Damien Desfontaines*, James Voss, Bryant Gipson, and Chinmoy Mandayam

Differentially private partition selection

Abstract: Many data analysis operations can be expressed as a GROUP BY query on an unbounded set of partitions, followed by a per-partition aggregation. To make such a query differentially private, adding noise to each aggregation is not enough: we also need to make sure that the set of partitions released is also differentially private.

This problem is not new, and it was recently formally introduced as *differentially private set union* [14]. In this work, we continue this area of study, and focus on the common setting where each user is associated with a single partition. In this setting, we propose a simple, *optimal* differentially private mechanism that maximizes the number of released partitions. We discuss implementation considerations, as well as the possible extension of this approach to the setting where each user contributes to a fixed, small number of partitions.

Keywords: data privacy, differential privacy

DOI 10.2478/popets-2022-0017

Received 2021-05-31; revised 2021-09-15; accepted 2021-09-16.

1 Introduction

Suppose that a centralized service collects information on its users, and that an engineer wants to understand the prevalence of different device models among the users. They could run a SQL query similar to the following.

```
SELECT
  device_model,
  COUNT(UNIQUE user_id)
FROM database
GROUP BY device_model
```

Many common data analysis tasks follow a simple structure, similar to this example query: a `GROUP BY` operation that defines a set of *partitions* (here, device models), followed by one or several aggregations. To make such a query differentially private, it is not enough to add noise to each count. Indeed, in the example above, suppose that a device model is particularly rare, and that a single user is associated to this device model. The presence or absence of this user determines whether this partition appears in the output: even if the count is noisy, the differential privacy property is not satisfied. Thus, in addition to the counts, the *set of partitions present in the output* must also be differentially private. There are two main ways of ensuring this property.

A first option is to determine the set of output partitions *in advance*, without looking at the private data. In this case, even if some of the partitions do not appear in the private data, they must still be returned, with noise added to the zero value. Conversely, if the private data has partitions that do not appear in the predefined list, they must be dropped from the output. This option is feasible when grouping users by some fixed categories, or if partitions can only take a small number of predefined values.

However, this is not always the case. Text-based partitions like search queries or user agents might take arbitrary values, and often cannot be known without access to the private dataset. Furthermore, when building a generic DP engine, usability is paramount, and requiring users to annotate their dataset with all possible values that can be taken by a given field is a significant burden. This makes a second option attractive: generating the list of partitions from the private data itself, in a differentially private way. This problem was formally introduced in [14] as *differentially private set union*. Each user is associated with one or several partitions, and the goal is to release as many partitions as possible while making sure that the output is differentially private.

In [14], the main motivation to study this set union primitive is natural language processing: the discovery of words and n-grams is essential to these tasks, and can be modeled as a set union problem. In this context, each user can contribute to many different partitions. In the context of data analysis queries, however, it is common that each contributes only to a small number of partitions, often just *one*. This happens when the partition is a characteristic of each user, for example demographic attributes or the answer to a survey. In

***Corresponding Author: Damien Desfontaines:** Tumult Labs, damien@desfontain.es
James Voss: Google, jrvoss@google.com
Bryant Gipson: Google, bryantgipson@google.com
Chinmoy Mandayam: Google, cvm@google.com

the above SQL query example, if the user ID is a device identifier, each contributes to at most one device model.

In this work, we focus on this particular single-contribution case, and provide an *optimal* partition selection strategy for this special case. More specifically, we show that there is a fundamental upper bound on the probability of returning a partition associated with k users, and present an algorithm that achieves this bound.

This paper is structured as follows. After discussing prior work in more detail and introducing definitions, we present a partition selection mechanism for the case where each user contributes to one partition, and prove its optimality. We then discuss possible extensions to cases where each user contributes to multiple partitions as well as implementation considerations.

1.1 Prior work

In this section, we review existing literature on the problem of releasing a set of partitions from an unbounded set while satisfying differential privacy. This did not get specific attention until [14], but the first algorithm that solves it was introduced in [21], for the specific context of privately releasing search queries. This algorithm works as follows: build a histogram of all partitions, count unique users associated with each partition, add Laplace noise to each count, and keep only the partitions whose counts are above a fixed threshold. The scale of the noise and the value of the threshold determine ε and δ. This method is simple and natural; it was adapted to work in more general-purpose query engines in [26].

In [14], the authors focus on the more general problem of differentially private set union. The main use case for this work is word and n-gram discovery in Natural Language Processing: data used in training models must not leak private information about individuals. In this context, each user potentially contributes to many elements; the *sensitivity* of the mechanism can be high. The authors propose two strategies applicable in this context. First, they use a *weighted* histogram so that if a user contributes to fewer elements than the maximum sensitivity, these elements can add more weight to the histogram count. Second, they introduce *policies* that determine which elements to add to the histogram depending on which histogram counts are already above the threshold. These strategies obtain significant utility improvements over the simple Laplace-based strategy.

In this work, in contrast to [14], we focus on the *low-sensitivity* use case: each user contributes to exactly one element. This different setting is common in data analysis: when the GROUP BY key partitions the set of users in distinct partitions, each user can only contribute to one element to the set union. Choosing the contributions of each user is therefore not relevant; the only question is to optimize the probability of releasing each element in the final result. For this specific problem, we introduce an optimal approach, which maximizes this probability.

Public partitions

When the domain of possible partitions is known in advance and considered public data, no partition selection is necessary. This assumption is typically made implicitly in existing work on histogram publication, either by assuming that the domain is known exactly and not too large [1, 6, 15, 27–30], or that the attributes are numeric and bounded [24]. In the former case, no partition selection is necessary; the strategy usually revolves around grouping known partitions together to limit the impact of the noise. In the latter case, the possible partitions are also indirectly known in advance (all possible intervals in a fixed numerical range), and the problem is to find which intervals to use to slice the data. With such pre-existing knowledge about the partitions, our approach does not provide any benefit.

Domain of fixed size

When the domain does not conform to one of the assumptions described above, the data domain might still be a subset of some large domain. For example, integer attributes are typically stored using 64 bits. Similarly, it is reasonable to assume that search queries or URLs are strings whose size is bounded by some large number.

We can use this fact to perform partition selection by adding noise to all possible partitions, including the ones that do not contain any data, and only return the ones that are above a given threshold. This process can be simulated in an efficient way, without actually enumerating all partitions [5]. Other methods might be possible; for example, one could imagine simulating the sparse vector technique [8] or one of its multiple-queries variants [9, 19, 23, 25] to ask the number of users in all possible partitions, while ignoring the privacy cost of answers below a threshold.

We are not aware of any work using these techniques for the specific problem of partition selection. We also postulate that they are likely to fail for extremely large domain sizes (like long strings); the technique in [5] out-

puts a number of false positive partitions linear in the domain size.

Differences with our approach

In this work, we focus on cases where all assumptions above fail because the domain of the data is unbounded or too large. As such, the only way to learn this domain is by looking at the private data, which must be done in a differentially private way. This assumption is particularly suited to building generic tooling, like general-purpose differentially private query engines [3, 18, 22, 26]. Indeed, to use such an engine, either all the domain of the input data must be enumerated in advance, or partition selection is necessary. But this requires the analyst or data owner to document the data domain for all input databases. This is a significant usability burden, which makes it difficult to scale the use of the query engine. This problem is the main motivator for our work.

1.2 Definitions

Differential privacy (DP) is a standard notion to quantify the privacy guarantees of statistical data. For the problem of differentially private set union, we use (ε, δ)-DP.

Definition 1 (Differential privacy [7])**.** *A randomized mechanism $\mathcal{M}$ is (ε, δ)-differentially private if for any two databases D and D', where D' can be obtained from D by either adding or removing one user, and for all sets S of possible outputs:*

$$\mathbb{P}\left[\mathcal{M}\left(D\right) \in S\right] \leq e^{\varepsilon}\mathbb{P}\left[\mathcal{M}\left(D'\right) \in S\right] + \delta.$$

Let us formalize the problem addressed in this work.

Definition 2 (Differentially private partition selection)**.** *Let U be a universe of* partitions*, possibly infinite. A* partition selection mechanism *is a mechanism $\mathcal{M}$ that takes a database D in which each user i contributes a subset $W_i \subset U$ of partitions, and outputs a subset $\mathcal{M}\left(D\right) \subseteq \cup_i W_i$.*

The problem of differentially private partition selection[1] *consists in finding a mechanism $\mathcal{M}$ that outputs as many partitions as possible while satisfying (ε, δ)-differential privacy.*

1 Also called *differentially private set union* [14].

In the main section of this paper, we assume that each user contributes to only one partition ($|W_i| = 1$ for all i). We first study the simplified problem of considering each partition independently. The only question then is: with which probability do we release this partition? And the strategy can simply be reduced to a function associating the number of users in a partition with the probability of keeping the partition. After finding an optimal primitive for this simpler problem, we show that it is actually optimal in a stronger sense, even among mechanisms that consider all partitions simultaneously.

Definition 3 (Partition selection primitive)**.** *A partition selection primitive is a function $\pi : \mathbb{N} \to [0, 1]$ such that $\pi(0) = 0$. The corresponding partition selection strategy consists in counting the number n of users in each partition, and releasing this partition with probability $\pi(n)$.*

We say that a partition selection primitive is (ε, δ)-differentially private if the corresponding partition selection strategy $\rho_\pi : \mathbb{N} \to \{\text{drop}, \text{keep}\}$, defined by:

$$\rho_\pi(n) = \begin{cases} \text{drop} & \text{with probability } 1 - \pi(n) \\ \text{keep} & \text{with probability } \pi(n) \end{cases}$$

is (ε, δ)-differentially private.

Note that partitions associated with no users are not present in the input data, so the probability of releasing them has to be 0: the definition requires $\pi(0) = 0$.

2 Main result

In this section, we define an (ε, δ)-DP partition selection primitive π_{opt} and prove that the corresponding partition selection strategy is optimal. In this context, optimal means that it maximizes the probability of releasing a partition with n users, for all n.

Definition 4 (Optimal partition selection primitive)**.** *A partition selection primitive π_{opt} is* optimal for (ε, δ)-DP *if it is (ε, δ)-DP, and if for all (ε, δ)-DP partition selection primitives π and all $n \in \mathbb{N}$:*

$$\pi(n) \leq \pi_{\text{opt}}\left(n\right).$$

We introduce our main result, then we prove it in two steps: we first prove that the optimal partition selection primitive can be obtained recursively, then derive the closed-form formula of our main result from the recurrence relation.

Theorem 1 (General solution for π_{opt})**.** *Let $\varepsilon > 0$ and $\delta \in (0, 1)$. Defining:*

$$n_1 = 1 + \left\lfloor \frac{1}{\varepsilon} \ln\left(\frac{e^\varepsilon + 2\delta - 1}{(e^\varepsilon + 1)\delta}\right)\right\rfloor,$$
$$n_2 = n_1 + \left\lfloor \frac{1}{\varepsilon} \ln\left(1 + \frac{e^\varepsilon - 1}{\delta}\left(1 - \pi_{\text{opt}}\left(n_1\right)\right)\right)\right\rfloor,$$

and $m = n - n_1$, the partition selection primitive π_{opt} defined by:

- $\pi_{\text{opt}}\left(n\right) = \frac{e^{n\varepsilon}-1}{e^\varepsilon - 1} \cdot \delta$ *if* $n \le n_1$,
- $\pi_{\text{opt}}\left(n\right) = \left(1 - e^{-m\varepsilon}\right)\left(1 + \frac{\delta}{e^\varepsilon - 1}\right) + e^{-m\varepsilon}\pi_{\text{opt}}\left(n_1\right)$ *if* $n > n_1$ *and* $n \le n_2$,
- 1 *otherwise*

is optimal for (ε, δ)-DP.

These formulas assume $\varepsilon > 0$ and $\delta > 0$. We also cover the special cases where $\varepsilon = 0$ or $\delta = 0$.

Theorem 2 (Special cases for π_{opt})**.**

1. *If $\delta = 0$, partition selection is impossible: the optimal partition selection primitive π_{opt} for $(\varepsilon, 0)$-DP is defined by $\pi_{\text{opt}}\left(n\right) = 0$ for all n.*
2. *If $\varepsilon = 0$, the optimal partition selection primitive π_{opt} for $(0, \delta)$-DP is defined by $\pi_{\text{opt}}\left(n\right) = \min\left(1, n\delta\right)$ for all n.*

2.1 Recursive construction

How do we construct a partition selection primitive π so that the partition is output with the highest possible probability under the constraint that π is (ε, δ)-DP? Using the definition of differential privacy, the following inequalities must hold for all $n \in \mathbb{N}$.

$$\pi(n+1) \le e^\varepsilon \pi(n) + \delta \tag{1}$$
$$\pi(n) \le e^\varepsilon \pi(n+1) + \delta \tag{2}$$
$$(1 - \pi(n+1)) \le e^\varepsilon (1 - \pi(n)) + \delta \tag{3}$$
$$(1 - \pi(n)) \le e^\varepsilon (1 - \pi(n+1)) + \delta. \tag{4}$$

These inequalities are not only necessary, but also *sufficient* for π to be DP. Thus, the optimal partition selection primitive can be constructed by recurrence, maximizing each value while still satisfying the inequalities above. As we will show, only inequalities (1) and (4) above need be included in the recurrence relationship. The latter can be rearranged as:

$$\pi_{\text{opt}}\left(n+1\right) \le 1 - e^{-\varepsilon}(1 - \pi_{\text{opt}}\left(n\right) - \delta)$$

which leads to the following recursive formulation for π_{opt}.

Lemma 1 (Recursive solution for π_{opt})**.** *Given $\delta \in [0, 1]$ and $\varepsilon \ge 0$, π_{opt} satisfies the following recurrence relationship: $\pi_{\text{opt}}\left(0\right) = 0$, and for all $n \ge 0$:*

$$\begin{aligned}\pi_{\text{opt}}\left(n+1\right) = \min(&\\ &e^\varepsilon \pi_{\text{opt}}\left(n\right) + \delta,\\ &1 - e^{-\varepsilon}(1 - \pi_{\text{opt}}\left(n\right) - \delta),\\ &1)\end{aligned} \tag{5}$$

Proof. Let π_0 be defined by recurrence as above; we will prove that $\pi_0 = \pi_{\text{opt}}$.

First, let us show that π_0 is monotonic. Fix $n \in \mathbb{N}$. It suffices to show for each argument of the min function in (5) is larger than $\pi_0\left(n\right)$.

First argument. Since $\varepsilon \ge 0$ implies $e^\varepsilon \ge 1$ and $\delta \ge 0$, we trivially have $e^\varepsilon \pi_0\left(n\right) + \delta \ge \pi_0\left(n\right)$.

Second argument. We have:

$$\begin{aligned}1 - e^{-\varepsilon}(1 - \pi_0\left(n\right) - \delta) &= 1 - e^{-\varepsilon}(1 - \pi_0\left(n\right)) + e^{-\varepsilon}\delta\\ &\ge 1 - (1 - \pi_0\left(n\right))\\ &= \pi_0\left(n\right)\end{aligned}$$

using that $1 - \pi_0\left(n\right) \ge 0$ since $\pi_0\left(n\right) \le 1$ by (5).

Third argument. This is immediate given (5) and the fact that $\pi_0\left(0\right) = 0$.

It follows that $\pi_0\left(n+1\right) \ge \pi_0\left(n\right)$.

Because π_0 is monotonic, it immediately satisfies inequalities (2) and (3), and inequalities (1) and (4) are satisfied by definition.

Since π_0 satisfies all four inequalities above, it is (ε, δ)-DP. Its optimality follows immediately by recurrence: for each $n+1$, if $\pi(n+1) > \pi_{\text{opt}}\left(n+1\right)$, it cannot be (ε, δ)-DP, as one of the inequalities above is not satisfied: π_0 is the fastest-growing DP partition selection strategy, and therefore equal to π_{opt}. □

Note that the special cases for π_{opt} in Theorem 2 can be immediately derived from Lemma 1.

2.2 Derivation of the closed-form solution

Let us now show that the closed-form solution of Theorem 1 can be derived from the recursive solution in 1. First, we show that there is a crossover point n_1, below which only the first term of the recurrence relation matters, and after which only the second term matters (until $\pi_{\text{opt}}\left(n\right)$ reaches 1).

Lemma 2. *Assume $\varepsilon > 0$ and $\delta > 0$. There are crossover points $n_1, n_2 \in \mathbb{N}$ such that $0 < n_1 \le n_2$ and*

- $\pi_{\text{opt}}\left(n\right) = 0$ *if* $n = 0$,

- $\pi_{\text{opt}}(n) = \pi_{\text{opt}}(n-1)\, e^{\varepsilon} + \delta$ *if* $n > 0$ *and* $n \leq n_1$,
- $\pi_{\text{opt}}(n) = 1 - e^{-\varepsilon}(1 - \pi_{\text{opt}}(n-1) - \delta)$ *if* $n > n_1$ *and* $n \leq n_2$,
- $\pi_{\text{opt}}(n) = 1$ *otherwise.*

Proof. We consider the arguments in the min statement in (5), substituting x for $\pi_{\text{opt}}(n)$:

$$\begin{aligned}\alpha_1(x) &= e^{\varepsilon}x + \delta\\ \alpha_2(x) &= 1 - e^{-\varepsilon}(1 - x - \delta)\\ \alpha_3(x) &= 1\end{aligned}$$

This substitution allows us to work directly in the space of probabilities instead of restricting ourselves to the sequence $(\pi_{\text{opt}}(n))_{n=0}^{\infty}$. Taking the first derivative of these functions yields:

$$\begin{aligned}\alpha_1'(x) &= e^{\varepsilon}\\ \alpha_2'(x) &= e^{-\varepsilon}\\ \alpha_3'(x) &= 0\end{aligned}$$

Since the derivative of $\alpha_1(x) - \alpha_2(x)$ is $e^{\varepsilon} - e^{-\varepsilon} > 0$, there exists at most one crossover point x_1 such that $\alpha_1(x) < \alpha_2(x)$ for all $x < x_1$, $\alpha_2(x_1) = \alpha_1(x_1)$, and $\alpha_1(x) > \alpha_2(x)$ for all $x > x_1$. Setting $\alpha_1(x) = \alpha_2(x)$ and solving for x yields:

$$e^{\varepsilon}x + \delta = 1 - e^{-\varepsilon}(1 - x - \delta)$$

which leads to:

$$e^{\varepsilon}x - e^{-\varepsilon}x = 1 - \delta - e^{-\varepsilon}(1 - x - \delta)$$

and finally:

$$x_1 = (1-\delta)\cdot\frac{1 - e^{-\varepsilon}}{e^{\varepsilon} - e^{-\varepsilon}}.$$

Since the derivative of $\alpha_2(x) - \alpha_3(x)$ is $e^{-\varepsilon} > 0$, there exists at most one crossover point x_2 such that $\alpha_2(x) < \alpha_3(x)$ for all $x < x_2$, $\alpha_2(x_2) = \alpha_3(x_2)$, and $\alpha_2(x) > \alpha_3(x)$ for all $x > x_2$. Setting $\alpha_2(x) = \alpha_3(x)$ and solving for x yields:

$$x_2 = 1 - \delta.$$

From the formulas for x_1 and x_2, it is immediate that $0 < x_1 < x_2 < 1$. As such, the interval $[0, 1]$ can be divided into three non-empty intervals:

1. On $[0, x_1]$, $\alpha_1(x)$ is the active argument of $\min(\alpha_1(x), \alpha_2(x), \alpha_3(x))$.
2. On $[x_1, x_2]$, $\alpha_2(x)$ is the active argument of $\min(\alpha_1(x), \alpha_2(x), \alpha_3(x))$.
3. On $[x_2, 1]$, $\alpha_3(x)$ is the active argument of $\min(\alpha_1(x), \alpha_2(x), \alpha_3(x))$.

The existence of the crossover points is not enough to prove the lemma: we must also show that these points are reached in a finite number of steps. For all $n \geq 1$ such that $\pi_{\text{opt}}(n) \neq 1$, we have:

$$\begin{aligned}&\pi_{\text{opt}}(n) - \pi_{\text{opt}}(n-1)\\ &\quad= \min(\\ &\qquad e^{\varepsilon}\pi_{\text{opt}}(n-1) + \delta,\\ &\qquad 1 - e^{-\varepsilon}(1 - \pi_{\text{opt}}(n-1) - \delta)\\ &\quad) - \pi_{\text{opt}}(n-1)\\ &\quad\geq \min\left(\delta, \left(1 - e^{-\varepsilon}\right)(1 - \pi_{\text{opt}}(n-1)) + e^{-\varepsilon}\delta\right)\\ &\quad\geq e^{-\varepsilon}\delta.\end{aligned}$$

Since $\pi_{\text{opt}}(n) - \pi_{\text{opt}}(n-1)$ is bounded from below by a strictly positive constant $e^{-\varepsilon}\delta$, the sequence achieves the maximal probability 1 for finite n. □

This allows us to derive the closed-form solution for $n < n_1$ and for $n_1 \leq n < n_2$ stated in Theorem 1.

Lemma 3. *Assume* $\varepsilon > 0$ *and* $\delta \leq 0$. *If* $n \leq n_1$, *then* $\pi_{\text{opt}}(n) = \frac{e^{n\varepsilon}-1}{e^{\varepsilon}-1}\cdot\delta$. *If* $n_1 \leq n < n_2$, *then denoting* $m = n - n_1$*:*

$$\pi_{\text{opt}}(n) = \left(1 - e^{-m\varepsilon}\right)\left(1 + \frac{\delta}{e^{\varepsilon} - 1}\right) + e^{-m\varepsilon}\pi_{\text{opt}}(n_1).$$

Proof. For $n < n_1$, expanding the recurrence relation yields:

$$\begin{aligned}\pi_{\text{opt}}(n) &= \pi_{\text{opt}}(n-1)\, e^{\varepsilon} + \delta\\ &= \delta\sum_{k=0}^{n-1} e^{k\varepsilon}\\ &= \frac{e^{n\varepsilon} - 1}{e^{\varepsilon} - 1}\cdot\delta.\end{aligned}$$

For $n_1 \leq n < n_2$, denoting $m = n - n_1$, expanding the recurrence relation yields:

$$\begin{aligned}\pi_{\text{opt}}(n) &= 1 - e^{-\varepsilon}(1 - \pi_{\text{opt}}(n-1) - \delta)\\ &= \left(1 - e^{-\varepsilon} + \delta e^{-\varepsilon}\right)\sum_{k=0}^{m-1} e^{-k\varepsilon} + e^{-m\varepsilon}\pi_{\text{opt}}(n_1)\\ &= \left(1 - e^{-\varepsilon} + \delta e^{-\varepsilon}\right)\frac{1 - e^{-m\varepsilon}}{1 - e^{-\varepsilon}} + e^{-m\varepsilon}\pi_{\text{opt}}(n_1)\\ &= \left(1 - e^{-m\varepsilon}\right)\left(1 + \frac{\delta}{e^{\varepsilon} - 1}\right) + e^{-m\varepsilon}\pi_{\text{opt}}(n_1).\end{aligned}$$

□

We can now find a closed-form solution for n_1 and for n_2.

Lemma 4. *The first crossover point n_1 is:*

$$n_1 = 1 + \left\lfloor \frac{1}{\varepsilon} \ln\left(\frac{e^\varepsilon + 2\delta - 1}{\delta(e^\varepsilon + 1)} \right) \right\rfloor \tag{6}$$

Proof. Using the formula for x_1 in the proof of Lemma 2, we see that $\pi_{\text{opt}}(n-1) \leq x_1$ whenever:

$$\frac{e^{(n-1)\varepsilon} - 1}{e^\varepsilon - 1} \cdot \delta \leq \frac{1-\delta}{e^\varepsilon + 1}.$$

Rearranging terms, we can rewrite this inequality as:

$$\begin{aligned} n &\leq 1 + \frac{1}{\varepsilon} \ln\left[\frac{(1-\delta)(e^\varepsilon - 1)}{\delta(e^\varepsilon + 1)} + 1 \right] \\ &= 1 + \frac{1}{\varepsilon} \ln\left[\frac{(1-\delta)(e^\varepsilon - 1) + \delta(e^\varepsilon + 1)}{\delta(e^\varepsilon + 1)} \right] \\ &= 1 + \frac{1}{\varepsilon} \ln\left[\frac{e^\varepsilon + 2\delta - 1}{\delta(e^\varepsilon + 1)} \right]. \end{aligned}$$

Since n is an integer, the supremum value defining n_1 is achieved by taking the floor of the right-hand side of this inequality, which concludes the proof. □

Lemma 5. *The second crossover point n_2 is:*

$$n_2 = n_1 + \left\lfloor \frac{1}{\varepsilon} \ln\left(1 + \frac{e^\varepsilon - 1}{\delta}\left(1 - \pi_{\text{opt}}(n_1)\right) \right) \right\rfloor$$

Proof. We want to find the maximal m such that:

$$\left(1 - e^{-m\varepsilon}\right)\left(1 + \frac{\delta}{e^\varepsilon - 1}\right) + e^{-m\varepsilon}\pi_{\text{opt}}(n_1) \leq 1.$$

We can rewrite this condition into:

$$-e^{-m\varepsilon}\left(1 + \frac{\delta}{e^\varepsilon - 1} - \pi_{\text{opt}}(n_1)\right) \leq \frac{-\delta}{e^\varepsilon - 1}$$

which leads to:

$$\begin{aligned} e^{m\varepsilon} &\leq \frac{e^\varepsilon - 1}{\delta}\left(1 + \frac{\delta}{e^\varepsilon - 1} - \pi_{\text{opt}}(n_1)\right) \\ &\leq 1 + \frac{e^\varepsilon - 1}{\delta}\left(1 - \pi_{\text{opt}}(n_1)\right) \end{aligned}$$

and finally:

$$m \leq \frac{1}{\varepsilon} \ln\left(1 + \frac{e^\varepsilon - 1}{\delta}\left(1 - \pi_{\text{opt}}(n_1)\right) \right)$$

since m must be an integer, we take the floor of the right-hand side of this inequality to obtain the result. □

2.3 More generic optimality result

Theorem 1 provides an optimal partition selection primitive in the sense of Definition 4: a mechanism using this primitive on each partition separately is optimal among the class of mechanisms that consider every partition separately. The mechanism cannot use auxiliary knowledge about relationships within partitions, and the decision for a given partition cannot depend on the data in other partitions. Can we extend the optimality result to a larger class of algorithms, that take the full list of partitions as input?

We can answer that question in the affirmative, in the particular case where each user contributions a single partition. First, we need to define what optimality means in a more general context. Recall that a partition selection mechanism takes a database D in which each user contributes a subset $W_i \subset U$ of partitions, and outputs a subset $\mathcal{M}(D) \subseteq \cup_i W_i$. The goal is to output as many partitions as possible, which we capture by maximizing the expected value of the number of output partitions.

Definition 5 (Optimal partition selection mechanism)**.** *A partition selection mechanism $\mathcal{M}$ is* optimal for (ε, δ)-DP and sensitivity κ *if it is (ε, δ)-DP, and if for all (ε, δ)-DP partition selection mechanisms $\mathcal{M}'$, and all databases D in which each user contributes at most κ partitions:*

$$\mathbb{E}\left[\left|\mathcal{M}'(D)\right|\right] \leq \mathbb{E}\left[\left|\mathcal{M}(D)\right|\right].$$

We can now prove our more generic optimality result.

Theorem 3. *Let $\mathcal{M}_{\text{opt}}$ be the partition selection mechanism that, on input D, returns each partition k with probability $\pi_{\text{opt}}(|\{i \mid W_i = \{k\}\}|)$. Then $\mathcal{M}_{\text{opt}}$ is optimal for (ε, δ)-DP and sensitivity* 1.

Proof. Let $\mathcal{M}$ be a partition selection mechanism. Since we assume that every user contributes to at most one partition ($\kappa = 1$), it is equivalent to consider the input of $\mathcal{M}$ to be the histogram $(n_i)_{i \in U}$, where n_i is the number of users associated to partition i. Of course, if $n_k = 0$ for some k, then k must not be in the output set.

Now, for a given partition k, fix all values of the histogram except $n_k = n$, and denote $f(n) = \mathbb{P}\left[k \in \mathcal{M}\left((n_i)_{i \in U}\right)\right]$. Then $f(n)$ must satisfy inequalities 1 to 4 from Section 2.1 in order for $\mathcal{M}$ to be (ε, δ)-DP. Then, by Theorem 1, $f(n) \leq \pi_{\text{opt}}(n)$. Now, for a given input $(n_i)_{i \in U}$, the expected size of the output set is given by:

$$\sum_{k \in U} \mathbb{P}\left[k \in \mathcal{M}\left((n_i)_{i \in U}\right)\right]$$

which is bounded by $\sum_{k\in U} \pi_{\text{opt}}(n_k)$. This is exactly the expected output size obtained with $\mathcal{M}_{\text{opt}}$, which concludes the proof. □

3 Thresholding interpretation

In this section, we show that modulo a minor change in ε or δ, the optimal partition selection primitive π_{opt} can be interpreted as a *noisy thresholding* operation, similar to the Laplace-based strategy, but using a truncated version of the geometric distribution. We first define this distribution, then we use it to prove this second characterization of π_{opt}.

Definition 6 (k-TSGD)**.** *Given $p \in (0,1)$ and $k \in \mathbb{N}$ such that $k \geq 1$, the k-*truncated symmetric geometric distribution *(k-TSGD) of parameter p is the distribution defined on $\mathbb{Z}$ such that:*

$$\mathbb{P}[X = x] = \begin{cases} c \cdot (1-p)^{|x|} & \text{if } x \in [-k,k] \cap \mathbb{Z} \\ 0 & \text{otherwise} \end{cases} \tag{7}$$

where $c = \frac{p}{1+(1-p)-2(1-p)^{k+1}}$ is a normalization constant ensuring that the total probability is 1.

This distribution can also be obtained by taking a symmetric two-sided geometric distribution[2] [13], with success probability p and conditioning on the event that the result is in $[-k,k]$. As such, the k-truncated symmetric geometric distribution is the discrete analogue of the truncated Laplace distribution [11]. A similar construction was also defined in [12] to prove a lower bound on loss with (ε,δ)-differential privacy, but is not a proper probability distribution, since its total mass does not sum up to one[3].

Given privacy parameters ε and δ, we can set the values of p and k such that adding noise drawn from the truncated geometric distribution achieves (ε,δ)-differential privacy for counting queries.

Definition 7 (Truncated geometric mechanism)**.** *Given privacy parameters $\varepsilon > 0$ and $\delta > 0$, let $p = 1 - e^{-\varepsilon}$ and $k = \left\lceil \frac{1}{\varepsilon} \ln\left(\frac{e^\varepsilon + 2\delta - 1}{(e^\varepsilon+1)\delta}\right)\right\rceil$. Let the true result of an integer-valued query with sensitivity 1 be $\mu \in \mathbb{Z}$. Then the truncated geometric mechanism returns $\mu + X$, where X is drawn from the k-TSGD with success probability p. The result has the distribution:*

$$\mathbb{P}[Y = y] = \begin{cases} c \cdot e^{-|y-\mu|\varepsilon} & \text{if } y \in [\mu-k, \mu+k] \cap \mathbb{Z} \\ 0 & \text{otherwise} \end{cases}$$

where $c = \frac{1-e^{-\varepsilon}}{1+e^{-\varepsilon}-2e^{-(k+1)\varepsilon}}$ is a normalization constant ensuring that the total probability is 1.

The value of k is the smallest value such that

$$\mathbb{P}[X = k] = \frac{e^{-k\varepsilon}(1-e^{-\varepsilon})}{1+e^{-\varepsilon}-2e^{-(k+1)\varepsilon}} \leq \delta$$

for the k-TSGD.

Theorem 4. *The truncated geometric mechanism satisfies (ε,δ)-differential privacy.*

Proof. This follows the same line of reasoning as the proof of Theorem 1 in [11]. The only difference is the change from a continuous distribution to a discrete distribution, since all the values are integers. If the result of the query before adding noise is μ, then for an adjacent database, the corresponding value μ' must be in $\{\mu-1, \mu, \mu+1\}$. If $\mu' = \mu$, the distribution of the output after adding noise is unchanged, trivially satisfying the (ε,δ)-differential privacy property. By symmetry, it is sufficient to analyze the case when $\mu' = \mu + 1$. Here, the new distribution of the output of the mechanism is

$$\mathbb{P}\left[Y' = y\right] = c \cdot e^{-|y-\mu-1|\varepsilon}$$

if $y \in [\mu-k+1, \mu+k+1] \cap \mathbb{Z}$, and $\mathbb{P}[Y' = y]$ otherwise.

By symmetry, to show that (ε,δ)-differential privacy is satisfied, we only need to show that $\mathbb{P}[Y' \in S] \leq e^\varepsilon \mathbb{P}[Y \in S] + \delta$ for all $S \subset \mathbb{Z}$. For all values $y \in \mathbb{Z}$ except $\mu+k+1$, $\mathbb{P}[Y' = y] \leq e^\varepsilon \mathbb{P}[Y = y]$. Also, $\mathbb{P}[Y = \mu+k+1] = 0$ and $\mathbb{P}[Y' = \mu+k+1] = \mathbb{P}[X = k] > 0$. Therefore, $\mathbb{P}[Y' \in S] - e^\varepsilon \mathbb{P}[Y \in S]$ is maximized when $S = \{\mu+k+1\}$. This means that the condition is satisfied if $\mathbb{P}[X = k] \leq \delta$. From the definition of k in the truncated geometric mechanism:

$$k = \left\lceil \frac{1}{\varepsilon} \ln\left(\frac{e^\varepsilon + 2\delta - 1}{(e^\varepsilon+1)\delta}\right)\right\rceil$$

which leads to:

$$e^{k\varepsilon} \geq \frac{e^\varepsilon + 2\delta - 1}{(e^\varepsilon+1)\delta}$$

thus:

$$(e^{k\varepsilon})(e^\varepsilon+1)\delta - 2\delta \geq e^\varepsilon - 1$$

and:

$$(1 + e^{-\varepsilon} - 2e^{-(k+1)\varepsilon})\delta \geq e^{-k\varepsilon}(1-e^{-\varepsilon})$$

2 Also called the discrete Laplace distribution [17].

3 In the proof of Theorem 8, the sum for non-negative i is assumed to sum up to 1/2, but 0 is counted twice when summing non-negative and non-positive i.

and finally:

$$\begin{aligned}\delta &\geq \frac{e^{-k\varepsilon}(1-e^{-\varepsilon})}{1+e^{-\varepsilon}-2e^{-(k+1)\varepsilon}} \\ &\geq \mathbb{P}\left[X=k\right].\end{aligned}$$

□

Let us get some intuition why thresholding the truncated geometric mechanism leads to an optimal partition selection primitive. First, we compute the tail cumulative distribution function for the output of the truncated geometric mechanism. Summing the probability masses gives a geometric series:

$$\mathbb{P}\left[Y \geq y\right] = \begin{cases} 1 & \text{if } y \leq \mu - k \\ 1 - \frac{e^{(k+y-\mu)\epsilon}-1}{e^{\epsilon}-1} c e^{-k\epsilon} & \text{if } \mu - k \leq y \leq \mu - 1 \\ \frac{e^{(\mu+k+1-y)\epsilon}-1}{e^{\epsilon}-1} c e^{-k\epsilon} & \text{if } \mu \leq y \leq \mu + k \\ 0 & \text{if } y > \mu + k. \end{cases}$$

If we define $\delta = ce^{-k\epsilon}$ and rearrange the the cases as functions of μ, we get:

$$\mathbb{P}\left[Y \geq y\right] = \begin{cases} 0 & \text{if } \mu < y - k \\ \frac{e^{(\mu+k+1-y)\epsilon}-1}{e^{\epsilon}-1}\delta & \text{if } y - k \leq \mu \leq y \\ 1 - \frac{e^{(k+y-\mu)\epsilon}-1}{e^{\epsilon}-1}\delta & \text{if } y + 1 \leq \mu \leq y + k \\ 1 & \text{if } \mu \geq y + k. \end{cases} \tag{8}$$

The $\mu \leq y$ cases of the formula are the same as the closed-form formula for π_{opt} in Theorem 1 for values less than n_1. The $\mu > y$ cases are simply the symmetric reflection of the former. We formalize this intuition and show that whenever $\frac{1}{\varepsilon}\ln\left(\frac{e^{\varepsilon}+2\delta-1}{(e^{\varepsilon}+1)\delta}\right)$ is an integer, the two approaches are exactly the same.

Theorem 5 (Noisy thresholding is optimal)**.** *If $\delta \in (0,1)$ and $\varepsilon > 0$ are such that $k = \frac{1}{\varepsilon}\ln\left(\frac{e^{\varepsilon}+2\delta-1}{(e^{\varepsilon}+1)\delta}\right)$ is an integer, then for all n:*

$$\pi_{\text{opt}}\left(n\right) = \mathbb{P}\left[n + X \geq k + 1\right]$$

where X is a random variable sampled from a k-truncated symmetric geometric distribution of success probability $(1-e^{-\varepsilon})$.

Proof. When $k = \frac{1}{\varepsilon}\ln\left(\frac{e^{\varepsilon}+2\delta-1}{(e^{\varepsilon}+1)\delta}\right)$ is an integer, we have $n_1 = k+1$, and

$$e^{k\varepsilon} = \frac{e^{\varepsilon}+2\delta-1}{(e^{\varepsilon}+1)\delta}$$

which leads to

$$(e^{k\varepsilon})(e^{\varepsilon}+1)\delta = e^{\varepsilon} - 1 + 2\delta$$

and

$$(e^{(k+1)\varepsilon}-1)\delta + (e^{k\varepsilon}-1)\delta = e^{\varepsilon} - 1.$$

On further rearranging, we get

$$\frac{e^{(k+1)\varepsilon}-1}{e^{\varepsilon}-1}\cdot\delta + \frac{e^{k\varepsilon}-1}{e^{\varepsilon}-1}\cdot\delta = 1,$$

and thus:

$$1 - \pi_{\text{opt}}\left(n_1\right) = \pi_{\text{opt}}\left(n_1 - 1\right).$$

From Lemma 2, we also get

$$1 - \pi_{\text{opt}}\left(n\right) = e^{-\varepsilon}((1 - \pi_{\text{opt}}\left(n-1\right)) - \delta)$$

if $n_1 < n \leq n_2$. Since we also have

$$\pi_{\text{opt}}\left(n\right) = e^{\varepsilon}(\pi_{\text{opt}}\left(n-1\right) + \delta)$$

if $0 < n \leq n_1$, we find that for $n_1 < n \leq n_2$,

$$\begin{aligned}\pi_{\text{opt}}\left(n\right) &= 1 - \pi_{\text{opt}}\left(2n_1 - 1 - n\right) \\ &= 1 - \frac{e^{(2k+1-n)\varepsilon}-1}{e^{\varepsilon}-1}\cdot\delta.\end{aligned}$$

Consequently, for such special combinations of ε and δ

$$n_2 = 2n_1 - 1 = 2k + 1.$$

Now, rewriting the formula for π_{opt} in Theorem 1 using $\mu = n$ and $k = n_1 - 1$ gives us that $\pi_{\text{opt}}\left(\mu\right)$ is:

- 0 if $\mu \leq 0$,
- $\frac{e^{(\mu+k+1-(k+1))\varepsilon}-1}{e^{\varepsilon}-1}\cdot\delta$ if $\mu \leq k+1$,
- $\left(1 - e^{(\mu-(k+1))\varepsilon}\right)\left(1 + \frac{\delta}{e^{\varepsilon}-1}\right) + e^{(\mu-(k+1))\varepsilon}\frac{e^{(k+1)\varepsilon}-1}{e^{\varepsilon}-1}\cdot\delta$ if $k+1 < \mu \leq 2k+1$,
- 1 otherwise.

Comparing this with (8) shows that for this combination of ε and δ, and for the corresponding derived values of p and k,

$$\pi_{\text{opt}}\left(\mu\right) = \mathbb{P}\left[Y \geq k+1\right].$$

□

This characterization suggests a simple implementation of the optimal partition selection primitive, at a minor cost in ε or δ. Given arbitrary ε and δ, one can replace ε by $\hat{\varepsilon} \leq \varepsilon$, or δ by $\hat{\delta} \leq \delta$ to ensure that k from Theorem 5 is an integer. In our definition of the truncated geometric mechanism, we choose the latter strategy, requiring a slightly lower δ by using an integer upper bound on k, and using $p = 1 - e^{-\varepsilon}$ to fully utilize the ε budget. We then apply the truncated geometric mechanism to the number of unique users in each partition, and return

this partition if the noisy count is larger than k. Further, this noisy count may also be published for such a partition, while still satisfying (ε, δ)-differential privacy.

To see this, consider an arbitrarily large finite family of partitions[4] Q such that each user in a database D is associated with at most one partition $q \in Q$. Consider the following mechanism.

Definition 8 (k-TSGD thresholded release)**.** *For a database D, let $c_q(D)$ be the number of users associated with partition q. Let $Q_D \subset Q$ be the finite subset $\{q \mid q \in Q \text{ and } c_q(D) > 0\}$ of partitions present in the dataset D. Let the noise X_q for $q \in Q$ be i.i.d. random variables drawn from the k-TSGD of parameters $p = 1 - e^{-\varepsilon}$ and $k = \left\lceil \frac{1}{\varepsilon} \ln\left(\frac{e^\varepsilon + 2\delta - 1}{(e^\varepsilon + 1)\delta}\right)\right\rceil$. Let $\hat{c}_q(D) = c_q(D) + X_q$. Then, the k-TSGD thresholded release mechanism produces the set*

$$\{(q, \hat{c}_q(D)) \mid q \in Q_D \text{ and } \hat{c}_q(D) > k\}.$$

Theorem 6. *The k-TSGD thresholded release mechanism satisfies (ε, δ)-differential privacy.*

Proof. Consider the mechanism that adds a k-TSGD of parameter $p = 1 - e^{-\varepsilon}$ and $k = \left\lceil \frac{1}{\varepsilon} \ln\left(\frac{e^\varepsilon + 2\delta - 1}{(e^\varepsilon + 1)\delta}\right)\right\rceil$ to every possible partition count, including those not present in the dataset. That is, we apply the truncated geometric mechanism to the unique-user counts for all possible partitions (even partitions not contained in the database), which produces the set

$$\{(q, \hat{c}_q(D)) \mid q \in Q\}.$$

This mechanism is (ε, δ)-differentially private: a single user's addition or removal changes only one partition, and on this partition, Theorem 4 shows that the output satisfies (ε, δ)-differential privacy. Combined with the condition that the noise values are independent, this means that the entire mechanism is also (ε, δ)-differentially private.

Adding a thresholding step to release the noised values only when they are greater than k is only post-processing. Therefore, the entire mechanism that releases

$$\{(q, \hat{c}_q(D)) \mid q \in Q \text{ and } \hat{c}_q(D) > k\}$$

is also (ε, δ)-differentially private.

Now, notice that this mechanism is *exactly the same* as if we had only added noise to the partitions in Q_D: the noise added to zero in empty partitions will be at most k, so these partitions will be removed from the output in the thresholding step. Since these two mechanisms are identical and one is (ε, δ)-differentially private, both are (ε, δ)-differentially private. □

We note that this can be extended to the case where the set of allowed partitions Q is countably infinite, using standard techniques from measure theory [16]. Thus, this mechanism is the (ε, δ)-differential privacy equivalent of Algorithm Filter in [4], which achieves $(\varepsilon, 0)$-differential privacy when the set of possible partitions is known beforehand and not very large.

To demonstrate the utility of such an operation, consider a slight variation of the example query presented in the introduction.

```
SELECT
  device_model,
  COUNT(user_id),
  AVG(latency)
FROM database
GROUP BY device_model
```

For simplicity, let us assume that each user contributes only one row with a single value for the latency. Then, this may be implemented in the following way.

```
SELECT
  device_model,
  COUNT(user_id),
  SUM(latency) / COUNT(user_id)
FROM database
GROUP BY device_model
```

The available (ε, δ) budget must be split up for the partition selection, sum and count. Instead of instantiating separate noise values for partition selection and the count and having to split up the (ε, δ), we can use noisy thresholding on the count. This may be used to obtain a more accurate count or to leave more of the (ε, δ) budget for the sum estimation.

4 Numerical evaluation

Theorem 1 shows that the optimal partition selection primitive π_{opt} outperforms all other options. How does it compare with the naive strategy of adding Laplace noise and thresholding the result?

Definition 9 (Laplace partition selection [21])**.** *We denote by* $\text{Lap}(b)$ *a random variable sampled from a*

4 For example, all possible partitions representable by bytestrings that fit within available data storage

Laplace distribution of mean 0 *and of scale* b*. The following partition selection strategy* ρ_{Lap}*, called* Laplace-based partition selection*, is* (ε, δ)*-differentially private:*

$$\rho_{\text{Lap}}(n) = \begin{cases} \text{drop} & \text{if } n + \text{Lap}\left(\frac{1}{\varepsilon}\right) < 1 - \frac{\ln(2\delta)}{\varepsilon} \\ \text{keep} & \text{otherwise.} \end{cases}$$

We denote by π_{Lap} *the corresponding partition selection primitive:* $\pi_{\text{Lap}}(n) = \mathbb{P}\left[\rho_{\text{Lap}}(n) = \text{keep}\right]$.

As expected, using the optimal partition selection primitive translates to a larger probability of releasing a partition with the same user. As exemplified in Figure 1, the difference is especially large in the high-privacy regime.

To better understand the dependency on ε and δ, we also compare the *midpoint* obtained for both partition selection strategies ρ: the number n for which the probability of releasing a partition with n users is 0.5. For Laplace-based partition selection, this n is simply the threshold. As Figure 2 shows, the gains are especially substantial when ε is small, and not significant for $\varepsilon > 1$. Figure 3 shows the dependency on δ: for a fixed ε, there is a *constant* interval between the midpoints of both strategies. Thus, the relative gains are larger for a larger δ, since the midpoint is also smaller.

We also verified experimentally that on each partition, the selection mechanism runs in constant, very short time, on the order of 100 nanoseconds on a standard machine. This is not surprising: Theorem 1 provides a simple, closed-form formula for computing $\pi_{\text{opt}}(n)$, and generating the random decision based on this probability is computationally trivial. The performance impact of Laplace-based thresholding is similarly negligible: the only real cost of such simple partition selection strategies is to count the number of unique users n in each partition, which is orders of magnitude more computationally intensive than computing $\pi(n)$.

5 Discussion

The approach presented in this work is both easy to implement and efficient. Counting the number of unique users per partition can be done in one pass over the data and is massively parallelizable. Furthermore, since there is a relatively small value k such that the probability of keeping a partition with $n \geq k$ users is 1, the counting process can be interrupted as soon as a partition reaches k users. This keeps memory usage low (in $O(k)$) without requiring approximate count-distinct algorithms like HyperLogLog [10] for which a more complex sensitivity analysis would be needed.

Extension to multiple partitions per user

Our approach could, in principle, be extended to cases where each user can contribute to $\kappa > 1$ partitions. Following the intuition of Lemma 1, we could list a set of recursive equations defining $\pi_{\text{opt}}(n)$ as a function of $\pi_{\text{opt}}(i)$ for $i < n$. Sadly, the system of equations quickly gets too large to solve naively. Consider, for example, the case where $\kappa = 2$. The differential privacy constraint requires, for all $n \geq i \geq 0$ and all $S \subseteq \{\text{drop}, \text{keep}\}^2$:

$$\begin{aligned} &\mathbb{P}\left[(\rho_\pi(n+1), \rho_\pi(i+1)) \in S\right] \\ &\quad \leq e^\varepsilon \cdot \mathbb{P}\left[(\rho_\pi(n), \rho_\pi(i)) \in S\right] + \delta \\ &\mathbb{P}\left[(\rho_\pi(n), \rho_\pi(i)) \in S\right] \\ &\quad \leq e^\varepsilon \cdot \mathbb{P}\left[(\rho_\pi(n+1), \rho_\pi(i+1)) \in S\right] + \delta \end{aligned}$$

Thus, to maximize $\pi(n)$, we have to consider $32n$ inequalities: n possible values of i, $2^{(2^2)} = 16$ possible values of S, and two inequalities. When κ increases, the total number of inequalities to compute all elements up to n is $O\left(n^\kappa 2^{\kappa^2}\right)$. Some of these inequalities are trivial (e.g., when $S = \emptyset$ or $S = \{\text{drop}, \text{keep}\}^\kappa$), but most are not. We do not know whether it is possible to only consider a small number of these inequalities, and obtain the others "for free".

Furthermore, the recurrence-based proof of optimality of π_{opt} only holds when we assume that each user contributes to *exactly* κ partitions in the original dataset. As discussed previously, this is relatively frequent when $\kappa = 1$, but it rarely happens for larger values of κ: in typical datasets, some users contribute to more partitions than others. In that case, *weighing* the contributions of each user differently can bring additional benefits, as can changing each user's strategy based on those of previous users [14]. For this generalized problem, it seems difficult to even define what optimality means.

The simplest option to use our approach for $\kappa > 1$ is to divide the total privacy budget in κ. For generic tooling with strict scalability requirements where the analyst manually specifies κ, we recommend using our method (splitting the privacy budget) for $\kappa \leq 3$, and weighted Gaussian thresholding (described in [14]) for $\kappa \geq 4$. Figure 4 compares the mid-point of the partition selection strategy between π_{opt}, Laplace-based thresholding, and (non-weighted) Gaussian-based thresholding. It shows that the crossing point happens for $\kappa = 3$, this stays true for varying values ε and δ.

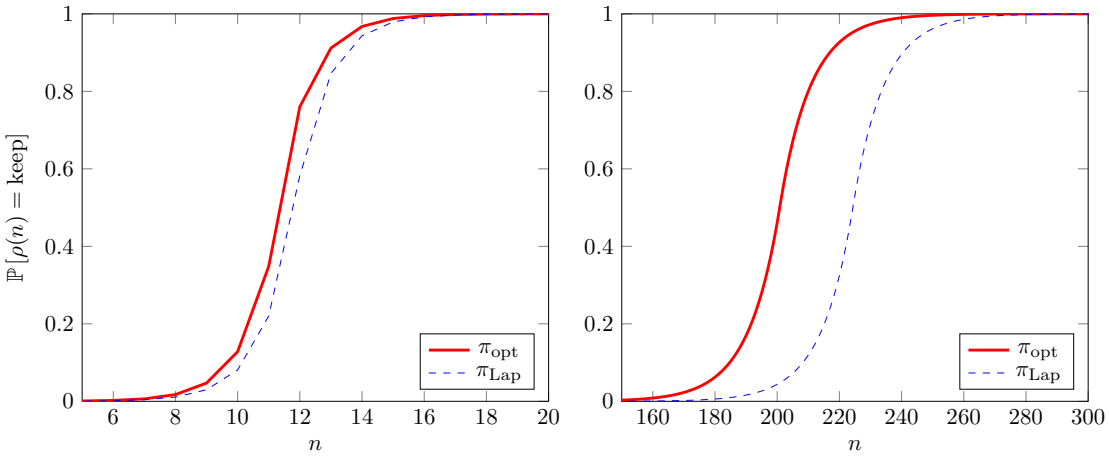

Fig. 1. Probability of releasing a partition depending on the number of unique users, comparing Laplace-based partition selection with π_{opt}. On the left, $\varepsilon = 1$ and $\delta = 10^{-5}$, on the right, $\varepsilon = 0.1$ and $\delta = 10^{-10}$.

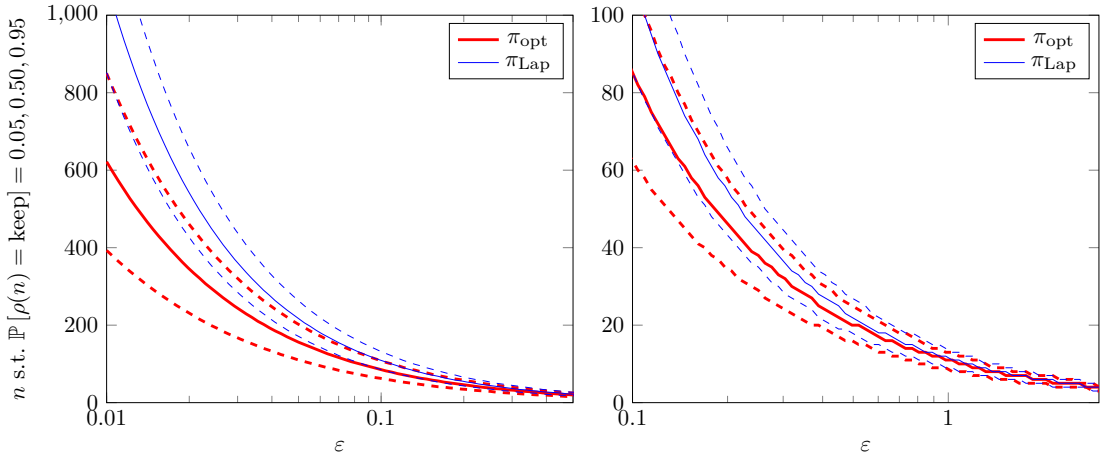

Fig. 2. Comparison of the 5th, 50th and 95th percentile of the partition selection strategy ρ as a function of ε, for $\delta = 10^{-5}$. The mid-point is plotted as a solid line, while the 5th and 95th percentiles are dashed.

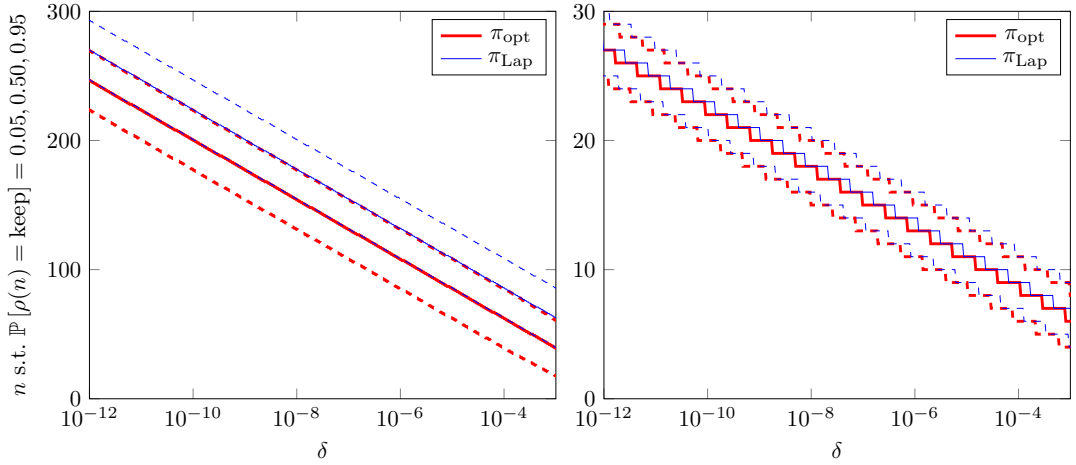

Fig. 3. Comparison of the 5th, 50th and 95th percentile of the partition strategy ρ as a function of δ, for $\varepsilon = 0.1$ (left) and $\varepsilon = 1$ (right). The mid-point is plotted as a solid line, while the 5th and 95th percentiles are dashed.

Comparison with weighted Gaussian thresholding is less trivial, since its benefits depend on the data distribution. However, weighted Gaussian thresholding is always better than Gaussian-based thresholding, and is straightforward to implement in a massively parallelizable fashion. We have also observed that its utility benefits are only significant for large κ, so our recommendation to use π_{opt} for $\kappa \leq 3$ is likely robust.

Policy-based approaches like those described in [14] also provide more utility, but they are not as scalable: since the strategy for each user depends on the choices made by all previous users, the computation cannot be parallelized. It also requires to keep an in-memory histogram of all partitions seen so far, which also does not scale to extremely large datasets. Improving the scalability of such policy-based approaches is an interesting open problem, on which further research would be valuable.

Extension to bounded differential privacy

In the definition of differential privacy we use in this work, neighboring databases differ in a single user being added or removed. This notion is called *unbounded* differential privacy in [20], by contrast to *bounded* differential privacy, in which neighboring datasets differ in a single user *changing* their data. (ε, δ)-unbounded DP implies $(2\varepsilon, 2\delta)$-DP, which provides a trivial way to extend our method to the bounded version of the definition: simply divide the privacy budget by two. This method outperforms Laplace-based thresholding, since Laplace noise of scale $2/\varepsilon$ must be added in the bounded setting (since L_1-sensitivity is 2 and no longer 1). Further, when k from Theorem 5 is an integer, this noise distribution exactly achieves the lower bound on the loss from [12], and is therefore optimal for arbitrary symmetric loss functions.

Other possible extensions

The truncated geometric mechanism can be used as a building block to replace the Laplace or geometric mechanism in situations where (ε, δ)-DP with $\delta > 0$ is acceptable. Similarly to the truncated Laplace mechanism [11], this building block is optimal for integer-valued functions.

To see how such a building block could be used in practice, consider the problem of releasing a histogram where some partitions are known in advance (call them *public partitions*), and some are not and must be discovered using the private data (*private partitions*). Note that some public partitions might be absent from the private data. In that case, one could add truncated geometric noise to all partitions (public *and* private), and use two distinct thresholds: one given by the formula for k in Definition 7, and an arbitrary one t.

- k is used to threshold the partitions present in the private data *but not in the list of public partitions*;
- t is used to threshold the public partitions (whether or not they are also in the private data).

The second threshold t can be arbitrary, and allows an analyst to control the trade-off between false positives and false negatives. For example, setting $t = 0$ guarantees that all public partitions that appear in the private data are present in the output (no false negatives), at the cost of having a potentially large number of public partitions appearing in the output even though they were not present (many false positives). Conversely, setting $t = k$ guarantees that *only* the partitions present in the private data can be present in the output (no false positives), at the cost of dropping potentially many of them (many false negatives). Intermediate values of t can allow an analyst to more finely tune this trade-off depending on the application.

We postulate that this building block could be used in a variety of different settings, and combined with existing techniques. For example, one could build a variant of the standard vector technique [8] that uses the truncated geometric mechanism instead of the Laplace mechanism to add noise to the output of the queries and to the threshold. This could be used to efficiently simulate the standard vector technique on a very large number of queries, most of which are deterministically below the threshold and can be skipped during computation. Formalizing this intuition and using it for partition selection with $\kappa > 1$ is left to future work.

6 Conclusion

We introduced an optimal primitive for differentially private partition selection, a special case of differentially private set union where the sensitivity is 1. This optimal approach is simple to implement and efficient. It outperforms Laplace-based thresholding; the utility gain is especially significant in the high-privacy (small ε) regime. Besides the possible research directions outlined previously, this work leaves two open questions. Is it possible to extend this optimal approach to larger sensitivities in

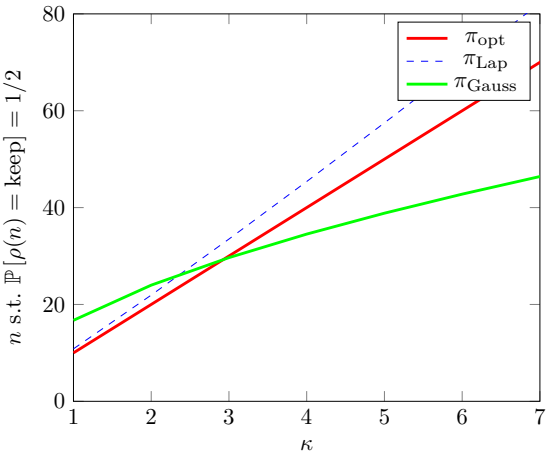

Fig. 4. Comparison of the mid-point of the partition selection strategy ρ as a function of κ, for $\varepsilon = 1$ and $\delta = 10^{-5}$. For $\kappa > 1$, the privacy budget is divided by κ for π_{opt} and π_{Lap}; for π_{Gauss}, we use the formula in [2] to derive the standard deviation of Gaussian noise, and we split the δ between noise and thresholding to minimize the threshold.

a simple and efficient manner? Furthermore, is it possible to combine this primitive with existing approaches to differentially private set union [14], like weighted histograms or policy-based strategies?

7 Acknowledgments

The authors gratefully acknowledge Alex Kulesza, Chao Li, Michael Daub, Kareem Amin, Peter Dickman, Peter Kairouz, and the PETS reviewers for their helpful feedback on this work.

D.D. was employed by Google and ETH Zurich at the time of this work. This research received no specific grant from any funding agency in the public, commercial, or not-for-profit sector.

References

[1] G. Acs, C. Castelluccia, and R. Chen. Differentially private histogram publishing through lossy compression. In *2012 IEEE 12th International Conference on Data Mining*, pages 1–10. IEEE, 2012.

[2] B. Balle and Y.-X. Wang. Improving the gaussian mechanism for differential privacy: Analytical calibration and optimal denoising. In *International Conference on Machine Learning*, pages 394–403. PMLR, 2018.

[3] J. Bater, X. He, W. Ehrich, A. Machanavajjhala, and J. Rogers. Shrinkwrap: Differentially-private query processing in private data federations. *arXiv preprint arXiv:1810.01816*, 2018.

[4] G. Cormode, C. Procopiuc, D. Srivastava, and T. T. Tran. Differentially private summaries for sparse data. In *Proceedings of the 15th International Conference on Database Theory*, pages 299–311, 2012.

[5] G. Cormode, M. Procopiuc, D. Srivastava, and T. T. Tran. Differentially private publication of sparse data. *arXiv preprint arXiv:1103.0825*, 2011.

[6] B. Ding, M. Winslett, J. Han, and Z. Li. Differentially private data cubes: optimizing noise sources and consistency. In *Proceedings of the 2011 ACM SIGMOD International Conference on Management of data*, pages 217–228, 2011.

[7] C. Dwork, F. McSherry, K. Nissim, and A. Smith. Calibrating noise to sensitivity in private data analysis. In *Theory of Cryptography Conference*, pages 265–284. Springer, 2006.

[8] C. Dwork, M. Naor, O. Reingold, G. N. Rothblum, and S. Vadhan. On the complexity of differentially private data release: efficient algorithms and hardness results. In *Proceedings of the forty-first annual ACM symposium on Theory of computing*, pages 381–390, 2009.

[9] C. Dwork and A. Roth. The algorithmic foundations of differential privacy. *Foundations and Trends in Theoretical Computer Science*, 9(3-4):211–407, 2014.

[10] P. Flajolet, É. Fusy, O. Gandouet, and F. Meunier. Hyperloglog: the analysis of a near-optimal cardinality estimation algorithm. In *Discrete Mathematics and Theoretical Computer Science*, pages 137–156. Discrete Mathematics and Theoretical Computer Science, 2007.

[11] Q. Geng, W. Ding, R. Guo, and S. Kumar. Tight analysis of privacy and utility tradeoff in approximate differential privacy. In *International Conference on Artificial Intelligence and Statistics*, pages 89–99, 2020.

[12] Q. Geng and P. Viswanath. Optimal noise adding mechanisms for approximate differential privacy. *IEEE Transactions on Information Theory*, 62(2):952–969, Feb 2016.

[13] A. Ghosh, T. Roughgarden, and M. Sundararajan. Universally utility-maximizing privacy mechanisms. *SIAM Journal on Computing*, 41(6):1673–1693, 2012.

[14] S. Gopi, P. Gulhane, J. Kulkarni, J. H. Shen, M. Shokouhi, and S. Yekhanin. Differentially private set union. *arXiv preprint arXiv:2002.09745*, 2020.

[15] M. Hay, V. Rastogi, G. Miklau, and D. Suciu. Boosting the accuracy of differentially-private histograms through consistency. *arXiv preprint arXiv:0904.0942*, 2009.

[16] N. Holohan, D. J. Leith, and O. Mason. Differential privacy in metric spaces: Numerical, categorical and functional data under the one roof. *Information Sciences*, 305:256–268, 2015.

[17] S. Inusah and T. J. Kozubowski. A discrete analogue of the laplace distribution. *Journal of statistical planning and inference*, 136(3):1090–1102, 2006.

[18] N. Johnson, J. P. Near, and D. Song. Towards practical differential privacy for sql queries. *Proceedings of the VLDB Endowment*, 11(5):526–539, 2018.

[19] H. Kaplan, Y. Mansour, and U. Stemmer. The sparse vector technique, revisited. *arXiv preprint arXiv:2010.00917*, 2020.

[20] D. Kifer and A. Machanavajjhala. No free lunch in data privacy. In *Proceedings of the 2011 ACM SIGMOD International Conference on Management of data*, pages 193–204, 2011.

[21] 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*, pages 171–180, 2009.

[22] I. Kotsogiannis, Y. Tao, X. He, M. Fanaeepour, A. Machanavajjhala, M. Hay, and G. Miklau. Privatesql: a differentially private sql query engine. *Proceedings of the VLDB Endowment*, 12(11):1371–1384, 2019.

[23] J. Lee and C. W. Clifton. Top-k frequent itemsets via differentially private fp-trees. In *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 931–940, 2014.

[24] H. Li, L. Xiong, and X. Jiang. Differentially private synthesization of multi-dimensional data using copula functions. In *Advances in database technology: proceedings. International conference on extending database technology*, volume 2014, page 475. NIH Public Access, 2014.

[25] M. Lyu, D. Su, and N. Li. Understanding the sparse vector technique for differential privacy. *arXiv preprint arXiv:1603.01699*, 2016.

[26] R. J. Wilson, C. Y. Zhang, W. Lam, D. Desfontaines, D. Simmons-Marengo, and B. Gipson. Differentially private SQL with bounded user contribution. *arXiv preprint arXiv:1909.01917*, 2019.

[27] X. Xiao, G. Wang, and J. Gehrke. Differential privacy via wavelet transforms. *IEEE Transactions on knowledge and data engineering*, 23(8):1200–1214, 2010.

[28] Y. Xiao, L. Xiong, L. Fan, and S. Goryczka. Dpcube: differentially private histogram release through multidimensional partitioning. *arXiv preprint arXiv:1202.5358*, 2012.

[29] J. Xu, Z. Zhang, X. Xiao, Y. Yang, G. Yu, and M. Winslett. Differentially private histogram publication. *The VLDB Journal*, 22(6):797–822, 2013.

[30] J. Zhang, G. Cormode, C. M. Procopiuc, D. Srivastava, and X. Xiao. Privbayes: Private data release via bayesian networks. *ACM Transactions on Database Systems (TODS)*, 42(4):1–41, 2017.