits in expectation. Second, they presented deterministic-time protocols that, with high probability, generate a batch of m biased bits whose statistical distance to $\mathbb{B}\text{ernoulli}(p)$ is at most $2^{-\ell}$. Their protocols are described as circuits:

(1) The first has depth $\Theta\big(\ln^2(m \cdot \ell)\big)$, gate count $\Theta(m \cdot \ell \cdot (\ell + \ln m) \cdot \ln m)$, and input size $\Theta(m)$.
(2) The second has depth $\Theta(\ln m)$, gate count $\Theta\big(m^2 \cdot \ell\big)$, and input size $\Theta(m)$.
(3) The third has depth $\Theta(\ln(m+\ell))$, gate count $\Theta(m\ell \cdot \ln(m+\ell))$, and input size $\Theta(m \ln(m+\ell))$.

When directly simulating these circuits on a central server, they require $\Omega(m \cdot \ell \cdot \ln(m+\ell))$ running time.

Champion et al. [13] design an MPC protocol that, given $p \in (0,1)$, samples m random variables whose total variation distance to m i.i.d. Bernoulli random variables $\mathbb{B}\text{ernoulli}(p)$ is at most ℓ. The protocol achieves this with an amortized communication and computation cost of $O(\ln(\ell + \ln m))$. A key ingredient in their approach is the circuit construction for stacks proposed by Zahur and Evans [46].

Canonne et al. [12] show how to generate random variables that exactly follow discrete Gaussian or discrete Laplacian distributions under the word-RAM model using rejection sampling. Their paper also includes a subroutine for sampling an exact Bernoulli random variable $\mathbb{B}\text{ernoulli}(\exp(-\gamma))$ for some positive rational number $\gamma \in \mathbb{Q}_+$. Their algorithms require an unlimited amount of memory, run in $O(1)$ expected time, but could be vulnerable to potential timing attacks [29].

Knott et al. [32] present a software framework for secure MPC primitives in machine learning, which includes sampling algorithms for the Bernoulli, continuous Laplace, and Gaussian distributions. They represent floating-point values using fixed-point encoding with a length of L bits. The framework also provides algorithms for evaluating complex functions such as e^x, $\sin x$, and $1/x$. However, the authors do not explicitly discuss the running time or the number of bits required for initialization and intermediate computations to achieve a specified final precision when evaluating these functions via their algorithms.

Wei et al. [44] provide MPC protocols that realize the Bernoulli sampling and approximate geometric random sampling algorithms from Dwork et al. [20]. Using these building blocks, they construct an MPC protocol for generating approximate discrete Laplace variables, which they further employ to develop an MPC protocol that approximates the discrete Gaussian sampling algorithm of Canonne et al. [12].

Keller et al. [30] propose MPC protocols that approximate the discrete Laplace and discrete Gaussian sampling algorithms of Canonne et al. [12] in a more direct and streamlined manner.

Franzese et al. [23] present an MPC protocol for noise generation over a finite domain, based on the table lookup method. Given a probability distribution μ over a finite domain, their method constructs $O(\ell)$ tables, each of size $O(|\mathrm{C}_{2^{-\ell}}|)$, where $\mathrm{C}_{2^{-\ell}}$ denote a subset of minimum size such that $\mu(\mathrm{C}_{2^{-\ell}}) \geq 1 - 2^{-\ell}$. These tables are then used to sample a random variable whose statistical distance to μ is at most $O\big(2^{-\ell}\big)$, with a running time proportional to the number of tables accessed during sampling.

5.3 Differentially Private Frequency Estimation

This subsection reviews differentially private algorithms for estimating item frequencies, including frequency oracles and private histograms. A private histogram explicitly releases a noisy version of the entire frequency vector, while a frequency oracle is a data structure that allows on-demand, query-based access to noisy frequency estimates. Every private histogram implicitly defines a frequency oracle.

Additional Error Measure. To motivate frequency oracles, we first introduce another commonly considered error measure, the per-query accuracy guarantee.

Definition 5.1 ((α,β)-Accurate Estimator). *Let $\alpha \in \mathbb{R}_+$ and $\beta \in [0,1]$. A random variable X is an (α,β)-accurate estimator of $t \in \mathbb{R}$ if* $\Pr[|X - t| \geq \alpha] \leq \beta$.

A histogram $\tilde{h}$ achieves (α,β) per-query accuracy if, for each $i \in [d]$, $\tilde{h}[i]$ is an (α,β)-accurate estimator of $\vec{h}[i]$. In the sparse histogram setting, these two notions coincide. Balcer and Vadhan [3] show that, for any $\beta \in (0, 1/2]$, an ε-DP histogram $\tilde{h}$ with $\|\tilde{h}\|_0 \leq n'$ cannot achieve (α,β) per-query accuracy for $\alpha \in \Omega\big(\min\{\frac{1}{\varepsilon}\ln\frac{d}{n'\cdot\beta}, n\}\big)$, which matches the simultaneous accuracy upper bounds obtained by our algorithm in Section 4.

There exist other private succinct representations of $\vec{h}$ beyond sparse histograms, known as *frequency oracles*, which can bypass the per-query accuracy lower bounds discussed below.

Frequency Oracle. Let $\vec{h} \in [0\,..\,n]^d$ be a histogram with at most n non-zero entries. A *frequency oracle* for $\vec{h}$ is a data structure that, given $i \in [d]$, returns an estimate of $\vec{h}[i]$.

Balcer and Vadhan [3] propose an ε-DP frequency oracle using $O(n \cdot \ln d)$ bits of space. It provides expected per-query error $O(1/\varepsilon)$, query time $\tilde{O}(n/\varepsilon)$, and expected simultaneous error $O((1/\varepsilon)\cdot\ln d)$ over all $i \in [d]$.

Lebeda et al. [36] give an (ε,δ)-DP frequency oracle using $O(n \cdot \ln(d+n))$ bits of space. For $\delta = 0$, it achieves expected per-query error $O(1/\varepsilon)$, query time $O(\ln d)$, and expected simultaneous error $O((1/\varepsilon)\cdot\ln d)$. For $\delta > 0$, the per-query error remains $O(1/\varepsilon)$, the query time becomes $O(\ln(1/\delta))$, and the simultaneous error is $O((1/\varepsilon)\cdot\ln(1/\delta))$.

Lolck and Pagh [37] further reduce the space to $O(n \cdot \ln n)$ bits and the query time to $O(\ln\ln d)$ in the $\delta = 0$ setting, using techniques from error-correcting codes.

Private Histogram. While frequency oracles offer space-efficient point-wise estimates, they are insufficient for solving the commonly studied variant of the heavy hitters problem [10, 45]: privately identifying all $i \in [d]$ such that $\vec{h}[i] \geq \Delta$, for the smallest possible threshold Δ. Addressing this task with a frequency oracle directly requires querying all d domain elements, incurring an unacceptable $O(d)$ query overhead.

To bypass this bottleneck, we focus on an alternative data structure. A *private histogram* explicitly releases a noisy version of the entire frequency vector $\vec{h}$, allowing for efficient global operations like heavy hitter identification without sweeping the domain.

Dwork et al. [21] introduced the foundational ε-DP Laplace mechanism for this task: it publishes a privatized histogram $\tilde{h}$ of $\vec{h}$ by adding continuous Laplacian noise $\mathbb{L}\mathrm{ap}(\exp(-\varepsilon/2))$ to each entry independently, where $\mathbb{L}\mathrm{ap}(\exp(-\varepsilon/2))$ is supported over $\mathbb{R}$ with density function $p(x) \propto \exp(-\varepsilon\cdot|x|/2), \forall x \in \mathbb{R}$. The expected error is $O(1/\varepsilon)$ for a single entry in $\tilde{h}$, and the expected maximum error is $O(1/\varepsilon \cdot \ln d)$ over all entries. It is known that these errors are asymptotically optimal [6, 28]. However, this continuous mechanism still has a running time of $O(d)$ and assumes real arithmetic.

To avoid real arithmetic, Ghosh et al. [25] proposed replacing the continuous Laplace noise $\mathbb{L}\mathrm{ap}(\exp(-\varepsilon/2))$ with a discrete variant, denoted $\mathbb{DL}\mathrm{ap}(\exp(-\varepsilon/2))$. This distribution, referred to as the two-sided geometric distribution in the original paper, has probability mass function $P(z) \propto \exp(-\varepsilon\cdot|z|/2)$ for all $z \in \mathbb{Z}$. The discrete variant preserves the optimal asymptotic error bounds of the Laplace mechanism. However, sampling from $\mathbb{DL}\mathrm{ap}(\exp(-\varepsilon/2))$ requires unbounded memory access and has running time bounded only in expectation, exposing it to potential timing attacks.

To avoid the $O(d)$ running time, the *stability-based histogram algorithm*, first proposed by Korolova et al. [34] and later presented by Bun et al. [11], adds $\mathbb{DL}\text{ap}\,(\exp(-\varepsilon/2))$ noise only to the nonzero entries of $\vec{h}$ and releases only those entries whose noisy counts exceed a threshold $\tau \in \Theta(\frac{1}{\varepsilon} \cdot \ln \frac{1}{\delta})$. This method satisfies (ε, δ)-differential privacy, achieves a strict running time of $\tilde{O}(n)$, and incurs an expected error of $O(\frac{1}{\varepsilon})$ for each entry whose true count is at least $\Omega\big(\frac{1}{\varepsilon} \cdot \ln \frac{1}{\delta}\big)$. The expected maximum error over all entries is $O\big(\frac{1}{\varepsilon} \cdot \ln \frac{1}{\delta}\big)$.

Cormode et al. [15] present an ε-DP variant of the stability-based histogram that adds $\mathbb{DL}\text{ap}\left(e^{-\varepsilon/2}\right)$ to all entries of $\vec{h}$ and then releases only those whose noisy counts exceed a threshold $\tau \in \Theta\big(\frac{1}{\varepsilon} \cdot \ln d\big)$. This variant achieves an expected error of $O\big(\frac{1}{\varepsilon}\big)$ for each entry whose true count is at least $\Omega\big(\frac{1}{\varepsilon} \cdot \ln d\big)$, and the expected maximum error over all entries is $O\big(\frac{1}{\varepsilon} \cdot \ln d\big)$. However, a naive implementation of this method still requires $O(d)$ running time. The key observation is that, for entries whose true count is zero, the probability that their noisy count exceeds the threshold τ is $p_\tau \in \tilde{O}\big(\frac{1}{d}\big)$. Therefore, the number of such false positive entries follows the binomial distribution $\mathbb{B}\text{in}(d - \|\vec{h}\|_0, p_\tau)$, which has an expected value of $\tilde{O}(1)$, where $\|\vec{h}\|_0$ is the number of nonzero entries in $\vec{h}$. Recently, Qiu and Yi [40] presented a variant of the algorithm by Cormode et al. [15] and explicitly discussed how to sample $\mathbb{B}\text{in}(d - \|\vec{h}\|_0, p_\tau)$ in the Real-RAM model in $O(1)$ expected time. However, it remains open whether one can sample from $\mathbb{B}\text{in}(d - \|\vec{h}\|_0, p_\tau)$ in strict $\tilde{O}(n)$ time in the Word-RAM model.

To overcome the aforementioned issues, Balcer and Vadhan [3] proposed releasing the n largest noisy counts, which ensures that the output size is always bounded by n. Their approach achieves the same asymptotic error as that of Cormode et al. [15]. Instead of adding noise to all entries in $\vec{h}$ and then reporting the top n, they design a more sophisticated algorithm that samples these noisy counts directly, achieving a strict running time of $\tilde{O}(n^2)$.

Lebeda and Tetek [35] study the problem of constructing private sparse histograms in the streaming setting. They propose an (ε, δ)-DP algorithm that operates in strict time and outputs a sparse histogram using $2 \cdot k$ words of space. The histogram includes all elements whose frequencies are at least $n/(1+k) + \Omega(1/\varepsilon \cdot \ln(1/\delta))$, along with their frequency estimates. By treating the frequencies of elements not included in the histogram as zero, the expected maximum frequency estimation error is bounded by $n/(1+k) + O(1/\varepsilon \cdot \ln(1/\delta))$. Their algorithm also has an ε-DP variant achieving expected maximum estimation error $n/(1+k) + O(1/\varepsilon \cdot \ln d)$, which reduces to $O(1/\varepsilon \cdot \ln d)$ when $k = n$. However, their approach relies on existing techniques for releasing the top-n noisy counts, whereas the best known algorithm with strict time and bounded memory for this problem has time complexity $\tilde{O}\big(n^2\big)$ [3].

Acknowledgments

The authors would like to thank Jonathan Ullman for insightful discussions that inspired a simpler intuition for the padding technique. We also thank Christian Janos Lebeda and Rasmus Pagh for helpful conversations regarding the running time of the hash tables used in this work, and the anonymous reviewers for their valuable feedback.

A Missing Proofs for Section 4

A.1 Privacy Guarantee

Proof of Lemma 4.2. Recall that in *Case 2*, we have $\text{SUPP}(\vec{h}) \cup \{i^*\} = \text{SUPP}(\vec{h}')$, and therefore

$$\vec{h}'[i^*] = 1,\ \vec{h}[i^*] = 0,\ \vec{h}[j^*] > \vec{h}'[j^*] > 0.$$

By definition of μ and μ', for each S,

$$\mu(S) = \Pr[I = S] = \sum_{J \subseteq \text{SUPP}(\vec{h})} \Pr[I = S \mid I_1 = J] \cdot \Pr[I_1 = J], \tag{5}$$

$$\mu'(S) = \Pr[I' = S] = \sum_{J' \subseteq \text{SUPP}(\vec{h}')} \Pr\left[I' = S \mid I'_1 = J'\right] \cdot \Pr\left[I'_1 = J'\right]. \tag{6}$$

We can partition the subsets in $\text{SUPP}(\vec{h}')$ into the ones which contain i^* and those which do not as follows:

$$\{J' : J' \in \text{SUPP}(\vec{h}')\} = \{J : J \in \text{SUPP}(\vec{h})\} \cup \{J \cup \{i^*\} : J \in \text{SUPP}(\vec{h})\}.$$

Therefore

$$\Pr[I' = S] = \sum_{J \subseteq \text{SUPP}(\vec{h})} \Big(\Pr\left[I' = S \mid I'_1 = J\right] \Pr\left[I'_1 = J\right] + \Pr\left[I' = S \mid I'_1 = J \cup \{i^*\}\right] \Pr\left[I'_1 = J \cup \{i^*\}\right] \Big).$$

Based on the definition of $\mu'_{\vec{\mathcal{E}}}$, we have

$$\mu'_{\vec{\mathcal{E}}}(S) = \sum_{J \subseteq \text{SUPP}(\vec{h})} \Pr\left[I' = S \mid I'_1 = J\right] \cdot \Pr\left[I'_1 = J \mid i^* \notin I'_1\right] \tag{7}$$

Comparing it with Equation (5), it follows directly from the construction of Algorithm 2 that

$$\Pr\left[I' = S \mid I'_1 = J\right] = \Pr[I = S \mid I_1 = J].$$

To compare $\Pr[I_1 = J]$ and $\Pr\left[I'_1 = J \mid i^* \notin I'_1\right]$, we need the following lemma.

Lemma A.1. *For each $i \in [d]$, let Z_i (or Z'_i) be the indicator for $i \in I_1$ (or $i \in I'_1$). For each $J \subseteq \text{SUPP}(\vec{h})$, define $r_J \doteq \Pr\left[Z_{j^*} = \mathbb{1}_{[j^* \in J]}\right] / \Pr\left[Z'_{j^*} = \mathbb{1}_{[j^* \in J]}\right]$. Then*

$$\Pr\left[I'_1 = J\right] = (1 - p_\tau) \cdot r_J \cdot \Pr[I_1 = J], \qquad \Pr\left[I'_1 = J \cup \{i^*\}\right] = p_\tau \cdot r_J \cdot \Pr[I_1 = J].$$

Based on the privacy guarantee of the noise sampler in Algorithm 2 (Theorem 1.4), it holds that $r_J \in (e^{-\varepsilon/2}, e^{\varepsilon/2})$. Therefore, for each $J \subseteq \text{SUPP}(\vec{h})$,

$$\begin{aligned}\Pr\left[I'_1 = J \mid i^* \notin I'_1\right] &= \frac{\Pr\left[I'_1 = J, i^* \notin I'_1\right]}{\Pr\left[i^* \notin I'_1\right]} = \frac{\Pr\left[I'_1 = J\right]}{1 - p_\tau} \\ &= r_J \cdot \Pr[I_1 = J] \in (e^{-\varepsilon/2}, e^{\varepsilon/2}) \cdot \Pr[I_1 = J],\end{aligned}$$

where by definition of p_τ, $\Pr\left[i^* \notin I'_1\right] = 1 - p_\tau$ and $\Pr\left[I'_1 = J, i^* \notin I'_1\right] = \Pr\left[I'_1 = J\right]$ since for each $J \subseteq \text{SUPP}(\vec{h})$, $i^* \notin J$. It concludes that

$$\mu'_{\vec{\mathcal{E}}}(S) \in (e^{-\varepsilon/2}, e^{\varepsilon/2}) \cdot \mu(S). \tag{8}$$

□

Proof of Lemma 4.3. Following the same argument as in the proof of Lemma 4.2, for each S we have

$$\mu'_{\mathcal{E}}(S) = \sum_{J \subseteq \text{SUPP}(\vec{h})} \Pr\left[I' = S \mid I'_1 = J \cup \{i^*\}\right] \cdot \Pr\left[I'_1 = J \cup \{i^*\} \mid i^* \in I'_1\right]. \tag{9}$$

We compare this with

$$\mu(S) = \Pr[I = S] = \sum_{J \subseteq \text{SUPP}(\vec{h})} \Pr[I = S \mid I_1 = J] \cdot \Pr[I_1 = J].$$

Step 1: Comparing the marginal probabilities. By Lemma A.1,

$$\begin{aligned}\Pr\left[I'_1 = J \cup \{i^*\} \mid i^* \in I'_1\right] &= \frac{\Pr\left[I'_1 = J \cup \{i^*\},\, i^* \in I'_1\right]}{\Pr\left[i^* \in I'_1\right]} = \frac{\Pr\left[I'_1 = J \cup \{i^*\}\right]}{p_\tau} \\ &= r_J \cdot \Pr[I_1 = J] \in (e^{-\varepsilon/2}, e^{\varepsilon/2}) \cdot \Pr[I_1 = J],\end{aligned}$$

where $\Pr\left[i^* \in I'_1\right] = p_\tau$ by definition.

Step 2: Comparing the conditional probabilities. We have the following lemma.

Lemma A.2. *Let $\kappa \doteq \frac{d}{k}$. For each $J \subseteq \text{SUPP}(\vec{h})$,*

$$\Pr\left[I' = S \mid I'_1 = J \cup \{i^*\}\right] \le \kappa \cdot \Pr[I = S \mid I_1 = J]. \tag{10}$$

Applying the lemma and combining both steps,

$$\begin{aligned}\mu(S) &\ge \sum_{J \subseteq \text{SUPP}(\vec{h})} \frac{1}{\kappa} \cdot \Pr\left[I' = S \mid I'_1 = J \cup \{i^*\}\right] \cdot e^{-\varepsilon/2} \cdot \Pr\left[I'_1 = J \cup \{i^*\} \mid i^* \in I'_1\right] &(11)\\ &\ge \frac{1}{\kappa} \cdot e^{-\varepsilon/2} \cdot \mu'_{\mathcal{E}}(S). &(12)\end{aligned}$$

□

Proof of Lemma 4.4. Recall in this case, $\text{SUPP}(\vec{h}) \cup \{i^*\} = \text{SUPP}(\vec{h}') \cup \{j^*\}$. First, observe that the subsets in $\text{SUPP}(\vec{h})$ can be partitioned into those containing j^* and those that do not. For each $J \subseteq \text{SUPP}(\vec{h})$ with $j^* \notin J$, we have $J \subseteq \text{SUPP}(\vec{h}) \cap \text{SUPP}(\vec{h}')$. Hence,

$$\begin{aligned}\mu(S) = \Pr[I = S] = &\sum_{J \subseteq \text{SUPP}(\vec{h}) \cap \text{SUPP}(\vec{h}')} \Pr[I = S \mid I_1 = J] \cdot \Pr[I_1 = J] \\ &+ \sum_{J \subseteq \text{SUPP}(\vec{h}) \cap \text{SUPP}(\vec{h}')} \Pr[I = S \mid I_1 = J \cup \{j^*\}] \cdot \Pr[I_1 = J \cup \{j^*\}].\end{aligned} \tag{13}$$

Similarly,

$$\begin{aligned}\mu'(S) = \Pr[I' = S] = &\sum_{J \subseteq \text{SUPP}(\vec{h}) \cap \text{SUPP}(\vec{h}')} \Pr\left[I' = S \mid I'_1 = J\right] \cdot \Pr\left[I'_1 = J\right] \\ &+ \sum_{J \subseteq \text{SUPP}(\vec{h}) \cap \text{SUPP}(\vec{h}')} \Pr\left[I' = S \mid I'_1 = J \cup \{i^*\}\right] \cdot \Pr\left[I'_1 = J \cup \{i^*\}\right].\end{aligned} \tag{14}$$

Further, by the definition of $\mu_{\tilde{\mathcal{F}}}$, $\mu'_{\tilde{\mathcal{E}}}$, $\mu'_{\mathcal{E}}$, $\mu_{\mathcal{F}}$, it holds that for each S

$$\mu_{\tilde{\mathcal{F}}}(S) = \sum_{J \subseteq \text{SUPP}(\vec{h}) \cap \text{SUPP}(\vec{h}')} \Pr\left[I = S \mid I_1 = J\right] \cdot \Pr\left[I_1 = J \mid j^* \notin I_1\right]. \tag{15}$$

$$\mu_{\mathcal{F}}(S) = \sum_{J \subseteq \text{SUPP}(\vec{h}) \cap \text{SUPP}(\vec{h}')} \Pr\left[I = S \mid I_1 = J \cup \{j^*\}\right] \cdot \Pr\left[I_1 = J \cup \{j^*\} \mid j^* \in I_1\right]. \tag{16}$$

$$\mu'_{\tilde{\mathcal{E}}}(S) = \sum_{J \subseteq \text{SUPP}(\vec{h}) \cap \text{SUPP}(\vec{h}')} \Pr\left[I' = S \mid I'_1 = J\right] \cdot \Pr\left[I'_1 = J \mid i^* \notin I'_1\right]. \tag{17}$$

$$\mu'_{\mathcal{E}}(S) = \sum_{J \subseteq \text{SUPP}(\vec{h}) \cap \text{SUPP}(\vec{h}')} \Pr\left[I' = S \mid I'_1 = J \cup \{i^*\}\right] \cdot \Pr\left[I'_1 = J \cup \{i^*\} \mid i^* \in I'_1\right]. \tag{18}$$

Comparison of $\mu_{\tilde{\mathcal{F}}}$ and $\mu'_{\tilde{\mathcal{E}}}$. Following directly from the construction of Algorithm 2, we have

$$\Pr\left[I' = S \mid I'_1 = J\right] = \Pr\left[I = S \mid I_1 = J\right].$$

Let the Z_i (Z'_i) be defined as in Lemma A.1. For each $J \subseteq \text{SUPP}(\vec{h}) \cap \text{SUPP}(\vec{h}')$, it holds that

$$\begin{aligned}\Pr\left[I'_1 = J \mid i^* \notin I'_1\right] &= \prod_{i \in \text{SUPP}(\vec{h}') \setminus \{i^*\}} \Pr\left[Z'_i = \mathbb{1}_{[i \in J]}\right] \\ &= \prod_{i \in \text{SUPP}(\vec{h}) \setminus \{j^*\}} \Pr\left[Z_i = \mathbb{1}_{[i \in J]}\right] = \Pr\left[I_1 = J \mid j^* \notin I_1\right].\end{aligned}$$

Next,

$$\begin{aligned}\Pr\left[I'_1 = J \cup \{i^*\} \mid i^* \in I'_1\right] &= \prod_{i \in \text{SUPP}(\vec{h}') \setminus \{i^*\}} \Pr\left[Z'_i = \mathbb{1}_{[i \in J]}\right] \\ &= \prod_{i \in \text{SUPP}(\vec{h}) \setminus \{j^*\}} \Pr\left[Z_i = \mathbb{1}_{[i \in J]}\right] = \Pr\left[I_1 = J \cup \{j^*\} \mid j^* \in I_1\right].\end{aligned}$$

Therefore,

$$\begin{aligned}\Pr\left[I_1 = J \mid j^* \notin I_1\right] &= \Pr\left[I_1 = J \cup \{j^*\} \mid j^* \in I_1\right] \\ &= \Pr\left[I'_1 = J \mid i^* \notin I'_1\right] = \Pr\left[I'_1 = J \cup \{i^*\} \mid i^* \in I'_1\right].\end{aligned} \tag{19}$$

It immediately concludes that $\mu_{\tilde{\mathcal{F}}} = \mu'_{\tilde{\mathcal{E}}}$.

Comparison of $\mu_{\tilde{\mathcal{F}}}$ and $\mu'_{\mathcal{E}}$. Based on Equation (19), it suffices to compare for each $J \subseteq \text{SUPP}(\vec{h}) \cap \text{SUPP}(\vec{h}')$, $\Pr\left[I = S \mid I_1 = J\right]$ and $\Pr\left[I' = S \mid I'_1 = J \cup \{i^*\}\right]$. There are four cases to be discussed.

Case (i). : $J \not\subseteq S$. Then

$$\Pr\left[I = S \mid I_1 = J\right] = \Pr\left[I' = S \mid I'_1 = J \cup \{i^*\}\right] = 0.$$

Case (ii). : $J \subseteq S$ and $i^* \notin S$. Then

$$\Pr\left[I = S \mid I_1 = J\right] \geq \Pr\left[I' = S \mid I'_1 = J \cup \{i^*\}\right] = 0.$$

Case (iii). : $J \subseteq S$ and $i^* \in S$. Recall the definition of I_2 in Algorithm 2, we have

$$\Pr[I = S \mid I_1 = J] = \Pr[I_2 = S \setminus J] = \frac{1}{\binom{d-|J|}{n+k-|J|}}, \tag{20}$$

$$\Pr\left[I' = S \mid I'_1 = J \cup \{i^*\}\right] = \Pr\left[I'_2 = S \setminus (J \cup \{i^*\})\right] = \frac{1}{\binom{d-|J|-1}{n+k-|J|-1}}, \tag{21}$$

Denote $a = d - |J|$ and $b = n + k - |J|$.

$$\frac{\Pr[I = S \mid I_1 = J]}{\Pr\left[I' = S \mid I'_1 = J \cup \{i^*\}\right]} = \frac{\binom{a-1}{b-1}}{\binom{a}{b}} = \frac{b}{a} \geq \frac{k}{d}. \tag{22}$$

It concludes that $\mu_{\tilde{\mathcal{F}}} \geq \frac{k}{d} \cdot \mu'_{\mathcal{E}}$.

Comparison of $\mu'_{\tilde{\mathcal{E}}}$ and $\mu_{\mathcal{F}}$. The comparison follows by symmetry from the analysis of $\mu_{\tilde{\mathcal{F}}}$ and $\mu'_{\mathcal{E}}$. □

A.2 Missing Proofs for Appendix A.1

Proof of Lemma A.1. Recall in Algorithm 2 that

$$p_\tau \doteq \Pr\left[1 + \mathbb{A}\mathrm{px}\mathbb{D}\mathbb{L}\mathrm{ap}_{n,\varepsilon,\varepsilon\gamma/d}(1) \geq \tau\right] = \Pr\left[\hat{h}'[i^*] \geq \tau\right] = \Pr\left[i^* \in I'_1\right].$$

For each $J \subseteq \textsc{supp}(\vec{h})$, it holds that

$$\begin{aligned}
\Pr\left[I'_1 = J\right] &= \Pr\left[i^* \notin I'_1\right] \cdot \Pr\left[Z'_{j^*} = \mathbb{1}_{[j^* \in J]}\right] \cdot \prod_{i \in \textsc{supp}(\vec{h}) \setminus \{j^*\}} \Pr\left[Z'_i = \mathbb{1}_{[i \in J]}\right] \\
&= (1 - p_\tau) \cdot r_J \cdot \Pr\left[Z_{j^*} = \mathbb{1}_{[j^* \in J]}\right] \cdot \prod_{i \in \textsc{supp}(\vec{h}) \setminus \{j^*\}} \Pr\left[Z_i = \mathbb{1}_{[i \in J]}\right] \\
&= (1 - p_\tau) \cdot r_J \cdot \Pr[I_1 = J],
\end{aligned}$$

Similarly,

$$\begin{aligned}
\Pr\left[I'_1 = J \cup \{i^*\}\right] &= \Pr\left[i^* \in I'_1\right] \cdot \Pr\left[Z'_{j^*} = \mathbb{1}_{[j^* \in J]}\right] \cdot \prod_{i \in \textsc{supp}(\vec{h}) \setminus \{j^*\}} \Pr\left[Z'_i = \mathbb{1}_{[i \in J]}\right] \\
&= p_\tau \cdot r_J \cdot \Pr\left[Z_{j^*} = \mathbb{1}_{[j^* \in J]}\right] \cdot \prod_{i \in \textsc{supp}(\vec{h}) \setminus \{j^*\}} \Pr\left[Z_i = \mathbb{1}_{[i \in J]}\right] \\
&= p_\tau \cdot r_J \cdot \Pr[I_1 = J].
\end{aligned}$$

□

Proof of Lemma A.2. There are several cases.

Case One. : $J \not\subseteq S$, then

$$\Pr[I = S \mid I_1 = J] = \Pr\left[I' = S \mid I'_1 = J\right] = \Pr\left[I' = S \mid I'_1 = J \cup \{i^*\}\right] = 0.$$

Case Two. : $J \subseteq S$ and $i^* \notin S$, then

$$\Pr\left[I' = S \mid I'_1 = J \cup \{i^*\}\right] = 0 \leq \Pr[I = S \mid I_1 = J]. \tag{23}$$

Case Three. : $J \subseteq S$ and $i^* \in S$. First,

$$\Pr\left[I = S \mid I_1 = J\right] = \Pr\left[I_2 = S \setminus J\right] = \frac{1}{\binom{d-|J|}{n+k-|J|}}, \tag{24}$$

$$\Pr\left[I' = S \mid I'_1 = J \cup \{i^*\}\right] = \Pr\left[I'_2 = S \setminus (J \cup \{i^*\})\right] = \frac{1}{\binom{d-|J|-1}{n+k-|J|-1}}. \tag{25}$$

Denote $a = d - |J|$ and $b = n + k - |J|$.

$$\frac{\Pr\left[I' = S \mid I'_1 = J \cup \{i^*\}\right]}{\Pr\left[I = S \mid I_1 = J\right]} = \frac{\binom{a}{b}}{\binom{a-1}{b-1}} = \frac{a}{b}. \tag{26}$$

There is an intuitive explanation for this ratio: conditioned on $i^* \notin I_1 = J, \Pr\left[i^* \in S\right] = \Pr\left[i^* \in I_2\right] = \frac{b}{a}$. It is easy to see that conditioned on $i^* \in I_2$, the set I_2 has exactly the same distribution as I'_2 and therefore

$$\Pr\left[I_2 = S \setminus J \mid i^* \in I_2\right] = \Pr\left[I'_2 = S \setminus (J \cup \{i^*\})\right].$$

□

References

[1] E. Anderson, M. Chase, F. B. Durak, K. Laine, and C. Weng, "Precio: Private aggregate measurement via oblivious shuffling," in *Proceedings of the 2024 on ACM SIGSAC Conference on Computer and Communications Security, CCS 2024, Salt Lake City, UT, USA, October 14-18, 2024*. ACM, 2024, pp. 1819–1833.

[2] Y. Arbitman, M. Naor, and G. Segev, "Backyard cuckoo hashing: Constant worst-case operations with a succinct representation," in *2010 IEEE 51st Annual Symposium on Foundations of Computer Science*, 2010, pp. 787–796.

[3] V. Balcer and S. P. Vadhan, "Differential privacy on finite computers," *J. Priv. Confidentiality*, vol. 9, no. 2, 2019. [Online]. Available: https://doi.org/10.29012/jpc.679

[4] B. Balle, J. Bell, A. Gascón, and K. Nissim, "The privacy blanket of the shuffle model," in *Advances in Cryptology - CRYPTO 2019 - 39th Annual International Cryptology Conference, Santa Barbara, CA, USA, August 18-22, 2019, Proceedings, Part II*, ser. Lecture Notes in Computer Science, A. Boldyreva and D. Micciancio, Eds., vol. 11693. Springer, 2019, pp. 638–667.

[5] R. Bassily and A. D. Smith, "Local, private, efficient protocols for succinct histograms," in *Proceedings of the Forty-Seventh Annual ACM on Symposium on Theory of Computing, STOC 2015, Portland, OR, USA, June 14-17, 2015*, R. A. Servedio and R. Rubinfeld, Eds. ACM, 2015, pp. 127–135.

[6] A. Beimel, H. Brenner, S. P. Kasiviswanathan, and K. Nissim, "Bounds on the sample complexity for private learning and private data release," *Mach. Learn.*, vol. 94, no. 3, pp. 401–437, 2014. [Online]. Available: https://doi.org/10.1007/s10994-013-5404-1

[7] J. Bell, A. Gascón, B. Ghazi, R. Kumar, P. Manurangsi, M. Raykova, and P. Schoppmann, "Distributed, private, sparse histograms in the two-server model," in *Proceedings of the 2022 ACM SIGSAC Conference on Computer and Communications Security, CCS 2022, Los Angeles, CA, USA, November 7-11, 2022*, H. Yin, A. Stavrou, C. Cremers, and E. Shi, Eds. ACM, 2022, pp. 307–321. [Online]. Available: https://doi.org/10.1145/3548606.3559383

[8] M. Ben-Or, S. Goldwasser, and A. Wigderson, "Completeness theorems for non-cryptographic fault-tolerant distributed computation (extended abstract)," in *Proceedings of the 20th Annual ACM Symposium on Theory of Computing, May 2-4, 1988, Chicago, Illinois, USA*, J. Simon, Ed. ACM, 1988, pp. 1–10. [Online]. Available: https://doi.org/10.1145/62212.62213

[9] A. Bittau, Ú. Erlingsson, P. Maniatis, I. Mironov, A. Raghunathan, D. Lie, M. Rudominer, U. Kode, J. Tinnés, and B. Seefeld, "Prochlo: Strong privacy for analytics in the crowd," in *Proceedings of the 26th Symposium on Operating Systems Principles, Shanghai, China, October 28-31, 2017*. ACM, 2017, pp. 441–459. [Online]. Available: https://doi.org/10.1145/3132747.3132769

[10] M. Bun, J. Nelson, and U. Stemmer, "Heavy hitters and the structure of local privacy," *ACM Trans. Algorithms*, vol. 15, no. 4, pp. 51:1–51:40, 2019.

[11] M. Bun, K. Nissim, and U. Stemmer, "Simultaneous private learning of multiple concepts," *J. Mach. Learn. Res.*, vol. 20, pp. 94:1–94:34, 2019. [Online]. Available: https://jmlr.org/papers/v20/18-549.html

[12] C. L. Canonne, G. Kamath, and T. Steinke, "The discrete gaussian for differential privacy," in *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual*, H. Larochelle, M. Ranzato, R. Hadsell, M. Balcan, and H. Lin, Eds., 2020. [Online]. Available: https://proceedings.neurips.cc/paper/2020/hash/b53b3a3d6ab90ce0268229151c9bde11-Abstract.html

[13] J. Champion, A. Shelat, and J. R. Ullman, "Securely sampling biased coins with applications to differential privacy," in *Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security, CCS 2019, London, UK, November 11-15, 2019*, L. Cavallaro, J. Kinder, X. Wang, and J. Katz, Eds. ACM, 2019, pp. 603–614. [Online]. Available: https://doi.org/10.1145/3319535.3354256

[14] A. Cheu, A. D. Smith, J. R. Ullman, D. Zeber, and M. Zhilyaev, "Distributed differential privacy via shuffling," in *Advances in Cryptology - EUROCRYPT 2019 - 38th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Darmstadt, Germany, May 19-23, 2019, Proceedings, Part I*, ser. Lecture Notes in Computer Science, Y. Ishai and V. Rijmen, Eds., vol. 11476. Springer, 2019, pp. 375–403.

[15] G. Cormode, C. M. Procopiuc, D. Srivastava, and T. T. L. Tran, "Differentially private summaries for sparse data," in *15th International Conference on Database Theory, ICDT '12, Berlin, Germany, March 26-29, 2012*, A. Deutsch, Ed. ACM, 2012, pp. 299–311. [Online]. Available: https://doi.org/10.1145/2274576.2274608

[16] H. Corrigan-Gibbs and D. Boneh, "Prio: Private, robust, and scalable computation of aggregate statistics," in *14th USENIX Symposium on Networked Systems Design and Implementation, NSDI 2017, Boston, MA, USA, March 27-29, 2017*, A. Akella and J. Howell, Eds. USENIX Association, 2017, pp. 259–282.

[17] Y. B. Dov, L. David, M. Naor, and E. Tzalik, "Resistance to timing attacks for sampling and privacy preserving schemes," in *4th Symposium on Foundations of Responsible Computing, FORC 2023, June 7-9, 2023, Stanford University, California, USA*, ser. LIPIcs, K. Talwar, Ed., vol. 256. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2023, pp. 11:1–11:23. [Online]. Available: https://doi.org/10.4230/LIPIcs.FORC.2023.11

[18] C. Dwork, "Differential privacy," in *Automata, Languages and Programming, 33rd International Colloquium, ICALP 2006, Venice, Italy, July 10-14, 2006, Proceedings, Part II*, ser. Lecture Notes in Computer Science, M. Bugliesi, B. Preneel,

V. Sassone, and I. Wegener, Eds., vol. 4052. Springer, 2006, pp. 1–12.
[19] C. Dwork and A. Roth, "The algorithmic foundations of differential privacy," *Found. Trends Theor. Comput. Sci.*, vol. 9, no. 3-4, pp. 211–407, 2014.
[20] C. Dwork, K. Kenthapadi, F. McSherry, I. Mironov, and M. Naor, "Our data, ourselves: Privacy via distributed noise generation," in *Advances in Cryptology - EUROCRYPT 2006, 25th Annual International Conference on the Theory and Applications of Cryptographic Techniques, St. Petersburg, Russia, May 28 - June 1, 2006, Proceedings*, ser. Lecture Notes in Computer Science, S. Vaudenay, Ed., vol. 4004. Springer, 2006, pp. 486–503. [Online]. Available: https://doi.org/10.1007/11761679_29
[21] C. Dwork, F. McSherry, K. Nissim, and A. D. Smith, "Calibrating noise to sensitivity in private data analysis," in *Theory of Cryptography, Third Theory of Cryptography Conference, TCC 2006, New York, NY, USA, March 4-7, 2006, Proceedings*, ser. Lecture Notes in Computer Science, S. Halevi and T. Rabin, Eds., vol. 3876. Springer, 2006, pp. 265–284.
[22] G. C. Fanti, V. Pihur, and Ú. Erlingsson, "Building a RAPPOR with the unknown: Privacy-preserving learning of associations and data dictionaries," *Proc. Priv. Enhancing Technol.*, vol. 2016, no. 3, pp. 41–61, 2016.
[23] O. Franzese, C. Fang, R. Garg, S. Jha, N. Papernot, X. Wang, and A. Dziedzic, "Secure noise sampling for differentially private collaborative learning," Cryptology ePrint Archive, Paper 2025/1025, 2025. [Online]. Available: https://eprint.iacr.org/2025/1025
[24] B. Ghazi, N. Golowich, R. Kumar, R. Pagh, and A. Velingker, "On the power of multiple anonymous messages: Frequency estimation and selection in the shuffle model of differential privacy," in *Advances in Cryptology - EUROCRYPT 2021 - 40th Annual International Conference on the Theory and Applications of Cryptographic Techniques, Zagreb, Croatia, October 17-21, 2021, Proceedings, Part III*, ser. Lecture Notes in Computer Science, A. Canteaut and F. Standaert, Eds., vol. 12698. Springer, 2021, pp. 463–488. [Online]. Available: https://doi.org/10.1007/978-3-030-77883-5_16
[25] A. Ghosh, T. Roughgarden, and M. Sundararajan, "Universally utility-maximizing privacy mechanisms," in *Proceedings of the 41st Annual ACM Symposium on Theory of Computing, STOC 2009, Bethesda, MD, USA, May 31 - June 2, 2009*, M. Mitzenmacher, Ed. ACM, 2009, pp. 351–360.
[26] O. Goldreich, S. Micali, and A. Wigderson, "How to play any mental game or A completeness theorem for protocols with honest majority," in *Proceedings of the 19th Annual ACM Symposium on Theory of Computing, 1987, New York, New York, USA*, A. V. Aho, Ed. ACM, 1987, pp. 218–229. [Online]. Available: https://doi.org/10.1145/28395.28420
[27] Google, "Secure noise generation," 2020. [Online]. Available: https://github.com/google/differential-privacy/blob/master/common_docs/Secure_Noise_Generation.pdf
[28] M. Hardt and K. Talwar, "On the geometry of differential privacy," in *Proceedings of the 42nd ACM Symposium on Theory of Computing, STOC 2010, Cambridge, Massachusetts, USA, 5-8 June 2010*, L. J. Schulman, Ed. ACM, 2010, pp. 705–714. [Online]. Available: https://doi.org/10.1145/1806689.1806786
[29] J. Jin, E. McMurtry, B. I. P. Rubinstein, and O. Ohrimenko, "Are we there yet? timing and floating-point attacks on differential privacy systems," in *43rd IEEE Symposium on Security and Privacy, SP 2022, San Francisco, CA, USA, May 22-26, 2022*. IEEE, 2022, pp. 473–488. [Online]. Available: https://doi.org/10.1109/SP46214.2022.9833672
[30] H. Keller, H. Möllering, T. Schneider, O. Tkachenko, and L. Zhao, "Secure noise sampling for DP in MPC with finite precision," in *Proceedings of the 19th International Conference on Availability, Reliability and Security, ARES 2024, Vienna, Austria, 30 July 2024 - 2 August 2024*. ACM, 2024, pp. 25:1–25:12. [Online]. Available: https://doi.org/10.1145/3664476.3664490
[31] F. Kerschbaum, S. Lee, and H. Wu, "Optimal pure differentially private sparse histograms in near-linear deterministic time," *CoRR*, vol. abs/2507.17017, 2025. [Online]. Available: https://doi.org/10.48550/arXiv.2507.17017
[32] B. Knott, S. Venkataraman, A. Y. Hannun, S. Sengupta, M. Ibrahim, and L. van der Maaten, "Crypten: Secure multi-party computation meets machine learning," in *Advances in Neural Information Processing Systems 34: Annual Conference on Neural Information Processing Systems 2021, NeurIPS 2021, December 6-14, 2021, virtual*, M. Ranzato, A. Beygelzimer, Y. N. Dauphin, P. Liang, and J. W. Vaughan, Eds., 2021, pp. 4961–4973. [Online]. Available: https://proceedings.neurips.cc/paper/2021/hash/2754518221cfbc8d25c13a06a4cb8421-Abstract.html
[33] D. E. Knuth and A. C.-C. Yao, "The complexity of nonuniform random number generation," in *Algorithm and Complexity, New Directions and Results*, 1976. [Online]. Available: https://api.semanticscholar.org/CorpusID:115400979
[34] A. Korolova, K. Kenthapadi, N. Mishra, and A. Ntoulas, "Releasing search queries and clicks privately," in *Proceedings of the 18th International Conference on World Wide Web, WWW 2009, Madrid, Spain, April 20-24, 2009*, J. Quemada, G. León, Y. S. Maarek, and W. Nejdl, Eds. ACM, 2009, pp. 171–180. [Online]. Available: https://doi.org/10.1145/1526709.1526733
[35] C. J. Lebeda and J. Tetek, "Better differentially private approximate histograms and heavy hitters using the misra-gries sketch," *SIGMOD Rec.*, vol. 53, no. 1, pp. 7–14, 2024. [Online]. Available: https://doi.org/10.1145/3665252.3665255
[36] C. J. Lebeda, M. Aumüller, and R. Pagh, "Representing sparse vectors with differential privacy, low error, optimal space, and fast access," *J. Priv. Confidentiality*, vol. 12, no. 2, 2022. [Online]. Available: https://doi.org/10.29012/jpc.809
[37] D. R. Lolck and R. Pagh, "Shannon meets gray: Noise-robust, low-sensitivity codes with applications in differential privacy," in *Proceedings of the 2024 ACM-SIAM Symposium on Discrete Algorithms, SODA 2024,*

Alexandria, VA, USA, January 7-10, 2024, D. P. Woodruff, Ed. SIAM, 2024, pp. 1050–1066. [Online]. Available: https://doi.org/10.1137/1.9781611977912.40

[38] I. Mironov, "On significance of the least significant bits for differential privacy," in *the ACM Conference on Computer and Communications Security, CCS'12, Raleigh, NC, USA, October 16-18, 2012*, T. Yu, G. Danezis, and V. D. Gligor, Eds. ACM, 2012, pp. 650–661. [Online]. Available: https://doi.org/10.1145/2382196.2382264

[39] M. Mitzenmacher and E. Upfal, *Probability and Computing: Randomized Algorithms and Probabilistic Analysis*. Cambridge University Press, 2005.

[40] Y. Qiu and K. Yi, "Approximate DBSCAN under differential privacy," *CoRR*, vol. abs/2508.08749, 2025. [Online]. Available: https://doi.org/10.48550/arXiv.2508.08749

[41] Z. Ratliff and S. P. Vadhan, "A framework for differential privacy against timing attacks," in *Proceedings of the 2024 on ACM SIGSAC Conference on Computer and Communications Security, CCS 2024, Salt Lake City, UT, USA, October 14-18, 2024*, B. Luo, X. Liao, J. Xu, E. Kirda, and D. Lie, Eds. ACM, 2024, pp. 3615–3629. [Online]. Available: https://doi.org/10.1145/3658644.3690206

[42] ——, "Securing unbounded differential privacy against timing attacks," *arXiv preprint arXiv:2506.07868*, 2025.

[43] A. J. Walker, "An efficient method for generating discrete random variables with general distributions," *ACM Trans. Math. Softw.*, vol. 3, no. 3, pp. 253–256, 1977. [Online]. Available: https://doi.org/10.1145/355744.355749

[44] C. Wei, R. Yu, Y. Fan, W. Chen, and T. Wang, "Securely sampling discrete gaussian noise for multi-party differential privacy," in *Proceedings of the 2023 ACM SIGSAC Conference on Computer and Communications Security, CCS 2023, Copenhagen, Denmark, November 26-30, 2023*, W. Meng, C. D. Jensen, C. Cremers, and E. Kirda, Eds. ACM, 2023, pp. 2262–2276. [Online]. Available: https://doi.org/10.1145/3576915.3616641

[45] H. Wu and A. Wirth, "Asymptotically optimal locally private heavy hitters via parameterized sketches," in *International Conference on Artificial Intelligence and Statistics, AISTATS 2022, 28-30 March 2022, Virtual Event*, ser. Proceedings of Machine Learning Research, G. Camps-Valls, F. J. R. Ruiz, and I. Valera, Eds., vol. 151. PMLR, 2022, pp. 7766–7798.

[46] S. Zahur and D. Evans, "Circuit structures for improving efficiency of security and privacy tools," in *2013 IEEE Symposium on Security and Privacy, SP 2013, Berkeley, CA, USA, May 19-22, 2013*. IEEE Computer Society, 2013, pp. 493–507. [Online]. Available: https://doi.org/10.1109/SP.2013.40

Received December 2025; accepted February 2026