Differentially Private Histograms under Continual Observation: Streaming Selection into the Unknown

Adrian Rivera Cardoso **Ryan Rogers**
Data Science Applied Research, LinkedIn

Abstract

We generalize the continuous observation privacy setting from Dwork et al. (2010a) and Chan et al. (2011) by allowing each event in a stream to be a subset of some (possibly unknown) universe of items. We design differentially private (DP) algorithms for histograms in several settings, including top-k selection, with privacy loss that scales with polylog(T), where T is the maximum length of the input stream. We present a meta-algorithm that can use existing one-shot top-k private algorithms as a subroutine to continuously release DP histograms from a stream. Further, we present more practical DP algorithms for two settings: 1) continuously releasing the top-k counts from a histogram over a known domain when an event can consist of an arbitrary number of items, and 2) continuously releasing histograms over an unknown domain when an event has few items.

1 INTRODUCTION

Providing real-time statistics on streaming data is a common task in data analytics. For example, one may want to provide a running count on the number of people that have purchased a particular drug at a pharmacy. This data can be very useful for tracking and identifying local epidemics in a given region. However, this particular data is very sensitive so privacy techniques should be applied to protect those who are purchasing medications. Differential privacy (DP) has emerged as the go to method in industry to provide privacy for aggregate results. In this work, we study the problem of continually releasing aggregate counts over a stream of incoming data subject to DP.

Let $\mathcal{U}$ be a set of items and $\omega_{1:T} = \omega_1, \cdots, \omega_T$ be a stream of T events, e.g. pharmacy purchases, where $\omega_t \subseteq \mathcal{U}$.[1] Our goal is to release, at every time t, the counts of all items in the substream $\omega_{1:t}$, or the most frequent counts, subject to DP. This setting is referred to as the continual observation model of DP and originated in works from Dwork et al. (2010a) and Chan et al. (2011) where it is traditionally assumed that $\mathcal{U}$ is known and $|\omega_t| \leq 1$. In this paper we study settings where $\mathcal{U}$ is either known (Known Domain) or unknown (Unknown Domain), and where a bound Δ_0 on $|\omega_t|$ is known (Restricted ℓ_0-sensitivity) or where it can be as large as $|\mathcal{U}|$ (Unrestricted ℓ_0-sensitivity). In the unrestricted ℓ_0-sensitivity setting, we only want to return the top-k counts, rather than the full set of counts and have privacy loss increase with k or $\sqrt{k}$, rather than with $d := |\mathcal{U}|$. Simply applying restricted ℓ_0-sensitivity algorithms in the unrestricted ℓ_0-sensitivity setting would require setting $\Delta_0 = d$, so that privacy loss increases with d or $\sqrt{d}$.

The guarantee of a DP algorithm is that the output distributions for two similar input streams will be similar. As is common in the continual observation DP literature, we do not restrict the number of events ω_t that a user can impact, thus we provide *event level* privacy guarantees, as opposed to *user level* privacy, which would bound the number of events any user can contribute. In any of the settings we consider, we could apply the corresponding one-shot DP algorithms presented in Table 1 on the data available at time t. However, releasing a total of T answers would cause the total privacy loss to scale as $O(\sqrt{T})$ (using advanced composition privacy loss bounds). The goal of this work is to design algorithms for all settings in Table 1 and have the total privacy loss scale as $O(\text{polylog}(T))$, or equivalently have the noise that we include for DP

Proceedings of the 25[th] International Conference on Artificial Intelligence and Statistics (AISTATS) 2022, Valencia, Spain. PMLR: Volume 151.

[1] Our setting easily extends to each event consisting of items in $\mathcal{U}$ and counts of each item from that event. We can accommodate for this more general setting by scaling the noise by the maximum amount any item can change in an event, i.e. the ℓ_∞-sensitivity

scale with $O(\text{polylog}(T))$ for a constant privacy loss.

One-shot	Restricted ℓ_0-sensitivity	Unrestricted ℓ_0-sensitivity
Known Domain	`KnownGauss`	`KnownGumb`
Unknown Domain	$\texttt{LimitDom}_{\texttt{Lap}}$ / `UnkGauss` (this work)	`UnkGumb`

Table 1: DP algorithms for data analytics tasks in the one-shot analytics setting from prior work Dwork et al. (2006a), McSherry and Talwar (2007), and Durfee and Rogers (2019).

Continual Obs.	Restricted ℓ_0-sensitivity	Unrestricted ℓ_0-sensitivity
Known Domain	Binary Mechanism (Chan et al., 2011)	`SparseGumb` (this work)
Unknown Domain	`UnkBase` (this work)	`MetaAlgo` (this work)

Table 2: Our contributions: DP algorithms for data analytics tasks in the continual observation setting.

Existing DP algorithms for the continual observation setting include the celebrated Binary Mechanism (Chan et al., 2011), which can be applied to the known domain and restricted ℓ_0-sensitivity setting. To our knowledge, we are the first to consider continually releasing the item with the maximum count and its count at each round subject to DP, despite the one-shot DP algorithm being the classical Exponential Mechanism (McSherry and Talwar, 2007). Other works have considered the problem of continually returning the top-k (Chan et al., 2011) and heavy hitters in a stream (Chan et al., 2012), (Mir et al., 2011). The main difference in our setting is that a single event consists of multiple distinct items, while earlier works have events with at most one item or a user can modify at most one item at each event, which falls under the restricted ℓ_0-sensitivity with known domain setting. Our setting provides stronger levels of privacy because a single event in a stream can affect the count of multiple items at once. In the pharmacy example, an event would be a purchase occurring and the items would be the drugs that were purchased, which need not be a single drug. Note that Mir et al. (2011) considers a related but more restrictive privacy model, referred to as *pan-privacy* from Dwork et al. (2010b), that includes security considerations so that privacy is preserved even if an adversary can access internal states of the algorithm.

We point out that Dwork et al. (2010a) provides a general transformation from one-shot algorithms to those with privacy guarantees under continual observation. However, this general transformation requires the one-shot algorithm to return a scalar, which is then compared with the algorithm's outcomes at later rounds and only displays the new outcome if it is significantly different than the previous result, otherwise it will show the old result. Our one-shot algorithms return a histogram of counts with labels that can differ in each round, so it is not clear what scalar function to assign to determine when a new outcome should be used, rather than displaying an older result. We will use a similar idea to this general transformation in Section 5 when continually returning the top-k from a stream of events and only updating results if there is a count that should be in the top-k but is not at a current round. Our approach allows for the privacy loss to increase with the number of times the top-k should be updated, rather than when the counts from the previous round's top-k need to be updated due to counts increasing but the top-k remaining unchanged as would be the case by using the approach in Dwork et al. (2010a) without returning item labels.

We also design algorithms that can be used in scalable and distributed real-time analytics platforms where low latency is crucial, so retrieving and passing the algorithm a substream $\omega_{1:t}$ at each time step t is not feasible. Instead, algorithms in this setting only have access to the histogram at time t. An example of such a privacy platform in production is described in detail in Rogers et al. (2020). The Binary Mechanism can be implemented in this setting, since we only need access to the true counts over all items at each round t, rather than the full sequence of events, as long as the algorithm knows the length of the stream t and the noise it has used in previous rounds, which can be replicated via using the same seed. For the unrestricted ℓ_0-sensitivity with known domain setting, we design an algorithm that combines the Binary Mechanism, the Exponential Mechanism, and the Sparse Vector technique from Dwork et al. (2009) to continually release the top-k. We also show that the more practical version can closely match the error from the less practical version with access to the full event stream (see supplementary file). In the case when each event consists of at most Δ_0 items from an unknown set (restricted ℓ_0-sensitivity with unknown domain), we develop an algorithm that can be viewed as a combination of `UnkGauss` (a variant of $\texttt{LimitDom}_{\texttt{Lap}}$ (Durfee and Rogers, 2019) with an improved privacy guarantee) for one-shot analytics and the Binary Mechanism (Chan et al., 2011).

We now summarize our contributions, also shown in Table 2. First, we develop a general way to ap-

ply existing one shot DP top-k algorithms for the continual observation setting. Second, we design more practical continual observation DP algorithms for the restricted ℓ_0-sensitivity with unknown domain (`UnkBase`) and for the unrestricted ℓ_0-sensitivity with known domain (`SparseGumb`). Third, we present a unified argument for the privacy analyses in both `UnkBase` and `UnkGauss` that improves on prior analysis of $\texttt{LimitDom}_{\texttt{Lap}}$ from Durfee and Rogers (2019), which might be of independent interest.

2 PRELIMINARIES

Since we will provide event level privacy guarantees we define neighboring histograms as follows. Two streams $\omega_{1:T}$ and $\omega'_{1:T}$ are neighboring if for some $t \in [T] := \{1, \cdots, T\}$, $\omega_t \neq \omega'_t$ where $\omega_t = \emptyset$ or $\omega'_t = \emptyset$ but $\omega_{t'} = \omega'_{t'}$ for all $t' \in [T]$ such that $t \neq t'$. We will denote a histogram $\mathbf{h} = \{(h^u, u) : u \in \mathcal{U}, h^u \in \mathbb{N}\}$ to include counts and labels in $\mathcal{U}$. Given stream $\omega_{1:t}$ we define its histogram over $\mathcal{U}$ as $\mathbf{h}(\omega_{1:t}; \mathcal{U}) := \left\{ \left(h^u_t := \sum_{\ell=1}^{t} \mathbb{1}\{u \in \omega_\ell\}, u \right) : u \in \mathcal{U} \right\}$. We will refer to ϵ as the privacy loss parameter in the definition of DP.

Definition 2.1 (Dwork et al. (2006b), Dwork et al. (2006a)). *A randomized algorithm $M : \mathcal{X} \to \mathcal{Y}$ that maps input set $\mathcal{X}$ to some arbitrary outcome set $\mathcal{Y}$ is (ϵ, δ)-DP if for any neighboring datasets x, x' and outcome sets $S \subseteq \mathcal{Y}$, $\Pr[M(x) \in S] \leq e^\epsilon \Pr[M(x') \in S] + \delta$. When $\delta = 0$, we typically say that M is ϵ-DP or* pure *DP.*

The analysis of our algorithms will typically use a variant of DP called *zero-mean Concentrated DP* (zCDP) from Bun and Steinke (2016), which provides tighter composition bounds than traditional DP analysis.

We will state our privacy guarantees in terms of zCDP or DP. We can then convert zCDP to DP and back with the following result.

Lemma 2.1. *[Bun and Steinke (2016)] If M is (ϵ, δ)-DP then it is δ-approximate $\epsilon^2/2$-zCDP. If M is δ-approximate ρ-zCDP then M is also $(\epsilon(\rho, \delta'), \delta + \delta')$-DP where*

$$\epsilon(\rho, \delta') := \rho + 2\sqrt{\rho \ln(1/\delta')}. \tag{1}$$

3 BINARY MECHANISM: RESTRICTED ℓ_0-SENSITIVITY WITH KNOWN DOMAIN

The classical Binary Mechanism from Chan et al. (2011) provides a running count from a bit stream $\sigma_{1:T}$ where $\sigma_t \in \{0, 1\}$. The Binary Mechanism works by maintaining a binary tree and adding the t-th event from the stream into the t-th leaf. As this is done, one has to make sure the sum at each node is equal to the sum of its children. To compute the private count at t it suffices to add the (noisy) sums corresponding to step t. We map the tree of partial sums into a *partial sum table* $\mathbf{p}$ with entries $p_{i,j}$ for $i \in [\log_2(T)], j \in [T/2^{i-1}]$. The Binary Mechanism has multiple applications, including private matchings (Hsu et al., 2014), congestion games (Rogers and Roth, 2014), and private online learning (Guha Thakurta and Smith, 2013), (Cardoso and Cummings, 2019). The Binary Mechanism forms the basis for the new algorithms we present in the following sections, so we present it here.

Due to recent work comparing the overall privacy loss for Laplace noise and Gaussian noise from Cesar and Rogers (2020) and Canonne et al. (2020), we will use Gaussian noise, rather than Laplace noise in the original algorithm. Further, we note that there is nothing special with using a binary representation, so we will keep the base r arbitrary and optimize the base for the lowest overall variance subject to a given privacy level. Considering arbitrary bases for the Binary Mechanism was also considered in Qardaji et al. (2013), although they optimize for the mean squared error and we consider the worst error on any count. To help ease notation, we write

$$L_r := \lfloor \log_r(T) \rfloor + 1. \tag{2}$$

Let $s_j(t; r) \in \{0, 1, \cdots r-1\}$ be the jth digit in the representation of t with base r, i.e. $t = \sum_{j=0}^{\lfloor \log_r(t) \rfloor} s_j(t; r) r^j$. Keeping the base r arbitrary, we now present the generalized version of the Binary Mechanism in Algorithm 1, which we refer to as `BaseMech`.

We now state the privacy and utility guarantees of `BaseMech`. Due to a lack of space, some algorithms and all proofs can be found in the supplementary file.

Theorem 1. *For any base $r \in \{2, \cdots, T\}$, the $\texttt{BaseMech}(\sigma_{1:T}; \tau, r)$ is $\frac{1}{2\tau^2}$-zCDP.*

Proof. The proof follows the same argument as in Chan et al. (2011). Rather than outputting the noisy counts, we instead consider outputting the entire table of partial counts $\mathbf{p} := \{p(j, \ell) + Z_{j,\ell} : j \in [L_r], \ell \in [T]\}$, where $\{Z_{j,\ell}\} \overset{i.i.d.}{\sim} \mathtt{N}\left(0, L_r \tau^2\right)$. Let $\sigma_{1:T}$ and $\sigma'_{1:T}$ be two neighboring streams with partial sum tables $\mathbf{p} = \{p_{j,\ell}\}$ and $\mathbf{p}' = \{p'_{j,\ell}\}$, respectively. Due to the way we defined neighbors, these two partial sum counts can differ in at most L_r cells and can differ in each cell by at most 1. Hence, the ℓ_2-sensitivity of the partial sum table is at most L_r, and by adding $\mathtt{N}\left(0, L_r \tau^2\right)$ to each cell's

Algorithm 1 `BaseMech`; Return a running count

Input: Stream $\sigma_{1:T} = \sigma_1, \cdots, \sigma_T$, where $\sigma_t \in \{0,1\}$, noise level τ, and base $r \in \{2, \cdots, T\}$.
Output: Noisy counts $\hat{y}_{1:T} = \hat{y}_1, \cdots, \hat{y}_T$, $\hat{y}_t \in \mathbb{R}$ for all $t \in [T]$

Sample $\{Z_{i,j} : i \in [L_r], \text{ and } j \in [T]\} \overset{i.i.d.}{\sim} \mathtt{N}\left(0, L_r\tau^2\right)$
for $i \in [L_r]$ **do** ▷ Populate the partial sum table
  $\mathtt{START} = 1$
  for $j \in [\lfloor T/r^{i-1} \rfloor]$ **do**
    $\mathtt{END} = \mathtt{START} + r^{i-1} - 1$
    $p_{i,j} = \sum_{\ell=\mathtt{START}}^{\mathtt{END}} \sigma_\ell$
    $\mathtt{START} = \mathtt{END} + 1$.
for $t = 1, \cdots, T$ **do**
  Let $t = \sum_{j=0}^{\lfloor \log_r(t) \rfloor} s_j(t;r) r^j$, $t' = t$, and $\hat{y}_t = 0$
  while $t' > 0$ **do** ▷ fetch partial sums
    $i \leftarrow \min\{j : s_j(t';r) \neq 0\}$
    $\hat{y}_t \leftarrow \hat{y}_t + \sum_{\ell=t'/r^i - s_j(t;r)+1}^{t'/r^i} (p_{i,\ell} + Z_{i,\ell})$
    $t' \leftarrow t' - s_i(t;r) \cdot r^i$
Return $\hat{y}_{1:T} = \hat{y}_1, \cdots, \hat{y}_T$.

true count ensures $\frac{1}{2\tau^2}$-zCDP (see Lemma 2.5 in Bun and Steinke (2016)). □

We now present the utility guarantee of the `BaseMech` for any base r, which follows from tail bounds of Gaussian random variables.

Theorem 2. *For any $r \in \{2, \cdots, T\}$ and any time $t \in [T]$, the true count y_t and $\hat{y}_t$ from $\mathtt{BaseMech}(\sigma_{1:T}; \tau, r)$ satisfies the following for any $\eta > 0$*

$$\Pr[|\hat{y}_t - y_t| \geq \eta] \leq \Pr[|\mathtt{N}\left(0, (r-1)L_r^2\tau^2\right)| \geq \eta] \leq 2\exp\left(\frac{-\eta^2}{2(r-1)L_r^2\tau^2}\right).$$

Proof. The first inequality in the lemma holds since in the worst case, $\mathtt{BaseMech}(\sigma_{1:T}; \tau, r)$ will add at most $(r-1)L_r$ i.i.d samples from $\mathtt{N}\left(0, L_r\tau^2\right)$. The second inequality holds since for any $\eta > 0$, we have $\Pr[\mathtt{N}\left(0, \sigma^2\right) > \eta] \leq \exp\left(-\frac{\eta^2}{2\sigma^2}\right)$. □

Given T, base r can be selected in a way to minimize the overall variance of any single count, i.e.

$$r^* := \underset{r \in \{2, \cdots, T\}}{\operatorname{argmin}} \left\{(r-1)L_r^2\right\}. \tag{3}$$

In Figure 1 we plot the resulting standard deviation of noise with various bases r and compare it with what we would get by using base $r = 2$ as in the original Binary Mechanism. Note that it looks like we can reduce noise by about 15% at the same level of privacy and for most practical settings $r^* \in \{3, \cdots, 10\}$. Note

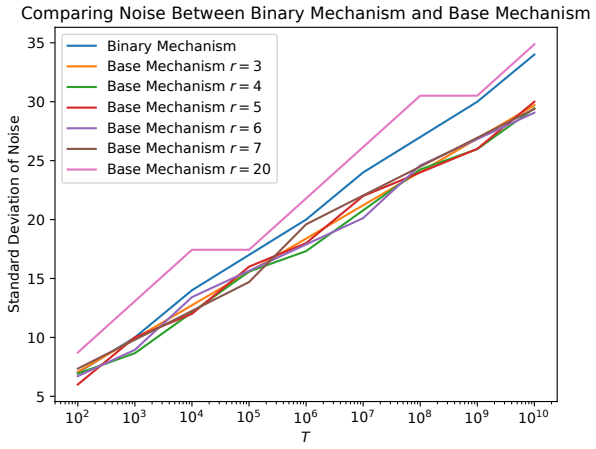

Figure 1: Comparison of the scale of the noise required in the classical Binary Mechanism and `BaseMech` with various choices of base r.

that the optimal choice of r is pretty stable, so that even if a gross upper bound T is used on the event stream, the true optimal base will not change very much. In our algorithms, we will keep the choice of base r as arbitrary and remove its dependence in the later algorithms since it will not impact the privacy claims.

In the streaming setting, when it is known that a user can only modify a limited number of counts at each round t, i.e. $|\omega_t| \leq \Delta_0$ and the domain $\mathcal{U}$ is known in advance, we can simply apply a stream of counts for each domain item. This setting was considered in Chan et al. (2011), and we provide the mechanism `KnownBase` in Algorithm 2.

Algorithm 2 `KnownBase`; Return a running histogram

Input: $\omega_{1:T}$, with $\omega_t \subseteq \mathcal{U}$, $r \in \{2, \cdots, T\}$, and τ.
Output: Noisy histograms $\hat{\mathbf{h}}_{1:T} = \hat{\mathbf{h}}_1, \cdots, \hat{\mathbf{h}}_T$.
for $u \in \mathcal{U}$ **do**
  Define $\sigma_t^u = \mathbb{1}\{u \in \omega_t\}$ for all $t \in [T]$
  $\hat{h}_{1:T}^u = \mathtt{BaseMech}(\sigma_{1:T}^u; \tau, r)$
Return $\hat{\mathbf{h}}_{1:T} = \left(\left\{(\hat{h}_t^u, u) : u \in \mathcal{U}\right\} : t \in [T]\right)$.

We then have the following privacy guarantee, which follows from the analysis in Chan et al. (2011) and composition of zCDP mechanisms (Bun and Steinke, 2016).

Lemma 3.1. *For streams $\omega_{1:T}$ such that $|\omega_t| \leq \Delta_0$ for each $t \in [T]$, $\mathtt{KnownBase}(\cdot; \tau)$ is $\frac{\Delta_0}{2\tau^2}$-zCDP.*

Proof. Let $\omega_{1:T}$ and $\omega'_{1:T}$ be two neighboring streams

where there is a round t where $\omega_t \neq \omega'_t$ where w.l.o.g. $|\omega_t| \leq \Delta_0$ and $\omega'_t = \emptyset$, while $\omega_{t'} = \omega'_{t'}$ for all $t' \neq t$. Hence, there can be at most Δ_0 many items $u \in \mathcal{U}$ such that $\sigma^u_{1:T} \neq \sigma'^u_{1:T}$, while all other streams are identical. Hence, we need only consider the total privacy of Δ_0 many instances of `BaseMech`, which is each $\frac{1}{2\tau^2}$-zCDP. Applying composition of zCDP mechanisms gives the result. □

4 META ALGORITHM FOR CONTINUAL OBSERVATION

In this section we propose a general scheme to return privatized histogram results in the various settings given in Table 1 but in the continual observation setting. Before describing the general scheme we briefly describe how the one-shot algorithms work. The known domain algorithms can be summarized as adding either Gaussian noise with standard deviation τ, then returning the list of items and their counts or adding Gumbel noise with scale $\tau/2$ and taking the top-k results then adding fresh Gaussian noise with standard deviation τ to those discovered items' counts. Note that the Exponential Mechanism can be implemented by adding Gumbel noise to counts and then returning the element with the largest noisy count. Further, the Exponential Mechanism with privacy parameter ϵ satisfies a property called *bounded range* (Durfee and Rogers, 2019), which results in $\frac{1}{8\epsilon^2}$-zCDP (Cesar and Rogers, 2020).

The unknown domain algorithms can be thought of as the same as the known domain algorithms, except we only have access to the top-$(\bar{k}+1)$ items for $\bar{k} \geq k$ from the full histogram of size d and we include a noisy threshold that will depend on the privacy parameters, so that only items above the noisy threshold will be shown. For completeness we present the pseudocode and privacy guarantees for each of the various algorithms in the supplementary file, except for $\texttt{UnkGauss}^{\bar{k}}$, which we analyze in a later section.

The key observation is that we can generalize the partial sum table from Section 3 to a *partial histogram table* were each entry contains the histogram formed by the corresponding substream from $\omega_{1:T}$. Depending on what setting from Table 1 we are in, we apply the corresponding one-shot DP algorithm to each cell j, ℓ for $j \in [\log_r(T)], \ell \in [T/r^{j-1}]$ of the partial histogram table, and aggregate the corresponding noisy histograms to provide a private result at time t. The pseudocode of `MetaAlgo` is in the supplementary file.

We state the various privacy guarantees in terms of the noise level τ and other parameters. The analysis follows by zCDP composition over at most L_r cells that can change in neighboring partial histogram tables.

Theorem 3. *If we have ℓ_0-sensitivity Δ_0 and known domain, then `MetaAlgo` is $\frac{\Delta_0}{2\tau^2}$-zCDP. If we have unrestricted ℓ_0-sensitivity and known domain, then `MetaAlgo` is $\frac{k}{\tau^2}$-zCDP. If we have ℓ_0-sensitivity Δ_0 and unknown domain, then `MetaAlgo` is $\left(\epsilon(\frac{\Delta_0}{2\tau^2}, \delta'), \Delta_0\delta + \delta'\right)$-DP for any $\delta' > 0$, where $\epsilon(\cdot,\cdot)$ is given in (1). If we have unrestricted ℓ_0-sensitivity and unknown domain, then `MetaAlgo` is $\left(\epsilon(\frac{k}{\tau^2}, \delta'), 2k\delta + \delta'\right)$-DP for any $\delta' > 0$, where $\epsilon(\cdot,\cdot)$ is given in (1).*

The main drawback with this meta-algorithm is that in order to implement it at a time t, we will need to know the full stream of events, so that we can apply each DP algorithm on different subsequences, which then need to be stored for later calculations. In the following sections we explore settings where our algorithms only have access to the aggregated histogram up to time t at each round, rather than the full stream of events.

5 UNRESTRICTED ℓ_0-SENSITIVITY, KNOWN DOMAIN

We now consider the case where there is no limit to how many items a user can contribute for a given event in a stream, unlike in Section 3 where the bound was Δ_0. To ensure there is some bound on privacy, we only display the top-k results at round t, which are computed based on all events that have occurred in the stream up to that round. This is particularly useful when no preprocessing of the data is in place to restrict the number of items for each event, yet we still want to ensure some bounded level of privacy, even for event level. Otherwise, we would need to add noise that scales with $|\mathcal{U}| = d$ due to users possibly contributing an arbitrary number of items.

Our algorithm consists of multiple classical DP algorithms, which makes the privacy analysis somewhat standard. Consider the case when we want to return the top-1 item at every round. Given a data generating distribution, one would expect that the top-1 item would not change very many times in a stream of events. Hence, we introduce a parameter s, which is the number of *switches* the algorithm is allowed to have. A switch takes place when a new item has count significantly larger than the currently selected one. We will add noise in our algorithm that scales with s.

We now discuss the algorithm at a high level, and present the full algorithm in the supplementary file. At the first round, we will want to find the item with the top count, which can be done with the Exponential Mechanism (McSherry and Talwar, 2007), i.e.

$\texttt{KnownGumb}$[1] with only the items returned, not their counts. Recall that we are in the known domain setting, so we will have the same domain at each round, which consists of d items. Finding this top item will cost a single unit of privacy loss in our composition, despite one user being able to have a set of items the size of the full domain. Once we have the top selected item, we can use the $\texttt{BaseMech}$ algorithm to produce a running count for this selected item. However, we need to check the counts of other items at each round to see if there is one with higher count. For this, we will use the Sparse Vector technique from Dwork et al. (2009) to continually check whether there is an item with larger count than the currently selected item. We will only switch the top item if there is another item with count $\eta_t \geq 0$ more that the currently selected item's count at round t. We can then set η_t for a given utility level and it does not impact the privacy analysis, so we keep it arbitrary here. Once we find that there is an item with larger count than the current top item, we will then use the Exponential Mechanism again to find a new top item, and continually return counts for the new item using $\texttt{BaseMech}$. By the end of the stream, we know that there can be at most $s+1$ many items with counts from $\texttt{BaseMech}$. It is then easy to generalize this idea to allow for top-k results at each round, rather than top-1. We call this generalization $\texttt{SparseGumb}^{s,k}(\omega_{1:T}; \{\eta_t\}, \tau)$ and it has the following privacy guarantee.

Theorem 4. *For any $\{\eta_t\}_{t=1}^T$ with $\eta_t \geq 0$ for all $t \in [T]$, $\texttt{SparseGumb}^{s,k}(\cdot; \{\eta_t\}, \tau)$ is $\frac{k+2}{\tau^2}$-zCDP.*

In the supplementary file we evaluate via simple experiments the utility of $\texttt{SparseGumb}^{s,k}(\cdot; \{\eta_t\}, \tau)$. As can be expected, given a fixed privacy budget, it does not beat $\texttt{MetaAlgo}$ in the same setting. However, with the right parameters it can get close to $\texttt{MetaAlgo}$, with the advantage of just requiring the histogram at round t instead of the full stream of events which is obviously beneficial in the real-time analytics setting described in the introduction, since we do not need to apply various DP algorithms to different substreams.

6 REVISITING ONE SHOT RESTRICTED ℓ_0-SENSITIVITY AND UNKNOWN DOMAIN

In this section we present a new one-shot algorithm, $\texttt{UnkGauss}^{\bar{k}}$ for the restricted ℓ_0-sensitivity and unknown domain setting. The first algorithm, $\texttt{LimitDom}_{\texttt{Lap}}$, for this setting when only a limited number of top elements are available in the true histogram was developed by Durfee and Rogers (2019). Through a new analysis (which we will also use in Section 7) we show that $\texttt{UnkGauss}^{\bar{k}}$ attains a better privacy guarantee than $\texttt{LimitDom}_{\texttt{Lap}}$ with the same level of noise. The algorithm is simple; given access to the $(\bar{k}+1)$ highest ranked elements in the histogram, it adds Gaussian noise to each element and releases only those with noisy counts above a threshold $h_\perp := h_{(k+1)} + 1 + \sqrt{2}\tau\Phi^{-1}(1-\delta)$ with $\delta > 0$, which we label as $\perp$, and also has noise added to it. The pseudocode of $\texttt{UnkGauss}^{\bar{k}}(\cdot; \tau, \delta)$ can be found in the supplementary file.

Theorem 5. *For histograms with ℓ_0-sensitivity Δ_0 and ℓ_∞-sensitivity 1, $\texttt{UnkGauss}^{\bar{k}}(\cdot; \tau, \delta)$ is $(\epsilon(\frac{\Delta_0}{2\tau^2}, \delta'), \Delta_0\delta + \delta')$-DP for any $\delta' > 0$ with $\epsilon(\cdot,\cdot)$ in (1).*

We now describe the proof technique used to prove Theorem 5, which will also be used in Section 7 to analyze our more practical DP algorithm for the continual observation setting with ℓ_0-sensitivity Δ_0 and unknown domain. We first set up some notation. Let M be a mechanism that takes input dataset (histogram) $\mathbf{h}$ to some arbitrary outcome space. For any two histograms $\mathbf{h}^{(0)}$ and $\mathbf{h}^{(1)}$, we define the *good* outcome sets $\mathcal{G}_M$, as outcomes that can occur with input $\mathbf{h}^{(0)}$ and $\mathbf{h}^{(1)}$ and the bad outcome sets $\mathcal{B}_M^b$ for $b \in \{0,1\}$, as outcomes of M that can occur with input $\mathbf{h}^{(b)}$ but not $\mathbf{h}^{(1-b)}$.

The following result allows us to determine the privacy of a particular mechanism by analyzing the privacy of a related mechanism with access to both neighboring datasets.

Lemma 6.1. *Let $\mathbf{h}^{(0)}$ and $\mathbf{h}^{(1)}$ be two neighboring datasets. Suppose there exists a mechanism $A(b; \mathbf{h}^{(0)}, \mathbf{h}^{(1)})$ where $b \in \{0,1\}$ such that for any outcome set $S \subseteq \mathcal{G}_M$, we have $\Pr[M(\mathbf{h}^{(b)}) \in S] = \Pr[A(b; \mathbf{h}^{(0)}, \mathbf{h}^{(1)}) \in S]$. Further, suppose that $\Pr[M(\mathbf{h}^{(b)}) \in \mathcal{B}_M^b] \leq \delta$ for $b \in \{0,1\}$. If $A(\cdot; \mathbf{h}^{(0)}, \mathbf{h}^{(1)})$ is (ϵ, δ')-DP, then M is $(\epsilon, \delta + \delta')$-DP.*

Hence, to prove the privacy of $\texttt{UnkGauss}^{\bar{k}}$, we show that bad outcomes occur with negligible probability and that there is a mechanism on shared outcomes of neighboring datasets that is DP. Note that the parameter $\bar{k}$ in $\texttt{UnkGauss}^{\bar{k}}$ means that we only have the top-$(\bar{k}+1)$ elements available from the original histogram. It might be the case that $\bar{k}$ is larger than the number of elements in the histogram that actually have positive count. Hence, $\texttt{UnkGauss}^{\bar{k}}$ might add noise to fewer than $\bar{k}+1$ elements.

Consider a slight variant of $\texttt{UnkGauss}^{\bar{k}}$, which we denote as $\texttt{UnkGauss}^{\bar{k}}_{\top}$, that pads the histogram with zero counts and labels $\{\top_i\}$ to ensure that there are exactly $\bar{k}+1$ many elements to add noise to. The next lemma shows that adding noise to these *dummy* elements but then dropping them from the outcome is the same as simply not even considering these dummy elements to

begin with.

Lemma 6.2. *Let $\mathbf{h} = \{(h_u, u) : u \in \mathcal{U}\}$ be a histogram with labels for $|\mathcal{U}| = p$ elements and $M(\mathbf{h})$ return $\bar{k}+1$ counts with labels for $\bar{k} \geq p$ with noise from some distribution $\mathcal{P}$ where $\{Z_i\} \overset{i.i.d.}{\sim} \mathcal{P}$ and*

$$\begin{aligned} M(\mathbf{h}) = &\{(h_u + Z_u, u) : u \in \mathcal{U}\} \\ &\cup \{(h_\perp + Z_\perp, \perp)\} \\ &\cup \{(Z_{\top_j}, \top_j) : j \in \{1, \cdots, \bar{k} - p\}\}. \end{aligned}$$

Let $M'(\mathbf{h})$ drop elements with counts lower than $\perp$ and then drop any element with label in $\{\top_i\}$. Let $\hat{M}$ be the mechanism that adds i.i.d. noise from $\mathcal{P}$ to only counts in $\mathbf{h}$ and $\hat{M}'$ drop elements with counts lower than the count labeled $\perp$. Then $M'(\mathbf{h})$ is equal in distribution to $\hat{M}'(\mathbf{h})$.

Therefore, we prove the privacy of $\texttt{UnkGauss}^{\bar{k}}_{\top}$, rather than $\texttt{UnkGauss}^{\bar{k}}$, since the latter is equal in distribution to a post-processing function of the former and cannot increase the privacy loss of $\texttt{UnkGauss}^{\bar{k}}_{\top}$. Our privacy analysis consists of analyzing the Gaussian Mechanism and bounding bad events, i.e. events that cannot occur in both neighboring histograms. We define two domains of labels from a given histogram $\mathbf{h} = \{(h_u, u) : u \in \mathcal{U}\}$ with ordered indices $h_{i_{(1)}} \geq h_{i_{(2)}} \geq \cdots \geq h_{i_{(d)}}$. The first only considers elements with positive count and the second pads the domain with zero counts and *dummy* labels:

$$\mathcal{D}^{\bar{k}}(\mathbf{h}) := \{i_{(j)} \in \mathcal{U} : j \leq \bar{k} \text{ and } h_{i_{(\bar{k})}} > 0\} \cup \{\perp\}$$

$$\mathcal{D}^{\bar{k}}_{\top}(\mathbf{h}) := \begin{cases} \{i_{(j)} \in \mathcal{U} : j \leq \bar{k}\} \cup \{\perp\}, \text{ if } h_{i_{(\bar{k})}} > 0 \\ \{i_{(j)} \in \mathcal{U} : j \leq p\} \cup \{\top_1, \cdots \top_{\bar{k}-p}\} \cup \{\perp\}, \\ \qquad\qquad \text{if } h_{i_{(p)}} > h_{i_{(p+1)}} = 0 \end{cases}$$

Note that the labels $\{\top_j\}$ in $\mathbf{h}$ do not exist, and so for any index $\top_j$, its count is $h_{\top_j} = 0$. Consider the Gaussian mechanism $\texttt{GaussMech}^{\bar{k}}_{\perp}(b; \mathbf{h}^{(0)}, \mathbf{h}^{(1)}, \tau)$ that takes a bit $b \in \{0,1\}$ and two neighboring histograms $\mathbf{h}^{(0)}$ and $\mathbf{h}^{(1)}$ with noise added to the top-$(\bar{k}+1)$ elements from each histogram. Because the labels need not be the same in the top-$(\bar{k}+1)$ in $\mathbf{h}^{(0)}$ and $\mathbf{h}^{(1)}$, we assign a common label to the differing *bad* indices, denoted as $\{B_\ell : \ell = 1, \cdots, |\mathcal{D}^{\bar{k}}_{\top}(\mathbf{h}^{(0)}) \setminus \mathcal{D}^{\bar{k}}_{\top}(\mathbf{h}^{(1)})|\}$.

Note that once we fix neighboring histograms, $\texttt{GaussMech}^{\bar{k}}_{\perp}(\cdot; \mathbf{h}^{(0)}, \mathbf{h}^{(1)}, \tau)$ is simply the Gaussian mechanism on a new histogram $\mathbf{v}^{(b)}$ that uses the counts from $\mathbf{h}^{(b)}$ but whose labels include the common labels from $\mathbf{h}^{(0)}$ and $\mathbf{h}^{(1)}$, including the $\perp$ element and dummy elements $\{\top_j\}$, as well as the bad indices $\{B_j\}$. Hence, we want to show that $\mathbf{v}^{(0)}$ and $\mathbf{v}^{(1)}$ can differ in at most Δ_0 bins, i.e. the ℓ_0-sensitivity of $\mathbf{h}^{(b)}$, and in any bin that changes, the counts can differ by at most 1, i.e. the ℓ_∞-sensitivity of $\mathbf{h}^{(b)}$. We know that for any $j \in \mathcal{D}^{\bar{k}}_{\top}(\mathbf{h}^{(0)}) \cap \mathcal{D}^{\bar{k}}_{\top}(\mathbf{h}^{(1)})$ that $|h^{(0)}_j - h^{(1)}_j| \leq 1$ and hence $|v^{(0)}_j - v^{(1)}_j| \leq 1$. We now consider the differing labels.

Lemma 6.3. *Let $\mathbf{h}^{(0)}$ and $\mathbf{h}^{(1)}$ be neighbors with ℓ_∞-sensitivity 1. For any $i \in \mathcal{D}^{\bar{k}}_{\top}(\mathbf{h}^{(b)}) \setminus \mathcal{D}^{\bar{k}}_{\top}(\mathbf{h}^{(1-b)})$ and $j \in \mathcal{D}^{\bar{k}}_{\top}(\mathbf{h}^{(1-b)}) \setminus \mathcal{D}^{\bar{k}}_{\top}(\mathbf{h}^{(b)})$ we have $|h^{(b)}_i - h^{(1-b)}_j| \leq 1$. Furthermore, $|h^{(b)}_{(\bar{k}+1)} - h^{(1-b)}_{(\bar{k}+1)}| \leq 1$.*

We now show (in the supplementary) that the ℓ_0-sensitivity between $\mathbf{v}^{(0)}$ and $\mathbf{v}^{(1)}$ is the same as between $\mathbf{h}^{(0)}$ and $\mathbf{h}^{(1)}$.

Lemma 6.4. *If $\mathbf{h}^{(0)}$ and $\mathbf{h}^{(1)}$ differ in at most Δ_0 bins, then $\mathbf{v}^{(0)}$ and $\mathbf{v}^{(1)}$ differ in at most Δ_0 bins.*

With these two results we have the following, which follows directly from the sensitivity analysis of the intermediate histogram $\mathbf{v}^{(b)}$ from Lemmas 6.3 and 6.4.

Lemma 6.5. *For any two neighboring histograms $\mathbf{h}^{(0)}$ and $\mathbf{h}^{(1)}$ with ℓ_0-sensitivity Δ_0 and ℓ_∞-sensitivity 1, the procedure $\texttt{GaussMech}^{\bar{k}}_{\perp}(\cdot; \mathbf{h}^{(0)}, \mathbf{h}^{(1)}, \tau)$ is $\frac{\Delta_0}{2\tau^2}$-zCDP*

We now show that, for a pair of fixed neighboring datasets, $\texttt{UnkGauss}^{\bar{k}}$ is equivalent to running a post processing function on $\texttt{GaussMech}^{\bar{k}}_{\perp}$ for certain outcomes and that we can bound the probability of other outcomes where they do not align. We now define *good* and *bad* outcome sets.

Definition 6.1. *Given neighbors $\mathbf{h}^{(0)}, \mathbf{h}^{(1)}$, we define $\mathcal{S}^{(b)}_{\texttt{Gauss}}$ as the outcomes of $\texttt{UnkGauss}^{\bar{k}}(\mathbf{h}^{(b)}; \tau, \delta)$. We then write* bad outcomes *as $\mathcal{B}^{(b)}_{\texttt{Gauss}} := \mathcal{S}^{(b)}_{\texttt{Gauss}} \setminus \mathcal{S}^{(1-b)}_{\texttt{Gauss}}$, for $b \in \{0,1\}$.*

Next, we bound the probability of outputting something in $\mathcal{B}^{(b)}_{\texttt{Gauss}}$, and also show that we can achieve DP for the remaining outputs that are common in $\mathbf{h}^{(0)}$ and $\mathbf{h}^{(1)}$. For bounding the bad outcomes, it suffices to consider each element in $\mathcal{D}^{\bar{k}}_{\top}(\mathbf{h}^{(b)}) \setminus \mathcal{D}^{\bar{k}}_{\top}(\mathbf{h}^{(1-b)})$ and bound the probability that its respective noisy value is above a threshold $h_\perp$ with added noise. Note that the threshold computation will have a simpler analysis than prior work due to the sum of two Gaussians being Gaussian, whereas Durfee and Rogers (2019) considered Laplace noise which does not satisfy the same property.

Lemma 6.6. *For neighboring $\mathbf{h}^{(0)}, \mathbf{h}^{(1)}$ with ℓ_0-sensitivity Δ_0 and ℓ_∞-sensitivity 1, with $b \in \{0,1\}$ we have $\Pr[\texttt{UnkGauss}^{\bar{k}}(\mathbf{h}^{(b)}; \tau, \delta) \in \mathcal{B}^{(b)}_{\texttt{Gauss}}] \leq \delta\Delta_0$.*

We can now prove Theorem 5 using Lemma 6.1. From Lemma 6.6, we have the probability of bad outcomes being negligible. We now need to define a mechanism $A(b; \mathbf{h}^{(0)}, \mathbf{h}^{(1)})$ that matches $\texttt{UnkGauss}^{\bar{k}}$ on good outcomes and is DP. Lemma 6.5 shows

that $\texttt{GaussMech}_{\perp}^{\bar{k}}(\cdot; \mathbf{h}^{(0)}, \mathbf{h}^{(1)}, \tau)$ is DP. We then define a post processing function on $\texttt{GaussMech}_{\perp}^{\bar{k}}$. First, we sort in descending order the elements up until we get label $\perp$ and then we eliminate the rest. Next, we drop all the dummy labels $\{\top_i\}$ and their noisy counts. We know from Lemma 6.2 that sorting up to $\perp$ and dropping the dummy labels is equivalent to never considering the dummy elements in the first place. Note that this post processing function on $\texttt{GaussMech}_{\perp}^{\bar{k}}(b; \mathbf{h}^{(0)}, \mathbf{h}^{(1)}, \tau)$ is equivalent to our main algorithm $\texttt{UnkGauss}^{\bar{k}}$ on good outcomes. Because post-processing cannot increase the privacy loss parameters, we can use Lemma 6.1 with $A(b; \mathbf{h}^{(0)}, \mathbf{h}^{(1)})$ as this post-processing function of $\texttt{GaussMech}_{\perp}^{\bar{k}}(b; \mathbf{h}^{(0)}, \mathbf{h}^{(1)}, \tau)$.

7 RESTRICTED ℓ_0-SENSITIVITY AND UNKNOWN DOMAIN

We turn back to the continual observation setting where a user can contribute at most Δ_0 many items at any round, but the domain of items is unknown. When the domain is not given in advance, it is impossible for an item that no one contributed to in the stream to actually be returned. However, it is important to point out that the mere existence of a particular item shows that someone in the dataset must have contributed such an item. We will then impose a threshold so that the probability that we display an item with a single count is very small. We emphasize that even if the domain were known in advance, it still might be desirable to consider this setting, since the domain might be incredibly large making `KnownBase` computationally expensive.

Algorithm 3 `UnkBase` ; Return a running histogram

Input: $\omega_{1:T} = \omega_1, \cdots, \omega_T$, base r, noise scale τ, δ.
Output: Noisy histograms $\hat{\mathbf{h}}_{1:T}$.
for $t \in [T]$ **do**
  $\mathcal{D}_t = \emptyset, \mathcal{Z} = \emptyset$,
  Let $\mathcal{U}_t$ be the set of items in $\omega_{1:t}$
  for $u \in \mathcal{U}_t$ **do**
    Let $\mathcal{I}_t(r)$ be the cells (j, ℓ) in the r-nary tree used in the representation of t with base r
    for each cell $(j, \ell) \in \mathcal{I}_t(r)$ **do**
      if $(u, j, \ell) \notin \mathcal{Z}$ **then**
        Let $Z_{j,\ell}^u \sim \mathtt{N}\left(0, L_r \tau^2\right)$,
        $\mathcal{Z} \leftarrow \mathcal{Z} \cup \{(u, j, \ell)\}$
    $\hat{h}_t^u = h_t^u + \sum_{(j,\ell) \in \mathcal{I}_t(r)} Z_{j,\ell}^u$
    if $\hat{h}_t^u > m_\delta$ from (4) **then**
      $\mathcal{D}_t \leftarrow \mathcal{D}_t \cup \{u\}$
Return $(\{(\hat{h}_t^u, u) : u \in \mathcal{D}_t\} : t \in [T])$.

We present the main algorithm of this section in Algorithm 3.[2] We can summarize `UnkBase` as simply taking the items, denoted as $\mathcal{U}_t$, that have appeared in the stream up to time t, form their current histogram $\mathbf{h}(\omega_{1:t}; \mathcal{U}_t)$, add noise in the way one would in `KnownBase`, but only show items if their noisy count is above the following threshold,

$$m_\delta := \tau L_r \sqrt{r-1} \Phi^{-1}(1 - \delta/T) + 1. \tag{4}$$

Our analysis of `UnkBase` can be thought of as a generalization of the stability based histograms studied in earlier work (Korolova et al., 2009; Bun et al., 2016; Vadhan, 2017), where only elements with positive counts exist in the histogram. Directly applying the stability based histogram approach would result in another variant of the `MetaAlgo`, as the histogram in each cell in the partial sum table would have counts below a certain threshold removed. We opted to using `UnkGauss` in the presentation of the `MetaAlgo` because it is more general than the original stability based histogram approaches, due to it only needing access to the top-$\bar{k}$ counts with positive counts, as opposed to all positive counts.

The novelty of our approach is then in extending the stability based histogram approach to the case where we only have access to positive counts up to and including round $t \in [T]$, rather than in all sub-streams. Lemma 6.1 allows us to consider a DP algorithm with access to a given pair of neighboring datasets x, x' and only consider outcomes that can occur with both neighbors, i.e. *good* outcomes. Hence, we can then consider the two partial sum tables that suffices to compute the running counts for either x or x'. The problem between the two partial sum tables is that there are table cells with elements present for x but not for x'. To address this issue, we introduce zero count elements to each cell, so that each cell has the same number of elements that get noise added to it. Note that the labels of the zero counts need to be made common across the two partial sum tables, which we can do because we are constructing a DP algorithm that knows x and x'. We can then analyze the privacy of this resulting partial sum table using composition of Gaussian mechanisms. The last part in our analysis is to bound the probability of all *bad* outcomes, which in this case is when any of the elements that had zero count in x' yet positive count in x appear in any histogram in any $t \in [T]$, which we can do by applying a threshold that is determined by the tail bound of the sum of at most L_r Gaussians, which itself is Gaussian (another reason to use Gaussian noise!).

[2] The sets $\mathcal{I}_t(r)$ can be built using the same idea as in `BaseMech`, all one needs is the r-nary representation of t. For clarity of exposition we do not build the sets in the pseudocode of `UnkBase`.

We will provide an outline of our analysis, which is detailed in the supplementary file. Note that `UnkBase`, at a high level, is an algorithm that adds Gaussian noise to all items with positive count for each partial histogram table cell in the `BaseMech` subroutine. Hence, each partial histogram table cell will consist of a noisy histogram of all items that actually appeared in the corresponding subsequence of $\omega_{1:T}$. There can be at most $L_r = \lfloor \log_r(T) \rfloor + 1$ many cells of the partial histogram table that have histograms that can differ. Recall that each cell may have differing amounts of items with positive counts, so we will pad each histogram with dummy labels $\{\top_i\}$ so that each cell has the same cardinality of terms. As we did for $\texttt{UnkGauss}^{\bar{k}}$, we will instead analyze a slight variant of `UnkBase`, which we call $\texttt{UnkBase}^{\bar{d}}_{\top}$, that pads the set of items in a cell with dummy items $\{\top_i\}$ that we add noise to in the partial histogram table, so that each cell has the same cardinality $\bar{d}$ of items, where $\bar{d}$ is an upper bound on the full domain size d and is only needed in the analysis. Similar to Lemma 6.2, we show in the supplementary file that simply dropping these dummy items later is equivalent to having never considered them.

To make sure that on neighboring streams we have the same labels for the histogram in each cell, we will consider a similar routine to $\texttt{GaussMech}^{\bar{k}}_{\perp}$ from the previous section, where we denote items in the histogram of each cell as either having common labels (labels that can occur in both neighboring streams), bad labels (labels that can only occur in one neighboring stream), and dummy labels (those with zero count in both streams). We refer to this mechanism with common labels as $\texttt{GaussMech}^{\bar{d}}$, which has the following guarantee.

We then have the following privacy guarantee of $\texttt{GaussMech}^{\bar{d}}$.

Lemma 7.1. *For any two neighboring histograms $\mathbf{h}^{(0)}$ and $\mathbf{h}^{(1)}$ with ℓ_0-sensitivity Δ_0 and ℓ_∞-sensitivity 1, the procedure $\texttt{GaussMech}^{\bar{d}}(\cdot; \mathbf{h}^{(0)}, \mathbf{h}^{(1)}, \tau)$ is $\frac{\Delta_0}{2\tau^2}$-zCDP.*

Proof. Follows the same analysis as in Lemmas 6.3 and 6.4 □

We now show that we can connect $\texttt{GaussMech}^{\bar{d}}$ with our $\texttt{UnkBase}^{\bar{d}}_{\top}$ algorithm on good outcomes, which brings us a step closer to being able to use Lemma 6.1.

Lemma 7.2. *For neighbors $\omega^{(0)}_{1:T}$ and $\omega^{(1)}_{1:T}$ and outcomes $\mathcal{G}_{\texttt{Unk}}$ that can occur in both $\texttt{UnkBase}^{\bar{d}}_{\top}(\omega^{(b)}_{1:T}; \tau)$ for $b \in \{0, 1\}$, there exists an $A(\cdot; \omega^{(0)}_{1:T}, \omega^{(1)}_{1:T})$ that is $\frac{\Delta_0}{2\tau^2}$-zCDP and for any outcome set $S \subseteq \mathcal{G}_{\texttt{Unk}}$ we have $\Pr[\texttt{UnkBase}^{\bar{d}}_{\top}(\omega^{(b)}; \tau) \in S] = \Pr[A(b; \mathbf{h}^{(0)}, \mathbf{h}^{(1)}) \in S]$.*

We are then left showing that outcomes that can only occur in one of the neighboring streams occurs with negligible probability, which we show via bounding the tails of multiple Gaussians with high probability, which is why we use the threshold in (4).

Lemma 7.3. *Fix neighbors $\omega^{(0)}_{1:T}$ and $\omega^{(1)}_{1:T}$ with ℓ_0-sensitivity Δ_0 and define $\mathcal{B}^{(b)}_{\texttt{Unk}}$ to be the set of outcomes that can occur in $\texttt{UnkBase}^{\bar{d}}_{\top}(\omega^{(b)}_{1:T}; \tau; \delta)$ but not in $\texttt{UnkBase}^{\bar{d}}_{\top}(\omega^{(1-b)}_{1:T}; \tau, \delta)$. Then we have $\Pr[\texttt{UnkBase}^{\bar{d}}_{\top}(\omega^{(b)}_{1:T}; \tau, \delta) \in \mathcal{B}^{(b)}_{\texttt{Unk}}] \leq \Delta_0 \delta$.*

We can now state our privacy result, which follows from the privacy of $\texttt{UnkBase}^{\bar{d}}_{\top}$ and recalling that `UnkBase` is a post-processing function of $\texttt{UnkBase}^{\bar{d}}_{\top}$.

Theorem 6. *$\texttt{UnkBase}(\cdot; \tau, \delta)$ is $(\epsilon(\frac{\Delta_0}{2\tau^2}, \delta'), \Delta_0\delta + \delta')$-DP for any $\delta' > 0$ with $\epsilon(\cdot, \cdot)$ given in (1).*

We also show in the supplementary file that with high probability, `UnkBase` does not hide items if their counts are above $m_\delta + c\tau$ for some constant $c > 0$. Additionally, for those items $u \in \mathcal{U}$ that the algorithm releases, we provide bounds on the difference between their true count h^u_t and their noisy count $\hat{h}^u_t$ by noticing that $\hat{h}^u_t$ is the original count h^u_t plus truncated Gaussian noise.

8 CONCLUSION

We have revisited the problem of continually releasing differentially private histograms in the model introduced by Dwork et al. (2010a) and Chan et al. (2011). We considered event level privacy, where events in a stream can consist of multiple elements. We then considered the various DP algorithms for the restricted/unrestricted ℓ_0-sensitivity with known/unknown domain settings. These various settings of releasing privatized histograms was originally introduced in Durfee and Rogers (2019) and Rogers et al. (2020), but not for continual release. We showed that we can use these existing DP algorithms for continual observation, but it required running the DP algorithms on various subsequences of the event streams, which might be prohibitively expensive in run time for many applications. We then presented more practical DP algorithms that take the aggregated counts at each round to return a noisy histogram continually for the unrestricted ℓ_0-sensitivity with unknown domain setting along with the unrestricted ℓ_0-sensitivity with known domain setting. There are multiple open research directions here, such as providing a utility analysis for `SparseGumb`. Furthermore, are there algorithms for the restricted ℓ_0-sensitivity and unknown domain for the practical setting where only the top-$\bar{k}$, rather than all elements with positive counts, as in Durfee and Rogers (2019)?

9 Acknowledgments

We would like the thank the following people for helpful comments throughout this research project: Parvez Ahammad, David Durfee, Souvik Ghosh, Koray Mancuhan, and Diana Negoescu.

References

M. Bun and T. Steinke. Concentrated differential privacy: Simplifications, extensions, and lower bounds. In *Theory of Cryptography Conference (TCC)*, pages 635–658, 2016.

M. Bun, K. Nissim, and U. Stemmer. Simultaneous private learning of multiple concepts. In *ITCS*, 2016.

C. L. Canonne, G. Kamath, and T. Steinke. The discrete gaussian for differential privacy, 2020.

A. R. Cardoso and R. Cummings. Differentially private online submodular minimization. In K. Chaudhuri and M. Sugiyama, editors, *Proceedings of Machine Learning Research*, volume 89 of *Proceedings of Machine Learning Research*, pages 1650–1658. PMLR, 16–18 Apr 2019. URL http://proceedings.mlr.press/v89/cardoso19b.html.

M. Cesar and R. Rogers. Bounding, concentrating, and truncating: Unifying privacy loss composition for data analytics, 2020.

T. H. Chan, E. Shi, and D. Song. Private and continual release of statistics. *ACM Trans. Inf. Syst. Secur.*, 14(3):26:1–26:24, 2011. doi: 10.1145/2043621.2043626. URL https://doi.org/10.1145/2043621.2043626.

T.-H. H. Chan, M. Li, E. Shi, and W. Xu. Differentially private continual monitoring of heavy hitters from distributed streams. In *Proceedings of the 12th International Conference on Privacy Enhancing Technologies*, PETS'12, page 140–159, Berlin, Heidelberg, 2012. Springer-Verlag. ISBN 9783642316791. doi: 10.1007/978-3-642-31680-7_8. URL https://doi.org/10.1007/978-3-642-31680-7_8.

D. Durfee and R. Rogers. Practical differentially private top-k selection with pay-what-you-get composition. *CoRR*, abs/1905.04273, 2019. URL http://arxiv.org/abs/1905.04273.

C. Dwork, K. Kenthapadi, F. McSherry, I. Mironov, and M. Naor. Our data, ourselves: Privacy via distributed noise generation. In *Advances in Cryptology (EUROCRYPT 2006)*, 2006a.

C. Dwork, F. McSherry, K. Nissim, and A. Smith. Calibrating noise to sensitivity in private data analysis. In *Proceedings of the Third Theory of Cryptography Conference*, pages 265–284, 2006b.

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*, STOC '09, page 381–390, New York, NY, USA, 2009. Association for Computing Machinery. ISBN 9781605585062. doi: 10.1145/1536414.1536467. URL https://doi.org/10.1145/1536414.1536467.

C. Dwork, M. Naor, T. Pitassi, and G. N. Rothblum. Differential privacy under continual observation. In *Proceedings of the Forty-Second ACM Symposium on Theory of Computing*, STOC '10, page 715–724, New York, NY, USA, 2010a. Association for Computing Machinery. ISBN 9781450300506. doi: 10.1145/1806689.1806787. URL https://doi.org/10.1145/1806689.1806787.

C. Dwork, M. Naor, T. Pitassi, G. N. Rothblum, and S. Yekhanin. Pan-private streaming algorithms. In A. C. Yao, editor, *Innovations in Computer Science - ICS 2010, Tsinghua University, Beijing, China, January 5-7, 2010. Proceedings*, pages 66–80. Tsinghua University Press, 2010b. URL http://conference.iiis.tsinghua.edu.cn/ICS2010/content/papers/6.html.

A. Guha Thakurta and A. Smith. (nearly) optimal algorithms for private online learning in full-information and bandit settings. In C. J. C. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems*, volume 26, pages 2733–2741. Curran Associates, Inc., 2013. URL https://proceedings.neurips.cc/paper/2013/file/c850371fda6892fbfd1c5a5b457e5777-Paper.pdf.

J. Hsu, Z. Huang, A. Roth, T. Roughgarden, and Z. S. Wu. Private matchings and allocations. In *Proceedings of the Forty-Sixth Annual ACM Symposium on Theory of Computing*, STOC '14, page 21–30, New York, NY, USA, 2014. Association for Computing Machinery. ISBN 9781450327107. doi: 10.1145/2591796.2591826. URL https://doi.org/10.1145/2591796.2591826.

A. Korolova, K. Kenthapadi, N. Mishra, and A. Ntoulas. Releasing search queries and clicks privately. In *Proceedings of the 18th International Conference on World Wide Web*, WWW '09, page 171–180, New York, NY, USA, 2009. Association for Computing Machinery. ISBN 9781605584874. doi: 10.1145/1526709.1526733. URL https://doi.org/10.1145/1526709.1526733.

M. Lyu, D. Su, and N. Li. Understanding the sparse vector technique for differential privacy. *Proc. VLDB Endow.*, 10(6):637–648, Feb. 2017. ISSN

2150-8097. doi: 10.14778/3055330.3055331. URL https://doi.org/10.14778/3055330.3055331.

F. McSherry and K. Talwar. Mechanism design via differential privacy. In *48th Annual Symposium on Foundations of Computer Science*, 2007.

D. Mir, S. Muthukrishnan, A. Nikolov, and R. N. Wright. Pan-private algorithms via statistics on sketches. In *Proceedings of the Thirtieth ACM SIGMOD-SIGACT-SIGART Symposium on Principles of Database Systems*, PODS '11, page 37–48, New York, NY, USA, 2011. Association for Computing Machinery. ISBN 9781450306607. doi: 10.1145/1989284.1989290. URL https://doi.org/10.1145/1989284.1989290.

W. Qardaji, W. Yang, and N. Li. Understanding hierarchical methods for differentially private histograms. *Proc. VLDB Endow.*, 6(14): 1954–1965, Sept. 2013. ISSN 2150-8097. doi: 10.14778/2556549.2556576. URL https://doi.org/10.14778/2556549.2556576.

R. Rogers, S. Subramaniam, S. Peng, D. Durfee, S. Lee, S. K. Kancha, S. Sahay, and P. Ahammad. Linkedin's audience engagements api: A privacy preserving data analytics system at scale, 2020.

R. M. Rogers and A. Roth. Asymptotically truthful equilibrium selection in large congestion games. In *Proceedings of the Fifteenth ACM Conference on Economics and Computation*, EC '14, page 771–782, New York, NY, USA, 2014. Association for Computing Machinery. ISBN 9781450325653. doi: 10.1145/2600057.2602856. URL https://doi.org/10.1145/2600057.2602856.

S. Vadhan. The complexity of differential privacy. In *Tutorials on the Foundations of Cryptography*, 2017.

G. K. Zipf. *The Psychobiology of Language, an Introduction to Dynamic Philology.* Houghton Mifflin, Boston, 1935.

Supplementary Material: Differentially Private Histograms under Continual Observation: Streaming Selection into the Unknown

A Algorithms from Table 1

Algorithm 4 `KnownGauss`; Gaussian mechanism over known domain $\mathcal{U}$ with ℓ_0 sensitivity Δ_0

Input: Histogram $\mathbf{h} = \{(h_u, u) : u \in \mathcal{U}\}$, along with noise scale τ.
Output: Noisy result.
Return $\{(\mathtt{N}\left(h_u, \tau^2\right), u) : u \in \mathcal{U}\}$

Theorem 7 (Bun and Steinke (2016)). *Assume that* $||\mathbf{h} - \mathbf{h}'||_\infty \leq 1$ *and* $||\mathbf{h} - \mathbf{h}'||_0 \leq \Delta_0$ *for any neighbors* $\mathbf{h}, \mathbf{h}'$. *Then, the algorithm* $\mathtt{KnownGauss}(\cdot; \tau)$ *is* $\frac{\Delta_0}{2\tau^2}$*-zCDP and hence* $\left(\frac{\Delta_0}{2\tau^2} + \frac{1}{\tau}\sqrt{2\Delta_0 \ln(1/\delta')}, \delta'\right)$*-DP for any* $\delta' > 0$.

Algorithm 5 $\mathtt{KnownGumb}^k$; Exponential Mechanism over known domain $\mathcal{U}$

Input: Histogram $\mathbf{h} = \{(h_u, u) : u \in \mathcal{U}\}$, number of outcomes k, and noise scale τ.
Output: Ordered set of k indices and counts.
Set $S = \emptyset$
for $u \in \mathcal{U}$ **do**
  Set $v_u = h_u + \mathtt{Gumbel}(\tau/2)$
  Update $S \leftarrow S \cup \{(v_u, u)\}$
Sort S where $v_{u_1} \geq \cdots \geq v_{u_{|\mathcal{U}|}}$
Return $\left\{(\mathtt{N}\left(h_{u_1}, \tau^2\right), u_1), ..., (\mathtt{N}\left(h_{u_k}, \tau^2\right), u_k)\right\}$

Theorem 8 (Cesar and Rogers (2020)). *Assume that* $||\mathbf{h} - \mathbf{h}'||_\infty \leq 1$ *and* $||\mathbf{h} - \mathbf{h}'||_0$ *is unrestricted for any neighbors* $\mathbf{h}, \mathbf{h}'$. *Then,* $\mathtt{KnownGumb}^k(\cdot; \tau)$ *is* $\frac{k}{\tau^2}$*-zCDP and hence* $\left(\frac{k}{\tau^2} + \frac{2}{\tau}\sqrt{k \ln(1/\delta')}, \delta'\right)$*-DP for any* $\delta' > 0$.

Algorithm 6 $\mathtt{UnkGumb}^{k,\bar{k}}$; Unknown domain mechanism with access to $\bar{k} + 1 \geq k$ elements

Input: Histogram $\mathbf{h}$; outcomes k, cut off at $\bar{k} + 1$, and noise scale τ.
Output: Ordered set of indices and counts.
Sort $h_{(1)} \geq h_{(2)} \geq \cdots \geq h_{(\bar{k}+1)}$, with respective labels $i_{(1)}, \cdots, i_{(\bar{k}+1)}$.
Set $h_\perp = h_{(\bar{k}+1)} + 1 + \tau \ln(1/\delta)$, with label $\perp$.
Set $v_\perp = h_\perp + \mathtt{Gumbel}(\tau/2)$, with label $\perp$.
Set $S = \emptyset$
for $j \leq \bar{k}$ **do**
  if $h_{(j)} > h_{(\bar{k}+1)}$ **then**
    Add $S \leftarrow S \cup \{(v_{(j)} = h_{(j)} + \mathtt{Gumbel}(\tau/2), i_{(j)})\}$.
Sort S by its counts.
Let $v_1, ..., v_j, v_\perp$ be the descending list of counts up until $v_\perp$, with respective labels $u_1, \cdots, u_j$.
if $j < k$ **then**
  Return $\{(\mathtt{N}\left(h_{u_1}, \tau^2\right), u_1), ..., (\mathtt{N}\left(h_{u_j}, \tau^2\right), u_j), \perp\}$
else
  Return $\{(\mathtt{N}\left(h_{u_1}, \tau^2\right), u_1), ..., (\mathtt{N}\left(h_{u_k}, \tau^2\right), u_k)\}$

Theorem 9 (Durfee and Rogers (2019)). *Assume* $||\mathbf{h} - \mathbf{h}'||_\infty \leq 1$ *for any neighbors* $\mathbf{h}, \mathbf{h}'$. *For any* $\delta > 0$, $\mathtt{UnkGumb}^{k,\bar{k}}(\cdot; \tau, \delta)$ *is* $\left(\frac{k}{\tau^2} + \frac{2}{\tau}\sqrt{k \ln(1/\delta')}, \bar{k}\delta + \delta'\right)$*-DP.*

B Concentrated Differential Privacy

The analysis of our algorithms will typically use a variant of DP called *zero-mean Concentrated DP* (zCDP) from Bun and Steinke (2016), which provides tighter composition bounds than traditional DP analysis.

This variant of DP is based on the Rényi divergence of order $\alpha > 1$ between two distributions P and Q over the same domain, denoted as $D_\alpha(P||Q)$ where

$$D_\alpha(P||Q) := \frac{1}{\alpha - 1} \log \mathbb{E}_{z\sim P}\left[\left(\frac{P(z)}{Q(z)}\right)^{\alpha-1}\right].$$

Definition B.1 (Zero-mean Concentrated Differential Privacy)**.** *A randomized algorithm $M : \mathcal{X} \to \mathcal{Y}$ is δ-approximately ρ-zCDP if for any neighbors $x, x' \in \mathcal{X}$, there exists events E and E', such that $\Pr[E], \Pr[E'] \geq 1-\delta$ and for every $\alpha > 1$ we have the following bound in terms of the Rényi divergence $D_\alpha(\cdot||\cdot)$ of order α*

$$D_\alpha(M(x)|_E||M(x')|_{E'}) \leq \alpha\rho, \text{ and}$$
$$D_\alpha(M(x')|_{E'}||M(x)|_E) \leq \alpha\rho.$$

where $M(x)|_E$ is the distribution of $M(x)$ conditioned on event E and similarly for $M(x')|_{E'}$. If $\delta = 0$, then we say M is ρ-zCDP.

A useful property of zCDP is that composing multiple zCDP mechanisms results in another zCDP mechanism where the privacy parameters add up.

Lemma B.1 (Bun and Steinke (2016))**.** *Let $M_1 : \mathcal{X} \to \mathcal{Y}$ be δ_1-approximate ρ_1-zCDP and $M_2 : \mathcal{X} \times \mathcal{Y} \to \mathcal{Y}'$ be δ_2-approximate ρ_2-zCDP in its first argument, i.e. $M_2(\cdot, y)$ is δ_2-approximate ρ_2-zCDP for all $y \in \mathcal{Y}$. Then the mechanism $M : \mathcal{X} \to \mathcal{Y}'$ where $M(\cdot) = M_2(\cdot, M_1(\cdot))$ is $(\delta_1 + \delta_2)$-approximate $(\rho_1 + \rho_2)$-zCDP.*

C Missing Pseudocode for `MetaAlgo` in Section 4

Algorithm 7 `MetaAlgo`; Return a running histogram in various settings from Table 1

Input: Stream $\omega_{1:T} = \omega_1, \cdots, \omega_T$, where $\omega_t \subseteq \mathcal{U}$, and use L_r from (2).
Output: Noisy histograms $\hat{h}_{1:T} = \hat{h}_1, \cdots, \hat{h}_T$ with labels for each count at each round.
if Domain $\mathcal{U}$ is known **then**
  if ℓ_0-Sensitivity Δ_0 **then**
    for Each cell j, ℓ of the partial histogram table **do**
      $\{(\hat{p}^u_{j,\ell}, u)\} = \texttt{KnownGauss}(\{(p^u_{j,\ell}, u) : u \in \mathcal{U}\}; \sqrt{L_r}\tau)$
  else
    for Each cell j, ℓ of the partial histogram table **do**
      $\{(\hat{p}^u_{j,\ell}, u)\} = \texttt{KnownGumb}^k(\{(p^u_{j,\ell}, u) : u \in \mathcal{U}\}; \sqrt{L_r}\tau)$
else
  if ℓ_0-Sensitivity Δy_0 **then**
    for Each cell j, ℓ of the partial histogram table **do**
      Let $\mathcal{U}_{j,\ell}$ be the set of items in the portion of ω used in $p_{j,\ell}$
      $\{(\hat{p}^u_{j,\ell}, u)\} = \texttt{UnkGauss}^{\bar{k}}(\{(p^u_{j,\ell}, u) : u \in \mathcal{U}_{j,\ell}\}; \sqrt{L_r}\tau, \delta/L_r)$
  else
    for Each cell j, ℓ of the partial histogram table **do**
      Let $\mathcal{U}_{j,\ell}$ be the set of items in the portion of ω used in $p_{j,\ell}$
      $\{(\hat{p}^u_{j,\ell}, u)\} = \texttt{UnkGumb}^{k,\bar{k}}(\{(p^u_{j,\ell}, u) : u \in \mathcal{U}_{j,\ell}\}; \sqrt{L_r}\tau, \delta/L_r)$
Return counts and labels at each round $t \in [T]$ using the corresponding noisy partial histograms $\{(\hat{p}^u_{j,\ell}, u)\}$.

D Omitted Portions of Section 5

We present the pseudocode for $\texttt{SparseGumb}^{s,k}$ in Algorithm 8 and the missing analysus from the main version.

Algorithm 8 $\texttt{SparseGumb}^{s,k}$; Continually return top-k

Input: $\omega_{1:T}$ with s, top-k returned, $\{\eta_t\}_{t=1}^T$, and τ.
Output: Noisy histograms $\hat{\mathbf{h}}_{1:T}$ for top-k items.
Let h_t^u be the count for item u from $\mathbf{h}(\omega_{1:t};\mathcal{U})$.
Let $\tau_1 = \sqrt{s}\tau$ and $\tau_2 = \sqrt{s+1}\tau$.
$\{i_1, \cdots, i_k\} = \texttt{KnownGumb}^k(\mathbf{h}(\omega_{1:1};\mathcal{U}); \tau_2)$. ▷ Select top-$k$
Let $\sigma_{1:T}^u$ be the binary stream for item $u \in \{i_1, \cdots, i_k\}$.
for $i \in \{i_1, \cdots, i_k\}$ **do**
  Get the current counts: $\hat{h}_{1:T}^i = \texttt{BaseMech}(\sigma_{1:T}^i; \tau_2)$ ▷ Return running counts of top-k
Set $\mathcal{D}_1 = \{i_1, \cdots, i_k\}$ and sample $Z \sim \texttt{Lap}(2\tau_1)$.
for $t \in \{2. \cdots, T\}$ **do**
  $\texttt{CONT} = True.$
  if $s = 0$ **then**
    $\texttt{CONT} = False$
  while $\texttt{CONT}$ **do**
    Set $i^* = \text{argmin}_{i \in \{i_1, \cdots, i_k\}} \{\hat{h}_t^i\}$
    Set threshold $\hat{m}_t = \hat{h}_t^{i^*} + \eta_t + Z$ ▷ Set threshold as in Sparse Vector
    for $u \in \mathcal{U} \setminus \{i_1, \cdots, i_k\}$ **do**
      if $h_t^u + \texttt{Lap}(4\tau_1) > \hat{m}_t$ **then** ▷ Check if there is a new element in the top-k.
        $\{i_1, \cdots, i_k\} = \texttt{KnownGumb}^k(\mathbf{h}(\omega_{1:t}); \tau_2)$. ▷ Select top-$k$
        for $i \in \{i_1, \cdots, i_k\}$ **do**
          Set $\hat{h}_{1:T}^i = \texttt{BaseMech}(\sigma_{1:T}^i; \tau_2)$ ▷ Return running counts of the new top-k
        $s \leftarrow s - 1$. ▷ Reduce by one the number of switches
        $\texttt{CONT} = False$
        Redraw $Z \sim \texttt{Lap}(2\tau_1)$
        break
  $\mathcal{D}_t = \{i_1, \cdots, i_k\}$
Return $\left(\left\{(\hat{h}_t^u, u) : u \in \mathcal{D}_t\right\} : t \in [T]\right)$

Proof of Theorem 4. We rely on the privacy analysis of multiple subroutines. We know that each call, of the $(s+1)$ calls, to the routine $\texttt{KnownGumb}^k(\cdot, \sqrt{s+1}\tau)$, without releasing counts, is $\frac{k}{2(s+1)\tau^2}$-zCDP. Further, there can be at most $(s+1)\cdot k$ many different instances of $\texttt{BaseMech}$, each of which is $\frac{1}{2(s+1)\tau^2}$-zCDP. Lastly, we use the Sparse Vector technique to determine which rounds we should find a new top-k in. Note that we use different thresholds at each round t, but we do not update the noise on the threshold unless we update the top-k. We then use the general version of Sparse Vector from Lyu et al. (2017) to conclude that each time we select a round to run $\texttt{KnownGumb}^k$, it is $\frac{2}{\sqrt{s}\tau}$-DP and hence $\frac{2}{s\tau^2}$-zCDP. We then apply composition of zCDP mechanisms to get the result. □

We will conduct experiments to see how the number of switches s and $\{\eta_t\}_{t=1}^T$ impact the accuracy of the current round's selected item and the true maximum count. We will also need to set the additional threshold amounts η_t for each round $t \in [T]$. At each switch, we will use the Exponential Mechanism to return the privatized max item. If the item that is currently selected before a switch has a count that is significantly smaller than the maximum item, then we will want the Exponential Mechanism to select a different item. As a rule of thumb, we then consider setting η_t based on the utility of the Exponential Mechanism, which selects an item whose count is close to the maximum count with high probability.

Lemma D.1 (McSherry and Talwar (2007))**.** *Let I_τ be the index selected from the Exponential Mechanism $\texttt{KnownGumb}^1(\{h^u, u : u \in \mathcal{U}\}; \tau)$, without its count, and let i^* be the argmax of $\{h^u : u \in \mathcal{U}\}$. Then for $\beta > 0$, we have*

$$\Pr[h^{I_\tau} > h^{i*} - O\left(\tau \ln\left(|\mathcal{U}|/\beta\right)\right)] \geq 1 - \beta$$

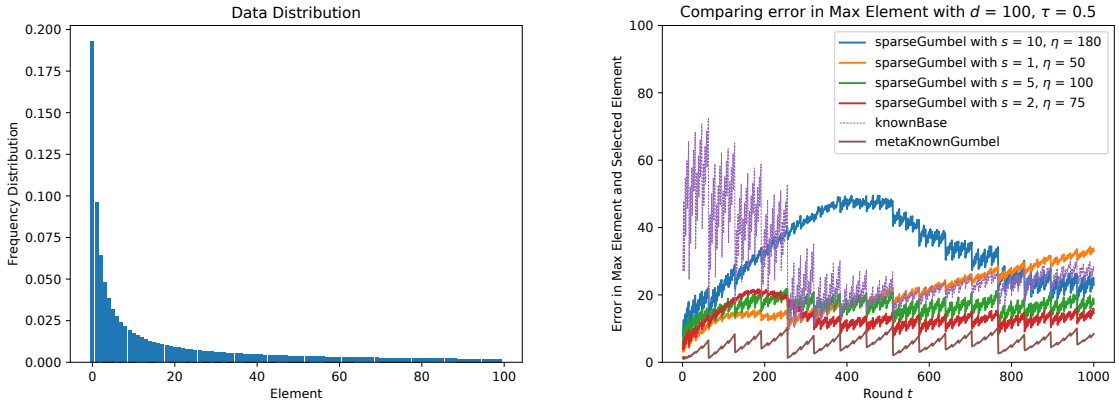

Figure 2: We compare $\texttt{SparseGumb}^{s,1}$ with `KnownBase` and `MetaAlgo` with unrestricted ℓ_0-sensitivity, denoted as 'metaKnownGumbel' in the plot.

We factor in the error from the Laplace noise that is added to each true count of the non-selected items, which at a given round will all be within $O(\sqrt{s}\tau \ln(d))$ of the true count with high probability, as well as the error from the current selected count, which is used in the threshold and will be within $O(\sqrt{r-1}L_r\tau)$ of its true count with high probability. Putting these errors together with the error incurred from the Exponential Mechanism given in Lemma D.1, we then consider setting $\eta_t \equiv \eta$ where

$$\eta = \tau\sqrt{s} \cdot O\left(\sqrt{r-1}L_r + \ln(d)\right).$$

To generate a stream of data, we sample an item from a distribution following Zipf's Law (Zipf, 1935), as it models many data sources that occur in nature well, given in the left plot of Figure 2, with $d = 100$ items. The right plot in Figure 2 shows the error between the count of the true max item with the noisy count of the selected top-1 item $\texttt{SparseGumb}^{s,1}$ at each round $t \in [1000]$ with various s and $\eta \equiv \eta_t$. We compare this algorithm with both the `KnownBase` algorithm, where we use $\Delta_0 = d$, since we are assuming unrestricted ℓ_0-sensitivity, and we also compare the results with `MetaAlgo` in the same setting. In our experiments, we will equalize the privacy level in all algorithms. Hence, we will use $\tau \leftarrow \tau\sqrt{d}$ in `KnownBase`, $\tau \leftarrow \tau\sqrt{2}$ in `MetaAlgo`, and $\tau \leftarrow \tau\sqrt{6}$ in $\texttt{SparseGumb}^{s,1}$, so that each will be $\frac{1}{2\tau^2}$-zCDP. As expected `MetaAlgo` outperforms the other algorithms, but recall that at each round t it needs the full stream $\omega_{1:t}$, which may be impractical in some situations (see Section 1). Instead, `SparseGumb` only requires the current aggregate histogram. It is interesting to notice the behavior of `SparseGumb` with respect to s, with very few switches ($s = 1$) the algorithm runs out of switches before the maximum element is learned and thus the error seems to increase linearly. If we allow `SparseGumb` more switches, it does not run out of switches very quickly, unfortunately the magnitude of the noise scales with $\sqrt{s}$ thus hurting accuracy. The right number of switches is a parameter that the practitioner needs to tune to balance the amount of noise incurred and the number of times the distribution is expected to change. The plots show the average error at each $t \in [1000]$ over 1000 independent trials.

E Omitted Portions of Section 6

We present the pseudocode for $\texttt{UnkGauss}^{\bar{k}}$ for the one shot restricted ℓ_0-sensitivity and unknown domain setting and missing analysis from the main version.

Algorithm 9 $\texttt{UnkGauss}^{\bar{k}}$; Δ_0-Restricted Sensitivity Gaussian Mechanism with top-$(\bar{k}+1)$

Input: Histogram $\mathbf{h} = \{(h_u, u) : u \in \mathcal{U}\}$, cut off at $\bar{k}$, along with parameters τ, δ.
Output: Noisy histogram with labels $\{i_j\}$ and noisy counts $\{v_{i_j}\}$.
Let $h_{i_{(1)}} \geq h_{i_{(2)}} \geq \cdots \geq h_{i_{(\bar{k})}} \geq h_{i_{(\bar{k}+1)}} \geq \cdots \geq h_{i_{(d)}}$, with corresponding labels $i_{(j)} \in \mathcal{U}$ for $j \in [d]$
Set $v_\perp = h_{i_{(\bar{k}+1)}} + 1 + \sqrt{2}\tau\Phi^{-1}(1-\delta) + \mathtt{N}\left(0, \tau^2\right)$ ▷ Set (data-dependent) noisy threshold
Set discovered set $\mathcal{D} = \emptyset$
for $u \in \mathcal{U}$ such that $u \in \{i_{(j)} : j \in [\bar{k}]\}$ and $h_u > 0$ **do** ▷ Add noise to each element in top-$\bar{k}$
  Set $v_u = \mathtt{N}\left(h_u, \tau^2\right)$ with label u.
  $\mathcal{D} \leftarrow \mathcal{D} \cup \{u\}$
Sort $\{v_u : u \in \mathcal{D}\} \cup v_\perp$
Let $v_{i_{(1)}}, ..., v_{i_{(\ell)}}$ be the counts in descending order until $v_\perp$, with relative labels $i_{(1)}, i_{(2)}, \cdots, i_{(\ell)}$.
Return $\{(v_{i_{(1)}}, i_{(1)}), ..., (v_{i_{(\ell)}}, i_{(\ell)})\}$ ▷ Return elements above the noisy threshold

Proof of Lemma 6.1. Fix an outcome set S, we then have

$$\begin{aligned}
\Pr[M(\mathbf{h}^{(b)}) \in S] &= \Pr[M(\mathbf{h}^{(b)}) \in S \cap \mathcal{G}_M] + \Pr[M(\mathbf{h}^{(b)}) \in S \cap \mathcal{B}_M^b] \\
&\leq \Pr[M(\mathbf{h}^{(b)}) \in S \cap \mathcal{G}_M] + \delta \\
&= \Pr[A(b; \mathbf{h}^{(0)}, \mathbf{h}^{(1)}) \in S \cap \mathcal{G}_M] + \delta \\
&\leq e^\epsilon \Pr[A(1-b; \mathbf{h}^{(0)}, \mathbf{h}^{(1)}) \in S \cap \mathcal{G}_M] + \delta' + \delta \\
&= e^\epsilon \Pr[M(\mathbf{h}^{(1-b)}) \in S \cap \mathcal{G}_M] + \delta' + \delta \\
&= e^\epsilon \Pr[M(\mathbf{h}^{(1-b)}) \in S] + \delta' + \delta
\end{aligned}$$

□

Proof of Lemma 6.2. We need to show that adding independent noise to $\bar{k}$ counts, some of which have $\{\top_i\}$ labels and then dropping these terms is equivalent to having never considered those elements.

Let $f(\cdot)$ be the density function for distribution $\mathcal{P}$, $f_{M'}(\cdot)$ be the density of M', and $f_{\hat{M}'}$ be the density of $\hat{M}'$. We fix an outcome of counts $(z_1, z_2, \cdots, z_k)$ and denote the set of indices that are not in this outcome to be I after dropping counts of $\{\top_i\}$. The density for mechanism $\hat{M}'$ is then

$$\begin{aligned}
f_{\hat{M}'}(z_1, \cdots, z_k) &= \prod_{i=1}^{k} f(z_i - h_i) \int_{-\infty}^{\min\{z_i : i \in [k]\}} f\left(z_\perp - h_\perp\right) \prod_{\ell \in I} \left(\int_{-\infty}^{z_\perp} f(z_\ell - h_\ell) dz_\ell \right) dz_\perp \\
&= \prod_{i=1}^{k} f(z_i - h_i) \int_{-\infty}^{\min\{z_i : i \in [k]\}} f\left(z_\perp - h_\perp\right) \prod_{\ell \in I} \left(\int_{-\infty}^{z_\perp} f(z_\ell - h_\ell) dz_\ell \right) dz_\perp \\
&\qquad \cdot \prod_{j=1}^{\bar{k}-p-1} \int_{\mathbb{R}} f(z_j) dz_j \\
&= f_{M'}(z_1, \cdots, z_k).
\end{aligned}$$

□

Proof of Lemma 6.3. Without loss of generality, we assume that $\mathbf{h}^{(0)}$ has larger counts than $\mathbf{h}^{(1)}$. If $i \in \mathcal{D}_\top^{\bar{k}}(\mathbf{h}^{(0)}) \setminus \mathcal{D}_\top^{\bar{k}}(\mathbf{h}^{(1)})$ then we know that $h_i^{(0)} \geq h_{(\bar{k})}^{(0)}$ but $h_i^{(1)} \leq h_{(\bar{k})}^{(1)}$. We also know that $h_i^{(0)} \leq h_i^{(1)} + 1$. Putting this together, we have

$$h_{(\bar{k})}^{(0)} \leq h_i^{(0)} \leq h_i^{(1)} + 1 \leq h_{(\bar{k})}^{(1)} + 1.$$

Similarly, for $j \in \mathcal{D}_\top^{\bar{k}}(\mathbf{h}^{(1)}) \setminus \mathcal{D}_\top^{\bar{k}}(\mathbf{h}^{(0)})$ we have $h_j^{(0)} \leq h_{(\bar{k})}^{(0)}$ and $h_j^{(1)} \geq h_{(\bar{k})}^{(1)}$. Further, $h_j^{(1)} \leq h_j^{(0)}$, which gives us

$$h_{(\bar{k})}^{(1)} \leq h_j^{(1)} \leq h_j^{(0)} \leq h_{(\bar{k})}^{(0)}.$$

Algorithm 10 $\texttt{GaussMech}^{\bar{k}}_{\perp}$; Gaussian Mechanism over Limited Domain

Input: Bit $b \in \{0,1\}$, neighboring histograms $\mathbf{h}^{(0)}$ and $\mathbf{h}^{(1)}$, cut off $\bar{k}$, and parameter τ.
Output: Histogram $\mathbf{v}$ with labels in $\mathcal{D}^{\bar{k}}_{\top}(\mathbf{h}^{(0)}) \cap \mathcal{D}^{\bar{k}}_{\top}(\mathbf{h}^{(1)})$ and $\{B_\ell : \ell \in [|\mathcal{D}^{\bar{k}}_{\top}(\mathbf{h}^{(b)}) \setminus \mathcal{D}^{\bar{k}}_{\top}(\mathbf{h}^{(1-b)})|]\}$
We relabel the indices in both $\mathbf{h}^{(0)}$ and $\mathbf{h}^{(1)}$ to form the following histogram $\mathbf{v}^{(b)}$
for $i \in \mathcal{D}^{\bar{k}}_{\top}(\mathbf{h}^{(0)}) \cap \mathcal{D}^{\bar{k}}_{\top}(\mathbf{h}^{(1)})$ **do**
  $\mathbf{v}^{(b)} \leftarrow \mathbf{v}^{(b)} \cup \{(h^{(b)}_i, i)\}$, where $h^{(b)}_{\top_j} = 0$. ▷ Keep common labels
Initialize $\ell = 1$
for $j \in \mathcal{D}^{\bar{k}}_{\top}(\mathbf{h}^{(b)}) \setminus \mathcal{D}^{\bar{k}}_{\top}(\mathbf{h}^{(1-b)})$ **do** ▷ Create "bad" labels for uncommon elements
  $\mathbf{v}^{(b)} \leftarrow \mathbf{v}^{(b)} \cup \{(h^{(b)}_j, B_\ell)\}$.
  $\ell = \ell + 1$
$\mathbf{v}^{(b)} \leftarrow \mathbf{v}^{(b)} \cup \{(h^{(b)}_{(\bar{k}+1)} + 1 + \sqrt{2}\tau\Phi^{-1}(1-\delta), \perp)\}$
Add $\texttt{N}\left(0, \tau^2\right)$ to each count in $\mathbf{v}^{(b)}$ to form the noisy histogram $\hat{\mathbf{v}}$ ▷ Apply Gaussian Mechanism
Return $\hat{\mathbf{v}}$

Combining the two, we have

$$h^{(1)}_{(\bar{k})} \le h^{(1)}_j \le h^{(0)}_i \le h^{(1)}_{(\bar{k})} + 1 \implies h^{(0)}_i - h^{(1)}_j \le 1.$$

Lastly, we have $h^{(1)}_{(\bar{k}+1)} \le h^{(0)}_{(\bar{k}+1)}$. Now assume that $h^{(0)}_{(\bar{k}+1)} > h^{(1)}_{(\bar{k}+1)} + 1$. This can only occur if the $(\bar{k}+1)$-th ranked element in $\mathbf{h}^{(0)}$ is not the same as the $(\bar{k}+1)$-th ranked element in $\mathbf{h}^{(1)}$, otherwise their count would differ by at most 1. Hence, there must be some element i with count $h^{(1)}_i \le h^{(1)}_{(\bar{k}+1)}$, but $h^{(0)}_i \ge h^{(0)}_{(\bar{k}+1)}$, since that would change the label for the $(\bar{k}+1)$-th ranked element between $\mathbf{h}^{(1)}$ and $\mathbf{h}^{(0)}$. However, $h^{(0)}_i \le h^{(1)}_i + 1$ and thus

$$h^{(0)}_{(\bar{k}+1)} \le h^{(0)}_i \le h^{(1)}_i + 1 \le h^{(1)}_{(\bar{k}+1)} + 1.$$

□

Proof of Lemma 6.4. Let ℓ be the number of bins that differ between $\mathbf{h}^{(0)}$ and $\mathbf{h}^{(1)}$ on labels in $\mathcal{D}^{\bar{k}}_{\top}(\mathbf{h}^{(0)}) \cap \mathcal{D}^{\bar{k}}_{\top}(\mathbf{h}^{(1)})$. Without loss of generality, we assume that $\mathbf{h}^{(0)}$ has larger counts than $\mathbf{h}^{(1)}$. We know by definition that $\ell \le \Delta_0$. We now show that

$$|\mathcal{D}^{\bar{k}}_{\top}(\mathbf{h}^{(b)}) \setminus \mathcal{D}^{\bar{k}}_{\top}(\mathbf{h}^{(1-b)})| \le \Delta_0 - \ell.$$

It suffices to only consider $|\mathcal{D}^{\bar{k}}_{\top}(\mathbf{h}^{(0)}) \setminus \mathcal{D}^{\bar{k}}_{\top}(\mathbf{h}^{(1)})|$ since $\mathcal{D}^{\bar{k}}_{\top}(\mathbf{h}^{(0)}) \setminus \mathcal{D}^{\bar{k}}_{\top}(\mathbf{h}^{(1)})$ has the same cardinality. Note that for any $i \in \mathcal{D}^{\bar{k}}_{\top}(\mathbf{h}^{(0)}) \setminus \mathcal{D}^{\bar{k}}_{\top}(\mathbf{h}^{(1)})$, that implies $h^{(0)}_i > h^{(1)}_i$, and we know only $\Delta_0 - \ell$ such additional indices can exist. If $h^{(0)}_i = h^{(1)}_i$, then the position of index i cannot have moved up the ordering from $\mathbf{h}^{(1)}$ to $\mathbf{h}^{(0)}$ because we assumed $\mathbf{h}^{(0)}$ had larger counts. Therefore, if $i \notin \mathcal{D}^{\bar{k}}_{\top}(\mathbf{h}^{(1)})$ and $h^{(0)}_i = h^{(1)}_i$ we must also have $i \notin \mathcal{D}^{\bar{k}}_{\top}(\mathbf{h}^{(0)})$. Hence, $|\mathcal{D}^{\bar{k}}_{\top}(\mathbf{h}^{(0)}) \setminus \mathcal{D}^{\bar{k}}_{\top}(\mathbf{h}^{(1)})| \le \Delta_0 - \ell$ □

In order to prove Lemma 6.6, we will need to define a mechanism that takes an input domain of indices, as well as a histogram.

Definition E.1. *[Sorted Gaussian Mechanism over Limited Domain] We define the* sorted Gaussian mechanism over limited domain *to be* $\texttt{GaussMax}^{\bar{k}}$ *that takes as input a histogram along with a domain set of indices* $\mathcal{D}$ *and returns an ordered list of elements until* $\perp$*'s count, that is*

$$\texttt{GaussMax}^{\bar{k}}(\mathbf{h}, \mathcal{D}) = \{(v_{i_{(1)}}, i_{(1)}), ..., (v_{i_{(1)}}, i_{(j)}), (v_\perp, \perp)\}$$

where $(v_{i_{(1)}}, ..., v_{i_{(j)}}, v_\perp)$ *is the sorted list until* $v_\perp$ *of* $v_i = \texttt{N}\left(h_{(i)}, \tau^2\right)$ *and* $v_\perp = \texttt{N}\left(h_\perp, \tau^2\right)$*, for each* $i \in \mathcal{D}$ *and*

$$h_\perp := h_{(\bar{k}+1)} + 1 + \sqrt{2}\tau\Phi^{-1}(1-\delta) \tag{5}$$

Note that $\texttt{GaussMax}^{\bar{k}}(\mathbf{h}, \mathcal{D}^{\bar{k}}(\mathbf{h}))$ and $\texttt{UnkGauss}^{\bar{k}}(\mathbf{h})$ are equal in distribution. We will use the following result to prove Lemma 6.6.

Lemma E.1. *Given an histogram* $\mathbf{h}$ *and some domain* $\mathcal{D}$ *that can include dummy* $\{\top_i\}$*. For any* $i \in \mathcal{D}$ *such that* $h_i \leq h_{(\bar{k}+1)} + 1$*, then*

$$\Pr[i \in \texttt{GaussMax}^{\bar{k}}(\mathbf{h}, \mathcal{D})] \leq \delta.$$

Proof. For simplicity, we will set $T = \tau\sqrt{2}\Phi^{-1}(1-\delta)$, which implies $h_\perp = h_{(\bar{k}+1)} + 1 + T$ and plug back in at the end of the analysis. By construction of our mechanism, we know that the noisy estimate of h_i must be greater than the noisy estimate of our threshold $h_\perp = h_{(\bar{k}+1)} + 1 + T$ to be a possible output, which implies

$$\Pr[i \in \texttt{GaussMax}^{\bar{k}}(\mathbf{h}, \mathcal{D})] \leq \Pr[h_i + \texttt{N}\left(0, \tau^2\right) > h_\perp + \texttt{N}\left(0, \tau^2\right)].$$

By assumption, $h_i \leq h_{(\bar{k}+1)} + 1$, and by the fact that the sum of two independent Gaussians $Z_1, Z_2 \sim \texttt{N}\left(0, \tau^2\right)$ is also Gaussian, i.e. $Z_1 + Z_2 \sim \texttt{N}\left(0, 2\tau^2\right)$,

$$\Pr[i \in \texttt{GaussMax}^{\bar{k}}(\mathbf{h}, \mathcal{D})] \leq \Pr[\texttt{N}\left(0, 2\tau^2\right) > T] = 1 - \Phi\left(\frac{T}{\sqrt{2}\tau}\right).$$

Plugging in $T = \sqrt{2}\sigma\Phi^{-1}(1-\delta)$ gives the result.

□

We can now prove Lemma 6.6.

Proof of Lemma 6.6. This will follow from a simple union bound on each $i \in \mathcal{D}^{\bar{k}}_\top(\mathbf{h}^{(0)}) \setminus \mathcal{D}^{\bar{k}}_\top(\mathbf{h}^{(1)})$ where we consider each subset of $\mathcal{B}^{(0)}_{\texttt{Gauss}}$ such that each outcome contains i, or more formally we define $\mathcal{B}^{(0)}_{\texttt{Gauss}}(i) := \{o \in \mathcal{B}^{(0)}_{\texttt{Gauss}} : i \in o\}$ This then implies that

$$\Pr[\texttt{GaussMax}^{\bar{k}}(\mathbf{h}^{(0)}, \mathcal{D}^{\bar{k}}_\top(\mathbf{h})) \in \mathcal{B}^{(0)}_{\texttt{Gauss}}] \leq \sum_{i \in \mathcal{D}^{\bar{k}}(\mathbf{h}^{(0)}) \setminus \mathcal{D}^{\bar{k}}(\mathbf{h}^{(1)})} \Pr[\texttt{GaussMax}^{\bar{k}}(\mathbf{h}, \mathcal{D}^{\bar{k}}_\top(\mathbf{h})) \in \mathcal{B}^{(0)}_{\texttt{Gauss}}(i)]$$

because each outcome $o \in \mathcal{B}^{(0)}_{\texttt{Gauss}}$ must contain some $i \in \mathcal{D}^{\bar{k}}_\top(\mathbf{h}^{(0)}) \setminus \mathcal{D}^{\bar{k}}_\top(\mathbf{h}^{(1)})$ by construction. Furthermore, by construction we also have

$$\Pr[\texttt{GaussMax}^{\bar{k}}(\mathbf{h}^{(0)}, \mathcal{D}^{\bar{k}}(\mathbf{h}^{(0)})) \in \mathcal{B}^{(0)}_{\texttt{Gauss}}(i)] = \Pr[i \in \texttt{GaussMax}^{\bar{k}}(\mathbf{h}^{(0)}, \mathcal{D}^{\bar{k}}(\mathbf{h}^{(0)}))]$$

Our claim then immediately follows from Lemma E.1 and the fact that the size of $\mathcal{D}^{\bar{k}}(\mathbf{h}^{(0)}) \setminus \mathcal{D}^{\bar{k}}(\mathbf{h}^{(1)})$ is at most Δ_0 by Lemma 6.4. □

We can now prove Theorem 5.

Proof of Theorem 5. We will use Lemma 6.1 to prove this result. From Lemma 6.6, we have the probability of bad outcomes being negligible. We now need to define a mechanism $A(b; \mathbf{h}^{(0)}, \mathbf{h}^{(1)})$ that matches $\texttt{UnkGauss}^{\bar{k}}$ on good outcomes and is DP. Lemma 6.5 shows that $\texttt{GaussMech}^{\bar{k}}_\perp(\cdot; \mathbf{h}^{(0)}, \mathbf{h}^{(1)})$ is DP. We then define a post processing function on $\texttt{GaussMech}^{\bar{k}}_\perp$. First, we sort in descending order the elements up until we hit $\perp$ and then we eliminate the rest. Next, we drop all the dummy labels $\{\top_i\}$ and their noisy counts. We know from Lemma 6.2 that sorting up to $\perp$ and dropping the dummy labels is equivalent to never considering the dummy elements in the first place. Note that this post processing function on $\texttt{GaussMech}^{\bar{k}}_\perp(b; \mathbf{h}^{(0)}, \mathbf{h}^{(1)})$ is equivalent to our main algorithm $\texttt{UnkGauss}^{\bar{k}}$ for good outcomes. Because post-processing cannot increase the privacy loss parameters, we can use Lemma 6.1 with $A(b; \mathbf{h}^{(0)}, \mathbf{h}^{(1)})$ as this post-processing function of $\texttt{GaussMech}^{\bar{k}}_\perp(b; \mathbf{h}^{(0)}, \mathbf{h}^{(1)})$. □

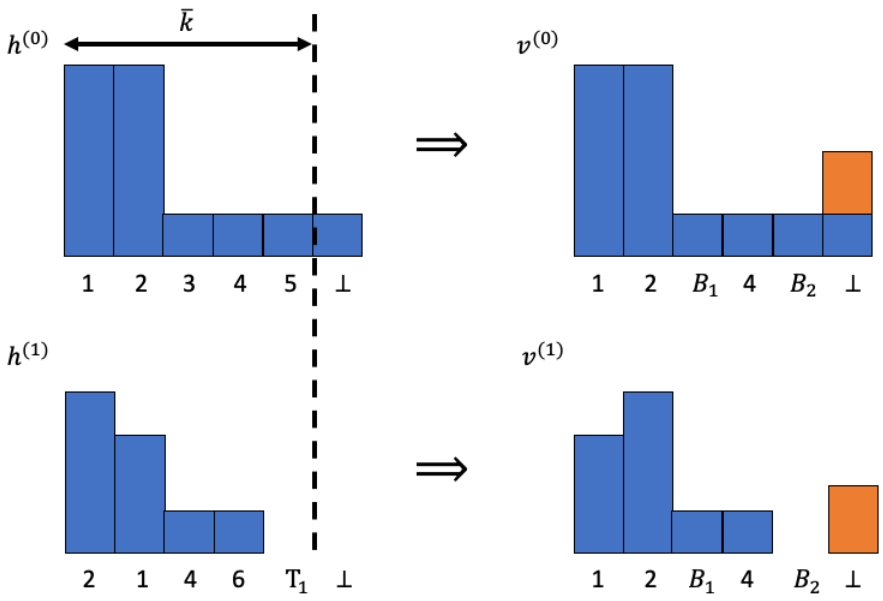

Figure 3: Visualizing the construction of $v^{(b)}$ in $\texttt{GaussMech}_{\perp}^{\bar{k}}(b; \mathbf{h}^{(0)}, \mathbf{h}^{(1)})$. Since labels $3, 5$ are present in $h^{(0)}$ but not $h^{(1)}$ they get relabeled to B_1, B_2 respectively in $v^{(0)}$. Similarly, since the labels $6, \perp_1$ are present in $h^{(1)}$ but not in $h^{(0)}$ they get relabeled to B_1, B_2 in $v^{(1)}$. The additional amount added to $\perp$ in orange denotes the added amount to the count $h_{\perp}$.

F Omitted Portions of Section 7

We point out that Algorithm 3 discovers a new set of items $\mathcal{D}_t$ at each round $t \in [T]$, essentially wiping away the set of items that have already appeared at previous rounds. However, we still ensure the same privacy level if the algorithm remembers previous items that were discovered but may not have noisy count above the threshold at a later round. This would avoid the strange behavior of some items having a count in some rounds and then disappearing in other rounds, however one would need to remember all the items that were previously discovered at each round.

F.1 Privacy Analysis

We now present the full analysis. As we did for $\texttt{UnkGauss}^{\bar{k}}$, we will instead analyze a slight variant of `UnkBase`, which we call $\texttt{UnkBase}_{\top}^{\bar{d}}$, see Algorithm 11. $\texttt{UnkBase}_{\top}^{\bar{d}}$ pads the set of items with dummy items $\{\top_i\}$ that we add noise to in each cell of the partial sum table, so that each cell has the same cardinality $\bar{d}$ of items, which is some upper bound on the dimension of the set of items.

Similar to Lemma 6.2, we will show that simply dropping these dummy items later is equivalent to having never considered them.

Lemma F.1. *Let $h \in \mathbb{N}^p$ be a histogram with labels $\{i_1, \cdots, i_p\}$. Let $M(h)$ be the following for $\bar{d} \geq p$ and $Z_j \overset{i.i.d.}{\sim} \mathcal{P}_j$ and $\hat{h}_j = h_j + Z_j$,*

$$\{(\hat{h}_1, i_1), \cdots, (\hat{h}_p, i_p), (Z_{p+1}, \top_1), \cdots, (Z_{\bar{d}}, \top_{\bar{d}-p})\}.$$

Let $M'(h)$ be the mechanism that drops all items with counts lower than some threshold m and drops any item with label in $\{\top_i\}$. Now let $\hat{M}(h)$ be the same as $M(h)$ except it does not include the $\{\top_i\}$ items. Then $M'(h)$ is equal in distribution to $\hat{M}'(h)$.

Proof. We need to show that adding independent noise to $\bar{d}$ counts, of which some have $\{\top_i\}$ labels and then dropping these terms is equivalent to having never considered those items.

Algorithm 11 $\texttt{UnkBase}^{\bar{d}}_{\top}$; Return a running histogram

Input: Same as `UnkBase` and an upper bound $\bar{d}$.
Output: Noisy histograms $\hat{\mathbf{h}}_{1:T}$.
Use threshold m from (4).
for cell (j,ℓ) in partial histogram table **do**
▷ Create the noisy partial histogram table, pad if necessary
Define $\mathcal{U}_{j,\ell}$ as the set of items in the corresponding substream of $\omega_{1:T}$ and let $d_{j,\ell} = |\mathcal{U}_{j,\ell}|$.
Include dummy items $\top_{j,\ell} = \{\top^1_{j,\ell}, \cdots \top^{\bar{d}-d_{j,\ell}}_{j,\ell}\}$, let $n_{j,\ell} = |\top_{j,\ell}|$.
Form the partial histogram $p_{j,\ell} = (p^u_{j,\ell} : u \in \mathcal{U}_{j,\ell} \cup \top_{j,\ell})$ for this cell.
Add independent noise to each count in this cell to get $\hat{p}_{j,\ell} = (p^u_{j,\ell} + \mathtt{N}\left(0, L_r\tau^2\right) : u \in \mathcal{U}_{j,\ell} \cup \top_{j,\ell})$.
We then have histogram with labels $\hat{\mathbf{p}}_{j,\ell} = \{(\hat{p}^u_{j,\ell}, u) : u \in \mathcal{U}_{j,\ell} \cup \top_{j,\ell}\}$
for $t \in [T]$ **do**
$\mathcal{D}_t = \emptyset$
Let $\mathcal{I}_t(r)$ be set of cells (j,ℓ) that are used in the representation of t with base r.
Let $\mathcal{U}_t$ be the union of items present in each cell used for the count at time t
Let $\top_t$ be the union of dummy items present in each cell.
for each (j,ℓ) cell in $\mathcal{I}_t(r)$ **do**
▷ Relabel dummy items in each cell with items that have newly appeared in the stream
for $u \in \mathcal{U}_t \setminus \mathcal{U}_{j,\ell}$ **do**
Replace the dummy label with largest index $n_{j,\ell}$ to u, i.e. $(\hat{p}^u_{j,\ell}, u) \leftarrow (\hat{p}^{\top^{n_{j,\ell}}_{j,\ell}}_{j,\ell}, \top^{n_{j,\ell}}_{j,\ell})$.
Update $\mathcal{U}_{j,\ell} \leftarrow \mathcal{U}_{j,\ell} \cup \{u\}$, $\top_{j,\ell} \leftarrow \top_{j,\ell} \setminus \{\top^{n_{j,\ell}}_{j,\ell}\}$, and $n_{j,\ell} \leftarrow n_{j,\ell} - 1$.
for $u \in \mathcal{U}_t \cup \top_t$ **do** ▷ Aggregate histograms from each cell $\mathcal{I}_t(r)$
$\hat{h}^u_t = \sum_{(j,\ell)\in\mathcal{I}_t(r)} \hat{p}^u_{j,\ell}$
if $\hat{h}^u_t > m_\delta$ **then** ▷ Only add elements above the threshold
$\mathcal{D}_t \leftarrow \mathcal{D}_t \cup \{u\}$
Return $(\{(\hat{h}^u_t, u) : u \in \mathcal{D}_t\} : t \in [T])$.

Let $f(\cdot)$ be the density function for distribution $\mathcal{P}$, $f_{M'}(\cdot)$ be the density of $\hat{M}$, and $f_{\hat{M}'}$ be the density of $\hat{M}'$. We fix an outcome of counts $(z_1, z_2, \cdots, z_k)$ with $k \le p$ and denote the set of indices that are not in this outcome to be I after dropping counts of $\{\top_i\}$. We then have the density for mechanism $\hat{M}'$ as

$$\begin{aligned} f_{\hat{M}'}(z_1, \cdots, z_k) &= \prod_{i=1}^{k} f_i(z_i - h_i) \int_\infty^m \cdots \int_\infty^m \prod_{\ell\in I} f_\ell(z_\ell - h_\ell) dz_\ell \\ &= \prod_{i=1}^{k} f_i(z_i - h_i) \int_\infty^m \cdots \int_\infty^m \prod_{\ell\in I} f_\ell(z_\ell - h_\ell) dz_\ell \cdot \prod_{j=p+1}^{\bar{k}} \int_{\mathbb{R}} f(z_j) dz_j \\ &= f_{M'}(z_1, \cdots, z_k). \end{aligned}$$

□

Algorithm 12 is a variant of the Gaussian Mechanism that we use when given two neighboring histograms. Note its similarity with Algorithm 10. The algorithm $\texttt{GaussMech}^{\bar{d}}$ takes a parameter $\bar{d}$ which is an upper bound on the number of distinct bins of the histograms, this ensures that each cell has access to a full histogram. We will assume that we have access to the full histogram, including the items with 0 counts, rather than just having the top-$(\bar{k}+1)$ as it was assumed in $\texttt{GaussMech}^{\bar{k}}_{\perp}$.

Proof of Lemma 7.2. In $\texttt{UnkBase}^{\bar{d}}_{\top}(\mathbf{h}^{(b)}; \tau)$, we are essentially applying the Gaussian mechanism to a histogram in each cell of the partial histogram table. Consider a cell (i,j) that differs between streams $\omega^{(0)}_{1:T}$ and $\omega^{(1)}_{1:T}$, of which there can be as many as L_r cells. We apply $\texttt{GaussMech}^{\bar{d}}(b; \mathbf{h}^{(0)}, \mathbf{h}^{(1)}, L_r\tau)$ to this cell's histogram of counts. Doing this across all cells that can actually change between the two neighboring streams, we can apply composition and Lemma 7.1 to get that releasing the full partial sum table is $\frac{\Delta_0}{2\tau^2}$-zCDP.

Algorithm 12 $\texttt{GaussMech}^{\bar{d}}$; Gaussian Mechanism over Full Domain

Input: Bit $b \in \{0,1\}$, neighboring histograms $\mathbf{h}^{(0)}$ and $\mathbf{h}^{(1)}$, upper bound $\bar{d}$, and parameter τ.
Output: Histogram $\hat{\mathbf{v}}$ with labels in $\mathcal{D}^{\bar{d}}_{\top}(\mathbf{h}^{(0)}) \cap \mathcal{D}^{\bar{d}}_{\top}(\mathbf{h}^{(1)})$ and $\{B_\ell : \ell \in [|\mathcal{D}^{\bar{d}}_{\top}(\mathbf{h}^{(b)}) \setminus \mathcal{D}^{\bar{d}}_{\top}(\mathbf{h}^{(1-b)})|]\}$
Let $\mathbf{v}^{(b)} = \emptyset$
We relabel the labels in both $\mathbf{h}^{(0)}$ and $\mathbf{h}^{(1)}$ to form the following histogram $\mathbf{v}^{(b)}$
for $i_{(j)} \in \mathcal{D}^{\bar{d}}_{\top}(\mathbf{h}^{(0)}) \cap \mathcal{D}^{\bar{d}}_{\top}(\mathbf{h}^{(1)})$ **do** ▷ Add common labels
  $\mathbf{v}^{(b)} \leftarrow \mathbf{v}^{(b)} \cup \{(h^{(b)}_{i_{(j)}}, i_{(j)})\}$, where $h^{(b)}_{\top_j} = 0$.

Initialize set of labels that have a non-dummy label and have a different label in the two datasets

$$\mathcal{B} = \left\{ \left\{ \mathcal{D}^{\bar{d}}_{\top}(\mathbf{h}^{(1)}) \setminus \mathcal{D}^{\bar{d}}_{\top}(\mathbf{h}^{(0)}) \right\} \cup \left\{ \mathcal{D}^{\bar{d}}_{\top}(\mathbf{h}^{(0)}) \setminus \mathcal{D}^{\bar{d}}_{\top}(\mathbf{h}^{(1)}) \right\} \right\} \setminus \{\top_i : i \in [\bar{d}]\}$$

for $j \in \mathcal{D}^{\bar{d}}_{\top}(\mathbf{h}^{(b)}) \setminus \mathcal{D}^{\bar{d}}_{\top}(\mathbf{h}^{(1-b)})$ **do** ▷ Add uncommon labels
  if $h^{(b)}_j \geq h^{(1-b)}_j$ **then**
    $\mathbf{v}^{(b)} \leftarrow \mathbf{v}^{(b)} \cup \{(h^{(b)}_j, j)\}$.
  else
    Select a label from $\mathcal{B}$, call it a
    $\mathbf{v}^{(b)} \leftarrow \mathbf{v}^{(b)} \cup \{(0, a)\}$.
    $\mathcal{B} \leftarrow \mathcal{B} \setminus \{a\}$
Add $\texttt{N}\left(0, \tau^2\right)$ to each count in $\mathbf{v}^{(b)}$ to form the noisy histogram $\hat{\mathbf{v}}$ ▷ Gaussian Mechanism
Return $\hat{\mathbf{v}}$

We then apply a post-processing function that adds up the corresponding cells of the table to get the aggregate count for each time $t \in [T]$ and removes any item that has count lower than m. Because we are only considering good outcomes, this will ensure that any count with a bad label $\{B_i\}$ is not in the result, hence bad labels have noisy counts less than m. Note that these bad labels are the only terms that could have had different labels than the counts returned in $\texttt{UnkBase}^{\bar{d}}_{\top}(\omega^{(b)}_{1:T}; \tau)$. Hence, the resulting mechanism is equivalent to $\texttt{UnkBase}^{\bar{d}}_{\top}(\omega^{(b)}_{1:T}; \tau)$ for outcomes in $\mathcal{G}_{\texttt{Unk}}$. □

We next need to figure out the right threshold m to set that will ensure that bad outcomes occur with negligible probability. The only way an item u that occurred once in a stream $\omega_{1:T}$ but not in another neighboring stream $\omega'_{1:T}$ can be returned is if there is a noisy count $\hat{h}^u_t > m$ for some $t \in [T]$ and item u that was present in ω but not in ω' or vice versa. Since the counts are computed as a function of the partial histogram table $\hat{\mathbf{p}}$, we need to make sure that all the noisy counts in this table for items that are not common in ω and ω' cannot add up to something larger than m.

Proof of Lemma 7.3. We first consider the probability that an item from $\mathcal{B}^{(b)}_{\texttt{Unk}}$ can be returned at a given time t, which must mean that there is a dummy label $\top^i_t$ for one stream and a real label a for the other stream at time t. Let's consider the first time t that the labels do not align in the neighboring streams. The only way this could happen is if this were the first time a appeared in the stream, since all prior items in both streams are the same. Hence, the true count of a at time t will be 1 in one stream and 0 in the other. The additional noise must have caused its count to appear above the threshold m. We then compute the probability that a count can appear above threshold m. This threshold then needs to be set so that all future times will also not have noisy count on a above the threshold until someone else has item a in the stream. Further, there can be at most Δ_0 many items like a, implying that all items that a user contributes at a round t are all the first time they appeared in the stream.

There are multiple ways to do this. One is to bound the probability that all of the independent Gaussians that are used to compute the count for item a are below $m/\left((r-1)L_r\right)$, hence any sum of at most $(r-1)L_r$ terms is below m. Another way, is to bound the probability that any sum of these independent Gaussians is below m,

and take a union bound over all T rounds. We opt for the latter approach.

$$\begin{aligned}
&\Pr[\texttt{UnkBase}^{\bar{d}}_{\top}(\omega^{(b)}_{1:T};\tau,\delta)\in\mathcal{B}^{(b)}_{\texttt{Unk}}] = \Pr[\exists a\in\mathcal{B}^{(b)}_{\texttt{Unk}} \text{ s.t. label } a\in\texttt{UnkBase}^{\bar{d}}_{\top}(\omega^{(b)}_{1:T};\tau,\delta)]\\
&\leq \Pr_{\{Z^u_{i,j}\}\overset{i.i.d.}{\sim}\mathtt{N}(0,L_r\tau^2)}\left[\max_{t\in[T],u\in\mathcal{B}^{(b)}_{\texttt{Unk}}}\left\{1+\sum_{(j,\ell)\in\mathcal{I}_t(b)}Z^u_{i,j}\right\}>m_\delta\right]\\
&\leq \Delta_0 T\cdot\Pr\left[\mathtt{N}\left(0,(r-1)L_r^2\tau^2\right)>m_\delta-1\right]\\
&= \Delta_0 T\cdot\left(1-\Phi\left(\frac{m_\delta-1}{L_r\tau\sqrt{r-1}}\right)\right).
\end{aligned}$$

The last inequality follows from a union bound. Setting m_δ as in (4) gives the bound of $\Delta_0\delta$. □

Proof of Theorem 6. We first show that $\texttt{UnkBase}^{\bar{d}}_{\top}$ is DP. This follows by applying Lemma 6.1 with Lemmas 7.2 and 7.3. Now we use Lemma F.1 to show that at any round t, dropping the dummy labels $\top_t$ is equivalent to never adding noise to them. However, we may use the noise allocated for a dummy item in some cells at later rounds. In particular, we replace the dummy label when a new item appears at a later point in $\texttt{UnkBase}^{\bar{d}}_{\top}$. Whether this noise was drawn earlier for that cell or at the time that it is actually used, both give the same distribution. Hence, the post-processing function of dropping dummy labels at each round of $\texttt{UnkBase}^{\bar{d}}_{\top}(\omega_{1:T};\tau,\delta)$ is equivalent to running $\texttt{UnkBase}(\omega_{1:T};\tau,\delta)$. □

F.2 Utility Analysis

We then turn to analyzing the utility of UnkBase. First we consider the probability that a particular item will appear in the result at time t.

Lemma F.2. *Let $\mathcal{D}_t$ be the discovered set at round t in $\texttt{UnkBase}(\omega_{1:T};\tau,\delta)$. Let $h^u_t=\sum_{\ell=1}^t\mathbb{1}\{u\in\omega_\ell\}$ be the true count for item u in the stream up to round t and assume that it is larger than the threshold $h^u_t=m_\delta+c\cdot\tau$ for some $c>0$. We can then bound the probability that u is part of the discovered set at time t,*

$$\Pr[u\in\mathcal{D}_t]\geq\Phi\left(\frac{c}{\sqrt{r-1}L_r}\right).$$

Proof. We will write $\mathcal{I}_r(t)$ as the set of indices in the partial histogram table that gets used to compute the counts at time t. Recall that we will add noise $\mathtt{N}\left(0,|\mathcal{I}_r(t)|L_r\tau^2\right)$ to the true count $h^u_t=m_\delta+c\cdot\tau$ at time t. We then need to ensure that the noisy count will be above the threshold m_δ given in (4). Hence, we have

$$\begin{aligned}
\Pr[u\in\mathcal{D}_t] &= \Pr[\mathtt{N}\left(h^u_t,|\mathcal{I}_r(t)|L_r\tau^2\right)>m_\delta] = 1-\Phi\left(\frac{-c}{\sqrt{|\mathcal{I}_r(t)|L_r}}\right)\\
&= \Phi\left(\frac{c}{\sqrt{|\mathcal{I}_r(t)|L_r}}\right)\geq\Phi\left(\frac{c}{\sqrt{r-1}L_r}\right)
\end{aligned}$$

□

Additionally, for those items $u\in\mathcal{U}$ that the algorithm releases, we provide bounds on the difference between their true count h^u_t and their noisy count $\hat{h}^u_t$ by noticing that $\hat{h}^u_t$ is the original count h^u_t plus Gaussian noise truncated at threshold m_δ.

Lemma F.3. *Given a stream $\omega_{1:T}$ and an item $u\in\mathcal{D}_t$ that is part of the discovered set from $(\hat{h}^u_t,u)\in\texttt{UnkBase}(\omega_{1:T};\tau,\delta)$ at time t, we can then bound the error on its true count $h^u_t=m_\delta+c\cdot\tau$ at time t for $0<c<\eta$ with high probability,*

$$\begin{aligned}
&\Pr[|h^u_t-\hat{h}^u_t|\geq\eta\tau\mid u\in\mathcal{D}_t]\\
&\leq\frac{1-\Phi\left(\frac{\eta}{\sqrt{(r-1)}L_r}\right)}{\Phi\left(\frac{c}{\sqrt{(r-1)}L_r}\right)}.
\end{aligned}$$

Proof. Note that $\hat{h}_t^u$ is the original count h_t^u plus Gaussian noise truncated at the threshold m_δ. More specifically, let $\mathcal{I}_r(t)$ be the set of indices in the partial histogram table that get used to compute the counts at time t. We then have $\hat{h}_t^u$ is distributed as a truncated (at m_δ) Gaussian with mean h_t^u and variance $|\mathcal{I}_r(t)|L_r\tau^2$. Using the fact that $|\mathcal{I}_r(t)| \leq (r-1)L_r$ we have

$$\begin{aligned}
\Pr[|h_t^u - \hat{h}_t^u| \geq \eta\tau \mid u \in \mathcal{D}_t] &= \Pr_{Z \sim \mathtt{N}(h_t^u, \mathcal{I}_r(t)L_r\tau^2)}[|Z - h_t^u| \geq \eta\tau \mid Z > m_\delta] \\
&\leq \Pr[Z < h_t^u - \eta\tau \mid Z > m_\delta] + \Pr[Z > h_t^u + \eta\tau \mid Z > m_\delta] \\
&= \frac{2\left(1 - \Phi\left(\frac{\eta}{\sqrt{\mathcal{I}_r(t)L_r}}\right)\right) - \Phi\left(\frac{-c}{\sqrt{\mathcal{I}_r(t)L_r}}\right)}{1 - \Phi\left(\frac{-c}{\sqrt{\mathcal{I}_r(t)L_r}}\right)} \\
&= \frac{1 - 2\Phi\left(\frac{\eta}{\sqrt{\mathcal{I}_r(t)L_r}}\right) + \Phi\left(\frac{c}{\sqrt{\mathcal{I}_r(t)L_r}}\right)}{\Phi\left(\frac{c}{\sqrt{\mathcal{I}_r(t)L_r}}\right)} \\
&\leq \frac{1 - \Phi\left(\frac{\eta}{\sqrt{\mathcal{I}_r(t)L_r}}\right)}{\Phi\left(\frac{c}{\sqrt{\mathcal{I}_r(t)L_r}}\right)} \\
&\leq \frac{1 - \Phi\left(\frac{\eta}{\sqrt{r-1}L_r}\right)}{\Phi\left(\frac{c}{\sqrt{r-1}L_r}\right)}
\end{aligned}$$

□