Sample-and-threshold differential privacy: Histograms and applications

Anonymous Author
Anonymous Institution

Abstract

Federated analytics relies on the collection of accurate statistics about distributed users with a suitable guarantee. In this paper, we show how a strong (ε, δ)-privacy guarantee can be achieved for the fundamental problem of histogram generation in a federated setting, via a highly practical sampling-based procedure. Given such histograms, related problems such as heavy hitters and quantiles can be answered with provable error and privacy guarantees. Our experimental results demonstrate that this sample-and-threshold approach is both accurate and scalable.

1 Introduction

Building private histograms is a task that underpins a variety of machine learning and data analytics tasks. Histograms enable building usable discrete representations, distributions and marginals. Materializing histograms is thus a core subroutine in instantiating graphical models for synthetic data generation (McKenna et al., 2021), and hence they support numerous statistical analyses and inference tasks. The problem has been heavily studied in the setting of differential privacy, with a number of results shown under variant models, such as the central model (Dwork, 2006; Xu et al., 2012; Dwork and Roth, 2014), local model (Bassily and Smith, 2015; Wang et al., 2017; Acharya et al., 2019) and shuffle model (Erlingsson et al., 2020; Balcer and Cheu, 2020; Li et al., 2020).

In this paper, we revisit this foundational question, and show how differential privacy can be obtained via a simple sample-and-threshold mechanism, which can be readily implemented in a distributed setting. Importantly, all the randomization needed for privacy is derived from the sampling operator: there is no additional explicit addition of noise. This is particularly beneficial in scenarios when sampling is inherent, i.e., federated settings when only a uniformly chosen fraction of users are contacted. In this case, privacy essentially comes "for free". Equipped with an efficient mechanism for histogram computation, we can apply it to a range of core analytics tasks (quantiles and heavy hitters), which in turn enable a broad spectrum of other computations.

Our contributions. In this paper, we present a histogram mechanism that extends prior work as follows:

- We show that a simple sample-and-threshold approach provides an (ε, δ)-differential privacy guarantee for histograms.
- We show that the resulting mechanism can also answer heavy hitter, quantile and range queries.
- We show that the associated counts provide accurate frequency estimates for items from the input.
- Our proofs are compact and self-contained.

In more detail, we show that a Poisson sampling based approach is sufficient to provide differential privacy. The key is to choose a sampling rate that is not too large compared to population size, and to prune items with low frequency in the sample, so that the presence of an item in the pruned sample does not indicate exactly how many instances were in the original population. While prior work has considered the ability of sampling to amplify the privacy bounds of a differentially private mechanism, in this work we show that sampling itself provides a DP histogram mechanism, similar to the pioneering work of Zhu et al. (2020) on heavy hitters. Consequently, the sample-and-threshold histogram mechanism can be implemented effectively while requiring very little effort from participating users. The chief points of comparison are results in the shuffle model of differential privacy. We claim that the sampling step is arguably simpler than many shuffle approaches (which require users

Preliminary work. Under review by AISTATS 2022. Do not distribute.

to perturb their inputs, or add additional "chaff" messages to mask their values), while being of equivalent complexity to implement the server-side aggregation of messages. Deployed federated systems (Bonawitz et al., 2019) already implicitly sample from a large collection of eligible users, so the mechanism does not introduce any significant additional overhead or error.

2 Preliminaries

We consider the case where there are n individuals who each hold a value x_i, so that the collection of all user inputs defines a dataset D. Our goal is to construct a histogram of the frequency distribution according to a fixed set of buckets B. For convenience, we assume that each input x_i is already mapped into its corresponding bucket, and that the buckets are indexed by integers, so that each $x_i \in [B]$. We will describe a randomized mechanism, $\mathscr{M}$, that can process datasets D to give a distribution over output histograms, H.

The objective is to ensure that a sampled output histogram, H, is close to the true histogram H^*, while ensuring that the output meets (ε, δ)-differential privacy (DP) (Dwork and Roth, 2014). Formally, we require

$$\Pr[\mathscr{M}(D) \in \mathbb{H}] \leq \exp(\varepsilon) \Pr[\mathscr{M}(D') \in \mathbb{H}] + \delta \qquad (1)$$

for any subset of possible output histograms $\mathbb{H}$ and for neighboring inputs D, D' that differ in the input value of one individual. As usual for (ε, δ)-DP, we expect δ to be small, typically much less than $1/n$.

Computational model. Our mechanism is designed to operate in a federated (distributed) setting, where each client sends a message based on their input to a server, which then combines this information before reporting it to an analyst. Specifically, the server aggregates the messages to produce the multiset of values reported (i.e., builds the frequency histogram of messages), and deletes some values which fall below a threshold τ. This model sits between the shuffle and centralized DP model: the procedure is conceptually similar to the 'shuffling' procedure, but with the minor additional step of removing small counts; meanwhile, it is easy to implement in the central DP model with a trusted aggregator. To fully achieve the benefits of this model, we assume that there is an entity which aggregates the data, similar to a shuffler in the shuffle model. For a shuffler, applying the threshold would be a trivial final step before the shuffler releases the histogram. Indeed, we anticipate that this would be natural to do in any system that implements aggregation via secure hardware (e.g., SGX extensions). Then the data analyst only sees output under differential privacy, and is shielded from seeing any intermediate results without a formal privacy property. The model can also be compared to the early notion of "k-anonymity", where the output is constrained so that every output item corresponds to at least k individuals in the input (Samarati and Sweeney, 1998). Here, we obtain k-anonymity for $k = \tau$, the threshold value. Although k-anonymity has been criticized as a weak privacy notion, it carries an intuitive appeal for many lay users, and here we show that in this case we also achieve differential privacy.

3 Sampling-Based Histogram Mechanism

In the (B-bucket) histogram problem, each client i holds a single item x_i corresponding to a bucket $b_i \in [B]$, and our aim is to produce a private histogram of item frequencies, such that a frequency associated with x in the private histogram approximates the frequency of x over the input distribution.

The algorithm is based on Bernoulli sampling. Each client out of n is sampled with probability $p_s = m/n$, so the expected size of the sample is m (we later discuss different ways to implement this sampling). Our subsequent analysis will set an upper bound on the sample size m in order to give a required privacy guarantee. The algorithm makes use of a threshold τ, so that items whose sampled counts are at least τ are reported in the histogram, while items whose count falls below τ are omitted from the histogram. Note then that the mechanism introduces no spurious items into the output: any item which is not present in the input cannot appear in the output histogram. In addition, the costs of the algorithm are independent of the dimensionality of the underlying histogram, B.

Lemma 1. *The probability that the number of samples of an item is more than τ times its expectation is at most δ, for $\tau = 3 + \ln 1/\delta$.*

The proof of this claim, and of most technical lemmas, is deferred to the supplementary material. We next give a bound on the ratio of probabilities of seeing the same output on neighboring inputs.

Lemma 2. *Given two neighboring inputs D, D', such that D differs in one item from D', the ratio of probabilities of seeing a cell with a given value v is bounded by $\frac{k+1}{k+1-v}$, where $k+1$ is the number of copies of the given item in input D and $k > v$.*

Theorem 1. *The resulting histogram obeys (ε, δ)-differential privacy, for $\delta = O(\exp(-\tau))$ and $\varepsilon = O(\frac{m}{n} \ln(1/\delta)) \leq 1$.*

Proof. Consider the treatment of an item x between two neighboring inputs D and D'. If $f_x = f'_x$, i.e., the number of copies of x is the same in both inputs, then x is

treated identically in both cases. Otherwise, wlog we are looking at an x such that $f_x = f'_x + 1 = k+1$. We condition on the event that the number of samples of the item x is not more than τ times its expectation. Call this event E. By Lemma 1, event E holds except with probability $\delta = \exp(-(\tau-1)^2/(\tau+1)) = O(\exp(-\tau))$. We condition on E holding, and just account for this probability in our final reckoning.

Suppose that the count of x for D is less than n/m. Then, by our assumption of E, D will not sample τ copies of x, and so both D and D' would output the same histogram. Hence, the probability of all outputs are equal on D and on D'.

Otherwise, the count of x ($k+1$ for D) is at least n/m, and by our assumption D samples at most $v \le \tau m(k+1)/n$ copies of x. Then, by Lemma 2, we can state that for the mechanism M, the probability of seeing a given output histogram H satisfies:

$$\frac{\Pr[M(D)=H|E]}{\Pr[M(D')=H|E]} \le \frac{k+1}{k+1-v} \le \frac{k+1}{k+1-\tau(k+1)m/n} = \frac{n}{n-\tau m} \quad (2)$$

We will assume that $m = c_\varepsilon \frac{n}{\tau}$ for a constant $c_\varepsilon \le 1 - 1/e$ that depends on ε. The effect is to ensure that the sample size m is a small fraction of n. Substituting this assumption in (2), we conclude

$$\frac{\Pr[M(D)=H|E]}{\Pr[M(D')=H|E]} = \frac{n}{n-c_\varepsilon n} = \frac{1}{1-c_\varepsilon} := \exp(\varepsilon) \quad (3)$$

That is, except with probability δ, we have ε-differential privacy (1). Rearranging, we set $c_\varepsilon = 1-\exp(-\varepsilon)$. For small ε, we can approximate $c_\varepsilon = \varepsilon$. We can also write

$$\varepsilon = \ln \frac{1}{1-c_\varepsilon} = \ln 1 + \tfrac{c_\varepsilon}{1-c_\varepsilon} = \ln 1 + \tfrac{m\tau/n}{1-c_\varepsilon} \le \ln(1+\tfrac{em\tau}{n})$$

where the last step uses $c_\varepsilon \le 1-1/e$ for $\varepsilon \le 1$. This proves $(O(\frac{m\tau}{n}), O(\exp(-\tau))$-differential privacy, as claimed. □

3.1 Fixed sized sampling

For practical efficiency, we would often like to work with a fixed size sample. However, the above histogram protocol performs Poisson sampling instead. The reason is that if the fixed size of the sample, m, is known, then we are effectively also releasing the number of samples that were suppressed by the τ threshold (by adding up the released counts, and subtracting from m). This potentially leaks information. Consider the case where D' contains n copies of the same item, while D contains $n-1$ copies of the same item, and one unique item. With probability m/n, the mechanism on input D samples the unique item along with $m-1$ other items, and so produces a sample of size $m-1$. But on input D', there is zero probability of producing a sample smaller than m. This forces $\delta \ge m/n$, which is typically too large for (ε,δ)-DP (we usually seek $\delta \ll 1/n$).

Performing Poisson sampling with p_s addresses this problem: the expected sample size is the same, but we no longer leak the true size of the sample before thresholding. Indeed, we can see that the (observable) size of the sample is differentially private: given two inputs D and D' such that D has one additional unique item, the distribution of sample sizes are close, up to a factor of $1+p_s/(1-p_s) = 1+m/(n-m)$, which is below $\exp(\varepsilon_i)$ by (3).

Implementing Poisson sampling may appear costly: naively, the server would contact n clients instead of m, where we expect $n \gg m$. However, we can perform the sampling by contacting much fewer clients, since the size of the sample is tightly concentrated around its expectation.

Lemma 3. *Sampling $m + O(\sqrt{m})$ clients is sufficient to apply the sample-and-threshold mechanism, with high probability.*

3.2 Accuracy Bounds

The histogram produced by the mechanism is ultimately based on sampling and pruning, so for any item whose frequency is sufficiently above the pruning threshold, then its frequency within the histogram is an (almost) unbiased estimate of its true frequency. There is a small gap, since even for an item with high frequency, there is a small chance that it is not sampled often enough, and so its estimate will fall below the threshold τ (in which case we do not report the item).

Probability of omitting a heavy item. We first consider the probability that a frequent item is not reported by the algorithm.

Lemma 4. *The sample and threshold histogram protocol omits an item whose true frequency is W with probability at most* $\exp(-(\frac{Wm}{n}-\tau)^2\frac{n}{2Wm})$.

Numeric Example. When $w := Wm/n$ is sufficiently bigger than τ, this gives a very strong probability. For example, consider the case $n = 10^6$, $\varepsilon = 1$, and we set $\tau = 20$ to obtain a δ of 10^{-8}. The expected sample size $m = 31,606$, and for an item that occurs 0.2% of the time in the input, we expect to sample it $w = 63$ times. This gives a bound of $\exp(-14) < 10^{-7}$ that such an item is not detected.

Frequency estimation bounds. More generally, we can use the frequency of any item in the histogram as an estimate for its true occurrence rate.

Lemma 5. *We can estimate the frequency of any item whose relative frequency is ϕ within γ relative error with probability $O(\exp(-\gamma^2\phi m))$.*

Numeric Example. We can substitute values into this expression to explore the space. For example, if we set $\varepsilon = 1$, $\ln(1/2\beta) = 10$, $\tau = 10$ and $\gamma = 1/\sqrt{10}$, then we obtain $\phi = 3 \times 10^3/n$ — in other words, provided $n > 3 \times 10^5$, we can accurately find estimates of frequencies that occur 1% of the time (except with vanishingly small probability).

Remark. It is instructive to compare these bounds to those that hold for the shuffle model. According to Balcer and Cheu (2020), addition of appropriately parameterized Bernoulli random noise to reports from n clients yields (ε, δ)-DP, with error that scales as $O(\frac{1}{\varepsilon^2 n}\log(1/\delta))$ for $\varepsilon \leq 1$, provided n is large enough. Expressing our bound on the estimate of any frequency, we obtain error $O(1/\sqrt{m}) = O(\sqrt{\frac{\ln 1/\delta}{\varepsilon n}})$ from sampling, plus error from rounding small values down to zero, which is bounded by $O(\tau/m) = O(\ln(1/\delta)/m) = O(\ln^2(1/\delta)/(\varepsilon n))$. Naively, it might seem that the shuffle bounds are preferable, due to the stronger dependence on n ($O(1/n)$ vs. $O(1/\sqrt{n})$). However, this misses the point that in practical federated computing settings, the server can contact only a fixed size cohort of m clients out of a much larger (and sometimes unknown) population n. In this case, results in both the shuffle and sample-and-threshold paradigms incur *the same* sampling error of $O(1/\sqrt{m})$. Then shuffling introduces additional noise of $O(\frac{1}{\varepsilon^2 m}\log(1/\delta))$, whereas sample-and-threshold incurs zero additional noise on items that exceed the τ threshold, and at most $O(\ln(1/\delta)/m)$ on small items. Hence, we argue that when shuffling implicitly samples from the input, the sample-and-threshold approach has superior error guarantees. We confirm this observation empirically in Section 5, where we compare accuracy of both approaches.

4 Heavy hitters and Quantiles via Histograms

4.1 Heavy Hitters

We next show how to use the basic histogram protocol to find the (hierarchical) heavy hitters from the input. This result follows the outline and notation of the TrieHH algorithm (Zhu et al., 2020), to allow easy comparison.

The heavy hitters algorithm proceeds over L levels, to build up a trie of depth L. At each level, we materialize a histogram of those prefixes of items from the input that extend the current trie. This allows us to add items to the current trie based on the threshold τ, and include the observed count of prefix for each node in the trie, provided it is more than τ. We can view the TrieHH protocol as materializing a histogram at each level, with progressively finer cells. In the protocol as originally described, cells whose ancestor in a previous level did not exceed the τ threshold are not eligible for consideration. However, the privacy proof still applies if we do not enforce such restrictions. We denote our version of the protocol using the new histogram protocol as TrieHH++, to indicate that the trie is augmented with count information.

Lemma 6. *The TrieHH++ protocol satisfies (ε, δ)-DP for $\delta = L\exp(-\frac{(\tau-1)^2}{1+\tau})$ and $\varepsilon = O(Lm\ln(1/\delta)/n)$.*

The essence of the proof is that the output of the algorithm is the L-fold composition of a differentially private mechanism, with some post-processing. By the differential privacy of the basic histogram protocol (Theorem 1), the result follows.

Remark. We remark that if the objective is only to find the heavy hitters, then the factor of L can be dropped from these bounds. That is, instead of proceeding in rounds, we simply apply the basic histogram protocol to the full inputs, and report the items which survive the thresholding process (along with their associated counts if desired). Following the above analysis, the resulting output is (ε, δ)-differentially private, when setting $\tau = 3 + \ln 1/\delta$ and $m \leq \frac{\varepsilon n}{\tau}$ to get $\varepsilon = \tau m/n$. The motivation for having L rounds given by Zhu et al. (2020) is to reduce the exposure of the server to private information: they only observe prefixes from clients that extend shorter prefixes that are already known to be popular. However, this does not impact on the formal differential privacy properties of the output.

4.2 Quantiles

Finding the quantiles is a common analytics task to describe the distribution of values held by the clients. We describe two approaches to finding quantiles, both making use of our histogram mechanism.

Single quantiles via interactive search. Given client inputs which fall in the range $[0, 1]$, we seek a value f such that the fraction of clients whose value is below f is (approximately) ϕ.

Lemma 7. *Given a $\phi > \tau/m$, we can use h applications of the histogram mechanism to find a value f such that $f \pm 2^{-h}$ is a $\phi \pm O(m^{-1/2})$ quantile, with $(O(hm\ln(1/\delta)/n, h\delta)$-DP.*

This approach is very effective for single queries, but is less desirable when we have a large number of quantile queries to answer in parallel, in which case the hierarchical histogram approach is preferred.

Quantiles and range queries via hierarchical histograms. A common technique to answer quantile and range queries in one-dimension is to make use of hierachical histograms: histograms with geometrically decreasing bucket sizes, so that any range can be expressed as the union of a small number of buckets. We can observe that the trie built as part of the TrieHH++ protocol is exactly such a hierarchical histogram, and hence can be used to answer quantile queries, with the same privacy (and similar accuracy) guarantees as for heavy hitters.

Assume again that each client has an input value in the range $[0,1]$ (say), and we can interpret these as prefixes, corresponding to subranges. If we set the branching factor of the trie, α, to 4, then the input value $\frac{1}{3}$ falls in the range $[0.25, 0.5]$ for a prefix of length 1; and in the range $[\frac{5}{16}, \frac{6}{16}]$ for a prefix of length 2. With this mapping of input values to prefixes, the algorithm proceeds as before, and outputs the (DP) trie with weights on nodes.

To answer a range query $[0,r]$, we decompose the range greedily into chunks that can be answered by the trie. For example, if $\alpha = 4$, and we want the range $[0, 0.7]$, we find the chunks $[0, \frac{1}{4}], [\frac{1}{4}, \frac{2}{4}]$ at level 1; $[\frac{8}{16}, \frac{9}{16}], [\frac{9}{16}, \frac{10}{16}], [\frac{10}{16}, \frac{11}{16}]$ at level 2; and so on. If the trie has L levels, then any prefix query can be answered with $L(\alpha - 1)$ probes to the histograms ($\alpha - 1$ for each level). Moreover, quantile queries are answered by finding range queries whose weight is (approximately) the desired quantile ϕ.

Due to the pruning, we will not have information on any ranges whose sampled weight is less than τ, corresponding to a τ/m fraction of mass. This will give a worst-case error bound of $(\alpha-1)\tau/m$ per level, and so $L(\alpha-1)\tau/m$ over all levels. Based on our setting of m proportional to $\varepsilon n/(L\tau)$, we obtain a total error of $(\alpha-1)(L\tau)^2/\varepsilon n$. In summary, as a consequence of the privacy guarantee from Lemma 6, we can state:

Lemma 8. *We can build a set of L $(O(Lm\ln(1/\delta)/n, L\delta)$-private histograms to answer any quantile query ϕ to find a value f such that $f \pm 2^{-L}$ is a $\phi \pm O((L\tau)^2/\varepsilon n)$ quantile.*

Numeric Example. Picking similar test values as above shows that this can give reasonable accuracy for n large enough. For $\tau = 10$, $L = 10$, $\alpha = 2$, $\varepsilon = 1$, the error bound yields $10^4/n$. So for $n > 10^6$, we obtain rank queries (and quantiles) in this space with error around 0.01.

5 Experiments

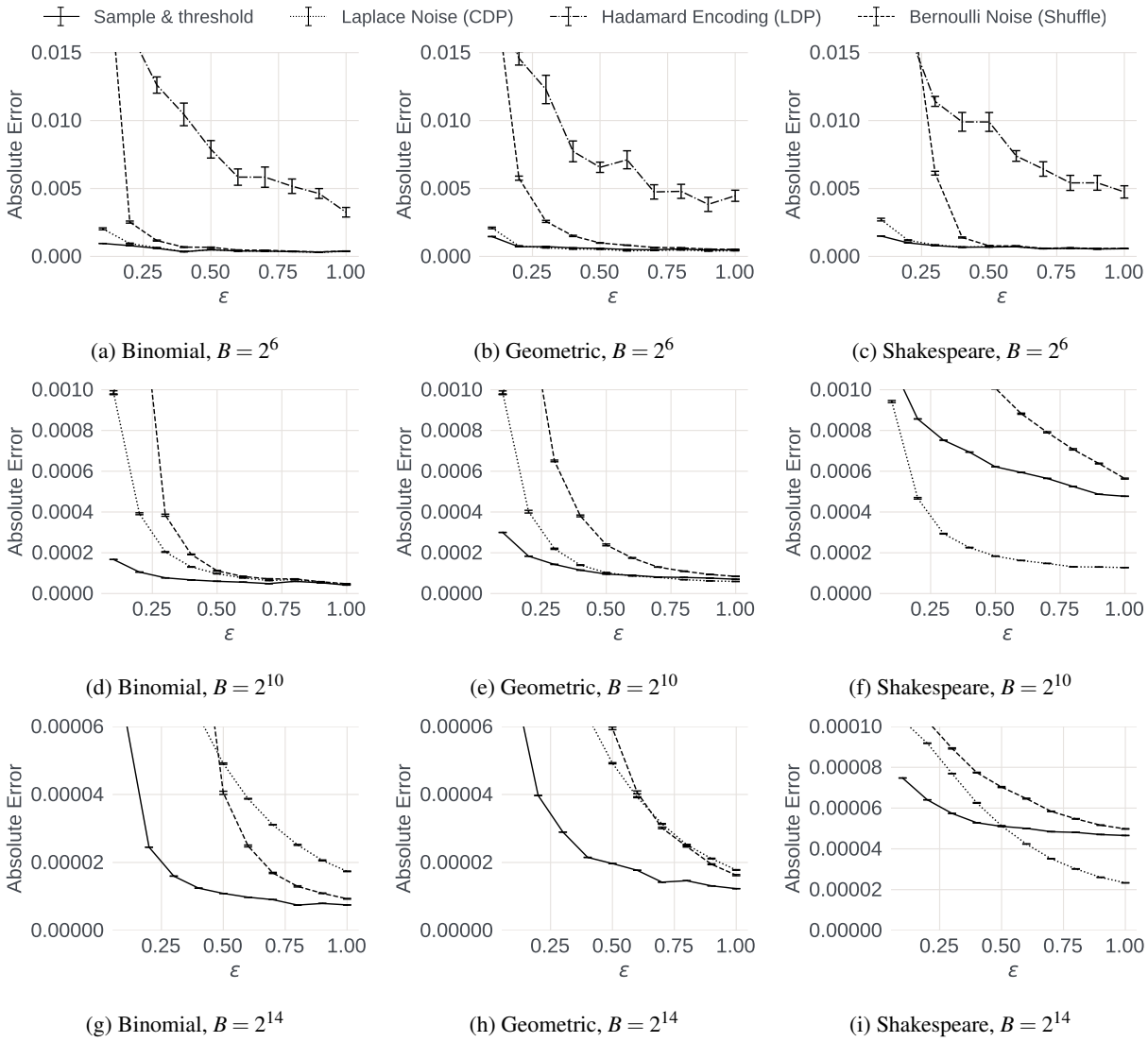
To validate our theoretical understanding, we performed a set of experiments on synthetic data using the histogram mechanism that we have developed. The mechanism performs sampling on a population of size n for a target sample size m, and applies an appropriate threshold to the resulting sample, to achieve an (ε, δ)-DP guarantee. We compared against alternative mechanisms that also provide the same level of privacy when applied to the sampled set of clients: central differential privacy, via Laplace noise addition, local differential privacy based on Hadamard encoding of elements from the domain (Acharya et al., 2019), and a shuffling-approach which adds Bernoulli noise (Balcer and Cheu, 2020).

We worked with the text from the complete works of Shakespeare [1], where we extract each word, consistently map the words to one of the B buckets, and count the total number of words in each bucket. We also use synthetic data generated by distributions providing different frequency distributions: Geometric and Binomial distributions over the B cells of the histogram. For the Binomial data, each client draws from the Binomial distribution with $n = B$ and $p = 0.5$ to choose a histogram bucket. For the Geometric data, each client draws from a Geometric distribution with $p = 1/\sqrt{B}$ to pick a histogram bucket. These parameters are chosen to model the non-uniform frequency distributions seen in practice, where the most popular items occur approximately 1-5% of the time.

We experimented with a range of privacy parameters ε, δ, histogram sizes B, and population sizes n. We pick a default $\delta = 10^{-8}$, which yields a threshold $\tau = 21$. We simulate a population of $n = 10^6$ clients, and measure the accuracy of recovering the frequencies for each mechanism. We compare the absolute difference of the estimated frequencies to those from the full population, and also measure the recall for the top-k heaviest buckets for $k = B/10$, i.e., the largest 10% of frequencies. In the plots, we focus on showing results for the range of $\varepsilon = 0.1$ (high privacy) to $\varepsilon = 1.0$ (medium privacy) regimes, consistent with the range where all the mechanisms have privacy guarantees. We vary the size of the histograms (B) from tens up to tens of thousands. Error bars show the standard error over 10 repetitions of each mechanism. Plots for other parameter settings are withheld for brevity, but support the same conclusions.

Accuracy results. Our results on accuracy are shown in Figure 1. Each row shows results for a different histogram size, from small ($B = 2^6$), to large ($B = 2^{14}$); each

[1] `http://shakespeare.mit.edu/`



(a) Binomial, $B = 2^6$ (b) Geometric, $B = 2^6$ (c) Shakespeare, $B = 2^6$

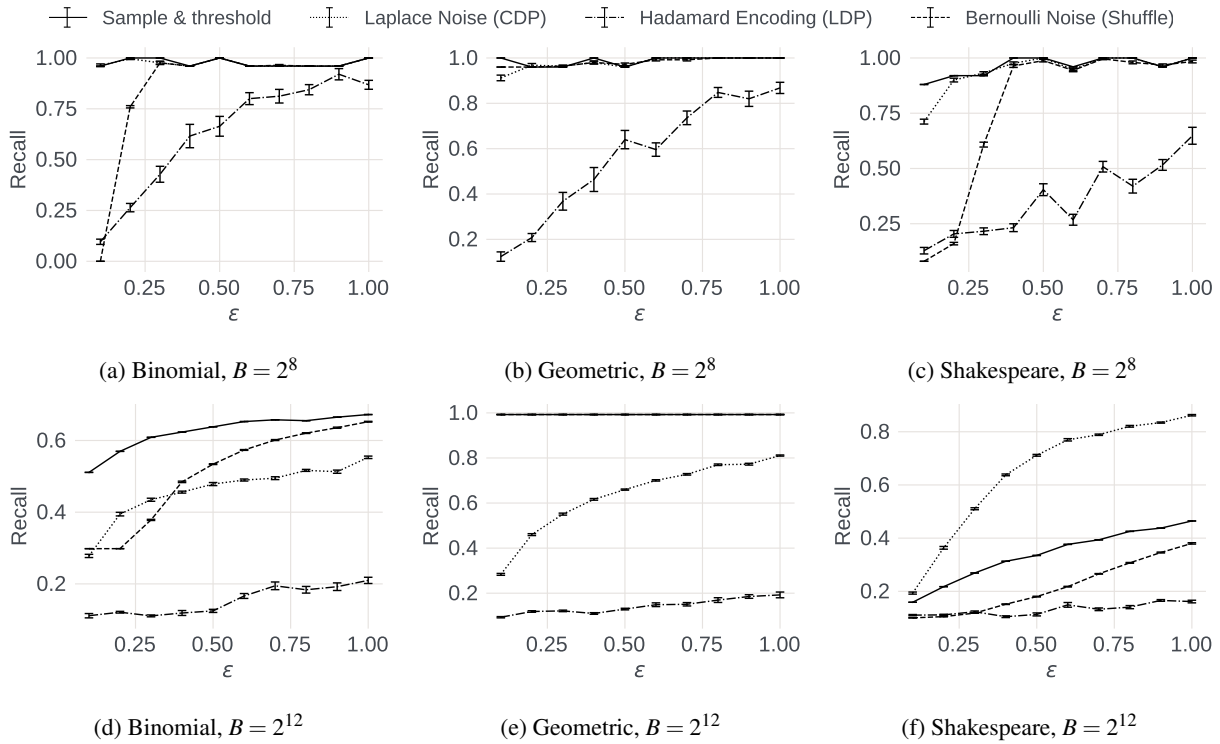
(d) Binomial, $B = 2^{10}$ (e) Geometric, $B = 2^{10}$ (f) Shakespeare, $B = 2^{10}$

(g) Binomial, $B = 2^{14}$ (h) Geometric, $B = 2^{14}$ (i) Shakespeare, $B = 2^{14}$

Figure 1: Accuracy results on Binomial, Geometric and Shakespeare datasets

column shows results on a different dataset (Binomial, Geometric or Shakespeare data). The y-axis shows the absolute error, expressed as a fraction of the total input size. Hence, we want this to be as low as possible, and ideally much smaller than 0.1%, say.

Some results immediately stand out: the results from local differential privacy are much weaker, and frequently the error is sufficiently large that the line does not appear on the plots (similar results were seen for other choices of frequency oracle, such as direct encoding and unary encoding—we use the Hadamard encoding as it obtained the best accruacy for these experiments). This is consistent with our understanding of LDP, and further motivates the desire to achieve accuracy closer to the centralized case in federated settings. The approach from the shuffle model, where each client adds Bernoulli noise to each cell of the histogram (i.e., for each cell they report a 1 value with some probability q) incurs higher error for small ε (where more noise is added by the sampled clients). The gap is larger as the size of the histogram increases, since there are more chances for cells to incur more noise. Most intriguingly, the approach of adding Laplace noise, which is the gold standard in the centralized case, does not obtain the least error in this setting. Rather, the sample and threshold approach, which does not add explicit noise, but just removes small sampled counts, often achieves less error, particularly for small ε, where the magnitude of the Laplace noise is larger. This is more pronounced for larger histograms. The exception is for the Shakespeare data for larger histograms (Figures 1f and 1i). Here, the combination of skewed data, and smaller sample sizes for smaller ε, mean that



(a) Binomial, $B = 2^8$ (b) Geometric, $B = 2^8$ (c) Shakespeare, $B = 2^8$

(d) Binomial, $B = 2^{12}$ (e) Geometric, $B = 2^{12}$ (f) Shakespeare, $B = 2^{12}$

Figure 2: Top-k recall results on Binomial, Geometric and Shakespeare datasets

only a small fraction of the histogram buckets pass the threshold (often, fewer than 10% of buckets). Although the contribution of the buckets to the distribution is small, this means that while the federated approach sample and threshold improves over shuffling, it does not reach the accuracy of centralized noise addition when there are many infrequent items.

Last, we note that the magnitude of the error decreases as the histogram size increases. This is in part since the magnitude of the bucket frequencies decreases, and we are showing the (mean) error per bucket. As a sanity test, we also computed accuracy of the trivial approach of reporting zero for each bucket. The error for this approach falls above the range of each graph plotted, giving reassurance that we are achieving non-trivial accuracy for the histogram problem.

Recall results. To better understand the ability of the different approaches to capture the high counts (as needed for finding heavy hitters), we measure the recall of the top-k items, for $k = B/10$. That is, we test whether the histogram correctly reports the (true) top-10% of items among the 10% of largest items recovered. Figure 2 shows the results across different datasets. For moderate sized histograms ($B = 2^8$), the sample and threshold approach achieves close to perfect recall for all datasets. Other methods are comparable, but weaker for small ε. Again, it is the Shakespeare data for a larger histogram that presents the greatest challenge (in Figure 2f). Here, the same issue as above affects sample and threshold: a large fraction of small frequencies mean that these do not meet the threshold for the sample size. Accepting a larger ε, or working over a larger population to obtain a larger sample size would be needed to improve the recall. However, it could be argued that items missed are not very significant: already at $\varepsilon = 0.6$, the threshold applied means that only items with frequency less than 0.1% are likely to be dropped.

6 Related work

Histograms. Due to their broad applications, histograms are one of the most heavily studied tasks in differential privacy (DP). One of the first DP results is that a private histogram can be created by adding independent Laplace noise to each entry of the exact histogram (Dwork, 2006; Dwork and Roth, 2014). For multi-dimensional data, histograms of low-degree marginal distributions can be created via noise addition to the Hadamard transform of the data (Barak et al., 2007). These results assume a given set of histogram bucket boundaries; Xu et al. (2012) considered choosing bucket boundaries privately to minimize squared error. The histogram problem has also been heavily

studied in the local model of DP, where each individual adds noise to their input independently. Here, histograms are often implemented via 'frequency oracles', and used to identify frequent items from the input (Bassily and Smith, 2015). Optimized constructions make use of hashing (Wang et al., 2017) and Hadamard transforms (Acharya et al., 2019) to minimize the variance of the estimate. More recently, results are shown in the shuffle model, where messages from individuals are anonymized by a "shuffler", so the analyst sees only the multiset of messages received without attribution (Erlingsson et al., 2020). Under shuffling, for a fixed privacy level ε, accuracy bounds closer to the central case are achievable via the introduction of small amounts of random noise from each participant (Balcer and Cheu, 2020; Li et al., 2020).

Heavy hitters. The problem of finding the most frequent items from a collection is a core analytics task that supports a range of objectives, from simple popularity charts, to instantiating complex language models. Due to the sensitivity of data used within these applications, it is necessary to apply strong privacy protections to the data. There have been multiple efforts to address this problem in the Local DP setting (Bassily and Smith, 2015; Wang et al., 2017, 2020; Erlingsson et al., 2014; Apple, 2017; Bassily et al., 2020) and shuffle model (Ghazi et al., 2021). The closest work to ours is recent work on Federated Heavy Hitters discovery (Zhu et al., 2020), which describes an (ε, δ)-DP algorithm to collect information from a set of distributed clients, who each hold a (private) item. We can treat these items as strings of characters over a fixed alphabet. The algorithm proceeds in a series of L rounds to build up a trie describing the frequent items among the client population. In each round, the server contacts a random sample of m clients, and shares the current trie with them. Each client replies if its item extends the trie, and if so the client "votes" for the prefix that its item extends, along with the next character. The server receives these votes, and tallies them. Popular prefixes are added to the trie, and are candidates for further extension in the next round. The procedure stops after the trie has been built out to L levels, or if the trie cannot be extended beyond a certain level.

Quantiles and range queries. The quantiles of a distribution give a compact description of its (one-dimensional) CDF, generalizing the median. The problem has also been studied in the central, local and shuffled models. Many solutions first solve range queries, then reduce quantile queries to range queries. Xiao et al. (2010) propose using the Haar wavelet transform with noise, while Qardaji et al. (2013) use hierarchical histograms. Cormode et al. (2019) compare both methods in the local setting and observe similar levels of accuracy. In the shuffle model, quantiles are addressed via frequency histograms in the work of Ghazi et al. (2021).

Sampling and DP. It is well-known that sampling can be used to amplify the guarantees of differential privacy when combined with a DP mechanism on the sample: Li et al. (2012) combine sampling with k-anonymization to achieve a DP guarantee, Balle et al. (2020) show results for Poisson sampling, and fixed-size sampling with and without replacement, while Imola and Chaudhuri (2021) study privacy amplification when sampling according to differentially private parameters. By contrast, we consider mechanisms where sampling in isolation (with a threshold) provides the DP guarantee directly. This idea is inspired by Zhu et al. (2020), which materializes a set of items based on sampling and thresholding. The key advance in our work is to show that we can output the sampled frequencies as well as the sampled items, and hence produce private histograms. Our work complements other efforts in the federated setting to achieve privacy guarantees with a restricted set of operations—for instance, Kairouz et al. (2021) seek to perform federated learning via noise addition *without* sampling.

7 Concluding Remarks

In this paper, we have shown how the sample-and-threshold approach can be applied to the fundamental problem of private histogram computation, and related tasks like heavy hitter and quantile estimation. The key technical insight is that sampling a large enough number of indistinguishable examples introduces sufficient uncertainty to meet the differential privacy guarantee. As with other works on private histograms, we assume that the bucket boundaries of the histogram are given. Adaptive division of the histogram buckets is possible, as seen in the TrieHH++ protocol. Nevertheless, this approach can give poor results in extreme cases, such as when the bulk of the data resides in a very small fraction of the input domain.

It is natural to consider what other computations might benefit from this sample-and-threshold approach. Direct application of the technique makes sense when many users hold copies of the same value. Hence, it is not well-suited to questions like finding sums and means of general distributions, unless we additionally apply some rounding and noise addition to input values first. The approach may be of value for more complex computations, such as clustering or outlier removal, where dropping rare items is a benefit, or tasks where we seek to discover descriptions of patterns in the data that have large support, such as frequent itemsets.

References

Acharya, J., Sun, Z., and Zhang, H. (2019). Hadamard response: Estimating distributions privately, efficiently, and with little communication. In *The 22nd International Conference on Artificial Intelligence and Statistics*, volume 89 of *Proceedings of Machine Learning Research*, pages 1120–1129. PMLR.

Apple (2017). Apple differential privacy technical overview. `https://www.apple.com/privacy/docs/Differential_Privacy_Overview.pdf`, last accessed 19/07/21.

Balcer, V. and Cheu, A. (2020). Separating local & shuffled differential privacy via histograms. In *1st Conference on Information-Theoretic Cryptography, ITC 2020, June 17-19, 2020, Boston, MA, USA*, volume 163 of *LIPIcs*, pages 1:1–1:14. Schloss Dagstuhl - Leibniz-Zentrum für Informatik.

Balle, B., Barthe, G., and Gaboardi, M. (2020). Privacy profiles and amplification by subsampling. *J. Priv. Confidentiality*, 10(1).

Barak, B., Chaudhuri, K., Dwork, C., Kale, S., McSherry, F., and Talwar, K. (2007). Privacy, accuracy, and consistency too: a holistic solution to contingency table release. In *Proceedings of the Twenty-Sixth ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems*, pages 273–282. ACM.

Bassily, R., Nissim, K., Stemmer, U., and Thakurta, A. (2020). Practical locally private heavy hitters. *J. Mach. Learn. Res.*, 21:16:1–16:42.

Bassily, R. and Smith, A. D. (2015). Local, private, efficient protocols for succinct histograms. In *Proceedings of the Forty-Seventh Annual ACM on Symposium on Theory of Computing*, pages 127–135. ACM.

Bonawitz, K. A., Eichner, H., Grieskamp, W., Huba, D., Ingerman, A., Ivanov, V., Kiddon, C., Konečný, J., Mazzocchi, S., McMahan, B., Overveldt, T. V., Petrou, D., Ramage, D., and Roselander, J. (2019). Towards federated learning at scale: System design. In *Proceedings of Machine Learning and Systems*. mlsys.org.

Cormode, G., Kulkarni, T., and Srivastava, D. (2019). Answering range queries under local differential privacy. *Proc. VLDB Endow.*, 12(10):1126–1138.

Dwork, C. (2006). Differential privacy. In *Automata, Languages and Programming, 33rd International Colloquium, ICALP 2006, Venice, Italy, July 10-14, 2006, Proceedings, Part II*, volume 4052 of *Lecture Notes in Computer Science*, pages 1–12. Springer.

Dwork, C. and Roth, A. (2014). The algorithmic foundations of differential privacy. *Found. Trends Theor. Comput. Sci.*, 9(3-4):211–407.

Erlingsson, Ú., Feldman, V., Mironov, I., Raghunathan, A., Song, S., Talwar, K., and Thakurta, A. (2020). Encode, shuffle, analyze privacy revisited: Formalizations and empirical evaluation. *CoRR*, abs/2001.03618.

Erlingsson, Ú., Pihur, V., and Korolova, A. (2014). RAPPOR: randomized aggregatable privacy-preserving ordinal response. In Ahn, G., Yung, M., and Li, N., editors, *Proceedings of the 2014 ACM SIGSAC Conference on Computer and Communications Security*, pages 1054–1067. ACM.

Ghazi, B., Golowich, N., Kumar, R., Pagh, R., and Velingker, A. (2021). On the power of multiple anonymous messages: Frequency estimation and selection in the shuffle model of differential privacy. In *Advances in Cryptology - EUROCRYPT*, volume 12698 of *Lecture Notes in Computer Science*, pages 463–488. Springer.

Imola, J. and Chaudhuri, K. (2021). Privacy amplification via bernoulli sampling. *CoRR*, abs/2105.10594.

Kairouz, P., McMahan, B., Song, S., Thakkar, O., Thakurta, A., and Xu, Z. (2021). Practical and private (deep) learning without sampling or shuffling. In Meila, M. and Zhang, T., editors, *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pages 5213–5225. PMLR.

Lane, D. (2003). *Introduction to Statistics*.

Li, N., Qardaji, W., and Su, D. (2012). On sampling, anonymization, and differential privacy or, k-anonymization meets differential privacy. In *Proceedings of the 7th ACM Symposium on Information, Computer and Communications Security*, ASIACCS '12, page 32–33.

Li, X., Liu, W., Chen, Z., Huang, K., Qin, Z., Zhang, L., and Ren, K. (2020). DUMP: A dummy-point-based framework for histogram estimation in shuffle model. *CoRR*, abs/2009.13738.

McKenna, R., Miklau, G., and Sheldon, D. (2021). Winning the NIST contest: A scalable and general approach to differentially private synthetic data. *CoRR*, abs/2109.04978.

Qardaji, W. H., Yang, W., and Li, N. (2013). Understanding hierarchical methods for differentially private histograms. *Proc. VLDB Endow.*, 6(14):1954–1965.

Samarati, P. and Sweeney, L. (1998). Protecting privacy when disclosing information: k-anonymity and its enforcement through generalization and suppression. Technical report, Harvard Data Privacy Lab.

Wang, T., Blocki, J., Li, N., and Jha, S. (2017). Locally differentially private protocols for frequency es-

timation. In Kirda, E. and Ristenpart, T., editors, *26th USENIX Security Symposium, USENIX Security*, pages 729–745. USENIX Association.

Wang, T., Lopuhaä-Zwakenberg, M., Li, Z., Skoric, B., and Li, N. (2020). Locally differentially private frequency estimation with consistency. In *27th Annual Network and Distributed System Security Symposium, NDSS 2020, San Diego, California, USA, February 23-26, 2020*. The Internet Society.

Xiao, X., Wang, G., and Gehrke, J. (2010). Differential privacy via wavelet transforms. In *Proceedings of the 26th International Conference on Data Engineering, ICDE*, pages 225–236. IEEE Computer Society.

Xu, J., Zhang, Z., Xiao, X., Yang, Y., and Yu, G. (2012). Differentially private histogram publication. In *IEEE 28th International Conference on Data Engineering (ICDE 2012), Washington, DC, USA (Arlington, Virginia), 1-5 April, 2012*, pages 32–43. IEEE Computer Society.

Zhu, W., Kairouz, P., McMahan, B., Sun, H., and Li, W. (2020). Federated heavy hitters discovery with differential privacy. In *The 23rd International Conference on Artificial Intelligence and Statistics, AISTATS*, volume 108 of *Proceedings of Machine Learning Research*, pages 3837–3847. PMLR.

A Omitted technical material

Proof of Lemma 1. Given an item that occurs k times in the input, each occurrence has probability $p_s = m/n$ of being picked. The expected number of sampled occurrences is then $kp_s = km/n$.

Let X denote the random variable that counts the number of successes (times the prefix is picked) out of the k trials, so $\mathsf{E}[X] = km/n$. Then, X is a sum of k Bernoulli random variables with parameter p_s. We do a case split on p_s:

Case: $p_s \leq 1/k$**.** If $p_s \leq 1/k$, we apply an (additive) Chernoff-Hoeffding bound to the mean of the k trials:

$$\begin{aligned}\Pr[X \geq \tau] &= \Pr\left[\frac{1}{k}X - \frac{1}{k}\mathsf{E}[X] \geq (\tau p_s - p_s)\right] \\ &\leq \exp\left(-D\left(\frac{\tau}{k}\middle\|\frac{1}{k}\right)k\right).\end{aligned}$$

Here, $D(p\|q)$ denotes the K-L divergence (relative entropy) between the (Bernoulli) distributions with parameters p and q. We have

$$\begin{aligned}-D(p\|q)k &= -\tau\ln\left(\frac{\tau}{k}\cdot\frac{k}{1}\right) - (k-\tau)\ln\left(\frac{k-\tau}{k}\cdot\frac{k}{k-1}\right) \\ &= -\tau\ln\tau - (k-\tau)\ln\left(1-\frac{\tau-1}{k-1}\right) \\ &= -\tau\ln\tau + (k-\tau)\ln\left(\frac{k-1}{k-\tau}\right) \\ &= -\tau\ln\tau + (k-\tau)\ln\left(1+\frac{\tau-1}{k-\tau}\right) \\ &\leq -\tau\ln\tau + \tau - 1\end{aligned}$$

$$\text{Hence,} \qquad \Pr[X \geq \tau] \leq \exp(-\tau\ln\tau + \tau - 1) \qquad (4)$$

For this case, to achieve a target error bound δ, we rearrange to obtain $\frac{\tau}{e}\ln\frac{\tau}{e} = \frac{1}{e}\ln(1/e\delta)$, and apply Lambert's W function. This gives $\frac{\tau}{e} = W(\frac{1}{e}\ln(1/e\delta)$, i.e., $\tau = eW(\frac{1}{e}\ln\frac{1}{e\delta})$. Note that this case corresponds to the scenario where we do not publish the counts, but only indicate which items occurred more than τ times in the sample.

Case: $p_s > 1/k$**.** If $p_s > 1/k$, we apply a (multiplicative) Chernoff bound:

$$\begin{aligned}\Pr[X \geq \tau\mathsf{E}[X]] &\leq \exp(-(\tau-1)^2\mathsf{E}[X]/(1+\tau)) \\ &= \exp(-(\tau-1)^2 kp_s/(1+\tau)) \\ &\leq \exp(-(\tau-1)^2/(1+\tau))\end{aligned}$$

In this case, to achieve a target error bound δ, we can pick $\tau = 3+\ln(1/\delta)$, and obtain

$$\exp(-(2+\ln 1/\delta)^2/(4+\ln 1/\delta)) < \exp(-\ln(1/\delta)) = \delta.$$

The second case is stricter for all $\tau > 1$, so we will use this setting of τ in what follows. □

Proof of Lemma 2. The case to focus on is when input D has one extra copy of a particular item compared to D', at some intermediate stage of the algorithm. For notation, we will write $S_k(n,s,v)$ to denote the number of ways to succeed in collecting exactly v instances of the target item while picking s items out of n, when there are k total instances of the item. We can observe that there is a simple combinatorial expression for this quantity: we count the number of combinations where we pick a particular subset of size v from the k instances, and a particular subset of size $s-v$ from the remaining $n-k$ examples.

$$S_k(n,s,v) = \binom{k}{v}\binom{n-k}{s-v} \qquad (5)$$

Our goal is to bound the ratio of probabilities of seeing a count of v copies of the item in the output of D, who has $k+1$ copies of the item, and of D' who holds k copies. The probability that the sample size is exactly s is given by $P_s = p_s^s(1-p_s)^{n-s}$. For a given sample size s, the probability for D is $S_{k+1}(n,s,v)P_s$, and for D' it is $S_k(n,s,v)P_s$. Then this ratio of probabilities is given by

$$\begin{aligned}\frac{S_{k+1}(n,m,v)P_s}{S_k(n,m,v)P_s} &= \frac{\binom{k+1}{v}\binom{n-k-1}{m-v}}{\binom{k}{v}\binom{n-k}{m-v}} = \frac{(k+1)(n-k-m-v)}{(n-k)(k+1-v)} \\ &= \left(1-\frac{m+v}{n-k}\right)\left(\frac{k+1}{k+1-v}\right) \leq \frac{k+1}{k+1-v}\end{aligned}$$

Then we can bound this ratio across all sample sizes as simply $\sum_{s=0}^{n}\frac{k+1}{k+1-v}P_s = \frac{k+1}{k+1-v}$. □

Proof of Lemma 3. Observe that, with high probability, the size of the (Poisson) sample will be close to expected value of m. In particular, by a Chernoff bound, the probability that the sample size is more than $c\sqrt{m}$ larger than m is

$$\Pr[s > (1+c/\sqrt{m})m] \leq \exp(-c^2/3).$$

Hence, for c a suitable constant (say, 10), this probability is negligibly small. To realize this sampling, we contact a fixed size number of clients $s = m + c\sqrt{m}$, and then have each client perform a Bernoulli test on whether to participate: with probability m/s, it participates, otherwise it abstains. An abstaining client can, for example, vote for a unique element (e.g., an item based on a hash of its identifier), and so be automatically discounted from the protocol, without revealing this information to the aggregator. □

Proof of Lemma 4. For an item with (absolute) frequency W out of the n input items, it is reported if the number of sampled occurrences exceeds τ. Similar to the analysis above, we can apply a Chernoff-Hoeffding bound to the random variable X that counts the number of occurrences of the item. Now, the probability of each sample picking the item is W/n, and the expected number in the sample is $Wm/n > \tau$. For convenience, we will write $w = Wm/n$ for this expectation. We have that[2]

$$\begin{aligned}\Pr[X \leq \tau] &= \Pr\left[X \leq \frac{\tau}{w} w\right] \\ &= \Pr\left[X \leq \left(1 - \frac{w-\tau}{w}\right) \mathsf{E}[X]\right] \\ &= \exp\left(-\frac{(w-\tau)^2}{2w}\right)\end{aligned}$$

□

Proof of Lemma 5. Applying the same Chernoff-Hoeffding bound as above, we have for $\gamma < 1$,

$$\Pr[|X - \mu| > \gamma\mu] = 2\exp(-\gamma^2\mu/3) = \beta$$

Rearranging, we obtain $\mu = \frac{3}{\gamma^2}\ln(1/2\beta)$. Suppose we aim to find all items whose frequency is at least ϕ, and estimate their frequency with relative error at most γ. Then we have $\mu = \phi m \geq \phi\frac{\varepsilon n}{\tau} = \frac{3}{\gamma^2}\ln(1/2\beta)$. □

Proof of Lemma 6. In more detail, we can view the protocol as publishing a histogram at each level, where the granularity of the cells is refined in each round. The protocol enforces that if a prefix is not included at a particular level, then none of its extensions are published in any subsequent level. However, we can view this as "post-processing", and analyze the simpler algorithm that does not enforce this constraint. Applying Theorem 1, we have that each round satisfies $(\varepsilon_i, \delta_i)$-DP for some ε_i and δ_i.

Then we argue that the output of the full protocol is the L-fold composition of the mechanisms M_i. Assuming $\varepsilon_i = \varepsilon'$ and $\delta_i = \delta'$ for all i, then using basic composition, we obtain a bound of $(L\varepsilon', L\delta')$-differential privacy, leading to the result stated in the theorem claim. For $\varepsilon_i = \varepsilon' < 1$, we can also obtain a tighter bound, of $(L\varepsilon'^2 + \varepsilon'\sqrt{L\log 1/(\delta' L)}, 2L\delta')$ using advanced composition (Dwork and Roth, 2014). □

[2]Here we are sampling without replacement. However, the bounds for sampling with replacement are still valid here.

Proof of Lemma 7. The quantile query can be carried out by a binary search: we begin by creating a histogram with buckets $[0, \frac{1}{2}], [\frac{1}{2}, 1]$, and recursively try different split points $[0,t], [t,1]$ until we obtain a result with approximately a ϕ fraction of points in the first bucket, at which point we can report t as the ϕ-quantile. Provided ϕ is sufficiently larger than τ/m (and smaller than $1 - \tau/m$), then we are unlikely to hit any cases where a bucket count is removed. As a result, the error will primarily the error from sampling, which is $O(1/\sqrt{m})$ (Lane, 2003), plus the error from rounding, which is 2^{-h} if we perform h steps of binary search. That is, we find a result t such that there is a point in the range $[t - 2^{-h}, t + 2^{-h}] := t \pm 2^{-h}$ that dominates $\phi \pm O(1/\sqrt{m})$. The privacy guarantee is $\varepsilon = O(hm\ln(1/\delta)/n)$, from the composition of h queries. □