The Price of Differential Privacy under Continual Observation

Palak Jain [*1] **Sofya Raskhodnikova** [*1] **Satchit Sivakumar** [*1] **Adam Smith** [*1]

Abstract

We study the accuracy of differentially private mechanisms in the continual release model. A continual release mechanism receives a sensitive dataset as a stream of T inputs and produces, after receiving each input, an output that is accurate for all the inputs received so far. We provide the first strong lower bounds on the error of continual release mechanisms. In particular, for two fundamental problems that are closely related to empirical risk minimization and widely studied and used in the standard (batch) model, we prove that the worst case error of every continual release algorithm is $\tilde{\Omega}(T^{1/3})$ times larger than that of the best batch algorithm. Previous work shows only a $\Omega(\log T)$ gap between the worst case error achievable in these two models. We also formulate a model that allows for adaptively selected inputs, thus capturing dependencies that arise in many applications of continual release. Even though, in general, both privacy and accuracy are harder to attain in this model, we show that our lower bounds are matched by the error of simple algorithms that work even for adaptively selected inputs.

1. Introduction

Differentially private (DP) data analysis (Dwork, McSherry, Nissim, and Smith, 2006b) studies the design of algorithms that publish aggregate statistics about input datasets while preserving the privacy of individuals whose data they contain. The published aggregates often include learning models trained on the private data, as in Apple's auto-complete feature, in Google's search query suggestions, and large-capacity language models deployed by Microsoft (Apple, 2017; Yang et al., 2018; Yu et al., 2022).

Many deployments of differential privacy operate in the *batch model*: that is, they collect their input all at once and produce a single output. However, in numerous situations, sensitive data are collected over time, and published models and statistics need to be updated regularly. Examples include COVID-19 dashboards that display statistics about the number of COVID cases and deaths, ad campaign analytics, recommendation systems, and predictive language models. To investigate privacy in these situations, Dwork, Naor, Pitassi, and Rothblum (2010a) and Chan, Shi, and Song (2011) introduced the *continual release* model (sometimes referred to as the *continual observation* model). In the continual release model, a mechanism receives a sensitive dataset as a stream of T input records and produces, after receiving each record, an accurate output on the obtained inputs. Intuitively, the mechanism is DP if releasing the entire vector of T outputs satisfies differential privacy. Some deployments of differential privacy already fit this model (Apple, 2017). Furthermore, the continual release model arises as an intermediate step inside the analysis of some batch algorithms, such as the DP-FTRL learning algorithm (Kairouz et al., 2021). Despite the model's prevalence, the theoretical understanding of continual release is still limited, and only a handful of techniques have been developed to tackle it. The main challenge for privacy in this model is that each individual record contributes to outputs at multiple time steps.

Dwork et al. (2010a) and Chan et al. (2011) considered the problem of computing summation in the continual release model when each record consists of one bit. Dwork et al. (2010a) showed that an error of $\Omega(\log T)$ is necessary to privately release all running sums. Additionally, both works designed the *binary tree mechanism*, a continual release mechanism that achieves (additive) error $O(\log^2 T)$ for this problem. Since then, this mechanism has been shown to accurately solve (with $\operatorname{polylog} T$ error) many problems in the continual release model. (Further related work is discussed in Appendix B.) Given this success, one might conjecture that these results could be extended to a wide range of problems or at least to problems closely related to summation. Indeed, the largest previously known gap between the worst case error achievable in the batch and continual release models is $\Omega(\log T)$, ex-

[*]Equal contribution [1]Department of Computer Science, Boston University, Boston, Massachusetts, USA. Correspondence to: Palak Jain <palakj@bu.edu>, Sofya Raskhodnikova <sofya@bu.edu>, Adam Smith <ads22@bu.edu>, Satchit Sivakumar <satchit@bu.edu>.

Proceedings of the 40^{th} International Conference on Machine Learning, Honolulu, Hawaii, USA. PMLR 202, 2023.

hibited by summation.

1.1. Our Contributions

We ask what price DP algorithms must pay in accuracy to solve a problem in the continual release model instead of the batch model. We show that for two fundamental problems related to summation and widely studied in the batch model, the gap is exponentially larger than the $\log T$ separation for summation. In addition, we formalize a more realistic and broadly applicable model of continual release that allows for adaptively chosen inputs. Surprisingly, for the problems we consider, there is no increase in the error when we step up our privacy and accuracy requirements to deal with this more challenging setting.

In the first problem, called MaxSum, each input consists of d binary attributes[1] and the goal is to approximate the maximum of the attribute sums. We define the error of a continual-release mechanism as the maximum error over all the time steps. For MaxSum, the error at each time step is the absolute value of the difference between the true answer and the output of the mechanism at that time step. The second problem, $\mathsf{SumSelect}$, is the "argmax" version of MaxSum: the goal is to find the index of the largest attribute sum. The error at a particular time step for this problem is the absolute difference between the maximum sum and the attribute sum for the index returned by the mechanism at that time step.

Motivation Both problems are abstractions of practically relevant tasks. For instance, consider a company monitoring the performance of d predictive models on a sequence of labeled examples: if for each data point we record whether each model made a successful prediction, then MaxSum corresponds to the success rate of the best model, and $\mathsf{SumSelect}$ corresponds to empirical risk minimization—that is, the index of the best model. If the data collected by a public health agency consists of records indicating which of d medical conditions each person suffers from, then MaxSum corresponds to the number of cases of the most prevalent condition so far, and $\mathsf{SumSelect}$ corresponds to the name of this condition.

Algorithms for both tasks are key ingredients in differentially private solutions to more complex problems such as empirical risk minimization (Bassily et al., 2014), synthetic data generation (Hardt et al., 2012), and high-dimensional optimization (Talwar et al., 2015). Consequently, these tasks and their variants have been thoroughly investigated in several models. Known algorithms and lower bounds for these tasks provide pivotal pieces of our current understanding of the central model (McSherry & Talwar, 2007; Bafna & Ullman, 2017; Steinke & Ullman, 2017; Durfee & Rogers, 2019; Qiao et al., 2021), the local model (Kasiviswanathan et al., 2011; Duchi et al., 2013; Edmonds et al., 2020), the shuffle model, and the pan-private model (Cheu & Ullman, 2021). Additionally, accurate continual-release variants of $\mathsf{SumSelect}$, if they existed, would enable variants of the DP-FTRL learning framework (Kairouz et al., 2021) for a wider range of parameter spaces than are currently known.

We prove tight bounds on the error for these two problems in the continual release model in terms of the stream length (or "time horizon") T, the number of attributes (or the *dimension*) d, and the privacy parameter ε. To provide a comparison to the continual release model, we assume that algorithms in the batch model get input datasets of size T. Intuitively, a batch algorithm $\mathcal{A}$ is differentially private if, for all datasets $\mathbf{x}$ and $\mathbf{x}'$ that differ in one record, all events under the distributions $\mathcal{A}(\mathbf{x})$ and $\mathcal{A}(\mathbf{x}')$ have similar probabilities. The definitition of differential privacy (Definition 2.2) takes two parameters: ε and δ. The setting where $\delta = 0$ is referred to as *pure differential privacy*. To provide a meaningful privacy guaranty in the setting where $\delta > 0$ (referred to as *approximate differential privacy*, the parameter δ has to be small (Kasiviswanathan & Smith, 2014); in our case, $\delta = o(\varepsilon/T^2)$.) For continual release mechanisms, we study *event-level* privacy, where each user's data appears in a single record, as opposed to *user-level* privacy, where a user's data could be distributed over multiple records. See Dwork et al. (2010a) for discussion of these two variants.

Separation Between the Continual Release and the Batch Models We demonstrate a strong separation between the continual release and the batch models. A comparison of the error achievable in the two models is presented in Table 1. The first row gives results on $\mathsf{Summation}$ from previous work; the second and the third row give results on MaxSum and $\mathsf{SumSelect}$. The first column summarizes the error in the batch model: $O(1)$ for MaxSum and $O\big(\log d\big)$ for $\mathsf{SumSelect}$. The former is obtained by an instantiation of the Laplace mechanism of Dwork et al. (2006b) and the latter, by an instantiation of the exponential mechanism of McSherry & Talwar (2007). In contrast, we show that in the continual release model, these tasks require error that is polynomial in either T or d; this is presented in the second column of the table (the results shown in this column are for approximate differential privacy).

More detailed versions of our results are given in Table 2. For approximate differential privacy, we show that when d is sufficiently large, MaxSum_d and $\mathsf{SumSelect}_d$ require error that scales as roughly $\tilde{\Omega}(\sqrt[3]{T})$. For pure differential privacy, the error required for MaxSum and $\mathsf{SumSelect}$ is even larger, roughly $\tilde{\Omega}(\sqrt{T})$.

[1] Our algorithms and analyses work more generally, when inputs are from $[0, 1]^d$ (not just $\{0, 1\}^d$).

Table 1. Bounds on the error of (ε, δ)-DP mechanisms in the continual release model compared to their batch model counterparts. The corresponding upper and lower bounds differ only in the $\text{polylog}(T)$ terms, highlighted in blue. For approximate differential privacy, the lower bounds apply when $\delta = o(\varepsilon/T^2)$, and the upper bounds apply when $\delta > \text{poly}(\frac{1}{T})$. For simplicity, the dependence on ε is suppressed in the table.

	Batch Model	Continual Release Model		Reference
		Lower Bounds	Upper Bounds	
Summation	$\Theta(1)$	$\Omega(\log T)$	$O(\log^2 T)$	Dwork et al. (2010a) Chan et al. (2011)
MaxSum	$\Theta(1)$	$\tilde{\Omega}\left(\min\left\{\sqrt[3]{T}, \sqrt{d}\right\}\right)$	$\tilde{O}\left(\min\left\{\sqrt[3]{T},\ \sqrt{d}\,\text{polylog}(T)\right\}\right)$	Thm. 3.1 Cor. E.3, E.6
SumSelect	$\Theta(\log d)$	$\tilde{\Omega}\left(\min\left\{\sqrt[3]{T\log^2 d}, \sqrt{d}, T\right\}\right)$	$\tilde{O}\left(\min\left\{\sqrt[3]{T\log^2 d}, \sqrt{d}\,\text{polylog}(T), T\right\}\right)$	Thm. 4.1 Cor. E.3, E.6

Table 2. More detailed statements of our results on the additive error of $(\varepsilon, 0)$-DP and (ε, δ)-DP mechanisms in the continual release model. The corresponding upper and lower bounds differ only in the $\text{polylog}(T)$ terms, highlighted in blue. For approximate differential privacy, the lower bounds apply when $\delta = o(\varepsilon/T^2)$, and the upper bounds apply when $\delta > \text{poly}(\frac{1}{T})$. All our results are stated and proved for $\varepsilon < 1$, but they apply for ε bounded by any larger constant as well.

	Approximate DP ($\delta > 0$)	Pure DP ($\delta = 0$)	Reference
MaxSum	$\tilde{\Omega}\left(\min\left\{\sqrt[3]{\frac{T}{\varepsilon^2}}, \frac{\sqrt{d}}{\varepsilon}, T\right\}\right)$	$\tilde{\Omega}\left(\min\left\{\sqrt{\frac{T}{\varepsilon}}, \frac{d}{\varepsilon}, T\right\}\right)$	Thm. 3.1
	$\tilde{O}\left(\min\left\{\sqrt[3]{\frac{T}{\varepsilon^2}}, \frac{\sqrt{d}\,\text{polylog}(T)}{\varepsilon}, T\right\}\right)$	$\tilde{O}\left(\min\left\{\sqrt{\frac{T}{\varepsilon}}, \frac{d\,\text{polylog}(T)}{\varepsilon}, T\right\}\right)$	Cor. E.3, E.6
SumSelect	$\tilde{\Omega}\left(\min\left\{\sqrt[3]{\frac{T\log^2 d}{\varepsilon^2}}, \frac{\sqrt{d}}{\varepsilon}, T\right\}\right)$	$\tilde{\Omega}\left(\min\left\{\sqrt{\frac{T\log d}{\varepsilon}}, \frac{d}{\varepsilon}, T\right\}\right)$	Thm. 4.1
	$\tilde{O}\left(\min\left\{\sqrt[3]{\frac{T\log^2 d}{\varepsilon^2}}, \frac{\sqrt{d}\,\text{polylog}(T)}{\varepsilon}, T\right\}\right)$	$\tilde{O}\left(\min\left\{\sqrt{\frac{T\log d}{\varepsilon}}, \frac{d\,\text{polylog}(T)}{\varepsilon}, T\right\}\right)$	Cor. E.3, E.6

Lower Bound Technique: Reductions via Sequential Embedding We obtain our lower bounds via a novel sequential embedding technique. This approach embeds multiple separate instances of an appropriately chosen base problem *on the same sensitive dataset* in the batch model into a single instance of a continual release problem. It allows us to use a continual release algorithm to solve multiple instances of the base problem. We can then invoke lower bounds for the batch model to obtain hardness results in the continual release model. For both MaxSum and SumSelect, we can adjust the parameters of our reductions to get nearly tight lower bounds for all values of the dimension d and time horizon T for both pure and (ε, δ)-differential privacy.

For MaxSum, the corresponding base problem is to compute one of the sums over which the maximum is taken. Such sums correspond to 1-way marginals of a dataset, and we apply lower bounds for releasing all 1-way marginals in the batch model by Hardt & Talwar (2010) (for pure differential privacy) and Bun et al. (2018) (for (ε, δ)-differential privacy) in conjunction with our reduction. For SumSelect, the base task in the batch model is selecting the index of the largest coordinate sum restricted to a subset of coordinates. Multiple disjoint instances of the base task are captured by a problem we call k-IndSelect$_d$, where k indicates the number of instances. To obtain lower bounds for this problem in the batch model, we use a simple packing argument in the case of pure differential privacy and prove a new lower bound for selecting top-k sums using a result of Steinke & Ullman (2017) for a related problem in the case of (ε, δ)-differential privacy.

The main idea behind our sequential embedding technique is to embed a sensitive dataset into the first part of the stream given to the continual release algorithm. The remaining parts of the stream do not depend on the dataset and are chosen to force the continual release algorithm to output good approximations to separate instances of the base problem on the sensitive dataset. The technique was subsequently used to obtain lower bounds for other problems (Ghazi et al., 2023); see Section 1.2.

Continual Release with Adaptively Chosen Inputs The continual release model of Dwork et al. (2010a) and Chan et al. (2011) assumes that the input stream is chosen *obliviously*, before the algorithm is run. This means that the data at time t cannot depend on the values the algorithm returned at prior steps, even though they arrive after earlier outputs are released. We formalize a more realistic model that allows the stream elements to be chosen online by an adversary that observes the outputs of the continual release algorithm. This model gives a more faithful representation of real-life settings, where the data may change based on prior feedback from the algorithm. For example, a city might adjust its social distancing policy based on DP estimates of COVID cases, which could in turn affect the number of new cases. Recommendations by Amazon might change what the customers are buying and affect which products are most popular. The new model captures these and many other scenarios better than the original continual release model and is in line with the explosion of research on adversarially robust streaming (Mironov et al., 2011; Ben-Eliezer et al., 2022a;b; Hassidim et al., 2022; Cohen et al., 2022; Beimel et al., 2022). The model where input is chosen adaptively has been considered implicitly in previous work, specifically, in the application of continual-release algorithms for summation to online learning (Thakurta & Smith, 2013). In Section 5.1, we provide a general, explicit game-based formulation of the *continual release model with adaptively chosen inputs*.

In general, both privacy and accuracy are harder to attain in this model[2]. However, for the specific problems we consider, it turns out that there is no overhead in terms of accuracy when the input stream is selected adaptively. We show that our lower bounds (that hold even when the input stream is selected obliviously) are matched by algorithms that work even with adaptively selected streams. We achieve this by analyzing variants of classical continual release algorithms in the new model.

Each of our lower bounds (summarized in Table 2) is the minimum of three terms, corresponding to different parameter regimes. They are matched (up to polylogarithmic factors in T and $1/\delta$), in each regime, by the best of two simple mechanisms and one trivial mechanism. The trivial mechanism always outputs an arbitrary value in the right range. The first simple mechanism is based on recomputing the value of the desired statistic (e.g., MaxSum) at regular intervals and providing the same answer until it is recomputed again. The second simple mechanism uses the binary tree mechanism to track all d coordinates separately and takes the maximum (or, in the case of $\mathsf{SumSelect}$, argmax) of the noisy values. We analyze these mechanisms in the continual release model with adaptively chosen inputs for MaxSum, $\mathsf{SumSelect}$, and general functions of sensitivity 1 in Appendix E .

Privacy analysis in the adaptive setting is subtle. In the nonadaptive setting, one argues that the algorithm's outputs on any two fixed datasets that differ in exactly one element are *indistinguishable*. In contrast, the two input streams in the adaptive setting may diverge in an arbitrary number of records based on prior outputs of the mechanism. Because of this, one cannot generally reduce the proof of privacy with adaptively chosen inputs to a proof in the nonadaptive model. Instead, we build on techniques from simulation-based proofs in cryptography to argue directly, for specific algorithms, that the joint distributions of the two input streams and their corresponding output distributions are indistinguishable.

We note that adaptive composition—a standard tool in the analysis of differentially private mechanisms—is inadequate for dealing with the issue of adaptively chosen inputs. A continual release mechanism does not have to use independent randomness at each time step so one cannot, in general, apply adaptive composition directly.

1.2. Discussion and Open Questions

This paper has two key contributions: first, we establish strong lower bounds on the error of continual release mechanisms; second, we introduce the model with adaptively chosen inputs and then analyze algorithms for MaxSum and $\mathsf{SumSelect}$ in this model. Together, our mechanisms and lower bounds provide a comprehensive characterization of the error for MaxSum and $\mathsf{SumSelect}$ up to polylogarithmic factors in T and $1/\delta$ across all parameter regimes.

Our new model is relevant in practical settings: our analysis applies to a deployed (binary-tree based) machine learning protocol by Kairouz et al. (2021). Furthermore, a follow-up on our work by Denisov et al. (2022) investigates improvements to the binary tree mechanism that have been deployed and proves they are private in the model with adaptively chosen inputs. The structure of the mechanisms they consider requires the use of the adaptive model in the analysis even when the data are fixed ahead of time.

In the course of their work, Denisov et al. (2022) delve deeper into the model with adaptively chosen inputs: they show that any mechanism that adds additive Gaussian noise is private in the adaptive model; they also construct an (artificial) protocol that is DP in the continual release model with nonadaptive inputs, but not DP with adaptive inputs. It is open to separate the two continual release models in the sense our work separates the batch model and the continual release model: by providing problems that require a large error blowup in the more demanding model.

Our lower bounds point to fundamental differences between the continual release and the batch models. In the batch model, low sensitivity of a function can be easily exploited to provide an accurate DP algorithm for releasing

[2] In a subsequent work, Denisov et al. (2022) give an example of a protocol that is private in the original continual release model, but not in the model with adaptively chosen inputs.

this function. It was consistent with prior work that a version of the widely-used exponential mechanism tailored to $\mathsf{SumSelect}$ in the continual release model could match the accuracy of summation. Our work rules out this possibility. We show that in the continual release model, low sensitivity alone is insufficient. To get DP mechanisms with very low (say, polylogarthmic in T) error for problems that do not reduce to low-dimensional summation or histograms, one must find and exploit new kinds of structure in the function being repeatedly evaluated.

Notably, our lower bounds apply even to "offline" algorithms that receive the entire input stream before producing output; i.e., they do not rely on the algorithm's uncertainty of what comes later in the stream. It would be interesting to find natural problems that separate the offline and the continual release models, as discussed in Denisov et al. (2022).

Finally, our sequential embedding technique has already found application in follow-up work of Ghazi et al. (2023), where it is used to establish lower bounds for other practically important problems such as counting distinct elements.

1.3. Organization of Paper

In Section 2, we define the continual release model with nonadaptively chosen inputs and state the problems we consider. Additional preliminaries appear in Appendix A. Related work not covered in the introduction is described in Appendix B. Sections 3–5 present our technical results and the definition of the continual release model with adaptively chosen inputs. All omitted proofs appear in appendices.

2. Definitions

2.1. Continual Release with Nonadaptively Chosen Inputs

A *mechanism* in the continual release model (Dwork et al., 2010a; Chan et al., 2011) is an algorithm that receives its input $\mathbf{x} = (x_1, \ldots, x_T) \in \mathcal{X}^T$ as a stream. At each time step $t \in [T]$, it gets a record x_t and outputs an answer a_t. The output stream $(a_1, \ldots, a_T)$ is denoted by $\mathbf{a}$. We use $\mathbf{x}_{[t]} = (x_1, \ldots, x_t)$ for $t \in [T]$ to denote the first t records in a stream $\mathbf{x}$ (similarly, $\mathbf{a}_{[t]} = (a_1, \ldots, a_t)$.) The total number of records in the stream, denoted by T, is called the *time horizon*. For simplicity, we assume T is known to the mechanism.

We consider two variants of the continual release model. (1) The continual release model of Dwork et al. (2010a) and Chan et al. (2011): This model assumes that the input stream $\mathbf{x}$ is fixed before the mechanism runs. This means that the data at time t cannot depend on the values the algorithm returned at prior steps, even though they arrive after earlier outputs are released. (2) The continual release model with adaptively chosen inputs: This model allows an adversary to choose each input record x_t for $t \in \{2, \ldots, T\}$ based on the previous outputs $a_1, \ldots, a_{t-1}$ of the mechanism. This model gives a more faithful representation of real-life settings, where the data may change based on prior feedback from the algorithm. We formalize this model in Section 5.1.

All our lower bounds are for the model with nonadaptively chosen inputs and, consequently, imply the same lower bounds for the model with adaptively chosen inputs. In contrast, all our algorithms work with adaptively chosen inputs (and, consequently, in the special case when inputs are chosen nonadaptively).

We refer to standard algorithms that get their input in one batch and produce one output as *batch* algorithms. For clarity, we refer to continual release algorithms as *mechanisms*.

Accuracy We start by defining how well a given output approximates the value of a function. We use a notion of error that depends on the function. Given a function $f : \mathcal{X}^* \to \mathcal{Y}$, a dataset $\mathbf{x} \in \mathcal{X}^*$, and an answer $a \in \mathcal{Y}$, let $\mathsf{ERR}_f(\mathbf{x}, a)$ be a nonnegative number that quantifies how far off a is from $f(\mathbf{x})$. Specifically, when $\mathcal{Y} = \mathbb{R}^k$,

$$\mathsf{ERR}_f(\mathbf{x}, a) = \|f(\mathbf{x}) - a\|_\infty. \tag{1}$$

Later (in (2)), we define a different notion of error for the optimization problem $\mathsf{SumSelect}$. The error for an optimization problem corresponds to the deficit in the objective function.

Definition 2.1 (Accuracy of a Mechanism)**.** *In the continual release model with nonadaptively chosen inputs, a mechanism $\mathcal{M}$ is* (α, T)-accurate *for f if, for all fixed input streams $\mathbf{x} = (x_1, \ldots, x_T)$, the maximum error $\mathsf{ERR}_f(\mathbf{x}_{[t]}, a_t)$ over the outputs $a_1, \ldots, a_T$ of mechanism $\mathcal{M}$ is bounded by α with high probability, that is,*

$$\Pr_{\text{coins of } \mathcal{M}}\left[\max_{t \in [T]} \mathsf{ERR}_f(\mathbf{x}_{[t]}, a_t) \le \alpha\right] \ge \frac{2}{3}.$$

Privacy Finally, we define privacy in the continual release model with nonadaptively chosen inputs.

Definition 2.2 (Privacy of a Mechanism)**.** *Given a mechanism $\mathcal{M}$, define $\mathcal{A}_\mathcal{M}$ to be the batch model algorithm that receives an input dataset $\mathbf{x}$, runs $\mathcal{M}$ on stream $\mathbf{x}$, and returns the output stream $\mathbf{a}$ of $\mathcal{M}$. The mechanism $\mathcal{M}$ is* (ε, δ)-differentially private (DP) in the continual release model with nonadaptively chosen inputs *if $\mathcal{A}_\mathcal{M}$ is (ε, δ)-DP in the batch model.*

Definition 2.2 refers to *event-level* privacy, where each user's data appears in a single record, as opposed to *user-*

level privacy, where a user's data could be distributed over multiple records.

2.2. Problem Definitions

We consider two functions on datasets, where each record consists of d binary attributes. The first function, MaxSum_d, returns the maximum attribute sum for the input records. The second function, $\mathsf{SumSelect}_d$, returns the index of such a maximum sum.

Definition 2.3. *Let $d \in \mathbb{N}$ and $\mathcal{X} = \{0,1\}^d$. For a dataset $\mathbf{x} \in \mathcal{X}^*$ and $j \in [d]$, the j^{th}* attribute *of record x_i is its j^{th} coordinate, denoted $x_i[j]$. Let $t \in \mathbb{N}$ and $\mathbf{x}_{[t]} \in \mathcal{X}^t$. The function $\mathsf{MaxSum}_d : \mathcal{X}^* \to \mathbb{N}$ is*

$$\mathsf{MaxSum}_d(\mathbf{x}_{[t]}) \overset{def}{=} \max_{j\in[d]} \Big(\sum_{i\in[t]} x_i[j] \Big).$$

The function $\mathsf{SumSelect}_d : \mathcal{X}^* \to [d]$ is

$$\mathsf{SumSelect}_d(\mathbf{x}_{[t]}) \overset{def}{=} \arg\max_{j\in[d]} \Big(\sum_{i\in[t]} x_i[j] \Big).$$

If multiple indices j attain the maximum sum, the function value is defined to be the smallest such index.

We study the accuracy of differentially private algorithms for computing these two functions. Our accuracy goal, stated in Definition 2.1, uses the notion ERR_f. We define the error $\mathsf{ERR}_{\mathsf{MaxSum}}$ as in (1). For $\mathsf{SumSelect}$, it is defined by:

$$\mathsf{ERR}_{\mathsf{SumSelect}}(\mathbf{x}_{[t]}, a_t) = \mathsf{MaxSum}_d(\mathbf{x}_{[t]}) - \sum_{i\in[t]} x_i[a_t]. \tag{2}$$

3. Lower Bounds for MaxSum

In this section, we prove Theorem 3.1 that provides strong lower bounds on the accuracy parameter α for any accurate mechanism for MaxSum_d in the continual release model with nonadaptively chosen inputs. Our lower bounds match the upper bounds from Section 5 for MaxSum_d in the continual release model with adaptively chosen inputs up to logarithmic factors in the time horizon T and the number of coordinates d.

Theorem 3.1. *For all $\varepsilon \in (0,1], \delta \in [0,1), \alpha \geq 0, d \in \mathbb{N}$, sufficiently large $T \in \mathbb{N}$, and mechanisms $\mathcal{M}$ in the continual release model with nonadaptively chosen inputs that are (ε,δ)-differentially private and (α,T)-accurate for MaxSum_d, the following statements hold.*

1. *If $\delta > 0$ and $\delta = o(\varepsilon/T)$, then $\alpha = \Omega\Big(\min\Big\{\frac{T^{1/3}}{\varepsilon^{2/3}\log^{2/3}(\varepsilon T)}, \frac{\sqrt{d}}{\varepsilon \log d}, T\Big\}\Big)$.*
2. *If $\delta = 0$, then $\alpha = \Omega\Big(\min\Big\{\sqrt{\frac{T}{\varepsilon}}, \frac{d}{\varepsilon}, T\Big\}\Big)$.*

MaxSum_d can be released in the batch model with $\alpha = O(1/\varepsilon)$ via the Laplace mechanism (Dwork et al., 2006b). Hence, Theorem 3.1 shows a strong separation between the batch model of differential privacy and continual release.

3.1. 1-way Marginal Queries in Batch Model

To prove our lower bounds for MaxSum, we reduce from the problem of approximating 1-way marginals in the batch model. The function $\mathsf{Marginals}_d : \mathcal{X}^* \to [0,1]^d$ maps a dataset $\mathbf{y}$ of any size n to a vector $(q_1(\mathbf{y}), \ldots, q_d(\mathbf{y}))$, where q_j, called the j^{th} marginal, is defined as $q_j(\mathbf{y}) = \frac{1}{n}\sum_{i=1}^n \mathbf{y}_i[j]$. The error $\mathsf{ERR}_{\mathsf{Marginals}}$ is defined as in (1). Next, we define accuracy for batch algorithms.

Definition 3.2 (Accuracy of Batch Algorithms)**.** *Let $\gamma \in [0,1]$, $n, d \in \mathbb{N}$, and $\mathcal{X} = \{0,1\}^d$. Let $f : \mathcal{X}^n \to \mathbb{R}^d$ be a function on datasets. Batch algorithm $\mathcal{A}$ is (γ, n)*-accurate *for f if for all datasets $\mathbf{y} \in \mathcal{X}^n$,*

$$\Pr_{\text{coins of } \mathcal{A}} [\mathsf{ERR}_f(\mathbf{y}, \mathcal{A}(\mathbf{y})) \leq \gamma] \geq \frac{2}{3}.$$

We use the lower bounds from Bun et al. (2018); Hardt & Talwar (2010) for the problem of estimating $\mathsf{Marginals}_d$ in the batch model. They are stated in Items 1 and 2 of Lemma 3.3 for approximate differential privacy and pure differential privacy, respectively. Item 2 in Lemma 3.3 is a slight modification of the lower bound from Hardt & Talwar (2010) and follows from a simple packing argument.

Lemma 3.3. *For all $\varepsilon \in (0,1]$, $\delta \in [0,1]$, $\gamma \in (0,1)$, $d, n \in \mathbb{N}$, and algorithms $\mathcal{A}$ that are (ε,δ)-differentially private and (γ,n)-accurate for $\mathsf{Marginals}_d$, the following statements hold.*

1. Bun et al. (2018): *If $\delta > 0$ and $\delta = o(1/n)$, then $n = \Omega\left(\frac{\sqrt{d}}{\gamma\varepsilon\log d}\right)$.*
2. Hardt & Talwar (2010): *If $\delta = 0$, then $n = \Omega\left(\frac{d}{\gamma\varepsilon}\right)$.*

3.2. Proof Sketch of Theorem 3.1

We give a proof sketch of Theorem 3.1; formal details can be found in Appendix C. Let $\mathcal{M}$ be an (ε,δ)-DP and (α,T)-accurate mechanism for MaxSum_d in the continual release model with nonadaptively chosen inputs. We use $\mathcal{M}$ to construct an (ε,δ)-DP batch algorithm $\mathcal{A}$ that is $(\frac{\alpha}{n}, n)$-accurate for $\mathsf{Marginals}_d$. The main idea in the construction, (presented in Algorithm 2 in Appendix C) , is to force $\mathcal{M}$ to output an estimate of the sum for one attribute at a time by making the sum in that attribute the largest. First, $\mathcal{A}$ sends its own dataset $\mathbf{y}$ to $\mathcal{M}$. Then it sends n

additional records with 1 in the first attribute and 0 everywhere else. After this, the first attribute sum is the largest, and the answer produced by $\mathcal{M}$ at this point can be used to estimate the first marginal. Then $\mathcal{A}$ equalizes the number of extraneous 1's for each attribute by sending n additional records with 0 in the first attribute and 1 everywhere else. It repeats this for each attribute, collecting the answers from $\mathcal{M}$, and then outputs its estimates for the marginals.

This gives an accurate algorithm for the marginals problem, which is captured in the following lemma.

Lemma 3.4 (Informal)**.** *Let $\mathcal{A}$ be the algorithm informally described in the previous paragraph. For all $\varepsilon > 0, \delta \geq 0, \alpha \in \mathbb{R}^+$ and $d, n, T \in \mathbb{N}$, where $T \geq 2dn$, if mechanism $\mathcal{M}$ is (ε, δ)-DP and (α, T)-accurate for* MaxSum_d *in the continual release model with nonadaptively chosen inputs, then batch algorithm $\mathcal{A}$ is (ε, δ)-DP and $(\frac{\alpha}{n}, n)$-accurate for* $\mathsf{Marginals}_d$.

Observe that both lower bounds on α stated in Theorem 3.1 are the minimum of three terms. To prove them, it suffices to show that, for all ranges of parameters, one of the terms is a lower bound on α.

The rest of the proof of Theorem 3.1 follows by a case analysis: (1) For $\varepsilon \leq \frac{2}{T}$, we use a group privacy argument to show that $\alpha > T/9$ (for both pure and approximate differential privacy). (2) For $\varepsilon > \frac{2}{T}$, we use Lemma 3.4 and Lemma 3.3 to lower bound α. The details of the proof can be found in Appendix C.

4. Lower Bounds for SumSelect

In this section, we prove Theorem 4.1 that provides strong lower bounds on the accuracy parameter α of any (α, T)-accurate algorithm $\mathcal{M}$ for $\mathsf{SumSelect}_d$ in the continual release model with nonadaptively chosen inputs. Our lower bounds match the upper bounds from Section 5 for $\mathsf{SumSelect}_d$ in the continual release model with adaptively chosen inputs up to logarithmic factors in the time horizon T and the number of coordinates d. We give the formal details of the proof in Appendix D.

Theorem 4.1. *For all $\varepsilon \in (0, 1], \delta \in [0, 1), \alpha > 0$, sufficiently large d,$T \in \mathbb{N}$, and mechanisms $\mathcal{M}$ in the continual release model with nonadaptively chosen inputs that are (ε, δ)-DP and (α, T)-accurate for* $\mathsf{SumSelect}_d$*, the following statements hold.*

1. *If $0 < \delta = o(\varepsilon/T^2)$, then*
$$\alpha = \tilde{\Omega}\Big(\min\Big\{\frac{T^{1/3}\log^{2/3} d}{\varepsilon^{2/3}}, \frac{\sqrt{d}}{\varepsilon}, T\Big\}\Big).$$

2. *If $\delta = 0$, then*
$$\alpha = \Omega\Big(\min\Big\{\sqrt{\frac{T}{\varepsilon}\log\Big(2 + \frac{d}{\sqrt{\varepsilon T}}\Big)}, \frac{d}{\varepsilon}, T\Big\}\Big)$$
$$= \tilde{\Omega}\Big(\min\Big\{\sqrt{\frac{T\log(d)}{\varepsilon}}, \frac{d}{\varepsilon}, T\Big\}\Big).$$

4.1. Proof sketch of Theorem 4.1

To prove our lower bounds for SumSelect in the continual release model with nonadaptively chosen inputs, we first reduce from a problem called k-IndSelect in the batch model. In the batch model, k-$\mathsf{IndSelect}_d$ solves the problem of selecting the index of the largest coordinate sum in each of k disjoint subsets (which have d coordinates each). We reduce the problem of solving k-$\mathsf{IndSelect}_d$ in the batch model to solving $\mathsf{SumSelect}_{dk}$ in the continual release model.

We then prove new lower bounds for k-IndSelect in the batch model: (1) In the case of approximate differential privacy ((ε, δ)-DP), we obtain a lower bound for k-IndSelect by reducing to a related problem and invoking a result of Steinke & Ullman (2017). (2) In the case of pure differential privacy ($(\varepsilon, 0)$-DP), we prove a lower bound by using a standard packing argument.

Finally, we use the new lower bounds for k-IndSelect in conjunction with our reduction and careful case analysis to obtain our lower bounds in the continual release model.

5. Continual Release with Adaptively Chosen Inputs

In this section, we provide an explicit game-based formulation of the *continual release model with adaptively chosen inputs*. Here, the inputs to the mechanism can be chosen online by an adversary that observes the outputs of the mechanism on prior stream elements. Unlike the setting considered in the prior sections, the input data at time t in this setting can depend on the values returned by the algorithm at prior steps. Because of this, both privacy and accuracy are harder to attain in this model in general.

Later in the section we describe differentially private mechanisms for two types of problems: $\mathsf{SumSelect}_d$ and approximating functions with bounded sensitivity (ℓ_2 sensitivity in the case of approximate differential privacy and ℓ_1 sensitivity in the case of pure DP).[3] For these problems we show that there is no overhead in terms of accuracy when the input stream is selected adaptively, that is, our lower bounds (that hold even when the input stream is selected obliviously) are matched by algorithms that work even with adaptively selected streams. We achieve this by analyzing variants of classical continual release algorithms in the new model. Since a proof of privacy in the model with adaptively chosen inputs cannot be generally reduced to a

[3] The latter class of problems captures MaxSum, which has sensitivity 1.

proof in the nonadaptive model [4], our analysis adapts techniques from simulation-based cryptography to argue indistinguishability of the relevant distributions directly.The main challenge in analyzing privacy is that input streams in 'neighboring' interactions with the private mechanism may differ in many records, which necessitates using different techniques than privacy proofs in prior work, which analyze privacy with respect to input streams that differ in a single record.

Our mechanisms are (α, T)-accurate, where the upper bounds for α match the lower bounds obtained in previous sections in the continual release model with nonadaptively chosen inputs up to logarithmic factors in the time horizon T, the number of coordinates d, and the inverse of the privacy parameter $\frac{1}{\delta}$.

5.1. Model Definition

In the continual release model with adaptively chosen inputs, a mechanism $\mathcal{M}$ interacts with a randomized adversarial process $\mathcal{A}dv$ that runs for T timesteps; at timestep $t \in [T]$, the process $\mathcal{A}dv$ receives a_t from $\mathcal{M}$, updates its internal state, and produces input record x_{t+1} that is sent to $\mathcal{M}$ at timestep $t+1$. The adversarial process $\mathcal{A}dv$ can choose x_{t+1} based on the previous input records $\mathbf{x}_{[t]}$ and $\mathcal{M}$'s previous outputs $\mathbf{a}_{[t]}$. We make no assumptions on $\mathcal{A}dv$ regarding running time or complexity; its only limitation is that it does not see the internal coins of $\mathcal{M}$.

Definition 5.1. *A mechanism $\mathcal{M}$ is* (α,T)-accurate *for a function f in the continual release model with adaptively chosen inputs if for all processes $\mathcal{A}dv$, the error of $\mathcal{M}$ with respect to $\mathcal{A}dv$ is at most α with high probability, that is,*

$$\Pr_{\textit{coins of } \mathcal{M}, \mathcal{A}dv} \left[\max_{t \in [T]} \mathsf{ERR}_f(a_t; \mathbf{x}_{[t]}) \leq \alpha \right] \geq \frac{2}{3}.$$

A similar notion of accuracy was considered in work on adversarial streaming (Ben-Eliezer et al., 2020; Hassidim et al., 2022; Kaplan et al., 2021), though those articles do not directly address privacy.

Next, we define *(event-level) privacy in the continual release model with adaptively chosen inputs* which is trickier than in the continual release model with nonadaptively chosen inputs. One difficulty here is that the definition of 'neighboring' input streams must still allow for adaptive online generation of the input streams by an adversarial process $\mathcal{A}dv$. This concept is implicit in the work of Thakurta & Smith (2013), but to our knowledge has not been previously defined. Privacy is defined with respect to the game $\Pi_{\mathcal{M},\mathcal{A}dv}$, described in Algorithm 1, between mechanism $\mathcal{M}$ and an adversary $\mathcal{A}dv$. In all timesteps except one, $\mathcal{A}dv$ outputs a single input record which $\Pi_{\mathcal{M},\mathcal{A}dv}$ simply forwards to $\mathcal{M}$. However, there is a special *challenge timestep* $t^* \in [T]$, selected by $\mathcal{A}dv$, in which $\mathcal{A}dv$ provides two records $x_{t^*}^{(L)}$ and $x_{t^*}^{(R)}$. The game comes in two versions, specified by its input parameter $\mathsf{side} \in \{L, R\}$ which is not known to $\mathcal{A}dv$ or $\mathcal{M}$: in one version, the record $x_{t^*}^{(L)}$ is handed to $\mathcal{M}$ at timestep t^*; in the other, the record $x_{t^*}^{(R)}$ is handed to $\mathcal{M}$ instead. The mechanism is private if the distributions on the adversary's view, which consists of its internal randomness and the transcript of messages it sends and receives, are close in the two versions of the game.

If we consider $x_{t^*}^{(L)}$ to be the data of person t^*, and $x_{t^*}^{(R)}$ to be a dummy value (say, all 0's) then: (1) The parameter side then controls whether the data of person t^* is included in the computation or not. (2) The privacy requirement is that an outside attacker cannot tell whether t^*'s data was used, even if the attacker has full knowledge of the process generating the data stream.

Algorithm 1 Privacy game $\Pi_{\mathcal{M},\mathcal{A}dv}$ for the continual release model with adaptively chosen inputs

Input: time horizon $T \in \mathbb{N}$, $\mathsf{side} \in \{L, R\}$ (not known to $\mathcal{A}dv$).
for $t = 1$ to T **do**
  $\mathcal{A}dv$ outputs $\mathsf{type}_t \in \{\mathsf{challenge}, \mathsf{regular}\}$, where $\mathsf{challenge}$ is chosen once during the game.
  if $\mathsf{type}_t = \mathsf{regular}$ **then**
    $\mathcal{A}dv$ outputs $x_t \in \mathcal{X}$ which is sent to $\mathcal{M}$.
  end if
  if $\mathsf{type}_t = \mathsf{challenge}$ **then**
    $t^* \leftarrow t$.
    $\mathcal{A}dv$ outputs $(x_t^{(L)}, x_t^{(R)}) \in \mathcal{X}^2$.
    $x_t^{(\mathsf{side})}$ is sent to $\mathcal{M}$.
  end if
  $\mathcal{M}$ outputs a_t which is given to $\mathcal{A}dv$.
end for

Definition 5.2. *The* view *of $\mathcal{A}dv$ in privacy game $\Pi_{\mathcal{M},\mathcal{A}dv}$ consists of $\mathcal{A}dv$'s internal randomness and the transcript of messages it sends and receives. Let $V_{\mathcal{M},\mathcal{A}dv}^{(\mathsf{side})}$ denote $\mathcal{A}dv$'s view at the end of the game run with input* $\mathsf{side} \in \{L, R\}$.

One could also define the adversary's view as its internal state at the end of the game. The version we define contains enough information to compute that internal state, but is simpler to work with.

In addition to (ε, δ)-DP, we consider a related notion, called zCDP (Bun & Steinke, 2016). See Section A.1 for background on zCDP and the notion of ρ-closeness of random variables ($\simeq_\rho$).

[4] Subsequently to the initial appearance of our work, Denisov et al. (2022) give an example of a mechanism that is private with nonadaptively chosen inputs but not with adaptively chosen inputs

Table 3. Our (asymptotic) upper bounds on the error of continual release mechanisms with adaptively chosen inputs.

Problem	ρ-zCDP		$(\varepsilon, 0)$-DP	
	Tree	Recomputation	Tree	Recomputation
MaxSum	$\frac{\sqrt{d}\log T\sqrt{\log(dT)}}{\sqrt{\rho}}$	$\sqrt[3]{\frac{T\log T}{\rho}}$	$\frac{d(\log d)\log^3 T}{\varepsilon}$	$\sqrt{\frac{T\log T}{\varepsilon}}$
SumSelect	$\frac{\sqrt{d}\log T\sqrt{\log(dT)}}{\sqrt{\rho}}$	$\frac{T^{1/3}\log^{2/3}(dT)}{\rho^{1/3}}$	$\frac{d(\log d)\log^3 T}{\varepsilon}$	$\sqrt{\frac{T\log(dT)}{\varepsilon}}$

Definition 5.3. *A mechanism $\mathcal{M}$ is* (ε, δ)-DP in the continual release model with adaptively chosen inputs *if, for all adversaries $\mathcal{A}dv$,*

$$V^{(L)}_{\mathcal{M},\mathcal{A}dv} \approx_{\varepsilon,\delta} V^{(R)}_{\mathcal{M},\mathcal{A}dv}.$$

A mechanism $\mathcal{M}$ is ρ-zCDP in the continual release model with adaptively chosen inputs *if for all adversaries $\mathcal{A}dv$,*

$$V^{(L)}_{\mathcal{M},\mathcal{A}dv} \simeq_{\rho} V^{(R)}_{\mathcal{M},\mathcal{A}dv}.$$

The symbol $\simeq_\rho$ denotes ρ-closeness (Definition A.11).

5.2. Summary of Upper Bounds for Adaptive Inputs

Our upper bounds on the error of differentially private mechanisms for MaxSum_d and $\mathsf{SumSelect}_d$ in the continual release model with adaptively chosen inputs are summarized in Table 3. The corresponding theorems are stated in Appendix E.1. The upper bounds in the table are attained by two simple mechanisms: one uses the binary tree mechanism and the other recomputes the target function at regular intervals. The bounds stated for MaxSum_d and obtained via recomputing periodically apply more generally: to all sensitivity-1 functions. Detailed proofs are given in Appendix E.

Acknowledgments

We are grateful to Kobbi Nissim for being part of the conversations that got this work started and for subsequent helpful comments. We are also grateful to Jon Ullman and Thomas Steinke for insights into lower bounds for the top-k selection problem. A.S. and P.J. were supported in part by NSF awards CCF-1763786 and CNS-2120667 as well as Faculty Awards from Google and Apple. S.S. was supported by NSF award CNS-2046425 and Cooperative Agreement CB20ADR0160001 with the Census Bureau.

References

Agarwal, N. and Singh, K. The price of differential privacy for online learning. In Precup, D. and Teh, Y. W. (eds.), *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pp. 32–40. PMLR, 06–11 Aug 2017.

Apple. Learning with privacy at scale, 2017.

Bafna, M. and Ullman, J. The price of selection in differential privacy. In Kale, S. and Shamir, O. (eds.), *Proceedings of the 2017 Conference on Learning Theory*, volume 65 of *Proceedings of Machine Learning Research*, pp. 151–168. PMLR, 07–10 Jul 2017.

Bassily, R., Smith, A. D., and Thakurta, A. Private empirical risk minimization: Efficient algorithms and tight error bounds. In *55th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2014, Philadelphia, PA, USA, October 18-21, 2014*, pp. 464–473. IEEE Computer Society, 2014. doi: 10.1109/FOCS.2014.56.

Beimel, A., Kaplan, H., Mansour, Y., Nissim, K., Saranurak, T., and Stemmer, U. Dynamic algorithms against an adaptive adversary: generic constructions and lower bounds. In Leonardi, S. and Gupta, A. (eds.), *STOC '22: 54th Annual ACM SIGACT Symposium on Theory of Computing, Rome, Italy, June 20 - 24, 2022*, pp. 1671–1684. ACM, 2022. doi: 10.1145/3519935.3520064.

Ben-Eliezer, O., Jayaram, R., Woodruff, D. P., and Yogev, E. A framework for adversarially robust streaming algorithms. In *Proceedings of the 39th ACM SIGMOD-SIGACT-SIGAI Symposium on Principles of Database Systems*, PODS'20, pp. 63–80, New York, NY, USA, 2020. Association for Computing Machinery. ISBN 9781450371087. doi: 10.1145/3375395.3387658.

Ben-Eliezer, O., Eden, T., and Onak, K. Adversarially robust streaming via dense-sparse trade-offs. In Bringmann, K. and Chan, T. (eds.), *5th Symposium on Simplicity in Algorithms, SOSA@SODA 2022, Virtual Conference, January 10-11, 2022*, pp. 214–227. SIAM, 2022a. doi: 10.1137/1.9781611977066.15.

Ben-Eliezer, O., Jayaram, R., Woodruff, D. P., and Yogev, E. A framework for adversarially robust streaming algorithms. *J. ACM*, 69(2):17:1–17:33, 2022b. doi: 10.1145/3498334.

Bolot, J., Fawaz, N., Muthukrishnan, S., Nikolov, A., and Taft, N. Private decayed predicate sums on streams. In *Proceedings of the 16th International Conference on Database Theory*, ICDT '13, pp. 284–295, New York, NY, USA, 2013. Association for Computing Machinery. ISBN 9781450315982. doi: 10.1145/2448496.2448530.

Bun, M. and Steinke, T. Concentrated differential privacy: Simplifications, extensions, and lower bounds. In Hirt, M. and Smith, A. D. (eds.), *Theory of Cryptography - 14th International Conference, TCC 2016-B, Beijing, China, October 31 - November 3, 2016, Proceedings, Part I*, volume 9985 of *Lecture Notes in Computer Science*, pp. 635–658, 2016. doi: 10.1007/978-3-662-53641-4_24.

Bun, M., Ullman, J., and Vadhan, S. Fingerprinting codes and the price of approximate differential privacy. *SIAM Journal on Computing*, 47(5):1888–1938, 2018.

Cardoso, A. R. and Rogers, R. Differentially private histograms under continual observation: Streaming selection into the unknown. In Camps-Valls, G., Ruiz, F. J. R., and Valera, I. (eds.), *International Conference on Artificial Intelligence and Statistics, AISTATS 2022, 28-30 March 2022, Virtual Event*, volume 151 of *Proceedings of Machine Learning Research*, pp. 2397–2419. PMLR, 2022.

Chan, T. H., Shi, E., and Song, D. Private and continual release of statistics. *IACR Cryptol. ePrint Arch.*, 2010: 76, 2010.

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

Cheu, A. and Ullman, J. R. The limits of pan privacy and shuffle privacy for learning and estimation. In Khuller, S. and Williams, V. V. (eds.), *STOC '21: 53rd Annual ACM SIGACT Symposium on Theory of Computing, Virtual Event, Italy, June 21-25, 2021*, pp. 1081–1094. ACM, 2021. doi: 10.1145/3406325.3450995.

Cohen, E., Lyu, X., Nelson, J., Sarlós, T., Shechner, M., and Stemmer, U. On the robustness of countsketch to adaptive inputs. In Chaudhuri, K., Jegelka, S., Song, L., Szepesvári, C., Niu, G., and Sabato, S. (eds.), *International Conference on Machine Learning, ICML 2022, 17-23 July 2022, Baltimore, Maryland, USA*, volume 162 of *Proceedings of Machine Learning Research*, pp. 4112–4140. PMLR, 2022.

Denisov, S., McMahan, H. B., Rush, J., Smith, A. D., and Thakurta, A. G. Improved differential privacy for SGD via optimal private linear operators on adaptive streams. In *NeurIPS*, 2022.

Duchi, J., Jordan, M., and Wainwright, M. Local privacy and statistical minimax rates. In *IEEE Symposium on Foundations of Computer Science*, FOCS '13, pp. 429–438, Berkeley, CA, USA, 2013.

Durfee, D. and Rogers, R. M. Practical differentially private top-k selection with pay-what-you-get composition. In Wallach, H. M., Larochelle, H., Beygelzimer, A., d'Alché-Buc, F., Fox, E. B., and Garnett, R. (eds.), *Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada*, pp. 3527–3537, 2019.

Dwork, C., Kenthapadi, K., McSherry, F., Mironov, I., and Naor, M. Our data, ourselves: Privacy via distributed noise generation. In *International Conference on the Theory and Applications of Cryptographic Techniques*, EUROCRYPT '06, pp. 486–503, St. Petersburg, Russia, 2006a.

Dwork, C., McSherry, F., Nissim, K., and Smith, A. Calibrating noise to sensitivity in private data analysis. In *Theory of cryptography conference*, pp. 265–284. Springer, 2006b.

Dwork, C., Naor, M., Pitassi, T., and Rothblum, G. N. Differential privacy under continual observation. In Schulman, L. J. (ed.), *Proceedings of the 42nd ACM Symposium on Theory of Computing, STOC 2010, Cambridge, Massachusetts, USA, 5-8 June 2010*, pp. 715–724. ACM, 2010a. doi: 10.1145/1806689.1806787.

Dwork, C., Naor, M., Pitassi, T., Rothblum, G. N., and Yekhanin, S. Pan-private streaming algorithms. In Yao, A. C. (ed.), *Innovations in Computer Science - ICS 2010, Tsinghua University, Beijing, China, January 5-7, 2010. Proceedings*, pp. 66–80. Tsinghua University Press, 2010b.

Dwork, C., Rothblum, G. N., and Vadhan, S. P. Boosting and differential privacy. In *51th Annual IEEE Symposium on Foundations of Computer Science, FOCS 2010, October 23-26, 2010, Las Vegas, Nevada, USA*, pp. 51–60. IEEE Computer Society, 2010c. doi: 10.1109/FOCS.2010.12.

Dwork, C., Naor, M., Reingold, O., and Rothblum, G. N. Pure differential privacy for rectangle queries via private partitions. In Iwata, T. and Cheon, J. H. (eds.), *Advances in Cryptology - ASIACRYPT 2015 - 21st International Conference on the Theory and Application of Cryptology and Information Security, Auckland, New Zealand, November 29 - December 3, 2015, Proceedings, Part II*, volume 9453 of *Lecture Notes in Computer Science*, pp. 735–751. Springer, 2015. doi: 10.1007/978-3-662-48800-3_30.

Edmonds, A., Nikolov, A., and Ullman, J. R. The power of factorization mechanisms in local and central differential privacy. In Makarychev, K., Makarychev, Y., Tulsiani, M., Kamath, G., and Chuzhoy, J. (eds.), *Proccedings of* *the 52nd Annual ACM SIGACT Symposium on Theory of Computing, STOC 2020, Chicago, IL, USA, June 22-26, 2020*, pp. 425–438. ACM, 2020. doi: 10.1145/3357713.3384297.

Fichtenberger, H., Henzinger, M., and Ost, W. Differentially private algorithms for graphs under continual observation. In Mutzel, P., Pagh, R., and Herman, G. (eds.), *29th Annual European Symposium on Algorithms, ESA 2021, September 6-8, 2021, Lisbon, Portugal (Virtual Conference)*, volume 204 of *LIPIcs*, pp. 42:1–42:16. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2021. doi: 10.4230/LIPIcs.ESA.2021.42.

Ghazi, B., Kumar, R., Nelson, J., and Manurangsi, P. Private counting of distinct and k-occurring items in time windows. In Kalai, Y. T. (ed.), *14th Innovations in Theoretical Computer Science Conference, ITCS 2023, January 10-13, 2023, MIT, Cambridge, Massachusetts, USA*, volume 251 of *LIPIcs*, pp. 55:1–55:24. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2023. doi: 10.4230/LIPIcs.ITCS.2023.55.

Hardt, M. and Talwar, K. On the geometry of differential privacy. In *Proceedings of the 42nd Annual ACM Symposium on the Theory of Computing*, STOC '10, pp. 705–714, New York, NY, USA, 2010. ACM.

Hardt, M., Ligett, K., and McSherry, F. A simple and practical algorithm for differentially private data release. In Bartlett, P. L., Pereira, F. C. N., Burges, C. J. C., Bottou, L., and Weinberger, K. Q. (eds.), *Advances in Neural Information Processing Systems 25*, pp. 2348–2356, 2012.

Hassidim, A., Kaplan, H., Mansour, Y., Matias, Y., and Stemmer, U. Adversarially robust streaming algorithms via differential privacy. *J. ACM*, 69(6):42:1–42:14, 2022. doi: 10.1145/3556972.

Hay, M., Rastogi, V., Miklau, G., and Suciu, D. Boosting the accuracy of differentially private histograms through consistency. *Proc. VLDB Endow.*, 3(1):1021–1032, 2010. doi: 10.14778/1920841.1920970.

Hay, M., Machanavajjhala, A., Miklau, G., Chen, Y., and Zhang, D. Principled evaluation of differentially private algorithms using DPBench. In Özcan, F., Koutrika, G., and Madden, S. (eds.), *Proceedings of the 2016 International Conference on Management of Data, SIGMOD Conference 2016, San Francisco, CA, USA, June 26 - July 01, 2016*, pp. 139–154. ACM, 2016. doi: 10.1145/2882903.2882931.

Jain, P., Kothari, P., and Thakurta, A. Differentially private online learning. In Mannor, S., Srebro, N., and Williamson, R. C. (eds.), *Proceedings of the 25th Annual Conference on Learning Theory*, volume 23 of *Proceedings of Machine Learning Research*, pp. 24.1–24.34, Edinburgh, Scotland, 25–27 Jun 2012. JMLR Workshop and Conference Proceedings.

Kairouz, P., Mcmahan, B., Song, S., Thakkar, O., Thakurta, A., and Xu, Z. Practical and private (deep) learning without sampling or shuffling. In Meila, M. and Zhang, T. (eds.), *Proceedings of the 38th International Conference on Machine Learning*, volume 139 of *Proceedings of Machine Learning Research*, pp. 5213–5225. PMLR, 18–24 Jul 2021.

Kaplan, H., Mansour, Y., Nissim, K., and Stemmer, U. Separating adaptive streaming from oblivious streaming using the bounded storage model. In Malkin, T. and Peikert, C. (eds.), *Advances in Cryptology - CRYPTO 2021 - 41st Annual International Cryptology Conference, CRYPTO 2021, Virtual Event, August 16-20, 2021, Proceedings, Part III*, volume 12827 of *Lecture Notes in Computer Science*, pp. 94–121. Springer, 2021.

Kasiviswanathan, S. P. and Smith, A. D. On the 'semantics' of differential privacy: A Bayesian formulation. *J. Priv. Confidentiality*, 6(1), 2014. doi: 10.29012/jpc.v6i1.634.

Kasiviswanathan, S. P., Lee, H. K., Nissim, K., Raskhodnikova, S., and Smith, A. D. What can we learn privately? *SIAM J. Comput.*, 40(3):793–826, 2011. doi: 10.1137/090756090.

McKenna, R. and Sheldon, D. R. Permute-and-Flip: a new mechanism for differentially private selection. In Larochelle, H., Ranzato, M., Hadsell, R., Balcan, M. F., and Lin, H. (eds.), *Advances in Neural Information Processing Systems*, volume 33, pp. 193–203. Curran Associates, Inc., 2020.

McSherry, F. and Talwar, K. Mechanism design via differential privacy. In *Proceedings of the 48th Annual IEEE Symposium on Foundations of Computer Science*, FOCS '07, pp. 94–103, USA, 2007. IEEE Computer Society. ISBN 0769530109. doi: 10.1109/FOCS.2007.41.

Mironov, I., Naor, M., and Segev, G. Sketching in adversarial environments. *SIAM J. Comput.*, 40(6):1845–1870, 2011. doi: 10.1137/080733772.

Perrier, V., Asghar, H. J., and Kaafar, D. Private continual release of real-valued data streams. In *26th Annual Network and Distributed System Security Symposium, NDSS 2019, San Diego, California, USA, February 24-27, 2019*. The Internet Society, 2019.

Qiao, G., Su, W., and Zhang, L. Oneshot differentially private top-k selection. In Meila, M. and Zhang, T. (eds.), *Proceedings of the 38th International Conference on*

Machine Learning, volume 139 of *Proceedings of Machine Learning Research*, pp. 8672–8681. PMLR, 18–24 Jul 2021.

Rényi, A. On measures of entropy and information. *Proceedings of the Fourth Berkeley Symposium on Mathematical Statistics and Probability, Volume 1: Contributions to the Theory of Statistics, pages 547–561, Berkeley, Calif., 1961. University of California Press*, abs/2101.10836, 1961.

Song, S., Little, S., Mehta, S., Vinterbo, S. A., and Chaudhuri, K. Differentially private continual release of graph statistics. *CoRR*, abs/1809.02575, 2018.

Steinke, T. and Ullman, J. R. Tight lower bounds for differentially private selection. In Umans, C. (ed.), *58th IEEE Annual Symposium on Foundations of Computer Science, FOCS 2017, Berkeley, CA, USA, October 15-17, 2017*, pp. 552–563. IEEE Computer Society, 2017. doi: 10.1109/FOCS.2017.57.

Talwar, K., Thakurta, A., and Zhang, L. Nearly optimal private LASSO. In Cortes, C., Lawrence, N. D., Lee, D. D., Sugiyama, M., and Garnett, R. (eds.), *Advances in Neural Information Processing Systems 28: Annual Conference on Neural Information Processing Systems 2015, December 7-12, 2015, Montreal, Quebec, Canada*, pp. 3025–3033, 2015.

Thakurta, A. G. and Smith, A. (Nearly) optimal algorithms for private online learning in full-information and bandit settings. In Burges, C. J. C., Bottou, L., Welling, M., Ghahramani, Z., and Weinberger, K. Q. (eds.), *Advances in Neural Information Processing Systems*, volume 26. Curran Associates, Inc., 2013.

Ullman, J., 2021. Personal communication.

Xiao, X., Wang, G., and Gehrke, J. Differential privacy via wavelet transforms. *IEEE Trans. Knowl. Data Eng.*, 23 (8):1200–1214, 2011. doi: 10.1109/TKDE.2010.247.

Yang, T., Andrew, G., Eichner, H., Sun, H., Li, W., Kong, N., Ramage, D., and Beaufays, F. Applied federated learning: Improving google keyboard query suggestions. *CoRR*, abs/1812.02903, 2018.

Yu, D., Naik, S., Backurs, A., Gopi, S., Inan, H. A., Kamath, G., Kulkarni, J., Lee, Y. T., Manoel, A., Wutschitz, L., Yekhanin, S., and Zhang, H. Differentially private fine-tuning of language models. In *The Tenth International Conference on Learning Representations, ICLR 2022, Virtual Event, April 25-29, 2022*. OpenReview.net, 2022.

A. Differential Privacy

We first introduce the notion of (ε, δ)-indistinguishability.

Definition A.1 ((ε, δ)-Indistinguishability)**.** *Random variables R_1 and R_2 over the same outcome space $\mathcal{Y}$ are (ε, δ)-*indistinguishable *(denoted $R_1 \approx_{\varepsilon,\delta} R_2$) if for all subsets $S \subseteq \mathcal{Y}$, the following hold:*

$$\Pr[R_1 \in S] \leq e^{\varepsilon} \Pr[R_2 \in S] + \delta;$$
$$\Pr[R_2 \in S] \leq e^{\varepsilon} \Pr[R_1 \in S] + \delta.$$

A dataset $\mathbf{x} = (x_1, \dots, x_n) \in \mathcal{X}^n$ is a vector of elements, called *records*, from a universe $\mathcal{X}$. Two datasets are *neighbors* if they differ in one record (i.e., one coordinate). Informally, differential privacy requires that an algorithm's output distributions are similar on all pairs of neighboring datasets. In the batch model, the algorithm receives datasets as one batch as opposed to in an online fashion.

Definition A.2 (Differential Privacy in Batch Model (Dwork et al., 2006b;a))**.** *A randomized algorithm $\mathcal{A} : \mathcal{X}^n \to \mathcal{Y}$ is (ε, δ)-*differentially private (DP) *if for every pair of neighboring datasets $\mathbf{x}, \mathbf{x}' \in \mathcal{X}^n$,*

$$\mathcal{A}(\mathbf{x}) \approx_{\varepsilon,\delta} \mathcal{A}(\mathbf{x}').$$

The case $\delta = 0$ is referred to as pure *differential privacy, whereas the case $\delta > 0$ is called* approximate *differential privacy.*

Differential privacy protects groups of individuals.

Lemma A.3 (Group Privacy (Dwork et al., 2006b))**.** *Every (ε, δ)-DP algorithm $\mathcal{A}$ is $(\ell\varepsilon, \delta')$-DP for groups of size ℓ, where $\delta' = \delta \frac{e^{\ell\varepsilon}-1}{e^{\varepsilon}-1}$; that is, for all datasets $\mathbf{x}, \mathbf{x}'$ such that $\|\mathbf{x} - \mathbf{x}'\|_0 \leq \ell$,*

$$\mathcal{A}(\mathbf{x}) \approx_{\ell\varepsilon,\delta'} \mathcal{A}(\mathbf{x}').$$

Differential privacy is closed under post-processing.

Lemma A.4 (Post-Processing (Dwork et al., 2006b; Bun & Steinke, 2016))**.** *If $\mathcal{A}$ is an (ε, δ)-DP algorithm with output space $\mathcal{Y}$ and $\mathcal{B}$ is a randomized map from $\mathcal{Y}$ to $\mathcal{Z}$, then the algorithm $\mathcal{B} \circ \mathcal{A}$ is (ε, δ)-DP.*

Definition A.5 (Sensitivity)**.** *Let $f : \mathcal{X}^n \to \mathbb{R}^m$ be a function. Its ℓ_1-sensitivity is*

$$\max_{\text{neighbors } \mathbf{x},\mathbf{x}' \in \mathcal{X}^n} \|f(\mathbf{x}) - f(\mathbf{x}')\|_1.$$

To define ℓ_2-sensitivity, we replace the ℓ_1 norm with the ℓ_2 norm.

Our algorithms use the standard Laplace and Exponential mechanisms to ensure differential privacy.

Definition A.6 (Laplace Distribution)**.** *The Laplace distribution with parameter b and mean 0, denoted $\mathsf{Lap}(b)$, has probability density*

$$h(r) = \frac{1}{2b} e^{-\frac{|r|}{b}} \text{ for all } r \in \mathbb{R}.$$

Lemma A.7 (Laplace Mechanism, (Dwork et al., 2006b))**.** *Let $f : \mathcal{X}^n \to \mathbb{R}^m$ be a function with ℓ_1-sensitivity at most Δ_1. Then the Laplace mechanism is algorithm*

$$\mathcal{A}_f(\mathbf{x}) = f(\mathbf{x}) + (Z_1, \dots, Z_m),$$

where $Z_i \sim \mathsf{Lap}\left(\frac{\Delta_1}{\varepsilon}\right)$. Algorithm $\mathcal{A}_f$ is $(\varepsilon, 0)$-DP.

Lemma A.8 (Exponential Mechanism (McSherry & Talwar, 2007))**.** *Let L be a set of outputs and $g : L \times \mathcal{X}^n \to \mathbb{R}$ be a function that measures the quality of each output on a dataset. Assume that for every $m \in L$, the function $g(m, .)$ has ℓ_1-sensitivity at most Δ. Then, for all $\varepsilon, n > 0$ and for all datasets $y \in \mathcal{X}^n$, there exists an $(\varepsilon, 0)$-DP mechanism that outputs an element $m \in L$ such that, for all $a > 0$, we have*

$$\Pr\left[\max_{i \in [L]} g(i, y) - g(m, y) \geq 2\Delta \frac{(\ln |L| + a)}{\varepsilon}\right] \leq e^{-a}.$$

Definition A.9 (Gaussian Distribution)**.** *The Gaussian distribution with parameter σ and mean 0, denoted $\mathcal{N}(0, \sigma^2)$, has probability density*

$$h(r) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{r^2}{2\sigma^2}} \text{ for all } r \in \mathbb{R}.$$

A.1. ρ-zCDP

This section describes "zero-concentrated differential privacy" (zCDP), a variant of differential privacy that is less stringent than pure differential privacy, but more stringent than approximate differential privacy. In contrast to (ε, δ)-differential privacy, zCDP requires output distributions on all pairs of neighboring datasets to be ρ-close (Definition A.11) instead of (ε, δ)-indistinguishable. In Appendix E , we show that our algorithms are zCDP and then use conversion from zCDP to (ε, δ)-differential privacy (Lemma A.15) to restate our upper bounds in the same terms as our lower bounds for easy comparison between the two.

Definition A.10 (Rényi Divergence (Rényi, 1961))**.** *Let Q and Q' be distributions on $\mathcal{Y}$. For $\xi \in (1, \infty)$, the Rényi divergence of order ξ between Q and Q'(also called the ξ-Rényi Divergence) is defined as*

$$D_{\xi}(Q \| Q') = \frac{1}{\xi - 1} \log \left(\mathop{\mathbb{E}}_{r \sim Q'} \left[\left(\frac{Q(r)}{Q'(r)} \right)^{\xi - 1} \right] \right). \quad (3)$$

Here $Q(\cdot)$ and $Q'(\cdot)$ denote either probability masses (in the discrete case) or probability densities (when they exist). More generally, one can replace $\frac{Q(.)}{Q'(.)}$ with the the Radon-Nikodym derivative of Q with respect to Q'.

Definition A.11 (ρ-Closeness)**.** *Random variables R_1 and R_2 over the same outcome space $\mathcal{Y}$ are* ρ-close *(denoted $R_1 \simeq_\rho R_2$) if for all $\xi \in (1, \infty)$,*

$$D_\xi(R_1 \| R_2) \le \xi\rho \text{ and } D_\xi(R_2 \| R_1) \le \xi\rho,$$

where $D_\xi(R_1\|R_2)$ is the ξ-Rényi divergence between the distributions of R_1 and R_2.

Definition A.12 (zCDP in Batch Model (Bun & Steinke, 2016))**.** *A randomized batch algorithm $\mathcal{A} : \mathcal{X}^n \to \mathcal{Y}$ is ρ-zero-concentrated differentially private (ρ-zCDP), if, for all neighboring datasets $\mathbf{y}, \mathbf{y}' \in \mathcal{X}^n$,*

$$\mathcal{A}(\mathbf{y}) \simeq_\rho \mathcal{A}(\mathbf{y}').$$

One major benefit of using zCDP is that this definition of privacy admits a clean composition result. We use it when analysing the privacy of the algorithms in Appendix E .

Lemma A.13 (Composition (Bun & Steinke, 2016))**.** *Let $\mathcal{A} : \mathcal{X}^n \to \mathcal{Y}$ and $\mathcal{A}' : \mathcal{X}^n \times \mathcal{Y} \to \mathcal{Z}$ be batch algorithms. Suppose $\mathcal{A}$ is ρ-zCDP and $\mathcal{A}'$ is ρ'-zCDP. Define batch algorithm $\mathcal{A}'' : \mathcal{X}^n \to \mathcal{Y} \times \mathcal{Z}$ by $\mathcal{A}''(\mathbf{y}) = \mathcal{A}'(\mathbf{y}, \mathcal{A}(\mathbf{y}))$. Then $\mathcal{A}''$ is $(\rho + \rho')$-zCDP.*

The *Gaussian mechanism*, defined next, is used in Section 5. It privately estimates a real-valued function on a database by adding Gaussian noise to the value of the function.

Lemma A.14 (Gaussian Mechanism (Bun & Steinke, 2016))**.** *Let $f : \mathcal{X}^n \to \mathbb{R}$ be a function with ℓ_2-sensitivity at most Δ_2. Let $\mathcal{A}$ be the batch algorithm that, on input $\mathbf{y}$, releases a sample from $\mathcal{N}(f(\mathbf{y}), \sigma^2)$. Then $\mathcal{A}$ is $(\Delta_2^2/2\sigma^2)$-zCDP.*

The final lemma in this section relates zero-concentrated differential privacy to (ε, δ)-differential privacy.

Lemma A.15 (Conversion from zCDP to DP (Bun & Steinke, 2016))**.** *For all $\rho, \delta > 0$, if batch algorithm $\mathcal{A}$ is ρ-zCDP, then $\mathcal{A}$ is $(\rho + 2\sqrt{\rho \log(1/\delta)}, \delta)$-DP.*

B. Further Related Work

Event-level privacy For streams with a limited number of ones, Dwork et al. (2015) give a polynomial improvement on the upper bound for summation from Dwork et al. (2010a) and Chan et al. (2010). Bolot et al. (2013) and Perrier et al. (2019) extended the tree mechanism of Dwork et al. (2010a) to work for weighted sums with exponentially decaying coefficients and for sums of bounded real values, respectively. A line of work has studied applications of the tree mechanism to answering range queries (see, e.g., Hay et al. (2010); Xiao et al. (2011); Hay et al. (2016); Dwork et al. (2010a; 2015); Edmonds et al. (2020)). Song et al. (2018) generalized the continual release model to graph data and obtained a mechanism that released graph statistics, such as the degree distribution and subgraph counts, on bounded degree graphs. Fichtenberger et al. (2021) studied a variety of other graph problems in the continual release setting, including minimum cut and densest subgraph. DP online learning is investigated in a sequence of works (Jain et al., 2012; Thakurta & Smith, 2013; Agarwal & Singh, 2017) that use the summation primitive developed by Dwork et al. (2010a) to obtain sublinear regret guarantees for many hypothesis classes. The adaptive continual release model arises implicitly in those works, but to our knowledge, it was not formulated explicitly. Cardoso & Rogers (2022) study, among other problems, SumSelect (called *top-1 selection with unrestricted ℓ_0 sensitivity* in their work) in the continual release model. Their focus is on empirical performance on streams that arise in practice, in which the index of the largest sum changes seldom. The recomputation-based algorithm we present for SumSelect can be seen as a special case of their KnownBase algorithm. They evaluate the accuracy of the algorithm empirically whereas our work provides theoretical bounds on the error. One of the contributions of Cardoso & Rogers (2022) is making the algorithms work in a more restrictive computational model, in which the algorithm stores only the current values of the sums at any given time step and the seed of a pseudorandom function. The algorithms we present here can also be implemented in their model using the techniques in their paper.

User-level differential privacy User-level privacy in the *continual release model* was first studied by Dwork et al. (2010a) and Chan et al. (2011). *User-level* privacy is more stringent than *event-level* privacy, so the lower bounds in our paper apply directly to that model. Even though, in general, event-level privacy does not imply user-level privacy, the recomputation technique used in some of our algorithms gives *user-level* privacy whenever the mechanism employed for the recomputations is user-level private.

Pan-Privacy Pan-privacy, defined by Dwork et al. (2010b), is a model that protects against intrusions into the memory of the algorithm as it processes a stream. In pan-privacy, as in continual release, the input is presented as a stream. However, the requirement of pan-privacy is orthogonal to that of continual release; see Dwork et al. (2010b) for details.

C. Proofs Omitted from Section 3

In this section, we prove Theorem 3.1 by formalizing the proof sketch from Section 3.2.

For vectors $\mathbf{u} = (u_1, \ldots, u_\ell)$ and $\mathbf{v} = (v_1, \ldots, v_m)$, let $\mathbf{u} \circ \mathbf{v} = (u_1, \ldots, u_\ell, v_1, \ldots, v_m)$. For a vector $\mathbf{v}$, let $\mathbf{v}^n$

Algorithm 2 Algorithm $\mathcal{A}$ for estimating all 1-way marginals

1: **Input:** $\mathbf{y} = (y_1, \ldots, y_n) \in \mathcal{X}^n$, where $\mathcal{X} = \{0,1\}^d$, and black-box access to mechanism $\mathcal{M}$.
2: **Output:** $\mathbf{b} = (b_1, \ldots, b_d) \in \mathbb{R}^d$.
3: Let $\mathbf{e}_j$ be a vector of length d with 1 in coordinate j and 0 everywhere else; let $\overline{\mathbf{e}_j} \leftarrow (1)^d - \mathbf{e}_j$.
4: Construct a stream $\mathbf{x} \leftarrow \mathbf{y} \circ (\mathbf{e}_1)^n \circ (\overline{\mathbf{e}_1})^n \circ \cdots \circ (\mathbf{e}_{d-1})^n \circ (\overline{\mathbf{e}_{d-1}})^n \circ (\mathbf{e}_d)^n$ with $2dn$ records.
5: **for** $t \in [T]$ **do**
6: Send x_t to $\mathcal{M}$ and get the corresponding output a_t.
7: **end for**
8: **for** $j \in [d]$ **do**
9: $b_j \leftarrow a_{2jn}/n - j$.
10: **end for**
11: Output $\mathbf{b} \leftarrow (b_1, \ldots, b_d)$.

denote the vector $\mathbf{v} \circ \mathbf{v} \circ \cdots \circ \mathbf{v}$ representing n concatenated copies of $\mathbf{v}$. Algorithm 2 represents the algorithm reducing the one-way marginals problem to solving MaxSum in the continual release model. We prove the following lemma capturing the accuracy guarantees of the reduction.

Lemma C.1. *Let $\mathcal{A}$ be Algorithm 2. For all $\varepsilon > 0, \delta \geq 0, \alpha \in \mathbb{R}^+$ and $d, n, T \in \mathbb{N}$, where $T \geq 2dn$, if mechanism $\mathcal{M}$ is (ε, δ)-DP and (α, T)-accurate for* MaxSum_d *in the continual release model with nonadaptively chosen inputs, then batch algorithm $\mathcal{A}$ is (ε, δ)-DP and $(\frac{\alpha}{n}, n)$-accurate for* $\mathsf{Marginals}_d$.

Proof of Lemma C.1. We start by reasoning about privacy. Fix neighboring datasets $\mathbf{y}$ and $\mathbf{y}'$ that are inputs to algorithm $\mathcal{A}$. Let $\mathbf{x}$ and $\mathbf{x}'$ be the streams constructed in Step 4 of $\mathcal{A}$ when it is run on $\mathbf{y}$ and $\mathbf{y}'$, respectively. By construction, $\mathbf{x}$ and $\mathbf{x}'$ are neighbors. Since $\mathcal{M}$ is (ε, δ)-DP, and $\mathcal{A}$ only post-processes the outputs received from $\mathcal{M}$, Lemma A.4 implies that $\mathcal{A}$ is (ε, δ)-DP.

Now we reason about accuracy. Let $\mathbf{x} = (x_1, \ldots, x_{2dn})$ be the input stream provided to $\mathcal{M}$ when $\mathcal{A}$ is run on dataset $\mathbf{y}$. By construction of $\mathbf{x}$, the marginals $q_j(\mathbf{y})$ for all $j \in [d]$ and MaxSum_d are related as follows:

$$q_j(\mathbf{y}) = \frac{1}{n} \sum_{i \in [n]} y_i[j] = \frac{1}{n} \Big(\sum_{i \in [2jn]} x_i[j] - jn \Big)$$
$$= \frac{1}{n} \cdot \mathsf{MaxSum}_d(\mathbf{x}_{[2jn]}) - j. \tag{4}$$

The attribute with the largest sum in $\mathbf{x}_{[2jn]}$ is j because $(\mathbf{e}_1)^n \circ (\overline{\mathbf{e}_1})^n \circ \cdots \circ (\mathbf{e}_{j-1})^n \circ (\overline{\mathbf{e}_{j-1}})^n \circ (\mathbf{e}_j)^n$ contributes jn ones to this attribute and $(j-1)n$ ones to each attribute in $[n]/\{j\}$, whereas the maximum sum of any attribute in $\mathbf{y}$ is n.

Since the transformation from $\mathcal{M}$ to $\mathcal{A}$ is deterministic, the coins of $\mathcal{A}$ are the same as the coins of $\mathcal{M}$. By (4) and the computation of the estimates for the $\mathsf{Marginals}_d$ in Step 9 of Algorithm 2,

$$\begin{aligned}
&\Pr_{\text{coins of } \mathcal{A}} \left[\mathsf{ERR}_{\mathsf{Marginals}}(\mathbf{y}, \mathcal{A}(\mathbf{y})) \leq \frac{\alpha}{n} \right] \\
&= \Pr_{\text{coins of } \mathcal{A}} \left[\max_{j \in [d]} |q_j(\mathbf{y}) - b_j| \leq \frac{\alpha}{n} \right] \\
&= \Pr_{\text{coins of } \mathcal{M}} \left[\max_{t \in \{2n, \ldots, 2dn\}} \left| \mathsf{MaxSum}_d(\mathbf{x}_{[t]}) - a_t \right| \leq \alpha \right] \\
&\geq \Pr_{\text{coins of } \mathcal{M}} \left[\max_{t \in [T]} \left| \mathsf{MaxSum}_d(\mathbf{x}_{[t]}) - a_t \right| \leq \alpha \right] \\
&= \Pr_{\text{coins of } \mathcal{M}} \left[\max_{t \in [T]} \mathsf{ERR}_{\mathsf{MaxSum}}(\mathbf{x}_{[t]}, a_t) \leq \alpha \right] \geq \frac{2}{3},
\end{aligned}$$

where we used that $\mathcal{M}$ is (α, T)-accurate for MaxSum_d. Thus, $\mathcal{A}$ is $(\frac{\alpha}{n}, n)$-accurate for $\mathsf{Marginals}_d$. □

Now, we are ready to prove Theorem 3.1.

Proof of Theorem 3.1. The accuracy parameter α is nondecreasing as a function of d since any mechanism $\mathcal{M}$ for MaxSum_d can be used to approximate $\mathsf{MaxSum}_{d'}$ for all $d' < d$ with the same accuracy and privacy guarantees (by padding each length-d' input record with $d - d'$ zeroes).

Recall that both lower bounds on α stated in Theorem 3.1 are the minimum of three terms. To prove them, it suffices to show that, for all ranges of parameters, one of the terms is a lower bound on α.

First, consider the case when $\varepsilon \leq \frac{2}{T}$. We will show that in this case (for both pure and approximate differential privacy), $\alpha > T/9$. Since α is a nondecreasing function of d, it is sufficient to show this for $d = 1$. Suppose for the sake of contradiction that $\alpha \leq T/9$. Let $\mathbf{x} = (0)^T$ and $\mathbf{x}' = (0)^{3T/4}(1)^{T/4}$ be datastreams that differ on $T/4$ records. Let a_T and a'_T be the final outputs of $\mathcal{M}$ on input streams $\mathbf{x}$ and $\mathbf{x}'$, respectively. By accuracy of $\mathcal{M}$, we have $\Pr[a_T \leq T/9] \geq 2/3$. Applying Lemma A.3 on group privacy with $\varepsilon \leq 2/T$ and $\ell = T/4$, we get $\Pr[a'_T > T/9] \leq \sqrt{e} \cdot \Pr[a_t > T/9] + \frac{2\delta}{\varepsilon} < 2/3$ for sufficiently large T, since $\delta = o(\varepsilon/T)$. But $\mathsf{MaxSum}_d(\mathbf{x}') = T/4$, so $\mathcal{M}$ is not $(T/9, T)$-accurate, a contradiction. Hence, $\alpha = \Omega(T)$.

Now assume $\varepsilon > \frac{2}{T}$, i.e., $\varepsilon T > 2$. We start by proving Item 1 (when $\delta = o(\varepsilon/T)$). Let $\mathcal{A}$ be the algorithm for $\mathsf{Marginals}_d$ with black-box access to $\mathcal{M}$, as defined in Algorithm 2. If $T \geq 2dn$ and $\frac{\alpha}{n} < 1$, then by Lemma C.1, algorithm $\mathcal{A}$ is (ε, δ)-differentially private and $(\frac{\alpha}{n}, n)$-accurate for $\mathsf{Marginals}_d$. (We require $\frac{\alpha}{n} < 1$ for the accuracy guarantee on $\mathcal{A}$ to be meaningful.) We can then use Lemma 3.3 to lower bound α.

Case 1: $d \leq (\varepsilon T \log(\varepsilon T))^{2/3}$**.** If there exists a dataset size $n \in (\alpha, \frac{T}{2d}]$, then by Item 1 of Lemma 3.3, $n = \Omega\big(\frac{n\sqrt{d}}{\alpha\cdot\varepsilon\log d}\big)$, and hence $\alpha = \Omega\big(\frac{\sqrt{d}}{\varepsilon\log d}\big)$. If no such n exists, then $\alpha + 1 \geq \frac{T}{2d}$, and hence $\alpha = \Omega(\frac{T}{d}) = \Omega\big(\frac{T^{1/3}}{\varepsilon^{2/3}\log^{2/3}(\varepsilon T)}\big)$. Combining the expressions for the two parameter ranges, we get that $\alpha = \Omega\big(\min\big\{\frac{T^{1/3}}{\varepsilon^{2/3}\log^{2/3}(\varepsilon T)}, \frac{\sqrt{d}}{\varepsilon\log d}\big\}\big)$.

Case 2: $d > (\varepsilon T \log(\varepsilon T))^{2/3}$**.** Set $d' = \lfloor(\varepsilon T\log(\varepsilon T))^{2/3}\rfloor$. Observe that $d \geq 1$ because $\varepsilon T > 2$. By our previous padding argument, a mechanism for MaxSum_d can be used to approximate $\mathsf{MaxSum}_{d'}$ for $d' = (\varepsilon T)^{2/3}$ with the same accuracy and privacy guarantees. Therefore, $\alpha = \Omega\left(\min\left\{\frac{T^{1/3}}{\varepsilon^{2/3}\log^{2/3}(\varepsilon T)}, \frac{\sqrt{d'}}{\varepsilon\log d'}\right\}\right) = \Omega\left(\frac{T^{1/3}}{\varepsilon^{2/3}\log^{2/3}(\varepsilon T)}\right)$.
This completes the proof of Item 1.

The proof of Item 2 (with $\delta = 0$) proceeds along the same lines, except that we consider the cases $d \leq \sqrt{\varepsilon T}$ and $d > \sqrt{\varepsilon T}$ and use Item 2 from Lemma 3.3 instead of Item 1. If a dataset size $n \in (\alpha, \frac{T}{2d}]$ exists, by Item 2 of Lemma 3.3, we get $n = \Omega\left(\frac{nd}{\alpha\varepsilon}\right)$, and hence $\alpha = \Omega\left(\frac{d}{\varepsilon}\right)$. If no such n exists, then $\alpha + 1 \geq \frac{T}{2d}$, and hence $\alpha = \Omega(T/d) = \Omega(\sqrt{T/\varepsilon})$. If $d > \sqrt{\varepsilon T}$, a padding argument gives that $\alpha = \Omega(\sqrt{T/\varepsilon})$. □

D. Proofs Omitted from Section 4

To prove our lower bounds for $\mathsf{SumSelect}$ in the continual release model with nonadaptively chosen inputs, we reduce from a problem called k-$\mathsf{IndSelect}$ that solves k disjoint instances of the problem of selecting the index of the largest marginal in the batch model.

To define the function k-$\mathsf{IndSelect}$, let $n, d, k \in \mathbb{N}$, and $\mathcal{X} = \{0,1\}^{kd}$. Let $\mathbf{y}[i:j]$ denote the dataset $\mathbf{y} \in \mathcal{X}^n$ with each record restricted to the coordinates between (and including) i and j. The function k-$\mathsf{IndSelect}_d : \mathcal{X}^n \to [d]^k$ corresponds to dividing the dataset into k *blocks* $\mathbf{y}[1:d], \mathbf{y}[d+1:2d], \ldots, \mathbf{y}[(k-1)d+1:kd]$, with n records each, and applying $\mathsf{SumSelect}_d$ independently on each block. It maps a dataset $\mathbf{y}$ of size n to a vector $(h_1(\mathbf{y}), \ldots, h_k(\mathbf{y}))$, where h_r is defined as the $\mathsf{SumSelect}_d$ function applied to block r:

$$h_r(\mathbf{y}) = \mathsf{SumSelect}_d\Big(\mathbf{y}\left[(r-1)d+1 : rd\right]\Big).$$

The accuracy for k-$\mathsf{IndSelect}$ is defined as in Definition 3.2. To apply it, we define the error $\mathsf{ERR}_{k\text{-}\mathsf{IndSelect}}$. Note that the error is scaled differently than for $\mathsf{SumSelect}$ because the goal is to select the index of the largest marginal in each block, not of the largest sum. For $\mathbf{b} = (b_1, \ldots, b_k) \in [d]^k$, define $\mathsf{ERR}_{k\text{-}\mathsf{IndSelect}}(\mathbf{y}, \mathbf{b})$

$$= \max_{r\in[k]}\left(\frac{1}{n}\cdot\mathsf{ERR}_{\mathsf{SumSelect}}(\mathbf{y}[r(d-1)+1:rd], b_r)\right).$$

Next, we give lower bounds for (ε, δ)-differentially private approximation of k-$\mathsf{IndSelect}$ in the batch model.

Lemma D.1. *For all $\varepsilon, \delta \in (0,1]$, $\gamma \in [0, \frac{1}{20}]$, sufficiently large $d, k, n \in \mathbb{N}$, and batch algorithms $\mathcal{A}$ that are (ε,δ)-differentially private and (γ, n)-accurate for k-$\mathsf{IndSelect}_d$, the following statements hold.*

1. *If $\delta \in (0, \frac{1}{n^2}]$, then $n = \Omega(\frac{\sqrt{k\cdot\ln(1/\delta)}\cdot\log d}{\varepsilon\gamma})$*
2. *If $\delta = 0$, then $n = \Omega\left(\frac{k\cdot\log d}{\varepsilon\gamma}\right)$.*

Proof Sketch of Item 1 in Lemma D.1. We are aware of two approaches to proving this result, both of which were communicated to us by Jonathan Ullman (Ullman, 2021). The first uses the top-k selection lower bound of Steinke & Ullman (2017). In that problem, there is a single collection of d coordinates and the goal is to return the indices of $k < d$ coordinates whose sums are roughly largest.

If one divides the coordinates into $10k$ equal groups, there is a constant probability that the collection of coordinates with the largest sum in each group is a good approximate solution for the top-k selection problem. An algorithm for k-$\mathsf{IndSelect}_d$ can thus be used to solve the top-k selection (out of dk coordinates) problem for such instances with roughly the same error and privacy parameter. The lower bound of Steinke & Ullman (2017) on n then applies. The statement we give here relies on a strengthening of the main result in Steinke & Ullman (2017) that incorporates δ, communicated to us by Thomas Steinke.

Another approach is to use the composition framework of Bun et al. (2018). One can use a folklore result that selection among $d > 2^m$ coordinates can be used to mount a reconstruction attack on an appropriate dataset of size m. Composed with the lower bound for 1-way marginals in Bun et al. (2018), one obtains a lower bound for k-$\mathsf{IndSelect}_d$. □

Proof of Item 2 in Lemma D.1. The proof proceeds via a standard packing argument. For $\mathbf{u} \in [d]^k$, define $y^*_{\mathbf{u}} \in \{0,1\}^{dk}$ to be the record where each block $r \in [k]$ of d coordinates has a 1 in coordinate u_r and all zeros everywhere else. Let $\mathbf{y}_{\mathbf{u}}$ be the dataset that consists of $2\gamma n$ copies of $y^*_{\mathbf{u}}$ and $(1-2\gamma)n$ copies of the all-zero record (assuming, for simplicity, that $2\gamma n$ is an integer). Since $\mathcal{A}$ is (γ, n)-accurate, $\Pr_{\text{coins of }\mathcal{A}}[\mathsf{ERR}_{k\text{-}\mathsf{IndSelect}}(\mathbf{y}_{\mathbf{u}}, \mathcal{A}(\mathbf{y}_{\mathbf{u}})) \leq \gamma] \geq \frac{2}{3}$ for all $\mathbf{u} \in [d]^k$. This means that for all $\mathbf{u} \in [d]^k$,

$$\Pr_{\text{coins of }\mathcal{A}}[\mathcal{A}(\mathbf{y}_{\mathbf{u}}) = \mathbf{u}] \geq \frac{2}{3}.$$

For all $\mathbf{u}, \mathbf{u}' \in [d]$, by group privacy, $\mathcal{A}(\mathbf{y_u}) \approx_{(\gamma\varepsilon n, 0)} \mathcal{A}(\mathbf{y_{u'}})$, which implies that

$$\Pr[\mathcal{A}(\mathbf{y_u}) = \mathbf{u}'] \geq e^{-\gamma\varepsilon n} \Pr[\mathcal{A}(\mathbf{y_{u'}}) = \mathbf{u}'] \geq \frac{2}{3} e^{-\gamma\varepsilon n}. \tag{5}$$

Since the probability of any event is at most 1,

$$1 \geq \Pr_{\text{coins of } \mathcal{A}} [\mathcal{A}(\mathbf{y_u}) \neq \mathbf{u}] = \sum_{\mathbf{u}' \neq \mathbf{u}} \Pr[\mathcal{A}(\mathbf{y_u}) = \mathbf{u}'] \geq \frac{2}{3} e^{-\gamma\varepsilon n}(d^k - 1),$$

where the last inequality holds by (5). We get that $e^{\gamma\varepsilon n} \geq \frac{d^k-1}{2} \cdot \frac{2}{3}$, and thus $n = \Omega\left(\frac{k \log d}{\gamma\varepsilon}\right)$. ☐

D.1. Proof of Theorem 4.1

Let $\mathcal{M}$ be an (ε, δ)-DP and (α, T)-accurate mechanism for $\mathsf{SumSelect}$ in the continual release model with nonadaptively chosen inputs. We use $\mathcal{M}$ to construct an (ε, δ)-DP algorithm $\mathcal{A}$ that is $(\frac{\alpha}{n}, n)$-accurate for k-$\mathsf{IndSelect}_d$ in the batch model. We motivate our approach by first discussing an idea that doesn't quite work. Let $\mathcal{M}$ be an accurate mechanism for $\mathsf{SumSelect}_d$ in the continual release model with nonadaptively chosen inputs and $\mathbf{y}$ be a dataset with n records from $\{0,1\}^{dk}$. A naive approach to solving k-$\mathsf{IndSelect}_d$ in the batch model is to run k instantiations of $\mathcal{M}$ for n time steps each, one on each block of d coordinates, to select the coordinate with the maximum sum in that block. However, running k instantiations of $\mathcal{M}$, as described, would result in a significant degradation of privacy, because every datapoint is used k times, once for each instantiation of $\mathcal{M}$. We instead reduce to $\mathsf{SumSelect}_{dk}$ and run a single instantiation of $\mathcal{M}$ for about nk time steps, where each datapoint in $\mathbf{y}$ is sent to $\mathcal{M}$ only once. This approach doesn't suffer from privacy degradation.

Algorithm $\mathcal{A}$ proceeds in k stages; the r^{th} stage is dedicated to selecting the coordinate with the maximum sum in the r^{th} block. In the first stage, $\mathcal{A}$ streams $\mathbf{y}$ to $\mathcal{M}$. In order to select the coordinate with the maximum sum from the first block, $\mathcal{A}$ then sends $2n$ records of the form $(1^d 0^d \ldots 0^d)$ to $\mathcal{M}$. Then the sums of the coordinates in the first block of $\mathbf{y}$ become much larger than the sums in the other blocks. This ensures that at the end of the first stage, $\mathcal{M}$ selects the coordinate with the maximum sum in the first block. In the second stage, $\mathcal{A}$ sends $2n$ records of the form $(0^d 1^d \ldots 1^d)$ to $\mathcal{M}$ in order to balance out the number of extraneous 1's for each coordinate. In order to select the coordinate with the maximum sum from the second block, $\mathcal{A}$ sends $2n$ records of the form $(0^d 1^d 0^d \ldots 0^d)$ to $\mathcal{M}$. At the end of the second stage, $\mathcal{M}$ selects the coordinate with the maximum sum in the second block. Algorithm $\mathcal{A}$ proceeds similarly for every block.

The details of the algorithm appear in Algorithm 3. For ease of indexing, $\mathcal{A}$ sends all-zero records in time steps $n+1$ to $2n$ in Step 4 of Algorithm 3, to ensure that all stages have $4n$ time steps.

Algorithm 3 Batch algorithm $\mathcal{A}$ for k-$\mathsf{IndSelect}$

1: **Input:** k, $\mathbf{y} = (y_1, \ldots, y_n) \in \mathcal{X}^n$, where $\mathcal{X} = \{0,1\}^{dk}$, and black-box access to mechanism $\mathcal{M}$.
2: **Output:** $\mathbf{b} = (b_1, \ldots, b_k) \in [d]^k$.
3: Let $\mathbf{v}_j$ be a vector of length dk with d ones in coordinates $[dj] \setminus [d(j-1)]$ and 0 everywhere else; let $\overline{\mathbf{v}_j} \leftarrow 1^{dk} - \mathbf{v}_j$.
4: Construct a stream $\mathbf{x} \leftarrow \mathbf{y} \circ (0^{dk})^n \circ (\mathbf{v}_1)^{2n} \circ (\overline{\mathbf{v}_1})^{2n} \circ \cdots \circ (\mathbf{v}_{k-1})^{2n} \circ (\overline{\mathbf{v}_{k-1}})^{2n} \circ (\mathbf{v}_k)^{2n}$ with $4kn$ records.
5: **for** $t \in [T]$ **do**
6:     Send the record x_t to $\mathcal{M}$ and get the corresponding output a_t.
7: **end for**
8: **for** $r \in [k]$ **do**
9:     $b_r \leftarrow a_{4rn} - d(r-1)$. If $b_r \notin [d]$, then $b_r \leftarrow 1$.
10: **end for**
11: Output $\mathbf{b} \leftarrow (b_1, \ldots, b_k)$.

Lemma D.2. *Let $\mathcal{A}$ be Algorithm 3. For all $\varepsilon > 0, \delta \geq 0$, $\alpha \in \mathbb{R}^+$, and $T, d, k, n \in \mathbb{N}$, where $T \geq 4kn$, if mechanism $\mathcal{M}$ is (ε, δ)-differentially private and (α, T)-accurate for* $\mathsf{SumSelect}_{dk}$ *in the continual release model with nonadaptively chosen inputs, then batch algorithm $\mathcal{A}$ is (ε, δ)-differentially private and $(\frac{\alpha}{n}, n)$-accurate for k-*$\mathsf{IndSelect}_d$.

Proof. We start by reasoning about privacy. Fix neighboring datasets $\mathbf{y}$ and $\mathbf{y}'$ that are inputs to algorithm $\mathcal{A}$. Let $\mathbf{x}$ and $\mathbf{x}'$ be the streams constructed in Step 4 of $\mathcal{A}$ when it is run on $\mathbf{y}$ and $\mathbf{y}'$, respectively. By construction, $\mathbf{x}$ and $\mathbf{x}'$ are neighboring streams. Since $\mathcal{M}$ is (ε, δ)-DP, and $\mathcal{A}$ only post-processes the outputs received from $\mathcal{M}$, Lemma A.4 implies that $\mathcal{A}$ is (ε, δ)-DP.

Next, we reason about accuracy. Fix a dataset $\mathbf{y}$ and the corresponding data stream $\mathbf{x}$ sent to $\mathcal{M}$. Consider a setting τ of the random coins of $\mathcal{A}$. Since the transformation from $\mathcal{M}$ to $\mathcal{A}$ is deterministic, they correspond to coins used by $\mathcal{M}$ when $\mathcal{A}$ runs it as a subroutine. Let α_τ be the realized error of $\mathcal{M}$ with coins τ, that is,

$$\alpha_\tau = \max_{t \in [4kn]} \left(\mathsf{ERR}_{\mathsf{SumSelect}_{dk}}(\mathbf{x}_{[t]}, a_t) \right),$$

where a_t are the answers with coins τ. Similarly, let γ_τ be the realized error of $\mathcal{A}$ with coins τ, that is,

$$\begin{aligned}\gamma_\tau &= \mathsf{ERR}_{k\text{-}\mathsf{IndSelect}_{dk}}(\mathbf{y}, \mathbf{b}) \\ &= \frac{1}{n} \cdot \max_{r \in [k]} \left(\mathsf{ERR}_{\mathsf{SumSelect}_d}(\mathbf{y}[(r-1)d + 1 : rd], b_r) \right),\end{aligned}$$

where $\mathbf{b} = (b_1, \ldots, b_k)$ is the output of $\mathcal{A}$ run with coins τ.

The main observation in the accuracy analysis is that if α_τ is small, so is γ_τ. Note that if $\alpha \geq n$, the accuracy guarantee for $\mathcal{A}$ is vacuous. Now assume $\alpha < n$. For all blocks $r \in [k]$, the sums in $\mathbf{x}_{[4rn]} = \mathbf{y} \circ (0^{dk})^n \circ (\mathbf{v}_1)^{2n} \circ (\overline{\mathbf{v}_1})^{2n} \circ \cdots \circ (\mathbf{v}_{r-1})^{2n} \circ (\overline{\mathbf{v}_{r-1}})^{2n} \circ (\mathbf{v}_r)^{2n}$ of all coordinates not in block r are smaller than the sums of coordinates in block r by at least n. Consider coins τ with $\alpha_\tau \leq \alpha$. Since $\alpha_\tau < n$, the index a_{4rn} returned by $\mathcal{M}$ is in block r for all $r \in [k]$. Moreover, the error for each block is at most $\frac{\alpha_\tau}{n}$. Therefore, $\gamma_\tau \leq \frac{\alpha_\tau}{n} \leq \frac{\alpha}{n}$. Considering the probability of this event over all coins τ, we get

$$\Pr_{\text{coins } \tau \text{ of } \mathcal{A}} \left[\gamma_\tau \leq \frac{\alpha}{n}\right] \geq \Pr_{\text{coins } \tau \text{ of } \mathcal{M}} [\gamma_\tau \leq \alpha] \geq \frac{2}{3},$$

where the last inequality holds because $\mathcal{M}$ is (α, T)-accurate. We conclude that $\mathcal{A}$ is $(\frac{\alpha}{n}, n)$-accurate. $\square$

Finally, we prove Theorem 4.1.

Proof of Theorem 4.1. This proof's structure resembles that of Theorem 3.1. First, for the case of $\varepsilon \leq \frac{2}{T}$, we prove that $\alpha = \Omega(T)$. Let $\mathbf{e}_j$ be a record of length d with 1 in coordinate j and 0 everywhere else. Let $\mathbf{x} = (\mathbf{e}_1)^{T/4} \circ (0^d)^{3T/4}$ and $\mathbf{x}' = (\mathbf{e}_2)^{T/4} \circ (0^d)^{3T/4}$. Proceeding as in the proof of Theorem 3.1 (using group privacy and the error associated with selection) yields $\alpha = \Omega(T)$.

For all other values of ε, we reduce from k-$\mathsf{IndSelect}$, relying on the lower bounds for k-$\mathsf{IndSelect}$ from Lemma D.1. Fix T, d, ε. Given an integer k, the reduction of Lemma D.2 maps a batch instance of k-$\mathsf{IndSelect}_{d'}$ of size n to an instance of $\mathsf{SumSelect}_d$ with $d = d'k$ and $T = 4nk$. The reduction applies as long as $d' = \frac{d}{k} \geq 2$ and $n = \frac{T}{4k} \geq 1$ are integers. We will ignore the integrality requirement (which can be addressed by appropriate padding) and allow any k between 1 and $\min(\frac{d}{2}, \frac{T}{4})$.

When $\delta = o(\frac{\varepsilon}{T^2})$, we get $\delta = o(\frac{1}{n^2})$ (since $\varepsilon < 1$ and $T > n$), since our reduction in Lemma D.2 preserves δ, we are in the range of δ where Lemma D.1 Part 1 applies. This gives us a lower bound on the error of $\min\left(\Omega\left(\frac{\sqrt{k}\log d'}{\varepsilon}\right), n\right)$ when k and d' are sufficiently large constants. In our setting, this translates to a lower bound of $\Omega(\alpha_k)$ for $\alpha_k = \min\left(\frac{\sqrt{k}\log(2+d/k)}{\varepsilon}, \frac{T}{k}\right)$. (We add 2 inside the logarithms to avoid 0 or subconstant log terms; this does not change the asymptotics.) For simplicity, we omit the dependency on $\log(1/\delta)$ in the lower bounds.

Our goal is to select the value of $k \in [1, \min(\frac{d}{2}, \frac{T}{4})]$ that maximizes α_k. For fixed T, d, ε, let $k^* = k^*(T, d, \varepsilon) = \max(1, k')$ where k' denotes the largest value of k where the two terms defining α_k equalize (that is, k' satisfies $k'\sqrt{k'}\log(2 + d/k') = \varepsilon T$). We use two basic facts about α_k: first, for $d \geq 1$, the function α_k is increasing on $[1, k^*)$ and decreasing on (k^*, ∞).

Second, its maximum value α_{k^*} is $\tilde{\Omega}(\frac{T^{1/3}\log^{2/3} d}{\varepsilon^{2/3}})$. To see why this is, note that $k' = (\varepsilon T/\log(2 + d'))^{2/3}$, and so $\alpha_{k'} = \frac{\sqrt{k'}\log(2+d')}{\varepsilon} = \frac{T^{1/3}\log^{2/3}(2+d')}{\varepsilon^{2/3}}$. Consider two cases: if $k' \leq \sqrt{d}$, then $d' \geq \sqrt{d}$ and so $\log(2 + d') = \Theta(\log d)$. On the other hand, if $k' \geq \sqrt{d}$, then $\alpha_{k'}$, which is always at least $\sqrt{k'}$, is bounded below by $d^{1/4}$. Therefore the factor of $\log(d)$ is polynomial in $\log(\alpha_{k'})$ and absorbed by the $\tilde{\Omega}$ notation.

We consider four regimes for the triple (T, d, ε). Let C denote a constant such that Lemma D.1 applies for $d' \geq C$.

(*a*) $k^*(T, d, \varepsilon) = 1$: In this case, α_k is maximized at $k = 1$. Since setting k to be a sufficiently large constant does not change the asymptotics of the lower bound, we can set k sufficiently large for Lemma D.1 to apply, and obtain a lower bound of $\Omega(T/k) = \Omega(T)$.

(*b*) $k^*(T, d, \varepsilon) > \min(\frac{d}{C}, \frac{T}{4})$ and $Cd \leq T$: In this case, we set $k = d/C$. We get a lower bound of $\alpha_k = \frac{\sqrt{k}\log(2+d/k)}{\varepsilon}$ (since $k \leq k^*$), which is $\Omega(\frac{\sqrt{d}}{\varepsilon})$.

(*c*) $k^*(T, d, \varepsilon) > \min(\frac{d}{C}, \frac{T}{4})$ and $Cd > T$: This case is not possible for large T. For it to occur, we must have $k^* > T/4$, which implies that $\alpha_{k^*} < 4$. Since $\alpha_{k^*} = \tilde{\Omega}(\frac{T^{1/3}\log^{2/3} d}{\varepsilon^{2/3}})$, we get that $\varepsilon > 1$ (for sufficiently large T), contradicting our assumptions.

(*d*) $k^*(T, d, \varepsilon) \in [1, \min(\frac{d}{C}, \frac{T}{4})]$: In this case, we set $k = k^*$ and obtain a lower bound of $\alpha_{k^*} = \tilde{\Omega}(\frac{T^{1/3}\log^{2/3} d}{\varepsilon^{2/3}})$ (Note that if k^* is too small, we can set $k = ck^*$ for a sufficiently large constant c without changing the asymptotics, as in part a).

Thus, for all possible relationships between T, d and ε, we obtain a lower bound that is one of three terms in the theorem statement.

The setting in which $\delta = 0$ is similar. For a given $k \in [1, \min(\frac{d}{2}, \frac{T}{4})]$, we obtain a lower bound of $\Omega(\alpha_k)$ for $\alpha_k = \min\left(\frac{k\log\left(2+\frac{d}{k}\right)}{\varepsilon}, \frac{T}{k}\right)$. The remaining calculations parallel the case where $\delta > 0$, except that now $\alpha_{k^*} = \tilde{\Omega}\left(\sqrt{\frac{T\log d}{\varepsilon}}\right)$. $\square$

E. Details Omitted from Section 5

E.1. Formal Statements

In this subsection, we state theorems that summarize the performance guarantees of our mechanisms for MaxSum_d and $\mathsf{SumSelect}_d$. We prove these theorems in the following subsections. The theorems provide a stronger privacy guarantee than (ε, δ)-DP, specifically, concentrated differential privacy. We state implications for (ε, δ)-DP in Corollary E.3. The upper bounds in our theorems are attained by two simple mechanisms: one uses the binary tree mechanism and the other recomputes the target function at regular intervals. Prior to our work, it was not known that these mechanisms are private in the setting with adaptively chosen inputs. As mentioned in Section 1.1, a proof of privacy with adaptively chosen inputs cannot be generally reduced to a proof in the nonadaptive model. Instead, we build on techniques from simulation-based proofs in cryptography to argue the indistinguishability of input-output distributions directly.

Theorem E.1 (zCDP, Binary-Tree-Based Mechanisms)**.** *For all $\rho \in (0, 1]$, $d \in \mathbb{N}$, and sufficiently large $T > 0$, there exist ρ-zCDP mechanisms $\mathcal{M}$ and $\mathcal{M}'$ in the continual release model with adaptively chosen inputs such that $\mathcal{M}$ is (α, T)-accurate for MaxSum_d and $\mathcal{M}'$ is (α, T)-accurate for $\mathsf{SumSelect}_d$, where $\alpha = \mathrm{O}\left(\frac{\sqrt{d}\log T\sqrt{\log(dT)}}{\sqrt{\rho}}\right)$.*

The next theorem uses the idea of recomputing at regular intervals, which applies quite generally. Item 1 of Theorem E.2 applies for general sensitivity-1 functions (which include MaxSum_d); a similar result holds for bounded-sensitivity functions with output space $\mathbb{R}^d$.

Theorem E.2 (zCDP, Mechanisms via Recomputing at Regular Intervals)**.** *For all $\rho \in (0, 1]$, $d \in \mathbb{N}$, sufficiently large $T > 0$, and all functions $f : \mathcal{X}^* \to \mathbb{R}$ with ℓ_2-sensitivity at most 1, there exist ρ-zCDP mechanisms $\mathcal{M}$ and $\mathcal{M}'$ in the continual release model with adaptively chosen inputs such that*

1. *Mechanism $\mathcal{M}$ is (α, T)-accurate for f for $\alpha = \mathrm{O}\left(\min\left\{\sqrt[3]{\frac{T\log T}{\rho}}, T\right\}\right)$;*
2. *Mechanism $\mathcal{M}'$ is (α, T)-accurate for $\mathsf{SumSelect}_d$ for $\alpha = \mathrm{O}\left(\min\left\{\frac{T^{1/3}\log^{2/3}(dT)}{\rho^{1/3}}, T\right\}\right)$.*

We combine Theorems E.1–E.2, use the conversion from zCDP to (ε, δ)-DP from Lemma A.15, and substitute $\rho = \frac{\varepsilon^2}{16\log(1/\delta)}$ to get the following corollary.

Corollary E.3. *For all $\varepsilon \in (0, 1]$, $\delta \in (0, \frac{1}{2}]$, $d \in \mathbb{N}$, and sufficiently large $T > 0$, there exist (ε, δ)-DP mechanisms $\mathcal{M}$ and $\mathcal{M}'$ in the continual release model with adaptively chosen inputs such that*

1. *$\mathcal{M}$ is (α, T)-accurate for MaxSum_d for $\alpha = \mathrm{O}\left(\min\left\{\frac{\sqrt[3]{T\log(1/\delta)\log T}}{\varepsilon^{2/3}}, \frac{\sqrt{d\log(dT)\log(1/\delta)}\,\log T}{\varepsilon}, T\right\}\right)$;*
2. *$\mathcal{M}'$ is (α, T)-accurate for $\mathsf{SumSelect}_d$ for $\alpha = \mathrm{O}\left(\min\left\{\frac{\sqrt[3]{T\log^2(dT)\log(1/\delta)}}{\varepsilon^{2/3}}, \frac{\sqrt{d\log(dT)\log(1/\delta)}\,\log T}{\varepsilon}, T\right\}\right)$.*

Simple variants of our mechanisms can be used to get the following theorems for $(\varepsilon, 0)$-differential privacy.

Theorem E.4 (Pure DP, Binary-Tree-Based Mechanisms)**.** *For all $\varepsilon \in (0, 1]$, $d \in \mathbb{N}$, and sufficiently large $T > 0$, there exist $(\varepsilon, 0)$-DP mechanisms $\mathcal{M}$ and $\mathcal{M}'$ in the continual release model with adaptively chosen inputs such that $\mathcal{M}$ is (α, T)-accurate for MaxSum_d and $\mathcal{M}'$ is (α, T)-accurate for $\mathsf{SumSelect}_d$ for $\alpha = \mathrm{O}\left(\frac{d(\log d)\log^3 T}{\varepsilon}\right)$.*

Theorem E.5 (Pure DP, Mechanisms via Recomputing at Regular Intervals)**.** *For all $\varepsilon \in (0, 1]$, $d \in \mathbb{N}$, sufficiently large $T > 0$, and all functions $f : \mathcal{X}^* \to \mathbb{R}$ with ℓ_1-sensitivity at most 1, there exist $(\varepsilon, 0)$-DP mechanisms $\mathcal{M}$ and $\mathcal{M}'$ in the continual release model with adaptively chosen inputs such that*

1. *Mechanism $\mathcal{M}$ is (α, T)-accurate for f for $\alpha = \mathrm{O}\left(\min\left\{\sqrt{\frac{T\log T}{\varepsilon}}, T\right\}\right)$;*
2. *Mechanism $\mathcal{M}'$ is (α, T)-accurate for $\mathsf{SumSelect}_d$ for $\alpha = \mathrm{O}\left(\min\left\{\sqrt{\frac{T\log(dT)}{\varepsilon}}, T\right\}\right)$.*

Theorems E.4–E.5 yield the following corollary.

Corollary E.6. *For all $\varepsilon \in (0, 1]$, $d \in \mathbb{N}$, and sufficiently large $T > 0$, there exist $(\varepsilon, 0)$-DP mechanisms $\mathcal{M}$ and $\mathcal{M}'$ in the continual release model with adaptively chosen inputs such that*

1. *$\mathcal{M}$ is (α, T)-accurate for MaxSum_d for $\alpha = \mathrm{O}\left(\min\left\{\sqrt{\frac{T\log T}{\varepsilon}}, T, \frac{d(\log d)\log^3 T}{\varepsilon}\right\}\right)$;*
2. *Mechanism $\mathcal{M}'$ is (α, T)-accurate for $\mathsf{SumSelect}_d$ for $\alpha = \mathrm{O}\left(\min\left\{\sqrt{\frac{T\log(dT)}{\varepsilon}}, T, \frac{d(\log d)\log^3 T}{\varepsilon}\right\}\right)$.*

E.2. Algorithms based on the Binary Tree Mechanism

In this section, we prove Theorem E.1 for $\mathsf{SumSelect}_d$. Theorem E.1 for MaxSum_d follows from the same analysis by considering the binary tree mechanism that outputs the highest noisy sum instead of the coordinate that achieves it.

In order to approximate $\mathsf{SumSelect}_d$ on a dataset with d attributes, we use the binary tree mechanism from Chan et al. (2011); Dwork et al. (2010a) to privately sum each of the attributes of the records $\mathbf{x}_{[t]}$ received so far, and then choose the attribute with the highest sum. For simplicity of exposition, in this section, we assume that T is a power of 2. In general, we can work with the smallest power of 2 greater than T. Throughout this section, $[i : j]$, where $i, j \in \mathbb{N}$, denotes the set of natural numbers $\{i, \ldots, j\}$.

Algorithm 4 Mechanism $\mathcal{M}$ for $\mathsf{SumSelect}_d$ in continual release model with adaptively chosen inputs

Input: time horizon $T \in \mathbb{N}$, privacy parameter ρ, stream $\mathbf{x} = (x_1, \dots, x_T) \in \mathcal{X}^T$, where $\mathcal{X} = \{0,1\}^d$.
Output: stream $(a_1, \dots, a_T) \in [d]^T$.
Init: Construct a complete binary tree with T leaves labeled $v_{[1:1]}, \dots, v_{[T:T]}$. Label every internal node $v_{[\ell:r]}$ if the subtree rooted at that node has leaves $v_{[\ell:\ell]}, \dots, v_{[r:r]}$. Initialize the partial sum $s_{[\ell:r]} \leftarrow 0^d$ for each node $v_{[\ell:r]}$ in the tree.
for $t = 1$ to T **do**
  Get record x_t from $\mathcal{A}dv$.
  ▷ *Compute noisy sums for nodes completed at time t*
  for each node $v_{[\ell:t]}$ **do**
    Draw noise $Z \sim \mathcal{N}(0, \sigma^2 \mathbb{I}^{d\times d})$, where $\sigma = \sqrt{\frac{d(\log T+1)}{2\rho}}$, and set $s_{[\ell:t]} \leftarrow \sum_{i=\ell}^{r} x_i + Z$.
  end for
  ▷ *Output Steps:*
  $I_t \leftarrow$ collection of at most $\log t + 1$ intervals that partition $[1:t]$, where each interval labels a node in the binary tree. (See Remark E.2.)
  $\mathsf{sum}_t \leftarrow \sum_{[\ell:r]\in I_t} s_{[\ell:r]}$.
  Output $a_t \leftarrow \arg\max_{j\in[d]} \mathsf{sum}_t[j]$.
end for

At the high level, the binary tree mechanism constructs a complete binary tree with T leaves. The leaves correspond to the input records $\mathbf{x}_{[t]}$, where each record $x_i \in \{0,1\}^d$. Each internal node in the tree corresponds to the sum of all the leaves in its subtree. Each node stores the noisy version of the corresponding sum computed by adding a noise vector drawn from $\mathcal{N}(0, \sigma^2\mathbb{I}^{d\times d})$ with $\sigma = \sqrt{\frac{d(\log T+1)}{2\rho}}$. The algorithm that releases the noisy sum is $\frac{\rho}{\log T+1}$-zCDP. Since each x_t participates in only $\log_2 T + 1$ sums in the tree, by *adaptive composition* of zCDP (Lemma A.13), the complete mechanism is ρ-zCDP (Theorem E.1). The sum of all the attributes at any timestep can be calculated by adding at most $\log T$ of the sums stored in the tree, one at each level. The algorithm that adds the corresponding noisy sums is (α, T)-accurate for $\alpha \approx \mathrm{O}\left(\frac{\sqrt{d}\log T \log(Td)}{\sqrt{\rho}}\right)$. The formal description of the algorithm appears in Algorithm 4. The algorithm uses a dyadic decomposition (described in Remark E.2) to decide which nodes of the tree it accesses to compute any particular output.

Remark (Dyadic Decomposition)**.** *For any natural number $t > 1$, the interval $[1:t]$ can be expressed as a union of at most $\log t + 1$ disjoint intervals as follows. Consider the binary expansion of t (which has at most $\log t + 1$ bits), and express t as a sum of distinct powers of 2 ordered from higher to lower powers. Then, the first interval $[1:r]$ will have size equal to the largest power of 2 in the sum. The second interval will start at $r+1$ and its size will be equal to the second largest power of 2 in the sum. Similarly, the remaining intervals are defined until all terms in the summation have been exhausted. For example, for $t = 7 = 4+2+1$, the intervals are $[1:4]$, $[5:6]$ and $\{7\}$.*

We present the privacy and accuracy analysis for Algorithm 4 in Lemmas E.7 and E.8, respectively, which together prove Theorem E.1 for $\mathsf{SumSelect}$.

Lemma E.7. *For all $\rho \in \mathbb{R}^+$, $d, T \in \mathbb{N}$, mechanism $\mathcal{M}$ described in Algorithm 4 is ρ-zCDP in the continual release model with adaptively chosen inputs.*

Proof. Consider an adversary $\mathcal{A}dv$ interacting with the privacy game $\Pi_{\mathcal{M},\mathcal{A}dv}$. We want to argue that the adversary's view is ρ-close in the two versions of the privacy game (for the two possible values of $\mathsf{side} \in \{L, R\}$.) We will achieve this by introducing a ρ-zCDP mechanism $\mathcal{M}_{\mathsf{gauss}}$ with input side and reducing our goal to the privacy of $\mathcal{M}_{\mathsf{gauss}}$.

For this, we use a simulation argument similar to those used in cryptography. Specifically, our proof defines two algorithms: (a) a ρ-zCDP mechanism $\mathcal{M}_{\mathsf{gauss}}$ that gets input $\mathsf{side} \in \{L, R\}$ and (b) a simulator $\mathcal{S}\mathrm{im}$ with query access to $\mathcal{M}_{\mathsf{gauss}}$ that does not know the value of side. The simulator $\mathcal{S}\mathrm{im}$ interacts with adversary $\mathcal{A}dv$ and satisfies a key guarantee:

> The view of the adversary $\mathcal{A}dv$ in its interaction with $\mathcal{S}\mathrm{im}$ is identically distributed to its view in the privacy game $\Pi_{\mathcal{M},\mathcal{A}dv}$, defined in Algorithm 1. (Figure 1 illustrates the structure of these two kinds of interaction.)

Since the simulator's outputs to $\mathcal{A}dv$ are a post-processing of the query responses from $\mathcal{M}_{\mathsf{gauss}}$, we can argue that the adversary's view is ρ-close in the two versions of the privacy game $\Pi_{\mathcal{M},\mathcal{A}dv}$.

To see why this is helpful, recall that we want to show that the probability of $\mathcal{A}dv$ guessing the value of side in the privacy game is small. If the probability of $\mathcal{A}dv$ guessing the value of side is the same in the privacy game as in its interaction with $\mathcal{S}\mathrm{im}$, then—since the simulator doesn't know the value of side—$\mathcal{A}dv$ can only learn as much about side from its interaction with $\mathcal{S}\mathrm{im}$ as one can learn by querying $\mathcal{M}_{\mathsf{gauss}}$. Intuitively, if $\mathcal{M}_{\mathsf{gauss}}$ does not reveal much about the value of side then neither does $\mathcal{M}$. We now describe $\mathcal{M}_{\mathsf{gauss}}$ and the simulator, and formalize the argument.

The mechanism $\mathcal{M}_{\mathsf{gauss}}$ (described in Algorithm 6) gets an input $\mathsf{side} \in \{L, R\}$. It receives at most $\log T + 1$

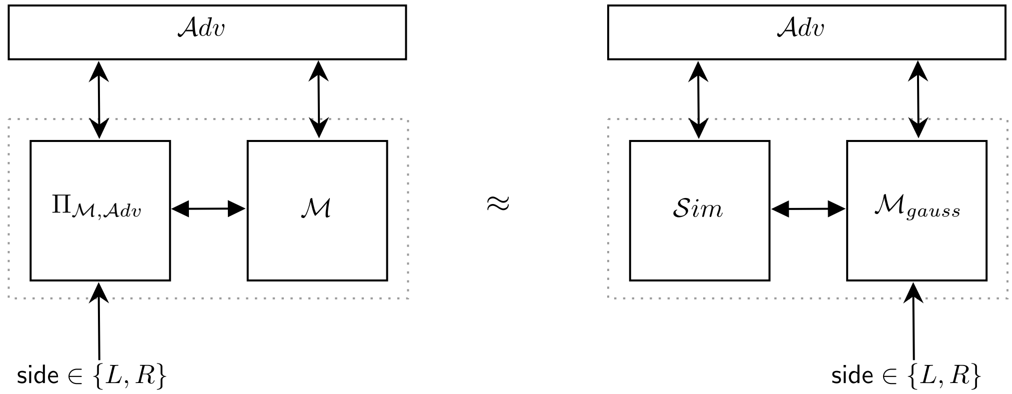

Figure 1. An illustration of the simulation argument from the proof of Lemma E.7. The left-hand side shows the game used to define privacy with adaptively selected inputs. The right-hand side shows the simulation structure described in the proof. For each value of side, the adversary's view is identical in these two settings.

queries of the form $v^{(L)}, v^{(R)}$ from $\mathcal{S}$im to which it responds with $p = v^{(\text{side})} + Z$ where the noise Z is drawn from $\mathcal{N}(0, \sigma^2 \mathbb{I}^{d\times d})$ for $\sigma = \sqrt{\frac{d(\log T+1)}{2\rho}}$. Observe that if $\mathcal{M}_{\text{gauss}}$ has only a single interaction with $\mathcal{S}$im and outputs a single noised value, then by the privacy guarantee of the Gaussian mechanism (Lemma A.14), $\mathcal{M}_{\text{gauss}}$ is $\frac{\rho}{\log T+1}$-zCDP. This can be seen by imagining that $\mathcal{M}_{\text{gauss}}$ is computing a function $f(\text{side}) = x_i^{\text{side}}$ and observing that the ℓ_2-sensitivity of f is $\sqrt{d}$. Since there are $\log T + 1$ interactions between $\mathcal{M}_{\text{gauss}}$ and $\mathcal{S}$im, $\mathcal{M}_{\text{gauss}}$ is an adaptive composition of $\log T + 1$ algorithms, each of which is $\frac{\rho}{\log T+1}$-zCDP. By Lemma A.13 on composition, $\mathcal{M}_{\text{gauss}}$ is ρ-zCDP.

The simulator $\mathcal{S}$im (described in Algorithm 5) interacts with the adversary without knowing the input side $\in \{L, R\}$ that is given to $\mathcal{M}_{\text{gauss}}$. It queries $\mathcal{M}_{\text{gauss}}$ exactly $\log T{+}1$ times and uses the query responses to provide outputs to the adversary. The aim of the simulator is to mimic the behaviour of $\Pi_{\mathcal{M},\mathcal{A}dv}$ even though it doesn't know side. The simulator constructs a binary tree as described in Algorithm 4. For all nodes in the binary tree except for those whose interval contains the challenge timestep t^*, the computation of the noisy subtree sums can be done by $\mathcal{S}$im without any help from $\mathcal{M}_{\text{gauss}}$. For the nodes whose interval does contain t^*, the simulator sends $(x_{t^*}^{(L)}, x_{t^*}^{(R)})$ to $\mathcal{M}_{\text{gauss}}$ and gets a noisy value of $x_{t^*}^{\text{side}}$. It can then compute the corresponding subtree sum by adding the input records corresponding to the remaining leaves. Notice that the simulator can produce these outputs online—at the same time that $\Pi_{\mathcal{M},\mathcal{A}dv}$ would.

The crucial point to note is that the view of the adversary $\mathcal{A}dv$ in the privacy game $\Pi_{\mathcal{M},\mathcal{A}dv}$ is identically distributed to its view in the interaction with $\mathcal{M}_{\text{gauss}}$ and $\mathcal{S}$im. Furthermore, the view of the adversary $\mathcal{A}dv$ when interacting with $\mathcal{S}$im and $\mathcal{M}_{\text{gauss}}$ is simply a post-processing of the outputs provided to it by $\mathcal{S}$im, which are a post-processing of the outputs provided to $\mathcal{S}$im by $\mathcal{M}_{\text{gauss}}$. Hence,

$$\mathcal{M}_{\text{gauss}} \text{ is } \rho\text{-zCDP} \implies V^{(L)}_{\mathcal{M},\mathcal{A}dv} \simeq_\rho V^{(R)}_{\mathcal{M},\mathcal{A}dv}.$$

It remains to argue that exactly $\log T + 1$ nodes have a subtree sum that depends on the inputs from the challenge timestep t^*. Each node $v_{[\ell,r]}$ whose subtree sum depends on the inputs from timestep t^* satisfies $t^* \in [\ell : r]$. This holds only for one node at each level of the binary tree created by $\mathcal{S}$im (because the intervals represented by the nodes at a particular level are disjoint.) Since the binary tree has depth $\log T + 1$, exactly $\log T + 1$ nodes have a subtree sum that depends on the inputs from the challenge timestep t^*. $\square$

Lemma E.8. *For all $\rho > 0$ and sufficiently large $T \in \mathbb{N}$, mechanism $\mathcal{M}$ is (α, T)-accurate for* SumSelect *in the continual release model with adaptively chosen inputs for $\alpha = O\left(\frac{\sqrt{d}\log T\sqrt{\log(Td)}}{\sqrt{\rho}}\right)$.*

Proof. Consider any adversarial process $\mathcal{A}dv$ interacting with $\mathcal{M}$. We first argue that, at every timestep t, the random variable $\text{ERR}_{\text{SumSelect}_d}(\mathbf{x}_{[t]}, a_t)$ corresponding to the error at any timestep t can be upper bounded by a random variable that is the sum of at most $2\log t$ independent Gaussian random variables. We then use tail bounds for Gaussian random variables, along with a union bound, to argue that, with high probability, the maximum value of this random variable is not too large. Finally, we take a union bound over timesteps to argue that, with high probability, $\max_{t\in[T]} \text{ERR}_{\text{SumSelect}_d}(\mathbf{x}_{[t]}, a_t)$ is not too large.

First, at any timestep t, let sum_t represent the vector of noisy sums defined in Step 10 of Algorithm 4. Therefore,

Algorithm 5 Simulator $\mathcal{S}$im for the proof of Lemma E.7

Input: time horizon $T \in \mathbb{N}$, privacy parameter $\rho \in \mathbb{R}^+$, black-box access to an adversary $\mathcal{A}dv$ and a mechanism $\mathcal{M}_{\mathsf{gauss}}$.
Output: stream $(a_1, \ldots, a_T) \in [d]^T$.
$\boldsymbol{\mathcal{A}dv}$**:** At each timestep $t \in [T] \setminus \{t^*\}$, $\mathcal{A}dv$ provides $\mathcal{S}$im with record $x_t \in \mathcal{X}$, where $\mathcal{X} = \{0,1\}^d$. At the challenge timestep t^* (chosen by $\mathcal{A}dv$), it provides records $x_{t^*}^{(L)}, x_{t^*}^{(R)} \in \mathcal{X}$. At each timestep $t \in [T]$, $\mathcal{S}$im provides $\mathcal{A}dv$ with output $a_t \in [d]$.
$\boldsymbol{\mathcal{M}_{\mathsf{gauss}}}$**:** $\mathcal{S}$im exchanges $\log_2 T + 1$ messages with $\mathcal{M}_{\mathsf{gauss}}$.
Init: Perform Step 1 (the initialization phase) of Algorithm 4.
$j \leftarrow 1$.
for $t \in [T]$ **do**
  if $t = t^*$ **then**
    Get input $(x_{t^*}^{(L)}, x_{t^*}^{(R)})$ from $\mathcal{A}dv$.
    for $i \in [\log T + 1]$ **do**
      Send $(x_{t^*}^{(L)}, x_{t^*}^{(R)})$ to $\mathcal{M}_{\mathsf{gauss}}$, and get back a response p_i.
    end for
  else
    Get record x_t from $\mathcal{A}dv$.
  end if
  for each node $v_{[\ell,t]}$ **do**
    if $t^* \notin [\ell : t]$ where $[\ell : t]$ denotes the integers $\{\ell, ..., t\}$ **then**
      Draw noise $Z \sim \mathcal{N}(0, \sigma^2 \mathbb{I}^{d\times d})$, where $\sigma = \sqrt{\frac{d(\log T+1)}{2\rho}}$.
      $s_{[\ell:t]} \leftarrow Z + \sum_{i=\ell}^{r} x_i$
    else
      $v_{[\ell:t]} \leftarrow \sum_{i\in[\ell:t]\setminus\{t^*\}} x_i + p_j$.
      $j \leftarrow j+1$.
    end if
  end for
  Output Steps: Perform Steps 9-11 (the output steps) of Algorithm 4.
end for

Algorithm 6 Mechanism $\mathcal{M}_{\mathsf{gauss}}$

Input: $\mathsf{side} \in \{L, R\}$ (not known to $\mathcal{S}$im).
Output: A natural number.
for $i = 1$ to $\log T + 1$ **do**
  Get records $v_i^{(L)}, v_i^{(R)} \in \{0,1\}^d$ from $\mathcal{S}$im.
  Draw noise from a multivariate Gaussian distribution $Z \sim \mathcal{N}(0, \sigma^2 \mathbb{I}^{d\times d})$, where $\sigma = \sqrt{\frac{d(\log T+1)}{2\rho}}$.
  Output $v_i^{(\mathsf{side})} + Z$
end for

each coordinate of this sum, $sum_t[j] = \sum_{i\in[t]} x_t[j] + \sum_{i\in[|I_t|]} Z_i$ is the sum of at most $\log t + 1$ noisy interval sums. Here, Z_i is a Gaussian random variable with mean 0 and standard deviation $\sigma = \sqrt{\frac{d(\log T+1)}{2\rho}}$, and all Z_is are mutually independent. Hence, by the linearity of expectation, and by the linearity of the variance of independent random variables, we get that $\sum_{i\in[|I_t|]} Z_i$ is a Gaussian random variable with mean 0 and standard deviation $\sqrt{\frac{d(\log T+1)|I_t|}{2\rho}} \le \sqrt{\frac{d(\log T+1)(\log t+1)}{2\rho}}$. Consider the vector N consisting of the absolute values of d random variables independently drawn from the distribution $\mathcal{N}(0,\sigma^2)$ where $\sigma = \sqrt{\frac{d|I_t|(\log T+1)}{2\rho}}$. The distribution of N is identical to the component-wise absolute values of the Gaussian noise vector $sum_t - \sum_{i\in[t]} x_i$. Then,

$$\sum_{i\in[t]} x_i[a_t] + \max_{j\in[d]} N[j] \ge \mathsf{MaxSum}_d(\mathbf{x}_{[t]}) - \max_{j\in[d]} N[j],$$

since if a_t is selected at timestep t, the noisy sum of coordinate a_t at timestep t is larger than the noisy sums of all other coordinates at timestep t (see Step 11 in Algorithm 4). Thus,

$$\begin{aligned} &\mathsf{ERR}_{\mathsf{SumSelect}_d}(\mathbf{x}_{[t]}, a_t) \\ &= \mathsf{MaxSum}_d(\mathbf{x}_{[t]}) - \sum_{i\in[t]} x_i[a_t] \le 2\max_{j\in[d]} N[j]. \end{aligned}$$

Next, we reason about $\max_{j\in[d]} N[j]$ using standard probability tools. Set $\ell = \sqrt{10\frac{d(\log T+1)(\log t+1)\log(dT)}{2\rho}}$. By Lemma F.2 on concentration of the maximum of the absolute values of Gaussian random variables, and since $\sigma \le \sqrt{\frac{d(\log T+1)(\log t+1)}{2\rho}}$, we get that

$$\Pr\left[\max_{j\in[d]} N[j] > \ell\right] \le 2de^{-\frac{\ell^2}{2\sigma^2}} \le 2de^{-5\log(dT)} \le \frac{2}{T^5}.$$

Then, with probability at most $\frac{2}{T^5}$ (over the coins of the algorithm $\mathcal{A}$ and the adversarial process $\mathcal{A}dv$),

$$\begin{aligned} \mathsf{ERR}(\mathbf{x}_{[t]}, a_t) &> 20\sqrt{\frac{d(\log T+1)(\log t+1)\log(dT)}{2\rho}} \\ &\ge 20\sqrt{\frac{d(\log T+1)^2\log(dT)}{2\rho}}, \end{aligned}$$

since $t \le T$. By a union bound over all $t \in [T]$, we get that $\max_{t\in[T]} \mathsf{ERR}_{\mathsf{SumSelect}_d}(\mathbf{x}_{[t]}, a_t) > 20\sqrt{\frac{d(\log T+1)^2\log(dT)}{2\rho}}$ with probability at most $\frac{2}{T^4} \le \frac{1}{3}$ for sufficiently large T. This proves the lemma. □

Proof Sketch of Theorem E.4. The proof of Theorem E.4 for $\mathsf{SumSelect}_d$ closely follows the exposition above. The mechanism used is the same as Algorithm 4, except that in Step 7 , Z is drawn from $Lap(\frac{d(\log T+1)}{\varepsilon})$ instead of a Gaussian distribution. The privacy proof is exactly as in Lemma E.7, except that we use that the composition of $\log T+1$ mechanisms that are $\big(\frac{\varepsilon}{\log T+1}, 0\big)$-DP is $(\varepsilon, 0)$-DP instead a composition theorem for ρ-zCDP. The accuracy proof closely follows that of Lemma E.8, with the main difference being that the the vector N is defined as the component-wise absolute value of d random variables independently drawn from the distribution of the sum of $|I_t|$ independent random variables distributed as $Lap\left(\frac{d(\log(T)+1)}{\varepsilon}\right)$. We then use the concentration inequality for the maximum of the absolute values of independent Laplace random variables over $d|I_t|$ random variables in Lemma F.3 with $a = 2\log T$ to argue that the absolute value of each Laplace random variable is smaller than $\frac{d(\log T+1)}{\varepsilon}(\log(d|I_t|)+10\log T)$ with probability at least $\frac{1}{T^{10}}$. This implies that $\max_{j\in[d]} N[j]$ is smaller than $\frac{d(\log T+1)^2}{\varepsilon}(\log(d(\log T+1))+10\log T)$ with probability at least $\frac{1}{T^{10}}$, upper bounding $|I_t|$ by $\log T+1$. Taking a union bound over T and using the fact that $\log(d(\log T+1)) \leq \log d(\log T+1)$ for sufficiently large T completes the proof. □

Theorems E.1 and E.4 for MaxSum_d are proved analogously. The main difference is that we output $\max_{j\in[d]} sum_t[j]$ instead of $\arg\max_{j\in[d]} sum_t[j]$ in Step 11 of Algorithm 4.

E.3. Algorithms that Recompute at Regular Intervals

In this section, we prove Item 1 of Theorem E.2 for sensitivity-1 functions. The proof of Item 2 of Theorem E.2 builds on the same idea of recomputing $\mathsf{SumSelect}_d$ every T/m timesteps, but it uses the report noisy max (with exponential noise) algorithm for $\mathsf{SumSelect}_d$ (McKenna & Sheldon, 2020) instead of adding Gaussian noise to the function. We omit the details, since the argument is essentially the same as in the rest of this section.

The mechanism recomputes the function every r timesteps. Between recomputations, it outputs the most recently computed value. We select r to balance the privacy cost of composition with the error due to returning stale values between recomputations.

Algorithm 7 Mechanism $\mathcal{M}$ for sensitivity-1 functions in continual release model with adaptively chosen inputs

Input: time horizon T, privacy parameter $\rho > 0$, recompute period $r \in [T-1]$, function f, stream $\mathbf{x} = (x_1, \ldots, x_T) \in \mathcal{X}^n$ where $\mathcal{X} = \{0,1\}^d$.
Output: stream $(a_1, \ldots, a_T) \in \mathbb{R}^T$.
$m \leftarrow \lfloor \frac{T-1}{r} \rfloor$.
for $k = 1$ to m **do**
  Get input record $x_{(k-1)r+1}$.
  Draw $Z_k \sim \mathcal{N}(0, \sigma^2)$, where $\sigma = \sqrt{\frac{m}{2\rho}}$.
  Output $a_{(k-1)r+1} \leftarrow f(\mathbf{x}_{[(k-1)r+1]}) + Z_k$.
  for $t = (k-1)r+2$ to kr **do**
    Get input record x_t.
    Output $a_t \leftarrow a_{(k-1)r+1}$.
  end for
end for

Claim E.9. *For all $\rho, T > 0$, $r \in [T-1]$, mechanism $\mathcal{M}$ defined in Algorithm 7 is ρ-zCDP in the continual release model with adaptively chosen inputs.*

Proof. Consider an adversary $\mathcal{A}dv$ interacting with $\mathcal{M}$. We define a mechanism $\mathcal{M}_{\mathsf{comp}}$, similar to Algorithm 6, and a simulator $\mathcal{S}im$ that interacts with the adversary $\mathcal{A}dv$ such that the view of adversary $\mathcal{A}dv$ in the interaction with $\mathcal{M}_{\mathsf{comp}}$ and $\mathcal{S}im$ is identically distributed to its view in the privacy game $\Pi_{\mathcal{M},\mathcal{A}dv}$, defined in Algorithm 1.

Algorithm 8 Mechanism $\mathcal{M}_{\mathsf{comp}}$

Input: side $\in \{L, R\}$ (not known to $\mathcal{S}im$)
Output: A natural number.
Get neighboring datasets $\mathbf{y}^{(L)}$, $\mathbf{y}^{(R)} \in \{0,1\}^d$ and a function f with ℓ_2 sensitivity at most 1 from $\mathcal{S}im$.
Draw noise $Z \sim \mathcal{N}(0, \sigma^2)$, where $\sigma = \sqrt{\frac{m}{2\rho}}$.
Output $f\left(y^{(\mathsf{side})}\right) + Z$

The mechanism $\mathcal{M}_{\mathsf{comp}}$ is defined in Algorithm 8. Since the function f has ℓ_2 sensitivity at most 1, then by the privacy of the Gaussian mechanism, and since the variance of the noise added is $\frac{m}{2\rho}$), $\mathcal{M}_{\mathsf{comp}}$ is $\frac{\rho}{m}$-zCDP with respect to the dataset consisting of side $\in \{L, R\}$.

The simulator $\mathcal{S}im$ (described in Algorithm 9) gets inputs from $\mathcal{A}dv$, but it does not know the input side $\in \{L, R\}$ that is given to $\mathcal{M}_{\mathsf{comp}}$. It interacts with $\mathcal{M}_{\mathsf{comp}}$ to provide outputs to the adversary $\mathcal{A}dv$. The aim of the Simulator is to mimic the behaviour of $\Pi_{\mathcal{M},\mathcal{A}dv}$ even though it doesn't know side. For all timesteps $t < t^*$ before the challenge timestep, the simulator behaves exactly like $\mathcal{M}$. Starting at the challenge timestep, for every $t \in [t^* : T]$ where $\mathcal{M}$ would recompute the noised value of the sum,

Algorithm 9 Simulator $\mathcal{S}\text{im}$ for the proof of Claim E.9

Input: time horizon T, privacy parameter $\rho > 0$, recompute period $r \in [T-1]$, function f. $\mathcal{S}\text{im}$ also has black-box access to an adversary $\mathcal{A}dv$ and a process $\mathcal{M}_{\mathsf{comp}}$.
Output: stream $(a_1, \ldots, a_T) \in \mathbb{R}^T$
$\mathcal{A}dv$: At each timestep $t \in [T] \setminus \{t^*\}$, $\mathcal{A}dv$ provides $\mathcal{S}\text{im}$ with record $x_t \in \mathcal{X}$. At the challenge timestep t^* (chosen by $\mathcal{A}dv$), it provides two records $x_{t^*}^{(L)}, x_{t^*}^{(R)} \in \mathcal{X}$. At every timestep $t \in [T]$, $\mathcal{S}\text{im}$ provides $\mathcal{A}dv$ with output $a_t \in \mathbb{R}$.
$\mathcal{M}_{\mathsf{comp}}$: $\mathcal{S}\text{im}$ exchanges T/r messages with $\mathcal{M}_{\mathsf{comp}}$.
Initialization: $m \leftarrow \lceil \frac{T}{r} \rceil$, $j \leftarrow 1$.
For timesteps $t < t^*$, run mechanism $\mathcal{M}$ in Algorithm 7 with inputs from $\mathcal{A}dv$, with the same T, ρ, r, f. Let a_t be $\mathcal{M}$'s output at timestep t.
for $t \geq t^*$ **do**
  if $t = t^*$ **then**
    Get input $(x_{t^*}^{(L)}, x_{t^*}^{(R)})$ from $\mathcal{A}dv$.
  end if
  if $t \neq t^*$ **then**
    Get record x_t from $\mathcal{A}dv$.
  end if
  if $t \mod r = 1$ **then**
    For each side $\in \{L, R\}$, let $\mathbf{y}_t^{(\mathsf{side})} = \{x_1, \ldots, x_{t^*-1}, x_{t^*}^{(\mathsf{side})}, x_{t^*+1}, \ldots, x_t\}$;
    $a_t \leftarrow \mathcal{M}_{\mathsf{comp}}\left(f, \mathbf{y}_t^{(L)}, \mathbf{y}_t^{(R)}\right)$.
  else
    $q \leftarrow \lfloor \frac{t}{r} \rfloor$; output $a_t \leftarrow a_{q+1}$.
  end if
end for

$\mathcal{S}\text{im}$ sends $\mathcal{M}_{\mathsf{comp}}$ the function f as well as neighboring datasets $\mathbf{y}_t^{(L)}, \mathbf{y}_t^{(R)}$ defined by

$$\mathbf{y}_t^{(\mathsf{side})} = \{x_1, \ldots, x_{t^*-1}, x_{t^*}^{(\mathsf{side})}, x_{t^*+1}, \ldots, x_t\}.$$

Since $\mathcal{S}\text{im}$ queries $\mathcal{M}_{\mathsf{comp}}$ at most m times, by adaptive composition, the output transcript of $\mathcal{M}_{\mathsf{comp}}$ is ρ-zCDP with respect to the dataset consisting of side.

The view of the adversary $\mathcal{A}dv$ in the real privacy game $\Pi_{\mathcal{M},\mathcal{A}dv}$ is identically distributed to its view in the interaction with $\mathcal{M}_{\mathsf{comp}}$ and $\mathcal{S}\text{im}$. Furthermore, the view of the adversary $\mathcal{A}dv$ when interacting with $\mathcal{S}\text{im}$ and $\mathcal{M}_{\mathsf{comp}}$ is simply a post-processing of the outputs provided to it by $\mathcal{S}\text{im}$, which are a post-processing of the outputs provided to $\mathcal{S}\text{im}$ by $\mathcal{M}_{\mathsf{comp}}$. As argued previously, the output of $\mathcal{M}_{\mathsf{comp}}$ when side $= L$ is ρ-close to its output transcript when side $= R$. Hence we have that $V_{\mathcal{M},\mathcal{A}dv}^{(L)} \simeq_\rho V_{\mathcal{M},\mathcal{A}dv}^{(R)}$. □

Claim E.10. *Fix $\rho > 0$, sufficiently large $T > 0$, and $2 \leq m \leq T$. Let $f : \mathcal{X}^* \to \mathcal{Z}$ be a function with ℓ_2-sensitivity at most 1. Then mechanism $\mathcal{M}$, defined in Algorithm 7 is (α, T)-accurate for f in the continual release model with adaptively chosen inputs where $\alpha = \frac{T}{m} + \sqrt{\frac{10m\log m}{\rho}}$.*

Proof. Consider any adversarial process $\mathcal{A}dv$ interacting with $\mathcal{M}$. Fix a timestep $t \in [T]$. Consider time horizon T divided into m stages, where the stage $k \in [m]$ is from timestep $(k-1)r+1$ to kr. Let timestep t be in stage k. Intuitively, since $\mathcal{M}$, defined in Algorithm 7, corresponds to recomputing the noisy sum every r timesteps (and using each recomputed value for the next r timesteps), the error can be decomposed into two parts: one caused by the drift in the true value of the function since the last recomputation and the other caused by noise addition. By the triangle inequality,

$$\begin{aligned}
&\mathsf{ERR}_f(\mathbf{x}_{[t]}, a_t) = \left|a_t - f(\mathbf{x}_{[t]})\right| \\
&\leq \left|f(\mathbf{x}_{[t]}) - f(\mathbf{x}_{[(k-1)r+1]}\right|) + \left|a_t - f(\mathbf{x}_{[(k-1)r+1]}\right|) \\
&\leq T/m + \left|a_{(k-1)r+1} - f(\mathbf{x}_{[(k-1)r+1]})\right| \leq \frac{T}{m} + |Z_k|.
\end{aligned}$$

The second inequality above holds because the ℓ_2-sensitivity of f is at most 1, and since we recompute every $r = T/m$ timesteps, the maximum change in the function f since the last recomputation is T/m. The third inequality follows from Steps 7 and 10 in Algorithm 7. Finally, observe that Z_k for $k \in [m]$ are mutually independent Gaussian random variables with mean 0 and standard deviation $\sqrt{\frac{m}{2\rho}}$. Hence, applying Lemma F.2 on the concentration of the maximum of the absolute values of Gaussian random variables (setting $\ell = \sqrt{\frac{10m\log m}{\rho}}$), and using the fact that $m \geq 2$,

$$\begin{aligned}
&\Pr_{\mathcal{A},\mathcal{A}dv}\left(\max_{t\in[T]} \mathsf{ERR}_f(\mathbf{x}_{[t]}, a_t) \geq \frac{T}{m} + \sqrt{\frac{10m\log m}{\rho}}\right) \\
&= \Pr_{\mathcal{A},\mathcal{A}dv}\left(\max_{k\in[m]} |Z_k| \geq \sqrt{\frac{10m\log m}{\rho}}\right) \leq \frac{2}{m^9} \leq \frac{1}{3}.
\end{aligned}$$

□

Proof of Item 1 in Theorem E.2. By Claim E.9, the mechanism $\mathcal{M}$ is ρ-zCDP in the continual release model with adaptively chosen inputs.

For $\rho \leq \frac{\log T}{T^2}$, consider the mechanism that doesn't touch the data and always outputs 0. Clearly it is 0-zCDP. Additionally, for this mechanism, $\alpha = O(T)$. For $\rho > \frac{\log T}{T^2}$, by Claim E.10, mechanism $\mathcal{M}$ is (α, T)-accurate for f in the continual release model with adaptively chosen inputs, where $\alpha = T/m + 10\sqrt{\frac{m\log m}{\rho}}$. Setting $m = \lfloor \frac{\rho^{1/3}T^{2/3}}{\log^{1/3} T} \rfloor$ gives $\alpha = \mathrm{O}\left(\min\left\{T, \sqrt[3]{\frac{T\log T}{\rho}}\right\}\right)$, where the min comes from the option of using the trivial mechanism. □

Proof Sketch of Item 1 in Theorem E.5. The mechanism $\mathcal{M}$ used is a variant of Algorithm 7. The only difference is that in Step 6 , instead of the random variable Z_k being distributed as a Gaussian, it is distributed as $Lap(\frac{m}{\varepsilon})$. The privacy proof follows a structure similar to that of Claim E.9, with the main difference being that instead of using a composition theorem for ρ-zCDP, we instead use that the composition of m mechanisms that are $(\frac{\varepsilon}{m}, 0)$-DP is $(\varepsilon, 0)$-DP.

For accuracy, we can prove a claim phrased exactly as Claim E.10, with $\alpha = \frac{T}{m} + \frac{m}{\varepsilon}[\log m + 2\log T]$ instead of $\alpha = \frac{T}{m} + \sqrt{\frac{10m\log m}{\rho}}$. The proof is similar, with the only difference being that instead of using Lemma F.2 on the maximum of i.i.d. Gaussian random variables, we instead use Lemma F.3 on the maximum of i.i.d. Laplace random variables, with $t = 2\log T$.

Finally, we prove the theorem as follows: for $\varepsilon > \frac{\log T}{T}$, setting $m = \lfloor\sqrt{\frac{\varepsilon T}{\log T}}\rfloor$ in the accuracy claim gives $\alpha = O(\sqrt{\frac{T}{\varepsilon}}\log T)$. For $\varepsilon \le \frac{\log T}{T}$, we can consider the mechanism that always outputs 0 at every timestep. This mechanism is $(0, 0)$-DP and (α, T)-accurate for f in the continual release model with adaptively chosen inputs with $\alpha = O(T)$. This completes the proof. □

Proof Sketch of Item 2 in Theorems E.2 and E.5. We sketch the proof of Item 2 of Theorem E.2. The proof of Item 2 of Theorem E.5 is essentially the same. The upper bound mechanism $\mathcal{M}$ used for this proof is a variant of Algorithm 7 where we recompute $\mathsf{SumSelect}_d$ using the exponential mechanism (McSherry & Talwar, 2007) with $\varepsilon' = \sqrt{\frac{2\rho}{m}}$ (for Item 2 of Theorem E.5 on pure DP, we use $\varepsilon' = \frac{\varepsilon}{m}$). The quality function of an attribute and dataset pair is defined to be the sum of that attribute over all entries in the dataset. The exponential mechanism instantiated as described above is used to privately compute $\mathsf{SumSelect}_d$ every T/m timesteps. Between recomputations, the attribute index produced at the last recomputation is used as the output.

The privacy proof follows a structure similar to that of Claim E.9. The main difference for this proof is that the simulator will now interact with an ideal mechanism that takes as input a differentially private algorithm as well as neighboring datasets to run the algorithm on. In particular, the neighboring datasets will be the inputs $x_{t^*}^{(L)}$ and $x_{t^*}^{(R)}$ from the challenge timestep, and the algorithm will be the exponential mechanism hardcoded with all the inputs of the adversary so far (except for the inputs from the challenge timestep.) The ideal mechanism will run the algorithm with challenge input $x_{t^*}^{(\mathsf{side})}$ and output the result. The adversary's view in the privacy game is clearly identical to its view when interacting with the simulator. Finally, the closeness of the adversary's view in the simulated world when $\mathsf{side} = L$ and when $\mathsf{side} = R$ follows directly from the privacy of the exponential mechanism and adaptive composition (Dwork et al., 2010c; Bun & Steinke, 2016).

For accuracy, we prove a claim akin to Claim E.10, with $\alpha = \frac{T}{m} + 2\sqrt{\frac{m}{2\rho}}[\log d + 5\log m]$. The proof is similar to that of Claim E.10; here, we define $|Z_k|$ as the error incurred by the k^{th} instantiation of the exponential mechanism, and use Lemma A.8 on the accuracy of the exponential mechanism (setting $a = 5\log m$) and take a union bound over the m recomputations to argue that the maximum error is greater than $\alpha = \frac{T}{m} + 2\sqrt{\frac{m}{2\rho}}[\log d + 5\log m]$ with probability at most $\frac{1}{m^4}$.

For $\rho > (\frac{\log(dT)}{T})^2$, by the accuracy claim, mechanism $\mathcal{M}$ is (α, T)-accurate for f in the continual release model with adaptively chosen inputs, where $\alpha = \frac{T}{m} + 2\sqrt{\frac{m}{2\rho}}[\log d + 2\log m]$. Setting $m = \lfloor\frac{\rho^{1/3}T^{2/3}}{(\log(dT)^{2/3}}\rfloor$ yields $\alpha = O\left(\frac{T^{1/3}\log(dT)^{2/3}}{\rho^{1/3}}\right)$. Finally, for $\rho \le (\frac{\log(dT)}{T})^2$, consider the mechanism that doesn't touch the data and always outputs 0. It is clearly 0-zCDP, and has $\alpha = O(T)$. □

F. Useful Concentration Inequalities

Lemma F.1. *For all random variables $R \sim \mathcal{N}(0, \sigma^2)$,*

$$\Pr[|R| > \ell] \le 2e^{-\frac{\ell^2}{2\sigma^2}}.$$

Lemma F.2. *Consider m random variables $R_1, \dots, R_m \sim \mathcal{N}(0, \sigma^2)$. Then*

$$\Pr\left[\max_{j\in[m]} |R_j| > \ell\right] \le 2me^{-\frac{\ell^2}{2\sigma^2}}.$$

Proof. By a union bound and Lemma F.1,

$$\Pr\left[\max_{i\in[m]} |R_i| > \ell\right] = \Pr(\exists i \in [m] \text{ such that } |R_i| > \ell) \le \sum_{i=1}^{m} \Pr(|R_i| > \ell) \le \sum_{i=1}^{m} 2e^{-\frac{\ell^2}{2\sigma^2}} = 2me^{-\frac{\ell^2}{2\sigma^2}}.$$

□

A similar union bound argument yields the following concentration inequality on the maximum of the absolute values of i.i.d. Laplace random variables.

Lemma F.3. *Fix $m \in \mathbb{N}$, $\lambda > 0$. Consider m random variables $R_1, \dots, R_m \sim Lap(\lambda)$. Then for all $a > 0$,*

$$\Pr\left(\max_{i\in[m]} |R_i| > \lambda(\log m + \log a)\right) \le e^{-a}.$$