Differentially Private Release and Learning of Threshold Functions

Mark Bun[*] Kobbi Nissim[†] Uri Stemmer[‡] Salil Vadhan[§]

April 28, 2015

Abstract

We prove new upper and lower bounds on the sample complexity of (ε, δ) differentially private algorithms for releasing approximate answers to threshold functions. A threshold function c_x over a totally ordered domain X evaluates to $c_x(y) = 1$ if $y \leq x$, and evaluates to 0 otherwise. We give the first nontrivial lower bound for releasing thresholds with (ε, δ) differential privacy, showing that the task is impossible over an infinite domain X, and moreover requires sample complexity $n \geq \Omega(\log^* |X|)$, which grows with the size of the domain. Inspired by the techniques used to prove this lower bound, we give an algorithm for releasing thresholds with $n \leq 2^{(1+o(1)) \log^* |X|}$ samples. This improves the previous best upper bound of $8^{(1+o(1)) \log^* |X|}$ (Beimel et al., RANDOM '13).

Our sample complexity upper and lower bounds also apply to the tasks of learning distributions with respect to Kolmogorov distance and of properly PAC learning thresholds with differential privacy. The lower bound gives the first separation between the sample complexity of properly learning a concept class with (ε, δ) differential privacy and learning without privacy. For properly learning thresholds in ℓ dimensions, this lower bound extends to $n \geq \Omega(\ell \cdot \log^* |X|)$.

To obtain our results, we give reductions in both directions from releasing and properly learning thresholds and the simpler *interior point problem*. Given a database D of elements from X, the interior point problem asks for an element between the smallest and largest elements in D. We introduce new recursive constructions for bounding the sample complexity of the interior point problem, as well as further reductions and techniques for proving impossibility results for other basic problems in differential privacy.

Keywords: differential privacy, PAC learning, lower bounds, threshold functions, fingerprinting codes

[*]School of Engineering & Applied Sciences, Harvard University. `mbun@seas.harvard.edu`, `http://seas.harvard.edu/~mbun`. Supported by an NDSEG fellowship and NSF grant CNS-1237235.

[†]Dept. of Computer Science, Ben-Gurion University and Harvard University. Work done when K.N. was visiting the Center for Research on Computation & Society, Harvard University. Supported by NSF grant CNS-1237235, a gift from Google, Inc., and a Simons Investigator grant. `kobbi@cs.bgu.ac.il`, `kobbi@seas.harvard.edu`.

[‡]Dept. of Computer Science, Ben-Gurion University. `stemmer@cs.bgu.ac.il`. Supported by the Ministry of Science and Technology (Israel), by the Check Point Institute for Information Security, by the IBM PhD Fellowship Awards Program, and by the Frankel Center for Computer Science.

[§]Center for Research on Computation & Society, School of Engineering & Applied Sciences, Harvard University. `salil@seas.harvard.edu`, `http://seas.harvard.edu/~salil`. Supported by NSF grant CNS-1237235, a gift from Google, Inc., and a Simons Investigator grant.

1 Introduction

The line of work on *differential privacy* [DMNS06] is aimed at enabling useful statistical analyses on privacy-sensitive data while providing strong privacy protections for individual-level information. Privacy is achieved in differentially private algorithms through randomization and the introduction of "noise" to obscure the effect of each individual, and thus differentially private algorithms can be less accurate than their non-private analogues. Nevertheless, by now a rich literature has shown that many data analysis tasks of interest are compatible with differential privacy, and generally the loss in accuracy vanishes as the number n of individuals tends to infinity. However, in many cases, there is still is a price of privacy hidden in these asymptotics — in the rate at which the loss in accuracy vanishes, and in how large n needs to be to start getting accurate results at all (the "sample complexity").

In this paper, we consider the price of privacy for three very basic types of computations involving threshold functions: query release, distribution learning with respect to Kolmogorov distance, and (proper) PAC learning. In all cases, we show for the first time that accomplishing these tasks with differential privacy is *impossible* when the data universe is infinite (e.g. $\mathbb{N}$ or $[0,1]$) and in fact that the sample complexity must grow with the size $|X|$ of the data universe: $n = \Omega(\log^* |X|)$, which is tantalizingly close to the previous upper bound of $n = 2^{O(\log^* |X|)}$ [BNS13b]. We also provide simpler and somewhat improved upper bounds for these problems, reductions between these problems and other natural problems, as well as additional techniques that allow us to prove impossibility results for infinite domains even when the sample complexity does not need to grow with the domain size (e.g. for PAC learning of point functions with "pure" differential privacy).

1.1 Differential Privacy

We recall the definition of differential privacy. We think of a dataset as consisting of n rows from a data universe X, where each row corresponds to one individual. Differential privacy requires that no individual's data has a significant effect on the distribution of what we output.

Definition 1.1. A randomized algorithm $M : X^n \to Y$ is (ε, δ) *differentially private* if for every two datasets $x, x' \in X^n$ that differ on one row, and every set $T \subseteq Y$, we have

$$\Pr[M(x) \in T] \leq e^{\varepsilon} \cdot \Pr[M(x') \in T] + \delta.$$

The original definition from [DMNS06] had $\delta = 0$, and is sometimes referred to as *pure* differential privacy. However, a number of subsequent works have shown that allowing a small (but negligible) value of δ, referred to as *approximate differential privacy*, can provide substantial gains over pure differential privacy [DL09, HT10, DRV10, De12, BNS13b].

The common setting of parameters is to take ε to be a small constant and δ to be negligible in n (or a given security parameter). To simplify the exposition, we fix $\varepsilon = 0.1$ and $\delta = 1/n^{\log n}$ throughout the introduction (but precise dependencies on these parameters are given in the main body).

1.2 Private Query Release

Given a set Q of queries $q : X^n \to \mathbb{R}$, the *query release* problem for Q is to output accurate answers to all queries in Q. That is, we want a differentially private algorithm $M : X^n \to \mathbb{R}^{|Q|}$ such that for every dataset $D \in X^n$, with high probability over $y \leftarrow M(D)$, we have $|y_q - q(D)| \le \alpha$ for all $q \in Q$, for an error parameter α.

A special case of interest is the case where Q consists of *counting queries*. In this case, we are given a set Q of predicates $q : X \to \{0,1\}$ on individual rows, and then extend them to databases by averaging. That is, $q(D) = (1/n)\sum_{i=1}^{D} q(D_i)$ counts the fraction of the population that satisfies predicate q.

The query release problem for counting queries is one of the most widely studied problems in differential privacy. Early work on differential privacy implies that for every family of counting queries Q, the query release problem for Q has sample complexity at most $\tilde{O}(\sqrt{|Q|})$ [DN03, DN04, BDMN05, DMNS06]. That is, there is an $n_0 = \tilde{O}(\sqrt{|Q|})$ such that for all $n \ge n_0$, there is a differentially private mechanism $M : X^n \to \mathbb{R}^Q$ that solves the query release problem for Q with error at most $\alpha = 0.01$. (Again, we treat α as a small constant to avoid an extra parameter in the introduction.)

Remarkably, Blum, Ligett, and Roth [BLR08] showed that if the data universe X is finite, then the sample complexity grows much more slowly with $|Q|$ — indeed the query release problem for Q has sample complexity at most $O((\log|X|)(\log|Q|))$. Hardt and Rothblum [HR10] improved this bound to $\tilde{O}(\log|Q| \cdot \sqrt{\log|X|})$, which was recently shown to be optimal for some families Q [BUV14].

However, for specific query families of interest, the sample complexity can be significantly smaller. In particular, consider the family of *point functions* over X, which is the family $\{q_x\}_{x \in X}$ where $q_x(y)$ is 1 iff $y = x$, and the family of *threshold functions* over X, where $q_x(y)$ is 1 iff $y \le x$ (where X is a totally ordered set). The query release problems for these families correspond to the very natural tasks of producing ℓ_∞ approximations to the histogram and to the cumulative distribution function of the empirical data distribution, respectively. For point functions, Beimel, Nissim, and Stemmer [BNS13b] showed that the sample complexity has no dependence on $|X|$ (or $|Q|$, since $|Q| = |X|$ for these families). In the case of threshold functions, they showed that it has at most a very mild dependence on $|X| = |Q|$, namely $2^{O(\log^*|X|)}$.

Thus, the following basic questions remained open: Are there differentially private algorithms for releasing threshold functions over an infinite data universe (such as $\mathbb{N}$ or $[0,1]$)? If not, does the sample complexity for releasing threshold functions grow with the size $|X|$ of the data universe?

We resolve these questions:

Theorem 1.2. *The sample complexity of releasing threshold functions over a data universe X with differential privacy is at least $\Omega(\log^*|X|)$. In particular, there is no differentially private algorithm for releasing threshold functions over an infinite data universe.*

In addition, inspired by the ideas in our lower bound, we present a simplification of the algorithm of [BNS13b] and improve the sample complexity to $2^{(1+o(1))\log^*|X|}$ (from roughly $8^{\log^*|X|}$). Closing the gap between the lower bound of $\approx \log^*|X|$ and the upper bound of $\approx 2^{\log^*|X|}$ remains an intriguing open problem.

We remark that in the case of pure differential privacy ($\delta = 0$), a sample complexity lower bound of $n = \Omega(\log|X|)$ follows from a standard "packing argument" [HT10, BKN10]. For point functions, this is matched by the standard Laplace mechanism [DMNS06]. For threshold functions, a matching upper bound was recently obtained [RR14], building on a construction of [DNPR10]. We note that these algorithms have a slightly better dependence on the accuracy parameter α than our algorithm (linear rather than nearly linear in $1/\alpha$). In general, while packing arguments often yield tight lower bounds for pure differential privacy, they fail badly for approximate differential privacy, for which much less is known.

There is also a beautiful line of work on characterizing the ℓ_2-accuracy achievable for query release in terms of other measures of the "complexity" of the family Q (such as "hereditary discrepancy") [HT10, BDKT12, MN12, NTZ13]. However, the characterizations given in these works are tight only up to factors of $\mathrm{poly}(\log |X|, \log |Q|)$ and thus do not give good estimates of the sample complexity (which is at most $(\log |X|)(\log |Q|)$ even for pure differential privacy, as mentioned above).

1.3 Private Distribution Learning

A fundamental problem in statistics is *distribution learning*, which is the task of learning an unknown distribution $\mathcal{D}$ given i.i.d. samples from it. The query release problem for threshold functions is closely related to the problem of learning an arbitrary distribution $\mathcal{D}$ on $\mathbb{R}$ up to small error in Kolmogorov (or CDF) distance: Given n i.i.d. samples $x_i \leftarrow_{\mathrm{R}} \mathcal{D}$, the goal of a distribution learner is to produce a CDF $F : X \to [0,1]$ such that $|F(x) - F_{\mathcal{D}}(x)| \leq \alpha$ for all $x \in X$, where α is an accuracy parameter. While closeness in Kolmogorov distance is a relatively weak measure of closeness for distributions, under various structural assumptions (e.g. the two distributions have probability mass functions that cross in a constant number of locations), it implies closeness in the much stronger notion of total variation distance. Other works have developed additional techniques that use weak hypotheses learned under Kolmogorov distance to test and learn distributions under total variation distance (e.g. [DDS+13, DDS14, DK14]).

The well-known Dvoretzky-Kiefer-Wolfowitz inequality [DKW56] implies that without privacy, any distribution over X can be learned to within constant error with $O(1)$ samples. On the other hand, we show that with approximate differential privacy, the task of releasing thresholds is essentially equivalent to distribution learning. As a consequence, with approximate differential privacy, distribution learning instead requires sample complexity that grows with the size of the domain.

Theorem 1.3. *The sample complexity of learning arbitrary distributions on a domain X with differential privacy is at least $\Omega(\log^* |X|)$.*

We prove Theorem 1.3 by showing that the problem of distribution learning with respect to Kolmogorov distance with differential privacy is essentially equivalent to query release for threshold functions. Indeed, query release of threshold functions amounts to approximating the *empirical distribution* of a dataset with respect to Kolmogorov distance. Approximating the empirical distribution is of course trivial without privacy (since we are given it as input), but with privacy, it turns out to have essentially the same sample complexity as the usual distribution learning problem from i.i.d. samples. More generally, query release for a family Q of counting queries is equivalent to distribution learning with respect to the distance measure $d_Q(\mathcal{D}, \mathcal{D}') = \sup_{q \in Q} |\mathbb{E}[q(\mathcal{D})] - \mathbb{E}[q(\mathcal{D}')]|$.

1.4 Private PAC Learning

Kasiviswanathan et al. [KLN+11] defined *private PAC learning* as a combination of probably approximately correct (PAC) learning [Val84] and differential privacy. Recall that a PAC learning algorithm takes some n labeled examples $(x_i, c(x_i)) \in X \times \{0,1\}$ where the x_i's are i.i.d. samples of an arbitrary and unknown distribution on a data universe X and $c : X \to \{0,1\}$ is an unknown *concept* from some concept class C. The goal of the learning algorithm is to output a *hypothesis* $h : X \to \{0,1\}$ that approximates c well on the unknown distribution. We are interested in PAC learning algorithms $L : (X \times \{0,1\})^n \to H$ that are also differentially private. Here H is the *hypothesis class*; if $H \subseteq C$, then L is called a *proper* learner.

As with query release and distribution learning, a natural problem is to characterize the *sample complexity* — the minimum number n of samples in order to achieve differentially private PAC learning for a given concept class C. Without privacy, it is well-known that the sample complexity of (proper) PAC learning is proportional to the Vapnik–Chervonenkis (VC) dimension of the class C [VC71, BEHW89, EHKV89]. In the initial work on differentially private learning, Kasiviswanathan et al. [KLN+11] showed that $O(\log |C|)$ labeled examples suffice for privately learning any concept class C.[1] The VC dimension of a concept class C is always at most $\log |C|$, but is significantly lower for many interesting classes. Hence, the results of [KLN+11] left open the possibility that the sample complexity of private learning may be significantly higher than that of non-private learning.

In the case of *pure* differential privacy ($\delta = 0$), this gap in the sample complexity was shown to be unavoidable in general. Beimel, Kasiviswanathan, and Nissim [BKN10] considered the concept class C of point functions over a data universe X, which have VC dimension 1 and hence can be (properly) learned without privacy with $O(1)$ samples. In contrast, they showed that proper PAC learning with pure differential privacy requires sample complexity $\Omega(\log |X|) = \Omega(\log |C|)$. Feldman and Xiao [FX14] showed a similar separation even for improper learning — the class C of threshold functions over X also has VC dimension 1, but PAC learning with pure differential privacy requires sample complexity $\Omega(\log |X|) = \Omega(\log |C|)$.

For approximate differential privacy ($\delta > 0$), however, it was still open whether there is an asymptotic gap between the sample complexity of private learning and non-private learning. Indeed, Beimel et al. [BNS13b] showed that point functions can be properly learned with approximate differential privacy using $O(1)$ samples (i.e. with no dependence on $|X|$). For threshold functions, they exhibited a proper learner with sample complexity $2^{O(\log^* |X|)}$, but it was conceivable that the sample complexity could also be reduced to $O(1)$.

We prove that the sample complexity of proper PAC learning with approximate differential privacy can be asymptotically larger than the VC dimension:

Theorem 1.4. *The sample complexity of properly learning threshold functions over a data universe X with differential privacy is at least $\Omega(\log^* |X|)$.*

This lower bound extends to the concept class of ℓ-dimensional thresholds. An ℓ-dimensional threshold function, defined over the domain X^ℓ, is a conjunction of ℓ threshold functions, each defined on one component of the domain. This shows that our separation between the sample complexity of private and non-private learning applies to concept classes of every VC dimension.

Theorem 1.5. *For every finite, totally ordered X and $\ell \in \mathbb{N}$, the sample complexity of properly learning the class C of ℓ-dimensional threshold functions on X^ℓ with differential privacy is at least $\Omega(\ell \cdot \log^* |X|) = \Omega(\mathrm{VC}(C) \cdot \log^* |X|)$.*

Based on these results, it would be interesting to fully characterize the difference between the sample complexity of proper non-private learners and of proper learners with (approximate) differential privacy. Furthermore, our results still leave open the possibility that *improper* PAC

[1] As with the query release discussion, we omit the dependency on all parameters except for $|C|$, $|X|$ and $\mathrm{VC}(C)$.

learning with (approximate) differential privacy has sample complexity $O(\mathrm{VC}(C))$. We consider this to be an important question for future work.

We also present a new result on improper learning of point functions with pure differential privacy over infinite countable domains. Beimel et al. [BKN10, BNS13a] showed that for finite data universes X, the sample complexity of improperly learning point functions with pure differential privacy does not grow with $|X|$. They also gave a mechanism for learning point functions over infinite domains (e.g. $X = \mathbb{N}$), but the outputs of their mechanism do not have a finite description length (and hence cannot be implemented by an algorithm). We prove that this is inherent:

Theorem 1.6. *For every infinite domain X, countable hypothesis space H, and $n \in \mathbb{N}$, there is no (even improper) PAC learner $L : (X \times \{0,1\})^n \to H$ for point functions over X that satisfies pure differential privacy.*

1.5 Techniques

Our results for query release and proper learning of threshold functions are obtained by analyzing the sample complexity of a related but simpler problem, which we call the *interior-point problem.* Here we want a mechanism $M : X^n \to X$ (for a totally ordered data universe X) such that for every database $D \in X^n$, with high probability we have $\min_i D_i \leq M(D) \leq \max_i D_i$. We give reductions showing that the sample complexity of this problem is equivalent to the other ones we study:

Theorem 1.7. *Over every totally ordered data universe X, the following four problems have the same sample complexity (up to constant factors) under differential privacy:*

1. *The interior-point problem.*
2. *Query release for threshold functions.*
3. *Distribution learning (with respect to Kolmogorov distance).*
4. *Proper PAC learning of threshold functions.*

Thus we obtain our lower bounds and our simplified and improved upper bounds for query release and proper learning by proving such bounds for the interior-point problem, such as:

Theorem 1.8. *The sample complexity for solving the interior-point problem over a data universe X with differential privacy is $\Omega(\log^* |X|)$.*

Note that for every fixed distribution $\mathcal{D}$ over X there exists a simple differentially private algorithm for solving the interior-point problem (w.h.p.) over databases sampled i.i.d. from $\mathcal{D}$ – simply output a point z s.t. $\Pr_{x\sim\mathcal{D}}[x \geq z] = 1/2$. Hence, in order to prove Theorem 1.8, we show a (correlated) distribution $\mathcal{D}$ over *databases* of size $n \approx \log^* |X|$ on which privately solving the interior-point problem is impossible. The construction is recursive: we use a hard distribution over databases of size $(n-1)$ over a data universe of size logarithmic in $|X|$ to construct the hard distribution over databases of size n over X.

By another reduction to the interior-point problem, we show an impossibility result for the following *undominated-point* problem:

Theorem 1.9. *For every $n \in \mathbb{N}$, there does not exist a differentially private mechanism $M : \mathbb{N}^n \to \mathbb{N}$ with the property that for every dataset $D \in \mathbb{N}^n$, with high probability $M(D) \geq \min_i D_i$.*

Note that for the above problem, one cannot hope to construct a single distribution over databases that every private mechanism fails on. The reason is that for any such distribution $\mathcal{D}$, and any desired failure probability β, there is some number K for which $\Pr_{D\sim\mathcal{D}}[\max D > K] \leq \beta$, and hence that the mechanism that always outputs K solves the problem. Hence, given a mechanism $\mathcal{M}$ we must tailor a hard distribution $\mathcal{D}_{\mathcal{M}}$. We use a similar mechanism-dependent approach to prove Theorem 1.6.

2 Preliminaries

Throughout this work, we use the convention that $[n] = \{0, 1, \ldots, n-1\}$ and write log for $\log_2$. We use $\texttt{THRESH}_X$ to denote the set of all *threshold functions* over a totally ordered domain X. That is,

$$\texttt{THRESH}_X = \{c_x : x \in X\} \quad \text{where} \quad c_x(y) = 1 \text{ iff } y \leq x.$$

2.1 Differential Privacy

Our algorithms and reductions rely on a number of basic results about differential privacy. Early work on differential privacy showed how to solve the query release problem by adding independent Laplace noise to each exact query answer. A real-valued random variable is distributed as $\mathrm{Lap}(b)$ if its probability density function is $f(x) = \frac{1}{2b}\exp(-\frac{|x|}{b})$. We say a function $f : X^n \to \mathbb{R}^m$ has *sensitivity* Δ if for all neighboring $D, D' \in X^n$, it holds that $||f(D) - f(D')||_1 \leq \Delta$.

Theorem 2.1 (The Laplace Mechanism [DMNS06])**.** *Let $f : X^n \to \mathbb{R}^n$ be a sensitivity Δ function. The mechanism A that on input $D \in X^n$ adds independent noise with distribution* $\mathrm{Lap}(\Delta/\epsilon)$ *to each coordinate of $f(D)$ preserves ϵ-differential privacy.*

We will present algorithms that access their input database using (several) differentially private mechanisms and use the following composition theorem to prove their overall privacy guarantee.

Lemma 2.2 (Composition, e.g. [DL09])**.** *Let $\mathcal{M}_1 : X^n \to \mathcal{R}_1$ be $(\varepsilon_1, \delta_1)$-differentially private. Let $\mathcal{M}_2 : X^n \times \mathcal{R}_1 \to \mathcal{R}_2$ be $(\varepsilon_2, \delta_2)$-differentially private for any fixed value of its second argument. Then the composition $M(D) = \mathcal{M}_2(D, \mathcal{M}_1(D))$ is $(\varepsilon_1 + \varepsilon_2, \delta_1 + \delta_2)$-differentially private.*

3 The Interior Point Problem

3.1 Definition

In this work we exhibit a close connection between the problems of privately learning and releasing threshold queries, distribution learning, and solving the *interior point problem* as defined below.

Definition 3.1. An algorithm $A : X^n \to X$ solves the *interior point problem* on X with error probability β if for every $D \in X^n$,

$$\Pr[\min D \leq A(D) \leq \max D] \geq 1 - \beta,$$

where the probability is taken over the coins of A. The sample complexity of the algorithm A is the database size n.

We call a solution x with $\min D \leq x \leq \max D$ an *interior point* of D. Note that x need not be a member of the database D.

3.2 Lower Bound

We now prove our lower bound on the sample complexity of private algorithms for solving the interior point problem.

Theorem 3.2. *Fix any constant $0 < \varepsilon < 1/4$. Let $\delta(n) \le 1/(50n^2)$. Then for every positive integer n, solving the interior point problem on X with probability at least $3/4$ and with $(\varepsilon, \delta(n))$-differential privacy requires sample complexity $n \ge \Omega(\log^* |X|)$.*

Our choice of $\delta = O(1/n^2)$ is unimportant; any monotonically non-increasing convergent series will do. To prove the theorem, we inductively construct a sequence of database distributions $\{\mathcal{D}_n\}$ supported on data universes $[S(n)]$ (for $S(n+1) = 2^{\tilde{O}(S(n))}$) over which any differentially private mechanism using n samples must fail to solve the interior point problem. Given a hard distribution $\mathcal{D}_n$ over n elements $(x_1, x_2, \dots, x_n)$ from $[S(n)]$, we construct a hard distribution $\mathcal{D}_{n+1}$ over elements $(y_0, y_1, \dots, y_n)$ from $[S(n+1)]$ by setting y_0 to be a random number, and letting each other y_i agree with y_0 on the x_i most significant digits. We then show that if y is the output of any differentially private interior point mechanism on $(y_0, \dots, y_n)$, then with high probability, y agrees with y_0 on at least $\min x_i$ entries and at most $\max x_i$ entries. Thus, a private mechanism for solving the interior point problem on $\mathcal{D}_{n+1}$ can be used to construct a private mechanism for $\mathcal{D}_n$, and so $\mathcal{D}_{n+1}$ must also be a hard distribution.

The inductive lemma we prove depends on a number of parameters we now define. Fix $\frac{1}{4} > \varepsilon, \beta > 0$. Let $\delta(n)$ be any positive non-increasing sequence for which

$$P_n \triangleq \frac{e^\varepsilon}{e^\varepsilon + 1} + (e^\varepsilon + 1) \sum_{j=1}^{n} \delta(j) \le 1 - \beta$$

for every n. In particular, it suffices that

$$\sum_{n=1}^{\infty} \delta(n) \le \frac{\frac{1}{3} - \beta}{e^\epsilon + 1}.$$

Let $b(n) = 1/\delta(n)$ and define the function S recursively by

$$S(1) = 2 \quad \text{and} \quad S(n+1) = b(n)^{S(n)}.$$

Lemma 3.3. *For every positive integer n, there exists a distribution $\mathcal{D}_n$ over databases $D \in [S(n)]^n = \{0, 1, \dots, S(n) - 1\}^n$ such that for every $(\varepsilon, \delta(n))$-differentially private mechanism $\mathcal{M}$,*

$$\Pr[\min D \le \mathcal{M}(D) \le \max D] \le P_n,$$

where the probability is taken over $D \leftarrow_R \mathcal{D}_n$ and the coins of $\mathcal{M}$.

In this section, we give a direct proof of the lemma and in Appendix B, we show how the lemma follows from the construction of a new combinatorial object we call an "interior point fingerprinting code." This is a variant on traditional fingerprinting codes, which have been used recently to show nearly optimal lower bounds for other problems in approximate differential privacy [BUV14, DTTZ14, BST14].

Proof. The proof is by induction on n. We first argue that the claim holds for $n = 1$ by letting $\mathcal{D}_1$ be uniform over the singleton databases (0) and (1). To that end let $x \leftarrow_{\mathrm{R}} \mathcal{D}_1$ and note that for any $(\varepsilon, \delta(1))$-differentially private mechanism $\mathcal{M}_0 : \{0,1\} \to \{0,1\}$ it holds that

$$\Pr[\mathcal{M}_0(x) = x] \leq e^\varepsilon \Pr[\mathcal{M}_0(\bar{x}) = x] + \delta(1) = e^\varepsilon(1 - \Pr[\mathcal{M}_0(x) = x]) + \delta(1),$$

giving the desired bound on $\Pr[\mathcal{M}_0(x) = x]$.

Now inductively suppose we have a distribution $\mathcal{D}_n$ that satisfies the claim. We construct a distribution $\mathcal{D}_{n+1}$ on databases $(y_0, y_1, \ldots, y_n) \in [S(n+1)]^{n+1}$ that is sampled as follows:

- Sample $(x_1, \ldots, x_n) \leftarrow_{\mathrm{R}} \mathcal{D}_n$.
- Sample a uniformly random $y_0 \leftarrow_{\mathrm{R}} [S(n+1)]$. We write the base $b(n)$ representation of y_0 as $y_0^{(1)} y_0^{(2)} \ldots y_0^{(S(n))}$.
- For each $i = 1, \ldots, n$ let y_i be a base $b(n)$ number (written $y_i^{(1)} y_i^{(2)} \ldots y_i^{(S(n))}$) that agrees with the base $b(n)$ representation of y_0 on the first x_i digits and contains a random sample from $[b(n)]$ in every index thereafter.

Suppose for the sake of contradiction that there were an $(\varepsilon, \delta(n+1))$-differentially private mechanism $\hat{\mathcal{M}}$ that could solve the interior point problem on $\mathcal{D}_{n+1}$ with probability greater than P_{n+1}. We use $\hat{\mathcal{M}}$ to construct the following private mechanism $\mathcal{M}$ for solving the interior point problem on $\mathcal{D}_n$, giving the desired contradiction:

Algorithm 1 $\mathcal{M}(D)$

```
Input: Database $D = (x_1, \ldots, x_n) \in [S(n)]^n$
1. Construct $\hat{D} = (y_0, \ldots, y_n)$ by sampling from $\mathcal{D}_{n+1}$, but starting with the database $D$. That
   is, sample $y_0$ uniformly at random and set every other $y_i$ to be a random base $b(n)$ string
   that agrees with $y_0$ on the first $x_i$ digits.
2. Compute $y \leftarrow_{\mathrm{R}} \hat{\mathcal{M}}(\hat{D})$.
3. Return the length of the longest prefix of $y$ (in base $b(n)$ notation) that agrees with $y_0$.
```

The mechanism $\mathcal{M}$ is also $(\varepsilon, \delta(n+1))$-differentially private, since for all pairs of adjacent databases $D \sim D'$ and every $T \subseteq [S(n)]$,

$$\begin{aligned}
\Pr[\mathcal{M}(D) \in T] &= \mathop{\mathbb{E}}_{y_0 \leftarrow_{\mathrm{R}} [S(n+1)]} \Pr[\hat{\mathcal{M}}(\hat{D}) \in \hat{T} \mid y_0] \\
&\leq \mathop{\mathbb{E}}_{y_0 \leftarrow_{\mathrm{R}} [S(n+1)]} (e^\varepsilon \Pr[\hat{\mathcal{M}}(\hat{D}') \in \hat{T} \mid y_0] + \delta) \qquad \text{since } \hat{D} \sim \hat{D}' \text{ for fixed } y_0 \\
&= e^\varepsilon \Pr[\mathcal{M}(D') \in T] + \delta,
\end{aligned}$$

where $\hat{T}$ is the set of y that agree with y_0 in exactly the first x entries for some $x \in T$.

Now we argue that $\mathcal{M}$ solves the interior point problem on $\mathcal{D}_n$ with probability greater than P_n. First we show that $x \geq \min D$ with probability greater than P_{n+1}. Observe that by construction, all the elements of $\hat{D}$ agree in at least the first $\min D$ digits, and hence so does any interior point of $\hat{D}$. Therefore, if $\mathcal{M}'$ succeeds in outputting an interior point y of $\hat{D}$, then y must in particular agree with y_0 in at least $\min D$ digits, so $x \geq \min D$.

Now we use the privacy that $\hat{\mathcal{M}}$ provides to y_0 to show that $x \leq \max D$ except with probability at most $e^{\varepsilon}/b(n) + \delta(n+1)$. Fix a database D. Let $w = \max D$, and fix all the randomness of $\mathcal{M}$ but the $(w+1)$st entry of y_0 (note that since $w = \max D$, this fixes $y_1, \ldots, y_n$). Since the $(w+1)$st entry of y_0 is still a uniformly random element of $[b(n)]$, the privately produced entry y^{w+1} should not be able to do much better than randomly guessing $y_0^{(w+1)}$. Formally, for each $z \in [b(n)]$, let $\hat{D}_z$ denote the database $\hat{D}$ with $y_0^{(w+1)}$ set to z and everything else fixed as above. Then by the differential privacy of $\hat{\mathcal{M}}$,

$$\begin{aligned}
\Pr_{z\in[b(n)]}[\hat{\mathcal{M}}(\hat{D}_z)^{w+1} = z] &= \frac{1}{b(n)} \sum_{z\in[b(n)]} \Pr[\hat{\mathcal{M}}(\hat{D}_z)^{w+1} = z] \\
&\leq \frac{1}{b(n)} \sum_{z\in[b(n)]} \mathop{\mathbb{E}}_{z'\leftarrow_{\mathrm{R}}[b(n)]} \left[e^{\varepsilon} \Pr[\hat{\mathcal{M}}(\hat{D}_{z'})^{w+1} = z] + \delta(n+1)\right] \\
&\leq \frac{e^{\varepsilon}}{b(n)} + \delta(n+1),
\end{aligned}$$

where all probabilities are also taken over the coins of $\hat{\mathcal{M}}$. Thus $x \leq \max D$ except with probability at most $e^{\varepsilon}/b(n) + \delta(n+1)$. By a union bound, $\min D \leq x \leq \max D$ with probability greater than

$$P_{n+1} - \left(\frac{e^{\varepsilon}}{b(n)} + \delta(n+1)\right) \geq P_n.$$

This gives the desired contradiction. ☐

We now prove Theorem 3.2 by estimating the $S(n)$ guaranteed by Lemma 3.3.

Proof of Theorem 3.2. Let $S(n)$ be as in Lemma 3.3. We introduce the following notation for iterated exponentials:

$$\mathrm{tower}^{(0)}(x) = x \quad \text{and} \quad \mathrm{tower}^{(k)}(x) = 2^{\mathrm{tower}^{(k-1)}(x)}.$$

Observe that for $k \geq 1$, $x > 0$, and $M > 16$,

$$\begin{aligned}
M^{\mathrm{tower}^{(k)}(x)} &= 2^{\mathrm{tower}^{(k)}(x) \log M} \\
&= \mathrm{tower}^{(2)}(\mathrm{tower}^{(k-1)}(x) + \log\log M) \\
&\leq \mathrm{tower}^{(2)}(\mathrm{tower}^{(k-1)}(x + \log\log M)) \\
&= \mathrm{tower}^{(k+1)}(x + \log\log M).
\end{aligned}$$

By induction on n we get an upper bound of

$$S(n+1) \leq \mathrm{tower}^{(n)}(2 + n \log\log(cn^2)) \leq \mathrm{tower}^{(n+\log^*(cn^2))}(1).$$

This immediately shows that solving the interior point problem on $X = [S(n)]$ requires sample complexity

$$\begin{aligned} n &\geq \log^* S(n) - \log^*(cn^2) \\ &\geq \log^* S(n) - O(\log^* \log^* S(n)) \\ &= \log^* |X| - O(\log^* \log^* |X|). \end{aligned}$$

To get a lower bound for solving the interior point problem on X when $|X|$ is not of the form $S(n)$, note that a mechanism for X is also a mechanism for every X' s.t. $|X'| \leq |X|$. The lower bound follows by setting $|X'| = S(n)$ for the largest n such that $S(n) \leq |X|$. $\square$

3.3 Upper Bound

We now present a recursive algorithm, *RecPrefix*, for privately solving the interior point problem.

Theorem 3.4. *Let $\beta, \epsilon, \delta > 0$, let X be a finite, totally ordered domain, and let $n \in \mathbb{N}$ with $n \geq \frac{18500}{\epsilon} \cdot 2^{\log^* |X|} \cdot \log^*(|X|) \cdot \ln(\frac{4 \log^* |X|}{\beta\epsilon\delta})$. If RecPrefix (defined below) is executed on a database $S \in X^n$ with parameters $\frac{\beta}{3 \log^* |X|}, \frac{\epsilon}{2 \log^* |X|}, \frac{\delta}{2 \log^* |X|}$, then*

1. *RecPrefix is (ϵ, δ)-differentially private;*
2. *With probability at least $(1 - \beta)$, the output x satisfies $\min\{x_i : x_i \in S\} \leq x \leq \max\{x_i : x_i \in S\}$.*

The idea of the algorithm is that on each level of recursion, *RecPrefix* takes an input database S over X and constructs a database S' over a smaller universe X', where $|X'| = \log |X|$, in which every element is the length of the longest prefix of a pair of elements in S (represented in binary). In a sense, this reverses the construction presented in Section 3.2.

3.3.1 The exponential and choosing mechanisms

Before formally presenting the algorithm *RecPrefix*, we introduce several additional algorithmic tools. One primitive we will use is the exponential mechanism of McSherry and Talwar [MT07]. Let X^* denote the set of all finite databases over the universe X. A quality function $q : X^* \times \mathcal{F} \to \mathbb{N}$ defines an *optimization problem* over the domain X and a finite solution set $\mathcal{F}$: Given a database $S \in X^*$, choose $f \in \mathcal{F}$ that (approximately) maximizes $q(S, f)$. The exponential mechanism solves such an optimization problem by choosing a random solution where the probability of outputting any solution f increases exponentially with its quality $q(D, f)$. Specifically, it outputs each $f \in \mathcal{F}$ with probability $\propto \exp(\epsilon \cdot q(S, f)/2\Delta q)$. Here, the sensitivity of a quality function, Δq, is the maximum over all $f \in \mathcal{F}$ of the sensitivity of the function $q(\cdot, f)$.

Proposition 3.5 (Properties of the Exponential Mechanism)**.**

1. *The exponential mechanism is ϵ-differentially private.*
2. *Let q be a quality function with sensitivity at most 1. Fix a database $S \in X^n$ and let $\mathrm{OPT} = \max_{f \in \mathcal{F}}\{q(S, f)\}$. Let $t > 0$. Then exponential mechanism outputs a solution f with $q(S, f) \leq \mathrm{OPT} - tn$ with probability at most $|\mathcal{F}| \cdot \exp(-\epsilon t n/2)$.*

We will also use an (ε, δ)-differentially private variant of the exponential mechanism called the *choosing mechanism*, introduced in [BNS13b].

A quality function with sensitivity at most 1 is of *k-bounded-growth* if adding an element to a database can increase (by 1) the score of at most k solutions, without changing the scores of other solutions. Specifically, it holds that

1. $q(\emptyset, f) = 0$ for all $f \in \mathcal{F}$,
2. If $S_2 = S_1 \cup \{x\}$, then $q(S_1, f) + 1 \geq q(S_2, f) \geq q(S_1, f)$ for all $f \in \mathcal{F}$, and
3. There are at most k values of f for which $q(S_2, f) = q(S_1, f) + 1$.

The choosing mechanism is a differentially private algorithm for approximately solving bounded-growth choice problems. Step 1 of the algorithm checks whether a good solution exists (otherwise any solution is approximately optimal) and Step 2 invokes the exponential mechanism, but with the *small* set $G(S)$ of good solutions instead of $\mathcal{F}$.

```
Algorithm 2 Choosing Mechanism
Input: database $S$, quality function $q$, solution set $\mathcal{F}$, and parameters $\beta, \epsilon, \delta$ and $k$.
  1. Set $\widetilde{\mathrm{OPT}} = \mathrm{OPT} + \mathrm{Lap}(\frac{4}{\epsilon})$. If $\widetilde{\mathrm{OPT}} < \frac{8}{\epsilon}\ln(\frac{4k}{\beta\epsilon\delta})$ then halt and return $\perp$.
  2. Let $G(S) = \{f \in \mathcal{F} : q(S, f) \geq 1\}$. Choose and return $f \in G(S)$ using the exponential
     mechanism with parameter $\frac{\epsilon}{2}$.
```

The following lemmas give the privacy and utility guarantees of the choosing mechanism. We give a slightly improved utility result over [BNS13b], and the analysis is presented in Appendix A.

Lemma 3.6. *Fix $\delta > 0$, and $0 < \epsilon \leq 2$. If q is a k-bounded-growth quality function, then the choosing mechanism is (ϵ, δ)-differentially private.*

Lemma 3.7. *Let the choosing mechanism be executed on a k-bounded-growth quality function, and on a database S s.t. there exists a solution $\hat{f}$ with quality $q(S, \hat{f}) \geq \frac{16}{\epsilon}\ln(\frac{4k}{\beta\epsilon\delta})$. With probability at least $(1 - \beta)$, the choosing mechanism outputs a solution f with quality $q(S, f) \geq 1$.*

Lemma 3.8. *Let the choosing mechanism be executed on a k-bounded-growth quality function, and on a database S containing m elements. With probability at least $(1 - \beta)$, the choosing mechanism outputs a solution f with quality $q(S, f) \geq \mathrm{OPT} - \frac{16}{\epsilon}\ln(\frac{4km}{\beta\epsilon\delta})$.*

3.3.2 The *RecPrefix* algorithm

We are now ready to present and analyze the algorithm *RecPrefix*.

Algorithm 3 *RecPrefix*

```
Input: Database $S = (x_j)_{j=1}^n \in X^n$, parameters $\beta, \epsilon, \delta$.

  1. If $|X| \leq 32$, then use the exponential mechanism with privacy parameter $\epsilon$ and quality
     function $q(S,x) = \min\{\#\{j : x_j \geq x\}, \#\{j : x_j \leq x\}\}$ to choose and return a point $x \in X$.
  2. Let $k = \lfloor \frac{386}{\epsilon} \ln(\frac{4}{\beta\epsilon\delta}) \rfloor$, and let $Y = (y_1, y_2, \dots, y_{n-2k})$ be a random permutation of the
     smallest $(n-2k)$ elements in $S$.
  3. For $j = 1$ to $\frac{n-2k}{2}$, define $z_j$ as the length of the longest prefix for which $y_{2j-1}$ agrees with
     $y_{2j}$ (in base 2 notation).
  4. Execute RecPrefix recursively on $S' = (z_j)_{j=1}^{(n-2k)/2} \in (X')^{(n-2k)/2}$ with parameters $\beta, \epsilon, \delta$.
     Recall that $|X'| = \log|X|$. Denote the returned value by $z$.
  5. Use the choosing mechanism to choose a prefix $L$ of length $(z+1)$ with a large number
     of agreements among elements in $S$. Use parameters $\beta, \epsilon, \delta$, and the quality function $q$ :
     $X^* \times X^{z+1} \to \mathbb{N}$, where $q(S, I)$ is the number of agreements on $I$ among $x_1, \dots, x_n$.
  6. For $\sigma \in \{0,1\}$, define $L_\sigma \in X$ to be the prefix $L$ followed by $(\log|X| - z - 1)$ appearances
     of $\sigma$.
  7. Compute $\widehat{big} = \mathrm{Lap}(\frac{1}{\epsilon}) + \#\{j : x_j \geq L_1\}$.
  8. If $\widehat{big} \geq \frac{3k}{2}$ then return $L_1$. Otherwise return $L_0$.
```

We start the analysis of *RecPrefix* with the following two simple observations.

Observation 3.9. *There are at most* $\log^*|X|$ *recursive calls throughout the execution of RecPrefix on a database* $S \in X^*$.

Observation 3.10. *Let RecPrefix be executed on a database* $S \in X^n$*, where* $n \geq 2^{\log^*|X|} \cdot \frac{2312}{\epsilon} \cdot \ln(\frac{4}{\beta\epsilon\delta})$*. Every recursive call throughout the execution operates on a database containing at least* $\frac{1540}{\epsilon} \cdot \ln(\frac{4}{\beta\epsilon\delta})$ *elements.*

Proof. This follows from Observation 3.9 and from the fact that the i^{th} recursive call is executed on a database of size $n_i = \frac{n}{2^{i-1}} - k\sum_{\ell=0}^{i-2}(\frac{1}{2})^\ell \geq \frac{n}{2^i} - 2k$. □

We now analyze the utility guarantees of *RecPrefix* by proving the following lemma.

Lemma 3.11. *Let* β, ϵ, δ*, and* $S \in X^n$ *be inputs on which RecPrefix performs at most* N *recursive calls, all of which are on databases of at least* $\frac{1540}{\epsilon} \cdot \ln(\frac{4}{\beta\epsilon\delta})$ *elements. With probability at least* $(1 - 3\beta N)$*, the output* x *is s.t.*

1. $\exists x_i \in S$ *s.t.* $x_i \leq x$*;*
2. $|\{i : x_i \geq x\}| \geq k \triangleq \lfloor \frac{386}{\epsilon} \cdot \ln(\frac{4}{\beta\epsilon\delta}) \rfloor$.

Before proving the lemma, we make a combinatorial observation that motivates the random shuffling in Step 2 of *RecPrefix*. A pair of elements $y, y' \in S$ is useful in Algorithm *RecPrefix* if many of the values in S lie between y and y' – a prefix on which y, y' agree is also a prefix of every element between y and y'. A prefix common to a useful pair can hence be identified privately via stability-based techniques. Towards creating useful pairs, the set S is shuffled randomly. We will use the following lemma:

Claim 3.12. *Let $(\Pi_1, \Pi_2, \dots, \Pi_n)$ be a random permutation of $(1, 2, \dots, n)$. Then for all $r \geq 1$,*

$$\Pr\left[\left|\left\{i : |\Pi_{2i-1} - \Pi_{2i}| \leq \frac{r}{12}\right\}\right| \geq r\right] \leq 2^{-r}$$

Proof. We need to show that w.h.p. there are at most r "bad" pairs (Π_{2i-1}, Π_{2i}) within distance $\frac{r}{12}$. For each i, we call Π_{2i-1} the left side of the pair, and Π_{2i} the right side of the pair. Let us first choose r elements to be placed on the left side of r bad pairs (there are $\binom{n}{r}$ such choices). Once those are fixed, there are at most $(\frac{r}{6})^r$ choices for placing elements on the right side of those pairs. Now we have r pairs and $n - 2r$ unpaired elements that can be shuffled in $(n-r)!$ ways. Overall, the probability of having at least r bad pairs is at most

$$\frac{\binom{n}{r}(\frac{r}{6})^r(n-r)!}{n!} = \frac{(\frac{r}{6})^r}{r!} \leq \frac{(\frac{r}{6})^r}{\sqrt{r}r^r e^{-r}} = \frac{e^r}{\sqrt{r}6^r} \leq 2^{-r},$$

where we have used Stirling's approximation for the first inequality. $\square$

Suppose we have paired random elements in our input database S, and constructed a database S' containing lengths of the prefixes for those pairs. Moreover, assume that by recursion we have identified a length z which is the length at least r random pairs. Although those prefixes may be different for each pair, Claim 3.12 guarantees that (w.h.p.) at least one of these prefixes is the prefix of at least $\frac{r}{12}$ input elements. This will help us in (privately) identifying such a prefix.

Proof of Lemma 3.11. The proof is by induction on the number of recursive calls, denoted as t. For $t = 1$ (i.e., $|X| \leq 32$), the claim holds as long as the exponential mechanism outputs an x with $q(S, x) \geq k$ except with probability at most β. By Proposition 3.5, it suffices to have $n \geq \frac{1540}{\epsilon} \cdot \ln(\frac{4}{\beta\epsilon\delta})$, since $32\exp(-\varepsilon(n/2 - k)/2) \leq \beta$.

Assume that the stated lemma holds whenever *RecPrefix* performs at most $t - 1$ recursive calls. Let β, ϵ, δ and $S = (x_i)_{i=1}^n \in X^n$ be inputs on which algorithm *RecPrefix* performs t recursive calls, all of which are on databases containing at least $\frac{1540}{\epsilon} \cdot \ln(\frac{4}{\beta\epsilon\delta})$ elements. Consider the first call in the execution on those inputs, and let $y_1, \dots, y_{n-2k}$ be the random permutation chosen on Step 2. We say that a pair y_{2j-1}, y_{2j} is *close* if

$$\left| i : \begin{array}{c} y_{2j-1} \leq y_i \leq y_{2j} \\ \text{or} \\ y_{2j} \leq y_i \leq y_{2j-1} \end{array} \right| \leq \frac{k-1}{12}.$$

By Claim 3.12, except with probability at most $2^{-(k-1)} < \beta$, there are at most $(k-1)$ close pairs. We continue the proof assuming that this is the case.

Let $S' = (z_i)_{i=1}^{(n-2k)/2}$ be the database constructed in Step 3. By the inductive assumption, with probability at least $(1 - 3\beta(t-1))$, the value z obtained in Step 4 is s.t. (1) $\exists z_i \in S'$ s.t. $z_i \leq z$; and (2) $|\{z_i \in S' : z_i \geq z\}| \geq k$. We proceed with the analysis assuming that this event happened.

By (2), there are at least k pairs y_{2j-1}, y_{2j} that agree on a prefix of length at least z. At least one of those pairs, say y_{2j^*-1}, y_{2j^*}, is not *close*. Note that every y between y_{2j^*-1} and y_{2j^*} agrees on the same prefix of length z, and that there are at least $\frac{k-1}{12}$ such elements in S. Moreover, as the next bit is either 0 or 1, at least half of those elements agree on a prefix of length $(z+1)$. Thus, when using the choosing mechanism on Step 5 (to choose a prefix of length $(z+1)$), there exists at least one prefix with quality at least $\frac{k-1}{24} \geq \frac{16}{\epsilon} \cdot \ln(\frac{4}{\beta\epsilon\delta})$. By Lemma 3.8, the choosing mechanism ensures, therefore, that with probability at least $(1-\beta)$, the chosen prefix L is the prefix of at least one $y_{i'} \in S$, and, hence, this $y_{i'}$ satisfies $L_0 \leq y_{i'} \leq L_1$ (defined in Step 6). We proceed with the analysis assuming that this is the case.

Let $z_{\hat{j}} \in S'$ be s.t. $z_{\hat{j}} \leq z$. By the definition of $z_{\hat{j}}$, this means that $y_{2\hat{j}-1}$ and $y_{2\hat{j}}$ agree on a prefix of length at most z. Hence, as L is of length $z+1$, we have that either $\min\{y_{2\hat{j}-1}, y_{2\hat{j}}\} < L_0$ or $\max\{y_{2\hat{j}-1}, y_{2\hat{j}}\} > L_1$. If $\min\{y_{2\hat{j}-1}, y_{2\hat{j}}\} < L_0$, then L_0 satisfies Condition 1 of being a good output. It also satisfies Condition 2 because $y_{i'} \geq L_0$ and $y_{i'} \in Y$, which we took to be the smallest $n-2k$ elements of S. Similarly, L_1 is a good output if $\max\{y_{2\hat{j}-1}, y_{2\hat{j}}\} > L_1$. In any case, at least one out of L_0, L_1 is a good output.

If both L_0 and L_1 are good outputs, then Step 8 cannot fail. We have already established the existence of $L_0 \leq y_{i'} \leq L_1$. Hence, if L_1 is not a good output, then there are at most $(k-1)$ elements $x_i \in S$ s.t. $x_i \geq L_1$. Hence, the probability of $\widehat{big} \geq 3k/2$ and Step 8 failing is at most $\exp(-\frac{\epsilon k}{2}) \leq \beta$. It remains to analyze the case where L_0 is not a good output (and L_1 is).

If L_0 is not a good output, then every $x_j \in S$ satisfies $x_j > L_0$. In particular, $\min\{y_{2\hat{j}-1}, y_{2\hat{j}}\} > L_0$, and, hence, $\max\{y_{2\hat{j}-1}, y_{2\hat{j}}\} > L_1$. Recall that there are at least $2k$ elements in S which are bigger than $\max\{y_{2\hat{j}-1}, y_{2\hat{j}}\}$. As $k \geq \frac{2}{\epsilon}\ln(\frac{1}{\beta})$, the probability that $\widehat{big} < 3k/2$ and *RecPrefix* fails to return L_1 in this case is at most β.

All in all, *RecPrefix* fails to return an appropriate x with probability at most $3\beta t$. $\square$

We now proceed with the privacy analysis.

Lemma 3.13. *When executed for N recursive calls, RecPrefix is $(2\epsilon N, 2\delta N)$-differentially private.*

Proof. The proof is by induction on the number of recursive calls, denoted by t. For $t=1$ (i.e., $|X| \leq 32$), then by Proposition 3.5 the exponential mechanism ensures that *RecPrefix* is $(\epsilon, 0)$-differentially private. Assume that the stated lemma holds whenever *RecPrefix* performs at most $t-1$ recursive calls, and let $S_1, S_2 \in X^*$ be two neighboring databases on which *RecPrefix* performs t recursive calls.[2] Let $\mathcal{B}$ denote an algorithm consisting of steps 1-4 of *RecPrefix* (the output of $\mathcal{B}$ is the value z from Step 4). Consider the executions of $\mathcal{B}$ on S_1 and on S_2, and denote by Y_1, S'_1 and by Y_2, S'_2 the elements Y, S' as they are in the executions on S_1 and on S_2.

We show that the distributions on the databases S'_1 and S'_2 are similar in the sense that for each database in one of the distributions there exist a neighboring database in the other that have the same probability. Thus, applying the recursion (which is differentially private by the inductive assumption) preserves privacy. We now make this argument formal.

First note that as S_1, S_2 differ in only one element, there is a bijection between orderings Π and $\widehat{\Pi}$ of the smallest $(n-2k)$ elements of S_1 and of S_2 respectively s.t. Y_1 and Y_2 are neighboring databases. This is because there exists a permutation of the smallest $(n-2k)$ elements of S_1 that is a neighbor of the smallest $(n-2k)$ elements of S_2; composition with this fixed permutation yields the desired bijection. Moreover, note that whenever Y_1, Y_2 are neighboring databases, the same is true for S'_1 and S'_2. Hence, for every set of outputs F it holds that

[2]The recursion depth is determined by $|X|$, which is identical in S_1 and in S_2.

$$\begin{aligned}
\Pr[\mathcal{B}(S)\in F] &= \sum_{\Pi}\Pr[\Pi]\cdot\Pr[\mathit{RecPrefix}(S'_1)\in F|\Pi] \\
&\le e^{2\epsilon(t-1)}\cdot\sum_{\Pi}\Pr[\Pi]\cdot\Pr[\mathit{RecPrefix}(S'_2)\in F|\widehat{\Pi}]+2\delta(t-1) \\
&= e^{2\epsilon(t-1)}\cdot\sum_{\widehat{\Pi}}\Pr[\widehat{\Pi}]\cdot\Pr[\mathit{RecPrefix}(S'_2)\in F|\widehat{\Pi}]+2\delta(t-1) \\
&= e^{2\epsilon(t-1)}\cdot\Pr[\mathcal{B}(S')\in F]+2\delta(t-1)
\end{aligned}$$

So when executed for t recursive calls, the sequence of Steps 1-4 of *RecPrefix* is $(2\epsilon(t-1), 2\delta(t-1))$-differentially private. On Steps 5 and 7, algorithm *RecPrefix* interacts with its database through the choosing mechanism and using the Laplace mechanism, each of which is (ϵ,δ)-differentially private. By composition (Lemma 2.2), we get that *RecPrefix* is $(2t\epsilon, 2t\delta)$-differentially private. □

Combining Lemma 3.11 and Lemma 3.13 we obtain Theorem 3.4.

3.3.3 Informal Discussion and Open Questions

An natural open problem is to close the gap between our (roughly) $2^{\log^*|X|}$ upper bound on the sample complexity of privately solving the interior point problem (Theorem 3.4), and our $\log^*|X|$ lower bound (Theorem 3.2). Below we describe an idea for reducing the upper bound to $\mathrm{poly}(\log^*|X|)$.

In our recursive construction for the lower bound, we took n elements $(x_1,\ldots,x_n)$ and generated $n+1$ elements where y_0 is a random element (independent of the x_i's), and every x_i is the length of the longest common prefix of y_0 and y_i. Therefore, a change limited to one x_i affects only one y_i and privacy is preserved (assuming that our future manipulations on $(y_0,\ldots,y_n)$ preserve privacy). While the representation length of domain elements grows exponentially on every step, the database size grows by 1. This resulted in the $\Omega(\log^*|X|)$ lower bound.

In *RecPrefix* on the other hand, every level of recursion shrank the database size by a factor of $\frac{1}{2}$, and hence, we required a sample of (roughly) $2^{\log^*|X|}$ elements. Specifically, in each level of recursion, two input elements y_{2j-1}, y_{2j} were paired and a new element z_j was defined as the length of their longest common prefix. As with the lower bound, we wanted to ensure that a change limited to one of the inputs affects only one new element, and hence, every input element is paired only once, and the database size shrinks.

If we could pair input elements *twice* then the database size would only be reduced additively (which will hopefully result in a $\mathrm{poly}(\log^*|X|)$ upper bound). However, this must be done carefully, as we are at risk of deteriorating the privacy parameter ϵ by a factor of 2 and thus remaining with an exponential dependency in $\log^*|X|$. Consider the following thought experiment for pairing elements.

```
Input: $(x_1,\ldots,x_n)\in X^n$.
  1. Let $(y^0_1,\ldots,y^0_n)$ denote a random permutation of $(x_1,\ldots,x_n)$.
  2. For $t=1$ to $\log^*|X|$:
        For $i=1$ to $(n-t)$, let $y^t_i$ be the length of the longest common prefix of $y^{t-1}_i$
        and $y^{t-1}_{i+1}$.
```

As (most of the) elements are paired twice on every step, the database size reduces additively. In addition, every input element x_i affects at most $t+1$ elements at depth t, and the privacy loss is acceptable. However, this still does not solve the problem. Recall that every iteration of *RecPrefix* begins by randomly shuffling the inputs. Specifically, we needed to ensure that (w.h.p.) the number of "close" pairs is limited. The reason was that if a "not close" pair agrees on a prefix L, then L is the prefix "a lot" of other elements as well, and we could privately identify L. In the above process we randomly shuffled only the elements at depth 0. Thus we do not know if the number of "close" pairs is small at depth $t > 0$. On the other hand, if we changed the pairing procedure to shuffle at every step, then each input element x_i might affect 2^t elements at depth t, causing the privacy loss to deteriorate rapidly.

4 Query Release and Distribution Learning

4.1 Definitions

Recall that a *counting query* q is a predicate $q : X \to \{0,1\}$. For a database $D = (x_1, \dots, x_n) \in X^n$, we write $q(D)$ to denote the average value of q over the rows of D, i.e. $q(D) = \frac{1}{n}\sum_{i=1}^{n} q(x_i)$. In the query release problem, we seek differentially private algorithms that can output approximate answers to a family of counting queries Q simultaneously.

Definition 4.1 (Query Release)**.** Let Q be a collection of counting queries on a data universe X, and let $\alpha, \beta > 0$ be parameters. For a database $D \in X^n$, a sequence of answers $\{a_q\}_{q\in Q}$ is *α-accurate* for Q if $|a_q - q(D)| \le \alpha$ for every $q \in Q$. An algorithm $A : X^n \to \mathbb{R}^{|Q|}$ is (α, β)*-accurate for* Q if for every $D \in X^n$, the output $A(D)$ is α-accurate for Q with probability at least $1 - \beta$ over the coins of A. The sample complexity of the algorithm A is the database size n.

We are interested in the query release problem for the class $\mathtt{THRESH}_X$ of *threshold queries*, which we view as a class of counting queries.

We are also interested in the following *distribution learning* problem, which is very closely related to the query release problem.

Definition 4.2 (Distribution Learning with respect to Q)**.** Let Q be a collection of counting queries on a data universe X. Algorithm A is an (α, β)-accurate *distribution learner with respect to* Q with sample complexity n if for all distributions $\mathcal{D}$ on X, given an input of n samples $D = (x_1, \dots, x_n)$ where each x_i is drawn i.i.d. from $\mathcal{D}$, algorithm A outputs a distribution $\mathcal{D}'$ on X (specified by its PMF) satisfying $d_Q(\mathcal{D}, \mathcal{D}') \triangleq \sup_{q\in Q} |\mathbb{E}_{x\sim\mathcal{D}}[q(x)] - \mathbb{E}_{x\sim\mathcal{D}'}[q(x)]| \le \alpha$ with probability at least $1 - \beta$.

We highlight two important special cases of the distance measure d_Q in the distribution learning problem. First, when Q is the collection of *all* counting queries on a domain X, the distance d_Q is the *total variation distance* between distributions, defined by

$$d_{\mathrm{TV}}(\mathcal{D}, \mathcal{D}') \triangleq \sup_{S\subseteq X} |\Pr_{x\sim\mathcal{D}}[x \in S] - \Pr_{x\sim\mathcal{D}'}[x \in S]|.$$

Second, when X is a totally ordered domain and $Q = \mathtt{THRESH}_X$, the distance d_Q is the *Kolmogorov* (or CDF) distance. A distribution learner in the latter case may as well output a CDF that approximates the target CDF in ℓ_∞ norm. Specifically, we define

Definition 4.3 (Cumulative Distribution Function (CDF))**.** Let $\mathcal{D}$ be a distribution over a totally ordered domain X. The CDF $F_{\mathcal{D}}$ of $\mathcal{D}$ is defined by $F_{\mathcal{D}}(t) = \Pr_{x\sim\mathcal{D}}[x \le t]$. If X is finite, then any function $F : X \to [0,1]$ that is non-decreasing with $F(\max X) = 1$ is a CDF.

Definition 4.4 (Distribution Learning with respect to Kolmogorov distance)**.** Algorithm A is an (α, β)-accurate *distribution learner with respect to Kolmogorov distance* with sample complexity n if for all distributions $\mathcal{D}$ on a totally ordered domain X, given an input of n samples $D = (x_1, \dots, x_n)$ where each x_i is drawn i.i.d. from $\mathcal{D}$, algorithm A outputs a CDF F with $\sup_{x\in X} |F(x) - F_{\mathcal{D}}(x)|$ with probability at least $1 - \beta$.

The query release problem for a collection of counting queries Q is very closely related to the distribution learning problem with respect to Q. In particular, solving the query release problem on a dataset D amounts to learning the empirical distribution of D. Conversely, results in statistical learning theory show that one can solve the distribution learning problem by first solving the query release problem on a sufficiently large random sample, and then fitting a distribution to approximately agree with the released answers. The requisite size of this sample (without privacy considerations) is characterized by a combinatorial measure of the class Q called the VC dimension:

Definition 4.5. Let Q be a collection of queries over domain X. A set $S = \{x_1, \dots, x_k\} \subseteq X$ is *shattered* by Q if for every $T \subseteq [k]$ there exists $q \in Q$ such that $T = \{i : q(x_i) = 1\}$. The *Vapnik-Chervonenkis (VC) dimension* of Q, denoted $\mathrm{VC}(Q)$, is the cardinality of the largest set $S \subseteq X$ that is shattered by Q.

It is known [AB09] that solving the query release problem on $256\,\mathrm{VC}(Q)\ln(48/\alpha\beta)/\alpha^2$ random samples yields an (α, β)-accurate distribution learner for a query class Q.

4.2 Equivalences with the Interior Point Problem

4.2.1 Private Release of Thresholds vs. the Interior Point Problem

We show that the problems of privately releasing thresholds and solving the interior point problem are equivalent.

Theorem 4.6. *Let X be a totally ordered domain. Then,*

1. *If there exists an (ε, δ)-differentially private algorithm that is able to release threshold queries on X with (α, β)-accuracy and sample complexity $n/(8\alpha)$, then there is an (ε, δ)-differentially private algorithm that solves the interior point problem on X with error β and sample complexity n.*

2. *If there exists an (ε, δ)-differentially private algorithm solving the interior point problem on X with error $\alpha\beta/24$ and sample complexity m, then there is a $(5\varepsilon, (1 + e^{\varepsilon})\delta)$-differentially private algorithm for releasing threshold queries with (α, β)-accuracy and sample complexity*

$$n = \max\left\{\frac{6m}{\alpha}, \frac{25\log(24/\beta)\log^{2.5}(6/\alpha)}{\alpha\varepsilon}\right\}.$$

For the first direction, observe that an algorithm for releasing thresholds could easily be used for solving the interior point problem. Formally, *Proof of Theorem 4.6 item 1.* Suppose $\mathcal{A}$ is a private (α, β)-accurate algorithm for releasing thresholds over X for databases of size $\frac{n}{8\alpha}$. Define $\mathcal{A}'$ on databases of size n to pad the database with an equal number of $\min\{X\}$ and $\max\{X\}$ entries, and run $\mathcal{A}$ on the result. We can now return any point t for which the approximate answer to the query c_t is $(\frac{1}{2} \pm \alpha)$ on the (padded) database. □

We now show the converse, i.e., that the problem of releasing thresholds can be reduced to the interior point problem. Specifically, we reduce the problem to a combination of solving the interior point problem, and of releasing thresholds on a much smaller data universe. The latter task is handled by the following algorithm.

Lemma 4.7 ([DNPR10])**.** *For every finite data universe X, and $n \in \mathbb{N}$, $\varepsilon, \beta > 0$, there is an ε-differentially private algorithm A that releases all threshold queries on X with (α, β)-accuracy for*

$$\alpha = \frac{4 \log(1/\beta) \log^{2.5} |X|}{\varepsilon n}.$$

The idea of the reduction is to create noisy partitions of the input database into $O(1/\alpha)$ blocks of size roughly $\alpha n/3$. We then solve the interior point problem on each of these blocks, and think of the results as representatives for each block. By answering threshold queries on just the set of representatives, we can well-approximate all threshold queries. Moreover, since there are only $O(1/\alpha)$ representatives, the base algorithm above gives only $\text{polylog}(1/\alpha)$ error for these answers.

In Appendix C, we describe another reduction that, up to constant factors, gives the same sample complexity.

Proof of Theorem 4.6 item 2. Let $R : X^* \to X$ be an (ε, δ)-differentially private algorithm solving the interior point problem on X with error $\alpha\beta/24$ and sample complexity m. We may actually assume that R is differentially private in the sense that if $D \in X^*$ and D' differs from D up to the addition or removal of a row, then for every $S \subseteq X$, $\Pr[R(D) \in S] \leq e^{\varepsilon} \Pr[R(D') \in S] + \delta$, and that R solves the interior point problem with probability at least $1 - \alpha\beta/24$ whenever its input is of size at least m. This is because we can pad databases of size less than m with an arbitrary fixed element, and subsample the first m entries from any database with size greater than m.

Consider the following algorithm for answering thresholds on databases $D \in X^n$ for $n > m$:

Algorithm 4 *Thresh*(D)

Input: Database $D = (x_1, \dots, x_n) \in X^n$.

```
1. Sort $D$ in nondecreasing order $x_1 \le x_2 \le \cdots \le x_n$.
2. Set $k = 6/\alpha$ and let $t_0 = 1, t_1 = t_0 + \alpha n/3 + \nu_1, t_2 = t_1 + \alpha n/3 + \nu_2 \dots, t_k = t_{k-1} + \alpha n/3 + \nu_k$
   where each $\nu_\ell \sim \mathrm{Lap}(1/\varepsilon)$ independently.
3. Divide $D$ into blocks $D_1, \dots, D_k$, where $D_\ell = (x_{t_{\ell-1}}, \dots, x_{t_\ell - 1})$ (setting $x_j = \max X$ if $j > n$;
   note some $D_\ell$ may be empty).
4. Let $r_0 = \min X$, $r_1 = R(D_1), \dots, r_k = R(D_k)$ and define $\hat{D}$ from $D$ by replacing each $x_j$ with
   the largest $r_\ell$ for which $r_\ell \le x_j$.
5. Run algorithm $A$ from Lemma 4.7 on $\hat{D}$ over the universe $\{r_0, r_1, \dots, r_k\}$ to obtain threshold
   query answers $a_{r_0}, a_{r_1}, \dots, a_{r_k}$. Use privacy parameter $\varepsilon$ and confidence parameter $\beta/4$.
6. Answer arbitrary threshold queries by interpolation, i.e. for $r_\ell \le t < r_{\ell+1}$, set $a_t = a_{r_\ell}$.
7. Output $(a_t)_{t \in X}$.
```

Privacy Let $D = (x_1, \dots, x_n)$ where $x_1 \le x_2 \le \cdots \le x_n$, and consider a neighboring database $D' = (x_1, \dots, x_i', \dots, x_n)$. Assume without loss of generality that $x_i' \ge x_{i+1}$, and suppose

$$x_1 \le \cdots \le x_{i-1} \le x_{i+1} \le \cdots \le x_j \le x_i' \le x_{j+1} \le \cdots \le x_n.$$

We write vectors of noise values as $\nu = (\nu_1, \nu_2, \dots, \nu_k)$. There is a bijection between noise vectors ν and noise vectors ν' such that D partitioned according to ν and D' partitioned according to ν' differ on at most two blocks: namely, if ℓ, r are the indices for which $t_{\ell-1} \le i < t_\ell$ and $t_{r-1} \le j < t_r$ (we may have $\ell = r$), then we can take $\nu_\ell' = \nu_\ell - 1$ and $\nu_r' = \nu_r + 1$ with $\nu' = \nu$ at every other index. Note that D partitioned into $(D_1, \dots, D_k)$ according to ν differs from D' partitioned into $(D_1', \dots, D_k')$ according to ν' by a removal of an element from one block (namely D_ℓ) and the addition of an element to another block (namely D_r'). Thus, for every set $S \subseteq X^m$,

$$\Pr[(R(D_1), \dots, R(D_k)) \in S \mid \nu] \le e^{2\varepsilon} \Pr[(R(D_1'), \dots, R(D_k')) \in S \mid \nu'] + (1 + e^\varepsilon)\delta.$$

Moreover, under the bijection we constructed between ν and ν', noise vector ν' is sampled with density at most $e^{2\varepsilon}$ times the density of ν, so for every set $S \subseteq X^m$,

$$\begin{aligned}\Pr[(R(D_1), \dots, R(D_k)) \in S] &\le e^{2\varepsilon}(e^{2\varepsilon} \Pr[(R(D_1'), \dots, R(D_k')) \in S]) + (1 + e^\varepsilon)\delta \\ &= e^{4\varepsilon} \Pr[(R(D_1'), \dots, R(D_k')) \in S] + (1 + e^\varepsilon)\delta.\end{aligned}$$

Finally, the execution of A at the end of the algorithm is ε-differentially private, so by composition (Lemma 2.2), we obtain the asserted level of privacy.

Utility We can produce α-accurate answers to every threshold function as long as

1. The partitioning exhausts the database, i.e. every element of D is in some D_i,

2. Every execution of R succeeds at finding an interior point,
3. Every database D_i has size at most $5\alpha n/12$ (ensuring that we have error at most $5\alpha/6$ from interpolation),
4. The answers obtained from executing A all have error at most $\alpha/6$.

We now estimate the probabilities of each event. For each i we have $\nu_i \geq -\alpha n/6$ with probability at least

$$1 - \exp(-\alpha n \varepsilon/6) \geq 1 - \alpha\beta/24.$$

So by a union bound, every ν_i is at least $(-\alpha n/6)$ with probability at least $1-\beta/4$. If this is the case, then item 1 holds because $t_k = k \cdot \alpha n/3 + \nu_1 + \cdots + \nu_k \geq (6/\alpha)(\alpha n/3) + (6/\alpha)(-\alpha n/6) \geq n$. Moreover, if every $\nu_i \geq -\alpha n/6$, then item 2 also holds with probability at least $1-\beta/4$. This is because every $|D_i| \geq \alpha n/3 - \alpha n/6 \geq m$, and hence every execution of R on a subdatabase D_i succeeds with probability $1 - \alpha\beta/24$.

By a similar argument, property 3 holds as long as each noise value ν_i is at most $\alpha n/12$, which happens with probability at least $1-\beta/4$. Finally, property 4 holds with probability at least $1-\beta/4$ since

$$\alpha n \geq \frac{24}{\varepsilon} \log(4/\beta) \log^{2.5}(1 + 6/\alpha).$$

A union bound over the four properties completes the proof. □

4.2.2 Releasing Thresholds vs. Distribution Learning

Query release and distribution learning are very similar tasks: A distribution learner can be viewed as an algorithm for query release with small error w.r.t. the underlying distribution (rather than the fixed input database). We show that the two tasks are equivalent under differential privacy.

Theorem 4.8. *Let Q be a collection of counting queries over a domain X.*

1. *If there exists an (ε, δ)-differentially private algorithm for releasing Q with (α, β)-accuracy and sample complexity $n \geq 256\,\mathrm{VC}(Q) \ln(48/\alpha\beta)/\alpha^2$, then there is an (ε, δ)-differentially private $(3\alpha, 2\beta)$-accurate distribution learner w.r.t. Q with sample complexity n.*
2. *If there exists an (ε, δ)-differentially private (α, β)-accurate distribution learner w.r.t. Q with sample complexity n, then there is an (ε, δ)-differentially private query release algorithm for Q with (α, β)-accuracy and sample complexity $9n$.*

The first direction follows from a standard generalization bound, showing that if a given database D contains (enough) i.i.d. samples from a distribution $\mathcal{D}$, then (w.h.p.) accuracy with respect to D implies accuracy with respect to $\mathcal{D}$. We remark that the sample complexity lower bound on n required to apply item 1 of Theorem 4.8 does not substantially restrict its applicability: It is known that an (ε, δ)-differentially private algorithm for releasing Q always requires sample complexity $\Omega(\mathrm{VC}(Q)/\alpha\varepsilon)$ anyway [BLR08].

Proof of Theorem 4.8, item 1. Suppose $\tilde{\mathcal{A}}$ is an (ε, δ)-differentially private algorithm for releasing Q with (α, β)-accuracy and sample complexity $n \geq 256\,\mathrm{VC}(Q) \ln(48/\alpha\beta)/\alpha^2$. Fix a distribution $\mathcal{D}$ over X and consider a database D containing n i.i.d. samples from $\mathcal{D}$. Define the algorithm $\mathcal{A}$ that on input D runs $\tilde{\mathcal{A}}$ on D to obtain answers a_q for every query $q \in Q$. Afterwards, algorithm $\mathcal{A}$

uses linear programming [DNR+09] to construct a distribution $\mathcal{D}'$ that such that $|a_q - q(\mathcal{D}')| \leq \alpha$ for every $q \in Q$, where $q(\mathcal{D}') \triangleq \mathbb{E}_{x \sim \mathcal{D}'}[q(x)]$. This reconstruction always succeeds as long as the answers $\{a_q\}$ are α-accurate, since the empirical distribution of D is a feasible point for the linear program. Note that $\mathcal{A}$ is (ε, δ)-differentially private (since it is obtained by post-processing $\tilde{\mathcal{A}}$).

We first argue that $q(\mathcal{D}')$ is close to $q(D)$ for every $q \in Q$, and then argue that $q(D)$ is close to $q(\mathcal{D})$. By the utility properties of $\tilde{\mathcal{A}}$, with all but probability β,

$$|q(\mathcal{D}') - q(D)| \leq |q(\mathcal{D}') - a_q| + |a_q - q(D)| \leq 2\alpha.$$

for every $q \in Q$.

We now use the following generalization theorem to show that (w.h.p.) $q(D)$ is close to $q(\mathcal{D})$ for every $q \in Q$.

Theorem 4.9 ([AB09])**.** *Let Q be a collection of counting queries over a domain X. Let $D = (x_1, \ldots, x_n)$ consist of i.i.d. samples from a distribution $\mathcal{D}$ over X. If $d = \mathrm{VC}(Q)$, then*

$$\Pr\left[\sup_{q \in Q} |q(D) - q(\mathcal{D})| > \alpha\right] \leq 4\left(\frac{2en}{d}\right)^d \exp\left(-\frac{\alpha^2 n}{8}\right).$$

Using the above theorem, together with the fact that $n \geq 256\,\mathrm{VC}(Q)\ln(48/\alpha\beta)/\alpha^2$, we see that except with probability at least $1-\beta$ we have that $|q(D) - q(\mathcal{D}))| \leq \alpha$ for every $q \in Q$. By a union bound (and the triangle inequality) we get that $\mathcal{A}$ is $(3\alpha, 2\beta)$-accurate. □

In the special case where $Q = \texttt{THRESH}_X$ for a totally ordered domain X, corresponding to distribution learning under Kolmogorov distance, the above theorem holds as long as $n \geq 2\ln(2/\beta)/\alpha^2$. This follows from using the Dvoretzky-Kiefer-Wolfowitz inequality [DKW56, Mas90] in place of Theorem 4.9.

Theorem 4.10. *If there exists an (ε, δ)-differentially private algorithm for releasing $\texttt{THRESH}_X$ over a totally ordered domain X with (α, β)-accuracy and sample complexity $n \geq 2\ln(2/\beta)/\alpha^2$, then there is an (ε, δ)-differentially private $(2\alpha, 2\beta)$-accurate distribution learner under Kolmogorov distance with sample complexity n.*

We now show the other direction of the equivalence.

Lemma 4.11. *Suppose $\mathcal{A}$ is an (ε, δ)-differentially private and (α, β)-accurate distribution learner w.r.t. a concept class Q with sample complexity n. Then there is an (ε, δ)-differentially private algorithm $\tilde{\mathcal{A}}$ for releasing Q with (α, β)-accuracy and sample complexity $9n$.*

To construct the algorithm $\tilde{\mathcal{A}}$, we note that a distribution learner must perform well on the uniform distribution on the rows of any fixed database, and thus must be useful for releasing accurate answers for queries on such a database. Thus if we have a distribution learner $\mathcal{A}$, the mechanism $\tilde{\mathcal{A}}$ that samples m rows (with replacement) from its input database $D \in (X \times \{0,1\})^n$ and runs $\mathcal{A}$ on the result should output accurate answers for queries with respect to D. The random sampling has two competing effects on privacy. On one hand, the possibility that an individual is sampled multiple times incurs additional privacy loss. On the other hand, if $n > m$, then a "secrecy-of-the-sample" argument shows that random sampling actually improves privacy, since any individual is unlikely to have their data affect the computation at all. We show that if n is only a constant factor larger than m, these two effects offset, and the resulting mechanism is still differentially private.

Proof. Consider a database $D \in X^{9n}$. Let $\mathcal{D}$ denote the uniform distribution over the rows of D, and let $\mathcal{D}'$ be the distribution learned. Consider the algorithm $\tilde{\mathcal{A}}$ that subsamples (with replacement) n rows from D and runs $\mathcal{A}$ on it to obtain a distribution $\mathcal{D}'$. Afterwards, algorithm $\tilde{\mathcal{A}}$ answers every threshold query $q \in Q$ with $a_q = q(\mathcal{D}') \triangleq \mathbb{E}_{x \sim \mathcal{D}'}[q(x)]$.

Note that drawing n i.i.d. samples from $\mathcal{D}$ is equivalent to subsampling n rows of D (with replacement). Then with probability at least $1 - \beta$, the distribution $\mathcal{D}'$ returned by $\mathcal{A}$ is such that for every $x \in X$

$$|q(\mathcal{D}') - q(D)| = |q(\mathcal{D}') - q(\mathcal{D})| \leq \alpha,$$

showing that $\tilde{\mathcal{A}}$ is (α, β)-accurate. □

We'll now use a secrecy-of-the-sample argument (refining an argument that appeared implicitly in [KLN+11]), to show that $\tilde{\mathcal{A}}$ (from Lemma 4.11) is differentially private whenever $\mathcal{A}$ is differentially private.

Lemma 4.12. *Fix $\epsilon \leq 1$ and let $\mathcal{A}$ be an (ε, δ)-differentially private algorithm operating on databases of size m. For $n \geq 2m$, construct an algorithm $\tilde{\mathcal{A}}$ that on input a database D of size n subsamples (with replacement) m rows from D and runs $\mathcal{A}$ on the result. Then $\tilde{\mathcal{A}}$ is $(\tilde{\varepsilon}, \tilde{\delta})$-differentially private for*

$$\tilde{\varepsilon} = 6\varepsilon m/n \quad \text{and} \quad \tilde{\delta} = \exp(6\varepsilon m/n)\frac{4m}{n} \cdot \delta.$$

Proof. Let D, D' be adjacent databases of size n, and suppose without loss of generality that they differ on the last row. Let T be a random variable denoting the multiset of indices sampled by $\tilde{\mathcal{A}}$, and let $\ell(T)$ be the multiplicity of index n in T. Fix a subset S of the range of $\tilde{\mathcal{A}}$. For each $k = 0, 1, \ldots, m$ let

$$\begin{aligned} p_k &= \Pr[\ell(T) = k] = \binom{m}{k} n^{-k}(1 - 1/n)^{m-k} = \binom{m}{k}(n-1)^{-k}(1-1/n)^m, \\ q_k &= \Pr[\mathcal{A}(D|_T) \in S | \ell(T) = k], \\ q'_k &= \Pr[\mathcal{A}(D'|_T) \in S | \ell(T) = k]. \end{aligned}$$

Here, $D|_T$ denotes the subsample of D consisting of the indices in T, and similarly for $D'|_T$. Note that $q_0 = q'_0$, since $D|_T = D'|_T$ if index n is not sampled. Our goal is to show that

$$\Pr[\tilde{\mathcal{A}}(D) \in S] = \sum_{k=0}^{m} p_k q_k \leq e^{\tilde{\varepsilon}} \sum_{k=0}^{m} p_k q'_k + \tilde{\delta} = e^{\tilde{\varepsilon}} \Pr[\tilde{\mathcal{A}}(D') \in S] + \tilde{\delta}.$$

To do this, observe that by privacy, $q_k \leq e^{\varepsilon} q_{k-1} + \delta$, so

$$q_k \leq e^{k\varepsilon} q_0 + \frac{e^{k\varepsilon} - 1}{e^{\varepsilon} - 1}\delta.$$

Hence,

$$\begin{aligned}
\Pr[\tilde{\mathcal{A}}(D) \in S] &= \sum_{k=0}^{m} p_k q_k \\
&\le \sum_{k=0}^{m} \binom{m}{k} (n-1)^{-k} (1-1/n)^m \left(e^{k\varepsilon} q_0 + \frac{e^{k\varepsilon}-1}{e^{\varepsilon}-1}\delta \right) \\
&= q_0 (1-1/n)^m \sum_{k=0}^{m} \binom{m}{k} \left(\frac{e^{\varepsilon}}{n-1}\right)^k + \frac{\delta}{e^{\varepsilon}-1}(1-1/n)^m \sum_{k=0}^{m} \binom{m}{k} \left(\frac{e^{\varepsilon}}{n-1}\right)^k - \frac{\delta}{e^{\varepsilon}-1} \\
&= q_0 (1-1/n)^m \left(1 + \frac{e^{\varepsilon}}{n-1}\right)^m + \frac{\delta}{e^{\varepsilon}-1}(1-1/n)^m \left(1 + \frac{e^{\varepsilon}}{n-1}\right)^m - \frac{\delta}{e^{\varepsilon}-1} \\
&= q_0 \left(1 - \frac{1}{n} + \frac{e^{\varepsilon}}{n}\right)^m + \frac{\left(1 - \frac{1}{n} + \frac{e^{\varepsilon}}{n}\right)^m - 1}{e^{\varepsilon}-1}\delta. \qquad (1)
\end{aligned}$$

Similarly, we also have that

$$\Pr[\tilde{\mathcal{A}}(D') \in S] \ge q_0 \left(1 - \frac{1}{n} + \frac{e^{-\varepsilon}}{n}\right)^m - \frac{\left(1 - \frac{1}{n} + \frac{e^{-\varepsilon}}{n}\right)^m - 1}{e^{-\varepsilon}-1}\delta. \qquad (2)$$

Combining inequalities 1 and 2 we get that

$$\Pr[\tilde{\mathcal{A}}(D) \in S] \le \left(\frac{1 - \frac{1}{n} + \frac{e^{\varepsilon}}{n}}{1 - \frac{1}{n} + \frac{e^{-\varepsilon}}{n}}\right)^m \cdot \left\{ \Pr[\tilde{\mathcal{A}}(D') \in S] + \frac{1 - \left(1 - \frac{1}{n} + \frac{e^{-\varepsilon}}{n}\right)^m}{1 - e^{-\varepsilon}}\delta \right\} + \frac{\left(1 - \frac{1}{n} + \frac{e^{\varepsilon}}{n}\right)^m - 1}{e^{\varepsilon}-1}\delta,$$

proving that $\mathcal{A}'$ is $(\tilde{\varepsilon}, \tilde{\delta})$-differentially private for

$$\tilde{\varepsilon} \le m \cdot \ln\left(\frac{1 + \frac{e^{\varepsilon}-1}{n}}{1 + \frac{e^{-\varepsilon}-1}{n}}\right) \le \frac{6\varepsilon m}{n}$$

and

$$\begin{aligned}
\tilde{\delta} &\le \exp(6\varepsilon m/n) \frac{1 - \left(1 + \frac{e^{-\varepsilon}-1}{n}\right)^m}{1 - e^{-\varepsilon}} \cdot \delta + \frac{\left(1 + \frac{e^{\varepsilon}-1}{n}\right)^m - 1}{e^{\varepsilon}-1} \cdot \delta \\
&\le \exp(6\varepsilon m/n) \frac{1 - \exp\left(2\frac{e^{-\varepsilon}-1}{n/m}\right)}{1 - e^{-\varepsilon}} \cdot \delta + \frac{\exp\left(\frac{e^{\varepsilon}-1}{n/m}\right) - 1}{e^{\varepsilon}-1} \cdot \delta \\
&\le \exp(6\varepsilon m/n) \frac{2m}{n} \cdot \delta + \frac{2m}{n} \cdot \delta \\
&\le \exp(6\varepsilon m/n) \frac{4m}{n} \cdot \delta.
\end{aligned}$$

□

5 PAC Learning

5.1 Definitions

A concept $c : X \to \{0,1\}$ is a predicate that labels *examples* taken from the domain X. A *concept class* C over X is a set of concepts over the domain X. A learner is given examples sampled from an unknown probability distribution $\mathcal{D}$ over X that are labeled according to an unknown *target concept* $c \in C$ and outputs a hypothesis h that approximates the target concept with respect to the distribution $\mathcal{D}$. More precisely,

Definition 5.1. The *generalization error* of a hypothesis $h : X \to \{0,1\}$ (with respect to a target concept c and distribution $\mathcal{D}$) is defined by $\mathrm{error}_{\mathcal{D}}(c,h) = \Pr_{x\sim\mathcal{D}}[h(x) \neq c(x)]$. If $\mathrm{error}_{\mathcal{D}}(c,h) \leq \alpha$ we say that h is an *α-good* hypothesis for c on $\mathcal{D}$.

Definition 5.2 (PAC Learning [Val84])**.** Algorithm A is an (α,β)*-accurate PAC learner* for a concept class C over X using hypothesis class H with sample complexity m if for all target concepts $c \in C$ and all distributions $\mathcal{D}$ on X, given an input of m samples $D = ((x_i, c(x_i)), \ldots, (x_m, c(x_m)))$, where each x_i is drawn i.i.d. from $\mathcal{D}$, algorithm A outputs a hypothesis $h \in H$ satisfying $\Pr[\mathrm{error}_{\mathcal{D}}(c,h) \leq \alpha] \geq 1-\beta$.

The probability is taken over the random choice of the examples in D and the coin tosses of the learner A. If $H \subseteq C$ then A is called *proper*, otherwise, it is called *improper*.

Definition 5.3. The *empirical error* of a hypothesis h on a labeled sample $S = ((x_1,\ell_1),\ldots,(x_m,\ell_m))$ is $\mathrm{error}_S(h) = \frac{1}{m}|\{i : h(x_i) \neq \ell_i\}|$. If $\mathrm{error}_S(h) \leq \alpha$ we say h is *α-consistent* with S.

Classical results in statistical learning theory show that a sample of size $\Theta(\mathrm{VC}(C))$ is both necessary and sufficient for PAC learning a concept class C. That $O(\mathrm{VC}(C))$ samples suffice follows from a "generalization" argument: for any concept c and distribution $\mathcal{D}$, with probability at least $1-\beta$ over $m > O_{\alpha,\beta}(\mathrm{VC}(C))$ random labeled examples, *every* concept $h \in C$ that agrees with c on the examples has error at most α on $\mathcal{D}$. Therefore, C can be properly learned by finding any hypothesis $h \in C$ that agrees with the given examples.

Recall the class of *threshold functions*, which are concepts defined over a totally ordered domain X by $\mathtt{THRESH}_X = \{c_x : x \in X\}$ where $c_x(y) = 1$ iff $y \leq x$. The class of threshold functions has VC dimension $\mathrm{VC}(\mathtt{THRESH}_X) = 1$, and hence can be learned with $O_{\alpha,\beta}(1)$ samples.

A private learner is a PAC learner that is differentially private. Following [KLN$^+$11], we consider algorithms $A : (X \times \{0,1\})^m \to H$, where H is a hypothesis class, and require that

1. A is an (α,β)-accurate PAC learner for a concept class C with sample complexity m, and
2. A is (ε,δ)-differentially private.

Note that while we require utility (PAC learning) to hold only when the database D consists of random labeled examples from a distribution, the requirement of differential privacy applies to every pair of neighboring databases $D \sim D'$, including those that do not correspond to examples labeled by any concept.

Recall the relationship between distribution learning and releasing thresholds, where accuracy is measured w.r.t. the underlying distribution in the former and w.r.t. the fixed input database in the later. Analogously, we now define the notion of an *empirical learner* which is similar to a PAC learner where accuracy is measured w.r.t. the fixed input database.

Definition 5.4 (Empirical Learner)**.** Algorithm A is an (α, β)-*accurate empirical learner* for a concept class C over X using hypothesis class H with sample complexity m if for every $c \in C$ and for every database $D = ((x_i, c(x_i)), \dots, (x_m, c(x_m))) \in (X \times \{0,1\})^m$ algorithm A outputs a hypothesis $h \in H$ satisfying $\Pr[\text{error}_D(c, h) \leq \alpha] \geq 1 - \beta$.

The probability is taken over the coin tosses of A.

Note that without privacy (and ignoring computational efficiency) identifying a hypothesis with small empirical error is trivial for every concept class C and for every database of size at least 1. This is not the case with (ε, δ)-differential privacy,[3] and the sample complexity of every empirical learner for a concept class C is at least $\Omega(\text{VC}(C))$:

Theorem 5.5. *For every $\alpha, \beta \leq 1/8$, every $\delta \leq \frac{1}{8n}$ and $\epsilon > 0$, if $\mathcal{A}$ is an (ε, δ)-differentially private (α, β)-accurate empirical learner for a class C with sample complexity n, then $n = \Omega\left(\frac{1}{\alpha\epsilon}\text{VC}(C)\right)$.*

The proof of Theorem 5.5 is very similar the analysis of [BLR08] for lower bounding the sample complexity of releasing approximated answers for queries in the class C. As we will see in the next section, at least in some cases (namely, for threshold functions) the sample complexity must also have some dependency in the size of the domain X.

Proof of Theorem 5.5. Fix $d < \text{VC}(C)$, let $x_0, x_1, x_2, \dots, x_d$ be *shattered* by C, and denote $S = \{x_1, \dots, x_d\}$. Let D denote a database containing $(1 - 8\alpha)n$ copies of x_0 and $8\alpha n/d$ copies of every $x_i \in S$. For a concept c we use D_c to denote the database D labeled by c. We will consider concepts that label x_0 as 0, and label exactly half of the elements in S as 1. To that end, initiate $\tilde{C} = \emptyset$, and for every subset $S' \subseteq S$ of size $|S'| = |S|/2$, add to $\tilde{C}$ one concept $c \in C$ s.t. $c(x_0) = 0$ and for every $x_i \in S$ it holds that $c(x_i) = 1$ iff $x_i \in S'$ (such a concept exists since $S \cup \{x_0\}$ is shattered by C).

Now, let $c \in \tilde{C}$ be chosen uniformly at random, let $x \in S$ be a random element s.t. $c(x) = 1$, and let $y \in S$ be a random element s.t. $c(y) = 0$. Also let $c' \in \tilde{C}$ be s.t. $c'(x) = 0$, $c'(y) = 1$, and $c'(x_i) = c(x_i)$ for every $x_i \in S \setminus \{x, y\}$. Note that the marginal distributions on c and on c' are identical, and denote $h = \mathcal{A}(D_c)$ and $h' = \mathcal{A}(D_{c'})$.

Observe that x is a random element of S that is labeled as 1 in D_c, and that an α-consistent hypothesis for D_c must label at least $(1 - \frac{1}{8})d$ such elements as 1. Hence, by the utility properties of $\mathcal{A}$, we have that

$$\Pr[h(x) = 1] \geq (1 - \beta)(1 - 1/8) \geq 3/4.$$

Similarly, x is a random elements of S that is labeled as 0 in $D_{c'}$, and an α-consistent hypothesis for $D_{c'}$ must not label more than $d/8$ such elements as 1. Hence,

$$\Pr[h'(x) = 1] \leq \beta + (1 - \beta)\frac{1}{8} \leq 1/4.$$

Finally, as D_c and $D_{c'}$ differ in at most $16\alpha n/d$ entries, differential privacy ensures that

$$3/4 \leq \Pr[h(x) = 1] \leq e^{16\alpha\epsilon n/d} \cdot \Pr[h'(x) = 1] + e^{16\alpha\epsilon n/d} \cdot 16\alpha n\delta/d \leq e^{16\alpha\epsilon n/d} \cdot 1/2,$$

showing that $n \geq \frac{d}{40\alpha\epsilon}$. □

[3]The lower bound in Theorem 5.5 also holds for *label private* empirical learners, that are only required to provide privacy for the labels in the database.

5.2 Private Learning of Thresholds vs. the Interior Point Problem

We show that with differential privacy, there is a $\Theta(1/\alpha)$ multiplicative relationship between the sample complexities of properly PAC learning thresholds with (α, β)-accuracy and of solving the interior point problem with error probability $\Theta(\beta)$. Specifically, we show

Theorem 5.6. *Let X be a totally ordered domain. Then,*

1. *If there exists an (ε, δ)-differentially private algorithm solving the interior point problem on X with error probability β and sample complexity n, then there is a $(2\varepsilon, (1+e^{\varepsilon})\delta)$-differentially private $(2\alpha, 2\beta)$-accurate proper PAC learner for $\texttt{THRESH}_X$ with sample complexity $\max\left\{\frac{n}{2\alpha}, \frac{4\log(2/\beta)}{\alpha}\right\}$.*

2. *If there exists an (ε, δ)-differentially private (α, β)-accurate proper PAC learner for $\texttt{THRESH}_X$ with sample complexity n, then there is a $(2\varepsilon, (1+e^{\varepsilon})\delta)$-differentially private algorithm that solves the interior point problem on X with error β and sample complexity $27\alpha n$.*

We show this equivalence in two phases. In the first, we show a $\Theta(1/\alpha)$ relationship between the sample complexity of solving the interior point problem and the sample complexity of empirically learning thresholds. We then use generalization and resampling arguments to show that with privacy, this latter task is equivalent to learning with samples from a distribution.

Lemma 5.7. *Let X be a totally ordered domain. Then,*

1. *If there exists an (ε, δ)-differentially private algorithm solving the interior point problem on X with error probability β and sample complexity n, then there is a $(2\varepsilon, (1+e^{\varepsilon})\delta)$-differentially private algorithm for properly and empirically learning thresholds with (α, β)-accuracy and sample complexity $n/(2\alpha)$.*

2. *If there exists an (ε, δ)-differentially private algorithm that is able to properly and empirically learn thresholds on X with (α, β)-accuracy and sample complexity $n/(3\alpha)$, then there is a $(2\varepsilon, (1+e^{\varepsilon})\delta)$-differentially private algorithm that solves the interior point problem on X with error β and sample complexity n.*

Proof. For the first direction, let $\mathcal{A}$ be a private algorithm for the interior point problem on databases of size n. Consider the algorithm $\mathcal{A}'$ that, on input a database D of size $n/(2\alpha)$, runs $\mathcal{A}'$ on a database D' consisting of the largest $n/2$ elements of D that are labeled 1 and the smallest $n/2$ elements of D that are labeled 0. If there are not enough of either such element, pad D' with $\min\{X\}$'s or $\max\{X\}$'s respectively. Note that if x is an interior point of D' then c_x is a threshold function with error at most $\frac{n/2}{n/(2\alpha)}$ on D, and is hence α-consistent with D. For privacy, note that changing one row of D changes at most two rows of D'. Hence, applying algorithm $\mathcal{A}$ preserves $(2\varepsilon, (e^{\varepsilon}+1)\delta)$-differential privacy.

For the reverse direction, suppose $\mathcal{A}'$ privately finds an α-consistent threshold functions for databases of size $n/(3\alpha)$. Define $\mathcal{A}$ on a database $D' \in X^n$ to label the smaller $n/2$ points 1 and the larger $n/2$ points 0 to obtain a labeled database $D \in (X \times \{0,1\})^n$, pad D with an equal number of $(\min\{X\}, 1)$ and $(\max\{X\}, 0)$ entries to make it of size $n/(3\alpha)$, and run $\mathcal{A}'$ on the result. Note that if c_x is a threshold function with error at most α on D then x is an interior point of D', as otherwise c_x has error at least $\frac{n/2}{n/(3\alpha)} > \alpha$ on D. For privacy, note that changing one row of D' changes at most two rows of D. Hence, applying algorithm $\mathcal{A}'$ preserves $(2\varepsilon, (e^{\varepsilon}+1)\delta)$-differential privacy. $\square$

Now we show that the task of privately outputting an almost consistent hypothesis on any fixed database is essentially equivalent to the task of private (proper) PAC learning. One direction follows immediately from a standard generalization bound for learning thresholds:

Lemma 5.8. *Any algorithm $\mathcal{A}$ for empirically learning* $\texttt{THRESH}_X$ *with (α, β)-accuracy is also a $(2\alpha, \beta + \beta')$-accurate PAC learner for* $\texttt{THRESH}_X$ *when given at least $\max\{n, 4\ln(2/\beta')/\alpha\}$ samples.*

Proof. Let $\mathcal{D}$ be a distribution over a totally ordered domain X and fix a target concept $c = q_x \in \texttt{THRESH}_X$. It suffices to show that for a sample $S = ((x_i, c(x_i)), \dots (x_m, c(x_m)))$ where $m \geq 4\ln(2/\beta')/\alpha$ and the x_i are drawn i.i.d. from $\mathcal{D}$, it holds that

$$\Pr\left[\exists\, h \in C: \;\; \text{error}_{\mathcal{D}}(h, c) > 2\alpha \;\wedge\; \text{error}_S(h) \leq \alpha\right] \leq \beta'.$$

Let $x^- \leq x$ be the largest point with $\text{error}_{\mathcal{D}}(q_{x^-}, c) \geq 2\alpha$. If some $y \leq x$ has $\text{error}_{\mathcal{D}}(q_y, c) \geq 2\alpha$ then $y \leq x^-$, and hence for any sample S, $\text{error}_S(q_{x^-}) \leq \text{error}_S(q_y)$. Similarly let $x^+ \geq x$ be the smallest point with $\text{error}_{\mathcal{D}}(q_{x^+}, c) \geq 2\alpha$. Let $c^- = q_{x^-}$ and $c^+ = q_{x^+}$. Then it suffices to show that

$$\Pr\left[\text{error}_S(c^-) \leq \alpha \vee \text{error}_S(c^+) \leq \alpha\right] \leq \beta'.$$

Concentrating first on c^-, we define the error region $R^- = (x^-, x] \cap X$ as the interval where c^- disagrees with c. By a Chernoff bound, the probability that after m independent samples from $\mathcal{D}$, fewer than αm appear in R^- is at most $\exp(-\alpha m/4) \leq \beta'/2$. The same argument holds for c^+, so the result follows by a union bound. $\square$

In general, an algorithm that can output an α-consistent hypothesis from concept class $\mathcal{C}$ can also be used to learn $\mathcal{C}$ using $\max\{n, 64\,\text{VC}(\mathcal{C})\log(512/\alpha\beta')/\alpha\}$ samples [BEHW89]. The concept class of thresholds has VC dimension 1, so the generalization bound for thresholds saves an $O(\log(1/\alpha))$ factor over the generic statement.

For the other direction, we note that a distribution-free learner must perform well on the uniform distribution on the rows of any fixed database, and thus must be useful for outputting a consistent hypothesis on such a database.

Lemma 5.9. *Suppose $\mathcal{A}$ is an (ϵ, δ)-differentially private (α, β)-accurate PAC learner for a concept class $\mathcal{C}$ with sample complexity m. Then there is an (ϵ, δ)-differentially private (α, β)-accurate empirical learner for $\mathcal{C}$ with sample complexity $n = 9m$. Moreover, if $\mathcal{A}$ is proper, then so is the resulting empirical learner.*

Proof. Consider a database $D = \{(x_i, y_i)\} \in (X \times \{0, 1\})^n$. Let $\mathcal{D}$ denote the uniform distribution over the rows of D. Then drawing m i.i.d. samples from $\mathcal{D}$ is equivalent to subsampling m rows of D (with replacement). Consider the algorithm $\tilde{\mathcal{A}}$ that subsamples (with replacement) m rows from D and runs $\mathcal{A}$ on it. Then with probability at least $1 - \beta$, algorithm $\mathcal{A}$ outputs an α-good hypothesis on $\mathcal{D}$, which is in turn an α-consistent hypothesis for D. Moreover, by Lemma 4.12 (secrecy-of-the-sample), algorithm $\mathcal{A}$ is (ε, δ)-differentially private. $\square$

6 Thresholds in High Dimension

We next show that the bound of $\Omega(\log^* |X|)$ on the sample complexity of private proper-learners for $\texttt{THRESH}_X$ extends to conjunctions of ℓ independent threshold functions in ℓ dimensions. As we will see, every private proper-learner for this class requires a sample of $\Omega(\ell \cdot \log^* |X|)$ elements. This also yields a similar lower bound for the task of query release, as in general an algorithm for query release can be used to construct a private learner.

The significance of this lower bound is twofold. First, for reasonable settings of parameters (e.g. δ is negligible and items in X are of polynomial bit length in n), our $\Omega(\log^* |X|)$ lower bound for threshold functions is dominated by the dependence on $\log(1/\delta)$ in the upper bound. However, $\ell \cdot \log^* |X|$ can still be much larger than $\log(1/\delta)$, even when δ is negligible in the bit length of items in X^ℓ. Second, the lower bound for threshold functions only yields a separation between the sample complexities of private and non-private learning for a class of VC dimension 1. Since the concept class of ℓ-dimensional thresholds has VC dimension of ℓ, we obtain an $\omega(\mathrm{VC}(C))$ lower bound for concept classes even with arbitrarily large VC dimension.

Consider the following extension of $\mathtt{THRESH}_X$ to ℓ dimensions.

Definition 6.1. For a totally ordered set X and $\vec{a} = (a_1, \dots, a_\ell) \in X^\ell$ define the concept $c_{\vec{a}} : X^\ell \to \{0, 1\}$ where $c_{\vec{a}}(\vec{x}) = 1$ if and only if for every $1 \leq i \leq \ell$ it holds that $x_i \leq a_i$. Define the concept class of all thresholds over X^ℓ as $\mathtt{THRESH}_X^\ell = \{c_{\vec{a}}\}_{\vec{a} \in X^\ell}$.

Note that the VC dimension of $\mathtt{THRESH}_X^\ell$ is ℓ. We obtain the following lower bound on the sample complexity of privately learning $\mathtt{THRESH}_X^\ell$.

Theorem 6.2. *For every $n, \ell \in \mathbb{N}$, $\alpha > 0$, and $\delta \leq \ell^2/(1500n^2)$, any $(\varepsilon = \frac{1}{2}, \delta)$-differentially private and $(\alpha, \beta = \frac{1}{8})$-accurate proper learner for $\mathtt{THRESH}_X^\ell$ requires sample complexity $n = \Omega(\frac{\ell}{\alpha} \log^* |X|)$.*

This is the result of a general hardness amplification theorem for private proper learning. We show that if privately learning a concept class C requires sample complexity n, then learning the class C^ℓ of conjunctions of ℓ different concepts from C requires sample complexity $\Omega(\ell n)$.

Definition 6.3. For $\ell \in \mathbb{N}$, a data universe X and a concept class C over X, define a concept class C^ℓ over X^ℓ to consist of all $\vec{c} = (c_1, \dots, c_\ell)$, where $\vec{c} : X^\ell \to \{0, 1\}$ is defined by $\vec{c}(\vec{x}) = c_1(x_1) \wedge c_2(x_2) \wedge \cdots \wedge c_\ell(x_\ell)$.

Theorem 6.4. *Let $\alpha, \beta, \varepsilon, \delta > 0$. Let C be a concept class over a data universe X, and assume there is a domain element $p_1 \in X$ s.t. $c(p_1) = 1$ for every $c \in C$. Let $\mathcal{D}$ be a distribution over databases containing n examples from X labeled by a concept in C, and suppose that every (ε, δ)-differentially private algorithm fails to find an (α/β)-consistent hypothesis $h \in C$ for $D \sim \mathcal{D}$ with probability at least 2β. Then any (ε, δ)-differentially private and (α, β)-accurate proper learner for C^ℓ requires sample complexity $\Omega(\ell n)$.*

Note that in the the above theorem we assumed the existence of a domain element $p_1 \in X$ on which every concept in C evaluates to 1. To justify the necessity of such an assumption, consider the class of *point functions* over a domain X defined as $\mathtt{POINT}_X = \{c_x : x \in X\}$ where $c_x(y) = 1$ iff $y = x$. As was shown in [BNS13b], this class can be privately learned using $O_{\alpha,\beta,\epsilon,\delta}(1)$ labeled examples (i.e., the sample complexity has no dependency in $|X|$). Observe that since there is no $x \in X$ on which every point concept evaluates to 1, we cannot use Theorem 6.4 to lower bound the sample complexity of privately learning $\mathtt{POINT}_X^\ell$. Indeed, the class $\mathtt{POINT}_X^\ell$ is identical (up to renaming of domain elements) to the class $\mathtt{POINT}_{X^\ell}$, and can be privately learned using $O_{\alpha,\beta,\epsilon,\delta}(1)$ labeled examples.

Remark 6.5. *Similarly to Theorem 6.4 it can be shown that if privately learning a concept class C requires sample complexity n, and if there exists a domain element $p_0 \in X$ s.t. $c(p_0) = 0$ for every $c \in C$, then learning the class of* disjunctions *of ℓ concepts from C requires sample complexity ℓn.*

Proof of Theorem 6.4. Assume toward a contradiction that there exists an (ε, δ)-differentially private and (α, β)-accurate proper learner $\mathcal{A}$ for C^ℓ using $\ell n/9$ samples. Recall that the task of privately outputting a good hypothesis on any fixed database is essentially equivalent to the task of private PAC learning (See Section 5.2). We can assume, therefore, that $\mathcal{A}$ outputs an α-consistent hypothesis for every fixed database of size at least $n' \triangleq \ell n$ with probability at least $1 - \beta$.

We construct an algorithm $Solve_\mathcal{D}$ which uses $\mathcal{A}$ in order to find an (α/β)-consistent threshold function for databases of size n from $\mathcal{D}$. Algorithm $Solve_\mathcal{D}$ takes as input a set of n labeled examples in X and applies $\mathcal{A}$ on a database containing n' labeled examples in X^ℓ. The n input points are embedded along one random axis, and random samples from $\mathcal{D}$ are placed on each of the other axes (with n labeled points along each axis).

Algorithm 5 $Solve_\mathcal{D}$

```
Input: Database $D = (x_i, y_i)_{i=1}^n \in (X \times \{0,1\})^n$.
 1. Initiate $S$ as an empty multiset.
 2. Let $r$ be a (uniform) random element from $\{1, 2, \dots, \ell\}$.
 3. For $i = 1$ to $n$, let $\vec{z_i} \in X^\ell$ be the vector with $r^{\text{th}}$ coordinate $x_i$, and all other coordinates
    $p_1$ (recall that every concept in $C$ evaluates to 1 on $p_1$). Add to $S$ the labeled example
    $(\vec{z_i}, y_i)$.
 4. For every axis $t \neq r$:
    (a) Let $D' = (x'_i, y'_i)_{i=1}^n \in (X \times \{0,1\})^n$ denote a (fresh) sample from $\mathcal{D}$.
    (b) For $i = 1$ to $n$, let $\vec{z'_i} \in X^\ell$ be the vector whose $t^{\text{th}}$ coordinate is $x'_i$, and its other
        coordinates are $p_1$. Add to $S$ the labeled example $(\vec{z'_i}, y'_i)$.
 5. Let $(h_1, h_2, \dots, h_\ell) = \vec{h} \leftarrow \mathcal{A}(S)$.
 6. Return $h_r$.
```

First observe that $Solve_\mathcal{D}$ is (ε, δ)-differentially private. To see this, note that a change limited to one input entry affects only one entry of the multiset S. Hence, applying the (ε, δ)-differentially private algorithm $\mathcal{A}$ on S preserves privacy.

Consider the execution of $Solve_\mathcal{D}$ on a database D of size n, sampled from $\mathcal{D}$. We first argue that $\mathcal{A}$ is applied on a multiset S correctly labeled by a concept from C^ℓ. For $1 \leq t \leq \ell$ let $(x_i^t, y_i^t)_{i=1}^n$ be the sample from $\mathcal{D}$ generated for the axis t, let $(\vec{z_i^t}, y_i^t)_{i=1}^n$ denote the corresponding elements that were added to S, and let c_t be s.t. $c_t(x_i^t) = y_i^t$ for every $1 \leq i \leq n$. Now observe that

$$(c_1, c_2, \dots, c_\ell)(\vec{z_i^t}) = c_1(p_1) \wedge c_2(p_1) \wedge \cdots \wedge c_t(x_i^t) \wedge \cdots \wedge c_\ell(p_1) = y_i^t,$$

and hence S is perfectly labeled by $(c_1, c_2, \dots, c_\ell) \in C^\ell$.

By the properties of $\mathcal{A}$, with probability at least $1 - \beta$ we have that $\vec{h}$ (from Step 5) is an α-consistent hypothesis for S. Assuming that this is the case, there could be at most $\beta\ell$ "bad" axes on which $\vec{h}$ errs on more than $\alpha n/\beta$ points. Moreover, as r is a random axis, and as the points along the r^{th} axis are distributed exactly like the points along the other axes, the probability that r is a "bad" axis is at most $\frac{\beta\ell}{\ell} = \beta$. Overall, $Solve_{\mathcal{D}}$ outputs an (α/β)-consistent hypothesis with probability at least $(1-\beta)^2 > 1 - 2\beta$. This contradicts the hardness of the distribution $\mathcal{D}$. □

Now the proof of Theorem 6.2 follows from the lower bound on the sample complexity of privately finding an α-consistent threshold function (see Section 3.2):

Lemma 6.6 (Follows from Lemma 3.3 and 5.7)**.** *There exists a constant $\lambda > 0$ s.t. the following holds. For every totally ordered data universe X there exists a distribution $\mathcal{D}$ over databases containing at most $n = \frac{\lambda}{\alpha}\log^*|X|$ labeled examples from X such that every $(\frac{1}{2}, \frac{1}{50n^2})$-differentially private algorithm fails to find an α-consistent threshold function for $D \sim \mathcal{D}$ with probability at least $\frac{1}{4}$.*

We remark that, in general, an algorithm for query release can be used to construct a private learner with similar sample complexity. Hence, Theorem 6.2 also yields the following lower bound on the sample complexity of releasing approximated answers to queries from $\texttt{THRESH}_X^\ell$.

Theorem 6.7. *For every $n, \ell \in \mathbb{N}$, $\alpha > 0$, and $\delta \leq \ell^2/(7500n^2)$, any $(\frac{1}{150}, \delta)$-differentially private algorithm for releasing approximated answers for queries from $\texttt{THRESH}_X^\ell$ with $(\alpha, \frac{1}{150})$-accuracy must have sample complexity $n = \Omega(\frac{\ell}{\alpha}\log^*|X|)$.*

In order to prove the above theorem we use our lower bound on privately learning $\texttt{THRESH}_X^\ell$ together with the following reduction from private learning to query release.

Lemma 6.8 ([GHRU11, BNS13b])**.** *Let C be a class of predicates. If there exists a $(\frac{1}{150}, \delta)$-differentially private algorithm capable of releasing queries from C with $(\frac{1}{150}, \frac{1}{150})$-accuracy and sample complexity n, then there exists a $(\frac{1}{5}, 5\delta)$-differentially private $(\frac{1}{5}, \frac{1}{5})$-accurate PAC learner for C with sample complexity $O(n)$.*

Proof of Theorem 6.7. Let $\delta \leq \ell^2/(7500n^2)$. Combining our lower bound on the sample complexity of privately learning $\texttt{THRESH}_X^\ell$ (Theorem 6.2) together with the reduction stated in Lemma 6.8, we get a lower bound of $m \triangleq \Omega(\ell \cdot \log^*|X|)$ on the sample complexity of every $(\frac{1}{150}, \delta)$-differentially private algorithm for releasing queries from $\texttt{THRESH}_X^\ell$ with $(\frac{1}{150}, \frac{1}{150})$-accuracy.

In order to refine this argument and get a bound that incorporates the approximation parameter, let $\alpha \leq 1/150$, and assume towards contradiction that there exists a $(\frac{1}{150}, \delta)$-differentially private algorithm $\tilde{\mathcal{A}}$ for releasing queries from $\texttt{THRESH}_X^\ell$ with $(\alpha, \frac{1}{150})$-accuracy and sample complexity $n < m/(150\alpha)$.

We will derive a contradiction by using $\tilde{\mathcal{A}}$ in order to construct a $(\frac{1}{150}, \frac{1}{150})$-accurate algorithm for releasing queries from $\texttt{THRESH}_X^\ell$ with sample complexity less than m. Consider the algorithm $\mathcal{A}$ that on input a database D of size $150\alpha n$, applies $\tilde{\mathcal{A}}$ on a database $\tilde{D}$ containing the elements in D together with $(1 - 150\alpha)n$ copies of $(\min X)$. Afterwards, algorithm $\mathcal{A}$ answers every query $c \in \texttt{THRESH}_X^\ell$ with $a_c \triangleq \frac{1}{150\alpha}(\tilde{a_c} - 1 + 150\alpha)$, where $\{\tilde{a_c}\}$ are the answers received from $\tilde{\mathcal{A}}$.

Note that as $\tilde{\mathcal{A}}$ is $(\frac{1}{150}, \delta)$-differentially private, so is $\mathcal{A}$. We now show that $\mathcal{A}$'s output is $\frac{1}{150}$-accurate for D whenever $\tilde{\mathcal{A}}$'s output is α-accurate for $\tilde{D}$, which happens with all but probability $\frac{1}{150}$. Fix a query $c \in \texttt{THRESH}_X^\ell$ and assume that $c(D) = t/(150\alpha n)$. Note that $c(\min X) = 1$, and hence, $c(\tilde{D}) = t/n + (1 - 150\alpha)$. By the utility properties of $\tilde{\mathcal{A}}$,

$$\begin{aligned} a_c &= \frac{1}{150\alpha}(\tilde{a_c} - 1 + 150\alpha) \\ &\leq \frac{1}{150\alpha}(c(\tilde{D}) + \alpha - 1 + 150\alpha) \\ &= \frac{1}{150\alpha}(t/n + \alpha) \\ &= t/(150\alpha n) + 1/150 \\ &= c(D) + 1/150. \end{aligned}$$

Similar arguments show that $a_c \geq c(D) - 1/150$, proving that $\mathcal{A}$ is $(1/150, 1/150)$-accurate and contradicting the lower bound on the sample complexity of such algorithms. $\square$

7 Mechanism-Dependent Lower Bounds

7.1 The Undominated Point Problem

By a reduction to the interior point problem problem, we can prove an impossibility result for the problem of privately outputting something that is at least the minimum of a database on an unbounded domain. Specifically, we show

Theorem 7.1. *For every (infinite) totally ordered domain X with no maximum element (e.g., $X = \mathbb{N}$) and every $n \in \mathbb{N}$, there is no (ε, δ)-differentially private mechanism $M : X^n \to X$ such that for every $x = (x_1, \ldots, x_n) \in X^n$,*

$$\Pr[M(x) \geq \min_i x_i] \geq 2/3.$$

Besides being a natural relaxation of the interior point problem, this *undominated point problem* is of interest because we require new techniques to obtain lower bounds against it. Note that if we ask for a mechanism that works over a bounded domain (e.g., $[0, 1]$), then the problem is trivial. Moreover, this means that proving a lower bound on the problem when the domain is $\mathbb{N}$ cannot possibly go by way of constructing a single distribution that every differentially private mechanism fails on. The reason is that for any distribution $\mathcal{D}$ over $\mathbb{N}^n$, there is some number K where $\Pr_{D \leftarrow_{\mathrm{R}} \mathcal{D}}[\max D > K] \leq 2/3$, so the mechanism that always outputs K solves the problem.

Proof. Without loss of generality we may take $X = \mathbb{N}$, since every totally ordered domain with no maximum element contains an infinite sequence $x_0 < x_1 < x_2 < x_3 < \ldots$. To prove our lower bound we need to take advantage of the fact that we only need to show that for each differentially private mechanism M there exists a distribution, depending on M, over which M fails. To this end, for an increasing function $T : \mathbb{N} \to \mathbb{N}$, we say that a mechanism $M : \mathbb{N}^n \to \mathbb{N}$ is "T-bounded" if $\Pr[M(x_1, \ldots, x_n) \geq T(\max_i x_i)] < 1/8$. That is, M is T-bounded if it is unlikely to output anything larger than T applied to the max of its input. Note that any mechanism is T-bounded for some function T.

We can then reduce the impossibility of the undominated point problem for T-bounded mechanisms to our lower bound for the interior point problem. First, fix a function T. Suppose for the sake of contradiction that there were a T-bounded mechanism M that solves the undominated point problem on $(x_1, \ldots, x_n)$ with probability at least $7/8$. Then by a union bound, M must output something in the interval $[\min_i x_i, T(\max_i x_i))$ with probability at least $3/4$. Now, for $d \in \mathbb{N}$, consider the data universe $X_d = \{1, T(1), T(T(1)), T(T(T(1))), \ldots, T^{(d-1)}(1)\}$ and the differentially private mechanism $M' : X_d^n \to X_d$ that, on input a database D runs $M(D)$ and rounds the answer down to the nearest $T^i(d)$. Then M' solves the interior point problem on the domain X_d with probability at least $3/4$. By our lower bound for the interior point problem we have $n = \Omega(\log^* d)$, which is a contradiction since n is fixed and d is arbitrary.

□

7.2 Properly Learning Point Functions with Pure Differential Privacy

Using similar ideas as in the above section, we revisit the problem of privately learning the concept class $\texttt{POINT}_{\mathbb{N}}$ of point functions over the natural numbers. Recall that a point function c_x is defined by $c_x(y) = 1$ if $x = y$ and evaluates to 0 otherwise. Beimel et al. [BKN10] used a packing argument to show that $\texttt{POINT}_{\mathbb{N}}$ cannot be properly learned with *pure* ε-differential privacy (i.e., δ=0). However, more recent work of Beimel et al. [BNS13a] exhibited an ε-differentially private *improper* learner for $\texttt{POINT}_{\mathbb{N}}$ with sample complexity $O(1)$. Their construction required an uncountable hypothesis class, with each concept being described by a real number. This left open the question of whether $\texttt{POINT}_{\mathbb{N}}$ could be learned with a countable hypothesis class, with each concept having a finite description length.

We resolve this question in the negative. Specifically, we show that it is impossible to learn (even improperly) point functions over an infinite domain with pure differential privacy using a countable hypothesis class.

Theorem 7.2. *Let X be an infinite domain, let H be a countable collection of hypotheses $\{h : X \to \{0,1\}\}$, and let $\varepsilon \geq 0$. Then there is no ε-differentially private $(1/3, 1/3)$-accurate PAC learner for points over X using the hypothesis class H.*

Remark 7.3. *A learner implemented by an algorithm (i.e. a probabilistic Turing machine) must use a hypothesis class where each hypothesis has a finite description. Note that the standard proper learner for $\texttt{POINT}_X$ can be implemented by an algorithm. However, a consequence of our result is that there is no algorithm for privately learning $\texttt{POINT}_X$.*

Proof. For clarity, and without loss of generality, we assume that $X = \mathbb{N}$. Suppose for the sake of contradiction that we had an ε-differentially private learner M for point functions over $\mathbb{N}$ using hypothesis class H. Since H is countable, there is a finite subset of hypotheses H' such that $M((0,1)^n) \in H'$ with probability at least $5/6$, where $(0,1)^n$ is the dataset where all examples are the point 0 with the label 1. Indeed $\sum_{h \in H} \Pr[M((0,1)^n) = h] = 1$, so some finite partial sum of this series is at least $5/6$. Now to each point $x \in \mathbb{N}$ we will associate a distribution $\mathcal{D}_x$ on $\mathbb{N}$ and let $G_x \subseteq H'$ be the set of hypotheses h *in the finite set* H' for which

$$\Pr_{y \sim \mathcal{D}_x}[c_x(y) = h(y)] \geq 2/3.$$

We establish the following claim.

Claim 7.4. *There is an infinite sequence of points $x_1, x_2, x_3, \ldots$ together with distributions $\mathcal{D}_i := \mathcal{D}_{x_i}$ such that the sets $G_i := G_{x_i}$ are all disjoint.*

Given the claim, the result follows by a packing argument [HT10, BKN10]. By the utility of M, for each $\mathcal{D}_i$ there is a database $R_i \in (\mathbb{N} \times \{0,1\})^n$ in the support of $\mathcal{D}_i^n$ such that $\Pr[M(R_i) \in G_i] \geq 2/3 - 1/6 = 1/2$. By changing the database R_i to $(0,1)^n$ one row at a time while applying the differential privacy constraint, we see that

$$\Pr[M((0,1)^n) \in G_i] \geq \frac{1}{2} e^{-\varepsilon n}.$$

It is impossible for this to hold for infinitely many disjoint sets G_i. □

Proof of Claim 7.4. We inductively construct the sequence (x_i), starting with $x_1 = 0$. Now suppose we have constructed $x_1, \dots, x_i$ with corresponding good hypothesis sets $G_1, \dots, G_i$. Let $B = \cup_{j=1}^{i} G_i$ be the set of hypotheses with wish to avoid. Note that B is a finite set of hypotheses, so there are some $x, x' \in \mathbb{N}$ for which every $h \in B$ with $h(x) = 1$ also has $h(x') = 1$. Let $x_{i+1} = x$ and $\mathcal{D}_i$ be distributed uniformly over x and x'. Then for every hypothesis $h \in B$,

$$\Pr_{y \sim D_i}[c_{x_{i+1}}(y) = h(y)] \leq 1/2,$$

and hence G_{i+1} is disjoint from the preceding G_j's. □

Acknowledgments. We thank Amos Beimel, Adam Smith, Jonathan Ullman, and anonymous reviewers for helpful conversations and suggestions that helped guide our work. We also thank Gautam Kamath for pointing us to references on distribution learning.

References

[AB09] Martin Anthony and Peter L. Bartlett. *Neural Network Learning: Theoretical Foundations.* Cambridge University Press, New York, NY, USA, 1st edition, 2009.

[BDKT12] Aditya Bhaskara, Daniel Dadush, Ravishankar Krishnaswamy, and Kunal Talwar. Unconditional differentially private mechanisms for linear queries. In *Proceedings of the Forty-fourth Annual ACM Symposium on Theory of Computing*, STOC '12, pages 1269–1284, New York, NY, USA, 2012. ACM.

[BDMN05] Avrim Blum, Cynthia Dwork, Frank McSherry, and Kobbi Nissim. Practical privacy: the SuLQ framework. In Chen Li, editor, *PODS*, pages 128–138. ACM, 2005.

[BEHW89] Anselm Blumer, A. Ehrenfeucht, David Haussler, and Manfred K. Warmuth. Learnability and the Vapnik-Chervonenkis dimension. *J. ACM*, 36(4):929–965, October 1989.

[BKN10] Amos Beimel, Shiva Prasad Kasiviswanathan, and Kobbi Nissim. Bounds on the sample complexity for private learning and private data release. In *TCC*, pages 437–454, 2010.

[BLR08] Avrim Blum, Katrina Ligett, and Aaron Roth. A learning theory approach to non-interactive database privacy. In Cynthia Dwork, editor, *STOC*, pages 609–618. ACM, 2008.

[BNS13a] Amos Beimel, Kobbi Nissim, and Uri Stemmer. Characterizing the sample complexity of private learners. In Robert D. Kleinberg, editor, *ITCS*, pages 97–110. ACM, 2013.

[BNS13b] Amos Beimel, Kobbi Nissim, and Uri Stemmer. Private learning and sanitization: Pure vs. approximate differential privacy. In Prasad Raghavendra, Sofya Raskhodnikova, Klaus Jansen, and José D. P. Rolim, editors, *APPROX-RANDOM*, volume 8096 of *Lecture Notes in Computer Science*, pages 363–378. Springer, 2013.

[BS98] Dan Boneh and James Shaw. Collusion-secure fingerprinting for digital data. *IEEE Transactions on Information Theory*, 44(5):1897–1905, 1998.

[BST14] Raef Bassily, Adam Smith, and Abhradeep Thakurta. Private empirical risk minimization, revisited. *CoRR*, abs/1405.7085, 2014.

[BUV14] Mark Bun, Jonathan Ullman, and Salil P. Vadhan. Fingerprinting codes and the price of approximate differential privacy. In *Symposium on Theory of Computing, STOC 2014, New York, NY, USA, May 31 - June 03, 2014*, pages 1–10, 2014.

[DDS$^+$13] Constantinos Daskalakis, Ilias Diakonikolas, Rocco A. Servedio, Gregory Valiant, and Paul Valiant. Testing k-modal distributions: Optimal algorithms via reductions. In *Proceedings of the Twenty-Fourth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2013, New Orleans, Louisiana, USA, January 6-8, 2013*, pages 1833–1852, 2013.

[DDS14] Constantinos Daskalakis, Ilias Diakonikolas, and Rocco A. Servedio. Learning k-modal distributions via testing. *Theory of Computing*, 10:535–570, 2014.

[De12] Anindya De. Lower bounds in differential privacy. In *TCC*, pages 321–338, 2012.

[DK14] Constantinos Daskalakis and Gautam Kamath. Faster and sample near-optimal algorithms for proper learning mixtures of gaussians. In *Proceedings of The 27th Conference on Learning Theory, COLT 2014, Barcelona, Spain, June 13-15, 2014*, pages 1183–1213, 2014.

[DKW56] A. Dvoretzky, J. Kiefer, and J. Wolfowitz. Asymptotic minimax character of the sample distribution function and of the classical multinomial estimator. *Ann. Math. Statist.*, 27(3):642–669, 09 1956.

[DL09] Cynthia Dwork and Jing Lei. Differential privacy and robust statistics. In *Proceedings of the Forty-first Annual ACM Symposium on Theory of Computing*, STOC '09, pages 371–380, New York, NY, USA, 2009. ACM.

[DMNS06] Cynthia Dwork, Frank McSherry, Kobbi Nissim, and Adam Smith. Calibrating noise to sensitivity in private data analysis. In Shai Halevi and Tal Rabin, editors, *TCC*, volume 3876 of *Lecture Notes in Computer Science*, pages 265–284. Springer, 2006.

[DN03] Irit Dinur and Kobbi Nissim. Revealing information while preserving privacy. In *PODS*, pages 202–210. ACM, 2003.

[DN04] Cynthia Dwork and Kobbi Nissim. Privacy-preserving datamining on vertically partitioned databases. In Matthew K. Franklin, editor, *CRYPTO*, volume 3152 of *Lecture Notes in Computer Science*, pages 528–544. Springer, 2004.

[DNPR10] Cynthia Dwork, Moni Naor, Toniann Pitassi, and Guy N. Rothblum. Differential privacy under continual observation. In *STOC*, pages 715–724, 2010.

[DNR$^+$09] Cynthia Dwork, Moni Naor, Omer Reingold, Guy N. Rothblum, and Salil P. Vadhan. On the complexity of differentially private data release: efficient algorithms and hardness results. In Michael Mitzenmacher, editor, *STOC*, pages 381–390. ACM, 2009.

[DRV10] Cynthia Dwork, Guy N. Rothblum, and Salil P. Vadhan. Boosting and differential privacy. In *FOCS*, pages 51–60. IEEE Computer Society, 2010.

[DTTZ14] Cynthia Dwork, Kunal Talwar, Abhradeep Thakurta, and Li Zhang. Analyze gauss: Optimal bounds for privacy-preserving principal component analysis. In *Proceedings of the 46th Annual ACM Symposium on Theory of Computing*, STOC '14, pages 11–20, New York, NY, USA, 2014. ACM.

[EHKV89] Andrzej Ehrenfeucht, David Haussler, Michael J. Kearns, and Leslie G. Valiant. A general lower bound on the number of examples needed for learning. *Inf. Comput.*, 82(3):247–261, 1989.

[FX14] Vitaly Feldman and David Xiao. Sample complexity bounds on differentially private learning via communication complexity. *CoRR*, abs/1402.6278, 2014.

[GHRU11] Anupam Gupta, Moritz Hardt, Aaron Roth, and Jonathan Ullman. Privately releasing conjunctions and the statistical query barrier. In Lance Fortnow and Salil P. Vadhan, editors, *STOC*, pages 803–812. ACM, 2011.

[HR10] Moritz Hardt and Guy N. Rothblum. A multiplicative weights mechanism for privacy-preserving data analysis. In *FOCS*, pages 61–70. IEEE Computer Society, 2010.

[HT10] Moritz Hardt and Kunal Talwar. On the geometry of differential privacy. In *STOC*, pages 705–714, 2010.

[KLN$^+$11] Shiva Prasad Kasiviswanathan, Homin K. Lee, Kobbi Nissim, Sofya Raskhodnikova, and Adam Smith. What can we learn privately? *SIAM J. Comput.*, 40(3):793–826, 2011.

[Mas90] P. Massart. The tight constant in the dvoretzky-kiefer-wolfowitz inequality. *Ann. Probab.*, 18(3):1269–1283, 07 1990.

[MN12] S. Muthukrishnan and Aleksandar Nikolov. Optimal private halfspace counting via discrepancy. In *Proceedings of the Forty-fourth Annual ACM Symposium on Theory of Computing*, STOC '12, pages 1285–1292, New York, NY, USA, 2012. ACM.

[MT07] Frank McSherry and Kunal Talwar. Mechanism design via differential privacy. In *Proceedings of the 48th Annual IEEE Symposium on Foundations of Computer Science*, FOCS '07, pages 94–103, Washington, DC, USA, 2007. IEEE Computer Society.

[NTZ13] Aleksandar Nikolov, Kunal Talwar, and Li Zhang. The geometry of differential privacy: the sparse and approximate cases. In *STOC*, pages 351–360, 2013.

[RR14] Omer Reingold and Guy Rothblum. Personal communication, May 2014.

[Val84] L. G. Valiant. A theory of the learnable. *Commun. ACM*, 27(11):1134–1142, November 1984.

[VC71] Vladimir N. Vapnik and Alexey Y. Chervonenkis. On the uniform convergence of relative frequencies of events to their probabilities. *Theory of Probability and its Applications*, 16(2):264–280, 1971.

A The Choosing Mechanism

We supply the proofs of privacy and utility for the choosing mechanism.

Proof of Lemma 3.6. Let A denote the choosing mechanism (Algorithm 2). Let S, S' be neighboring databases of m elements. We need to show that $\Pr[A(S) \in R] \leq \exp(\epsilon) \cdot \Pr[A(S') \in R] + \delta$ for every set of outputs $R \subseteq \mathcal{F} \cup \{\perp\}$. Note first that $\mathrm{OPT}(S) = \max_{f\in\mathcal{F}}\{q(S,f)\}$ has sensitivity at most 1, so by the properties of the Laplace Mechanism,

$$\begin{aligned}
\Pr[A(S)=\perp] &= \Pr\left[\widetilde{\mathrm{OPT}}(S) < \frac{8}{\epsilon}\ln(\frac{4k}{\beta\epsilon\delta})\right] \\
&\leq \exp(\frac{\epsilon}{4})\cdot\Pr\left[\widetilde{\mathrm{OPT}}(S') < \frac{8}{\epsilon}\ln(\frac{4k}{\beta\epsilon\delta})\right] \\
&= \exp(\frac{\epsilon}{4})\cdot\Pr[A(S')=\perp]. \qquad (3)
\end{aligned}$$

Similarly, we have $\Pr[A(S) \neq \perp] \leq \exp(\varepsilon/4)\Pr[A(S') \neq \perp]$. Thus, we my assume below that $\perp \notin R$. (If $\perp \in R$, then we can write $\Pr[A(S) \in R] = \Pr[A(S) = \perp] + \Pr[A(S) \in R \setminus \{\perp\}]$, and similarly for S'.)

Case (a): $\mathrm{OPT}(S) < \frac{4}{\epsilon}\ln(\frac{4k}{\beta\epsilon\delta})$**.** It holds that

$$\begin{aligned}
\Pr[A(S)\in R] &\leq \Pr[A(S)\neq\perp] \\
&\leq \Pr\left[\mathrm{Lap}\left(\frac{4}{\epsilon}\right) > \frac{4}{\epsilon}\ln\left(\frac{4k}{\beta\epsilon\delta}\right)\right] \\
&\leq \delta \leq \Pr[A(S')\in R]+\delta.
\end{aligned}$$

Case (b): $\mathrm{OPT}(S) \geq \frac{4}{\epsilon}\ln(\frac{4k}{\beta\epsilon\delta})$**.** Let $G(S)$ and $G(S')$ be the sets used in step 2 in the execution S and on S' respectively. We will show that the following two facts hold:

Fact 1 : For every $f \in G(S) \setminus G(S')$, it holds that $\Pr[A(S) = f] \leq \frac{\delta}{k}$.

Fact 2 : For every possible output $f \in G(S) \cap G(S')$, it holds that $\Pr[A(S) = f] \leq e^{\epsilon} \cdot \Pr[A(S') = f]$.

We first show that the two facts imply that the lemma holds for Case (b). Let $B \triangleq G(S) \setminus G(S')$, and note that as q is of k-bounded growth, $|B| \leq k$. Using the above two facts, for every set of outputs $R \subseteq \mathcal{F}$ we have

$$\begin{aligned}\Pr[A(S) \in R] &= \Pr[A(S) \in R \setminus B] + \sum_{f \in R \cap B} \Pr[A(S) = f] \\ &\leq e^{\epsilon} \cdot \Pr[A(S') \in R \setminus B] + |R \cap B| \frac{\delta}{k} \\ &\leq e^{\epsilon} \cdot \Pr[A(S') \in R] + \delta.\end{aligned}$$

To prove Fact 1, let $f \in G(S) \setminus G(S')$. That is, $q(S, f) \geq 1$ and $q(S', f) = 0$. As q has sensitivity at most 1, it must be that $q(S, f) = 1$. As there exists $\hat{f} \in S$ with $q(S, \hat{f}) \geq \frac{4}{\epsilon} \ln(\frac{4k}{\beta \epsilon \delta})$, we have that

$$\Pr[A(S) = f] \leq \Pr\left[\begin{array}{c}\text{The exponential} \\ \text{mechanism chooses } f\end{array}\right] \leq \frac{\exp(\frac{\epsilon}{4} \cdot 1)}{\exp(\frac{\epsilon}{4} \cdot \frac{4}{\epsilon} \ln(\frac{4k}{\beta \epsilon \delta}))} = \exp\left(\frac{\epsilon}{4}\right) \frac{\beta \varepsilon \delta}{4k},$$

which is at most δ/k for $\epsilon \leq 2$.

To prove Fact 2, let $f \in G(S) \cap G(S')$ be a possible output of $A(S)$. We use the following Fact 3, proved below.

Fact 3 : $\sum_{h \in G(S')} \exp(\frac{\epsilon}{4} q(S', h)) \leq e^{\epsilon/2} \cdot \sum_{h \in G(S)} \exp(\frac{\epsilon}{4} q(S, h))$.

Using Fact 3, for every possible output $f \in G(S) \cap G(S')$ we have that

$$\begin{aligned}&\frac{\Pr[A(S) = f]}{\Pr[A(S') = f]} \\ &= \left(\Pr[A(S) \neq \perp] \cdot \frac{\exp(\frac{\epsilon}{4} q(f, S))}{\sum_{h \in G(S)} \exp(\frac{\epsilon}{4} q(h, S))}\right) \Bigg/ \left(\Pr[A(S') \neq \perp] \cdot \frac{\exp(\frac{\epsilon}{4} q(f, S'))}{\sum_{h \in G(S')} \exp(\frac{\epsilon}{4} q(h, S'))}\right) \\ &= \frac{\Pr[A(S) \neq \perp]}{\Pr[A(S') \neq \perp]} \cdot \frac{\exp(\frac{\epsilon}{4} q(f, S))}{\exp(\frac{\epsilon}{4} q(f, S'))} \cdot \frac{\sum_{h \in G(S')} \exp(\frac{\epsilon}{4} q(h, S'))}{\sum_{h \in G(S)} \exp(\frac{\epsilon}{4} q(h, S))} \leq e^{\frac{\epsilon}{4}} \cdot e^{\frac{\epsilon}{4}} \cdot e^{\frac{\epsilon}{2}} = e^{\epsilon}.\end{aligned}$$

We now prove Fact 3. Let $\mathcal{X} \triangleq \sum_{h \in G(S)} \exp(\frac{\varepsilon}{4} q(S, h))$. Since there exists a solution $\hat{f}$ s.t. $q(S, \hat{f}) \geq \frac{4}{\epsilon} \ln(\frac{4k}{\beta \epsilon \delta})$, we have $\mathcal{X} \geq \exp(\frac{\epsilon}{4} \cdot \frac{4}{\epsilon} \ln(\frac{4k}{\beta \epsilon \delta})) \geq \frac{4k}{\epsilon}$.

Now, recall that q is of k-bounded growth, so $|G(S') \setminus G(S)| \leq k$, and every $h \in (G(S') \setminus G(S))$ satisfies $q(S', h) = 1$. Hence,

$$\begin{aligned}\sum_{h \in G(S')} \exp\left(\frac{\epsilon}{4} q(S', h)\right) &\leq k \cdot \exp\left(\frac{\epsilon}{4}\right) + \sum_{h \in G(S') \cap G(S)} \exp\left(\frac{\epsilon}{4} q(S', h)\right) \\ &\leq k \cdot \exp\left(\frac{\epsilon}{4}\right) + \exp\left(\frac{\epsilon}{4}\right) \cdot \sum_{h \in G(S') \cap G(S)} \exp\left(\frac{\epsilon}{4} q(S, h)\right) \\ &\leq k \cdot \exp\left(\frac{\epsilon}{4}\right) + \exp\left(\frac{\epsilon}{4}\right) \cdot \sum_{h \in G(S)} \exp\left(\frac{\epsilon}{4} q(S, h)\right) \\ &= k \cdot e^{\epsilon/4} + e^{\epsilon/4} \cdot \mathcal{X} \leq e^{\epsilon/2} \mathcal{X},\end{aligned}$$

where the last inequality follows from the fact that $\mathcal{X} \geq 4k/\varepsilon$. This concludes the proof of Fact 3, and completes the proof of the lemma. $\square$

The utility analysis for the choosing mechanism is rather straightforward:

Proof of Lemma 3.7. Recall that the mechanism defines $\widetilde{\mathrm{OPT}}(S)$ as $\mathrm{OPT}(S) + \mathrm{Lap}(\frac{4}{\epsilon})$. Note that the mechanism succeeds whenever $\widetilde{\mathrm{OPT}}(S) \geq \frac{8}{\epsilon}\ln(\frac{4k}{\beta\epsilon\delta})$. This happens provided the Lap $\left(\frac{4}{\varepsilon}\right)$ random variable is at most $\frac{8}{\varepsilon}\ln(\frac{4k}{\beta\varepsilon\delta})$, which happens with probability at least $(1-\beta)$. □

Proof of Lemma 3.8. Note that if $\mathrm{OPT}(S) < \frac{16}{\epsilon}\ln(\frac{4km}{\beta\epsilon\delta})$, then every solution is a good output, and the mechanism cannot fail. Assume, therefore, that there exists a solution f s.t. $q(f,S) \geq \frac{16}{\epsilon}\ln(\frac{4km}{\beta\epsilon\delta})$, and recall that the mechanism defines $\widetilde{\mathrm{OPT}}(S)$ as $\mathrm{OPT}(S)+\mathrm{Lap}(\frac{4}{\epsilon})$. As in the proof of Lemma 3.7, with probability at least $1-\beta/2$, we have $\widetilde{\mathrm{OPT}}(S) \geq \frac{8}{\varepsilon}\ln\left(\frac{4k}{\beta\varepsilon\delta}\right)$. Assuming this event occurs, we will show that with probability at least $1-\beta/2$, the exponential mechanism chooses a solution f s.t. $q(S,f) \geq \mathrm{opt}(S) - \frac{16}{\epsilon}\ln(\frac{4km}{\beta\epsilon\delta})$.

By the growth-boundedness of q, and as S is of size m, there are at most km possible solutions f with $q(S,f) > 0$. That is, $|G(S)| \leq km$. By the properties of the Exponential Mechanism, we obtain a solution as desired with probability at least

$$\left(1 - km \cdot \exp\left(-\frac{\epsilon}{4} \cdot \frac{16}{\epsilon} \ln\left(\frac{4km}{\beta\epsilon\delta}\right)\right)\right) \geq \left(1 - \frac{\beta}{2}\right).$$

By a union bound, we get that the choosing mechanism outputs a good solution with probability at least $(1-\beta)$. □

B Interior Point Fingerprinting Codes

Fingerprinting codes were introduced by Boneh and Shaw [BS98] to address the problem of watermarking digital content. Suppose a content distributor wishes to distribute a piece of digital content to n legitimate users in such a way that any pirated copy of that content can be traced back to any user who helped in producing the copy. A *fingerprinting code* is a scheme for assigning each n users a codeword that can be hidden in their copy of the content, and then be uniquely traced back to the identity of that user. Informally, a finger printing code is *fully collusion-resistant* if when an arbitrary coalition T of users combine their codewords to produce a new pirate codeword the pirate codeword can still be successfully traced to a member of T, provided the pirate codeword satisfies a certain *marking assumption*. Traditionally, this marking assumption requires that if all users in T see the same bit b at index j of their codewords, then index j of their combined codeword must also be b.

Recent work has shown how to use fingerprinting codes to obtain lower bounds in differential privacy [BUV14, DTTZ14, BST14]. Roughly speaking, these works show how any algorithm with nontrivial accuracy for a given task can be used to create a pirate algorithm that satisfies the marking assumption for a fingerprinting code. The security of the fingerprinting code means that the output of this algorithm can be traced back to one of its inputs. This implies that the algorithm is not differentially private.

We show how our lower bound for privately solving the interior point problem can also be proved by the construction of an object we call an *interior point fingerprinting code*. The difference between this object and a traditional fingerprinting code lies in the marking assumption. Thinking of our codewords as being from an ordered domain X, our marking assumption is that the codeword produced by a set of T users must be an interior point of their codewords. The full definition of the code is as follows.

Definition B.1. For a totally ordered domain X, an *interior point fingerprinting code* over X consists of a pair of randomized algorithms $(\mathrm{Gen}, \mathrm{Trace})$ with the following syntax.

- Gen_n samples a codebook $C = (x_1, \dots, x_n) \in X^n$
- $\mathrm{Trace}_n(x)$ takes as input a "codeword" $x \in X$ and outputs either a user $i \in [n]$ or a failure symbol $\perp$.

The algorithms Gen and Trace are allowed to share a common state (e.g. their random coin tosses).

The adversary to a fingerprinting code consists of a subset $T \subseteq [n]$ of users and a pirate algorithm $\mathcal{A} : X^{|T|} \to X$. The algorithm $\mathcal{A}$ is given $C|_T$, i.e. the codewords x_i for $i \in T$, and its output $x \leftarrow_{\mathrm{R}} \mathcal{A}(C|_T)$ is said to be "feasible" if $x \in [\min_{i \in T} x_i, \max_{i \in T} x_i]$. The security guarantee of a fingerprinting code is that for all coalitions $T \subseteq [n]$ and all pirate algorithms $\mathcal{A}$, if $x = \mathcal{A}(C|_T)$, then we have

1. Completeness: $\Pr[\mathrm{Trace}(x) = \perp \wedge x \text{ feasible}] \leq \gamma$, where $\gamma \in [0, 1]$ is the *completeness error.*
2. Soundness: $\Pr[\mathrm{Trace}(x) \in [n] \setminus T] \leq \xi$, where $\xi \in [0, 1]$ is the *soundness error.*

The probabilities in both cases are taken over the coins of $\mathrm{Gen}, \mathrm{Trace}$, and $\mathcal{A}$.

Remark B.2. *We note that an interior point fingerprinting code could also be interpreted as an ordinary fingerprinting code (using the traditional marking assumption) with codewords of length* $|X|$ *of the form* 000011111. *As an example for using such a code, consider a vendor interested in fingerprinting movies. Using an interior point fingerprinting code, the vendor could produce fingerprinted copies by simply splicing two versions of the movie.*

We now argue as in [BUV14] that the existence of an interior point fingerprinting code yields a lower bound for privately solving the interior point problem.

Lemma B.3. *Let $\varepsilon \leq 1$, $\delta \leq 1/(12n)$, $\gamma \leq 1/2$ and $\xi \leq 1/(33n)$. If there is an interior point fingerprinting code on domain X for n users with completeness error γ and soundness error ξ, then there is no (ε, δ)-differentially private algorithm that, with probability at least $2/3$, solves the interior point problem on X for databases of size $n - 1$.*

Proof. Suppose for the sake of contradiction that there were a differentially private $\mathcal{A}$ for solving the interior point problem on X^{n-1}. Let $T = [n-1]$, and let $x = \mathcal{A}(C|_T)$ for a codebook $C \leftarrow_{\mathrm{R}} \mathrm{Gen}$.

$$1 - \gamma \leq \Pr[\mathrm{Trace}(x) \neq \perp \vee x \text{ not feasible}] \leq \Pr[\mathrm{Trace}(x) \neq \perp] + \frac{1}{3}.$$

Therefore, there exists some $i^* \in [n]$ such that

$$\Pr[\mathrm{Trace}(x) = i^*] \geq \frac{1}{n} \cdot \left(\frac{2}{3} - \gamma\right) \geq \frac{1}{6n}.$$

Now consider the coalition T' obtained by replacing user i^* with user n. Let $x' = \mathcal{A}(C|_{T'})$, again for a random codebook $C \leftarrow_{\mathrm{R}} \mathrm{Gen}$. Since $\mathcal{A}$ is differentially private,

$$\Pr[\mathrm{Trace}(x') = i^*] \geq e^{-\varepsilon} \cdot (\Pr[\mathrm{Trace}(x) = i^*] - \delta) > \frac{1}{33n} \geq \xi,$$

contradicting the soundness of the interior point fingerprinting code. $\square$

We now show how to construct an interior point fingerprinting code, using similar ideas as in the proof of Lemma 3.3. For n users, the codewords lie in a domain with size an exponential tower in n, allowing us to recover the $\log^* |X|$ lower bound for interior point queries.

Lemma B.4. *For every $n \in \mathbb{N}$ and $\xi > 0$ there is an interior point fingerprinting code for n users with completeness $\gamma = 0$ and soundness ξ on a domain X_n of size $|X_n| \leq \mathrm{tower}^{(n+\log^*(2n^2/\xi))}(1)$.*

Proof. Let $b(n) = 2n^2/\xi$, and define the function S recursively by $S(1) = 1$ and $S(n+1) = b(n)^{S(n)}$. By induction on n, we will construct codes for n users over a domain of size $S(n)$ with perfect completeness and soundness at most $\sum_{j=1}^{n} \frac{1}{b(j)} < \xi$. First note that there is a code with perfect completeness and perfect soundness for $n = 1$ user over a domain of size $S(1) = 1$. Suppose we have defined the behavior of $(\mathrm{Gen}_n, \mathrm{Trace}_n)$ for n users. Then we define

- Gen_{n+1} samples $C' = (x'_1, \dots, x'_n) \leftarrow_{\mathrm{R}} \mathrm{Gen}_n$ and $x_{n+1} \leftarrow_{\mathrm{R}} [S(n+1)]$. For each $i = 1, \dots, n$, let x_i be a base-$b(n)$ number (written $x_i^{(0)} x_i^{(1)} \dots x_i^{(S(n)-1)}$, where $x_i^{(0)}$ is the most significant digit) that agrees with x_{n+1} in the x'_i most-significant digits, and has random entries from $[b(n)]$ at every index thereafter. The output codebook is $C = (x_1, \dots, x_{n+1})$.
- $\mathrm{Trace}_{n+1}(x)$ retrieves the codebook C from its shared state with Gen_{n+1}. Let M be the maximum number of digits to which any x_i (for $i = 1, \dots, n$) agrees with x_{n+1}. If x agrees with x_{n+1} on more than M digits, accuse user $n+1$. Otherwise, let x' be the number of indices on which x agrees with x_{n+1}, and run $\mathrm{Trace}_n(x')$ with respect to codebook $C' = (x'_1, \dots, x'_n)$.

We reduce the security of this scheme to that of $(\mathrm{Gen}_n, \mathrm{Trace}_n)$. To check completeness, let $T \subseteq [n+1]$ be a pirate coalition and let $\mathcal{A}$ be a pirate algorithm. Consider the pirate algorithm $\mathcal{A}'$ for codes on n users that, given a set of codewords $C'|_{T'}$ where $T' = T \setminus \{n+1\}$, simulates Gen_{n+1} to produce a set of codewords $C|_T$ and outputs the number x' of indices on which $x = \mathcal{A}(C|_T)$ agrees with x_{n+1}.

If x is feasible for $C|_T$ and $x^{M+1} \neq x_{n+1}^{M+1}$, then x' is feasible for $C'|_{T'}$. Therefore,

$$\begin{aligned}\Pr[\mathrm{Trace}_{n+1}(x) = \perp \wedge x \text{ feasible for } C|_T] &= \Pr[x^{M+1} \neq x_{n+1}^{M+1} \wedge \mathrm{Trace}_n(x') = \perp \wedge x \text{ feasible for } C|_T] \\ &\leq \Pr[\mathrm{Trace}_n(x') = \perp \wedge x' \text{ feasible for } C'|_{T'}] = 0,\end{aligned}$$

by induction, proving perfect completeness.

To prove soundness, let $M' = \max x'_i$. Then

$$\begin{aligned}\Pr[\mathrm{Trace}_{n+1}(x) \in [n+1] \setminus T] &\leq \Pr[\mathrm{Trace}_{n+1}(x) = n+1 \wedge (n+1) \notin T] + \Pr[\mathrm{Trace}_{n+1}(x) \in [n] \setminus T] \\ &\leq \Pr[x^{M'+1} = x_{n+1}^{M'+1} \wedge (n+1) \notin T] + \Pr[\mathrm{Trace}_n(x') \in [n] \setminus T] \\ &\leq \frac{1}{b(n)} + \sum_{j=1}^{n-1} \frac{1}{b(j)} = \sum_{j=1}^{n} \frac{1}{b(j)} < \xi.\end{aligned}$$

□

Combining Lemmas B.3 and B.4 yields Theorem 1.8.

C Another Reduction from Releasing Thresholds to the Interior Point Problem

We give a somewhat different reduction showing that solving the interior point problem enables us to α-accurately release thresholds with a $\text{polylog}(1/\alpha)/\alpha$ blowup in sample complexity. It gives qualitatively the same parameters as Algorithm *Thresh* used to prove Theorem 4.6, but we believe the ideas used for this reduction may be useful in the design of other differentially private algorithms.

This reduction computes approximate $(\alpha/3)$-quantiles of its input, which can then be used to release thresholds with α-accuracy. To do so, it uses the strategy of [DNPR10] of using a complete binary tree to generate a sequence of $k = 3/\alpha$ noise values. The tree has k leaves and depth $\log k$, and at each node in the tree we sample a Laplace random variable. The noise value corresponding to a leaf is the sum of the samples along the path from that leaf to the root.

We take the sorted input database and divide it into equal-size blocks around the k $(\alpha/3)$-quantiles, and perturb the boundaries of the blocks by the k noise values. Solving the interior point problem on these buckets then gives approximate $(\alpha/3)$-quantiles. Moreover, the noisy bucketing step ensures that the final algorithm is differentially private.

We formally describe this algorithm as $Thresh_2$ below. Let R be an (ε, δ)-differentially private mechanism for solving the interior point problem on X that succeeds with probability at least $1 - \alpha\beta/6$ on databases of size m. In the algorithm below, let $P(i)$ denote the set of prefixes of the binary representation of i (including the empty prefix).

Algorithm 6 $Thresh_2(D)$

Input: Database $D = (x_1, \dots, x_n) \in X^n$

```
1. Sort $D$ in nondecreasing order
2. Let $k = 3/\alpha$ be a power of 2
3. For each $s \in \{0,1\}^\ell$ with $0 \le \ell \le \log k$, sample $\nu_s \sim \text{Lap}((\log k)/2\varepsilon)$
4. For each $i = 1, \dots, k$, let $\eta_i = \sum_{s \in P(i)} \nu_s$
5. Let $T_0 = \alpha n/6, T_1 = \alpha n/2 + \eta_1, \dots, T_{k-2} = \alpha n/6 + \alpha(k-2)n/3 + \eta_{k-1}, T_{k-1} = n - \alpha n/6$
6. Divide $D$ into blocks $D_1, \dots, D_{k-1}$, where $D_i = (x_{T_{i-1}}, \dots, x_{T_i - 1})$ (note $D_i$ may be empty)
7. Release $R(D_1), \dots, R(D_m)$, interpreted as approximate $(\alpha/3)$-quantiles.
```

We will show that this algorithm satisfies $(3\varepsilon, (1 + e^{\varepsilon})\delta)$-differential privacy, and is able to release approximate $k(= 3/\alpha)$-quantiles with $(\alpha/3, \beta)$-accuracy, and hence (α, β)-accurate answers to threshold queries, as long as

$$n \ge \max\left\{ \frac{6m}{\alpha}, \frac{99 \log^{2.5}(1/\alpha)}{\alpha\varepsilon} \right\}$$

Privacy Let $D = (x_1, \ldots, x_n)$ where $x_1 \leq x_2 \leq \cdots \leq x_n$, and let $D' = (x_1, \ldots, x_i', \ldots, x_n)$. Assume without loss of generality that $x_i' \geq x_{i+1}$, and suppose

$$x_1 \leq \cdots \leq x_{i-1} \leq x_{i+1} \leq \cdots \leq x_j \leq x_i' \leq x_{j+1} \leq \cdots \leq x_n.$$

Consider vectors of noise values $\nu = (\nu_1, \nu_2, \ldots, \nu_m)$. Then there is a bijection between noise vectors ν and noise vectors ν' such that D partitioned according to ν and D' partitioned according to ν' differ on at most 2 blocks (cf. [DNPR10]). Moreover, this bijection changes at most $2 \log m$ values ν_s by at most 1. Thus under this mapping, noise vector ν' is sampled with probability at most e^{ε} times the probability ν is sampled. We get that for any set S,

$$\begin{aligned}\Pr[(M(D_1), \ldots, M(D_m)) \in S] &\leq e^{\varepsilon}(e^{2\varepsilon} \Pr[(M(D_1'), \ldots, M(D_m')) \in S]) + (1 + e^{\varepsilon})\delta \\ &= e^{3\varepsilon} \Pr[(M(D_1'), \ldots, M(D_m')) \in S] + (1 + e^{\varepsilon})\delta.\end{aligned}$$

Utility We can produce $(\alpha/3)$-accurate estimates of every quantile as long as

1. Every noise value has magnitude at most $\alpha n/3$
2. Every execution of R succeeds

By the analysis of Lemma 4.7 in [DNPR10], with probability at least $1 - \beta/2$, every noise value η_i is bounded by $11 \log^{2.5}(1/\alpha)/\varepsilon \leq \alpha n/12$. This suffices to achieve item 1. Moreover, conditioned on the noise values being so bounded, each $|D_i| \geq \alpha n/6 \geq m$, so each execution of R individually succeeds with probability $1 - \alpha\beta/6$. Hence they all succeed simultaneously with probability at least $1 - \beta/2$, giving item 2.