New Separations in the Complexity of Differential Privacy

Citation

Permanent link
http://nrs.harvard.edu/urn-3:HUL.InstRepos:33840647

Terms of Use
This article was downloaded from Harvard University’s DASH repository, and is made available under the terms and conditions applicable to Other Posted Material, as set forth at http://nrs.harvard.edu/urn-3:HUL.InstRepos:dash.current.terms-of-use#LAA

Share Your Story
The Harvard community has made this article openly available. Please share how this access benefits you. Submit a story.

Accessibility
New Separations in the Complexity of Differential Privacy

A dissertation presented
by
Mark Mar Bun
to
The School of Engineering and Applied Sciences
in partial fulfillment of the requirements
for the degree of
Doctor of Philosophy
in the subject of
Computer Science

Harvard University
Cambridge, Massachusetts

July 2016
New Separations in the Complexity of Differential Privacy

Abstract

In this thesis, we study when algorithmic tasks can be performed on sensitive data while protecting the privacy of individuals whose information is collected. We focus on the notion of differential privacy, which gives a strong formal guarantee that no individual’s data has a significant impact on the outcome of a computation. Intense study over the last decade has shown that a rich variety of statistical analyses can be performed with differential privacy. However, the corresponding private algorithms generally require more computational resources than their non-private counterparts. The goal of this thesis is to improve our understanding of two basic measures of the inherent complexity of differential privacy: sample complexity and computational complexity.

In the first part of this thesis, we study the sample complexity — the minimum amount of data needed to obtain accurate results — of basic query release tasks. We show, for the first time, that approximate differential privacy can demand higher sample complexity than what is needed to ensure statistical accuracy alone. In particular:

- We establish tight lower bounds on the sample complexity of answering marginal queries with differential privacy. Our results are based on a new connection between privacy lower bounds and cryptographic fingerprinting codes.
- We show the first lower bounds against privately releasing the simple class of threshold functions. This reveals a price of privacy even for low-dimensional query families.
- We study how the sample complexity of private query release changes depending on whether queries are given to an algorithm offline, online or adaptively, and expose significant gaps between these three models.
Next, we examine the sample complexity of differentially private PAC learning. Again, we exhibit separations between what is statistically feasible and what is feasible with differential privacy.

- We exhibit a lower bound for properly learning threshold functions with differential privacy. This separates differentially private from non-private proper PAC learning.

- We initiate the study of learning multiple concepts simultaneously under differential privacy. By contrast with the non-private case, the sample complexity of private learning grows significantly with the number of concepts being learned.

Finally, we address the computational complexity of PAC learning with differential privacy.

- Under cryptographic assumptions, we give the first example of a concept class that is efficiently PAC learnable, but not efficiently learnable with differential privacy.
Contents

Abstract ................................................................. iii
Table of Contents ....................................................... v
Acknowledgments ......................................................... viii

1 Introduction ......................................................... 1
  1.1 Contributions of this Thesis .................................... 3

2 Background and Preliminaries ................................... 9
  2.1 Differential Privacy ............................................. 10
  2.2 Counting Queries and Accuracy ................................. 12
  2.3 Sample Complexity .............................................. 15
  2.4 Private Distribution Learning .................................. 22
  2.5 (Private) PAC Learning ......................................... 25
  2.6 Fingerprinting Codes ............................................ 28

3 Fingerprinting Codes and the Sample Complexity of Releasing Marginals .......................................................... 35
  3.1 Results and Techniques ......................................... 36
  3.2 Fingerprinting Codes vs. Differential Privacy ................. 42
  3.3 A Composition Theorem for Sample Complexity Lower Bounds ......................................................... 56
  3.4 Lower Bounds for $k$-Way Marginals ............................ 63

4 Private Release and Learning of Threshold Functions .......... 71
  4.1 Results and Techniques ......................................... 71
  4.2 The Interior Point Problem – Lower Bounds ................... 77
  4.3 The Interior Point Problem – Upper Bound ..................... 88
  4.4 Query Release and Distribution Learning ....................... 99
  4.5 PAC Learning .................................................. 103
  4.6 Thresholds in High Dimension ................................... 105

5 The Price of Online Queries in Differential Privacy .......... 111
  5.1 Results .......................................................... 112
  5.2 Techniques ..................................................... 115
  5.3 Models of Interactive Queries ................................... 118
5.4 Separating Offline and Online Queries ........................................... 120
5.5 Separating Adaptive and Non-Adaptive Online Queries .................. 128
5.6 Threshold Queries ............................................................... 133

6 Simultaneous Private Learning of Multiple Concepts 150
6.1 Results and Techniques .......................................................... 151
6.2 Model and Preliminaries ......................................................... 155
6.3 Upper Bounds on the Sample Complexity of Private Multi-Learners .... 164
6.4 Approximate Privacy Lower Bounds from Fingerprinting Codes ....... 174
6.5 Examples where the Direct Sum is Optimal ................................. 180

7 Computational Complexity of Private PAC Learning 183
7.1 Results and Techniques .......................................................... 183
7.2 Preliminaries ........................................................................... 192
7.3 EncThresh and its Learnability ............................................... 197
7.4 Constructing Strongly Correct ORE Schemes ............................... 207
7.5 Hardness of Representation Learning ........................................ 211

8 Conclusions .............................................................................. 216

Bibliography 219
List of Open Problems

<table>
<thead>
<tr>
<th>Section</th>
<th>Problem Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.3.7</td>
<td>Open Problem: Dependence of sample complexity on $\delta$</td>
<td>18</td>
</tr>
<tr>
<td>2.3.9</td>
<td>Open Problem: Dependence of sample complexity on $\alpha$</td>
<td>19</td>
</tr>
<tr>
<td>3.1.3</td>
<td>Open Problem: Efficiently releasing marginals</td>
<td>40</td>
</tr>
<tr>
<td>3.2.14</td>
<td>Open Problem: Generalized fingerprinting codes vs. sample complexity lower bounds</td>
<td>51</td>
</tr>
<tr>
<td>4.1.5</td>
<td>Open Problem: Sample complexity of improper private learning</td>
<td>75</td>
</tr>
<tr>
<td>4.3.10</td>
<td>Open Problem: Sample complexity of the interior point problem</td>
<td>97</td>
</tr>
<tr>
<td>4.5.3</td>
<td>Open Problem: Improperly learning thresholds</td>
<td>104</td>
</tr>
<tr>
<td>5.1.3</td>
<td>Open Problem: Online vs. adaptive models for counting queries</td>
<td>113</td>
</tr>
<tr>
<td>5.1.6</td>
<td>Open Problem: Releasing adaptively chosen thresholds</td>
<td>114</td>
</tr>
<tr>
<td>5.1.7</td>
<td>Open Problem: Offline vs. online models for natural queries</td>
<td>114</td>
</tr>
<tr>
<td>6.1.2</td>
<td>Open Problem: Private learning vs. sanitization</td>
<td>152</td>
</tr>
<tr>
<td>6.1.3</td>
<td>Open Problem: Multiplicative lower bounds for multi-learning</td>
<td>153</td>
</tr>
<tr>
<td>6.4.6</td>
<td>Open Problem: Privately learning $k$ thresholds</td>
<td>177</td>
</tr>
<tr>
<td>6.4.7</td>
<td>Open Problem: Optimal structured fingerprinting codes</td>
<td>178</td>
</tr>
<tr>
<td>7.1.2</td>
<td>Open Problem: Hardness of private PAC learning under weaker assumptions</td>
<td>185</td>
</tr>
<tr>
<td>7.1.3</td>
<td>Open Problem: Existence of ORE schemes</td>
<td>187</td>
</tr>
<tr>
<td>8.0.5</td>
<td>Open Problem: Combinatorial characterization of sample complexity</td>
<td>217</td>
</tr>
</tbody>
</table>
I am indebted to a number of amazing individuals for shaping my experience in graduate school. Foremost among these is my advisor, Salil Vadhan. In addition to being an incredible researcher, Salil is a wonderful teacher and mentor. Of the innumerable qualities that make him a superb advisor, one I wish to highlight is his contagious enthusiasm for research. Interacting with Salil — even if it’s to have him point out where I’ve gone horribly wrong — always leaves me with new excitement about all of the open possibilities. Any success I’ve had in graduate school I owe to Salil for his generosity in sharing his time and insights, his patient encouragement to pursue all of my research interests, and his sage advice on all matters scholarly and not-so-scholarly.

I also owe a great deal of gratitude to Kobbi Nissim. I was very fortunate to arrive at Harvard at the same time as Kobbi, and to enjoy his mentorship and collaboration throughout my time in graduate school. Kobbi has been a constant source of inspiration, knowledge, and encouragement. All of the work in this thesis is, one way or another, inspired by Kobbi’s questions and insights.

Others who have profoundly impacted my experience in graduate school include: Uri Stemmer, for introducing me to Turkish coffee; Thomas Steinke, for his partnership in all-you-can-eat sushi and giant chili; Justin Thaler, for agreeing to our joint trail mix venture; Jon Ullman, for being a trustworthy source of opinions on food of every caliber; and Bo Waggoner for always keeping sandwich material on hand. Of course, I have had the privilege to interact with many other fantastic researchers, whom I thank for sharing their time and advice, and whom I hope to share many meaningful food-related experiences with in the future.

I wish to thank the other inhabitants of MD138 over the years: Rediet, Victor, Jarek, Anudhyan, Yi-Hsiu, Elad, Jiayang, Pravesh, Chin, Zhixian, Scott, Tom, Jack, Vasileios, Zhengyu, Jiapeng, and Colin. To all of you, I am grateful that you managed to resist reporting me for the disastrous state of my desk, or for my repeated attempts to colonize yours. Thanks go out to the many other students, postdocs, faculty, and staff at Harvard who made my experience here so enjoyable. Special thanks go to Carol Harlow and Kimia Mavon for putting up with my gross incompetence at all things logistic.

I would like to thank Boaz Barak, Jelani Nelson, Madhu Sudan and Les Valiant for serving on my thesis committee, and Yaron Singer for serving on my qualifying exam committee. I am grateful for your wisdom and support over the years. Thanks also to Dan Spielman for hosting a stimulating
visit to Yale, and to Amin, Rasmus, Marko, Mariana, and Sushant for welcoming my intrusion with new collaborations and friendships.

I thank my family in Seattle — Luke, Mary, Yeng, and Therese — for sending copious amounts of love, support, and candy. And last, but certainly not least, I thank Amy and Stacey for adding a whole new dimension of joy (and challenges!) to my graduate school experience.

The work in this thesis was mostly supported by the Department of Defense (DoD) through the National Defense Science & Engineering Graduate Fellowship (NDSEG) Program, and in part by NSF grant CNS-123723.
Chapter 1

Introduction

Consider an institution, e.g. the National Institutes of Health, the Census Bureau, or a social networking company, in possession of a dataset containing sensitive information about individuals. For example, the dataset may consist of medical records, socioeconomic attributes, or geolocation data. The institution faces an important tradeoff when deciding how to make this dataset available for statistical analysis. On one hand, if the institution releases the dataset (or at least statistical information about it), it can enable important research and eventually inform policy decisions. On the other hand, for a number of ethical and legal reasons it is important to protect the individual-level privacy of the data subjects. The field of privacy-preserving data analysis aims to reconcile these two objectives. That is, it seeks to enable rich statistical analyses on sensitive datasets while protecting the privacy of the individuals who contributed to them.

This thesis focuses on differential privacy, a compelling definition of privacy for provably privacy-preserving data analysis. Emerging from a line of work on mathematically modeling privacy for statistical datasets [13,41,53], differential privacy was first defined by Dwork, McSherry, Nissim, and Smith [47]. Roughly speaking, it defines individual-level privacy as the condition that no individual’s information has a significant effect on the outcome of a computation. This in turn guarantees that the result of a differentially private mechanism cannot reveal too much information that is specific to any individual.

To satisfy differential privacy, an algorithm must necessarily introduce (carefully calibrated) random noise to its answers. Nevertheless, a rich literature has developed to determine when useful population-level information can be extracted from data while guaranteeing differential privacy.
Indeed, it is now known that many algorithmic tasks can be performed with differential privacy, spanning statistical query release [14], machine learning [13, 81], and algorithmic mechanism design [98] as a few examples.

However, achieving differential privacy in these tasks comes at a cost. For one, the error introduced for differential privacy can be much higher than the sampling error one expects in a (non-private) statistical analysis. As a simple example, the earliest work in differential privacy [13, 41, 47, 53] showed that adding independent noise to each of $k$ statistical queries on a dataset (the “Laplace mechanism”) yields a differentially private algorithm introducing error $\tilde{O}(\sqrt{k/n})$ per query. While this error vanishes as the number of data subjects $n \to \infty$, it can be dramatically larger than the $O((\log k/n)^{1/2})$ error one expects due to random sampling. A major research focus in differential privacy has been to design algorithms that reduce the amount of error introduced for privacy, or to prove lower bounds revealing when it is inherent. One way to measure the tradeoff between privacy and accuracy is via the sample complexity of differential privacy, i.e. how large the number of samples $n$ needs to be in order to get the error introduced for differential privacy under control.

Of course, the story of algorithmic results in differential privacy only begins with the Laplace mechanism. Since the introduction of differential privacy, many more sophisticated algorithms have been discovered, often giving exponential improvements in the sample complexity needed to perform very general tasks. Unfortunately, many of the differentially private algorithms achieving the best-known asymptotic sample complexity are computationally inefficient. In some cases, this high computational complexity is known to be necessary (under widely believed cryptographic assumptions).

In this thesis, we study several closely related questions regarding the complexity of differential privacy. Motivated by the discussion above, most of our work is driven by the following question:

When does differential privacy require additional complexity — in terms of either sample complexity or running time — over what is required to achieve statistical accuracy alone?

We address this question through a detailed study of the power and limitations of differential privacy for some of the most basic query release and learning tasks. Along the way, we introduce several new techniques for proving lower bounds in differential privacy based on combinatorial and cryptographic constructions.
1.1 Contributions of this Thesis

We divide our study of the complexity of differential privacy into two main parts. First, we focus on understanding the sample complexity of private query release problems, where an analyst wishes to obtain approximate answers to a number of real-valued queries on a sensitive dataset. Second, we address the sample complexity and computational complexity of differential privacy for PAC learning, a theoretical framework which captures many basic supervised learning tasks.

1.1.1 The Complexity of Private Query Release

Non-Interactive Query Release. Some of the most basic queries an analyst may ask about a sensitive dataset are counting queries, which take the form “What fraction of the individual records in $D$ satisfy some predicate $q$?” In addition to capturing natural statistics about a dataset, counting queries are useful in the service of more sophisticated analyses in the statistical queries framework [13]. The query release problem for a family of counting queries $Q$ is as follows: Given a dataset $D$, output (in a differentially private way) an approximate answer to each of the queries in $Q$ that is accurate to within, say, $\pm 0.01$.

For a given family of queries $Q$, what is the sample complexity of releasing answers to $Q$ with differential privacy? How does it compare to the sample complexity of achieving statistical accuracy alone? For this to be a meaningful benchmark, we think of $D$ as containing i.i.d. samples from a larger population, and of the analyst as using each answer $q(D)$ to estimate the value of $q$ on the population to within $\pm 0.01$. It is well-known that achieving this form of statistical accuracy requires a number of records that is proportional to the Vapnik-Chervonenkis (VC) dimension of $Q$, which can be as large as $\log |Q|$.

Starting with the breakthrough of Blum, Ligett, and Roth [14], a long line of algorithmic results [50, 55, 73, 75, 76, 105] has shown that the price of differential privacy is small for low-dimensional datasets, e.g. when each record in $D$ is a short binary string in $\{0, 1\}^d$. The best known upper bound on the sample complexity of releasing a worst-case family of queries $Q$ is $O(\sqrt{d} \log |Q|)$, achieved by the private multiplicative weights algorithm of Hardt and Rothblum [76]. However, these results leave open the possibility that the price of privacy is large for high-dimensional datasets, namely at least a multiplicative factor of $\sqrt{d}$. As an extreme case, suppose we wish to simply estimate
the mean of each of \( d \) binary attributes. Then without a privacy guarantee, \( \Theta(\log d) \) samples are necessary and sufficient to get statistical accuracy. Meanwhile, the best known differentially private algorithm requires \( \Omega(\sqrt{d}) \) samples—an exponential gap.

In this thesis, we show that such a gap is inherent. We show in Chapter 3 that estimating the means of \( d \) binary attributes with differential privacy requires sample complexity \( \tilde{\Omega}(\sqrt{d}) \). To prove this result, we establish a connection between privacy lower bounds and cryptographic fingerprinting codes—a connection that will prove useful for many of the other results in this thesis. This gives the first separation between the sample complexity of \((\varepsilon, \delta)\)-differential privacy and the sample complexity required for statistical accuracy alone. We then introduce a composition theorem showing how to amplify this lower bound via the reconstruction attacks from the seminal work of Dinur and Nissim [41] (and subsequent extensions [40, 82]). We apply this composition theorem to obtain a sample complexity lower bound for \( k \)-way marginal queries—queries of the form, “given a subset \( S \subset [d] \) with \( |S| = k \), what fraction of records \( x \) have \( x_i = 1 \) for all \( i \in S' \)”—that essentially matches what is achievable with the private multiplicative weights algorithm. The work in this chapter is joint with Jonathan Ullman and Salil Vadhan (STOC 2014) [29].

However, for specific query families, the sample complexity of private query release can be much smaller than what is guaranteed by general-purpose algorithms such as private multiplicative weights. One such family that has received considerable attention is that of threshold functions over the domain \([T] = \{1, \ldots, T\}\), defined by \( q_x(y) = 1 \) iff \( y \leq x \). The corresponding query release problem captures the task of approximating the cumulative distribution function of an empirical data distribution. Beimel, Nissim, and Stemmer [8] showed that the sample complexity of releasing thresholds has at worst an extremely mild dependence on \( T \)—namely \( 2^{O(\log^* T)} \)—much lower than the \( \Omega(\log T) \) requirement of the general-purpose algorithms. Meanwhile, since the class of thresholds has VC dimension 1, it is possible (non-privately) to achieve statistical accuracy with sample complexity independent of \( T \), and even for infinite domains (such as \( \mathbb{N} \)). Could the same be true for privately releasing thresholds? Or is there a price of differential privacy even for this combinatorially simple task?

In Chapter 4, we show that releasing thresholds over the data domain \( \{1, \ldots, T\} \) requires sample complexity \( \Omega(\log^* T) \). Hence, while the message of prior lower bounds was that there is a sample
price of privacy for answering “complex queries” (as measured by, say, VC dimension) or queries about high-dimensional data, we exhibit such a price even for combinatorially simple problems. We prove this result via a reduction to a simple search problem we call the “interior point problem,” for which we provide several different combinatorial arguments to understand its sample complexity. This perspective further leads us to somewhat improved upper bounds for releasing thresholds. The work in this chapter is joint with Kobbi Nissim, Uri Stemmer, and Salil Vadhan (FOCS 2015) [27].

Interactive Query Release. Our discussion so far has focused on the non-interactive query release problem, in which a differentially private mechanism must accurately answer all of the queries from a pre-specified family $Q$. It is also of interest to study differential privacy in an interactive setting, where a data analyst might only be interested in the answers to a relatively short sequence of queries $q_1, \ldots, q_k$ from a larger set of allowable queries $Q$. This is in fact the more natural setting for differentially private algorithms that are based on techniques from online learning [32,73,76,105], such as the aforementioned private multiplicative weights algorithm. In Chapter 5 we take a fine-grained view of the complexity of interactive query release, and study the relationship between three different interactive models for how an analyst might specify these queries. These models capture whether the queries are given to the mechanism all in a single batch (“offline”), whether they are chosen in advance but presented to the mechanism one at a time (“online”), or whether they may be chosen by an analyst adaptively (“adaptive”).

Differential privacy is quite amenable to the adaptive model: any adaptively-chosen sequence of differentially private algorithms remains collectively differentially private, with a graceful degradation of the privacy parameters [47,55]. This adaptive composability of differential privacy underlies the fact that many of the most powerful differentially private algorithms, including private multiplicative weights [76], work just as well in the adaptive model as they do in the offline model. It also drives the recent line of work using differential privacy to ensure statistical validity in adaptive data analysis [44,79]. On the other hand, the matching lower bounds of e.g. Chapter 3 hold for a fixed set of queries, and hence even in the easiest offline model.

These observations may suggest that the ability to answer online or adaptively-chosen queries comes “for free” in differential privacy. In this thesis, we show for the first time that these three models of interaction are actually distinct. Using techniques based on fingerprinting codes, we exhibit
a family of counting queries that separate the offline and online models. We also construct family of
generalized “search” queries that separate the online and adaptive models. These separations hold
for contrived families of queries, so we also investigate whether they may hold for natural families
of queries, such as thresholds on the real line. Our study leads us to an improved algorithm for
releasing adaptively chosen threshold queries. The work in this chapter is joint with Thomas Steinke
and Jonathan Ullman [28].

1.1.2 The Complexity of Private Learning

Sample Complexity. Formalizing a line of work [13] applying differential privacy to machine
learning tasks, Kasiviswanathan et al. [81] defined private PAC learning as a combination of probably
approximately correct (PAC) learning [117] and differential privacy. Recall that the goal of a PAC
learner is to take a sequence of $n$ examples drawn i.i.d. from an unknown distribution, that are
labeled by an unknown concept $c$ from some concept class $C$, and generalize these to a hypothesis
$h$ that closely approximates $c$ on the underlying distribution. We are interested in PAC learning
algorithms that are also differentially private with respect to their input examples.

As with private query release, a basic question is to understand the sample complexity of privately
PAC learning a given concept class $C$ under differential privacy. Without concern for privacy, it is
again well known that the sample complexity of PAC learning is proportional to the VC dimension
of the class $C$. In their initial study of differentially private learning, Kasiviswanathan et al. [81]
showed that $O(\log |C|)$ labeled examples suffice to privately learn any finite concept class $C$. Recall
that the VC dimension of a concept class $C$ is always at most $\log |C|$, but is significantly lower for
many interesting classes, including threshold functions.

A number of works [5, 7, 33, 63] have since sought to characterize the true sample complexity
of differentially private learning for specific concept classes $C$. However, it remained open whether
there is any asymptotic gap between the sample complexity of private learning and non-private PAC
learning. To make progress on this question, Beimel et al. [8] showed that threshold functions can
be properly privately learned with sample complexity $2^{O(\log^* T)}$, but as in the case of private query
release, it remained possible that the sample complexity could be made independent of $T$.

Extending the techniques used to prove our query release lower bound for thresholds, we show
in Chapter 4 that the sample complexity of (proper) PAC learning with approximate differential
privacy can actually be asymptotically larger than the VC dimension. In particular, properly learning threshold functions over the domain \([T]\) requires sample complexity \(\Omega(\log^* T)\). We also study a higher dimensional analogue of the thresholds class which gives an asymptotic separation for concept classes of every finite VC dimension.

In the next part of this thesis, we move beyond the question of sample complexity for learning a single concept. In real learning applications, a data analyst might be interested in learning multiple hypotheses that generalize many different attributes of a given dataset. For example, a medical researcher may wish to learn the risk factors for \(k\) different diseases simultaneously. Does she need to collect a fresh set of labeled examples for each disease, or can she reuse examples when learning the different diseases? In Chapter 6 we address this direct-sum problem for differentially private PAC learning: What is the cost of solving many private PAC learning instances simultaneously, and how does it compare to solving each instance separately? In the model we consider, individual examples are drawn i.i.d. from a domain \(X\) and labeled by \(k\) unknown concepts \((c_1, \ldots, c_k)\) taken from a concept class \(C\). The goal of a multi-learner is to output \(k\) hypotheses \((h_1, \ldots, h_k)\) that generalize the input examples while guaranteeing differential privacy.

Without a privacy guarantee, the sample complexity of learning \(k\) concepts simultaneously is still characterized (up to constant factors) by the VC dimension of the concept class \(C\). In particular, it is independent of the number of learning tasks \(k\). In this thesis, we examine how the situation changes when the learning is performed with differential privacy. We give new upper bounds showing how to reduce the sample complexity of learning \(k\) concepts beyond what composition theorems for differential privacy enable. We also give lower bounds showing that, quite unlike in the case of non-private learning, the sample complexity of private learning may grow polynomially with \(k\). Both threshold functions and fingerprinting codes perform an encore in our lower bounds, with thresholds providing a hard concept class, and fingerprinting codes serving as a tool to prove lower bounds. The work in this chapter is joint with Kobbi Nissim and Uri Stemmer (ITCS 2016) [26].

**Computational Complexity.** Requiring an increase in sample complexity is not the only way in which differential privacy comes at a cost. To complement our study of the sample complexity of private learning, we also address whether there is a computational price of differential privacy. For the query release problem, a number of works provide evidence of computational intractability
Motivated by the observation that all known paradigms (statistical query learning and Gaussian elimination) for polynomial-time private learning can be simulated with differential privacy, the initial work of Kasiviswanathan et al. [81] asked an analogous question for private PAC learning: Can private learning be computationally harder than non-private learning?

In Chapter 7, we provide an answer to this question. Namely, we show that under plausible cryptographic assumptions, there exists an efficiently PAC learnable concept class that is computationally infeasible to learn with differential privacy. Our concept class is again based on the class of threshold functions, but combined with the modern cryptographic primitive of order-revealing encryption. Our study naturally leads us to examine the computational complexity of statistical query learning using a different source of hardness than what had previously been considered in the literature. This work is joint with Mark Zhandry (TCC 2016-A) [30].
Chapter 2

Background and Preliminaries

This chapter summarizes the preliminaries in differential privacy, learning theory, and cryptography that will be useful throughout this thesis.

Section 2.1 – Differential Privacy. Here we give the definition of differential privacy and describe some of most basic algorithmic tools and techniques for achieving it.

Section 2.2 – Counting Queries and Accuracy. We describe the query release and sanitization problems for answering a family of counting queries. We also provide the guarantees of some of the state-of-the-art differentially private mechanisms for query release and data sanitization.

Section 2.3 – Sample Complexity. We define sample complexity for private query release. Then we describe a number of reductions showing how sample complexity depends on privacy parameters (ε and δ) as well as various error parameters.

Section 2.4 – Private Distribution Learning. Following the treatment in [27], we describe a distributional version of the private query release problem, which we call the “private distribution learning” problem. This problem is formally closely related to the query release problem, and we give reductions showing that the sample complexities of both problems are essentially equivalent.

Section 2.5 – (Private) PAC Learning. This section introduces the PAC model [117] and the problem of differentially private PAC learning [81]. Mirroring Section 2.4, we discuss the close relationship between private PAC learning and private empirical risk minimization.
Section 2.6 – Fingerprinting Codes. Finally, we introduce one of the main technical tools underlying many of the new lower bounds in this thesis. Fingerprinting codes [21] were introduced by Boneh and Shaw in the context of secure content distribution. In this section, we define fingerprinting codes and a new formulation of “error-robust” fingerprinting codes. We also state theorems regarding the existence of short secure fingerprinting codes.

2.1 Differential Privacy

Let $\mathcal{X}$ be a finite data universe (for instance, we will often take $\mathcal{X} = \{0, 1\}^d$ to be the set of binary strings of length $d$). We define a dataset $D \in \mathcal{X}^n$ to be an ordered tuple of $n$ rows $(x_1, \ldots, x_n) \in \mathcal{X}$ chosen from the data universe $\mathcal{X}$. We say that two datasets $D, D' \in \mathcal{X}^n$ are neighboring if they differ only by a single row, and we denote this relationship by $D \sim D'$.

**Definition 2.1.1** (Differential Privacy [45,47]). Let $A : \mathcal{X}^n \to \mathbb{R}$ be a randomized algorithm. The algorithm $A$ is $(\varepsilon, \delta)$-differentially private if for every two neighboring datasets $D \sim D'$ and every subset $S \subseteq \mathcal{R}$,

$$\Pr[A(D) \in S] \leq e^\varepsilon \Pr[A(D') \in S] + \delta.$$

We refer the reader to the monograph of Dwork and Roth [54] for a textbook treatment of differential privacy and its algorithmic aspects. Below, we summarize only some of the most basic results about differential privacy. We use the following well-known group privacy property of $(\varepsilon, \delta)$-differential privacy. We say that two datasets $D, D'$ are $c$-neighboring if they differ on at most $c$-elements, and denote this relation by $D \sim_c D'$.

**Lemma 2.1.2** ([47]). If $A : \mathcal{X}^n \to \mathcal{R}$ is $(\varepsilon, \delta)$-differentially private, then for every $c \in \mathbb{N}$ and every two $c$-adjacent datasets $D \sim_c D'$, and every $S \subseteq \mathcal{R}$,

$$\Pr[A(D) \in S] \leq e^{c\varepsilon} \Pr[A(D') \in S] + \frac{e^{c\varepsilon} - 1}{e^\varepsilon - 1} \delta.$$

The proof of this lemma follows simply by repeated application of the definition of 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. Here, a real-valued random variable is
distributed as Lap(b) if its probability density function is \( p(x) = \frac{1}{2b} \exp(-\frac{|x|}{b}) \). We say a function \( f : X^n \to \mathbb{R}^m \) has sensitivity \( \Delta \) if for all neighboring \( D, D' \in X^n \), it holds that \( ||f(D) - f(D')||_1 \leq \Delta \).

**Theorem 2.1.3 (The Laplace Mechanism [47]).** Let \( f : X^n \to \mathbb{R}^n \) be a sensitivity \( \Delta \) function. The mechanism \( A \) that on input \( D \in X^n \) adds independent noise with distribution Lap(\( \Delta/\epsilon \)) to each coordinate of \( f(D) \) is \( \epsilon \)-differentially private.

Another important building block is the exponential mechanism of McSherry and Talwar [91]. A quality function \( q : X^* \times \Theta \to \mathbb{N} \) defines an optimization problem over the domain \( X \) and a finite solution set \( \Theta \): Given a dataset \( D \in X^n \), choose \( \theta \in \Theta \) that (approximately) maximizes \( q(D, \theta) \). The exponential mechanism solves such an optimization problem sampling a random \( \theta \in \Theta \) with probability \( \propto \exp(\epsilon \cdot q(D, \theta)/2\Delta q) \). Here, the sensitivity of a quality function, \( \Delta q \), is the maximum over all \( \theta \in \Theta \) of the sensitivity of the function \( q(\cdot, \theta) \).

**Proposition 2.1.4 (Properties of the Exponential Mechanism [91]).**

1. The exponential mechanism is \( (\epsilon, 0) \)-differentially private.

2. Let \( q \) be a quality function with sensitivity at most \( \Delta \). Fix a dataset \( D \in X^n \) and let \( \text{OPT} = \max_{\theta \in \Theta} \{q(D, \theta)\} \). Let \( \beta > 0 \). Then with probability at least \( 1 - \beta \), the exponential mechanism outputs a solution \( \theta \) with

\[
q(D, \theta) \geq \text{OPT} - \frac{2\Delta \ln(|\Theta|/\beta)}{\epsilon}.
\]

We will present algorithms that access their input dataset using (several) differentially private mechanisms and use the following composition theorems to reason about their overall privacy guarantees.

**Lemma 2.1.5 (Composition, e.g. [46]).** Let \( A_1 : X^n \to R_1 \) be \( (\epsilon_1, \delta_1) \)-differentially private. Let \( A_2 : X^n \times R_1 \to R_2 \) be \( (\epsilon_2, \delta_2) \)-differentially private for any fixed value of its second argument. Then the composition \( A(D) = A_2(D, A_1(D)) \) is \( (\epsilon_1 + \epsilon_2, \delta_1 + \delta_2) \)-differentially private.

Inductively applying Lemma 2.1.5 shows that the composition of \( k \) mechanisms, each of which is \( (\epsilon, \delta) \)-differentially private, is \( (k\epsilon, k\delta) \)-differentially private. More sophisticated composition theorems [55] show that differential privacy can actually be taken to degrade with \( \sqrt{k} \), rather than with \( k \).
These results hold even when the sequence of \((\varepsilon, \delta)\)-differentially private mechanisms is not necessarily fixed once and for all, but may instead be chosen adaptively:

**Theorem 2.1.6** (Composition of Differential Privacy \([45, 46, 55]\)). Let \(\varepsilon, \delta, \delta' > 0\). Suppose an algorithm \(A\) accesses its input dataset \(D\) only through \(k\) adaptively chosen executions of \((\varepsilon, \delta)\)-differentially private algorithms. Then \(A\) is

1. \((k\varepsilon, k\delta)\)-differentially private, and
2. \((\varepsilon', k\delta + \delta')\)-differentially private for \(\varepsilon' = \sqrt{2k\ln(1/\delta')} \cdot \varepsilon + k\varepsilon(e^\varepsilon - 1)\).

Here when we refer to “adaptively chosen executions,” we mean that in each of \(k\) rounds, the algorithm \(A\) may select an \((\varepsilon, \delta)\)-differentially private algorithm \(A_k\) to run on the dataset \(D\), where the choice of the algorithm \(A_k\) may depend on the outcomes of all of the previous rounds \(A_1(D), \ldots, A_{k-1}(D)\). We refer the reader to [55] for a formal description of the adaptive composition experiment.

### 2.2 Counting Queries and Accuracy

One focus of this thesis is on the problem of privately answering *counting queries*. A counting query on \(X\) is defined by a predicate \(q : X \to \{0, 1\}\). Abusing notation, we define the evaluation of the query \(q\) on a dataset \(D = (x_1, \ldots, x_n) \in X^n\) to be its average value over the rows,

\[
q(D) = \frac{1}{n} \sum_{i=1}^{n} q(x_i).
\]

An important measure of the complexity of a family of counting queries is a combinatorial quantity called the *Vapnik-Chervonenkis* or *VC* dimension. This quantity characterizes the (non-private) sample complexity of several related statistical learning tasks, and also appears in sample complexity bounds for differentially private algorithms. The VC dimension of a query family \(Q\) is the largest number \(d\) for which the collection of all Boolean functions on \([d] = \{1, \ldots, d\}\) can be embedded in \(Q\). More formally,

**Definition 2.2.1.** Let \(Q\) be a collection of queries over domain \(X\). A set \(S = \{x_1, \ldots, x_d\} \subseteq X\) is *shattered* by \(Q\) if for every \(T \subseteq [d]\) there exists \(q \in Q\) such that \(T = \{i : q(x_i) = 1\}\). The
Vapnik-Chervonenkis (VC) dimension of \( Q \), denoted \( \text{VC}(Q) \), is the cardinality of the largest set \( S \subseteq \mathcal{X} \) that is shattered by \( Q \). (If arbitrarily large finite sets can be shattered by \( Q \), then we say \( \text{VC}(Q) = \infty \).)

We now list a few important examples of counting queries that will appear throughout this thesis.

**Definition 2.2.2 (Point Queries).** Let \( \mathcal{X} \) be any data universe. The class of \textit{point functions} is the set of all queries that evaluate to 1 on exactly one element of \( \mathcal{X} \), i.e. \( \text{Point}_\mathcal{X} = \{ q_x : x \in \mathcal{X} \} \) where \( q_x(y) = 1 \) iff \( y = x \). The VC dimension of \( \text{Point}_\mathcal{X} \) is 1 for any \( \mathcal{X} \).

**Definition 2.2.3 (Threshold Queries).** Let \( \mathcal{X} \) be any totally ordered data universe. The class of \textit{threshold functions} takes the form \( \text{Thresh}_\mathcal{X} = \{ q_y : y \in \mathcal{X} \} \) where \( q_y(x) = 1 \) iff \( x \leq y \). The VC dimension of \( \text{Thresh}_\mathcal{X} \) is 1 for any \( \mathcal{X} \).

Next, we define the family of \( k \)-way marginal queries. For all of our results it will be sufficient to consider only the set of \textit{monotone} \( k \)-way marginals.

**Definition 2.2.4 (Monotone \( k \)-way Marginals).** A \textit{(monotone) \( k \)-way marginal} \( q_S \) over \( \{0, 1\}^d \) is specified by a subset \( S \subseteq [d] \) of size \( |S| \leq k \). It takes the value \( q_S(x) = 1 \) if and only if \( x_i = 1 \) for every index \( i \in S \). The collection of all (monotone) \( k \)-way marginals is denoted by \( \mathcal{M}_{k,d} \).

The \textit{query release problem} for a family \( Q \) of counting queries is to, in a differentially private manner, produce a sequence of answers that closely approximate \( q(D) \) for every \( q \in Q \). We use the following notion of accuracy.

**Definition 2.2.5 (Worst-Case Accuracy for Counting Queries).** Let \( Q \) be a set of counting queries on \( \mathcal{X} \) and \( \alpha, \beta \in [0, 1] \) be parameters. For a dataset \( D \in \mathcal{X}^n \), a sequence of answers \( a = (a_q)_{q \in Q} \in \mathbb{R}^{|Q|} \) is \textit{\( \alpha \)-accurate for} \( Q \) if \( |q(D) - a_q| \leq \alpha \) for all queries \( q \in Q \).

Let \( A : \mathcal{X}^n \to \mathbb{R}^{|Q|} \) be a randomized algorithm. The algorithm \( A \) is \textit{\( (\alpha, \beta) \)-accurate for} \( Q \) if for every \( D \in \mathcal{X}^n \),

\[
\Pr[A(D) \text{ is } \alpha\text{-accurate for } Q] \geq 1 - \beta.
\]

When \( \beta = 1/3 \) we may simply write that \( A \) is \textit{\( \alpha \)-accurate for} \( Q \).

In Chapter 3, we will also be interested in the following relaxed notion of accuracy for counting queries:
Definition 2.2.6 (Average-Case Accuracy for Counting Queries). Let $\mathcal{Q}$ be a set of counting queries on $X$ and $\alpha, \beta, \gamma \in [0, 1]$ be parameters. For a dataset $D \in X^n$, a sequence of answers $a = (a_q)_{q \in \mathcal{Q}} \in \mathbb{R}^{\vert \mathcal{Q} \vert}$ is $(\alpha, \gamma)$-accurate for $\mathcal{Q}$ if $\vert q(D) - a_q \vert \leq \alpha$ for at least a $1 - \gamma$ fraction of queries $q \in \mathcal{Q}$.

Let $A : X^n \rightarrow \mathbb{R}^{\vert \mathcal{Q} \vert}$ be a randomized algorithm. The algorithm $A$ is $(\alpha, \beta, \gamma)$-accurate for $\mathcal{Q}$ if for every $D \in X^n$,

$$\Pr[A(D) \text{ is } (\alpha, \beta, \gamma)\text{-accurate for } \mathcal{Q}] \geq 1 - \beta.$$  

When $\beta = 1/3$ we may simply write that $A$ is $(\alpha, \gamma)$-accurate for $\mathcal{Q}$. (This notation will be used exclusively in Chapter 3.)

In the definition of accuracy, we have assumed that $A$ outputs a sequence of $\vert \mathcal{Q} \vert$ real-valued answers, with $a_q$ representing the answer to $q$. If we are not concerned with the running time of the algorithm, this assumption is without loss of generality: In certain settings, $A$ is allowed to output a “summary” $z \in \mathcal{R}$ for some range $\mathcal{R}$. In this case, we would also require that there exists an “evaluator” $E : \mathcal{R} \times \mathcal{Q} \rightarrow \mathbb{R}$ that takes a summary and a query and returns an answer $E(z, q) = a_q$ that approximates $q(D)$. This extra generality allows $A$ to run in less time than the number of queries it is answering. If we do not bound the running time of $A$ we can convert any such sanitizer to one that outputs a sequence of $\vert \mathcal{Q} \vert$ real-valued answers simply by running the evaluator for every $q \in \mathcal{Q}$.

An important example of such a summary is a synthetic dataset — a dataset $\hat{D} \in X^m$ of “fake” records that nevertheless captures the statistical properties of a sensitive dataset $D$. Given a synthetic dataset $\hat{D}$ and a query $q$, an evaluator $E(\hat{D}, q)$ simply computes $q(\hat{D})$. The problem of privately outputting a synthetic dataset is called the data sanitization problem.

Definition 2.2.7 (Sanitization). An algorithm $A : X^n \rightarrow X^m$ is an $(\alpha, \beta, \gamma)$-accurate sanitizer for a family $\mathcal{Q}$ of counting queries if for every $D \in X^n$, the algorithm $A$ produces a dataset $\hat{D} \in X^m$ such that

$$\Pr[(q(\hat{D}))_{q \in \mathcal{Q}} \text{ is } (\alpha, \gamma)\text{-accurate for } \mathcal{Q}] \geq 1 - \beta.$$  

Here, the probability is taken over the coins of $A$.

In an influential result, Blum, Ligett, and Roth \cite{14} showed that any concept class $\mathcal{Q}$ admits a
differentially private sanitizer for datasets of size $n \geq O(\text{VC}(Q) \log |\mathcal{X}|)$:

**Theorem 2.2.8 ([14]).** For any family $Q$ of counting queries over a domain $\mathcal{X}$, there exists an $(\alpha, \beta)$-accurate and $(\varepsilon, 0)$-differentially private sanitizer $A : \mathcal{X}^n \to \mathcal{X}^m$ for $Q$ when

$$n = O \left( \frac{\text{VC}(Q) \cdot \log |\mathcal{X}| \cdot \log(1/\alpha)}{\alpha^3 \varepsilon} + \frac{\log(1/\beta)}{\alpha \varepsilon} \right),$$

and $m = O(\text{VC}(Q) \log(1/\alpha)/\alpha^2)$.

When relaxing to $(\varepsilon, \delta)$-differential privacy, the private multiplicative weights algorithm of Hardt and Rothblum [76] can sometimes be accurate for smaller datasets (of size roughly $O(\log |Q| \sqrt{\log |\mathcal{X}|})$).

**Theorem 2.2.9 ([76]).** For any concept class $Q$ over a domain $\mathcal{X}$, there exists an $(\alpha, \beta)$-accurate and $(\varepsilon, \delta)$-differentially private sanitizer $A : \mathcal{X}^n \to \mathcal{X}^m$ for $Q$ when

$$n = O \left( \frac{(\log |Q| + \log(1/\beta)) \cdot \sqrt{\log |\mathcal{X}| \cdot \log(1/\delta)}}{\alpha^2 \varepsilon} \right),$$

and $m = O(\text{VC}(Q) \log(1/\alpha)/\alpha^2)$.

### 2.3 Sample Complexity

In this thesis we prove lower bounds on the sample complexity required to simultaneously achieve differential privacy and accuracy.

**Definition 2.3.1 (Sample Complexity).** Let $Q$ be a set of counting queries on $\mathcal{X}$ and let $\alpha, \beta, \gamma > 0$ be parameters, and let $\varepsilon, \delta$ be functions of $n$. We say that $Q$ has sample complexity $n^*$ for $(\alpha, \beta, \gamma)$-accuracy and $(\varepsilon, \delta)$-differential privacy if $n^*$ is the least $n \in \mathbb{N}$ such that there exists an $(\varepsilon, \delta)$-differentially private algorithm $A : \mathcal{X}^n \to \mathbb{R}^{|Q|}$ that is $(\alpha, \beta, \gamma)$-accurate for $Q$.

For context, we can restate some prior results on differentially private counting query release in our sample-complexity terminology. The following upper bounds apply to any worst-case family of counting queries $Q$.

**Theorem 2.3.2 (Combination of [13,14,41,47,53,73,76]).** For every set of counting queries $Q$ on $\mathcal{X}$ and every $\alpha, \delta > 0$, $Q$ has sample complexity at most

$$\min \left\{ \tilde{O} \left( \sqrt{|Q| \log(1/\delta)} \right), \tilde{O} \left( \sqrt{|\mathcal{X}| \log |Q|} \right), O \left( \frac{\text{VC}(Q) \log |\mathcal{X}|}{\alpha^3 \varepsilon} \right), O \left( \frac{\log |Q| \sqrt{\log |\mathcal{X}| \log(1/\delta)}}{\alpha^2 \varepsilon} \right) \right\}$$
for \((\alpha, \beta = 1/3, 0)\)-accuracy and \((\varepsilon, \delta)\)-differential privacy.

We are mostly interested in a setting of parameters where \(\alpha\) is not too small (e.g. constant) and 
\(\log |\mathcal{X}| \ll |\mathcal{Q}| \leq \text{poly}(|\mathcal{X}|)\). In this regime the best-known sample complexity will be achieved by the final expression, corresponding to the private multiplicative weights algorithm [76] using the analysis of [73].

The next theorem shows that, when the data universe is not too small, the private multiplicative weights algorithm is nearly-optimal as a function of \(|\mathcal{Q}|\) and \(1/\alpha\) when each parameter is considered individually.

**Theorem 2.3.3** (Combination of [41, 104]). For every \(s \in \mathbb{N}, \alpha \in (0, 1/4)\), there exists a set of \(s\) counting queries \(\mathcal{Q}\) on a data universe \(\mathcal{X}\) of size \(\max\{\log s, O(1/\alpha^2)\}\) such that \(\mathcal{Q}\) has sample complexity at least

\[
\max \left\{ \Omega \left( \frac{\log |\mathcal{Q}|}{\alpha \varepsilon} \right), \Omega \left( \frac{1}{\alpha^2 \varepsilon} \right) \right\}
\]

for \((\alpha, \beta = 1/3, 0)\)-accuracy and \((\varepsilon, \delta = o(1/n))\)-differential privacy.

### 2.3.1 Dependence of Sample Complexity on \(\varepsilon\) and \(\delta\)

When proving lower bounds, we focus on the case where \(\varepsilon = O(1)\) and \(\delta = o(1/n)\). This setting of parameters is essentially the most-permissive for which \((\varepsilon, \delta)\)-differential privacy is still a meaningful privacy definition. However, pinning down the exact dependence on \(\varepsilon\) and \(\delta\) is still of interest.

**Lemma 2.3.4.** Let \(\mathcal{Q}\) be a family of counting queries over universe \(\mathcal{X}\), and let \(\alpha, \beta, \gamma, \varepsilon, \delta \in [0, 1]\). Suppose \(\mathcal{Q}\) has sample complexity \(n\) for \((\alpha, \beta, \gamma)\)-accuracy and \((1, \delta)\)-differential privacy. Then \(\mathcal{Q}\) has sample complexity

\[
m = O \left( \max \left\{ \frac{n}{\varepsilon}, \frac{\text{VC}(\mathcal{Q}) \log (1/\alpha \beta)}{\alpha^2} \right\} \right)
\]

for \((2\alpha, 2\beta, \gamma)\)-accuracy and \((\varepsilon, \delta)\)-differential privacy.

**Proof.** Let \(\mathcal{A}\) be a \((1, \delta)\)-differentially private algorithm that is \((\alpha, \beta, \gamma)\)-accurate for \(\mathcal{Q}\), and operates on datasets \(D\) of size \(n\). Consider the algorithm \(\tilde{\mathcal{A}}\) that, on input a database \(D' \in \mathcal{X}^m\), samples \(n\) rows (with replacement) from its input dataset and runs \(\mathcal{A}\) on the result. Lemma 2.3.10 proved in Section 2.3.3 below shows that \(\tilde{\mathcal{A}}\) is \((\varepsilon, \delta)\)-differentially private if \(n \leq 6\varepsilon m\).
We now use a generalization argument to show that the accuracy of $A$ on the subsample implies accuracy of $\tilde{A}$ on the database $D'$. Let $D$ denote the random subsample given to $A$. We first argue that $q(D')$ is close to $q(D)$ for most $q \in Q$, and then argue that $q(D)$ is close to $q(D')$. By the utility properties of $A$, with all but probability $\beta$, we have $|a_q - q(D')| \leq \alpha$ for all but a $\gamma$ fraction of $q \in Q$.

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

**Theorem 2.3.5** ([1]). 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 $d = \text{VC}(Q)$, then

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

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

We now state, in a generic fashion, a reduction of Steinke and Ullman [111] that shows how to incorporate a dependence on the privacy parameters $\varepsilon, \delta$ in lower bounds as well. Taken together with Lemma 2.3.4, the special case of the following lemma where we take $\delta = \delta_0$ shows that (for reasonable parameter settings) a query family $Q$ has sample complexity $n$ for $(1, \delta)$-differential privacy if and only if it has sample complexity $\Theta(n/\varepsilon)$ and $(\varepsilon, \delta)$-differential privacy.

**Lemma 2.3.6.** Let $Q$ be a family of counting queries over universe $X$, and let $\alpha, \beta, \gamma, \varepsilon \in [0, 1]$. Suppose $Q$ has sample complexity $n$ for $(\alpha, \beta, \gamma)$-accuracy and $(\ln(\delta/\delta_0), \delta_0)$-differential privacy for some $\delta_0 > \delta > 2^{-n/2}$. Then $Q$ has sample complexity $m = \Omega(\frac{n}{\varepsilon} \cdot \ln(\varepsilon\delta_0/\delta))$ for $(\alpha, \beta, \gamma)$-accuracy and $(\varepsilon, \delta)$-differential privacy.

**Proof.** Let $A : X^m \rightarrow \mathbb{R}^{|Q|}$ be an alleged $(\varepsilon, \delta)$-differentially private and $(\alpha, \beta, \gamma)$-accurate mechanism for answering $Q$. For a parameter $k \in \mathbb{N}$ with $k \leq m/2$ (the condition $\delta > 2^{-n/2}$ ensures this will hold), consider the mechanism $A_k : X^{mk} \rightarrow \mathbb{R}^{|Q|}$, where $m_k = m/k$, defined as follows: On input a dataset $D$, the mechanism $A_k$ makes $k$ copies of $D$ and runs $A$ on the resulting dataset of size $m$.

By group privacy (Lemma 2.1.2), the mechanism $A_k$ is $(\varepsilon_k, \delta_k)$-differentially private for $\varepsilon_k = k\varepsilon$ and $\delta_k = \frac{ek\varepsilon - 1}{e\varepsilon - 1}\delta$. It is also immediate that $A_k$ remains $(\alpha, \beta, \gamma)$-accurate for answering $Q$.  

17
We now argue that for an appropriate choice of the parameter $k$, the mechanism $A_k$ accurately answers $Q$ with sample complexity $n$. Indeed, let $k = \frac{1}{\varepsilon} \cdot \ln(\varepsilon\delta_0/\delta)$. Then

$$\varepsilon_k = \ln(\varepsilon\delta_0/\delta),$$

$$\delta_k = \frac{e^{k\varepsilon} - 1}{e^\varepsilon - 1} \leq \frac{1}{\varepsilon} \cdot e^{k\varepsilon} \cdot \delta \leq \delta_0.$$ 

Hence, we must have $m_k = m \varepsilon / \ln(\varepsilon\delta_0/\delta) \geq n$, or equivalently, $m \geq \frac{n}{\varepsilon} \cdot \ln(\varepsilon\delta_0/\delta)$. □

In order to apply Lemma 2.3.6 to incorporate a dependence on $\delta$ into lower bounds, one needs to start with a meaningful lower bound for $(\varepsilon, \delta_0)$-differential privacy for values of $\varepsilon \gg 1$. Steinke and Ullman showed how to do this for 1-way marginal queries (ultimately yielding a dependence on $\delta$ that looks like a multiplicative $\sqrt{\log(1/\delta)}$), and we do the same for threshold queries in Chapter 4 (where the dependence is more like $\log(1/\delta)$). It remains an interesting open problem to understand how the sample complexity of general query release problems depends on $\delta$.

**Open Problem 2.3.7** (Dependence of sample complexity on $\delta$). Given a query family $Q$, how does the sample complexity of releasing $Q$ with $(1, \delta)$-differential privacy compare to the sample complexity of releasing $Q$ with $(1, \delta_0)$-differential privacy (for some canonical choice of $\delta_0$, e.g. $1/n^2$)?

### 2.3.2 Dependence of Sample Complexity on $\alpha$

We now argue that it is always possible to incorporate a linear dependence on $1/\alpha$ in sample complexity lower bounds:

**Lemma 2.3.8.** Let $Q$ be a family of counting queries over data universe $X$, and let $\beta, \gamma, \varepsilon, \delta \in [0, 1]$. Suppose $Q$ has sample complexity $n$ for $(\alpha_0, \beta, \gamma)$-accuracy and $(\varepsilon, \delta)$-differential privacy, where $\alpha_0 \in [0, 1]$ is a constant. Then $Q$ has sample complexity $m = \Omega(n/\alpha)$ for $(\alpha, \beta, \gamma)$-accuracy and $(\varepsilon, \delta)$-differential privacy.

**Proof.** Let $A : X^m \to \mathbb{R}^{|Q|}$ be an $(\varepsilon, \delta)$-differentially private and $(\alpha, \beta, \gamma)$-accurate mechanism for releasing answers to $Q$. We will use $A$ to construct a mechanism $A' : X^n \to \mathbb{R}^{|Q|}$ achieving constant accuracy $\alpha_0$ on databases of size $n = \lceil m\alpha/\alpha_0 \rceil$. To do so, fix a (publicly known) element $x_0 \in X$. On input a database $D' \in X^n$, the mechanism $A'$ “pads” $D'$ by appending $m - n$ copies of $x_0$, producing
a database $D$. It then runs $\mathcal{A}$ on $D$, obtaining answers $(a_q)_{q \in Q}$. Finally, it releases answers $(a'_q)_{q \in Q}$ where $a'_q = \frac{1}{n}(ma_q - (m - n)q(x_0))$.

The mechanism $\mathcal{A}'$ inherits $(\varepsilon, \delta)$-differential privacy from $\mathcal{A}$, since changing one row of $D'$ changes one row of the padded database $D'$. Now we argue accuracy. Suppose $a_q$ is an answer such that $|a_q - q(D)| \leq \alpha$. Note that by construction, $q(D) = \frac{1}{m}(mq(D') + (m - n)q(x_0))$, and hence $q(D') = \frac{1}{n}(mq(D) - (m - n)q(x_0))$. Thus we have

$$|a'_q - q(D')| = \frac{1}{n}|ma_q - (m - n)q(x_0) - (mq(D) - (m - n)q(x_0))|$$

$$= \frac{m}{n}|a_q - q(D)|$$

$$\leq \frac{m}{n} \cdot \alpha.$$

Taking $n = \lceil m\alpha/\alpha_0 \rceil$ makes this quantity at most $\alpha_0$, completing the proof. $\square$

A linear dependence on $1/\alpha$ is tight for certain query families, such as 1-way marginals. But as we will see in Section 3.4, the correct dependence is on $1/\alpha^2$ for other families of queries, such as $k$-way marginals with $k \geq 2$. By analogy with Open Problem 2.3.7, it is also of interest to understand how sample complexity depends on $\alpha$ in general:

**Open Problem 2.3.9** (Dependence of sample complexity on $\alpha$). Given a query family $Q$, how does the sample complexity of releasing $Q$ with $(\alpha, \beta, \gamma)$-accuracy compare to the sample complexity of releasing $Q$ with $(\alpha_0, \beta, \gamma)$-accuracy for constant $\alpha_0$?

### 2.3.3 Subsampling and Differential Privacy

We now refine an argument that appeared implicitly in [81] to show that $\tilde{\mathcal{A}}$ (from Lemma 2.3.4) is differentially private whenever $\mathcal{A}$ is differentially private. Random sampling with replacement 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.

**Lemma 2.3.10** (Secrecy of the Sample). Fix $\varepsilon, \delta \in [0, 1]$ and let $\mathcal{A}$ be an $(\varepsilon, \delta)$-differentially private...
algorithm operating on datasets of size $n$. For $m \geq 2n$, consider the algorithm $\tilde{A}$ that on input a dataset $D$ of size $m$ subsamples (with replacement) $n$ rows from $D$ and runs $A$ on the result. Then $\tilde{A}$ is $(\tilde{\epsilon}, \tilde{\delta})$-differentially private for

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

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

$$p_k = \Pr[\ell(T) = k] = \binom{n}{k} m^{-k} (1 - 1/m)^{n-k} = \binom{n}{k} (m - 1)^{-k} (1 - 1/m)^n,$$

$$q_k = \Pr[A(D|T) \in S | \ell(T) = k],$$

$$q_k' = \Pr[A(D'|T) \in S | \ell(T) = k].$$

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

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

To do this, we first show that

$$q_k \leq e^{\tilde{\epsilon}} q_{k-1} + \tilde{\delta}.$$ 

To establish this inequality, we define a coupling of the conditional random variables $U = (T|\ell(T) = k)$ and $U' = (T|\ell(T) = k - 1)$ with the property that $U$ and $U'$ are always at Hamming distance $1$ (and hence $D|U$ and $D'|U'$ are neighbors). Specifically, consider the joint distribution $(U, U')$ over $[m]^n \times [m]^n$ sampled as follows. Let $I' \subseteq [n]$ be a random set of indices with $|I'| = k - 1$. Let $i$ be a random index in $[n] - I'$, and let $I = I' \cup \{i\}$. Define $U$ and $U'$ by setting $U_j = m$ for every $j \in I$, setting $U'_j = m$ for every $j \in I'$, setting $U_j = U'_j$ to be a random element of $[m - 1]$ for each $j \notin I'$, and setting $U_i$ to be random element of $[m - 1]$. One can verify that the marginal distributions of $U$ and $U'$ are indeed uniform over the sequences in $[m]^n$ with $\ell(U) = k$ and $\ell(U') = k - 1$, and
moreover that $U$ and $U'$ are always at Hamming distance 1 apart, differing only at index $i$. Thus,

$$q_k = \Pr[A(D|T) \in S|\ell(T) = k]$$

$$= \mathbb{E}_{(U,U')} [\Pr[A(D|U) \in S]]$$

$$\leq \mathbb{E}_{(U,U')} [e^\varepsilon \Pr[A(D|U') \in S] + \delta]$$

$$= e^\varepsilon \Pr[A(D|T) \in S|\ell(T) = k - 1] + \delta$$

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

By induction on $k$,

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

Hence,

$$\Pr[\tilde{A}(D) \in S] = \sum_{k=0}^{n} p_k q_k$$

$$\leq \sum_{k=0}^{n} \binom{n}{k} (m - 1)^{-k} (1 - 1/m)^n \left( e^{k\varepsilon} q_0 + \frac{e^{k\varepsilon} - 1}{e^\varepsilon - 1} \delta \right)$$

$$= q_0 (1 - 1/m)^n \sum_{k=0}^{n} \binom{n}{k} \left( \frac{e^\varepsilon}{m - 1} \right)^k + \frac{\delta}{e^\varepsilon - 1} (1 - 1/m)^n \sum_{k=0}^{n} \binom{n}{k} \left( \frac{e^\varepsilon}{m - 1} \right)^k - \frac{\delta}{e^\varepsilon - 1}$$

$$= q_0 (1 - 1/m)^n \left( 1 + \frac{e^\varepsilon}{m - 1} \right)^n + \frac{\delta}{e^\varepsilon - 1} (1 - 1/m)^n \left( 1 + \frac{e^\varepsilon}{m - 1} \right)^n - \frac{\delta}{e^\varepsilon - 1}$$

$$= q_0 \left( 1 - \frac{1}{m} + \frac{e^\varepsilon}{m} \right)^n + \frac{1 - \frac{e^\varepsilon}{m}}{e^\varepsilon - 1} \frac{\delta}{e^\varepsilon - 1}.$$

Similarly, we also have that

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

Combining inequalities 2.1 and 2.2 we get that

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

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

$$\tilde{\varepsilon} \leq n \cdot \ln \left( 1 + \frac{e^\varepsilon - 1}{m} \right) \leq \frac{6\varepsilon m}{n}.$$
\[
\hat{\delta} \leq \exp(6\varepsilon/n) \frac{1 - \left(1 + \frac{e^{-\varepsilon} - 1}{n}\right)^m}{1 - e^{-\varepsilon}} \cdot \delta + \frac{(1 + \frac{e^{-\varepsilon} - 1}{n})^m - 1}{e^{\varepsilon} - 1} \cdot \delta
\]

\[
\leq \exp(6\varepsilon/n) \frac{1 - \exp\left(\frac{2e^{-\varepsilon} - 1}{n/m}\right)}{1 - e^{-\varepsilon}} \cdot \delta + \frac{\exp\left(\frac{e^{-\varepsilon} - 1}{n/m}\right) - 1}{e^{\varepsilon} - 1} \cdot \delta
\]

\[
\leq \exp(6\varepsilon/n) \frac{2m}{n} \cdot \delta + \frac{2m}{n} \cdot \delta
\]

\[
\leq \exp(6\varepsilon/n) \frac{4m}{n} \cdot \delta.
\]

\[\square\]

### 2.4 Private Distribution Learning

We are also interested in the following distribution learning problem, which can be thought of as a distributional or inference version of the query release problem.

**Definition 2.4.1** (Distribution Learning with respect to \(Q\)). Let \(Q\) be a collection of counting queries on a data universe \(X\). Algorithm \(A\) is an \((\alpha, \beta)\)-accurate distribution learner with respect to \(Q\) with sample complexity \(n\) if for all distributions \(D\) on \(X\), given an input of \(n\) samples \(D = (x_1, \ldots, x_n)\) where each \(x_i\) is drawn i.i.d. from \(D\), algorithm \(A\) outputs a distribution \(D'\) on \(X\) (specified by its PMF) satisfying \(d_Q(D, D') \triangleq \sup_{q \in Q} |E_{x \sim D}[q(x)] - E_{x \sim D'}[q(x)]| \leq \alpha\) 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

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

Second, when \(X\) is a totally ordered domain and \(Q = \text{Thresh}_X\), 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 \(\ell_\infty\) norm. Specifically, we define

**Definition 2.4.2** (Cumulative Distribution Function (CDF)). Let \(D\) be a distribution over a totally ordered domain \(X\). The CDF \(F_D\) of \(D\) is defined by \(F_D(t) = \Pr_{x \sim D}[x \leq t]\). If \(X\) is finite, then any function \(F: X \rightarrow [0, 1]\) that is non-decreasing with \(F(\max X) = 1\) is a CDF.
Definition 2.4.3 (Distribution Learning with respect to Kolmogorov distance). Algorithm $A$ is an $(\alpha, \beta)$-accurate distribution learner with respect to Kolmogorov distance with sample complexity $n$ if for all distributions $\mathcal{D}$ on a totally ordered domain $\mathcal{X}$, given an input of $n$ samples $D = (x_1, \ldots, x_n)$ where each $x_i$ is drawn i.i.d. from $\mathcal{D}$, algorithm $A$ outputs a CDF $F$ with $\sup_{x \in \mathcal{X}} |F(x) - F_D(x)|$ with probability at least $1 - \beta$.

The query release problem for a collection of counting queries $\mathcal{Q}$ is very closely related to the distribution learning problem with respect to $\mathcal{Q}$. 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 the VC dimension of $\mathcal{Q}$: It is known [1] that solving the query release problem on at least $256 \text{VC}(\mathcal{Q}) \ln(48/\alpha\beta)/\alpha^2$ random samples yields an $(\alpha, \beta)$-accurate distribution learner for a query class $\mathcal{Q}$, whereas sample complexity $\Omega(\text{VC}(\mathcal{Q})/\alpha)$ is necessary for this task.

Private Query Release vs. Private Distribution Learning

As discussed above, query release and distribution learning are very similar tasks: A distribution learner can be viewed as an algorithm for query release with small error with respect to the underlying distribution (rather than the fixed input dataset). We show that the sample complexities of the two tasks are in fact equivalent (up to constant factors) under differential privacy.

Theorem 2.4.4. Let $\mathcal{Q}$ be a collection of counting queries over a domain $\mathcal{X}$.

1. If there exists an $(\varepsilon, \delta)$-differentially private algorithm for releasing $\mathcal{Q}$ with $(\alpha, \beta)$-accuracy and sample complexity $n \geq 256 \text{VC}(\mathcal{Q}) \ln(48/\alpha\beta)/\alpha^2$, then there is an $(\varepsilon, \delta)$-differentially private $(3\alpha, 2\beta)$-accurate distribution learner with respect to $\mathcal{Q}$ with sample complexity $n$.

2. If there exists an $(\varepsilon, \delta)$-differentially private $(\alpha, \beta)$-accurate distribution learner with respect to $\mathcal{Q}$ with sample complexity $n$, then there is an $(\varepsilon, \delta)$-differentially private query release algorithm for $\mathcal{Q}$ with $(\alpha, \beta)$-accuracy and sample complexity $9n$.

The first direction follows from a standard generalization bound, showing that if a given dataset $D$ contains (enough) i.i.d. samples from a distribution $\mathcal{D}$, then (with high probability) accuracy
with respect to \( D \) implies accuracy with respect to \( D \). We remark that the sample complexity lower bound on \( n \) required to apply item 1 of Theorem 2.4.4 does not substantially restrict its applicability: It is known that an \((\varepsilon, \delta)\)-differentially private algorithm for releasing \( Q \) always requires sample complexity \( \Omega(\text{VC}(Q)/\alpha \varepsilon) \) anyway [14].

**Proof of Theorem 2.4.4, item 1.** Suppose \( \tilde{A} \) is an \((\varepsilon, \delta)\)-differentially private algorithm for releasing \( Q \) with \((\alpha, \beta)\)-accuracy and sample complexity \( n \geq 256 \text{VC}(Q) \ln(48/\alpha \beta)/\alpha^2 \). Fix a distribution \( D \) over \( X \) and consider a dataset \( D \) containing \( n \) i.i.d. samples from \( D \). Define the algorithm \( A \) that on input \( D \) runs \( \tilde{A} \) on \( D \) to obtain answers \( a_q \) for every query \( q \in Q \). Afterwards, algorithm \( A \) uses linear programming [50] to construct a distribution \( D' \) that such that \( |a_q - q(D')| \leq \alpha \) for every \( q \in Q \), where \( q(D') \triangleq \mathbb{E}_{x \sim D}[q(x)] \). This reconstruction always succeeds as long as the answers \( \{a_q\} \) are \( \alpha \)-accurate, since the empirical distribution of \( D \) is a feasible point for the linear program. Note that \( A \) is \((\varepsilon, \delta)\)-differentially private (since it is obtained by post-processing \( \tilde{A} \)).

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

\[
|q(D') - q(D)| \leq |q(D') - a_q| + |a_q - q(D)| \leq 2\alpha.
\]

for every \( q \in Q \).

We now use the generalization Theorem 2.3.5 to show that (w.h.p.) \( q(D) \) is close to \( q(D) \) for every \( q \in Q \). Theorem 2.3.5 together with the fact that \( n \geq 256 \text{VC}(Q) \ln(48/\alpha \beta)/\alpha^2 \) shows that except with probability at least \( 1 - \beta \) we have that \( |q(D) - q(D)| \leq \alpha \) for every \( q \in Q \). By a union bound (and the triangle inequality) we get that \( A \) is \((3\alpha, 2\beta)\)-accurate. \( \square \)

In the special case where \( Q = \text{Thresh}_X \) 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 [43, 89] in place of Theorem 2.3.5.

**Theorem 2.4.5.** If there exists an \((\varepsilon, \delta)\)-differentially private algorithm for releasing \( \text{Thresh}_X \) over a totally ordered domain \( X \) with \((\alpha, \beta)\)-accuracy and sample complexity \( n \geq 2 \ln(2/\beta)/\alpha^2 \), then there is an \((\varepsilon, \delta)\)-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 2.4.6.** Suppose \( A \) is an \((\varepsilon, \delta)\)-differentially private and \((\alpha, \beta)\)-accurate distribution learner with respect to a concept class \( Q \) with sample complexity \( n \). Then there is an \((\varepsilon, \delta)\)-differentially private algorithm \( \tilde{A} \) for releasing \( Q \) with \((\alpha, \beta)\)-accuracy and sample complexity \( 9n \).

To construct the algorithm \( \tilde{A} \), we note that a distribution learner must perform well on the uniform distribution on the rows of any fixed dataset, and thus must be useful for releasing accurate answers for queries on such a dataset. Hence if we have a distribution learner \( A \), the mechanism \( \tilde{A} \) that samples \( m \) rows (with replacement) from its input dataset \( D \in (\mathcal{X} \times \{0, 1\})^n \) and runs \( A \) on the result should output accurate answers for queries with respect to \( D \). The resulting mechanism remains differentially private by the secrecy-of-the-sample lemma (Lemma 2.3.10).

**Proof.** Consider a dataset \( D \in \mathcal{X}^m \). Let \( \mathcal{D} \) denote the uniform distribution over the rows of \( D \), and let \( \mathcal{D}' \) be the distribution learned. Consider the algorithm \( \tilde{A} \) that subsamples (with replacement) \( n \) rows from \( D \) and runs \( A \) on it to obtain a distribution \( \mathcal{D}' \). Afterwards, algorithm \( \tilde{A} \) answers every threshold query \( q \in Q \) with \( a_q = q(\mathcal{D}') \triangleq \mathbb{E}_{x \sim \mathcal{D}}[q(x)] \). Note that the mechanism \( \tilde{A} \) is \((\varepsilon, \delta)\)-differentially private by Lemma 2.3.10.

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 \( A \) is such that for every \( q \in Q \)

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

showing that \( \tilde{A} \) is \((\alpha, \beta)\)-accurate. \(\square\)

### 2.5 (Private) PAC Learning

#### 2.5.1 Definitions

A concept \( c : \mathcal{X} \rightarrow \{0, 1\} \) is a predicate that labels examples taken from the domain \( \mathcal{X} \). A *concept class* \( C \) over \( \mathcal{X} \) is a set of concepts over the domain \( \mathcal{X} \). For some number \( n \), a learner is given examples of the form \((x_i, c(x_i))\), where each \( x_i \) is sampled i.i.d. from an arbitrary unknown probability distribution \( \mathcal{D} \) over \( \mathcal{X} \), and labeled by a *target concept* \( c \in C \). The goal of the learner is to output a *hypothesis* \( h \) that approximates the target concept with respect to the distribution \( \mathcal{D} \). More precisely,
Definition 2.5.1. The generalization error of a hypothesis \( h : \mathcal{X} \to \{0, 1\} \) (with respect to a target concept \( c \) and distribution \( \mathcal{D} \)) is defined by \( \text{err}_\mathcal{D}(c, h) = \Pr_{x \sim \mathcal{D}}[h(x) \neq c(x)] \). If \( \text{err}_\mathcal{D}(c, h) \leq \alpha \) we say that \( h \) is an \( \alpha \)-good hypothesis for \( c \) on \( \mathcal{D} \).

Definition 2.5.2 (PAC Learning [117]). Algorithm \( A \) is an \((\alpha, \beta)\)-accurate PAC learner for a concept class \( C \) over \( \mathcal{X} \) using hypothesis class \( \mathcal{H} \) with sample complexity \( n \) if for all target concepts \( c \in C \) and all distributions \( \mathcal{D} \) on \( \mathcal{X} \), given an input of \( n \) samples \( S = ((x_i, c(x_i)), \ldots, (x_n, c(x_n))) \), where each \( x_i \) is drawn i.i.d. from \( \mathcal{D} \), algorithm \( A \) outputs a hypothesis \( h \in \mathcal{H} \) satisfying \( \Pr[\text{err}_\mathcal{D}(c, h) \leq \alpha] \geq 1 - \beta \).

The probability is taken over the random choice of the examples in \( S \) and the coin tosses of the learner \( A \). If \( \mathcal{H} \subseteq C \) then \( A \) is called proper, otherwise, it is called improper.

Definition 2.5.3. The empirical error of a hypothesis \( h \) on a labeled sample \( S = ((x_1, \ell_1), \ldots, (x_n, \ell_n)) \) is \( \text{err}_S(h) = \frac{1}{n} \sum_{i} [h(x_i) \neq \ell_i] \). If \( \text{err}_S(h) \leq \alpha \) we say \( h \) is \( \alpha \)-consistent with \( S \).

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

A private learner is a PAC learner that is differentially private. Following [81], we consider algorithms \( A : (\mathcal{X} \times \{0, 1\})^n \to \mathcal{H} \), where \( \mathcal{H} \) is a hypothesis class, and require that

1. \( A \) is an \((\alpha, \beta)\)-accurate PAC learner for a concept class \( C \) with sample complexity \( n \), and
2. \( A \) is \((\varepsilon, \delta)\)-differentially private.

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

Recall the relationship between distribution learning and query release, where accuracy is measured with respect to the underlying distribution in the former and with respect to the fixed
input dataset in the later. Analogously, we now define the notion of an *empirical learner* which is similar to a PAC learner where accuracy is measured with respect to the fixed input dataset.

**Definition 2.5.4 (Empirical Learner).** Algorithm $\mathcal{A}$ is an $(\alpha, \beta)$-accurate empirical learner for a concept class $\mathcal{C}$ over $X$ using hypothesis class $\mathcal{H}$ with sample complexity $n$ if for every $c \in C$ and for every dataset $S = ((x_1, c(x_1)), \ldots, (x_n, c(x_n))) \in (X \times \{0, 1\})^n$ algorithm $\mathcal{A}$ outputs a hypothesis $h \in \mathcal{H}$ satisfying $\Pr[err_S(c, h) \leq \alpha] \geq 1 - \beta$.

The probability is taken over the coin tosses of $A$.

**Private PAC Learning vs. Private Empirical Learning**

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 empirical risk minimization:

**Lemma 2.5.5 ([15]).** Any algorithm $\mathcal{A}$ for empirically learning a concept class $\mathcal{C}$ with $(\alpha, \beta)$-accuracy is also a $(2\alpha, \beta + \beta')$-accurate PAC learner for $\mathcal{C}$ when given at least $\max\{n, O(\text{VC}(\mathcal{C}) \log(1/\alpha\beta')/\alpha)\}$ samples.

Note that the sample complexity of differentially private empirical learning is already $n = \Omega\left(\frac{1}{\alpha\epsilon} \text{VC}(\mathcal{C})\right)$ using an analysis similar to the one in [14]. Hence, when applying Lemma 2.5.5 to a private empirical learner with sample complexity $n$, the maximum is not much larger than $n$.

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 2.5.6.** Suppose $\mathcal{A}$ is an $(\epsilon, \delta)$-differentially private $(\alpha, \beta)$-accurate PAC learner for a concept class $\mathcal{C}$ with sample complexity $n$. Then there is an $(\epsilon, \delta)$-differentially private $(\alpha, \beta)$-accurate empirical learner for $\mathcal{C}$ with sample complexity $m = 9n$. Moreover, if $\mathcal{A}$ is proper, then so is the resulting empirical learner.

**Proof.** Consider a dataset $S = \{(x_i, y_i)\} \in (X \times \{0, 1\})^m$. Let $\mathcal{D}$ denote the uniform distribution over the rows of $S$. Then drawing $n$ i.i.d. samples from $\mathcal{D}$ is equivalent to subsampling $n$ rows of $S$ (with replacement). Consider the algorithm $\tilde{\mathcal{A}}$ that subsamples (with replacement) $n$ rows from $S$.
and runs $A$ on it. Then with probability at least $1 - \beta$, algorithm $A$ outputs an $\alpha$-good hypothesis on $D$, which is in turn an $\alpha$-consistent hypothesis for $S$. Moreover, by Lemma 2.3.10 (Secrecy of the Sample), algorithm $A$ is $(\varepsilon, \delta)$-differentially private. 

### 2.6 Fingerprinting Codes

Fingerprinting codes were introduced by Boneh and Shaw [21] to address the problem of watermarking digital content. Roughly speaking, a (fully-collusion-resilient) fingerprinting code is a way of generating codewords for $n$ users in such a way that any codeword can be uniquely traced back to a user. Each legitimate copy of a piece of digital content has such a codeword hidden in it, and thus any illegal copy can be traced back to the user who copied it. Moreover, even if an arbitrary subset of the users collude to produce a copy of the content, then under a certain marking assumption, the codeword appearing in the copy can still be traced back to one of the users who contributed to it. The standard marking assumption is that if every colluder has the same bit $b$ in the $j$-th bit of their codeword, then the $j$-th bit of the “combined” codeword in the copy they produce must be also $b$. We refer the reader to the original paper of Boneh and Shaw [21] for the motivation behind the marking assumption and an explanation of how fingerprinting codes can be used to watermark digital content.

More precisely, a fingerprinting code is a pair of randomized algorithms $(Gen, Trace)$. The code generator $Gen$ outputs a codebook $C \in \{0, 1\}^{n \times d}$. Each row $c_i$ of $C$ is the codeword of user $i$. For a subset of users $S \subseteq [n]$, we use $C_S \in \{0, 1\}^{S \times d}$ to denote the set of codewords of users in $S$. The parameter $d$ is called the length of the fingerprinting code.

The security property of fingerprinting codes asserts that any codeword can be “traced” to a user $i \in [n]$. Moreover, we require that the fingerprinting code is “fully-collusion-resilient”—even if any “coalition” of users $S \subseteq [n]$ gets together and “combines” their codewords in any way that respects certain constraints known as a marking assumption, then the combined codeword $c'$ can be traced to a user $i \in S$. That is, there is a tracing algorithm $Trace$ that takes as inputs the codebook and combined codeword $c'$ and outputs either a user $i \in [n]$ or $\perp$, and we require that if $c'$ satisfies the constraints, then $Trace(C, c') \in S$ with high probability. Moreover, $Trace$ should accuse an innocent user, i.e. $Trace(C, c') \in [n] \setminus S$, with very low probability. When designing fingerprinting codes, one
tries to make the marking assumption on the combined codeword as weak as possible.

The basic marking assumption is that each bit of the combined word $c'$ must match the corresponding bit for some user in $S$. Formally, for a codebook $C \in \{0,1\}^{n \times d}$, and a coalition $S \subseteq [n]$, we define the set of feasible codewords for $C_S$ to be

$$F(C_S) = \left\{ c' \in \{0,1\}^d \mid \forall j \in [d], \exists i \in S, c'_j = c_{ij} \right\}.$$ 

Observe that the combined codeword is only constrained on coordinates $j$ where all users in $S$ agree on the $j$-th bit.

We are now ready to formally define a fingerprinting code.

**Definition 2.6.1 (Fingerprinting Codes).** For any $n, d \in \mathbb{N}$, $\xi \in (0,1]$, a pair of algorithms $(\text{Gen}, \text{Trace})$ is an $(n,d)$-fingerprinting code with security $\xi$ if $\text{Gen}$ outputs a codebook $C \in \{0,1\}^{n \times d}$ and for every (possibly randomized) adversary $A_{FP}$, and every coalition $S \subseteq [n]$, if we set $c' \leftarrow_r A_{FP}(C_S)$, then

1. $\Pr[c' \in F(C_S) \land \text{Trace}(C,c') = \bot] \leq \xi$,

2. $\Pr[\text{Trace}(C,c') \in [n] \setminus S] \leq \xi$,

where the probability is taken over the coins of $\text{Gen}$, $\text{Trace}$, and $A_{FP}$. The algorithms $\text{Gen}$ and $\text{Trace}$ may share a common state.

The common state between $\text{Gen}$ and $\text{Trace}$ should be thought of as auxiliary information about the realization of $C$ that may help $\text{Trace}$ identify a guilty pirate. Formally, we could model this shared state by having $\text{Gen}$ output an additional string $\text{aux}$ that is given to $\text{Trace}$ but not to $A_{FP}$. We make this additional shared state implicit to reduce notational clutter (but we choose to make the $C$ explicit to emphasize that $\text{Trace}$ depends on the codebook).

We also remark that our lower bounds based on fingerprinting codes will only require collusion resilience against coalitions $S$ of size $|S| \geq n - 1$. Our choice to state Definition 2.6.1 using resilience against arbitrary coalitions is more consistent with the literature on fingerprinting codes.

The original construction of fingerprinting codes, due to Boneh and Shaw, is simple and intuitive. It yields an $(n,d)$-fingerprinting code with $n \approx d^{1/3}$. We describe this construction in Chapter 6, where its specific structural properties will be useful in proving lower bounds for privately learning
multiple concepts. Tardos [112] constructed a family of fingerprinting codes with a nearly optimal number of users \( n \) for a given length \( d \).

**Theorem 2.6.2 ([112]).** For every \( d \in \mathbb{N} \), and \( \xi \in [0, 1] \), there exists an \((n,d)\)-fingerprinting code with security \( \xi \) for

\[
n = n(d, \xi) = \tilde{\Omega}(\sqrt{d/\log(1/\xi)}).
\]

As we will see in Chapter 3, fingerprinting codes satisfying Definition 2.6.1 imply lower bounds on the sample complexity for releasing 1-way marginals with \((\alpha,0)\)-accuracy (accuracy for every query). In order to prove sample-complexity lower bounds for \((\alpha,\beta,\gamma)\)-accuracy with \( \gamma > 0 \), we will need fingerprinting codes satisfying a stronger security property. Specifically, we will expand the feasible set \( F(C_S) \) to include all codewords that satisfy most feasibility constraints, and require that even codewords in this expanded set can be traced. Formally, for any \( \gamma \in [0, 1] \), we define

\[
F_\gamma(C_S) = \left\{ c' \in \{0,1\}^d \mid \Pr_{j \leftarrow [d]} [\exists i \in S, c'_{ij} = c_{ij}] \geq 1 - \gamma \right\}.
\]

Observe that \( F_0(C_S) = F(C_S) \).

**Definition 2.6.3 (Error-Robust Fingerprinting Codes).** For any \( n, d \in \mathbb{N} \), \( \xi, \gamma \in [0,1] \), a pair of algorithms \((Gen, Trace)\) is an \((n,d)\)-fingerprinting code with security \( \xi \) robust to a \( \gamma \) fraction of errors if \( Gen \) outputs a codebook \( C \in \{0,1\}^{n \times d} \) and for every (possibly randomized) adversary \( A_{FP} \), and every coalition \( S \subseteq [n] \), if we set \( c' \leftarrow_{\text{ran}} A_{FP}(C_S) \), then

1. \( \Pr[c' \in F_\gamma(C_S) \land Trace(C, c') = \bot] \leq \xi \),
2. \( \Pr[Trace(C, c') \in [n] \setminus S] \leq \xi \),

where the probability is taken over the coins of \( Gen, Trace \), and \( A_{FP} \). The algorithms \( Gen \) and \( Trace \) may share a common state.

Boneh and Naor [19] introduced a different notion of fingerprinting codes robust to adversarial “erasures”. In their definition, the adversary is allowed to output a string in \( \{0,1,?\}^d \), and in order to trace they require that the fraction of ? symbols is bounded away from 1 and that any non-? symbols respect the basic feasibility constraint. For this definition, constructions with nearly-optimal length \( d = \tilde{O}(n^2) \), robust to a \( 1 - o(1) \) fraction of erasures are known [17]. In contrast, our codes
need to be robust to adversarial “errors.” Robustness to a \( \gamma \) fraction of errors can be seen to imply robustness to nearly a \( 2\gamma \) fraction of erasures but the converse is false. Thus for corresponding levels of robustness our definition is strictly more stringent.

2.6.1 Constructing (Robust) Fingerprinting Codes

Building on the fingerprinting code of Tardos [112], we constructed in [29] (the work on which Chapter 3 is based) a fingerprinting code robust to some fixed constant fraction of errors. Steinke and Ullman [111] subsequently refined our work to obtain robustness against an optimal fraction of errors. Moreover, an extension of their techniques [56] (Lemma 5.4.6) will itself be important when we use fingerprinting codes in a non-black-box way in Chapter 5. While we will not prove Theorem 2.6.4 below, we will describe the construction and highlight some of the ideas of the proof.

**Theorem 2.6.4 ([111]).** For every \( d \in \mathbb{N} \), and \( \xi \in (0, 1) \), there exists an \((n, d)\)-fingerprinting code with security \( \xi \) robust to a \( \gamma \) fraction of errors for

\[
n = n(d, \xi, \gamma) = \Omega \left( \sqrt{\frac{d (\frac{1}{2} - \gamma)^4}{\log(1/\xi)}} \right).
\]

**Construction.** We now describe the construction of a code of length \( d \) for \( n \) users. For convenience, and for consistency with [56, 111], we consider codes defined over \( \{\pm 1\} \) rather than over \( \{0, 1\} \).

Given parameters \( n, \gamma \), let \( a = (1/2 - \gamma)/4n \) and define a distribution \( P \) supported on \((a, 1-a)\) with probability density function \( \mu(p) \propto \sqrt{p(1-p)} \). Define a distribution \( \tilde{P} \) on \([0, 1]\) that samples \( p \sim P \) with probability \((1/2 - \gamma)/2\), and makes a uniform choice of either 0 or 1 with the remaining probability.

**Gen(1\(^n\)).** Let \( d = O(n^2 \log(1/\xi)/(1/2 - \beta)^4) \). To sample a codebook \( C \in (\{\pm 1\}^d)^n \), first sample parameters \( p_1, \ldots, p_d \) independently and uniformly at random from \( \tilde{P} \). For each \( j = 1, \ldots, d \), sample the entries \( c_{1,j}, \ldots, c_{n,j} \in \{\pm 1\} \) of column \( j \) independently such that \( \Pr[c_{i,j} = 1] = p_j \) and \( \Pr[c_{i,j} = -1] = 1 - p_j \). The state \( Gen \) shares with \( Trace \) consists of the column biases \( p_1, \ldots, p_d \).

**Trace\((C, c')\).** Given column biases \( p_1, \ldots, p_d \), codebook \( C \in (\{\pm 1\}^d)^n \), and a pirate codeword
$c' \in \{\pm 1\}^d$, compute for each user $i = 1, \ldots, n$ a "score"

$$s_i(c') = \sum_{j=1}^{d} c'_j \phi^{p_j}(c_{i,j}),$$

where $\phi^p \equiv 0$ if $p = 0$ or $p = 1$, and $\phi^p(1) = \sqrt{(1-p)/p}$ and $\phi^p(-1) = -\sqrt{p/(1-p)}$ otherwise.

Set a threshold $\tau = O(n \log(1/\xi)/(1/2 - \beta)^2)$. Output the identity of any user $i$ for which $s_i(c') > \tau$ (or ⊥ if no such user is found).

**Proof idea.** For intuition as to why this code is secure, let us first focus our attention on the case where $\gamma = 0$, where it suffices to think of the parameters $p_1, \ldots, p_d$ as being sampled from the distribution $\mathcal{P}$ instead of the modified distribution $\hat{\mathcal{P}}$. The codebook $C$ consists of $d$ columns, each of which is equipped with a parameter $p_j \in (0, 1)$. The entries of column $j$ are sampled independently with bias $p_j$. To trace a pirate codeword $c'$, the algorithm Trace uses the parameters $p_j$ to compute, for each user $i$, a weighted correlation $s_i(c')$ with user $i$'s codeword $c_i$. It then accuses a user whose correlation with $c'$ is sufficiently high. The weighting of this correlation measure is set up to give more weight to indices $j$ where $c'_j$ agrees with an "atypical" value of $c_{i,j}$ (relative to the column bias $p_j$). For example, if a column bias $p_j$ is close to 0, but $c'_j = c_{i,j} = 1$, then the score function $s_i(c')$ takes on a significant contribution from index $j$.

Recall that the security of a fingerprinting code requires two properties: "completeness", i.e. that a guilty user is accused with high probability; and "soundness", i.e. that an innocent user is accused with negligible probability. Soundness is the easier condition to prove. Since the rows of the codebook are independent, the codeword of any user $i$ has expected score zero, even conditioned on the realizations of the all of the other users’ codewords. A concentration argument then shows that a coalition of users cannot frame a user $i$ who is not part of the coalition, since no matter what strategy a pirate coalition uses, user $i$’s score will be lower than the threshold $\tau$ with overwhelming probability.

The proof of completeness amounts to showing that, for any pirate algorithm producing a feasible pirate codeword $c'$, there must be some user $i$ for which the score $s_i(c') > \tau$. At the heart of Steinke and Ullman’s proof [111] of completeness is the following lemma, showing that each bit of a feasible pirate codeword contributes to the expected score of some user:
Lemma 2.6.5. Let $f : \{\pm 1\}^n \rightarrow [-1,1]$ be any function such that $f(-1^n) = -1$ and $f(1^n) = 1$. Suppose $p$ is sampled from the distribution $P$ and $c \in \{\pm 1\}^n$ is a vector of $n$ independent bits, where $\Pr[c_i = 1] = p$ and $\Pr[c_i = -1] = 1 - p$. Then

$$\mathbb{E}_{p,c} \left[ f(c) \cdot \sum_{i \in [n]} \phi^p(c_i) \right] \geq \frac{2 - 4na}{\pi} \quad (= \Omega(1)).$$

Thinking of $f$ as a pirate strategy restricted to a single column of the codebook, the lemma says that each column contributes $\Omega(1)$ expected correlation to the sum of the users’ scores. By averaging, there must be some user who receives expected correlation $\Omega(1/n)$. Thus, as long as $d \geq \tilde{O}(n^2)$, some user has a total expected score across all columns of at least $\tilde{O}(n)$. Concentration arguments can then be used to show that it is also the case that with high probability, some user has score at least $\tau = \tilde{O}(n)$, which justifies completeness.

We now turn our attention to the case where $\gamma > 0$, where we demand that the fingerprinting code is robust to the pirates introducing errors to a $\gamma$ fraction of the entries of their pirate codeword. To motivate the choice of the distribution $\tilde{P}$ for $\gamma > 0$, let us first see why the base distribution $P$ does not yield a code robust to a constant fraction of errors. The reason is that the only way in which a pirate coalition may introduce an error is to put a $-1$ in a column containing only 1’s or vice versa (recall that the set of codewords, $C \in \{-1,1\}^{n \times d}$, can be viewed as an $n \times d$ matrix). We call such columns “marked columns.” Thus, if the adversary is allowed to introduce $\geq m$ errors where $m$ is the number of marked columns then he can simply ignore the codewords and output either the all-$(-1)$ or all-$(+1)$ codeword, which cannot be traced. Thus, in order to tolerate a $\gamma$ fraction of errors, it is necessary that $m \geq \gamma d$ where $d$ is the length of the codeword, which is not the case when the biases are sampled from the distribution $P$. However, the construction outlined above does satisfy a weaker form of robustness. Namely, it secure even when the adversary is allowed to introduce $\approx \gamma m$ errors, rather than $\gamma d$ errors. The idea is that even when a small fraction of errors are introduced to the marked columns, the remaining marked columns still contribute significantly to the score of some (guilty) user.

To take advantage of this weak form of robustness, we consider the modified distribution $\tilde{P}$, which can be thought of as introducing a large number of “fake” marked columns to a codebook generated using the base distribution $P$. Note that a pirate adversary cannot distinguish real marked
columns from these fake columns. Therefore, the fraction of errors in the real marked columns must be close to the fraction of errors that are either real and marked or fake. Since the total fraction of errors in the entire codebook is at most $\gamma$, the fraction of errors in real marked columns is not much larger than $\gamma$ with high probability. This is enough for the weakly robust tracing procedure to be secure.
Chapter 3

Fingerprinting Codes and the Sample Complexity of Releasing Marginals

Consider a family of counting queries $Q$ defined over the data universe $X = \{0, 1\}^d$. Recall that even for a worst-case family of queries $Q$, releasing accurate answers to $Q$ on a population (i.e. the distribution learning problem with respect to $Q$) requires sample complexity at most $O(\log |Q|)$. The results stated in Chapter 2, culminating with the private multiplicative weights algorithm, show that the sample complexity of releasing answers to $Q$ with differential privacy is at most $O(\sqrt{d} \log |Q|)$. Hence, while the “price of differential privacy” is small for low-dimensional datasets, it may be high for datasets with a large number of attributes.

For instance, if we simply want to estimate the mean of each of $d$ binary attributes on a population without a privacy guarantee, then $\Theta(\log d)$ samples are necessary and sufficient to get statistical accuracy. However, the best known ($\varepsilon, \delta$)-differentially private algorithm (indeed, the Laplace mechanism!) requires $\Omega(\sqrt{d})$ samples. In the special case of pure ($\varepsilon, 0$)-differential privacy, a tight lower bound of $\Omega(d \log |Q|)$ is known ([74], using the techniques of [78]). However, for the general case of approximate ($\varepsilon, \delta$)-differential privacy the best known lower bound is $\Omega(\log |Q|) = \Omega(\log d)$ [41]. More generally, prior to the work in this chapter, there were no known lower bounds separating the sample complexity of ($\varepsilon, \delta$)-differential privacy from the sample complexity required for statistical accuracy alone.
3.1 Results and Techniques

In this chapter, we almost completely close the gap between upper and lower bounds on the sample complexity of answering a worst-case family of queries. In particular, we show that there is indeed a “price of approximate differential privacy” for high-dimensional datasets.

**Theorem 3.1.1 (Informal).** Any algorithm that takes as input a dataset $D \in (\{0, 1\}^d)^n$, satisfies approximate differential privacy, and estimates the mean of each of the $d$ attributes to within error $\pm 1/3$ requires sample complexity $n \geq \tilde{\Omega}(\sqrt{d})$.

We establish this lower bound using fingerprinting codes, described in Section 2.6 and introduced by Boneh and Shaw [21] for the problem of watermarking copyrighted content. Specifically, we use Tardos’ construction of optimal fingerprinting codes [112]. The use of “secure content distribution schemes” to prove lower bounds for differential privacy originates with the work of Dwork et al. [50], who used cryptographic “traitor-tracing schemes” to prove computational hardness results for differential privacy. Extending this connection, Ullman [115] used fingerprinting codes to construct a novel traitor-tracing scheme and obtain a strong computational hardness result for differential privacy.\(^1\) Here we show that a *direct* use of fingerprinting codes yields information-theoretic lower bounds on sample complexity.

Using the additional structure of Tardos’ fingerprinting code, we are able to prove *statistical minimax lower bounds* for inferring the marginals of a product distribution from samples while guaranteeing differential privacy for the sample. Specifically, suppose the dataset $D \in (\{0, 1\}^d)^n$ consists of $n$ independent samples from a product distribution over $\{0, 1\}^d$ such that the $i$-th coordinate of each sample is set to 1 with probability $p_i$, for some unknown $p = (p_1, \ldots, p_d) \in [0, 1]^d$.

We show that if there exists a differentially private algorithm that takes such a dataset as input, satisfies approximate differential privacy, and outputs $\hat{p}$ such that $\|\hat{p} - p\|_\infty \leq 1/3$, then $n \geq \tilde{\Omega}(\sqrt{d})$. Statistical minimax bounds of this type for differentially private inference problems were first studied by Duchi, Jordan, and Wainwright [42], who proved minimax bounds for algorithms that satisfy the stronger constraint of *local, pure differential privacy*.

We then give a *composition theorem* that allows us to combine Theorem 3.1.1 with other sample

---

\(^1\)In fact, one way to prove Theorem 3.1.1 is by replacing the one-way functions in [115] with a random oracle, and thereby obtain an information-theoretically secure traitor-tracing scheme.
complexity lower bounds to obtain even stronger lower bounds. For example, we can combine our new lower bound of $\tilde{\Omega}(\sqrt{d})$ with (a variant of) the known $\Omega(\log |Q|)$ lower bound to obtain a nearly-optimal sample complexity lower bound of $\tilde{\Omega}(\sqrt{d} \log |Q|)$ for some families of queries.

More generally, we can consider how the sample complexity changes if we want to answer counting queries accurately to within $\pm \alpha$. As above, if we assume the dataset contains samples from a population, and require only that the answers to queries on the sampled dataset and the population are close, to within $\pm \alpha$, then $\Theta(\log |Q|/\alpha^2)$ samples are necessary and sufficient for just statistical accuracy. When $|Q|$ is large (relative to $d$ and $1/\alpha$), the best sample complexity for differential privacy is again achieved by the private multiplicative weights algorithm, and is $O(\sqrt{d} \log |Q|/\alpha^2)$. On the other hand, the best known lower bound is $\Omega(\max\{\log |Q|/\alpha, 1/\alpha^2\})$, which follows from the techniques of [41]. Using our composition theorem, as well as our new lower bound, we are able to obtain a nearly-optimal sample complexity lower bound in terms of all these parameters.

In fact, we are able to prove our lower bound for the well-studied family of $k$-way marginal queries, also known as $k$-way conjunction queries (see e.g. [3, 32, 52, 72, 82, 114]). Recall that a $k$-way marginal query on a dataset $D \in (\{0,1\}^d)^n$ is specified by a set $S \subseteq [d]$, $|S| \leq k$, and a pattern $t \in \{0,1\}^{|S|}$ and asks “What fraction of records in $D$ has each attribute $j$ in $S$ set to $t_j$?” The number of $k$-way marginal queries on $\{0,1\}^d$ is about $2^k \binom{d}{k}$. For the special case of $k = 1$, the queries simply ask for the mean of each attribute, which was discussed above.

**Theorem 3.1.2 (Informal).** Any algorithm that takes a dataset $D \in (\{0,1\}^d)^n$, satisfies approximate differential privacy, and outputs an approximate answer to each of the $k$-way marginal queries to within $\pm \alpha$ (for $\alpha$ smaller than some universal constant and larger than an inverse polynomial in $d$) requires $n \geq \tilde{\Omega}(k\sqrt{d}/\alpha^2)$.

Since the number of $k$-way marginal queries is about $2^k \binom{d}{k}$, the sample complexity lower bound in Theorem 3.1.2 essentially matches the guarantee of private multiplicative weights. Using the relationship between private query release and private distribution learning described in Section 2.4, our lower bound also applies to the problem of privately learning a distribution with respect to $k$-way marginals.
3.1.1 Our Techniques

We now describe the main technical ingredients used to prove these results.

**Fingerprinting Codes.** We show that the existence of short fingerprinting codes implies sample complexity lower bounds for 1-way marginal queries. Recall that a 1-way marginal query \( q_j \) is specified by an integer \( j \in [d] \) and asks simply “What fraction of records in \( D \) have a 1 in the \( j \)-th bit?” Suppose a coalition of users takes their codewords and builds a dataset \( D \in (\{0, 1\}^d)^n \) where each record contains one of their codewords, and \( d \) is the length of the codewords. Consider the 1-way marginal query \( q_j(D) \). If every user in \( S \) has a bit \( b \) in the \( j \)-th bit of their codeword, then \( q_j(D) = b \). Thus, if an algorithm answers 1-way marginal queries on \( D \) with non-trivial accuracy, its output can be used to obtain a combined codeword that satisfies the marking assumption. By the tracing property of fingerprinting codes, we can use the combined codeword to identify one of the users in the dataset. However, if we can identify one of the users from the answers, then the algorithm cannot be differentially private.

This argument can be formalized to show that if there is a fingerprinting code for \( n \) users with codewords of length \( d \), then the sample complexity of answering 1-way marginals must be at least \( n \). The nearly-optimal construction of fingerprinting codes due to Tardos [112], gives fingerprinting codes with codewords of length \( d = \tilde{O}(n^2) \), which implies a lower bound of \( n \geq \tilde{\Omega}(\sqrt{d}) \) on the sample complexity required to answer 1-way marginals queries.

**Composition of Sample Complexity Lower Bounds.** Suppose we want to prove a lower bound of \( \tilde{\Omega}(k\sqrt{d}) \) for answering \( k \)-way marginals up to accuracy \( \pm 0.01 \) (a special case of Theorem 3.1.2). Given our lower bound of \( \tilde{\Omega}(\sqrt{d}) \) for 1-way marginals, and the known lower bound of \( \Omega(k) \) for answering \( k \)-way marginals implicit in [41,104], a natural approach is to somehow compose the two lower bounds to obtain a nearly-optimal lower bound of \( \tilde{\Omega}(k\sqrt{d}) \). Our composition technique uses the idea of the \( \Omega(k) \) lower bound from [41,104] to show that if we can answer \( k \)-way marginal queries on a large dataset \( D \) with \( n \) rows, then we can obtain the answers to the 1-way marginal queries on a “subdataset” of roughly \( n/k \) rows. Our lower bound for 1-way marginals tell us that \( n/k = \tilde{\Omega}(\sqrt{d}) \), so we deduce \( n = \tilde{\Omega}(k\sqrt{d}) \).

Actually, this reduction only gives accurate answers to most of the 1-way marginals on the
subdataset, so we need an extension of our lower bound for 1-way marginals to differentially private algorithms that are allowed to answer a small fraction of the queries with arbitrarily large error. Proving a sample complexity lower bound for this problem requires a “robust” fingerprinting code whose tracing algorithm can trace codewords that have errors introduced into a small fraction of the bits. The robust fingerprinting codes of length \( d = \tilde{O}(n^2) \) presented in Section 2.6.1 yield the desired lower bound.

Theorem 3.1.2 is proved by using this composition technique repeatedly to combine our lower bound for 1-way marginals with (variants of) several known lower bounds that capture the optimal dependence on \( \log |Q| \) and \( 1/\alpha^2 \).

**Are Fingerprinting Codes Necessary to Prove Differential Privacy Lower Bounds?** The connection between fingerprinting codes and differential privacy lower bounds extends to arbitrary families \( Q \) of counting queries. We introduce the notion of a generalized fingerprinting code with respect to \( Q \), where each codeword corresponds to a data universe element \( x \in \mathcal{X} \) and the bits of the codeword are given by \( q(x) \) for each \( q \in Q \), but is the same as an ordinary fingerprinting code otherwise. The existence of a generalized fingerprinting code with respect to \( Q \), for \( n \) users, implies a sample complexity lower bound of \( n \) for privately releasing answers to \( Q \). We also show a partial converse to the above result, which states that some sort of “fingerprinting-code-like object” is necessary to prove sample complexity lower bounds for answering counting queries under differential privacy. This object has similar semantics to a generalized fingerprinting code, however the marking assumption required for tracing is slightly stronger and the probability that tracing succeeds can be significantly smaller than what is required by the standard definition of fingerprinting codes. Our partial converse parallels the result of Dwork et al. [50] that shows computational hardness results for differential privacy imply a “traitor-tracing-like object.” We leave it as an open question (Open Problem 3.2.14) to pin down precisely the relationship between fingerprinting codes and information-theoretic lower bounds in differential privacy (and also between traitor-tracing schemes and computational hardness results for differential privacy).
3.1.2 Other Related Work

Previous Work

We have mostly focused on the sample complexity as a function of the number of queries, the number of attributes $d$, and the accuracy parameter $\alpha$. There have been several works focused on the sample complexity as a function of the specific family $\mathcal{Q}$ of queries. For $(\varepsilon, 0)$-differential privacy, Hardt and Talwar [78] showed how to approximately characterize the sample complexity of a family $\mathcal{Q}$ when the accuracy parameter $\alpha$ is sufficiently small. Nikolov, Talwar, and Zhang [95] extended their results to give an approximate characterization for $(\varepsilon, \delta)$-differential privacy and for the full range of accuracy parameters. Specifically, [95] give an $(\varepsilon, \delta)$-differentially private algorithm that answers any family of queries $\mathcal{Q}$ on $\{0, 1\}^d$ with error $\alpha$ using a number of samples that is optimal up to a factor of $\text{poly}(d, \log |\mathcal{Q}|)$ that is independent of $\alpha$. Thus, their algorithm has sample complexity that depends optimally on $\alpha$. However, their characterization may be loose by a factor of $\text{poly}(d, \log |\mathcal{Q}|)$. In fact, when $\alpha$ is a constant, the lower bound on the sample complexity given by their characterization is always $\mathcal{O}(1)$, whereas their algorithm requires $\text{poly}(d, \log |\mathcal{Q}|)$ samples to give non-trivially accurate answers. In contrast, our lower bounds are tight to within $\text{poly}(\log d, \log \log |\mathcal{Q}|, \log(1/\alpha))$ factors, and thus give meaningful lower bounds even when $\alpha$ is constant, but apply only to certain families of queries.

There have been attempts to prove optimal sample complexity lower bounds for $k$-way marginals. In particular, when $k$ is a constant, Kasiviswanathan et al. [82] and De [40] prove a lower bound of $\min\{|\mathcal{Q}|^{1/2}/\alpha, 1/\alpha^2\}$ on the sample complexity. Note that when $\alpha$ is a constant, these lower bounds are $\mathcal{O}(1)$. There has also been a line of work [32, 52, 114] aimed at designing computationally efficient algorithms for $k$-way marginals. While our results show that the private multiplicative weights algorithm is essentially optimal for $k$-way marginals in terms of sample complexity, it runs in time $\text{poly}(2^d, n, |\mathcal{Q}|)$ (whereas the natural benchmark for efficient query release is running time $\text{poly}(d, n, \log |\mathcal{Q}|)$). It remains an important open question to characterize the sample complexity of efficient query release for $k$-way marginals.

Open Problem 3.1.3 (Efficiently releasing marginals). What is the sample complexity of efficiently (i.e. in time $\text{poly}(d, n, k)$) answering $k$-way marginals with approximate differential privacy?

There have also been attempts to explicitly and precisely determine the sample complexity
of even simpler query families than $k$-way conjunctions, such as point functions and threshold functions [5, 7, 8] (see also Chapter 4). These works show that these families can have sample complexity much lower than $\tilde{O}(\sqrt{d \log |\Omega|/\alpha^2})$.

In addition to the general computational hardness results referenced above, there are several results that show stronger hardness results for restricted types of efficient algorithms [51, 72, 116].

Subsequent Work

Subsequent to our work, Steinke and Ullman [110] refined our use of fingerprinting codes to prove a lower bound of $\Omega(\sqrt{d \log(1/\delta)/\varepsilon})$ on the number of samples required to release the mean of each of the $d$ attributes under $(\varepsilon, \delta)$-differential privacy when $\delta \ll 1/n$. This lower bound is optimal up to constant factors, and improves on Theorem 3.1.1 by a factor of roughly $\sqrt{\log(1/\delta) \cdot \log d}$.

Our fingerprinting code technique has also been used to prove lower bounds for other types of differentially private data analyses. Namely, Dwork et al. [57] prove lower bounds for differentially private principal component analysis and Bassily, Smith, and Thakurta [4] prove lower bounds for differentially private empirical risk minimization. In order to establish lower bounds for privately releasing threshold functions, we construct in Chapter 4 a fingerprinting-code-like object that yields a lower bound for the problem of releasing a value between the minimum and maximum of a dataset. In Chapter 6, we show how fingerprinting codes can be used to show lower bounds for privately PAC learning multiple concepts.

Dwork et al. [56] observe that the privacy attack implicit in our negative results is closely related to the influential attacks that were employed by Homer et al. [80] (and further studied in [107]) to violate privacy of public genetic datasets. Using this connection, they show how to make Homer et al.’s attack robust to very general models of noise and how to make the attack work without detailed knowledge of the population the dataset represents.

A pair of works [79, 111] show that fingerprinting codes and the related traitor-tracing schemes imply both information-theoretic lower bounds and computational hardness results for the “false discovery” problem in adaptive data analysis. Specifically, they show lower bounds for answering an online sequence of adaptively chosen counting queries where the dataset is a sample from some unknown distribution and the answers must be accurate with respect to that distribution. These works [79, 111] effectively reverse a connection established in [44], which used differentially private
algorithms to obtain positive results for this problem.

Our technique for composing lower bounds in differential privacy has also found applications outside of privacy. Specifically, Liberty et al. [87] used this technique to prove nearly optimal lower bounds on the space required to “sketch” a dataset while approximately preserving answers to $k$-way marginal queries (called “frequent itemset queries” in their work).

### 3.2 Fingerprinting Codes vs. Differential Privacy

In this section we prove that there exists a simple family of $d$ queries that requires $n \geq \tilde{\Omega}(\sqrt{d})$ samples for both accuracy and privacy. Specifically, we prove that for the family of 1-way marginals on $d$ bits, sample complexity $\tilde{\Omega}(\sqrt{d})$ is required to produce differentially private answers that are accurate even just to within $\pm 1/3$. In contrast, without a privacy guarantee, $\Theta(\log d)$ samples from the population are necessary and sufficient to ensure that the answers to these queries on the dataset and the population are approximately the same. The best previous lower bound for $(\varepsilon, \delta)$-differential privacy is also $O(\log d)$, which follows from the techniques of [41, 104].

#### 3.2.1 Re-identifiable Distributions

All of our eventual lower bounds will take the form of a “re-identification” attack, in which we possess data from a large number of individuals, and identify one such individual who was included in the dataset. In this attack, we choose a distribution on datasets and give an adversary 1) a dataset $D$ drawn from that distribution and 2) either $A(D)$ or $A(D_{-i})$ for some row $i$, where $A$ is an alleged sanitizer. Here, the notation $D_{-i}$ denotes the dataset $D$ where row $i$ is replaced by a fixed but arbitrary “junk” row $\perp$. The adversary’s goal is to identify a row of $D$ that was given to the sanitizer. We say that the distribution is re-identifiable if there is an adversary who can identify such a row with sufficiently high confidence whenever $A$ outputs accurate answers. If the adversary can do so, it means that there must be a pair of adjacent datasets $D \sim D_{-i}$ such that the adversary can distinguish $A(D)$ from $A(D_{-i})$, which means $A$ cannot be differentially private.

**Definition 3.2.1 (Re-identifiable Distribution).** For a data universe $\mathcal{X}$ and $n \in \mathbb{N}$, let $\mathcal{D}$ be a distribution on $n$-row datasets $D \in \mathcal{X}^n$. Let $\mathcal{Q}$ be a family of counting queries on $\mathcal{X}$ and let $\eta, \xi, \alpha, \gamma \in [0, 1]$ be parameters. The distribution $\mathcal{D}$ is $(\eta, \xi)$-re-identifiable from $(\alpha, \gamma)$-accurate
answers to $Q$ if there exists a (possibly randomized) adversary $B: \mathcal{X}^n \times \mathbb{R}^{|Q|} \rightarrow [n] \cup \{\perp\}$ such that for every randomized algorithm $A: \mathcal{X}^n \rightarrow \mathbb{R}^{|Q|}$, the following both hold:

1. $\Pr_{D \sim \mathcal{D}} [(B(D, A(D)) = \perp) \land (A(D) \text{ is } (\alpha, \gamma)\text{-accurate for } Q)] \leq \eta$.

2. For every $i \in [n]$, $\Pr_{D \sim \mathcal{D}} [B(D, A(D_{-i})) = i] \leq \xi$.

Here the probability is taken over the choice of $D$ and $i$ as well as the coins of $A$ and $B$. We allow $\mathcal{D}$ and $B$ to share a common state.\footnote{As in the definition of fingerprinting codes, we could model this shared state by having $\mathcal{D}$ output an additional string $aux$ that is given to $B$ but not to $A$. However, we make the shared state implicit to reduce notational clutter.}

If $A$ is an $(\alpha, \gamma)$-accurate algorithm, then its output $A(D)$ will be $(\alpha, \gamma)$-accurate with probability at least $2/3$. Therefore, if $\eta < 2/3$, we can conclude that $\Pr [B(D, A(D)) \in [n]] \geq 1 - \eta - 1/3 = \Omega(1)$. In particular, there exists some $i^* \in [n]$ for which $\Pr [B(D, A(D)) = i^*] \geq \Omega(1/n)$. However, if $\xi = o(1/n)$, then $\Pr [B(D, A(D_{-i^*})) = i^*] \leq \xi = o(1/n)$. Thus, for this choice of $\eta$ and $\xi$ we will obtain a contradiction to $(\varepsilon, \delta)$-differential privacy for any $\varepsilon = O(1)$ and $\delta = o(1/n)$. We remark that this conclusion holds even if $\mathcal{D}$ and $B$ share a common state.

We summarize this argument with the following lemma.

**Lemma 3.2.2.** Let $Q$ be a family of counting queries on $\mathcal{X}$, $n \in \mathbb{N}$ and $\xi \in [0, 1]$. Suppose there exists a distribution on $n$-row datasets $D \in \mathcal{X}^n$ that is $(\eta, \xi)$-re-identifiable from $(\alpha, \gamma)$-accurate answers to $Q$. Then there is no $(\varepsilon, \delta)$-differentially private algorithm $A: \mathcal{X}^n \rightarrow \mathbb{R}^{|Q|}$ that is $(\alpha, \gamma)$-accurate for $Q$ for any $\varepsilon, \delta$ such that $e^{-\varepsilon}(1 - \eta - 1/3)/n - \delta \geq \xi$.

In particular, if there exists a distribution that is $(\eta, o(1/n))$-re-identifiable from $(\alpha, \gamma)$-accurate answers to $Q$ for $\eta = 1/3$, then no algorithm $A: \mathcal{X}^n \rightarrow \mathbb{R}^{|Q|}$ that is $(\alpha, \gamma)$-accurate for $Q$ can satisfy $(O(1), o(1/n))$-differential privacy.

### 3.2.2 Lower Bounds for 1-Way Marginals

We are now ready to state and prove the main result of this section, namely that there is a distribution on datasets $D \in \{0, 1\}^d$, for $n = \tilde{\Omega}(\sqrt{d})$, that is re-identifiable from accurate answers to 1-way marginals.
**Theorem 3.2.3.** For every $n, d \in \mathbb{N}$, and $\xi \in [0, 1]$ if there exists an $(n, d)$-fingerprinting code with security $\xi$, robust to a $\gamma$ fraction of errors, then there exists a distribution on $n$-row datasets $D \in \{0,1\}^d$ that is $(\xi, \xi)$-re-identifiable from $(1/3, \gamma)$-accurate answers to $M_{1,d}$.

In particular, if $\xi = o(1/n)$, then by Lemma 3.2.2 there is no algorithm $A : \{0,1\}^n \to \mathbb{R}^{M_{1,d}}$ that is $(O(1), o(1/n))$-differentially private and $(1/3, \gamma)$-accurate for $M_{1,d}$.

By combining Theorem 3.2.3 with Theorem 2.6.4 we obtain a sample complexity lower bound for 1-way marginals, and thereby establish Theorem 3.1.1 in the introduction.

**Corollary 3.2.4.** For every $d \in \mathbb{N}$, the family of 1-way marginals on $\{0,1\}^d$ has sample complexity at least $\tilde{\Omega}(\sqrt{d})$ for $(1/3, 1/75)$-accuracy and $(O(1), o(1/n))$-differential privacy.

**Proof of Theorem 3.2.3.** Let $(Gen, Trace)$ be the promised fingerprinting code. We define the re-identifiable distribution $D$ to simply be the output distribution of the code generator, $Gen$. And we define the privacy adversary $B$ to take the answers $a = A(D) \in [0,1]^{M_{1,d}}$, obtain $\bar{a} \in \{0,1\}^{M_{1,d}}$ by rounding each entry of $a$ to $\{0,1\}$, run the tracing algorithm $Trace$ on the rounded answers $\bar{a}$, and return its output. The shared state of $D$ and $B$ will be the shared state of $Gen$ and $Trace$.

Now we will verify that $D$ is $(\xi, \xi)$-re-identifiable. First, suppose that $A(D)$ outputs answers $a = (a_{qj})_{j \in [d]}$ that are $(1/3, \gamma)$-accurate for 1-way marginals. That is, there is a set $G \subseteq [d]$ such that $|G| \geq (1-\gamma)d$ and for every $j \in G$, the answer $a_{qj}$ estimates the fraction of rows having a 1 in column $j$ to within $1/3$. Let $\bar{a}_{qj}$ be $a_{qj}$ rounded to the nearest value in $\{0,1\}$. Let $j$ be a column in $G$. If column $j$ has all 1’s, then $a_{qj} \geq 2/3$, and $\bar{a}_{qj} = 1$. Similarly, if column $j$ has all 0’s, then $a_{qj} \leq 1/3$, and $\bar{a}_{qj} = 0$. Therefore, we have

$$a \text{ is } (1/3, \gamma)-\text{accurate} \implies \bar{a} \in F_\gamma(D).$$

(3.1)

By security of the fingerprinting code (Definition 4.2.10), we have

$$\Pr[\bar{a} \in F_\gamma(D) \land Trace(D, \bar{a}) = \bot] \leq \xi.$$

(3.2)

Combining (3.1) and (3.2) implies that

$$\Pr[A(D) \text{ is } (1/3, \gamma)-\text{accurate} \land Trace(D, \bar{a}) = \bot] \leq \xi.$$

But the event $Trace(D, \bar{a}) = \bot$ is exactly the same as $B(D, A(D)) = \bot$, and thus we have established
the first condition necessary for $D$ to be $(\xi, \xi)$-re-identifiable.

The second condition for re-identifiability follows directly from the soundness of the fingerprinting code, which asserts that for every adversary $A_{FP}$, in particular for $A$, it holds that

$$\Pr[\text{Trace}(D, A_{FP}(D-i)) = i] \leq \xi.$$ 

This completes the proof. \qed

**Remark 3.2.5.** Corollary 3.2.4 implies a lower bound of $\tilde{\Omega}(\sqrt{d})$ for any family $Q$ on a data universe $X$ in which we can “embed” the 1-way marginals on $\{0,1\}^d$ in the sense that there exists $q_1, \ldots, q_d \in Q$ such that for every string $x \in \{0,1\}^d$ there is an $x' \in \{0,1\}^d$ such that $(q_1(x'), \ldots, q_d(x')) = x$. (The maximum such $d$ is actually the VC dimension of $X$ when we view each element $x \in X$ as defining a mapping $q \mapsto q(x)$. See Definition 2.2.1.)

Our proof technique does not directly yield a lower bound with any meaningful dependence on the accuracy $\alpha$. Since the privacy adversary $B$ simply runs the tracing algorithm on the rounded answers it is given, it is not able to leverage subconstant accuracy to gain an advantage in re-identification. However, Lemma 2.3.8 lets us generically translate our lower bound for constant accuracy into a lower bound depending linearly on $1/\alpha$. For 1-way marginals, we get an essentially tight sample complexity lower bound of $\tilde{\Omega}(\sqrt{d}/\alpha)$ for $(\alpha, \gamma)$-accuracy.

**Corollary 3.2.6.** For every $d \in \mathbb{N}$, the family of 1-way marginals on $\{0,1\}^d$ has sample complexity at least $\tilde{\Omega}(\sqrt{d}/\alpha)$ for $(\alpha, 1/75)$-accuracy and $(O(1), o(1/n))$-differential privacy.

**Minimax Lower Bounds for Statistical Inference**

Using the additional structure of Tardos’ fingerprinting code, and the robust fingerprinting codes of Section 2.6.1, we can prove minimax lower bounds for an “inference version” of the problem computing the 1-way marginals of a product distribution.

For any $d \in \mathbb{N}$, and any marginals $p = (p_1, \ldots, p_d) \in [0,1]^d$, let $D_p$ denote the product distribution over strings $x \in \{0,1\}^d$ where each coordinate $x_i$ is an independent draw from a Bernoulli random variable with mean $p_i$ (i.e. $x_i$ is set to 1 with probability $p_i$ and set to 0 otherwise). We use $D_p^{\otimes n}$ to
denote \( n \) independent draws from \( D_p \). We say that a vector \( q \in [0,1]^d \) is \((\alpha, \gamma)\)-accurate for \( p \) if

\[
\Pr_{i \leftarrow \mathbb{R}[d]} [\|q_i - p_i\| \leq \alpha] \geq 1 - \gamma.
\]

We can now formally define the problem of inferring the marginals \( p \) as follows.

**Definition 3.2.7.** Let \( \alpha, \gamma \in [0,1] \) be parameters. An algorithm \( A : ([0,1]^d)^n \rightarrow \mathbb{R}^d \) \((\alpha, \gamma)\)-accurately infers the marginals of a product distribution if for every vector of marginals \( p \in [0,1]^d \),

\[
\Pr_{D \leftarrow \mathbb{R}[D_p^{\otimes n}], A\text{'s coins}} [A(D) is (\alpha, \gamma)\text{-accurate for } p] \geq 2/3.
\]

Our lower bound can thus be stated as follows.

**Theorem 3.2.8.** Suppose there is a function \( n = n(d) \) such that for every \( d \in \mathbb{N} \), there exists an algorithm \( A : ([0,1]^d)^n \rightarrow \mathbb{R}^d \) that satisfies \((O(1), o(1/n))\)-differential privacy and \((1/3, 1/75)\)-accurately infers the marginals of a product distribution. Then \( n = \Omega(\sqrt{d}) \).

**Proof Sketch.** The proof has the same general structure that we used to prove Theorem 3.2.3, combined with observations about the structure of the fingerprinting codes used in that proof. First, in Tardos’ (non-robust) fingerprinting code, the codebook \( D \) is chosen by first sampling marginals \( p \in [0,1]^d \) from an appropriate distribution and then sampling \( D \) from \( D_p^{\otimes n} \). The robust fingerprinting codes in Section 2.6.1 also have this property. Thus the instances used to prove Theorem 3.2.3 indeed consist of independent samples from a product distribution, which is what the inference problem assumes.

Next, recall that the proof of Theorem 3.2.3 shows that any string that is \((\alpha, \gamma)\)-accurate for the 1-way marginals of \( D \) can be traced successfully. It turns out that any string that is \((\alpha, \gamma)\)-accurate for the marginals \( p \) can also be traced successfully. Intuitively, this is because the rows of \( D \) are sampled independently from \( D_p \), so accuracy for the 1-way marginals of \( D \) and accuracy for \( p \) coincide with high probability, at least when \( n = \omega(\log d) \). Steinke and Ullman [110] explicitly show that this definition of accuracy suffices to trace regardless of the value of \( n \).

These two observations suffice to show that, when \( n \) is too small, a differentially private algorithm cannot be accurate for \( p \) with high probability over the choices of both \( p \) and \( D \). Thus, for every differentially private algorithm, there exists some \( p \) such that the algorithm is not accurate with high probability over the choice of \( D \), which means that the algorithm does not accurately infer the
marginals of an arbitrary product distribution.

### 3.2.3 Lower Bounds for Fingerprinting Code Length via Differential Privacy

By the contrapositive of Theorem 3.2.3, upper bounds on the sample complexity of answering 1-way marginals with differential privacy imply a lower bound on the length $d$ of any fingerprinting code with a given number of users $n$. As pointed out to us by Adam Smith, this yields a particularly simple, self-contained proof of Tardos’ [112] optimal lower bound on the length of fingerprinting codes. Specifically, using the well known Gaussian mechanism for achieving differential privacy, we can design a simple adversary $A_{FP}$ that violates the security of any traitor tracing scheme with length $d = o(n^2)$.

**Theorem 3.2.9.** There is a function $n = n(d) = \tilde{O}(\sqrt{d})$ such that for every $d$, there is no $(n, d)$-fingerprinting code with security $\xi < 1/6en$.

**Proof.** Before diving into the proof, we will state the following elementary fact about Gaussian random variables. The fact simply says that a Gaussian random variable with suitable variance is “close” to a shifted version of itself in a particular sense. This same fact is used to show that adding Gaussian noise of suitable variance provides differential privacy.

**Fact 3.2.10.** Let $c, c' \in \mathbb{R}^d$ satisfy $\|c - c'\|_2 \leq \sqrt{d}/n$, $\delta > 0$ be a parameter, and let $\sigma^2 = 2d\ln(1/\delta)/n^2$. Let $z \in \mathbb{R}^d$ be a random vector where each coordinate is an independent draw from a Gaussian distribution with mean 0 and variance $\sigma^2$. Then for any (measurable) set $T \subseteq \mathbb{R}^d$,

$$\Pr \left[ c + z \in T \right] \geq \left( 1/e \right) \Pr \left[ c' + z \in T \right] - \delta.$$

Now we proceed with the proof. Fix any choice of $d$. Assume towards a contradiction that there is an $(n, d)$-fingerprinting code $(Gen, Trace)$ with security $\xi < 1/6en$ for $n = \left\lceil \sqrt{18d\ln(6en)\ln(3d/2)} \right\rceil$. Observe that $n = n(d) = \tilde{O}(\sqrt{d})$ as promised in the theorem.

Let $A_{FP}(C_S)$ be the following adversary. Define the vector $\bar{c} \in [0, 1]^d$ as

$$\bar{c} = \frac{1}{n} \sum_{i \in S} c_i.$$

Now, let $z \in \mathbb{R}^d$ be a $d$-dimensional Gaussian where every coordinate is independent with mean 0 and variance $\sigma^2 = 2d\ln(1/\delta)/n^2$, for $\delta = 1/6en$. Finally, let $c'$ be $\bar{c}$ with each coordinate rounded to
{0,1}, and output the pirated codeword $c'$.

First we claim that $\mathcal{A}_{FP}$ outputs feasible codewords with at least constant probability.

**Claim 3.2.11.** For every $S$ such that $|S| \geq n - 1$, and every codebook $C = (c_{ij}) \in \{0,1\}^{n \times d}$,

$$\Pr_{c' \leftarrow R_{\mathcal{A}_{FP}(C_S)}}[c' \in F(C_S)] \geq 2/3.$$  

**Proof of Claim 3.2.11.** By a standard tail bound for the Gaussian, we have

$$\Pr[\forall j, |z_j| < \sigma \sqrt{\ln(3d/2)}] \geq 2/3.$$  

Thus, by our choice of $\sigma$ and $n \geq \sqrt{18d \ln(1/\delta) \ln(3d/2)}$ we have $\Pr[\forall j, |z_j| < 1/3] \geq 2/3$. Now the claim follows easily. Specifically, if $c_{ij} = 1$ for every $i \in S$, then $(1/n) \sum_{i \in S} c_{ij} \geq 1 - 1/n$, so $\hat{c}_j > 2/3 - 1/n$ and $c'_j = 1$. A similar argument applies if $c_{ij} = 0$ for every $i \in S$. \qed

Now it remains to show that $\mathcal{A}_{FP}$ cannot be traced successfully. By assumption $(\text{Gen}, \text{Trace})$ has security $\xi < 1/6en < 1/3$. Then we have in particular

$$\Pr_{c \leftarrow R_{\text{Gen}}, c' \leftarrow R_{\mathcal{A}_{FP}(C)}}[c' \in F(C) \land \text{Trace}(C, c') = \bot] < \xi.$$  

Combining with Claim 3.2.11 we have

$$\Pr_{c \leftarrow R_{\text{Gen}}, c' \leftarrow R_{\mathcal{A}_{FP}(C)}}\left[\text{Trace}(C, c') \in [n]\right] > 1 - 1/3 - \xi > 1/3.$$  

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

$$\Pr_{c \leftarrow R_{\text{Gen}}, c' \leftarrow R_{\mathcal{A}_{FP}(C)}}[\text{Trace}(C, c') = i^*] > 1/3n.$$  

To complete the proof, it now suffices to show that if $S = [n] \setminus \{i^*\}$, then

$$\Pr_{c \leftarrow R_{\text{Gen}}, c' \leftarrow R_{\mathcal{A}_{FP}(C_S)}}[\text{Trace}(C, c') = i^*] \geq 1/6en > \xi,$$  

which will contradict the security of the fingerprinting code.

To do so, first observe that if

$$\overline{c} = \frac{1}{n} \sum_{i \in [n]} c_i,$$  

and

$$\overline{c}^S = \frac{1}{n} \sum_{i \in S} c_i,$$
then \( \| \vec{c}_j - \vec{c}_j^2 \|_2 \leq \sqrt{d}/n \). Now, in case the tracing algorithm is randomized, let \( \text{Trace}_r \) denote the tracing algorithm when run with its random coins fixed to \( r \). For any string of random coins \( r \), define the set \( T_r = \{ t \in \mathbb{R}^d \mid \text{Trace}_r(C, \round(t)) = i^* \} \). Here, \( \round(\cdot) \) is the function that rounds each entry of its input to \( \{0, 1\} \).

By Fact 3.2.10 (with \( \delta = 1/6en > \xi \)), for every \( r \),

\[
\Pr_{\vec{c}} \left[ \vec{c}^2 + z \in T_r \right] \geq (1/e) \Pr_{\vec{c}} \left[ \vec{c} + z \in T_r \right] - \xi.
\]

Applying (3.3), and averaging over \( C \leftarrow R \text{ Gen} \) and \( r \), we have

\[
\Pr_{C \leftarrow R \text{ Gen}} \left[ \text{Trace}(C, c') = i^* \right] \geq (1/e)(1/3n) - 1/6en = 1/6en > \xi,
\]

which is the desired contradiction. This completes the proof.

\( \square \)

### 3.2.4 Fingerprinting Codes for General Query Families

In this section, we generalize the connection between fingerprinting codes and sample complexity lower bounds for arbitrary sets of queries. We show that a generalized fingerprinting code with respect to any family of counting queries \( \mathcal{Q} \) yields a sample complexity lower bound for \( \mathcal{Q} \), which is analogous to our lower bound for \( 1 \)-way marginals (Theorem 3.2.3). We then argue that some type of fingerprinting code is necessary to prove any sample complexity lower bound by exhibiting a tight connection between such lower bounds and a weak variant of our generalized fingerprinting codes.

We begin by defining our generalization of fingerprinting codes. Fix a finite data universe \( \mathcal{X} \) and a set of counting queries \( \mathcal{Q} \) over \( \mathcal{X} \). A generalized fingerprinting code with respect to the family \( \mathcal{Q} \) consists of a pair of randomized algorithms \( (\text{Gen}, \text{Trace}) \). The code generation algorithm \( \text{Gen} \) produces a codebook \( C \in \mathcal{X}^n \). Each row \( c_i \) of \( C \) is the codeword corresponding to user \( i \). A coalition \( S \subseteq [n] \) of pirates receives the subset \( C_S = \{ c_i : i \in S \} \) of codewords, and produces an answer vector \( a \in [0, 1]^{|\mathcal{Q}|} \). We replace the traditional marking condition on the pirates with the generalized constraint that they output a feasible answer vector. A natural way to define feasibility for answer vectors is to require a condition similar to \( (\alpha, \gamma) \)-accuracy, i.e. an answer vector \( a \) is feasible if \( |a_q - q(C_S)| \leq \alpha \) for all but a \( \gamma \) fraction of queries \( q \in \mathcal{Q} \). We then define a generalized set

\[\text{Note, for completeness, that } T_r \text{ is measurable, since the set of } c' \in \{0, 1\}^d \text{ such that } \text{Trace}_r(C, c') = i^* \text{ is finite (for every fixed } n, d \text{) and for every } c', \{ t \mid \round(t) = c' \} \text{ is a hypercube, so } T_r \text{ is a union of finitely many hypercubes.}\]
of feasible answer vectors by
\[ F_{\alpha, \gamma}(C_S) = \left\{ a \in [0,1]^{\lvert Q \rvert} \mid \Pr_{q \sim Q} [|a_q - q(C_S)| \leq \alpha] \geq 1 - \gamma \right\}. \]

When \( \alpha = 1 - 1/n \), the generalized set of feasible answer vectors captures the traditional marking assumption by rounding each entry of a feasible answer vector to 0 or 1.

**Definition 3.2.12.** A pair of algorithms \((Gen, Trace)\) is an \((n, Q)\)-fingerprinting code for \((\alpha, \gamma)\)-accuracy with security \((\eta, \xi)\) if \(Gen\) outputs a codebook \(C \in \mathcal{X}^n\) and for every (possibly randomized) adversary \(A_{FP}\), and every coalition \(S \subseteq [n]\) with \(\lvert S \rvert \geq n - 1\), if we set \(a \leftarrow R A_{FP}(C_S)\), then

1. \(\Pr[a \in F_{\alpha, \gamma}(C_S) \land Trace(C,a) = \bot] \leq \eta,\)

2. \(\Pr[Trace(C,a) \in [n] \setminus S] \leq \xi,\)

where the probability is taken over the coins of \(Gen, Trace\), and \(A_{FP}\). The algorithms \(Gen\) and \(Trace\) may share a common state.

The security properties of Definition 3.2.12 differ from those of an ordinary fingerprinting code in two ways so as to enable a clean statement of a composition theorem for generalized fingerprinting codes (Theorem 3.3.6). First, we use two separate security parameters \(\eta, \xi\) for the different types of tracing errors, as in the definition of re-identifiable distributions. Second, security only needs to hold for coalitions of size \(n - 1\) or \(n\). However, this condition implies security for coalitions of arbitrary size with an increased false accusation probability of \(n\xi\).

As in Theorem 3.2.3, the existence of a generalized \((n, Q)\)-fingerprinting code implies a sample complexity lower bound of \(n\) for privately releasing answers to \(Q\), with essentially the same proof.

**Theorem 3.2.13.** For every \(n \in \mathbb{N}\) and \(\eta, \xi \in [0,1]\), if there exists an \((n, Q)\)-fingerprinting code for \((\alpha, \gamma)\)-accuracy with security \((\eta, \xi)\), then there exists a distribution on \(n\)-row datasets \(D \in \mathcal{X}^n\) that is \((\eta, \xi)\)-re-identifiable from \((\alpha, \gamma)\)-accurate answers to \(Q\).

In particular, if \(\eta \leq 1/3\) and \(\xi = o(1/n)\), then there is no algorithm \(A : \mathcal{X}^n \rightarrow [0,1]^{\lvert Q \rvert}\) that is \((O(1), o(1/n))\)-differentially private and \((\alpha, \gamma)\)-accurate for \(Q\).

---

4 An equivalent way to view a codebook is as a set of \(n\) codewords \(C \subseteq \{(0,1)^{\lvert Q \rvert}\}^n\), where each user’s codeword is \(c_i = (q(x))_{q \in Q}\) for some \(x \in \mathcal{X}\). Notice that the case where \(Q\) is the class of 1-way marginals places no constraints on the structure of a codeword, i.e. a codeword can be any binary string. With this viewpoint, the goal of the pirates is to output an answer vector \(a \in [0,1]^{\lvert Q \rvert}\) with \(|a_q - \frac{1}{|S|} \sum_{i \in S} (c_i)_q| \leq \alpha\) for all but a \(\gamma\) fraction of the queries \(q \in Q\).
We now turn to investigate whether a converse to Theorem 3.2.13 holds. We show that a sample complexity lower bound for a family of queries \( Q \) is essentially equivalent to the existence of a weak type of fingerprinting code, where the tracing procedure depends on the family \( Q \) and the tracing error probabilities satisfy a certain affine constraint. It remains an interesting open question to determine the precise relationship between privacy lower bounds and our notion of generalized fingerprinting codes.

**Open Problem 3.2.14** (Generalized fingerprinting codes vs. sample complexity lower bounds). Is the existence of an \((n, Q)\)-generalized fingerprinting code equivalent to a sample complexity lower bound of \( n \) for releasing accurate answers to \( Q \)? An equivalence between generalized fingerprinting codes and weak fingerprinting codes (defined below) would give a positive answer to this question.

**Definition 3.2.15.** A pair of algorithms \((\text{Gen}, \text{Trace})\) is an \((n, Q)\)-weak fingerprinting code for \((\alpha, \gamma)\)-accuracy with security \((\varepsilon, \delta)\) if \( \text{Gen} \) outputs a codebook \( C \in \mathcal{X}^n \) and for every (possibly randomized) adversary \( A_{FP} \) that outputs a feasible answer vector with probability \( 2/3 \), and every coalition \( S \subseteq [n] \) with \( |S| \geq n - 1 \), if we set \( a \overset{\text{r}}{\leftarrow} A_{FP}(C_S) \), then

\[
\Pr[\text{Trace}(C, a) \neq \bot] > e^\varepsilon n \cdot \Pr[\text{Trace}(C, a) \in [n] \setminus S] + \delta,
\]

where the probabilities are taken over the coins of \( \text{Gen}, \text{Trace}, \) and \( A_{FP} \). The algorithms \( \text{Gen} \) and \( \text{Trace} \) may share a common state.

That is, we require the false accusation probability \( \Pr[\text{Trace}(C, a) \in [n] \setminus S] \) to be much smaller than the total probability of accusing any user. Note that a tracing algorithm that accuses a random user with probability \( p \) will falsely accuse a user with probability \( p/n \) when \( |S| = n - 1 \); however, this does not satisfy Definition 3.2.15 because we require the gap between the two probabilities to be at least a factor of \( e^\varepsilon n \).

Observe that taking \( \xi < (1 - \delta)/2e^\varepsilon n \) in Definition 3.2.12 yields an \((n, Q)\)-weak fingerprinting code with security \((\varepsilon, \delta)\). However, Definition 3.2.15 is weaker than Definition 3.2.12 in a few important ways. First, security only holds against pirates with a failure probability of at most \( 1/3 \). Second, while Definition 3.2.12 requires completeness error \( \Pr[\text{Trace}(C, a) = \bot] < \xi \), a weak fingerprinting code allows \( \Pr[\text{Trace}(C, a) = \bot] = 1 - o(1) \) as long as \( \Pr[\text{Trace}(C, a) \in [n] \setminus S] \) is sufficiently small.
The following theorem shows that the existence of an \((n, \mathcal{Q})\)-weak fingerprinting code is essentially equivalent to a sample complexity lower bound of \(n\) against \(\mathcal{Q}\).

**Theorem 3.2.16.** For every \(n \in \mathbb{N}\), if there exists an \((n, \mathcal{Q})\)-weak fingerprinting code for \((\alpha, \gamma)\)-accuracy with security \((\varepsilon, \delta)\), then there exists a distribution on \(n\)-row datasets \(D \in \mathcal{X}^n\) such that no \((\varepsilon/2, \delta/(2e^{\varepsilon/2}n))\)-differentially private algorithm \(A : \mathcal{X}^n \rightarrow \mathbb{R}^{||\mathcal{Q}||}\) outputs \((\alpha, \gamma)\)-accurate answers to \(\mathcal{Q}\).

Conversely, let \(\varepsilon \leq 3\) and suppose there is no \((\varepsilon, \delta)\)-differentially private \(A : \mathcal{X}^n \rightarrow \mathbb{R}^{||\mathcal{Q}||}\) that gives \((\alpha, \gamma)\)-accurate answers to \(\mathcal{Q}\) with probability at least \(1/2\). Then there exists an \((m = [n/\varepsilon], \mathcal{Q})\)-weak fingerprinting code for \((\alpha - \alpha', \gamma)\)-accuracy with security \((\varepsilon/6, \delta/(e^{\varepsilon/3} + e^{5\varepsilon/6}))\), for \(\alpha' = \tilde{O}(\sqrt{e} \text{VC(\mathcal{Q})}/n)\).

**Proof.** The forward direction follows the ideas of Lemma 3.2.2 and Theorem 3.2.3. Suppose for the sake of contradiction that there exists an \((\varepsilon', \delta')\)-differentially private \(A : \mathcal{X}^n \rightarrow \mathbb{R}^{||\mathcal{Q}||}\) that is \((\alpha, \gamma)\)-accurate for \(\mathcal{Q}\). Define a pirate strategy \(A_{FP}\) for coalitions of size \(|S| \geq n - 1\) by running \(A\) on its input \(C_S\) (possibly padded to size \(n\) by a junk row). Since \(A\) is \((\alpha, \gamma)\)-accurate, with probability at least \(2/3\) it produces an answer vector \(a\) such that \(|a - q(C_S)| \leq \alpha\) for all but a \(\gamma\) fraction of \(q \in \mathcal{Q}\). Hence, \(A_{FP}\) outputs a feasible answer vector with probability \(2/3\). Define

\[
p = \Pr_{C \leftarrow R_{Gen}} \Pr_{\text{coins}(A_{FP}), \text{coins}(\text{Trace})} [\text{Trace}(C, A_{FP}(C)) \neq \perp].
\]

Then there exists an \(i^*\) such that \(\Pr[\text{Trace}(C, A_{FP}(C)) = i^*] \geq p/n\). By differential privacy,

\[
\Pr[\text{Trace}(C, A_{FP}(C_{-i^*})) = i^*] \geq e^{-\varepsilon'} \cdot \left(\frac{p}{n - \delta'}\right).
\]

On the other hand, by the security of the weak fingerprinting code and differential privacy,

\[
e^{\varepsilon} \cdot n \cdot \Pr[\text{Trace}(C, A_{FP}(C_{-i^*}) = i^*] \leq \Pr[\text{Trace}(C, A_{FP}(C_{-i^*}) \neq \perp] - \delta
\leq e^{\varepsilon'} p + \delta' - \delta.
\]

This yields a contradiction whenever \(\varepsilon' \leq \varepsilon/2\) and \(\delta' \leq \delta/(1 + e^{\varepsilon/2}n)\).

We now show the converse direction, i.e. that the high sample complexity of \((\mathcal{Q}, \mathcal{X})\) implies the existence of a weak fingerprinting code. We begin with a technical lemma which shows that the high sample complexity of \(\mathcal{Q}\) also rules out mechanisms that satisfy only a one-sided constraint on the
probability of any event under the replacement of one row:

**Lemma 3.2.17.** Let $\varepsilon \leq 1/2$. Let $A$ be an $(\alpha, \gamma)$-accurate algorithm for $Q$ on datasets $D \in \mathcal{X}^m$. Suppose we have that for all datasets $D \in \mathcal{X}^m$, all $i \in [m]$, and all measurable $T \subseteq \text{Range}(A)$ that

$$\Pr_{j \leftarrow \mathcal{R}[m]} [A(D_{-j}) \in T] \leq e^\varepsilon \Pr_{\text{coins}(A)} [A(D_{-i}) \in T] + \delta.$$

Let $d = \text{VC}(Q)$ be the VC-dimension of $Q$ and let

$$\alpha' = \left( \frac{8}{m} \cdot \left( \ln 24 + d \cdot \ln \left( \frac{2em}{d} \right) \right) \right)^{1/2} + \frac{\varepsilon}{m}.$$

Then there exists a $(6\varepsilon, (e^{2\varepsilon} + e^{5\varepsilon})\delta)$-differentially private algorithm $B$ on datasets of size $n = \lfloor m/\varepsilon \rfloor$ that gives $(\alpha + \alpha', \gamma)$-accurate answers to $Q$ on any dataset $y \in \mathcal{X}^n$ with probability at least $1/2$.

**Proof.** On input a dataset $D \in \mathcal{X}^n$, consider the algorithm $B'$ that samples a random subset $D'$ consisting of $m$ rows from $D$ (without replacement) and returns $A(D')$. Then by our hypothesis on $A$, for every $i \in [n]$ and every measurable $T \subseteq \text{Range}(B) = \text{Range}(A)$ we have

$$\Pr_{j \leftarrow \mathcal{R}[n]} [B'(D_{-j}) \in T] \leq e^\varepsilon \Pr_{\text{coins}(B')} [B'(D_{-i}) \in T] + \delta.$$

On the other hand, a “secrecy-of-the-sample” argument [81] enables us to obtain the reverse inequality. For a row $k \in [n]$, consider the following two experiments:

Experiment 1: Sample a random subset $D'$ of $m$ rows from $D_{-k}$.

Experiment 2: Sample $j \leftarrow [n]$, and then sample a random subset $D'$ of $m$ rows from $D_{-j}$.

Any dataset $D'$ sampleable under Experiment 1 appears with probability $1/\binom{n}{m}$, but appears with probability at least

$$\frac{n-m}{n} \cdot \frac{1}{\binom{n}{m}} \geq (1-\varepsilon) \cdot \frac{1}{\binom{n}{m}}$$

under Experiment 2. Therefore,

$$\Pr_{j \leftarrow \mathcal{R}[n]} [B(D_{-j}) \in T] \geq e^{-2\varepsilon} \Pr_{\text{coins}(B)} [B(D_{-k}) \in T].$$
Combining the two inequalities shows that for every dataset $D \in \mathcal{X}^n$ and every $i, k \in [n],$

$$\Pr_{\text{coins}(\mathcal{B}')} [B'(D_k) \in T] \leq e^{3\varepsilon} \Pr_{\text{coins}(\mathcal{B}')} [B'(D_i) \in T] + e^{2\varepsilon}\delta.$$ 

A calculation shows that the algorithm $B(y_1, \ldots, y_{n-1}) = B'(y_1, \ldots, y_{n-1}, \bot)$ is $(6\varepsilon, (e^{2\varepsilon} + e^{5\varepsilon})\delta)$-differentially private.

Finally, uniform convergence of the sampling error of $B_0$ implies that it remains an accurate algorithm, and hence so is $B$. In particular, when $D_0$ is a random sample of $m$ rows from $D$ and $d$ is the VC-dimension of $Q$, we have [1]:

$$\Pr[\exists q \in Q : |q(D) - q(D')| > \alpha'] \leq 4 \cdot \left(\frac{2em}{d}\right)^d \cdot \exp\left(-\frac{(\alpha')^2m}{8}\right).$$

Taking $\alpha'$ as in the theorem statement makes the total failure probability of $B$ at most $1/2$. 

We now proceed to complete the proof of Theorem 3.2.16. Suppose $(Q, \mathcal{X})$ has sample complexity greater than $n$ for $(\alpha + \alpha', \gamma)$-accuracy (with failure probability $1/2$) and $(6\varepsilon, (e^{2\varepsilon} + e^{5\varepsilon})\delta)$-differential privacy. By Lemma 3.2.17, for every $(\alpha, \gamma)$-accurate mechanism $A$ for $Q$ there exists a dataset $D \in \mathcal{X}^m$ with $m = \lfloor n\varepsilon \rfloor$, a set $T$, and an index $i$ such that

$$\Pr_{\text{coins}(A)} [A(D_{-j}) \in T] > e^\varepsilon \Pr_{\text{coins}(A)} [A(D_{-i}) \in T] + \delta. \quad (3.4)$$

We now argue that it is without loss of generality to restrict our attention to mechanisms $A$ whose range is the finite set $I_m^{1/|Q|} = \{0, \frac{1}{2m}, \frac{1}{m}, \ldots, 1 - \frac{1}{2m}, 1\}^{1/|Q|}$. To see this, note that the exact answer to any counting query $q$ on a dataset $D \in \mathcal{X}^m$ is in the set $\{0, \frac{1}{m}, \frac{2}{m}, \ldots, 1 - \frac{1}{m}, 1\}$. Therefore, if an answer $a \in [0, 1]$ satisfies $|a - q(D)| \leq \alpha$, then the value

$$\bar{a} = \frac{1}{2m} \cdot ((a - \alpha)m] + [(a + \alpha)m]$$

is a point in $I_m$ that also satisfies $|\bar{a} - q(D)| \leq \alpha$. Thus, we will henceforth assume that the mechanism’s output lies in this finite range.

We now apply the min-max theorem from game theory (or equivalently, linear programming duality), to exhibit a fixed distribution on $(D, T, i)$ for which Inequality (3.4) holds. Specifically, consider a two-player zero-sum game in which Player 1 chooses a triple $(D, T, i)$, where $D \in \mathcal{X}^m$, $T \subseteq I_m^{1/|Q|}$, and $i \in [m]$, and Player 2 chooses a randomized function $A : \mathcal{X}^m \rightarrow I_m^{1/|Q|}$ that is
Let the payoff to Player 1 be

$$\Pr_{j \leftarrow_R [m]} [A(D_{-j}) \in T] - \epsilon^\ell I(A(D_{-i}) \in T).$$

By inequality (3.4), the value of this game is greater than $\delta$. So by the min-max theorem there exists a mixed strategy for Player 1 that achieves a payoff greater than $\delta$ against any mixed strategy for Player 2. (Note that we can apply the min-max theorem because we have assumed that the mechanism’s output lies in a finite range.) That is, there exists a distribution $D$ over triples $(D, T, i)$ such that for any randomized algorithm $A : \mathcal{X}^m \rightarrow I_m^{|Q|}$ that takes any $D$ to a feasible vector in $F_{\alpha, \gamma}(D)$ with probability at least $2/3$,

$$\Pr_{j \leftarrow_R [m]} [A(D_{-j}) \in T] > \epsilon^\ell \cdot \Pr_{coins(A)} [A(D_{-i}) \in T] + \delta. \tag{3.5}$$

Now consider the following code: $Gen$ samples a dataset $D = (x_1, \ldots, x_m)$, a set $T$, and an index $i$ according to the promised distribution $D$. The codebook $C$ is $(x_{\pi(1)}, \ldots, x_{\pi(m)})$ where $\pi : [m] \rightarrow [m]$ is a random permutation. On input an answer vector $a$, the algorithm $Trace$ checks whether $a \in T$. If it is, then $Trace$ outputs $\pi(i)$, and otherwise outputs $\bot$.

To analyze the security of this code, fix a coalition $S$ of $m - 1$ users using a pirate strategy $A_{FP}$. Because the codebook is a random permutation of the rows of $D$, it is equivalent to analyze the original dataset $D$ and a random coalition of $m - 1$ users. Thus the part of the codebook $C_S$ given to the pirates is a random set of $m - 1$ rows from $D$, i.e. $D_{-j}$ for a random $j \in [m]$ with the junk row at index $j$ removed. The condition that $A_{FP}$ outputs a feasible answer vector is equivalent to $a = A_{FP}(C_S)$ being an $(\alpha, \gamma)$-accurate answer vector. Therefore, letting $A : \mathcal{X}^m \rightarrow I_m^{|Q|}$ be the algorithm that runs $A_{FP}$ on its input with the junk row removed, we have

$$\Pr_{Gen, Trace, A_{FP}, coins(A_{FP})} [Trace(C, a) \neq \bot] = \Pr_{coins(A_{FP})} [A_{FP}(C_S) \in T] = \Pr_{j \leftarrow_R [m], coins(A)} [A(D_{-j}) \in T].$$
On the other hand, the probability that \( \text{Trace} \) outputs the user \( j \) not in the coalition is

\[
\Pr_{\text{Gen,Trace,AFP}}[\text{Trace}(C,a) = i] = \Pr_{j \sim \mathbb{R}[m], \text{coins}(\text{AFP})_{(D,T,i) \sim \mathbb{R}D}}[\text{Trace}(C,a) = i \land j = i] \\
= \frac{1}{m} \cdot \Pr_{\text{coins}(A), (D,T,i) \sim \mathbb{R}D}[A(D-i) \in T],
\]

because the events \( \{j = i\} \) and \( \{\text{Trace}(C,a) = i\} \) are independent. Thus by (3.5),

\[
\Pr[\text{Trace}(a) \neq \bot] > e^\varepsilon m \cdot \Pr[\text{Trace}(a) \in [m] \setminus S] + \delta,
\]

where both probabilities are taken over the coins of \( \text{Gen,Trace, and A}_{FP} \).

### 3.3 A Composition Theorem for Sample Complexity Lower Bounds

In this section we state and prove a composition theorem for sample complexity lower bounds. At a high-level the composition theorem starts with two pairs, \((Q, \mathcal{X})\) and \((Q', \mathcal{X}')\), for which we know sample-complexity lower bounds of \( n \) and \( n' \) respectively, and attempts to prove a sample-complexity lower bound of \( n \cdot n' \) for a related family of queries on a related data universe.

Specifically, our sample-complexity lower bound will apply to the “product” of \( Q \) and \( Q' \), defined on \( \mathcal{X} \times \mathcal{X}' \). We define the product \( Q \land Q' \) to be

\[
Q \land Q' = \{ q \land q' : (x,x') \mapsto q(x) \land q'(x') \mid q \in Q, q \in Q' \}.
\]

Since \( q, q' \) are boolean-valued, their conjunction can also be written \( q(x)q'(x') \).

We now begin to describe how we can prove a sample complexity lower bound for \( Q \land Q' \). First, we describe a certain product operation on datasets. Let \( D \in \mathcal{X}^n \), \( D = (x_1, \ldots, x_n) \), be a dataset. Let \( D'_1, \ldots, D'_n \in (\mathcal{X}')^n \) where \( D'_i = (x'_{i1}, \ldots, x'_{in}) \) be \( n \) datasets. We define the product dataset \( D^* = D \times (D'_1, \ldots, D'_n) \in (\mathcal{X} \times \mathcal{X}')^n \) as follows: For every \( i = 1, \ldots, n, j = 1, \ldots, n' \), let the \((i,j)\)-th row of \( D^* \) be \( x^*_{(i,j)} = (x_i, x'_{ij}) \). Note that we index the rows of \( D^* \) by \((i,j)\). We will sometimes refer to \( D'_1, \ldots, D'_n \) as the “subdatasets” of \( D^* \).

The key property of these datasets is that we can use a query \( q \land q' \in Q \land Q' \) to compute a “subset-sum” of the vector \( s_{q'} = (q'(D'_1), \ldots, q'(D'_n)) \) consisting of the answers to \( q' \) on each of the \( n \)
subdatasets. That is, for every \( q \in Q \) and \( q' \in Q' \),
\[
(q \land q')(D^*) = \frac{1}{n \cdot n'} \sum_{i=1}^{n} \sum_{j=1}^{n'} (q \land q')(x^*_{i,j}) = \frac{1}{n} \sum_{i=1}^{n} q(x_i)q'(D'_i). \tag{3.6}
\]

Thus, every approximate answer \( a_{q \land q'} \) to a query \( q \land q' \) places a subset-sum constraint on the vector \( s_{q'} \). (Namely, \( a_{q \land q'} \approx \frac{1}{n} \sum_{i=1}^{n} q(x_i)q'(D'_i) \)) If the dataset \( D \) and family \( Q \) are chosen appropriately, and the answers are sufficiently accurate, then we will be able to reconstruct a good approximation to \( s_{q'} \). Indeed, this sort of “reconstruction attack” is the core of many lower bounds for differential privacy, starting with the work of Dinur and Nissim [41]. The setting they consider is essentially the special case of what we have just described where \( D'_1, \ldots, D'_n \) are each just a single bit (\( X' = \{0, 1\} \), and \( Q' \) contains only the identity query). In Section 3.4 we will discuss choices of \( D \) and \( Q \) that allow for this reconstruction.

We now state the formal notion of reconstruction attack that we want \( D \) and \( Q \) to satisfy.

**Definition 3.3.1 (Reconstruction Attacks).** Let \( Q \) be a family of counting queries over a data universe \( X \). Let \( n \in \mathbb{N} \) and \( \alpha', \alpha, \gamma \in [0, 1] \) be parameters. Let \( D = (x_1, \ldots, x_n) \in X^n \) be a dataset. Suppose there is an adversary \( B_D : \mathbb{R}^{|Q|} \to [0, 1]^n \) with the following property: For every vector \( s \in [0, 1]^n \) and every sequence \( a = (a_q)_{q \in Q} \in \mathbb{R}^{|Q|} \) such that
\[
\left| a_q - \frac{1}{n} \sum_{i=1}^{n} q(x_i)s_i \right| < \alpha
\]
for at least a \( 1 - \gamma \) fraction of queries \( q \in Q \), \( B_D(a) \) outputs a vector \( t \in [0, 1]^n \) such that
\[
\frac{1}{n} \sum_{i=1}^{n} |t_i - s_i| \leq \alpha'.
\]
Then we say that \( D \in X^n \) enables an \( \alpha' \)-reconstruction attack from \( (\alpha, \gamma) \)-accurate answers to \( Q \).

A reconstruction attack itself implies a sample-complexity lower bound, as in [41]. However, we show how to obtain stronger sample complexity lower bounds from the reconstruction attack by applying it to a product dataset \( D^* \) to obtain accurate answers to queries on its subdatasets. For each query \( q' \in Q' \), we run the adversary promised by the reconstruction attack on the approximate answers given to queries of the form \( (q \land q') \in Q \land \{q'\} \). As discussed above, answers to these queries will approximate subset sums of the vector \( s_{q'} = (q'(D'_1), \ldots, q'(D'_n)) \). When the reconstruction attack is given these approximate answers, it returns a vector \( t_{q'} = (t_{q',1}, \ldots, t_{q',n}) \) such that
\[ t_{q',i} \approx s_{q',i} = q'(D'_i) \] on average over \( i \). Running the reconstruction attack for every query \( q' \) gives us a collection \( t = (t_{q',i})_{q' \in Q', i \in [n]} \) where \( t_{q',i} \approx q'(D'_i) \) on average over both \( q' \) and \( i \). By an application of Markov’s inequality, for most of the subdatasets \( D'_i \), we have that \( t_{q',i} \approx q'(D'_i) \) on average over both \( q' \) and \( i \). By an application of Markov’s inequality shows that for most queries \( q' \in Q' \) we have \( t_{q',i} \approx q'(D'_i) \), which is our definition of \((\alpha, \gamma)\)-accuracy (later enabling us to apply a re-identification adversary for \( Q' \)).

The algorithm we have described for obtaining accurate answers on the subdatasets is formalized in Figure 3.1.

Let \( a = (a_{q \wedge q'})_{q \in Q, q' \in Q'} \) be an answer vector.
Let \( B_D : \mathbb{R}^{Q} \to \{0, 1\}^n \) be a reconstruction attack.
For each \( q' \in Q' \)
- Let \( (t_{q',1}, \ldots, t_{q',n}) = B_D((a_{q \wedge q'})_{q \in Q}) \)
- Output \( (t_{q',i})_{q' \in Q', i \in [n]} \).

**Figure 3.1:** The reconstruction \( R^*_D(a) \).

We are now in a position to state the main lemma that enables our composition technique. The lemma says that if we are given accurate answers to \( Q \wedge Q' \) on \( D^* \) and the dataset \( D \in X^n \) enables a reconstruction attack from accurate answers to \( Q \), then we can obtain accurate answers to \( Q' \) on most of the subdatasets \( D'_1, \ldots, D'_n \in (X')^n \).

**Lemma 3.3.2.** Let \( D \in X^n \) and \( D'_1, \ldots, D'_n \in (X')^n \) be datasets and \( D^* \in (X \times X')^{n-n'} \) be as above. Let \( a = (a_{q \wedge q'})_{q \in Q, q' \in Q'} \in \mathbb{R}^{Q \wedge Q'} \). Let \( \alpha', \alpha, \gamma \in [0, 1] \) be parameters. Suppose that for some parameter \( c > 1 \), the dataset \( D \) enables an \( \alpha' \)-reconstruction attack from \((\alpha, c\gamma)\)-accurate answers to \( Q \). Then if \( (t_{q',i})_{q' \in Q', i \in [n]} = R^*_D(a) \) (Figure 3.1),

- \( a \) is \((\alpha, \gamma)\)-accurate for \( Q \wedge Q' \) on \( D^* \)
- \( \Pr_{i \leftarrow [n]} [(t_{q',i})_{q' \in Q'} \text{ is } (6\alpha'c, 2/c)\text{-accurate for } Q' \text{ on } D_i] \geq 5/6. \)

The additional bookkeeping in the proof is to handle the case where \( a \) is only accurate for most queries. In this case the reconstruction attack may fail completely for certain queries \( q' \in Q' \) and we need to account for this additional source of error.

**Proof of Lemma 3.3.2.** Assume the answer vector \( a = (a_{q \wedge q'})_{q \in Q, q' \in Q'} \) is \((\alpha, \gamma)\)-accurate for \( Q \wedge Q' \)

58
on $D^* = D \times (D_1', \ldots, D_n')$. By assumption, $D$ enables a reconstruction attack $B_D$ that succeeds in reconstructing an approximation to $s_{q'} = (q'(D_1'), \ldots, q'(D_n'))$ when given $(\alpha, c\gamma)$-accurate answers for the family of queries $Q \land \{q'\}$. Consider the set of $q'$ on which the reconstruction attack succeeds, i.e.

$$Q'_{\text{good}} = \{ q' \mid (a_{q \land q'})_{q \in Q} \text{ is } (\alpha, c\gamma)\text{-accurate for } Q \land \{q'\} \}.$$

Since $a$ is $(\alpha, \gamma)$-accurate, an application of Markov’s inequality shows that

$$\Pr \left[ q' \in Q'_{\text{good}} \right] \geq 1 - 1/c.$$ 

Thus, $|Q'_{\text{good}}| \geq (1 - 1/c)|Q'|$.

Recall that, by (3.6), we can interpret answers to $Q \land Q'$ as subset sums of answers to the subdatasets, so for every $q' \in Q'_{\text{good}},$

$$\left| a_{q \land q'} - \frac{1}{n} \sum_{i=1}^{n} q(x_i)q'(D'_i) \right| < \alpha$$

for at least a $1 - c\gamma$ fraction of queries $q \land q' \in Q \land \{q'\}$. Since $D$ enables a reconstruction attack from $(\alpha, c\gamma)$-accurate answers to $Q$, by Definition 3.3.1, $B_D((a_{q \land q'})_{q \in Q})$ recovers a vector $t_{q'} \in [0, 1]^n$ such that

$$\frac{1}{n} \sum_{i=1}^{n} |t_{q',i} - q'(D'_i)| < \alpha'.$$

Since this holds for every $q' \in Q'_{\text{good}},$ we have

$$\frac{1}{n} \mathbb{E}_{q' \leftarrow R_{Q'_{\text{good}}}, i \leftarrow R_{[n]}} \left[ |t_{q',i} - q'(D'_i)| \right] \leq \alpha'$$

$$\Rightarrow \mathbb{P}_{i \leftarrow R_{[n]}} \left[ \mathbb{E}_{q' \leftarrow Q'_{\text{good}}} \left[ |t_{q',i} - q'(D'_i)| \right] \leq 6\alpha' \right] \geq 5/6 \quad \text{(Markov)}$$

$$\Rightarrow \mathbb{P}_{i \leftarrow R_{[n]}} \left[ |t_{q',i} - q'(D'_i)| \leq 6\alpha' \text{ for at least a } 1 - 1/c \text{ fraction of } q' \in Q'_{\text{good}} \right] \geq 5/6 \quad \text{(Markov)}$$

$$\Rightarrow \mathbb{P}_{i \leftarrow R_{[n]}} \left[ |t_{q',i} - q'(D'_i)| \leq 6\alpha' \text{ for at least a } 1 - 2/c \text{ fraction of } q' \in Q' \right] \geq 5/6$$

(since $|Q'_{\text{good}}| \geq (1 - 1/c)|Q'|$)

The statement inside the final probability is precisely that $(t_{q',i})_{q' \in Q'}$ is $(6\alpha', 2/c)$-accurate for $Q'$ on $D'_i$. This completes the proof of the lemma.

We now explain how the main lemma allows us to prove a composition theorem for sample
complexity lower bounds. We start with a query family $\mathcal{Q}$ on a dataset $D \in \mathcal{X}^n$ that enables a reconstruction attack, and a distribution $\mathcal{D}'$ over datasets in $(\mathcal{X}')^{n'}$ that is re-identifiable from answers to a family $\mathcal{Q}'$. We show how to combine these objects to form a re-identifiable distribution $\mathcal{D}^*$ for queries $\mathcal{Q} \land \mathcal{Q}'$ over $(\mathcal{X} \times \mathcal{X}')^{n-n'}$, yielding a sample complexity lower bound of $n \cdot n'$.

A sample from $\mathcal{D}^*$ consists of $D^* = D \times (D'_1, \ldots, D'_{n'})$ where each subdataset $D'_i$ is an independent sample from $D_i'$. The main lemma above shows that if there is an algorithm $A$ that is accurate for $\mathcal{Q} \land \mathcal{Q}'$ on $D^*$, then an adversary can reconstruct accurate answers to $\mathcal{Q}'$ on most of the subdatasets $D'_1, \ldots, D'_{n'}$. Since these subdatasets are drawn from a re-identifiable distribution, the adversary can re-identify a member of one of the subdatasets $D'_i$. Since the identified member of $D'_i$ is also a member of $D^*$, we will have a re-identification attack against $D^*$ as well.

We are now ready to formalize our composition theorem.

**Theorem 3.3.3.** Let $\mathcal{Q}$ be a family of counting queries on $\mathcal{X}$, and let $\mathcal{Q}'$ be a family of counting queries on $\mathcal{X}'$. Let $\eta, \xi, \alpha', \alpha, \gamma \in [0, 1]$ be parameters. Assume that for some parameters $c > 1$, $\eta, \xi, \alpha', \alpha, \gamma \in [0, 1]$, the following both hold:

1. There exists a dataset $D \in \mathcal{X}^n$ that enables an $\alpha'$-reconstruction attack from $(\alpha, c\gamma)$-accurate answers to $\mathcal{Q}$.

2. There is a distribution $\mathcal{D}'$ on datasets $D \in (\mathcal{X}')^{n'}$ that is $(\eta, \xi)$-re-identifiable from $(6c\alpha', 2/c)$-accurate answers to $\mathcal{Q}'$.

Then there is a distribution on datasets $D^* \in (\mathcal{X} \times \mathcal{X}')^{n-n'}$ that is $(\eta + 1/6, \xi)$-re-identifiable from $(\alpha, \gamma)$-accurate answers to $\mathcal{Q} \land \mathcal{Q}'$.

**Proof.** Let $D = (x_1, \ldots, x_n) \in \mathcal{X}^n$ be the dataset that enables a reconstruction attack (Definition 3.3.1). Let $\mathcal{D}'$ be the promised re-identifiable distribution on datasets $D \in (\mathcal{X}')^{n'}$ and $\mathcal{B}' : (\mathcal{X}')^{n'} \times [\mathcal{Q}' \mid Q' \rightarrow [n'] \cup \{\perp\}$ be the promised adversary (Definition 3.2.1).

In Figure 3.2, we define a distribution $\mathcal{D}^*$ on datasets $D' \in (\mathcal{X} \times \mathcal{X}')^{n-n'}$. In Figure 3.3, we define an adversary $\mathcal{B}^* : (\mathcal{X} \times \mathcal{X}')^{n-n'} \times [\mathcal{Q} \land \mathcal{Q}' \mid (n') \rightarrow [n'] \cup \{\perp\}$ for a re-identification attack. The shared state of $\mathcal{D}^*$ and $\mathcal{B}^*$ will be the shared state of $\mathcal{D}'$ and $\mathcal{B}'$. The next two claims show that $\mathcal{D}^*$ satisfies the two properties necessary to be a $(\eta + 1/6, \xi)$-re-identifiable distribution (Definition 3.2.1).
Let $D = (x_1, \ldots, x_n) \in \mathcal{X}^n$ be a dataset that enables reconstruction. Let $\mathcal{D}'$ on $(\mathcal{X}')^n$ be a re-identifiable distribution.

For $i = 1, \ldots, n$, choose $D'_i \leftarrow \mathcal{D}'$ (independently)

Output $D^* = D \times (D'_1, \ldots, D'_n) \in (\mathcal{X} \times \mathcal{X}')^n$.

**Figure 3.2:** The new distribution $\mathcal{D}^*$.

Let $D^* = D \times (D'_1, \ldots, D'_n)$.

Run $\mathcal{R}_\mathcal{D}^*(\mathcal{A}(D^*))$ (Figure 3.1) to reconstruct a set of approximate answers $(t_{q',i}, q' \in \mathcal{Q}, i \in [n])$.

Choose a random $i \leftarrow [n]$.

Output $B'(D'_i, (t_{q',i}, q' \in \mathcal{Q}^*)$.

**Figure 3.3:** The privacy adversary $B^*(D^*, \mathcal{A}(D^*))$.

**Claim 3.3.4.**\[ \Pr_{D^* \leftarrow \mathcal{D}^*}[(B^*(D^*, \mathcal{A}(D^*)) = \bot) \land (\mathcal{A}(D^*) \text{ is } (\alpha, \gamma)\text{-accurate for } \mathcal{Q} \land \mathcal{Q}')] \leq \eta + 1/6.\]

**Proof of Claim 3.3.4.** Assume that $\mathcal{A}(D^*)$ is $(\alpha, \gamma)\text{-accurate for } \mathcal{Q} \land \mathcal{Q}'$. By Lemma 3.3.2, we have

\[ \Pr_{i \leftarrow [n]}[(A(D^*) \text{ is } (\alpha, \gamma)\text{-accurate for } \mathcal{Q} \land \mathcal{Q}')] \leq 1/6. \] \hspace{1cm} (3.7)

By construction of $B^*$,

\[ \Pr_{D^* \leftarrow \mathcal{D}^*}[(B^*(D^*, \mathcal{A}(D^*)) = \bot) \land (\mathcal{A}(D^*) \text{ is } (\alpha, \gamma)\text{-accurate for } \mathcal{Q} \land \mathcal{Q}')] \leq \Pr_{i \leftarrow [n]}[(B'(D'_i, (t_{q',i}, q' \in \mathcal{Q}^*) = \bot) \land ((t_{q',i}, q' \in \mathcal{Q'}) \text{ is } (6\alpha', 2/c)\text{-accurate for } \mathcal{Q}')] + \frac{1}{6}. \] \hspace{1cm} (3.8)

where the last inequality is by (3.7). Thus, it suffices to prove that

\[ \Pr_{i \leftarrow [n]}[(B'(D'_i, (t_{q',i}, q' \in \mathcal{Q}^*) = \bot) \land ((t_{q',i}, q' \in \mathcal{Q'}) \text{ is } (6\alpha', 2/c)\text{-accurate for } \mathcal{Q}')] \leq \eta \] \hspace{1cm} (3.9)

We prove this inequality by giving a reduction to the re-identifiability of $\mathcal{D}'$. Consider the following sanitizer $\mathcal{A}'$: On input $D' \leftarrow \mathcal{D}'$, $\mathcal{A}'$ first chooses a random index $i^* \leftarrow [n]$. Next, it samples $D'_1, \ldots, D'_{i^*-1}, D'_{i^*+1}, \ldots, D'_n \leftarrow \mathcal{D}'$ independently, and sets $D'_{i^*} = D'$. Finally, it runs $\mathcal{A}$ on
\(D^* = D \times (D'_1, \ldots, D'_n)\) and then runs the reconstruction attack \(R^*\) to recover answers \((t_{q',i})_{q' \in Q'; i \in [n]}\) and outputs \((t_{q',i})_{q' \in Q'}\).

Notice that since \(D'_1, \ldots, D'_n\) are all i.i.d. samples from \(D'\), their joint distribution is independent of the choice of \(i^*\). Specifically, in the view of \(B^*\), we could have chosen \(i^*\) after seeing its output on \(D^*\). Therefore, the following random variables are identically distributed:

1. \((t_{q',i})_{q' \in Q'}, (t_{q',i})_{q' \in Q'; i \in [n]}\) is the output of \(R^*_D(A(D^*))\) on \(D^* \leftarrow R D^*, i \leftarrow [n]\).

2. \(A'(D')\) where \(D' \leftarrow R D'\).

Thus we have

\[
\Pr_{D^* \leftarrow R D^*} \left[ (B'_i(D'_i, (t_{q',i})_{q' \in Q'} = \bot) \land ((t_{q',i}) \text{ is } (6c\alpha', 2/c)-\text{accurate for } Q') \right] \\
= \Pr_{D' \leftarrow R D'} \left[ (B'(D', A'(D')) = \bot) \land (A'(D') \text{ is } (6c\alpha', 2/c)-\text{accurate for } Q') \right] \leq \eta
\]

where the last inequality follows because \(D'\) is a \((\eta, \xi)\)-re-identifiable from \((6c\alpha', 2/c)\)-accurate answers to \(Q'\). Thus we have established (3.9). Combining (3.8) and (3.9) completes the proof of the claim.

The next claim follows directly from the definition of \(B^*\) and the fact that \(D'\) is \((\eta, \xi)\)-re-identifiable.

**Claim 3.3.5.** For every \((i, j) \in [n] \times [n']\),

\[
\Pr_{D \leftarrow R D^*} \left[ B^*(D, A(D_{-i,j})) = (i, j) \right] \leq \xi.
\]

Combining Claims 3.3.4 and 3.3.5 suffices to prove that \(D^*\) is \((\eta + 1/6, \xi)\)-re-identifiable from \((\alpha, \gamma)\)-accurate answers to \(Q \land Q'\), completing the proof of the theorem.

The proof of Theorem 3.3.3 also yields a composition theorem for generalized fingerprinting codes. Specifically, Theorem 3.3.6 below shows how to combine a reconstruction attack for a query family \(Q\) on a dataset \(D \in X^n\) with a \((n', Q')\)-generalized fingerprinting code to obtain a \((n \cdot n', Q \land Q')\)-generalized fingerprinting code.
**Theorem 3.3.6.** Let $Q$ be a family of counting queries on $\mathcal{X}$, and let $Q'$ be a family of counting queries on $\mathcal{X}'$. Let $\eta, \xi, \alpha', \alpha, \gamma \in [0,1]$ be parameters. Assume that for some parameters $c > 1$, $\eta, \xi, \alpha', \alpha, \gamma \in [0,1]$, the following both hold:

1. There exists a dataset $D \in \mathcal{X}^n$ that enables an $\alpha'$-reconstruction attack from $(\alpha, c\gamma)$-accurate answers to $Q$.

2. There exists a $(n', Q')$-generalized fingerprinting code for $(6c\alpha', 2/c)$-accuracy with security $(\eta, \xi)$.

Then there is a $(n \cdot n', Q \land Q')$-generalized fingerprinting code for $(\alpha, \gamma)$-accuracy with security $(\eta + 1/6, \xi)$.

### 3.4 Lower Bounds for $k$-Way Marginals

In this section we show how to use our composition theorem (Section 3.3) to combine our new lower bounds for 1-way marginal queries from Section 3.2.2 with (variants of) known lower bounds from the literature to obtain our main results. That is, we prove a lower bound for $k$-way marginal queries when $\alpha$ is not too small (at least inverse polynomial in $d$), thereby proving Theorem 3.1.2 in the introduction.

#### 3.4.1 Lower Bounds for $k$-Way Marginals

In this section, we carry out the composition of sample complexity lower bounds for $k$-way marginals as described in the introduction (Theorem 3.1.2). Recall that we obtain our new $\tilde{\Omega}(k\sqrt{d}/\alpha^2)$ lower bound by combining three lower bounds:

1. Our re-identification based $\tilde{\Omega}(\sqrt{d})$ lower bound for 1-way marginals (Section 3.2.2),

2. A known reconstruction-based lower bound of $\Omega(k)$ for $k$-way marginals.

3. A known reconstruction-based lower bound of $\Omega(1/\alpha^2)$ for $k$-way marginals.

The lower bound of $\Omega(k)$ for $k$-way marginals is a special case of a lower bound of $\Omega(\text{VC}(Q))$ due to [104] and based on [41], where $\text{VC}(Q)$ is the Vapnik-Chervonenkis (VC) dimension of $Q$. The lower bound of $\Omega(1/\alpha^2)$ for $k$-way marginals is due to [40,82].
To apply our composition theorem, we need to formulate these reconstruction attacks in the language of Definition 3.3.1. In particular, we observe that these reconstruction attacks readily generalize to allow us to reconstruct fractional vectors $s \in [0,1]^n$, instead of just boolean vectors as in [41,104].

The $\Omega(k)$ Lower Bound

First we state and prove the fact that the linear dependence on $k$ is necessary.

**Fact 3.4.1.** The set of $k$-way conjunctions $\mathcal{M}_{k,d}$ over any data universe $\{0,1\}^d$ with $d \geq k$ has VC-dimension $\text{VC}(\mathcal{M}_{k,d}) \geq k$.\(^5\)

**Proof.** For each $i = 1, \ldots, k$, let $x_i = (1, 1, \ldots, 0, \ldots, 1)$ where the zero is at the $i$-th index. We will show that $\{x_1, \ldots, x_k\}$ is shattered by $\mathcal{M}_{k,d}$. For a string $v \in \{0,1\}^k$, let the query $q_v(x)$ take the conjunction of the bits of $x$ at indices set to 0 in $v$. Then $q_v(x_i) = 1$ iff $v_i = 1$, so $(q_v(x_1), \ldots, q_v(x_k)) = (v_1, \ldots, v_k)$.

---

\(^5\)More precisely, $\text{VC}(\mathcal{M}_{k,d}) \geq k \log_2(|d/k|)$, but we use the simpler bound $\text{VC}(\mathcal{M}_{k,d}) \geq k$ to simplify calculations, since our ultimate lower bounds are already suboptimal by polylog($d$) factors for other reasons.
Input: Queries \( Q \), and \((a_q)_{q \in Q}\) that are \((\alpha,0)\)-accurate for \( s \).
Find any \( t \in [0,1]^n \) such that

\[
|q - \frac{1}{n} \sum_{i=1}^{n} q(x_i) t_i| \leq \alpha \quad \forall q \in Q.
\]

Output: \( t \).

Figure 3.4: The reconstruction adversary \( B(D,a) \).

Let \( T \) be the set of coordinates on which \( t_i > s_i \) and let \( S \) be the set of coordinates where \( s_i > t_i \). Note that

\[
\sum_{i=1}^{n} |t_i - s_i| = \sum_{i \in T} (t_i - s_i) + \sum_{i \in S} (s_i - t_i).
\]

We will show that absolute values of the sums over \( T \) and \( S \) are each at most \( 2\alpha \). Since \( \{x_1, \ldots, x_n\} \) is shattered by \( Q \), there is a query \( q \in Q \) such that \( q(x_i) = 1 \) iff \( i \in T \). Therefore, by the definitions of \( t \) and \((\alpha,0)\)-accuracy,

\[
|q - \frac{1}{n} \sum_{i=1}^{n} q(x_i) t_i| = \left| q - \frac{1}{n} \sum_{i \in T} t_i \right| \leq \alpha \quad \text{and} \quad \left| q - \frac{1}{n} \sum_{i \in S} s_i \right| \leq \alpha,
\]

so by the triangle inequality, \( \frac{1}{n} \sum_{i \in T} (t_i - s_i) \leq 2\alpha \). An identical argument shows that \( \frac{1}{n} \sum_{i \in S} (s_i - t_i) \leq 2\alpha \), proving that \( t \) is an accurate reconstruction.

The \( \Omega(1/\alpha^2) \) Lower Bound for \( k \)-Way Marginals

We can now state in our terminology the lower bound of De from [40] (building on [82]) showing that the inverse-quadratic dependence on \( \alpha \) is necessary.

**Theorem 3.4.3** (Restatement of [40]). Let \( k \) be any constant, \( d \geq k \) be any integer, and let \( \alpha \geq 1/d^{k/3} \) be a parameter. There exists a constant \( \gamma = \gamma(k) > 0 \) such that for every \( \alpha' > 0 \), there exists a dataset \( D \in (\{0,1\}^d)^n \) with \( n = \Omega_{\alpha',k}(1/\alpha^2) \) such that \( D \) enables an \( \alpha' \)-reconstruction attack from \((\alpha,\gamma)\)-accurate answers to \( \mathcal{M}_{k,d} \).

Although the above theorem is a simple extension of De’s lower bound, we sketch a proof for completeness, and refer the interested reader to [40] for a more detailed analysis.

**Proof Sketch.** The reconstruction attack uses the “\( \ell_1 \)-minimization” algorithm, which is shown in
To prove that the reconstruction attack succeeds, we will show that there exists a dataset

\begin{figure}[h]
\centering
\begin{align*}
\text{Input:} & \text{ Queries } Q, D = (x_1, \ldots, x_n) \in \{0,1\}^{n \times d}, \text{ and } a = (a_q)_{q \in Q}. \\
\text{Let } t \in [0,1]^n \text{ be } & \arg \min_{t \in [0,1]^n} \sum_{q \in Q} \left| a_q - \frac{1}{n} \sum_{i=1}^n q(x_i) t_i \right| \\
\text{Output:} & t.
\end{align*}
\caption{The reconstruction adversary $B_Q(D,a)$.}
\end{figure}

\begin{align*}
D = (x_1, \ldots, x_n) & \in \{0,1\}^{n \times d} \text{ such that for any } s \in [0,1]^n, \text{ if } a \text{ satisfies } \\
\Pr_{q \in \mathcal{M}_{k,d}} \left[ \left| a_q - \frac{1}{n} \sum_{i=1}^n q(x_i) s_i \right| \leq \alpha \right] & \geq 1 - \gamma,
\end{align*}

(i.e. $a$ has $(\alpha, \gamma)$-accurate answers) then $B_{\mathcal{M}_{k,d}}(D,a)$ returns a vector $t$ such that $\|t - s\|_1 \leq \alpha' \cdot n$. Henceforth we refer to such an $a$ simply as $(\alpha, \gamma)$-accurate for $\mathcal{M}_{k,d}$ on $(D,s)$, as a shorthand. The above guarantee must hold for suitable choices of $n, \gamma,$ and $\alpha'$ to satisfy the theorem.

We will argue that the reconstruction succeeds in two steps. First, we show that reconstruction succeeds if $D$ is “nice.” Second, we show that there exists “nice” $D$ that has the dimensions promised by the theorem.

To explain what we mean by a “nice” dataset $D$, for any $D = (x_1, \ldots, x_n) \in \{0,1\}^{n \times d}$ and family of queries $Q$ on $\{0,1\}^d$, we define the matrix $M = M_{D,Q} \in \{0,1\}^{n \times |Q|}$, as $M(i,q) = q(x_i)$.

De analyzes this reconstruction attack in terms of certain properties of the matrix $M$. Before stating the conclusion, we will need to define the notion of a Euclidean section. Informally, a matrix $M$ is a Euclidean Section if its rowspace contains only vectors that are “spread out.”

\begin{definition}[Euclidean Section] A matrix $M \in \{0,1\}^{n \times m}$ is a $\delta$-Euclidean section if for every vector $a$ in the rowspace of $M$ we have $\sqrt{m} \cdot \|a\|_2 \geq \|a\|_1 \geq \delta \sqrt{m} \cdot \|a\|_2$.
\end{definition}

\begin{lemma}[(40)] Let $D$ be a dataset and $Q$ be a set of queries such that $M_{D,Q} \in \{0,1\}^{n \times |Q|}$ is a $\delta$-Euclidean section and the least singular value of $M_{D,Q}$ is $\sigma$. Let $s \in [0,1]^n$ be arbitrary. There exists $\gamma = \gamma(\delta) > 0$ such that if $a$ are $(\alpha, \gamma)$-accurate answers for $Q$ on $(D,s)$, and $t = B_Q(D,a)$,
\end{lemma}

\footnote{For a matrix $M$ with rows $M_1, \ldots, M_n$, the rowspace of $M$ is $\{a = \sum_{i=1}^n c_i M_i \mid c_1, \ldots, c_n \in \mathbb{R}\}$.}
then $t$ satisfies

$$
\|s - t\|_1 \leq \eta \cdot n
$$

for $\eta = O(\alpha \sqrt{n} |Q| / \sigma)$. The constant hidden in the $O(\cdot)$ notation depends only on $\delta$.

Thus, it suffices to find dataset $D$ such that the matrix $M_{D, M_{k,d}}$ is a Euclidean section (for some fixed constant $\delta > 0$) and has no “small” singular values. A result of Rudelson [106] (strengthening that of Kasiviswanathan et al. [82]) guarantees that such a dataset exists.

**Lemma 3.4.6 ([106]).** Let $k \in \mathbb{N}$ be any constant. Let $d, n \in \mathbb{N}$ be such that $d^k \geq n \log n$. Let $D \in \{0, 1\}^{n \times d}$ be a uniform random matrix. Then with probability at least 9/10, the matrix $M_{D, M_{k,d}}$ defined above has least singular value at least $\sigma = \Omega(d^{k/2})$ (where the hidden constant in the $\Omega(\cdot)$ may depend on $k$) and is a $\delta$-Euclidean section for some constant $\delta > 0$ that depends only on $k$.

In particular, there exists a dataset $D \in \{0, 1\}^{n \times d}$ such that the Hadamard product $M$ satisfies the two properties above.

Using the above lemma, we can now complete the proof. Fix any constant $k \in \mathbb{N}$. Let $\alpha, d, n$ be any parameters such that $d \geq k$, $\alpha \geq 1/d^{k/3}$, and $d^k \geq n \log n$. The precise value of $n$ will be determined later. Let $D \in \{0, 1\}^{n \times d}$ be the dataset promised by Lemma 3.4.6. Let $\gamma = \gamma(k) > 0$ be a parameter to be chosen later. Let $\alpha' > 0$ be the desired accuracy of the reconstruction attack.

Now fix any $s \in [0, 1]^n$ and let $a \in [0, 1]^{\mathcal{M}_{k,d}}$ be $(\alpha, \gamma)$-accurate answers to $\mathcal{M}_{k,d}$ on $(D, s)$. Now, if we let $t = B_{\mathcal{M}_{k,d}}(D, a)$, by Lemma 3.4.5, provided that $\gamma$ is smaller than some constant that depends only on $\delta$, which in turn depends only on $k$, we will have $\|s - t\|_1 \leq \eta \cdot n$ for

$$
\eta = O\left(\frac{\alpha \sqrt{n} |Q|}{\sigma}\right) = O\left(\frac{\alpha \sqrt{n} (d^k/k)^{k/2}}{d^{k/2}}\right) = O(\alpha \sqrt{n}).
$$

Note that by Lemma 3.4.5, the hidden constant in the $O(\cdot)$ notation depends only on the parameter $\delta$ such that $M_{D, M_{k,d}}$ is a $\delta$-Euclidean section. By Lemma 3.4.6, the parameter $\delta$ depends only on $k$. Thus $\eta = O(\alpha \sqrt{n})$ where the hidden constant depends only on $k$. Now, we can choose $n = \Omega(1/\alpha^2)$ such that $\eta \leq \alpha'$. The hidden constant in the $\Omega(\cdot)$ will depend only on $k$ and $\alpha'$, as required by the theorem. Note that, since we have assumed $\alpha \geq 1/d^{k/3}$, we have $n \log n = \tilde{O}(d^{2k/3})$, and so we

---

7Rudelson actually proves these statements about a related matrix $M_{D, Q}$ where $Q \subseteq \mathcal{M}_{k,d}$. Since, for the $Q$ he considers, $|Q| \geq |\mathcal{M}_{k,d}|/(2k)^{k}$, these statements can easily be seen to hold for the matrix $M_{D, \mathcal{M}_{k,d}}$ itself. Specifically, adding this many more columns to the matrix $M_{D, Q}$ cannot decrease its least singular value (since $M_{D, Q}$ already has more columns than rows), and can only decrease the Euclidean section parameter $\delta$ by a factor of at most $(2k)^k$. 

---

67
can define \( n = \Omega_{k, \alpha'}(1/\alpha^2) \) while ensuring that \( d^k \geq n \log n \). Similarly, we required that \( \gamma \) is smaller than some constant that depends only on \( \delta \), which in turn depends only on \( k \). Thus, we can set \( \gamma = \gamma(k) > 0 \) to be some sufficiently small constant depending only on \( k \), as required by the theorem.

This completes our sketch of the proof.

\[ \square \]

**Putting Together the Lower Bound**

Now we show how to combine the various attacks to prove Theorem 3.1.2 in the introduction. We obtain our lower bound by applying two rounds of composition. In the first round, we compose the reconstruction attack of Theorem 3.4.3 described above with the re-identifiable distribution for 1-way marginals. We then take the resulting re-identifiable distribution and apply a second round of composition using the reconstruction attack based on the VC-dimension of \( k \)-way marginals.

We remark that it is necessary to apply the two rounds of composition in this order. In particular, we cannot prove Theorem 3.1.2 by composing first with the VC-dimension-based reconstruction attack. Our composition theorem requires a re-identifiable distribution from \((\alpha, \gamma)\)-accurate answers for \( \gamma > 0 \), whereas the reconstruction attack described in Lemma 3.4.2 requires \((\alpha, 0)\)-accurate answers, and the reconstruction can fail if some queries have error much larger than \( \alpha \). The resulting re-identifiable distribution obtained from composing with this reconstruction attack will also require \((\alpha, 0)\)-accurate answers, and thus cannot be composed further.

This limitation of Lemma 3.4.2 is inherent, because a sample complexity upper bound of \( \tilde{O}(\sqrt{d}/\alpha^2) \) can be achieved for answering any family of queries \( \mathcal{Q} \) with \((\alpha, \gamma)\)-accuracy (for any constant \( \gamma > 0 \)). Notice that this sample complexity is independent of \( \text{VC}(\mathcal{Q}) \).

We can now formally state and prove our sample-complexity lower bound for \( k \)-way marginals, thereby establishing Theorem 3.1.2 in the introduction.

**Theorem 3.4.7.** For every constant \( \ell \in \mathbb{N} \), every \( k, d \in \mathbb{N} \), \( \ell + 2 \leq k \leq d \), and every \( \alpha \geq 1/d^{\ell/3} \), there is an

\[ n = n(k, d, \alpha) = \tilde{\Omega}\left(\frac{k \sqrt{d}}{\alpha^2}\right) \]

such that there exists a distribution on \( n \)-row datasets \( D \in (\{0, 1\}^d)^n \) that is \((1/2, o(1/n))\)-re-identifiable from \((\alpha, 0)\)-accurate answers to the \( k \)-way marginals \( \mathcal{M}_{k,d} \).

**Proof.** We begin with the following two attacks:
1. By combining Theorem 3.2.3 and Theorem 2.6.4, there exists a distribution on datasets \( D' \in (\{0,1\}^{d/3})^{n_d} \) that is \((\eta = 1/6, \xi = o(1/n_d n_\alpha n_k))\)-re-identifiable from \((6c' = 1/3, 2/c = 1/75)\) accurate answers to the 1-way marginals \( M_{1,d/3} \) for \( n_d = \Omega(\sqrt{d}/\log(n_d n_\alpha n_k)) \). Here \( n_\alpha \) and \( n_k \) are set below (the subscript corresponds to the primary parameter that each of the \( n \)'s will depend on).

2. By Theorem 3.4.3 (with \( \alpha' = 1/2700 \) and \( k = \ell \)), there is a constant \( \gamma > 0 \) such that for any \( 7200\alpha/\gamma \geq 1/d^{\ell/3} \) there exists a dataset \( D \in (\{0,1\}^{d/3})^{n_\alpha} \), for \( n_\alpha = \Omega(1/\alpha^2) \) that enables a \((1/2700)\)-reconstruction attack from \((7200\alpha/\gamma, \gamma)\)-accurate answers to \( M_{\ell,d/3} \).

Applying Theorem 3.3.3 (with parameter \( c = 150 \)), we obtain item 1' below. We then bring in another reconstruction attack for the composition theorem.

1'. There exists a distribution on datasets in \((\{0,1\}^{2d/3})^{n_d n_\alpha} \) that is \((1/3, o(1/n_d n_\alpha n_k))\)-re-identifiable from \((6c' \alpha' = 7200\alpha/\gamma, 2/c' = \gamma/150)\)-accurate answers to \( \mathcal{M}_{\ell,d/3} \cap \mathcal{M}_{1,d/3} \subset \mathcal{M}_{\ell+1,2d/3} \) (By applying Theorem 3.3.3 to 1 and 2 above.)

2'. By Lemma 3.4.2 and Fact 3.4.1, there exists a dataset \( D \in (\{0,1\}^{d/3})^{n_k} \), for \( n_k = k - \ell - 1 \), that enables an \((\alpha' = 4\alpha)\)-reconstruction attack from \((\alpha,0)\)-accurate answers to the \((k - \ell - 1)\)-way marginals \( M_{k-\ell-1,d/3} \). Note that \((k - \ell - 1) \geq 1 \), since we have assumed \( k \geq \ell + 2 \).

We can then apply Theorem 3.3.3 to 1' and 2' (with parameter \( c' = 300/\gamma \)). Thereby we obtain a distribution \( D \) on datasets \( D \in (\{0,1\}^{d/3} \times \{0,1\}^{d/3} \times \{0,1\}^{d/3})^{n_d n_\alpha n_k} \) that is \((1/2, \xi)\)-re-identifiable from \((\alpha,0)\)-accurate answers to \( \mathcal{M}_{k-\ell-1,d/3} \cap \mathcal{M}_{\ell,d/3} \cap \mathcal{M}_{1,d/3} \subset \mathcal{M}_{k,d} \).

To complete the theorem, first note that \((\alpha,0)\)-accurate answers to \( \mathcal{M}_{k,d} \) imply \((\alpha,0)\)-accurate answers to any subset of \( \mathcal{M}_{k,d} \). So our lower bound for the subset \( \mathcal{M}_{k-\ell-1,d/3} \cap \mathcal{M}_{\ell,d/3} \cap \mathcal{M}_{1,d/3} \) is sufficient to obtain the desired lower bound. Finally, note that

\[
n = n_d n_\alpha n_k = \Omega\left(\frac{k \sqrt{d}}{\alpha^2}\right),
\]

as desired. This completes the proof.

Using the composition Theorem 3.3.6 in place of Theorem 3.3.3, we obtain a version of Theorem 3.4.7 in the language of generalized fingerprinting codes.
Theorem 3.4.8. For every constant $\ell \in \mathbb{N}$, every $k, d \in \mathbb{N}$, $\ell + 2 \leq k \leq d$, and every $\alpha \geq 1/d^{\ell/3}$, there is an $n = n(k, d, \alpha) = \tilde{\Omega}\left(\frac{k\sqrt{d}}{\alpha^2}\right)$ such that there exists a $(n, M_{k, d})$-generalized fingerprinting code with security $(1/2, o(1/n))$ for $(\alpha, 0)$-accuracy.
Chapter 4

Private Release and Learning of Threshold Functions

In this chapter, 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 \( n = 2^{O(\log^* |X|)} \) [8]. 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).

4.1 Results and Techniques

4.1.1 Private Query Release

Recall that the generic query release mechanisms discussed in Chapter 2 show that the sample complexity of releasing a worst-case family of queries \( Q \) is at most \( \tilde{O}(\min\{\text{VC}(Q) \cdot \log |X|, \log |Q| \cdot \sqrt{\log |X|}\}) \). However, for specific query families of interest, the sample complexity can be significantly
smaller. In particular, recall the family of point functions over $\mathcal{X}$, which is the family $\{q_x\}_{x \in \mathcal{X}}$ where $q_x(y)$ is 1 iff $y = x$, and the family of threshold functions over a totally ordered set $\mathcal{X}$, where $q_x(y)$ is 1 iff $y \leq x$. For point functions, Beimel, Nissim, and Stemmer [8] showed that the sample complexity has no dependence on $|\mathcal{X}|$ (or $|Q|$, since $|Q| = |\mathcal{X}|$ for these families). In the case of threshold functions, they showed that the dependence is at most $2^{O(\log^* |\mathcal{X}|)}$.

However, 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 $|\mathcal{X}|$ of the data universe?

We resolve these questions:

**Theorem 4.1.1.** The sample complexity of releasing threshold functions over a data universe $\mathcal{X}$ with differential privacy is at least $\Omega(\log^* |\mathcal{X}|)$. 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 [8] and improve the sample complexity to $2^{(1+o(1))\log^* |\mathcal{X}|}$ (from roughly $8^{\log^* |\mathcal{X}|}$). More importantly, our algorithm has an improved, nearly linear dependence on $1/\alpha$ in the case of vanishing error $\alpha \rightarrow 0$. Our algorithm is also computationally efficient, running in time nearly linear in the bit-representation of its input dataset. Closing the gap between the lower bound of $\approx \log^* |\mathcal{X}|$ and the upper bound of $\approx 2^{\log^* |\mathcal{X}|}$ remains an intriguing open problem.

We remark that in the case of pure differential privacy ($\delta = 0$), a sample complexity lower bound of $n = \Omega(\log |\mathcal{X}|)$ for releasing points and thresholds follows from a standard “packing argument” [5,78]. For point functions, this lower bound is matched by the Laplace mechanism. For threshold functions, an essentially matching upper bound was recently obtained [49], building on a construction of [48]. We note that these algorithms have a slightly better dependence on the accuracy parameter $\alpha$ 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 often fail badly for approximate differential privacy, for which much less is known.

We also recall the line of work characterizing the $\ell_2$-accuracy (rather than $\ell_\infty$-accuracy, which we consider in this work) achievable for query release in terms of other measures of the “complexity” of the family $Q$ (such as “hereditary discrepancy”) [10,78,93,95]. However, the characterizations given
in these works are again tight only up to factors of \( \text{poly}(\log |\mathcal{X}|, \log |Q|) \) and thus do not give good estimates of the sample complexity (which is at already at most \( O(\log |\mathcal{X}|) = O(\log |Q|) \) even for pure differential privacy, as mentioned above). In particular, the (partial) hereditary discrepancy of the class of threshold queries is only \( O(1) \), so these results do not give any meaningful lower bound for the task at hand.

### 4.1.2 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 \( \mathcal{X} \) up to small error in Kolmogorov (or CDF) distance: Given \( n \) i.i.d. samples \( x_i \sim \mathcal{D} \), the goal of a distribution learner is to produce a CDF \( F : \mathcal{X} \to [0, 1] \) such that \( |F(x) - F_D(x)| \leq \alpha \) for all \( x \in \mathcal{X} \), where \( \alpha \) 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. [37–39]).

The Dvoretzky-Kiefer-Wolfowitz inequality [43] gives a probabilistic bound on the Kolmogorov distance between a distribution and the empirical distribution of i.i.d. samples. It implies that without privacy, any distribution over \( \mathcal{X} \) can be learned to within arbitrarily small constant error via the empirical CDF of \( O(1) \) samples. On the other hand, we show that with approximate differential privacy, distribution learning instead requires sample complexity that grows with the size of the domain:

**Theorem 4.1.2.** *The sample complexity of learning arbitrary distributions on a totally ordered domain \( \mathcal{X} \) with differential privacy is at least \( \Omega(\log^* |\mathcal{X}|) \).*

We prove Theorem 4.1.2 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.

4.1.3 Private PAC Learning

As with query release and distribution learning, a natural problem is to characterize the sample
complexity of private PAC learning — the minimum number $n$ of samples necessary 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 (both proper and improper) PAC learning is proportional to the
Vapnik–Chervonenkis (VC) dimension of the class $C$ [15, 59, 119]. In the initial work on differentially
private learning, Kasiviswanathan et al. [81] showed that $O(\log |C|)$ labeled examples suffice to
privately learn any concept class $C$.\footnote{As with the query release discussion, we omit the dependency on all parameters except for $|C|$, $|X|$ and $\text{VC}(C)$.} 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 [81] 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 [5] 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 $\Omega(\log |X|) = \Omega(\log |C|)$. Chaudhuri and
Hsu [33] showed a similar result for the class $C$ of threshold functions, which also has VC dimension 1.
Specifically, they showed that when the domain is $X = [0, 1]$, the class of threshold functions cannot
be properly PAC learned with pure differential privacy. Feldman and Xiao [63] further showed that
this separation holds even for improper learning over finite domains — PAC learning thresholds 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 PAC learning.
Indeed, Beimel et al. [8] showed that point functions can be properly learned with approximate differential privacy using $O(1)$ samples (i.e. with no dependence on $|\mathcal{X}|$). For threshold functions, they exhibited a proper learner with sample complexity $2^{O(\log^* |\mathcal{X}|)}$, but it was conceivable that the sample complexity could also be reduced to $O(1)$. Moreover, Chaudhuri et al. [34] gave sample complexity upper bounds based on properties of the data distribution that do not depend explicitly on $|\mathcal{X}|$.

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

**Theorem 4.1.3.** The sample complexity of properly learning threshold functions over a data universe $\mathcal{X}$ with differential privacy is at least $\Omega(\log^* |\mathcal{X}|)$.

This lower bound extends to the concept class of $\ell$-dimensional thresholds. An $\ell$-dimensional threshold function, defined over the domain $\mathcal{X}^\ell$, is a conjunction of $\ell$ 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 4.1.4.** For every finite, totally ordered $\mathcal{X}$ and $\ell \in \mathbb{N}$, the sample complexity of properly learning the class $\mathcal{C}$ of $\ell$-dimensional threshold functions on $\mathcal{X}^\ell$ with differential privacy is at least $\Omega(\ell \cdot \log^* |\mathcal{X}|) = \Omega(\text{VC}(\mathcal{C}) \cdot \log^* |\mathcal{X}|)$.

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

**Open Problem 4.1.5** (Sample complexity of improper private learning). Exhibit a family of concept classes $\mathcal{C}$ with VC dimension $d$ (hence non-privately learnable with sample complexity $O(d)$) such that every $(\varepsilon, \delta)$-differentially private PAC learner for $\mathcal{C}$ requires sample complexity $\omega(d \log(1/\delta)/\varepsilon)$.

### 4.1.4 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 $\mathcal{A} : \mathcal{X}^n \to \mathcal{X}$ (for a totally ordered data universe $\mathcal{X}$) such that for every dataset $D \in \mathcal{X}^n$, with high probability we have $\min_i D_i \leq \mathcal{A}(D) \leq \max_i D_i$. We give reductions showing that the sample complexity of this problem is equivalent to the other ones we study:

**Theorem 4.1.6.** Over every totally ordered data universe $\mathcal{X}$, 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 4.1.7.** The sample complexity for solving the interior-point problem over a data universe $\mathcal{X}$ with differential privacy is $\Omega(\log^* |\mathcal{X}|)$.

We actually give three different proofs of Theorem 4.1.7. Our original proof, presented in Section 4.2.3, constructs for every $n$ a (correlated) distribution $\mathcal{D}$ over datasets of size $n \approx \log^* |\mathcal{X}|$ on which privately solving the interior-point problem is impossible. The construction is recursive: we use a hard distribution over datasets of size $(n - 1)$ on a data universe of size logarithmic in $|\mathcal{X}|$ to construct a hard distribution over datasets of size $n$ over $\mathcal{X}$. In Section 4.2.4, we offer an interpretation of this lower bound as a construction of a certain type of “interior-point fingerprinting code,” unifying it with the lower bounds for marginals presented in Chapter 3. Furthermore, inspired by the argument used to prove this lower bound, we give in Section 4.3 an algorithm for the interior-point problem, achieving sample complexity $2^{(1+o(1)) \log^* |\mathcal{X}|}$.

Subsequent to the conference publication of this work [27], Avi Wigderson pointed out the striking resemblance between the construction in Section 4.2.3 and Cole and Vishkin’s [36] distributed algorithm for 3-coloring a cycle. This is a classic problem in distributed computing in which $m$ processors are arranged in a cycle, wherein each processor can only communicate with its neighbors, and wish to come to an agreement as to a 3-coloring of the cycle. Using a similar “prefix matching”
argument to ours, Cole and Vishkin showed that the processors can come to a consensus after $O(\log^* m)$ rounds of communication.

The dependence on $\log^* m$ turns out to be essential to the distributed 3-coloring problem as well. In an influential work, Linial [88] gave a matching lower bound showing that $\Omega(\log^* m)$ rounds of communication are in fact necessary. His key insight was to strip away most of the details of the distributed computing model, and reduce the question of minimizing the number of rounds of communication to a clean combinatorial problem. Inspired by Linial’s approach (and the solution of this combinatorial problem which is credited to Awebuch), we discovered a new proof of Theorem 4.1.7 based on a reduction to a quantitative formulation of Ramsey’s Theorem [60, 102] for coloring hypergraphs. We present this new proof in Section 4.2.2, as it gives slightly improved parameters and we feel it gives complementary insight into the complexity of the interior point problem.

4.2 The Interior Point Problem – Lower Bounds

4.2.1 Definition

In this chapter 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 4.2.1.** An algorithm $A : \mathcal{X}^n \to \mathcal{X}$ solves the interior point problem on $\mathcal{X}$ with error probability $\beta$ if for every $D \in \mathcal{X}^n$,

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

where the probability is taken over the coins of $A$. The sample complexity of the algorithm $A$ is the dataset 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 (and should not) be a member of the dataset $D$. 

77
4.2.2 Lower Bound I: A Connection to Ramsey Theory

Ramsey’s Theorem

A hypergraph \((V, E)\) consists of a set of vertices, together with a set of hyperedges, where a hyperedge is a subset of vertices. An \(n\)-uniform hypergraph is a hypergraph where every hyperedge has size \(n\). Thus, a \(2\)-uniform hypergraph is a graph. The \((\mathcal{X}, n)\)-complete hypergraph has vertex set \(\mathcal{X}\) and hyperedge set \(\mathcal{X}^n\) consisting of all subsets of \(\mathcal{X}\) of size \(n\). (We stress that in this section, we use this notation to indicate subsets of size \(n\), and not ordered tuples.)

A hyperedge-coloring of a hypergraph using \(k\) colors is a mapping from its hyperedges to the set \([k]\). An \(m\)-hyperclique of an \(n\)-uniform hypergraph is a set of \(m\) vertices for which every subset of size \(n\) is a hyperedge. Under a hyperedge-coloring, a hyperclique is monochromatic if every hyperedge has the same color.

The qualitative form of Ramsey’s theorem asserts that any coloring of a sufficiently large complete hypergraph has a monochromatic hyperclique.

**Theorem 4.2.2.** For any natural numbers \(n, m, k\), there exists a number \(R(n, m, k) < \infty\) for which the following holds. Consider a hyperedge-coloring of the \((\mathcal{X}, n)\)-complete hypergraph using \(k\) colors. If \(|\mathcal{X}| \geq R(n, m, k)\), then this coloring induces a monochromatic \(m\)-hyperclique. That is, there exists a set of \(m\) vertices for which every \(n\)-hyperedge over this set has the same color.

We are also interested in a quantitative bound on \(R(n, m, k)\). First, we introduce the following notation for iterated exponentials.

\[
tower_b(x) = b^x, \quad \text{tower}_b(x_n, \ldots, x_1) = b^{x_n \text{tower}_b(x_{n-1}, \ldots, x_1)}.
\]

As somewhat more succinct notation, we also introduce:

\[
tower^{(0)}(x) = x, \quad \text{tower}^{(k)}(x) = 2^{\text{tower}^{(k-1)}(x)}.
\]

**Theorem 4.2.3 ([60]).**

\[
R(n, m, k) \leq \text{tower}_k(1, n - 1, n - 2, \ldots, 3, 2km - \log_k(m - n + 1) + O(k)).
\]

We will be primarily interested in estimating \(R(n, m, k)\) when \(m = \Theta(n)\) and \(k = n\). In this case, \(R(n, \Theta(n), n)\) is an exponential tower with height \(\Theta(n)\).
The Lower Bound

**Theorem 4.2.4.** Fix any constants $0 < \varepsilon, \beta < 1$. Let $\delta(n) \leq (1 - \beta)/(2(\varepsilon + 1)n)$. Then for every positive integer $n$, solving the interior point problem on $\mathcal{X}$ with probability at least $1 - \beta$ and with $(\varepsilon, \delta(n))$-differential privacy requires sample complexity $n \geq \Omega(|\mathcal{X}| \log^* n)$.

Theorem has a few minor technical advantages over the proof we will present in Section 4.2.3. Foremost is that we get a non-trivial lower bound even for $\delta$ as large as the maximal $\Omega(1/n)$ (whereas the construction in Section 4.2.3 requires $\delta(n) = (e^\varepsilon + 1)n$ to form a convergent series). However, we do not know how to use the ideas below to design an algorithm for solving the interior point problem (whereas the algorithm we present in Section 4.3 is inspired by the ideas used to prove the lower bound of Section 4.2.3).

**Proof.** The idea is the show that, given any differentially private $\mathcal{A} : \mathcal{X}^n \to \mathcal{X}$ for solving the interior point problem, one can extract a coloring of the $(\mathcal{X}, n)$-complete hypergraph with no large monochromatic hyperclique. Think of the vertices of the $(\mathcal{X}, n)$-complete hypergraph as points in $\mathcal{X}$, and the hyperedges as $n$-element datasets. Given a (randomized) algorithm $\mathcal{A} : \mathcal{X}^n \to \mathcal{X}$, we define a coloring $C_\mathcal{A} : \mathcal{X}^n \to [n]$ as follows. On a dataset $D = \{x_1, x_2, \ldots, x_n\} \subset \mathcal{X}$ with $x_1 < \ldots < x_n$, let $C_\mathcal{A}(D)$ be the index $i$ which maximizes $\Pr[\mathcal{A}(D) \in [x_i, x_{i+1})]$, breaking ties in favor of smaller $i$, and defining $x_{n+1} = +\infty$. That is, the color of a dataset $D$ (i.e. hyperedge) is the index for which the distribution of $\mathcal{A}(D)$ places maximal weight in its corresponding interval.

**Claim 4.2.5.** Let $\mathcal{A} : \mathcal{X}^n \to \mathcal{X}$ be an $(\varepsilon, \delta)$-differentially private algorithm for the interior point problem. Then for every set $S \subset \mathcal{X}$ of size $m' = n + r + 2$ with

$$r > \frac{\varepsilon^2 n}{1 - \beta - (\varepsilon + 1)\delta n} \quad (= \Theta(n)),$$

there exists a pair $D_0, D_1 \subset S$, each of size $n$, for which $C_\mathcal{A}(D_0) \neq C_\mathcal{A}(D_1)$.

We first see how the claim implies the interior point lower bound. Let $\mathcal{A} : \mathcal{X}^n \to \mathcal{X}$ be an $(\varepsilon, \delta)$-differentially private algorithm for the interior point problem. Consider the $(\mathcal{X}, n)$-complete hypergraph with hyperedge coloring given by $C_\mathcal{A}$. Then by Claim 4.2.5, this colored hypergraph has

---

2 Recall that we are using the notation $\mathcal{X}^n$ to denote subsets of $\mathcal{X}$ of size $n$. Thus, our lower bound applies even to algorithms that must only work when their input data points are distinct.
no monochromatic $m$-hyperclique. This implies $|X| < R(n, m', n)$. Applying Ramsey’s theorem and inverting this yields a sample complexity lower bound of $n \geq \Omega(\log^* |X'|)$.

Proof of Claim 4.2.5. The proof is a twist on a packing argument [5, 78]. Let $S \subset X$ be a set of size $m + 1 = n + r + 2$, and write the elements of $S$ as $x_0 < x_1 < \ldots < x_m$. Suppose for the sake of contradiction that every $n$-subset of $S$ is assigned the same color $i \in [n]$ under $C_A$. Let $D = \{x_0, x_1, \ldots, x_{i-1}, x_i, x_{i+r}, x_{i+r+1}, \ldots, x_m\}$, and consider the sequence of related datasets $D_1, \ldots, D_r$ defined by

$$D_j = \{x_1, \ldots, x_{i-1}, x_{(i-1)+j}, x_{(i-1)+j+1}, x_{i+r}, \ldots, x_{m-1}\}.$$  

That is, each $D_j$ is obtained from $D$ by moving items $x_0$ and $x_m$ to a consecutive pair of elements between $x_i$ and $x_{i+r}$. Since $A$ solves the interior point problem, we must have $\Pr[\min D_j \leq A(D_j) \leq \max D_j] \geq 1 - \beta$ for every $j$. By construction, the $i$th element of $D_j$ is $x_{(i-1)+j}$, so $\Pr[A(D_j) \in [x_{(i-1)+j}, x_{(i-1)+j+1}]] \geq (1 - \beta)/n$. Recalling that the Hamming distance between $D$ and each $D_j$ is two, differential privacy implies

$$\Pr[A(D) \in [x_{(i-1)+j}, x_{(i-1)+j+1}]] \geq e^{-2\varepsilon \left(1 - \frac{\beta}{n}\right)} - e^{-2\varepsilon} - e^{-\varepsilon} \delta > \frac{1}{r}.$$  

Since the sets $[x_{(i-1)+j}, x_{(i-1)+j+1}]$ for $j = 1, \ldots, r$ are disjoint, summing over the $j$’s yields a contradiction.

The condition $\varepsilon < 1$ is not essential to the proof of Theorem 4.2.2. In fact, one can take $\varepsilon$ to itself be a tower of height $O(n)$, and recover essentially the same lower bound. This holds because taking $m$ to be a tower of height $O(n)$ in Theorem 4.2.3 gives an upper bound on $R(n, m, n)$ that is still a tower of height $O(n)$.

**Proposition 4.2.6.** Fix $0 < \beta < 1$, and let $\varepsilon(n) \leq \text{tower}^{(n)}(1)$. Let $\delta(n) \leq (1 - \beta)/2(\varepsilon^2 + 1)n$. Then solving the interior point problem on $X$ with probability at least $1 - \beta$ and with $(\varepsilon(n), \delta(n))$-differential privacy requires sample complexity $n \geq \Omega(\log^* |X|)$.

---

3If $i = n$, we do the same construction as if $i = n - 1$.  

---

80
An important consequence of Proposition 4.2.6 is that it allows us to apply the reduction of Lemma 2.3.6 to incorporate a dependence on $\delta$ in the interior point lower bound. (While Lemma 2.3.6 is stated for the query release problem, it is easy to see that it also applies to the interior point problem.)

**Theorem 4.2.7.** Let $\exp(-n/10 \log^* (n)) < \delta < 1/15n^{1.01}$. Then solving the interior point problem over $\mathcal{X}$ with $(1, \delta)$-differential privacy requires $n \geq \Omega(\log^* |\mathcal{X}| \cdot \log(1/\delta))$.

The exact parameters of the upper bound on $\delta$ are not important — we could just as well take $\delta < O(1/n^{1+\eta})$ for any $\eta > 0$.

**Proof.** Our goal will be to show that for sufficiently large $n$ and $\exp(-n/10 \log^* (n)) < \delta < 1/15n^{1.01}$, that solving the interior point problem over $\mathcal{X}$ requires

$$|\mathcal{X}| \leq \text{tower}^{O(n/\log(1/\delta))}(1).$$

Rearranging implies the claim.

To this end, fix $n \in \mathbb{N}$ and $\exp(-n/10 \log^* (n)) < \delta < 1/15n^{1.01}$. Let $m = 2n/\ln(1/\delta)$. Let $\varepsilon = \ln(1/6m\delta^{0.995} - 1)$ and $\delta_0 = 1/6(\varepsilon + 1)m$. (These parameters are chosen so that $\delta_0/\delta = 1/\delta^{0.005}$.) Observe that since $\delta < 1/15n^{1.01}$, we have $\varepsilon > 0$. Moreover, since $\delta > \exp(-n/10 \log^* (n))$, we also have $\varepsilon = 1/6m\delta^{0.995} \leq \text{tower}^{(m)}(1)$. Then Proposition 4.2.6 shows that solving the interior point problem over $\mathcal{X}$ on a database of size $m$ with $(\varepsilon, \delta_0)$-differential privacy requires $|\mathcal{X}| \leq \text{tower}^{O(m)}(1)$. Thus Lemma 2.3.6 shows that solving the interior point problem on $\mathcal{X}$ on a database of size $n$ with $(1, \delta)$-differential privacy requires $|\mathcal{X}| \leq \text{tower}^{O(n/\log(\delta_0/\delta))}(1) = \text{tower}^{O(n/\log(1/\delta))}(1).$}

### 4.2.3 Lower Bound II: Prefix Matching

We now give a different version of the interior point lower bound.

**Theorem 4.2.8.** Fix any constant $0 < \varepsilon < 1/4$. Let $\delta(n) \leq 1/(50n^2)$. Then for every positive integer $n$, solving the interior point problem on $\mathcal{X}$ with probability at least $3/4$ and with $(\varepsilon, \delta(n))$-differential privacy requires sample complexity $n \geq \Omega(\log^* |\mathcal{X}|)$.

Our choice of $\delta = O(1/n^2)$ is unimportant; any monotonically non-increasing convergent series will do, though we cannot get all the way to the optimal $\delta = \Omega(1/n)$ as in the proof of Theorem 4.2.2.
To prove the theorem, we inductively construct a sequence of dataset distributions \( \{D_n\} \) supported on data universes \( [S(n)] \) (for \( S(n + 1) = 2^{O(S(n))} \)) over which any differentially private mechanism using \( n \) samples must fail to solve the interior point problem. Given a hard distribution \( D_n \) over \( n \) elements \((x_1, x_2, \ldots, x_n)\) from \([S(n)]\), we construct a hard distribution \( D_{n+1} \) over elements \((y_0, y_1, \ldots, y_n)\) 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 \((y_0, \ldots, y_n)\), 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 \( D_{n+1} \) can be used to construct a private mechanism for \( D_n \), and so \( D_{n+1} \) 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

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

for every \( n \). In particular, it suffices that

\[
\sum_{n=1}^{\infty} \delta(n) \leq \frac{1}{3} - \frac{\beta}{e^{\varepsilon} + 1}.
\]

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 4.2.9.** For every positive integer \( n \), there exists a distribution \( D_n \) over datasets \( D \in [S(n)]^n = \{0, 1, \ldots, S(n) - 1\}^n \) such that for every \((\varepsilon, \delta(n))\)-differentially private mechanism \( A \),

\[
\Pr[\min D \leq A(D) \leq \max D] \leq P_n,
\]

where the probability is taken over \( D \leftarrow \kappa D_n \) and the coins of \( A \).

In this section, we give a direct proof of the lemma and in Section 4.2.4, we show how the lemma follows from the construction of a new combinatorial object we call an “interior point fingerprinting code.”

**Proof.** The proof is by induction on \( n \). We first argue that the claim holds for \( n = 1 \) by letting \( D_1 \)
be uniform over the singleton datasets (0) and (1). To that end let $x \leftarrow \mathcal{D}_1$ and note that for any $(\varepsilon, \delta(1))$-differentially private mechanism $\mathcal{A}_0 : \{0, 1\} \rightarrow \{0, 1\}$ it holds that
\[
\Pr[\mathcal{A}_0(x) = x] \leq e^\varepsilon \Pr[\mathcal{A}_0(x) = x] + \delta(1) = e^\varepsilon(1 - \Pr[\mathcal{A}_0(x) = x]) + \delta(1),
\]
giving the desired bound on $\Pr[\mathcal{A}_0(x) = x]$.

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

- Sample $(x_1, \ldots, x_n) \leftarrow \mathcal{D}_n$.
- Sample a uniformly random $y_0 \leftarrow [S(n + 1)]$. 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 $\hat{\mathcal{A}}$ that could solve the interior point problem on $\mathcal{D}_{n+1}$ with probability greater than $P_{n+1}$. We use $\hat{\mathcal{A}}$ to construct the following private mechanism $\mathcal{A}$ for solving the interior point problem on $\mathcal{D}_n$, giving the desired contradiction:

\begin{algorithm}
\textbf{Algorithm 1} $\mathcal{A}(D)$
\textbf{Input:} dataset $D = (x_1, \ldots, x_n) \in [S(n)]^n$

1. Construct $\hat{D} = (y_0, \ldots, y_n)$ by sampling from $\mathcal{D}_{n+1}$, but starting with the dataset $D$. 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 $y \leftarrow \mathcal{A}(\hat{D})$.
3. Return the length of the longest prefix of $y$ (in base $b(n)$ notation) that agrees with $y_0$.
\end{algorithm}

The mechanism $\mathcal{A}$ is also $(\varepsilon, \delta(n+1))$-differentially private, since for all pairs of neighboring datasets $D \sim D'$ and every $T \subseteq [S(n)]$,}
\[ \Pr[\mathcal{A}(D) \in T] = \mathbb{E}_{y_0 \sim_R [S(n+1)]} \Pr[\hat{\mathcal{A}}(\hat{D}) \in \hat{T} \mid y_0] \]
\[ \leq \mathbb{E}_{y_0 \sim_R [S(n+1)]} (e^\varepsilon \Pr[\hat{\mathcal{A}}(\hat{D}') \in \hat{T} \mid y_0] + \delta(n + 1)) \quad \text{since } \hat{D} \sim \hat{D}' \text{ for fixed } y_0 \]
\[ = e^\varepsilon \Pr[\mathcal{A}(D') \in T] + \delta(n + 1), \]

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 \( \mathcal{A} \) solves the interior point problem on \( D_n \) with probability greater than \( P_n \).

First we show that \( x \min_{D} y \) 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 \( \mathcal{A} \) 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 \( x \min_{D} \).

Now we use the privacy that \( \hat{\mathcal{A}} \) provides to \( y_0 \) to show that \( x \max_{D} y \) except with probability at most \( e^\varepsilon / b(n) + \delta(n + 1) \). Fix a dataset \( D \). Let \( w = \max_{D} \) and fix all the randomness of \( \mathcal{A} \) 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 dataset \( \hat{D} \) with \( y_{0}^{(w+1)} \) set to \( z \) and everything else fixed as above. Then by the differential privacy of \( \hat{\mathcal{A}} \),

\[ \Pr_{z \in [b(n)]} [\hat{\mathcal{A}}(\hat{D}_z)^{w+1} = z] = \frac{1}{b(n)} \sum_{z \in [b(n)]} \Pr[\hat{\mathcal{A}}(\hat{D}_z)^{w+1} = z] \]
\[ \leq \frac{1}{b(n)} \sum_{z \in [b(n)]} \mathbb{E}_{z' \sim_R [b(n)]} \left[ e^\varepsilon \Pr[\hat{\mathcal{A}}(\hat{D}_z')^{w+1} = z] + \delta(n + 1) \right] \]
\[ \leq \frac{e^\varepsilon}{b(n)} + \delta(n + 1), \]

where all probabilities are also taken over the coins of \( \hat{\mathcal{A}} \). 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 4.2.8 by estimating the $S(n)$ guaranteed by Lemma 4.2.9.

**Proof of Theorem 4.2.8.** Let $S(n)$ be as in Lemma 4.2.9. We recall the notation for iterated exponentials:

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

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

\[
M^{tower^{(k)}(x)} = 2^{tower^{(k)}(x) \log M} = tower^{(2)}(tower^{(k-1)}(x) + \log \log M)
\leq tower^{(2)}(tower^{(k-1)}(x + \log \log M)) = tower^{(k+1)}(x + \log \log M).
\]

By induction on $n$ we get an upper bound of

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

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

\[
n \geq \log^* S(n) - \log^*(cn^2)
\geq \log^* S(n) - O(\log^* \log^* S(n))
= \log^* |\mathcal{X}| - O(\log^* \log^* |\mathcal{X}|).
\]

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

### 4.2.4 Lower Bound III: Interior Point Fingerprinting Codes

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 $\mathcal{X}$, 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 4.2.10.** For a totally ordered domain $\mathcal{X}$, an *interior point fingerprinting code* over $\mathcal{X}$ consists of a pair of randomized algorithms $(\text{Gen}, \text{Trace})$ with the following syntax.

- $\text{Gen}_n$ samples a codebook $C = (x_1, \ldots, x_n) \in \mathcal{X}^n$
- $\text{Trace}_n(x)$ takes as input a “codeword” $x \in \mathcal{X}$ and outputs either a user $i \in [n]$ or a failure symbol $\bot$.

The algorithms $\text{Gen}$ and $\text{Trace}$ 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 $\mathcal{A} : \mathcal{X}^{|T|} \rightarrow \mathcal{X}$. The algorithm $\mathcal{A}$ is given $C|_T$, i.e., the codewords $x_i$ for $i \in T$, and its output $x \leftarrow_r \mathcal{A}(C|_T)$ is said to be “feasible” if $x \in \left[\min_{i \in T} x_i, \max_{i \in T} x_i\right]$. 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:** $\Pr[\text{Trace}(x) = \bot \land x \text{ feasible}] \leq \gamma$, where $\gamma \in [0, 1]$ is the completeness error.

2. **Soundness:** $\Pr[\text{Trace}(x) \in [n] \setminus T] \leq \xi$, where $\xi \in [0, 1]$ is the soundness error.

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

**Remark 4.2.11.** 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 $|\mathcal{X}|$ of the form $000111111$. 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 each fingerprinted copy by simply splicing two versions of the movie at a single point.

We now argue as in Chapter 3 that the existence of an interior point fingerprinting code yields a lower bound for privately solving the interior point problem.

**Lemma 4.2.12.** 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 $\mathcal{X}$ for $n$ users with completeness error $\gamma$ and soundness error $\xi$, then there is no $(\varepsilon, \delta)$-differentially private algorithm that, with probability at least $2/3$, solves the interior point problem on $\mathcal{X}$ for datasets of size $n - 1$. 

86
Proof. Suppose for the sake of contradiction that there were a differentially private $A$ for solving the interior point problem on $X^{n-1}$. Let $T = [n - 1]$, and let $x = A(C|_T)$ for a codebook $C \xleftarrow{\text{r}} \text{Gen}$. 

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

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

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

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

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

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

We now show how to construct an interior point fingerprinting code, using similar ideas as in the proof of Lemma 4.2.9. 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 4.2.13.** 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 $\xi$ on a domain $X_n$ of size $|X_n| \leq \text{tower}^{(n+\log^* (2n^2/\xi))}(1)$.

**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 $(\text{Gen}_n, \text{Trace}_n)$ for $n$ users. Then we define

- $\text{Gen}_{n+1}$ samples $C' = (x'_1, \ldots, x'_n) \xleftarrow{\text{r}} \text{Gen}_n$ and $x_{n+1} \xleftarrow{\text{r}} [S(n+1)]$. For each $i = 1, \ldots, n$, let $x_i$ be a base-$b(n)$ number (written $x_i^{(0)}x_i^{(1)} \ldots x_i^{(S(n)-1)}$, 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 $C = (x_1, \ldots, x_{n+1})$.

- $\text{Trace}_{n+1}(x)$ retrieves the codebook $C$ from its shared state with $\text{Gen}_{n+1}$. Let $M$ be the maximum number of digits to which any $x_i$ (for $i = 1, \ldots, n$) 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 \( \text{Trace}_n(x') \) with respect to codebook \( C' = (x'_1, \ldots, x'_n) \).

We reduce the security of this scheme to that of \((\text{Gen}_n, \text{Trace}_n)\). 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 \( \text{Gen}_{n+1} \) 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,

\[
\Pr[\text{Trace}_{n+1}(x) = \bot \land x \text{ feasible for } C|_T] = \Pr[x^{M+1} \neq x_{n+1}^{M+1} \land \text{Trace}_n(x') = \bot \land x' \text{ feasible for } C'|_{T'}] \\
\leq \Pr[\text{Trace}_n(x') = \bot \land x' \text{ feasible for } C'|_{T'}] = 0,
\]

by induction, proving perfect completeness.

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

\[
\Pr[\text{Trace}_{n+1}(x) \in [n+1] \setminus T] \leq \Pr[\text{Trace}_{n+1}(x) = n + 1 \land (n + 1) \notin T] + \Pr[\text{Trace}_{n+1}(x) \in [n] \setminus T] \\
\leq \Pr[x^{M'+1} = x_{n+1}^{M'+1} \land (n + 1) \notin T] + \Pr[\text{Trace}_n(x') \in [n] \setminus T] \\
\leq \frac{1}{b(n)} + \sum_{j=1}^{n-1} \frac{1}{b(j)} = \sum_{j=1}^{n} \frac{1}{b(j)} < \xi.
\]

Combining Lemmas 4.2.12 and 4.2.13 yields Theorem 4.1.7.

### 4.3 The Interior Point Problem – Upper Bound

We now present a recursive algorithm, \( \text{RecPrefix} \), for privately solving the interior point problem.

**Theorem 4.3.1.** Let \( \beta, \epsilon, \delta > 0 \), let \( \mathcal{X} \) be a finite, totally ordered domain, and let \( n \in \mathbb{N} \) with

\[
n \geq \frac{18500}{\epsilon} \cdot 2^{|\mathcal{X}|} \cdot \log^* (|\mathcal{X}|) \cdot \ln\left(\frac{\log^* (|\mathcal{X}|)}{\beta \delta}\right).
\]

If \( \text{RecPrefix} \) (defined below) is executed on a dataset \( S \in \mathcal{X}^n \) with parameters \( \frac{\beta}{3 \log^* |\mathcal{X}|}, \frac{\epsilon}{2 \log^* |\mathcal{X}|}, \frac{\delta}{2 \log^* |\mathcal{X}|} \), then

1. \( \text{RecPrefix} \) is \((\epsilon, \delta)\)-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, \( \text{RecPrefix} \) takes an input dataset \( S \) over \( \mathcal{X} \) and constructs a dataset \( S' \) over a smaller universe \( \mathcal{X}' \), where \( |\mathcal{X}'| = \log |\mathcal{X}| \), 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 4.2.3. We remark that \( \text{RecPrefix} \) is computationally efficient, running in time \( O(n \cdot \log |\mathcal{X}|) \), which is linear in the bit-representation of the input dataset.

### 4.3.1 The Choosing Mechanism

Before formally presenting the algorithm \( \text{RecPrefix} \), we describe an \((\varepsilon, \delta)\)-differentially private variant of the exponential mechanism called the \textit{choosing mechanism}, introduced in [8].

A quality function \( q : \mathcal{X}^* \times \mathcal{F} \rightarrow \mathbb{R} \) with sensitivity at most 1 is of \( k \)-\textit{bounded-growth} if \textit{adding} an element to a dataset 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 \textit{small set} \( G(S) \) of good solutions instead of \( \mathcal{F} \).

#### Algorithm 2 Choosing Mechanism

**Input:** dataset \( S \), quality function \( q \), solution set \( \mathcal{F} \), and parameters \( \beta, \epsilon, \delta \) and \( k \).

1. Set \( \widehat{\text{OPT}} = \text{OPT} + \text{Lap}(\frac{4}{k}) \). If \( \widehat{\text{OPT}} < \frac{\epsilon}{\delta} \ln(\frac{4k}{3\delta}) \) 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 \( \frac{2}{\delta} \).

The following lemmas give the privacy and utility guarantees of the choosing mechanism. We give a slightly improved utility result over [8], and the analysis is presented below.
Lemma 4.3.2. Fix $\delta > 0$, and $0 < \epsilon \leq 2$. If $q$ is a $k$-bounded-growth quality function, then the choosing mechanism is $(\epsilon, \delta)$-differentially private.

Lemma 4.3.3. Let the choosing mechanism be executed on a $k$-bounded-growth quality function, and on a dataset $S$ s.t. there exists a solution $\hat{f}$ with quality $q(S, \hat{f}) \geq \frac{16}{\epsilon} \ln\left(\frac{4k}{\beta \epsilon \delta}\right)$. With probability at least $(1 - \beta)$, the choosing mechanism outputs a solution $f$ with quality $q(S, f) \geq 1$.

Lemma 4.3.4. Let the choosing mechanism be executed on a $k$-bounded-growth quality function, and on a dataset $S$ containing $m$ elements. With probability at least $(1 - \delta)$, the choosing mechanism outputs a solution $f$ with quality $q(S, f) \geq \text{OPT} - \frac{16}{\epsilon} \ln\left(\frac{4km}{\beta \epsilon \delta}\right)$.

Proof of Lemma 4.3.2. Let $A$ denote the choosing mechanism (Algorithm 2). Let $S, S'$ be neighboring datasets of $m$ elements. We need to show that

$$\Pr[A(S) = \bot] = \Pr\left[\hat{\text{OPT}}(S) < \frac{8}{\epsilon} \ln\left(\frac{4k}{\beta \epsilon \delta}\right)\right]$$

$$\leq \exp\left(\frac{\epsilon}{4}\right) \cdot \Pr\left[\hat{\text{OPT}}(S') < \frac{8}{\epsilon} \ln\left(\frac{4k}{\beta \epsilon \delta}\right)\right]$$

$$= \exp\left(\frac{\epsilon}{4}\right) \cdot \Pr[A(S') = \bot]. \tag{4.1}$$

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

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

$$\Pr[A(S) \in R] \leq \Pr[A(S) \neq \bot]$$

$$\leq \Pr\left[\text{Lap}\left(\frac{4}{\epsilon}\right) > \frac{4}{\epsilon} \ln\left(\frac{4k}{\beta \epsilon \delta}\right)\right]$$

$$\leq \delta \leq \Pr[A(S') \in R] + \delta.$$

Case (b): $\text{OPT}(S) \geq \frac{4}{\epsilon} \ln\left(\frac{4k}{\beta \epsilon \delta}\right)$. 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 \( B = G(S) \setminus G(S') \), 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

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

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 \( q(S, \hat{f}) \geq \frac{4}{e} \ln(\frac{4e}{\beta \epsilon \delta}) \), we have that

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

which is at most \( \delta/k \) 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

\[
\frac{\Pr[A(S) = f]}{\Pr[A(S') = f]} = \left( \frac{\Pr[A(S) \neq \bot]}{\Pr[A(S') \neq \bot]} \cdot \frac{\exp(\frac{\epsilon}{4} q(f, S))}{\sum_{h \in G(S)} \exp(\frac{\epsilon}{4} q(h, S))} \right) / \left( \frac{\Pr[A(S') \neq \bot]}{\sum_{h \in G(S')} \exp(\frac{\epsilon}{4} q(h, S'))} \cdot \exp(\frac{\epsilon}{4} q(f, S')) \right) \\
= \frac{\Pr[A(S) \neq \bot]}{\Pr[A(S') \neq \bot]} \cdot \frac{\exp(\frac{\epsilon}{4} q(f, S))}{\exp(\frac{\epsilon}{4} q(f, S'))} \cdot \frac{\sum_{h \in G(S')} \exp(\frac{\epsilon}{4} q(h, S'))}{\sum_{h \in G(S)} \exp(\frac{\epsilon}{4} q(h, S))} \leq e^\frac{\epsilon}{4} \cdot e^{\frac{\epsilon}{4}} \cdot e^{\frac{\epsilon}{4}} = e^\epsilon.
\]
We now prove Fact 3. Let $X \triangleq \sum_{h \in G(S)} \exp(\frac{\epsilon}{4} q(S, h))$. Since there exists a solution $\hat{f}$ s.t. $q(S, \hat{f}) \geq \frac{4}{\epsilon} \ln(\frac{4k}{\beta \delta})$), we have $X \geq \exp(\frac{\epsilon}{4} \cdot \frac{4}{\epsilon} \ln(\frac{4k}{\beta \delta})) \geq 4k$.

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,

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

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

The utility analysis for the choosing mechanism is rather straightforward:

**Proof of Lemma 4.3.3.** Recall that the mechanism defines $\widehat{OPT}(S)$ as $\text{OPT}(S) + \text{Lap}(\frac{4}{\epsilon})$. Note that the mechanism succeeds whenever $\widehat{OPT}(S) \geq \frac{8}{\epsilon} \ln(\frac{4k}{\beta \delta})$. This happens provided the Lap($\frac{4}{\epsilon}$) random variable is at most $\frac{8}{\epsilon} \ln(\frac{4k}{\beta \delta})$, which happens with probability at least $(1 - \beta)$. \square

**Proof of Lemma 4.3.4.** Note that if $\text{OPT}(S) < \frac{16}{\epsilon} \ln(\frac{4km}{\beta \delta})$, then every solution is a good output, and the mechanism cannot fail. Assume, therefore, that there exists a solution $f$ s.t. $q(f, S) \geq \frac{16}{\epsilon} \ln(\frac{4km}{\beta \delta})$, and recall that the mechanism defines $\widehat{OPT}(S)$ as $\text{OPT}(S) + \text{Lap}(\frac{4}{\epsilon})$. As in the proof of Lemma 4.3.3, with probability at least $1 - \beta/2$, we have $\widehat{OPT}(S) \geq \frac{8}{\epsilon} \ln\left(\frac{4k}{\beta \delta}\right)$. Assuming this event occurs, we will show that with probability at least $1 - \beta/2$, the exponential mechanism chooses a solution $f$ s.t. $q(S, f) \geq \text{OPT}(S) - \frac{16}{\epsilon} \ln(\frac{4km}{\beta \delta})$.

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

$$\left(1 - km \cdot \exp\left(-\frac{\epsilon}{4} \cdot \frac{16}{\epsilon} \ln\left(\frac{4km}{\beta \delta}\right)\right)\right) \geq \left(1 - \frac{\beta}{2}\right).$$
By a union bound, we get that the choosing mechanism outputs a good solution with probability at least \( (1 - \beta) \).

### 4.3.2 The \textit{RecPrefix} algorithm

We are now ready to present and analyze the algorithm \textit{RecPrefix}.

**Algorithm 3 \textit{RecPrefix}**

**Input:** dataset \( S = (x_j)_{j=1}^n \in \mathcal{X}^n \), parameters \( \beta, \epsilon, \delta \).

1. If \( |\mathcal{X}| \leq 32 \), 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 \( x \in \mathcal{X} \).

2. Let \( k = \lceil \frac{386}{\epsilon} \ln(\frac{4}{3\delta}) \rceil \), and let \( Y = (y_1, y_2, \ldots, y_{n-2k}) \) 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 \textit{RecPrefix} recursively on \( S' = (z_j)_{j=1}^{(n-2k)/2} \in (\mathcal{X}')^{(n-2k)/2} \) with parameters \( \beta, \epsilon, \delta \).

   Recall that \( |\mathcal{X}'| = \log |\mathcal{X}| \). 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 \( q : \mathcal{X}^* \times \mathcal{X}^{z+1} \to \mathbb{N} \), where \( q(S, I) \) is the number of agreements on \( I \) among \( x_1, \ldots, x_n \).

6. For \( \sigma \in \{0, 1\} \), define \( L_\sigma \in \mathcal{X} \) to be the prefix \( L \) followed by \( (\log |\mathcal{X}| - z - 1) \) appearances of \( \sigma \).

7. Compute \( \hat{\text{big}} = \text{Lap}(\frac{1}{2}) + \# \{ j : x_j \geq L_1 \} \).

8. If \( \hat{\text{big}} \geq \frac{3k}{\epsilon} \) then return \( L_1 \). Otherwise return \( L_0 \).

We start the analysis of \textit{RecPrefix} with the following two simple observations.

**Observation 4.3.5.** There are at most \( \log^* |\mathcal{X}| \) recursive calls throughout the execution of \textit{RecPrefix} on a dataset \( S \in \mathcal{X}^* \).
Observation 4.3.6. Let $\text{RecPrefix}$ be executed on a dataset $S \in \mathcal{X}^n$, where $n \geq 2^{\log^* |\mathcal{X}|} \cdot \frac{312}{\epsilon} \cdot \ln(\frac{4}{\beta \delta \epsilon})$. Every recursive call throughout the execution operates on a dataset containing at least $\frac{1540}{\epsilon} \cdot \ln(\frac{4}{\beta \delta \epsilon})$ elements.

Proof. This follows from Observation 4.3.5 and from the fact that the $i^{th}$ recursive call is executed on a dataset of size $n_i = \frac{n}{2^i} - k \sum_{\ell=0}^{i-2}(\frac{1}{2})^\ell \geq \frac{n}{2^i} - 2k$. \qed

We now analyze the utility guarantees of $\text{RecPrefix}$ by proving the following lemma.

Lemma 4.3.7. Let $\beta, \epsilon, \delta, \epsilon, S \in \mathcal{X}^n$ be inputs on which $\text{RecPrefix}$ performs at most $N$ recursive calls, all of which are on datasets of at least $\frac{1540}{\epsilon} \cdot \ln(\frac{4}{\beta \delta \epsilon})$ 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. $|\{i : x_i \geq x\}| \geq k \triangleq \left\lfloor \frac{386}{\epsilon} \cdot \ln(\frac{4}{\beta \delta \epsilon}) \right\rfloor$.

Before proving the lemma, we make a combinatorial observation that motivates the random shuffling in Step 2 of $\text{RecPrefix}$. A pair of elements $y, y' \in S$ is useful in Algorithm $\text{RecPrefix}$ 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 4.3.8. Let $(\Pi_1, \Pi_2, \ldots, \Pi_n)$ be a random permutation of $(1, 2, \ldots, n)$. 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 $(\Pi_{2i-1}, \Pi_{2i})$ within distance $\frac{r}{12}$. For each $i$, we call $\Pi_{2i-1}$ the left side of the pair, and $\Pi_{2i}$ 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 $\binom{\frac{n}{6}}{r} r^r$ 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

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

94
where we have used Stirling’s approximation for the first inequality.

Suppose we have paired random elements in our input dataset \( S \), and constructed a dataset \( S' \) 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 4.3.8 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 4.3.7. 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 \( \beta \). By Proposition 2.1.4, it suffices to have
\[
n \geq \frac{1540}{\epsilon} \cdot \ln\left(\frac{4}{3\epsilon\beta}\right),
\]
since \( 32 \exp(-\epsilon(n/2 - k)/2) \leq \beta \).

Assume that the stated lemma holds whenever \( RecPrefix \) performs at most \( t - 1 \) recursive calls. Let \( \beta, \epsilon, \delta \) 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 datasets containing at least \( \frac{1540}{\epsilon} \cdot \ln\left(\frac{4}{3\epsilon\beta}\right) \) elements. Consider the first call in the execution on those inputs, and let \( y_1, \ldots, y_{n-2k} \) be the random permutation chosen on Step 2. We say that a pair \( y_{2j-1}, y_{2j} \) is close if
\[
\left\{ \begin{array}{l}
y_{2j-1} \leq y_i \leq y_{2j} \\
i : \text{ or}\\y_{2j} \leq y_i \leq y_{2j-1}
\end{array} \right\} \leq \frac{k-1}{12}.
\]

By Claim 4.3.8, 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 dataset 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) \( \exists z_i \in S' \text{ s.t. } z_i \leq z \); 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
Lemma 4.3.9. When executed for $N$ recursive calls, $\text{RecPrefix}$ is $(2\epsilon N, 2\delta N)$-differentially private.

Proof. The proof is by induction on the number of recursive calls, denoted by $t$. If $t = 1$ (i.e., $|X| \leq 32$), then by Proposition 2.1.4 the exponential mechanism ensures that $\text{RecPrefix}$ is $(\epsilon, 0)$-differentially private. Assume that the stated lemma holds whenever $\text{RecPrefix}$ performs at most $t - 1$ recursive calls, and let $S_1, S_2 \in X^*$ be two neighboring datasets on which $\text{RecPrefix}$ performs $t$ recursive calls.\footnote{The recursion depth is determined by $|X|$, which is identical in $S_1$ and in $S_2$.} Let $B$ denote an algorithm consisting of steps 1-4 of $\text{RecPrefix}$ (the output of $B$ is the value $z$ from Step 4). Consider the executions of $B$ 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 $S_2$. If both $L_0$ and $L_1$ are good outputs, then Step 8 cannot fail. We have already established the existence of $L_0 \leq y_\ell \leq L_1$. 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 $\hat{\text{big}} \geq 3k/2$ and Step 8 failing is at most $\exp(-\frac{k}{2}) \leq \beta$. 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, $\min\{y_{2j-1}, y_{2j}\} > L_0$, and, hence, $\max\{y_{2j-1}, y_{2j}\} > L_1$. Recall that there are at least $2k$ elements in $S$ which are bigger than $\max\{y_{2j-1}, y_{2j}\}$. As $k \geq \frac{2}{\epsilon} \ln(\frac{1}{\delta})$, the probability that $\hat{\text{big}} < 3k/2$ and $\text{RecPrefix}$ fails to return $L_1$ in this case is at most $\beta$.

All in all, $\text{RecPrefix}$ fails to return an appropriate $x$ with probability at most $3\beta t$. \qed

We now proceed with the privacy analysis.

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

It remains to analyze the case where $L_0$ is not a good output, and let $z_j \in S$ be s.t. $z_j > L_0$. Hence, as $L$ is of length $z + 1$, we have that either $\min\{y_{2j-1}, y_{2j}\} < L_0$ or $\max\{y_{2j-1}, y_{2j}\} > L_1$. If $\min\{y_{2j-1}, y_{2j}\} < L_0$, then $L_0$ satisfies Condition 1 of being a good output. It also satisfies Condition 2 because $y_\ell \geq L_0$ and $y_\ell \in Y$, which we took to be the smallest $n-2k$ elements of $S$. Similarly, $L_1$ is a good output if $\max\{y_{2j-1}, y_{2j}\} > L_1$. In any case, at least one out of $L_0, L_1$ is a good output.

We now proceed with the privacy analysis.
We show that the distributions on the datasets $S_1'$ and $S_2'$ are similar in the sense that for each dataset in one of the distributions there exist a neighboring dataset 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 $\Pi$ and $\hat{\Pi}$ of the smallest $(n - 2k)$ elements of $S_1$ and of $S_2$ respectively s.t. $Y_1$ and $Y_2$ are neighboring datasets. 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 datasets, the same is true for $S_0'$ and $S_2'$. Hence, for every set of outputs $F$ it holds that

\[
\Pr[B(S) \in F] = \sum_{\Pi} \Pr[\Pi] \cdot \Pr[RecPrefix(S_1') \in F|\Pi] 
\leq e^{2\epsilon(t-1)} \cdot \sum_{\Pi} \Pr[\Pi] \cdot \Pr[RecPrefix(S_2') \in F|\hat{\Pi}] + 2\delta(t - 1) 
= e^{2\epsilon(t-1)} \cdot \sum_{\hat{\Pi}} \Pr[\hat{\Pi}] \cdot \Pr[RecPrefix(S_2') \in F|\hat{\Pi}] + 2\delta(t - 1) 
= e^{2\epsilon(t-1)} \cdot \Pr[B(S') \in F] + 2\delta(t - 1)
\]

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 dataset through the choosing mechanism and using the Laplace mechanism, each of which is $(\epsilon, \delta)$-differentially private. By composition (Lemma 2.1.5), we get that $RecPrefix$ is $(2t\epsilon, 2t\delta)$-differentially private.

Combining Lemma 4.3.7 and Lemma 4.3.9 we obtain Theorem 4.3.1.

**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 4.3.1), and our $\log^* |X|$ lower bound (Theorem 4.2.2).
**Open Problem 4.3.10** (Sample complexity of the interior point problem). Resolve the sample complexity of the interior point problem on data universe $\mathcal{X}$. In particular, is the true sample complexity closer to $\log^* |\mathcal{X}|$, or to $2^{\log^* |\mathcal{X}|}$?

Below we describe an idea for reducing the upper bound to $\text{poly}(\log^* |\mathcal{X}|)$.

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 dataset size grows by 1. This resulted in the $\Omega(\log^* |\mathcal{X}|)$ lower bound.

In $\text{RecPrefix}$ on the other hand, every level of recursion shrank the dataset size by a factor of $\frac{1}{2}$, and hence, we required a sample of (roughly) $2^{\log^* |\mathcal{X}|}$ 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 dataset size shrinks.

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

<table>
<thead>
<tr>
<th><strong>Input:</strong> $(x_1, \ldots, x_n) \in \mathcal{X}^n$.</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Let $(y^0_1, \ldots, y^0_n)$ denote a random permutation of $(x_1, \ldots, x_n)$.</td>
</tr>
<tr>
<td>2. For $t = 1$ to $\log^*</td>
</tr>
<tr>
<td>For $i = 1$ to $(n-t)$, let $y_i^t$ be the length of the longest common prefix of $y_i^{t-1}$ and $y_{i+1}^{t-1}$.</td>
</tr>
</tbody>
</table>

As (most of the) elements are paired twice on every step, the dataset 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 $\text{RecPrefix}$ 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 $x_t$ might affect $2^t$ elements at depth $t$, causing the privacy loss to deteriorate rapidly.

4.4 Query Release and Distribution Learning

4.4.1 Equivalences with the Interior Point Problem

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 4.4.1.** Let $X$ be a totally ordered domain. Then,

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

2. If there exists an $(\varepsilon, \delta)$-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 $(\alpha, \beta)$-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.

**Proof of Theorem 4.4.1 item 1.** Suppose $A$ is a private $(\alpha, \beta)$-accurate algorithm for releasing thresholds over $X$ for datasets of size $\frac{n}{8\alpha}$. Define $A'$ on datasets of size $n$ to pad the dataset with an equal
number of \( \min \{ \mathcal{X} \} \) and \( \max \{ \mathcal{X} \} \) 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) dataset. \( \square \)

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 4.4.2 ([48]).** For every finite data universe \( \mathcal{X} \), and \( n \in \mathbb{N}, \varepsilon, \beta > 0 \), there is an \( \varepsilon \)-differentially private algorithm \( \mathcal{A} \) that releases all threshold queries on \( \mathcal{X} \) with \( (\alpha, \beta) \)-accuracy for

\[
\alpha = \frac{4 \log(1/\beta) \log \log |\mathcal{X}|}{\varepsilon n}.
\]

The idea of the reduction is to create noisy partitions of the input dataset 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 polylog(1/\alpha) error for these answers.

This reduction furthermore preserves computational efficiency, requiring \( O(1/\alpha) \) invocations of the interior point algorithm on a subset of its input dataset, plus time needed to sort the input dataset and run the \( \tilde{O}(1/\alpha) \)-time algorithm of [48].

**Proof of Theorem 4.4.1 item 2.** Let \( R : \mathcal{X}^* \rightarrow \mathcal{X} \) be an \( (\varepsilon, \delta) \)-differentially private algorithm solving the interior point problem on \( \mathcal{X} \) with error \( \alpha \beta/24 \) and sample complexity \( m \). We may actually assume that \( R \) is differentially private in the sense that if \( D \in \mathcal{X}^* \) and \( D' \) differs from \( D \) up to the addition or removal of a row, then for every \( S \subseteq \mathcal{X} \), \( \Pr[R(D) \in S] \leq e^\varepsilon \Pr[R(D') \in S] + \delta \), 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 datasets of size less than \( m \) with an arbitrary fixed element, and subsample the first \( m \) entries from any dataset with size greater than \( m \).

Consider the following algorithm for answering thresholds on datasets \( D \in \mathcal{X}^n \) for \( n > m \):
Algorithm 4 \textit{Thresh}(D)

\textbf{Input:} dataset \(D = (x_1, \ldots, x_n) \in \mathcal{X}^n\).

1. Sort \(D\) in nondecreasing order \(x_1 \leq x_2 \leq \ldots \leq x_n\).

2. Set \(k = 6/\alpha\) and let \(t_0 = 1, t_1 = t_0 + an/3 + \nu_1, t_2 = t_1 + an/3 + \nu_2, \ldots, t_k = t_{k-1} + an/3 + \nu_k\) where each \(\nu_\ell \sim \text{Lap}(1/\varepsilon)\) independently.

3. Divide \(D\) into blocks \(D_1, \ldots, D_k\), where \(D_\ell = (x_{t_{\ell-1}}, \ldots, x_{t_\ell-1})\) (setting \(x_j = \max \mathcal{X}\) if \(j > n\); note some \(D_\ell\) may be empty).

4. Let \(r_0 = \min \mathcal{X}, r_1 = R(D_1), \ldots, r_k = R(D_k)\) and define \(\hat{D}\) from \(D\) by replacing each \(x_j\) with the largest \(r_\ell\) for which \(r_\ell \leq x_j\).

5. Run algorithm \(A\) from Lemma 4.4.2 on \(\hat{D}\) over the universe \(\{r_0, r_1, \ldots, r_k\}\) to obtain threshold query answers \(a_{r_0}, a_{r_1}, \ldots, a_{r_k}\). Use privacy parameter \(\varepsilon\) and confidence parameter \(\beta/4\).

6. Answer arbitrary threshold queries by interpolation, i.e. for \(r_\ell \leq t < r_{\ell+1}\), set \(a_t = a_{r_\ell}\).

7. Output \((a_t)_{t \in \mathcal{X}}\).

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

\[
 x_1 \leq \ldots \leq x_{i-1} \leq x_{i+1} \leq \ldots \leq x_j \leq x_i' \leq x_{j+1} \leq \ldots \leq x_n.
\]

We write vectors of noise values as \(\nu = (\nu_1, \nu_2, \ldots, \nu_k)\). There is a bijection between noise vectors \(\nu\) and noise vectors \(\nu'\) such that \(D\) partitioned according to \(\nu\) and \(D'\) partitioned according to \(\nu'\) differ on at most two blocks: namely, if \(\ell, r\) are the indices for which \(t_{\ell-1} \leq i < t_\ell\) and \(t_{r-1} \leq j < t_r\) (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 \((D_1, \ldots, D_k)\) according to \(\nu\) differs from \(D'\) partitioned into \((D'_1, \ldots, D'_k)\) according to \(\nu'\) by a removal of an element from one block (namely \(D_\ell\)) and the addition of an element to another block (namely \(D'_r\)). Thus, for every set \(S \subseteq \mathcal{X}^m\),

\[
 \Pr[(R(D_1), \ldots, R(D_k)) \in S | \nu] \leq e^{2\varepsilon} \Pr[(R(D'_1), \ldots, R(D'_k)) \in S | \nu'] + (1 + e^{\varepsilon})\delta.
\]
Moreover, under the bijection we constructed between $\nu$ and $\nu'$, noise vector $\nu'$ is sampled with density at most $e^{2\varepsilon}$ times the density of $\nu$, so for every set $S \subseteq \mathcal{X}^m$,

$$\Pr[(R(D_1), \ldots, R(D_k)) \in S] \leq e^{2\varepsilon} (e^{2\varepsilon} \Pr[(R(D'_1), \ldots, R(D'_k)) \in S]) + (1 + e^\varepsilon)\delta = e^{4\varepsilon} \Pr[(R(D'_1), \ldots, R(D'_k)) \in S] + (1 + e^\varepsilon)\delta.$$

Finally, the execution of $A$ at the end of the algorithm is $\varepsilon$-differentially private, so by composition (Lemma 2.1.5), we obtain the asserted level of privacy.

**Utility** We can produce $\alpha$-accurate answers to every threshold function as long as

1. The partitioning exhausts the dataset, i.e. every element of $D$ is in some $D_i$,
2. Every execution of $R$ succeeds at finding an interior point,
3. Every dataset $D_i$ 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

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

So by a union bound, every $\nu_i$ is at least $(-\alpha n/6)$ with probability at least $1 - \beta/4$. If this is the case, then item 1 holds because $t_k = k \cdot \alpha n/3 + \nu_1 + \ldots + \nu_k \geq (6/\alpha)(\alpha n/3) + (6/\alpha)(-\alpha n/6) \geq n$. 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 subdataset $D_i$ succeeds with probability $1 - \alpha \beta/24$.

By a similar argument, property 3 holds as long as each noise value $\nu_i$ 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.
4.5 PAC Learning

4.5.1 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 $(\alpha, \beta)$-accuracy and of solving the interior point problem with error probability $\Theta(\beta)$. Specifically, we show

**Theorem 4.5.1.** Let $\mathcal{X}$ be a totally ordered domain. Then,

1. If there exists an $(\varepsilon, \delta)$-differentially private algorithm solving the interior point problem on $\mathcal{X}$ with error probability $\beta$ 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 with sample complexity $\max\left\{ \frac{n}{2\alpha}, \frac{4\log(2/\beta)}{\alpha} \right\}$.

2. If there exists an $(\varepsilon, \delta)$-differentially private $(\alpha, \beta)$-accurate proper PAC learner for $\text{Thresh}_\mathcal{X}$ with sample complexity $n$, then there is a $(2\varepsilon, (1 + e^\varepsilon)\delta)$-differentially private algorithm that solves the interior point problem on $\mathcal{X}$ with error $\beta$ and sample complexity $27n$.

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 4.5.2.** Let $\mathcal{X}$ be a totally ordered domain. Then,

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

2. If there exists an $(\varepsilon, \delta)$-differentially private algorithm that is able to properly and empirically learn thresholds on $\mathcal{X}$ with $(\alpha, \beta)$-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 $\mathcal{X}$ with error $\beta$ and sample complexity $n$. 

103
Proof. For the first direction, let \( A \) be a private algorithm for the interior point problem on datasets of size \( n \). Consider the algorithm \( A' \) that, on input a dataset \( D \) of size \( n/(2\alpha) \), runs \( A' \) on a dataset \( D' \) 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 \( \min\{\mathcal{X}\} \)'s or \( \max\{\mathcal{X}\} \)'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 \( \alpha \)-consistent with \( D \). For privacy, note that changing one row of \( D \) changes at most two rows of \( D' \). Hence, applying algorithm \( A \) preserves \((2\varepsilon, (\varepsilon + 1)\delta)\)-differential privacy.

For the reverse direction, suppose \( A' \) privately finds an \( \alpha \)-consistent threshold functions for datasets of size \( n/(3\alpha) \). Define \( A \) on a dataset \( D' \in \mathcal{X}^n \) to label the smaller \( n/2 \) points 1 and the larger \( n/2 \) points 0 to obtain a labeled dataset \( D \in (\mathcal{X} \times \{0, 1\})^n \), pad \( D \) with an equal number of \( (\min\{\mathcal{X}\}, 1) \) and \( (\max\{\mathcal{X}\}, 0) \) entries to make it of size \( n/(3\alpha) \), and run \( A' \) on the result. Note that if \( c_x \) is a threshold function with error at most \( \alpha \) 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 \( A' \) preserves \((2\varepsilon, (\varepsilon + 1)\delta)\)-differential privacy. \( \square \)

The reduction of Lemma 4.5.2 relies crucially on the fact that the empirical learner \( A' \) is proper. If \( A' \) is an improper learner, it is not clear how to implement the step of extracting an interior point from a consistent hypothesis that is not itself a threshold function. It is an intriguing open problem to resolve the sample complexity of (empirically) learning threshold functions with approximate differential privacy.

Open Problem 4.5.3 (Improperly learning thresholds). What is the sample complexity of improperly learning the concept class \( \text{Thresh}_{\mathcal{X}} \) with \((\varepsilon, \delta)\)-differential privacy (via either empirical learning or PAC learning)?

An answer that is \( \omega(\log(1/\delta)/\varepsilon) \) would resolve Open Problem 4.1.5.

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

In general, an algorithm that can output an \( \alpha \)-consistent hypothesis from concept class \( \mathcal{C} \) can also be used to learn \( \mathcal{C} \) using \( \max\{n, 64 \text{VC}(\mathcal{C}) \log(512/\alpha^2)/\alpha \} \) samples [15]. 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.

**Lemma 4.5.4.** Any algorithm $\mathcal{A}$ for empirically learning $\text{Thresh}_X$ with $(\alpha, \beta)$-accuracy is also a $(2\alpha, \beta + \beta')$-accurate PAC learner for $\text{Thresh}_X$ 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 $c = q_x \in \text{Thresh}_X$. It suffices to show that for a sample $S = ((x_i, c(x_i)), \ldots (x_m, c(x_m)))$ where $m \geq 4\ln(2/\beta')/\alpha$ and the $x_i$ are drawn i.i.d. from $\mathcal{D}$, it holds that

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

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

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

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 $\alpha m$ 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. \qed

**4.6 Thresholds in High Dimension**

We next show that the bound of $\Omega(\log^* |X|)$ on the sample complexity of private proper-learners for $\text{Thresh}_X$ extends to conjunctions of $\ell$ independent threshold functions in $\ell$ 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. Since the concept class of $\ell$-dimensional thresholds has VC dimension of $\ell$, we obtain an $\omega(\text{VC}(\mathcal{C}))$ lower bound for concept classes even with arbitrarily large VC dimension.

Consider the following extension of $\text{Thresh}_X$ to $\ell$ dimensions.
Definition 4.6.1. For a totally ordered set \( \mathcal{X} \) and \( \bar{a} = (a_1, \ldots, a_\ell) \in \mathcal{X}^{\ell} \) define the concept 
\[ c_\bar{a} : \mathcal{X}^{\ell} \to \{0, 1\} \]
where \( c_\bar{a}(\bar{x}) = 1 \) 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 \( \mathcal{X}^{\ell} \) as 
\[ \text{Thresh}_{\mathcal{X}}^{\ell} = \{ c_\bar{a} \bar{a} \in \mathcal{X}^{\ell} \} . \]

Note that the VC dimension of \( \text{Thresh}_{\mathcal{X}}^{\ell} \) is \( \ell \). We obtain the following lower bound on the sample complexity of privately learning \( \text{Thresh}_{\mathcal{X}}^{\ell} \).

Theorem 4.6.2. For every \( n, \ell \in \mathbb{N}, \alpha > 0, \text{ and } \delta \leq \ell^2/(1500n^2) \), any \( (\varepsilon = \frac{1}{2}, \delta) \)-differentially private and \( (\alpha, \beta = \frac{1}{5}) \)-accurate proper learner for \( \text{Thresh}_{\mathcal{X}}^{\ell} \) requires sample complexity 
\[ n = \Omega(\frac{\ell^2}{\alpha} \log^* |\mathcal{X}|) . \]

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 \( C^{\ell} \) of conjunctions of \( \ell \) different concepts from \( C \) requires sample complexity \( \Omega(\ell n) \).

Definition 4.6.3. For \( \ell \in \mathbb{N} \), a data universe \( \mathcal{X} \) and a concept class \( C \) over \( \mathcal{X} \), define a concept class \( C^{\ell} \) over \( \mathcal{X} \) to consist of all \( \bar{c} = (c_1, \ldots, c_\ell) \), where \( \bar{c} : \mathcal{X}^{\ell} \to \{0, 1\} \) is defined by 
\[ \bar{c}(\bar{x}) = c_1(x_1) \land c_2(x_2) \land \cdots \land c_\ell(x_\ell) . \]

Theorem 4.6.4. Let \( \alpha, \beta, \varepsilon, \delta > 0 \). Let \( C \) be a concept class over a data universe \( \mathcal{X} \), and assume there is a domain element \( p_1 \in \mathcal{X} \) s.t. \( c(p_1) = 1 \) for every \( c \in C \). Let \( \mathcal{D} \) be a distribution over datasets containing \( n \) examples from \( \mathcal{X} \) labeled by a concept in \( C \), and suppose that every \( (\varepsilon, \delta) \)-differentially private algorithm fails to find an \( (\alpha/\beta) \)-consistent hypothesis \( h \in C \) for \( D \sim \mathcal{D} \) with probability at least \( 2\beta \). Then any \( (\varepsilon, \delta) \)-differentially private and \( (\alpha, \beta) \)-accurate proper learner for \( C^{\ell} \) requires sample complexity \( \Omega(\ell n) \).

Note that in the the above theorem we assumed the existence of a domain element \( p_1 \in \mathcal{X} \) 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 \( \mathcal{X} \) defined as 
\[ \text{Point}_{\mathcal{X}} = \{ c_x : x \in \mathcal{X} \} \] where \( c_x(y) = 1 \) iff \( y = x \). As was shown in [8], this class can be privately learned using \( O_{\alpha, \beta, \varepsilon, \delta}(1) \) labeled examples (i.e., the sample complexity has no dependency in \( |\mathcal{X}| \)). Observe that since there is no \( x \in \mathcal{X} \) on which every point concept evaluates to 1, we cannot use Theorem 4.6.4 to lower bound the sample complexity of privately learning \( \text{Point}_{\mathcal{X}}^{\ell} \). Indeed, the class \( \text{Point}_{\mathcal{X}}^{\ell} \) is identical (up to renaming of domain elements) to the class \( \text{Point}_{\mathcal{X}^{\ell}} \), and can be privately learned using \( O_{\alpha, \beta, \varepsilon, \delta}(1) \) labeled examples.
Remark 4.6.5. Similarly to Theorem 4.6.4 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 \( \ell \) concepts from \( C \) requires sample complexity \( \ell n \).

Proof of Theorem 4.6.4. Assume toward a contradiction that there exists an \((\varepsilon, \delta)\)-differentially private and \((\alpha, \beta)\)-accurate proper learner \( A \) for \( C^\ell \) using \( \ell n/9 \) samples. Recall that the task of privately outputting a good hypothesis on any fixed dataset is essentially equivalent to the task of private PAC learning (See Section 4.5.1). We can assume, therefore, that \( A \) outputs an \( \alpha \)-consistent hypothesis for every fixed dataset of size at least \( n' \triangleq \ell n \) with probability at least \( 1 - \beta \).

We construct an algorithm \( \text{Solve}_D \) which uses \( A \) in order to find an \((\alpha/\beta)\)-consistent threshold function for datasets of size \( n \) from \( D \). Algorithm \( \text{Solve}_D \) takes as input a set of \( n \) labeled examples in \( X \) and applies \( A \) on a dataset containing \( n' \) labeled examples in \( X^\ell \). The \( n \) input points are embedded along one random axis, and random samples from \( D \) are placed on each of the other axes (with \( n \) labeled points along each axis).

**Algorithm 5 \( \text{Solve}_D \)**

**Input:** dataset \( D = (x_i, y_i)_{i=1}^n \in (X \times \{0,1\})^n \).

1. Initiate \( S \) as an empty multiset.

2. Let \( r \) be a (uniform) random element from \( \{1, 2, \ldots, \ell\} \).

3. For \( i = 1 \) to \( n \), let \( \tilde{z}_i \in X^\ell \) be the vector with \( r \)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 \((\tilde{z}_i, y_i)\).

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 \( D \).
   (b) For \( i = 1 \) to \( n \), let \( \tilde{z}'_i \in X^\ell \) be the vector whose \( t \)th coordinate is \( x'_i \), and its other coordinates are \( p_1 \). Add to \( S \) the labeled example \((\tilde{z}'_i, y'_i)\).

5. Let \((h_1, h_2, \ldots, h_\ell) = \vec{h} \leftarrow A(S)\).

6. Return \( h_r \).

First observe that \( \text{Solve}_D \) is \((\varepsilon, \delta)\)-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 \((\varepsilon, \delta)\)-differentially private algorithm \( \mathcal{A} \) on \( S \) preserves privacy.

Consider the execution of Solve\( \mathcal{D} \) on a dataset \( D \) 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 \( \mathcal{C} \). 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 \( (\tilde{z}_i^t, y_i^t)_{i=1}^n \) 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

\[
(c_1, c_2, \ldots, c_\ell)(\tilde{z}_i^t) = c_1(p_1) \land c_2(p_1) \land \ldots \land c_\ell(x_i^t) \land \ldots \land c_\ell(p_1) = y_i^t,
\]

and hence \( S \) is perfectly labeled by \( (c_1, c_2, \ldots, c_\ell) \in \mathcal{C} \).

By the properties of \( \mathcal{A} \), with probability at least \( 1 - \beta \) we have that \( \tilde{h} \) (from Step 5) is an \( \alpha \)-consistent hypothesis for \( S \). Assuming that this is the case, there could be at most \( \beta \ell \) “bad” axes on which \( \tilde{h} \) 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 \( \beta \ell / \ell = \beta \). Overall, Solve\( \mathcal{D} \) outputs an \( (\alpha/\beta) \)-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 4.6.2 follows from the lower bound on the sample complexity of privately finding an \( \alpha \)-consistent threshold function (see Section 4.2.3):

**Lemma 4.6.6** (Follows from Lemma 4.2.9 and 4.5.2). There exists a constant \( \lambda > 0 \) s.t. the following holds. For every totally ordered data universe \( \mathcal{X} \) there exists a distribution \( \mathcal{D} \) over datasets containing at most \( n = \frac{\lambda}{\alpha} \log^* |\mathcal{X}| \) labeled examples from \( \mathcal{X} \) such that every \((\frac{1}{2}, \frac{1}{50n^2})\)-differentially private algorithm fails to find an \( \alpha \)-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 4.6.2 also yields the following lower bound on the sample complexity of releasing approximated answers to queries from Thresh\( \mathcal{X} \).

**Theorem 4.6.7.** 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 Thresh\( \mathcal{X} \) with \((\alpha, \frac{1}{150})\)-accuracy must have sample complexity \( n = \Omega(\frac{\ell}{\alpha} \log^* |\mathcal{X}|) \).
In order to prove the above theorem we use our lower bound on privately learning \( \text{Thresh}^f_X \) together with the following reduction from private learning to query release.

**Lemma 4.6.8** ([8, 72]). 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 4.6.7.** Let \( \delta \leq \ell^2/(7500n^2) \). Combining our lower bound on the sample complexity of privately learning \( \text{Thresh}^f_X \) (Theorem 4.6.2) together with the reduction stated in Lemma 4.6.8, 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 \( \text{Thresh}^f_X \) 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 \( \tilde{A} \) for releasing queries from \( \text{Thresh}^f_X \) with \((\alpha, \frac{1}{150})\)-accuracy and sample complexity \( n < m/(150\alpha) \).

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

Note that as \( \tilde{A} \) is \((\frac{1}{150}, \delta)\)-differentially private, so is \( A \). We now show that \( A \)'s output is \(\frac{1}{150}\)-accurate for \( D \) whenever \( \tilde{A} \)'s output is \( \alpha \)-accurate for \( \tilde{D} \), which happens with all but probability \( \frac{1}{150} \). Fix a query \( c \in \text{Thresh}^f_X \) 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 \( \tilde{A} \),

\[
\begin{align*}
a_c &= \frac{1}{150\alpha}(\tilde{a}_c - 1 + 150\alpha) \\
&\leq \frac{1}{150\alpha}(c(\tilde{D}) + \alpha - 1 + 150\alpha) \\
&= \frac{1}{150\alpha}(t/n + \alpha) \\
&= t/(150\alpha n) + 1/150 \\
&= c(D) + 1/150.
\end{align*}
\]
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.
Chapter 5

The Price of Online Queries in Differential Privacy

We next study the sample complexity of differential privacy for interactive query release. The focus of this chapter will be on understanding the relationship between different models of interaction between an analyst and a differentially private mechanism. We consider three different interactive models\(^1\) for how an analyst may specify queries. We list these in order of easiest to hardest for a mechanism to handle:

- **The Offline Model:** The analyst chooses the sequence of queries \(q_1, \ldots, q_k\), and they are given to the differentially private mechanism together in a batch. The mechanism may then answer the queries together.

- **The Online Model:** The analyst again chooses the queries \(q_1, \ldots, q_k\) in advance. However, the mechanism must now answer each query \(q_j\) before seeing \(q_{j+1}\).

- **The Adaptive Model:** The queries are not fixed in advance, and the analyst may choose query \(q_{j+1}\) in a way that depends on the answers to queries \(q_1, \ldots, q_j\).

In all three cases, we assume that \(q_1, \ldots, q_k\) are chosen from some family of allowable queries \(Q\), but may be chosen adversarially from this family.

---

\(^1\) Usually, the “interactive model” refers only to what we call the “adaptive model.” We prefer to call all of these models interactive, since they each require an interaction with a data analyst who issues the queries. We use the term “interactive” to distinguish these models from one where the algorithm only answers a fixed set of queries.
If a mechanism is required to answer arbitrary counting queries, i.e. if the class of allowable queries $Q$ is the class of all counting queries on a domain $X$, then these three models are essentially equivalent. For instance, the simple Laplace mechanism is able to take a dataset of $n$ individuals and answer $\tilde{\Omega}(n)$ adaptively chosen counting queries with error $o(1/\sqrt{n})$. In contrast, the seminal lower bound of Dinur and Nissim and its later refinements [41,58] shows that there exists a fixed set of $O(n)$ queries that cannot be answered by any differentially private algorithm with such little error, even in the easiest offline model. For a different example, the private multiplicative weights algorithm of Hardt and Rothblum [76] can in many cases answer an exponential number of arbitrary, adaptively-chosen statistical queries with a strong accuracy guarantee, whereas Chapter 3 showed that the accuracy guarantee of private multiplicative weights is nearly optimal even for a simple, fixed family of queries.

In this chapter, we show that these three models are all distinct. In fact, we show exponential separations between each of the three models. These are the first separations between these models in differential privacy.

5.1 Results

We show that when counting queries come from a restricted family of allowable queries, then it can become strictly easier to answer a subset of these queries in the offline model than it is to answer a sequence of queries presented online.

**Theorem 5.1.1** (Informal). There exists a data universe $X$ and a family of counting queries $Q$ on $X$ such that for every $n \in \mathbb{N},$

1. There is a differentially private algorithm that takes a dataset $D \in X^n$ and answers any set of $k = 2^{\Omega(\sqrt{n})}$ offline queries from $Q$ up to error $\pm 1/100$ from $Q$, but

2. No differentially private algorithm can take a dataset $D \in X^n$ and answer an arbitrary sequence of $k = O(n^2)$ online (but not adaptively-chosen) queries from $Q$ up to error $\pm 1/100$.

This result establishes that the online model is strictly harder than the offline model. We also demonstrate that the adaptive model is strictly harder than the online model. Here, the family of
queries we use in our separation is not a family of counting queries, but is rather a family of search queries with a specific definition of accuracy that we will define later.

**Theorem 5.1.2** (Informal). For every \( n \in \mathbb{N} \), there is a family of “search” queries \( Q \) on datasets in \( X^n \) such that

1. There is a differentially private algorithm that takes a dataset \( D \in \{\pm 1\}^n \) and accurately answers any online (but not adaptively-chosen) sequence of \( k = 2^{\Omega(n)} \) queries from \( Q \), but

2. No differentially private algorithm can take a dataset \( D \in \{\pm 1\}^n \) and accurately answer an adaptively-chosen sequence of \( k = O(1) \) queries from \( Q \).

We leave it as an interesting open question to separate the online and adaptive models for counting queries, or to show that the models are equivalent for counting queries.

**Open Problem 5.1.3** (Online vs. adaptive models for counting queries). Is there a family of counting queries \( Q \) and a number \( k \) such that

1. There is an \((\varepsilon, \delta)\)-differentially private algorithm that accurately answers any online sequence of \( k \) queries, but

2. No \((\varepsilon, \delta)\)-differentially private algorithm can accurately answer \( k \) adaptively chosen queries?

Although Theorems 5.1.1 and 5.1.2 separate the three models, these results use somewhat contrived families of queries. Thus, we also investigate whether the models are distinct for natural families of queries that are of use in practical applications. It is already of interest to focus on threshold queries as studied in Chapter 4. We consider the family \( Q_{\text{thresh}} \) of threshold queries defined on the (infinite) universe \([0, 1]\), where each query is specified by a point \( \tau \in [0, 1] \) and asks “what fraction of the elements of the dataset are at most \( \tau \)?” If we restrict our attention to so-called pure differential privacy (i.e. \((\varepsilon, \delta)\)-differential privacy with \( \delta = 0 \)), then we obtain an exponential separation between the offline and online models for answering threshold queries.

**Theorem 5.1.4** (Informal). For every \( n \in \mathbb{N} \),

1. There is a pure differentially private algorithm that takes a dataset \( D \in [0, 1]^n \) and answers any set of \( k = 2^{\Omega(n)} \) offline queries from \( Q_{\text{thresh}} \) up to error \( \pm 1/100 \), but
2. No pure differentially private algorithm takes a dataset \( D \in [0, 1]^n \) and answers an arbitrary sequence of \( k = O(n) \) online (but not adaptively-chosen) queries from \( Q_{\text{thresh}} \) up to error \( \pm 1/100 \).

We also ask whether or not such a separation exists for arbitrary differentially private algorithms (i.e. \((\varepsilon, \delta)\)-differential privacy with \( \delta > 0 \)). Theorem 5.1.4 shows that, for pure differential privacy, threshold queries have near-maximal sample complexity. That is, up to constant factors, the lower bound for online threshold queries matches what is achieved by the Laplace mechanism, which is applicable to arbitrary counting queries. This may lead one to conjecture that adaptive threshold queries also require near-maximal sample complexity subject to approximate differential privacy. However, we show that this is not the case:

**Theorem 5.1.5.** For every \( n \in \mathbb{N} \), there is a differentially private algorithm that takes a dataset \( D \in [0, 1]^n \) and answers any set of \( k = 2^{\Omega(n)} \) adaptively-chosen queries from \( Q_{\text{thresh}} \) up to error \( \pm 1/100 \).

In contrast, for any offline set of \( k \) thresholds \( \tau_1, \ldots, \tau_k \), we can round each element of the dataset up to an element in the finite universe \( \mathcal{X} = \{\tau_1, \ldots, \tau_k, 1\} \) without changing the answers to any of the queries. Then we can use known the algorithm of Chapter 4 to answer the queries using a very small dataset of size \( n = 2^{(1+o(1))\log^*(k)} \). We leave it as an interesting open question to settle the complexity of answering adaptively-chosen threshold queries in the adaptive model.

**Open Problem 5.1.6** (Releasing adaptively chosen thresholds). What is the sample complexity of releasing \( k \) adaptively chosen threshold functions over a huge data universe (e.g. the interval \([0, 1]\)) with \((\varepsilon, \delta)\)-differential privacy?

Answering this question may help resolve the question we sought to answer through our investigation of thresholds in the first place.

**Open Problem 5.1.7** (Offline vs. online models for natural queries). Is there a “natural” family of counting queries (e.g. thresholds or marginals) which separate the offline and online models for approximate differential privacy? Or even the offline and adaptive models?
5.2 Techniques

Separating Offline and Online Queries

To prove Theorem 5.1.1, we construct a sequence of queries $q_1, \ldots, q_k$ such that, for all $j \in [k]$,

- $q_j$ “reveals” the answers to $q_1, \ldots, q_{j-1}$, but
- $q_1, \ldots, q_{j-1}$ do not reveal the answer to $q_j$.

Thus, given the sequence $q_1, \ldots, q_k$ in the offline setting, the answers to $q_1, \ldots, q_{k-1}$ are revealed by $q_k$. So only $q_k$ needs to be answered and the remaining query answers can be inferred. However, in the online setting, each query $q_{j-1}$ must be answered before $q_j$ is presented and this approach does not work. This is the intuition for our separation.

To prove the online lower bound, we build on a lower bound of Chapter 3 for 1-way marginal (i.e. attribute mean) queries, which is based on the existence of short secure fingerprinting codes [21,112]. Consider the data universe $\{\pm 1\}^k$. Recall that given a dataset $D \in \{\pm 1\}^{n \times k}$, a marginal query is a specific type of statistical query that asks for the mean of a given column of $D$. In Chapter 3, we saw that unless $k \ll n^2$, there is no differentially private algorithm that answers all $k$ marginal queries with non-trivial accuracy. This was done by showing that such an algorithm would violate the security of a short fingerprinting code due to Tardos [112]. We are able to “embed” $k$ 1-way marginal queries into the sequence of online queries $q_1, \ldots, q_k$. Thus a modification of the lower bound for marginal queries applies in the online setting.

To prove the offline upper bound, we use the fact that every query reveals information about other queries. However, we must handle arbitrary sequences of queries, not just the specially-constructed sequences used for the lower bound. The key property of our family of queries is the following. Each element $x$ of the data universe $\mathcal{X}$ requires $k$ bits to specify. On the other hand, for any set of queries $q_1, \ldots, q_k$, we can specify $q_1(x), \ldots, q_k(x)$ using only $O(\log(nk))$ bits. Thus the effective size of the data universe given the queries is $\text{poly}(nk)$, rather than $2^k$. Then we can apply a differentially private algorithm that gives good accuracy as long as the data universe has subexponential size [14]. Reducing the size of the data universe is only possible once the queries have been specified; hence this approach only works in the offline setting.
Separating Online and Adaptive Queries

To prove Theorem 5.1.2, we start with the classical randomized response algorithm [120]. Specifically, given a dataset $x \in \{\pm 1\}^n$, randomized response produces a new dataset $y \in \{\pm 1\}^n$ where each coordinate $y_i$ is independently set to $+x_i$ with probability $(1+\alpha)/2$ and is set to $-x_i$ with probability $(1-\alpha)/2$.\(^2\) It is easy to prove that this algorithm is $(O(\alpha),0)$-differentially private. What accuracy guarantee does this algorithm satisfy? By design, it outputs a vector $y$ that has correlation approximately $\alpha$ with the dataset $x$ — that is, $\langle y, x \rangle \approx \alpha n$. On the other hand, it is also easy to prove that there is no differentially private algorithm (for any reasonable privacy parameters) that can output a vector that has correlation at least $1/2$ with the sensitive dataset.

Our separation between the online and adaptive models is based on the observation that, if we can obtain $O(1/\alpha^2)$ “independent” vectors $y^1, \ldots, y^k$ that are each roughly $\alpha$-correlated with $x$, then we can obtain a vector $z$ that is $(1/2)$-correlated with $x$, simply by letting $z$ be the coordinate-wise majority of the $y^j$s. Thus, no differentially private algorithm can output such a set of vectors. More precisely, we require that $\langle y^i, y^j \rangle \approx \alpha^2 n$ for $i \neq j$, which is achieved if each $y^j$ is an independent sample from randomized response.

Based on this observation, we devise a class of queries such that, if we are allowed to choose $k$ of these queries adaptively, then we obtain a set of vectors $y^1, \ldots, y^k$ satisfying the conditions above. This rules out differential privacy for $k = O(1/\alpha^2)$ adaptive queries. The key is that we can use adaptivity to ensure that each query asks for an “independent” $y^j$ by adding the previous answers $y^1, \ldots, y^{j-1}$ as constraints in the search query.

On the other hand, randomized response can answer each such query with high probability. If a number of these queries is fixed in advance, then, by a union bound, the vector $y$ output by randomized response is simultaneously an accurate answer to every query with high probability. Since randomized response is oblivious to the queries, we can also answer the queries in the online model, as long as they are not chosen adaptively.

At a high level, the queries that achieve this property are of the form “output a vector $y \in \{\pm 1\}^n$ that is approximately $\alpha$-correlated with $x$ and is approximately as uncorrelated as possible with the

\(^2\)When discussing our separation between online and adaptive queries, we diverge from the notation in the rest of this thesis by referring to datasets with the notation $x$ or $y$. This is to emphasize that each datum in the problem we consider is simply a single bit.
vectors $v^1, \ldots, v^m$. A standard concentration argument shows that randomized response gives an accurate answer to all the queries simultaneously with high probability. On the other hand, if we are allowed to choose the queries adaptively, then for each query $q_i$, we can ask for a vector $y^i$ that is correlated with $x$ but is as uncorrelated as possible with the previous answers $y^1, \ldots, y^{i-1}$.

**Threshold Queries**

For pure differential privacy, our separation between offline and online threshold queries uses a simple argument based on binary search. Our starting point is a lower bound showing that any purely differentially private algorithm that takes a dataset of $n$ points $x_1, \ldots, x_n \in \{1, \ldots, T\}$ and outputs an approximate median of these points requires $n = \Omega(\log T)$. This lower bound follows from a standard application of the “packing” technique of Hardt and Talwar [78]. On the other hand, by using binary search, any algorithm that can answer $k = O(\log T)$ adaptively-chosen threshold queries can be used to find an approximate median. Thus, any purely differentially private algorithm for answering such queries requires a dataset of size $n = \Omega(k)$. Using the structure of the lower bound argument, we show that the same lower bound holds for online non-adaptive queries as well. In contrast, we can treat $k$ offline threshold queries as though they were asked on a domain of size only $k + 1$. Using the algorithms of [31, 48, 49], we can answer $k$ offline queries on a dataset of only $n = O(\log k)$ elements, giving an exponential separation.

The basis of our improved algorithm for adaptive threshold queries under approximate differential privacy is a generalization of the sparse vector technique [48, 76, 105] (see [54, §3.6] for a textbook treatment). Our algorithm makes crucial use of a stability argument similar to the propose-test-release techniques of Dwork and Lei [46]. To our knowledge, this is the first use of a stability argument for any online or adaptive problem in differential privacy and may be of independent interest. In particular, our algorithm is given an input $D \in \mathcal{X}^n$, a threshold $t \in (0, 1)$, and an adaptive sequence of statistical (or low-sensitivity) queries $q_1, \ldots, q_k : \mathcal{X}^n \to [0, 1]$ and, for each query $q_j$, it reports (i) $q_j(D) \geq t$, (ii) $q_j(D) \leq t$, or (iii) $t - \alpha \leq q_j(D) \leq t + \alpha$. The sample complexity of this algorithm is $n = O(\sqrt{\varepsilon \log(k/\varepsilon \delta)/\varepsilon \alpha})$, where $k$ is the total number of queries, $c$ is an upper bound on the number of times (iii) may be reported, and $(\varepsilon, \delta)$-differential privacy is provided. We call this the **BetweenThresholds algorithm**.

Once we have this algorithm, we can use it to answer adaptively-chosen thresholds using an
approach inspired by [27] (the work on which Chapter 4 is based). The high-level ideal is to sort the dataset \( x(1) < x(2) < \ldots < x(n) \) and then partition it into chunks of consecutive sorted elements. For any chunk, and a threshold \( \tau \), we can use the between thresholds algorithm to determine (approximately) whether \( \tau \) lies below all elements in the chunk, above all elements in the chunk, or inside the chunk. Obtaining this information for every chunk is enough to accurately estimate the answer to the threshold query up to an error proportional to the size of the chunks. The sample complexity is dominated by the \( O(\log k) \) sample complexity of our \texttt{BetweenThresholds} algorithm multiplied by the number of chunks needed, namely \( O(1/\alpha) \).

5.3 Models of Interactive Queries

Search Queries. A search query \( q \) on \( \mathcal{X}^n \) is defined by a loss function \( L_q : \mathcal{X}^n \times \mathcal{R} \to [0, \infty) \), where \( \mathcal{R} \) is an arbitrary set representing the range of possible outputs. For a dataset \( D \in \mathcal{X}^n \) and an output \( y \in \mathcal{R} \), we will say that \( y \) is \( \alpha \)-accurate for \( q \) on \( D \) if \( L_q(D, y) \leq \alpha \). In some cases the value of \( L_q \) will always be either 0 or 1. Thus we simply say that \( y \) is accurate for \( q \) on \( D \) if \( L_q(D, y) = 0 \).

For example, if \( \mathcal{X}^n = \{\pm 1\}^n \), we can define a search query by \( \mathcal{R} = \{\pm 1\}^n \), and \( L_q(x, y) = 0 \) if \( \langle x, y \rangle \geq \alpha n \) and \( L_q(x, y) = 1 \) otherwise. In this case, the search query would ask for any vector \( y \) that has correlation \( \alpha \) with the dataset \( x \).

To see that counting queries are a special case of search queries, given a counting query \( q \) on \( \mathcal{X}^n \), we can define a search query \( L_q \) with \( \mathcal{R} = [0, 1] \) and \( L_q(D, a) = |q(D) - a| \). Then both definitions of \( \alpha \)-accurate align.

The goal of this work is to understand the implications of different ways to allow an adversary to query a sensitive dataset. In each of these models there is an algorithm \( M \) that holds a dataset \( D \in \mathcal{X}^n \), and a fixed family of (statistical or search) queries \( Q \) on \( \mathcal{X}^n \), and a bound \( k \) on the number of queries that \( M \) has to answer. There is also an adversary \( B \) that chooses the queries. The models differ in how the queries chosen by \( B \) are given to \( M \).

Offline

In the offline model, the queries \( q_1, \ldots, q_k \in Q \) are specified by the adversary \( B \) in advance and the algorithm \( M \) is given all the queries at once and must provide answers. Formally, we define the
following function \( \text{Offline}_{B \rightarrow M} : \mathcal{X}^n \rightarrow \mathcal{Q}^k \times \mathcal{R}^k \) depending on the adversary \( B \) and the algorithm \( M \).

**Figure 5.1:** \( \text{Offline}_{B \rightarrow M} : \mathcal{X}^n \rightarrow \mathcal{Q}^k \times \mathcal{R}^k \)

**Online Non-Adaptive**

In the *online non-adaptive* model, the queries \( q_1, \ldots, q_k \in \mathcal{Q} \) are again fixed in advance by the adversary, but are then given to the algorithm one at a time, and the algorithm must give an answer to query \( q_j \) before it is shown \( q_{j+1} \). We define a function \( \text{Online}_{B \rightarrow M} : \mathcal{X}^n \rightarrow \mathcal{Q}^k \times \mathcal{R}^k \) depending on the adversary \( B \) and the algorithm \( M \) as follows.

**Figure 5.2:** \( \text{Online}_{B \rightarrow M} : \mathcal{X}^n \rightarrow \mathcal{Q}^k \times \mathcal{R}^k \)

**Online Adaptive**

In the *online adaptive* model, the queries \( q_1, \ldots, q_k \in \mathcal{Q} \) are not fixed, and the adversary may choose each \( q_j \) based on the answers that the algorithm gave to the previous queries. We define a function \( \text{Adaptive}_{B \rightarrow M} : \mathcal{X}^n \rightarrow \mathcal{Q}^k \times \mathcal{R}^k \) depending on the adversary \( B \) and the algorithm \( M \) as follows.

**Definition 5.3.1** (Differential Privacy for Interactive Mechanisms). In each of the three cases — Offline, Online Non-Adaptive, or Online Adaptive — we say that \( M \) is \((\varepsilon, \delta)\)-differentially private if, for all adversaries \( B \), respectively \( \text{Offline}_{B \rightarrow M} \), \( \text{Online}_{B \rightarrow M} \), or \( \text{Adaptive}_{B \rightarrow M} \) is \((\varepsilon, \delta)\)-differentially private.
Input: $D \in X^n$. 
$M$ is given $D$. 
For $j = 1, \ldots, k$: 
$B$ chooses a query $q_j \in Q$. 
$M$ is given $q_j$ and outputs $a_j \in R$. 
Output: $(q_1, \ldots, q_k, a_1, \ldots, a_k) \in Q^k \times R^k$.

**Figure 5.3:** Adaptive $B_{-}^{-}M : X^n \rightarrow Q^k \times R^k$

**Definition 5.3.2** (Accuracy for Interactive Mechanisms). In each case — Offline, Online Non-Adaptive, or Online Adaptive queries — we say that $M$ is $(\alpha, \beta)$-accurate if, for all adversaries $B$ and all inputs $D \in X^n$,

$$
\Pr_{q_1, \ldots, q_k, a_1, \ldots, a_k} \left[ \max_{j \in [k]} L_{q_j}(D, a_j) \leq \alpha \right] \geq 1 - \beta, \tag{5.1}
$$

where $(q_1, \ldots, q_k, a_1, \ldots, a_k)$ is respectively drawn from one of Online $B_{-}^{-}M(D)$, Online $B_{+}^{-}M(D)$, or Adaptive $B_{-}^{-}M(D)$. We also say that $M$ is $\alpha$-accurate if the above holds with (5.1) replaced by

$$
\mathbb{E}_{q_1, \ldots, q_k, a_1, \ldots, a_k} \left[ \max_{j \in [k]} L_{q_j}(D, a_j) \right] \leq \alpha.
$$

**5.4 Separating Offline and Online Queries**

In this section we prove that online accuracy is strictly harder to achieve than offline accuracy, even for statistical queries. We prove our results by constructing a set of statistical queries that we call *prefix queries* for which it is possible to take a dataset of size $n$ and accurately answer superpolynomially many offline prefix queries in a differentially private manner, but it is impossible to answer more than $O(n^2)$ online prefix queries while satisfying differential privacy.

We now define the family of prefix queries. These queries are defined on the universe $X = \{\pm1\}^* = \bigcup_{j=0}^{\infty} \{\pm1\}^j$ consisting of all finite length binary strings.\(^4\) For $x, y \in \{\pm1\}^*$, we use $y \preceq x$ to denote that $y$ is a prefix of $x$. Formally,

$$
y \preceq x \iff |y| \leq |x| \text{ and } \forall i = 1, \ldots, |y| \quad y_i = x_i.
$$

\(^4\)All of the arguments in this section hold if we restrict to strings of length at most $k + \log n$. However, we allow strings of arbitrary length to reduce notational clutter.
Definition 5.4.1. For any finite set $S \subseteq \{\pm 1\}^*$ of finite-length binary strings, we define the prefix query $q_S : \{\pm 1\}^* \to \{\pm 1\}$ by

$$q_S(x) = 1 \iff \exists y \in S \ y \preceq x.$$ 

We also define

$$Q_{\text{prefix}} = \{ q_S | S \subseteq \{\pm 1\}^* \}$$

$$Q^B_{\text{prefix}} = \{ q_S | S \subseteq \{\pm 1\}^*, |S| \leq B \}$$

to be the set of all prefix queries and the set of prefix queries with sizes bounded by $B$, respectively.

5.4.1 Answering Offline Prefix Queries

We now prove that there is a differentially private algorithm that answers superpolynomially many prefix queries, provided that the queries are specified offline.

Theorem 5.4.2 (Answering Offline Prefix Queries). For every $\alpha, \varepsilon \in (0, 1/10)$, every $B \in \mathbb{N}$, and every $n \in \mathbb{N}$, there exists a

$$k = \min \left\{ 2^{\Omega(\sqrt{\alpha^3 \varepsilon n})}, 2^{\Omega(\alpha^3 \varepsilon n / \log(B))} \right\}$$

and an $(\varepsilon, 0)$-differentially private algorithm $M_{\text{prefix}} : \mathcal{X}^n \times (Q^B_{\text{prefix}})^k \to \mathbb{R}^k$ that is $(\alpha, 1/100)$-accurate for $k$ offline queries from $Q^B_{\text{prefix}}$.

We remark that it is possible to answer even more offline prefix queries by relaxing to $(\varepsilon, \delta)$-differential privacy for some negligibly small $\delta > 0$. However, we chose to state the results for $(\varepsilon, 0)$-differential privacy to emphasize the contrast with the lower bound, which applies even when $\delta > 0$, and to simplify the statement.

Our algorithm for answering offline queries relies on the existence of a good differentially private algorithm for answering arbitrary offline statistical queries. For concreteness, the so-called “BLR mechanism” of Blum, Ligett, and Roth [14] suffices, although different parameter tradeoffs can be obtained using different mechanisms. Differentially private algorithms with this type of guarantee exist only when the data universe is bounded, which is not the case for prefix queries. However, as we show, when the queries are specified offline, we can replace the infinite universe $\mathcal{X} = \{\pm 1\}^*$ with a finite, restricted universe $\mathcal{X}'$ and run the BLR mechanism. Looking ahead, the key to our
separation will be the fact that this universe restriction is only possible in the offline setting. Before we proceed with the proof of Theorem 5.4.2, we will state the guarantees of the BLR mechanism.

**Theorem 5.4.3** ([14]). For every $0 < \alpha, \varepsilon \leq 1/10$ and every finite data universe $\mathcal{X}$, if $Q_{S\mathcal{Q}}$ is the set of all statistical queries on $\mathcal{X}$, then for every $n \in \mathbb{N}$, there is a

$$k = 2^\Omega(\alpha^3 \varepsilon n / \log |\mathcal{X}|)$$

and an $(\varepsilon, 0)$-differentially private algorithm $M_{\text{BLR}} : \mathcal{X}^n \times Q_{S\mathcal{Q}}^k \rightarrow \mathbb{R}^k$ that is $(\alpha, 1/100)$-accurate for $k$ offline queries from $Q_{S\mathcal{Q}}$.

We are now ready to prove Theorem 5.4.2.

**Proof of Theorem 5.4.2.** Suppose we are given a set of queries $q_{S_1}, \ldots, q_{S_k} \in Q_{\text{prefix}}^B$ and a dataset $D \in \mathcal{X}^n$ where $\mathcal{X} = \{\pm 1\}^*$. Let $S = \bigcup_{j=1}^k S_j$. We define the universe $\mathcal{X}_S = S \cup \{\emptyset\}$ where $\emptyset$ denotes the empty string of length 0. Note that this universe depends on the choice of queries, and that $|\mathcal{X}_S| \leq kB + 1$. Since $\mathcal{X}_S \subset \mathcal{X}$, it will be well defined to restrict the domain of each query $q_{S_j}$ to elements of $\mathcal{X}_S$.

Next, given a dataset $D = (x_1, \ldots, x_n) \in \mathcal{X}^n$, and a collection of sets $S_1, \ldots, S_k \subset \mathcal{X}$, we give a procedure for mapping each element of $D$ to an element of $\mathcal{X}_S$ to obtain a new dataset $D^S = (x_1^S, \ldots, x_n^S) \in \mathcal{X}_S^n$ that is equivalent to $D$ with respect to the queries $q_{S_1}, \ldots, q_{S_k}$. Specifically, define $r_S : \mathcal{X} \rightarrow \mathcal{X}_S$ by

$$r_S(x) = \arg\max_{y \in \mathcal{X}_S, y \preceq x} |y|.$$ 

That is, $r_S(x)$ is the longest string in $\mathcal{X}_S$ that is a prefix of $x$. We summarize the key property of $r_S$ in the following claim

**Claim 5.4.4.** For every $x \in \mathcal{X}$, and $j = 1, \ldots, k$, $q_{S_j}(r_S(x)) = q_{S_j}(x)$.

**Proof of Claim 5.4.4.** First, we state a simple but important fact about prefixes: If $y, y'$ are both prefixes of a string $x$ with $|y| \leq |y'|$, then $y$ is a prefix of $y'$. Formally,

$$\forall x, y, y' \in \{0, 1\}^* \quad (y \preceq x \land y' \preceq x \land |y| \leq |y'|) \quad \Rightarrow \quad y \preceq y'. \quad (5.2)$$

Now, fix any $x \in \mathcal{X}$ and any query $q_{S_j}$ and suppose that $q_{S_j}(x) = 1$. Then there exists a string $y \in S_j$ such that $y \preceq x$. By construction, we have that $r_S(x) \preceq x$ and that $|r_S(x)| \geq |y|$. Thus,
by (5.2), we have that \( y \preceq r_S(x) \). Thus, there exists \( y \in S_j \) such that \( y \preceq r_S(x) \), which means \( q_s_j(r_S(x)) = 1 \), as required.

Next, suppose that \( q_S_j(r_S(x)) = 1 \). Then, there exists \( y \in S_j \) such that \( y \preceq r_S(x) \). By construction, \( r_S(x) \preceq x \), so by transitivity we have that \( y \preceq x \). Therefore, \( q_S_j(x) = 1 \), as required.

Given this lemma, we can replace every row \( x_i \) of \( D \) with \( x_i^S = r_S(x_i) \) to obtain a new dataset \( D^S \) such that for every \( j = 1, \ldots, k \),

\[
q_s_j(D^S) = \frac{1}{n} \sum_{i=1}^{n} q_s_j(x_i^S) = \frac{1}{n} \sum_{i=1}^{n} q_s_j(x_i) = q_s_j(D).
\]

Thus, we can answer \( q_S_1, \ldots, q_S_k \) on \( D^S \in X^n_S \), rather than on \( D \in X^n \). Note that each row of \( D^S \) depends only on the corresponding row of \( D \). Hence, for every set of queries \( q_S_1, \ldots, q_S_k \), if \( D \sim D' \) are adjacent datasets, then \( D^S \sim D'^S \) are also adjacent datasets. Consequently, applying a \((\varepsilon, \delta)\)-differentially private algorithm to \( D^S \) yields a \((\varepsilon, \delta)\)-differentially private algorithm as a function of \( D \).

In particular, we can give \( \alpha \)-accurate answers to these queries using the algorithm \( M_{BLR} \) as long as

\[
k \leq 2^{\Omega(\alpha^3 \varepsilon n / \log |X_S|)} = 2^{\Omega(\alpha^3 \varepsilon n / \log (kB+1))}.
\]

Rearranging terms gives the bound in Theorem 5.4.2. We specify the complete algorithm \( M_{\text{prefix}} \) in Figure 5.4.

\begin{center}
\begin{tabular}{l}
\( M_{\text{prefix}}(D; q_{S_1}, \ldots, q_{S_k}) \):
Write \( D = (x_1, \ldots, x_n) \in X^n \), \( S = \bigcup_{j=1}^{k} S_j \), \( X_S = S \cup \emptyset \).
For \( i = 1, \ldots, n \), let \( x_i^S = r_S(x_i) \) and let \( D^S = (x_1^S, \ldots, x_n^S) \in X^n_S \).
Let \( (a_1, \ldots, a_k) = M_{BLR}(D^S; q_{S_1}, \ldots, q_{S_k}) \).
Output \( (a_1, \ldots, a_k) \).
\end{tabular}
\end{center}

\textbf{Figure 5.4:} \( M_{\text{prefix}} \)

\[ \square \]
5.4.2 A Lower Bound for Online Prefix Queries

Next, we prove a lower bound for online queries. Our lower bound shows that the simple approach of perturbing the answer to each query with independent noise is essentially optimal for prefix queries. Since this approach is only able to answer \( k = O(n^2) \) queries, we obtain an exponential separation between online and offline statistical queries for a broad range of parameters.

**Theorem 5.4.5** (Lower Bound for Online Prefix Queries). There exists a function \( k = O(n^2) \) such that for every sufficiently large \( n \in \mathbb{N} \), there is no \((1, 1/30n)\)-differentially private algorithm \( M \) that takes a dataset \( D \in \mathcal{X}^n \) and is \((1/100, 1/100)\)-accurate for \( k \) online queries from \( \Omega^*_{\text{prefix}} \).

In this parameter regime, our algorithm from Section 5.4.1 answers \( k = \exp(\Omega(\sqrt{n})) \) offline prefix queries, so we obtain an exponential separation.

Our lower bound relies on a connection between fingerprinting codes and differential privacy [29, 56, 111, 115]. However, instead of using fingerprinting codes in a black-box way, we will make a direct use of the main techniques. Specifically, we will rely heavily on the following lemma, which is a simpler version of Lemma 2.6.5 from Section 2.6.

**Lemma 5.4.6** (Fingerprinting Lemma [56, 111]). Let \( f : \{\pm1\}^n \rightarrow [-1, 1] \) be any function. Suppose \( p \) is sampled from the uniform distribution over \([-1, 1]\) and \( c \in \{\pm1\}^n \) is a vector of \( n \) independent bits, where each bit has expectation \( p \). Letting \( \bar{c} \) denote the coordinate-wise mean of \( c \), we have

\[
\mathbb{E}_{p,c} \left[ f(c) \cdot \sum_{i \in [n]} (c_i - p) + 2|f(c) - \bar{c}| \right] \geq \frac{1}{3}.
\]

Roughly the fingerprinting lemma says that if we sample a vector \( c \in \{\pm1\}^n \) in a specific fashion, then for any bounded function \( f(c) \), we either have that \( f(c) \) has “significant” correlation with \( c_i \) for some coordinate \( i \), or that \( f(c) \) is “far” from \( \bar{c} \) on average.

In our lower bound, the vector \( c \) will represent a column of the dataset, so each coordinate \( c_i \) will correspond to the value of some row of the dataset. The function \( f(c) \) will represent the answer to some prefix query. We will use the accuracy of a mechanism for answering prefix queries to argue that \( f(c) \) is not far from \( \bar{c} \), and therefore conclude that \( f(c) \) must be significantly correlated with some coordinate \( c_i \). On the other hand, if \( c_i \) were excluded from the dataset, then \( c_i \) is sufficiently random that the mechanism’s answers cannot be significantly correlated with \( c_i \). We will use this to
derive a contradiction to differential privacy.

Proof of Theorem 5.4.5. First we define the distribution on the input dataset \( D = (x_1, \ldots, x_n) \) and the queries \( q_{S_1}, \ldots, q_{S_k} \).

Input dataset \( D \):

- Sample \( p^1, \ldots, p^k \in [-1, 1] \) independently and uniformly at random.
- Sample \( c^1, \ldots, c^k \in \{\pm 1\}^n \) independently, where each \( c^j \) is a vector of \( n \) independent bits, each with expectation \( p^j \).
- For \( i \in [n] \), define
  \[
  x_i = (\text{binary}(i), c^1_i, \ldots, c^k_i) \in \{\pm 1\}^{\lceil \log_2 n \rceil + k},
  \]
  where \( \text{binary}(i) \in \{\pm 1\}^{\lceil \log_2 n \rceil} \) is the binary representation of \( i \) where 1 is mapped to +1 and 0 is mapped to −1.\(^5\) Let \( D = (x_1, \ldots, x_n) \in (\{\pm 1\}^{\lceil \log_2 n \rceil + k})^n \).

Queries \( q_{S_1}, \ldots, q_{S_k} \):

- For \( i \in [n] \) and \( j \in [k] \), define
  \[
  z_{i,j} = (\text{binary}(i), c^1_i, \ldots, c^{j-1}_i, 1) \in \{\pm 1\}^{\lceil \log_2 n \rceil + j}.
  \]
- For \( j \in [k] \), define \( q_{S_j} \in Q_{\text{prefix}}^n \) by \( S_j = \{z_{i,j} \mid i \in [n]\} \).

These queries are designed so that the correct answer to each query \( j \in [k] \) is given by \( q_{S_j}(D) = \overline{c}^j \):

Claim 5.4.7. For every \( j \in [k] \), if the dataset \( D \) and the queries \( q_{S_1}, \ldots, q_{S_k} \) are constructed as above, then with probability 1,
\[
q_{S_j}(D) = \frac{1}{n} \sum_{i=1}^{n} q_{S_j}(x_i) = \frac{1}{n} \sum_{i=1}^{n} c^j_i = \overline{c}^j
\]

Proof of Claim 5.4.7. We have
\[
q_{S_j}(x_i) = 1 \iff \exists w \in S_j (w \preceq x_i) \iff \exists \ell \in [n] (z_{\ell,j} \preceq x_i).
\]

\(^5\)This choice is arbitrary, and is immaterial to our lower bound. The only property we need is that \( \text{binary}(i) \) uniquely identifies \( i \) and, for notational consistency, we require \( \text{binary}(i) \) to be a string over the alphabet \( \{\pm 1\} \).
By construction, we have \( z_{\ell,j} \preceq x_i \) if and only if \( \ell = i \) and \( x_i^j = c_i^j = 1 \), as required. Here, we have used the fact that the strings \( \text{binary}(i) \) are unique to ensure that \( z_{\ell,j} \preceq x_i \) if and only if \( \ell = i \).

We now show no differentially private algorithm \( M \) is capable of giving accurate answers to these queries. Let \( M \) be an algorithm that answers \( k \) online queries from \( Q^n_{\text{prefix}} \). Suppose we generate an input dataset \( D \) and queries \( q_{S_1}, \ldots, q_{S_k} \) as above, and run \( M(D) \) on this sequence of queries. Let \( a^1, \ldots, a^k \in [-1,1] \) denote the answers given by \( M \).

First, we claim that, if \( M(D) \) is accurate for the given queries, then each answer \( a^j \) is close to the corresponding value \( \bar{c}^j = \frac{1}{n} \sum_{i=1}^{n} c_i^j \).

**Claim 5.4.8.** If \( M \) is \((1/100, 1/100)\)-accurate for \( k \) online queries from \( Q^n_{\text{prefix}} \), then with probability 1 over the choice of \( D \) and \( q_{S_1}, \ldots, q_{S_k} \) above,

\[
\mathbb{E}_{M} \left[ \sum_{j \in [k]} |a^j - \bar{c}^j| \right] \leq \frac{k}{10}.
\]

**Proof of Claim 5.4.8.** By Claim 5.4.7, for every \( j \in [k] \), \( q_{S_j}(D) = \bar{c}^j \). Since, by assumption, \( M \) is \((1/100, 1/100)\)-accurate for \( k \) online queries from \( Q^n_{\text{prefix}} \), we have that with probability at least \( 99/100 \),

\[
\forall j \in [k] \quad |a^j - q_{S_j}(D)| \leq \frac{1}{100} \quad \implies \quad \forall j \in [k] \quad |a^j - \bar{c}^j| \leq \frac{1}{100}
\]

By linearity of expectation, this case contributes at most \( k/100 \) to the expectation. On the other hand, \( |a^j - q_{S_j}(D)| \leq 2 \), so by linearity of expectation the case where \( M \) is inaccurate contributes at most \( 2k/100 \) to the expectation. This suffices to prove the claim.

The next claim shows how the fingerprinting lemma (Lemma 5.4.6) can be applied to \( M \).

**Claim 5.4.9.**

\[
\mathbb{E}_{p,D,q,M} \left[ \sum_{j \in [k]} \left( a^j \sum_{i \in [n]} (c_i^j - p^j) + 2 |a^j - \bar{c}^j| \right) \right] \geq \frac{k}{3}.
\]

**Proof.** By linearity of expectation, it suffices to show that, for every \( j \in [k] \),

\[
\mathbb{E}_{p,D,q,M} \left[ a^j \sum_{i \in [n]} (c_i^j - p^j) + 2 |a^j - \bar{c}^j| \right] \geq \frac{1}{3}.
\]

Since each column \( c^j \) is generated independently from the columns \( c^1, \ldots, c^{j-1}, c^j \) and \( p^j \) are independent from \( q_{S_1}, \ldots, q_{S_j} \). Thus, at the time \( M \) produces the output \( a^j \), it does not have any
Thus, we have

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

To complete the proof, we show that (5.3) violates the differential privacy guarantee unless $n \geq \Omega(\sqrt{k})$.

To this end, fix any $p^1, \ldots, p^k \in [-1, 1]$, whence $c_1^*, \ldots, c_k^* \in \{-1\}$ are independent bits with $\mathbb{E}[c_j^*] = p_j$. Let $\bar{c}^1, \ldots, \bar{c}^k \in \{-1\}$ be independent bits with $\mathbb{E}[\bar{c}_j^*] = p_j$. The random variables $c_1^*, \ldots, c_k^*$ have the same marginal distribution as $c_1^1, \ldots, \bar{c}_k^k$. However, $c_1^1, \ldots, c_k^k$ are independent from $a_1, \ldots, a_k$, whereas $a_1, \ldots, a_k$ depend on $c_1^*, \ldots, c_k^*$. Consider the quantities

$$Z = \sum_{j \in [k]} a_j^i (c_j^* - p^j) \quad \text{and} \quad \tilde{Z} = \sum_{j \in [k]} a_j^i (\bar{c}_j^* - p^j).$$

Differential privacy implies that $Z$ and $\tilde{Z}$ have similar distributions. Specifically, if $M$ is $(1, 1/30n)$-differentially private, then

$$\mathbb{E}[|Z|] = \int_0^{2k} \Pr[|Z| > z] \, dz \leq \int_0^{2k} \left( e\Pr[|\tilde{Z}| > z] + \frac{1}{30n} \right) \, dz = e \mathbb{E}[|\tilde{Z}|] + \frac{k}{15n},$$

as $|Z|, |\tilde{Z}| \leq 2k$ with probability 1.

Now $\mathbb{E}[|Z|] \geq \mathbb{E}[Z] \geq 2k/15n$, by (5.3). On the other hand, $a_j^i$ is independent from $\bar{c}_j^*$ and

$$\mathbb{E}[\bar{c}_j^* - p^j] = 0,$$ so $\mathbb{E}[\tilde{Z}] = 0$. We now observe that

$$\mathbb{E}[\tilde{Z}]^2 \leq \mathbb{E}[\tilde{Z}^2] = \sum_{j \in [k]} \mathbb{Var}[a_j^i (\bar{c}_j^* - p^j)] \leq \sum_{j \in [k]} \mathbb{E}[\var{\bar{c}_j^* - p^j}] \leq k.$$

Thus, we have

$$\frac{2k}{15n} \leq \mathbb{E}[|Z|] \leq e \mathbb{E}[|\tilde{Z}|] + \frac{k}{15n} \leq e\sqrt{k} + \frac{k}{15n}.$$
The condition $2k/15n \leq e\sqrt{k} + k/15n$ is a contradiction unless $k \leq 225e^2n^2$. Thus, we can conclude that there exists a $k = O(n^2)$ such that no $(1, 1/30n)$-differentially private algorithm is accurate for more than $k$ online queries from $Q_{\text{prefix}}^n$, as desired. This completes the proof.

\section{5.5 Separating Adaptive and Non-Adaptive Online Queries}

In this section we prove that even among online queries, answering adaptively-chosen queries can be strictly harder than answering non-adaptively-chosen queries. Our separation applies to a family of search queries that we call \textit{correlated vector queries}. We show that for a certain regime of parameters, it is possible to take a dataset of size $n$ and privately answer an exponential number of fixed correlated vector queries, even if the queries are presented online, but it is impossible to answer more than a constant number of adaptively-chosen correlated vector queries under differential privacy.

The queries are defined on datasets $D \in \{\pm 1\}^n$ (hence the data universe is $\mathcal{X} = \{\pm 1\}$). For every query, the range $R = \{\pm 1\}^n$ is the set of $n$-bit vectors. We fix some parameters $0 < \alpha < 1$ and $m \in \mathbb{N}$. A query $q$ is specified by a set $V$ where $V = \{v^1, \ldots, v^m\} \subseteq \{\pm 1\}^n$ is a set of $n$-bit vectors. Roughly, an accurate answer to a given search query is any vector $y \in \{\pm 1\}^n$ that is approximately $\alpha$-correlated with the input dataset $x \in \{\pm 1\}^n$ and has nearly as little correlation as possible with every $v^j$. By “as little correlation as possible with $v^j$” we mean that $v^j$ may itself be correlated with $x$, in which case $y$ should be correlated with $v^j$ only insofar as this correlation comes through the correlation between $y$ and $x$. Formally, for a query $q_V$, we define the loss function $L_{q_V} : \mathcal{X}^n \times \mathcal{X}^n \to \{0, 1\}$ by

$$L_{q_V}(x, y) = 0 \iff \left| \langle y - \alpha x, x \rangle \right| \leq \frac{\alpha^2 n}{100} \quad \land \quad \forall v^j \in V \left| \langle y - \alpha x, v^j \rangle \right| \leq \frac{\alpha^2 n}{100}.$$ 

We remark that the choice of $\alpha^2 n/100$ is somewhat arbitrary, and we can replace this choice with $C$ for any $\sqrt{n} \ll C \ll n$ and obtain quantitatively different results. We chose to fix this particular choice in order to reduce notational clutter. We let

$$Q_{\text{corr}}^{n,\alpha,m} = \{q_V \mid V \subseteq \{\pm 1\}^n, |V| \leq m\}$$

be the set of all correlated vector queries on $\{\pm 1\}^n$ for parameters $\alpha, m$. 

128
5.5.1 Answering Online Correlated Vector Queries

Provided that all the queries are fixed in advance, we can privately answer correlated vector queries using the randomized response algorithm. This algorithm simply takes the input vector \( x \in \{\pm 1\}^n \) and outputs a new vector \( y \in \{\pm 1\}^n \) where each bit \( y_i \) is independent and is set to \( x_i \) with probability \( 1/2 + \rho \) for a suitable choice of \( \rho > 0 \). The algorithm will then answer every correlated vector query with this same vector \( y \). The following theorem captures the parameters that this mechanism achieves.

**Theorem 5.5.1** (Answering Online Correlated Vector Queries). For every \( 0 < \alpha < 1/2 \), there exists \( k = 2^{\Omega(\alpha^4 n)} \) such that, for every sufficiently large \( n \in \mathbb{N} \), there is a \((3\alpha, 0)\)-differentially private algorithm \( M_{\text{corr}} \) that takes a dataset \( x \in \{\pm 1\}^n \) and is \((1/k)\)-accurate for \( k \) online queries from \( Q_{\text{corr}}^{n, \alpha, k} \).

**Proof Theorem 5.5.1.** Our algorithm based on randomized response is presented in Figure 5.5 below.

![Figure 5.5: M_{\text{corr}}](image)

The algorithm \( M_{\text{corr}} \):

- **Input:** a dataset \( x \in \{\pm 1\}^n \).
- **Parameters:** \( 0 < \alpha < 1/2 \).
- For \( i = 1, \ldots, n \), independently set
  \[
y_i = \begin{cases} 
  +x_i & \text{with probability } \frac{1+\alpha}{2} \\
  -x_i & \text{with probability } \frac{1-\alpha}{2}.
  \end{cases}
\]

Let \( y = (y_1, \ldots, y_n) \in \{\pm 1\}^n \), and answer each query with \( y \).

To establish privacy, observe that by construction each output bit \( y_i \) depends only on \( x_i \) and is independent of all \( x_j, y_j \) for \( j \neq i \). Therefore, it suffices to observe that if \( 0 < \alpha < 1/2 \),

\[
1 \leq \frac{\Pr[y_i = +1 \mid x_i = +1]}{\Pr[y_i = +1 \mid x_i = -1]} = \frac{1+\alpha}{1-\alpha} \leq e^{3\alpha}
\]

and similarly

\[
1 \geq \frac{\Pr[y_i = -1 \mid x_i = +1]}{\Pr[y_i = -1 \mid x_i = -1]} = \frac{1-\alpha}{1+\alpha} \geq e^{-3\alpha}.
\]

To prove accuracy, observe that since the output \( y \) does not depend on the sequence of queries,
we can analyze the mechanism as if the queries \( q_V, \ldots, q_k \in Q_{\text{corr}}^{n, \alpha, k} \) were fixed and given all at once. Let \( V = \bigcup_{j=1}^k V_j \), and note that \( |V| \leq k^2 \). First, observe that \( \mathbb{E}[y] = \alpha x \). Thus we have

\[
\mathbb{E}_y[(y - \alpha x, x)] = 0 \quad \text{and} \quad \forall v \in V \quad \mathbb{E}_y[(y - \alpha x, v)] = 0
\]

Since \( x \) and every vector in \( V \) is fixed independently of \( y \), and the coordinates of \( y \) are independent by construction, the quantities \( \langle y, x \rangle \) and \( \langle y, v \rangle \) are each the sum of \( n \) independent \( \pm 1 \)-valued random variables. Thus, we can apply Hoeffding’s inequality\(^6\) and a union bound to conclude

\[
\Pr_y \left[ \langle y - \alpha x, x \rangle > \frac{\alpha^2 n}{100} \right] \leq 2 \exp \left( -\frac{\alpha^4 n}{20000} \right)
\]

\[
\Pr_y \left[ \exists v \in V \text{ s.t. } \langle y - \alpha x, v \rangle > \frac{\alpha^2 n}{100} \right] \leq 2k^2 \exp \left( -\frac{\alpha^4 n}{20000} \right)
\]

The theorem now follows by setting an appropriate choice of \( k = 2^\Omega(\alpha^4 n) \) such that \( 2(k^2 + 1) \cdot \exp \left( -\frac{\alpha^4 n}{20000} \right) \leq 1/k \).

5.5.2 A Lower Bound for Adaptive Correlated Vector Queries

We now prove a contrasting lower bound showing that if the queries may be chosen adaptively, then no differentially private algorithm can answer more than a constant number of correlated vector queries. The key to our lower bound is that fact that adaptively-chosen correlated vector queries allow an adversary to obtain many vectors \( y^1, \ldots, y^k \) that are correlated with \( x \) but pairwise nearly orthogonal with each other. As we prove, if \( k \) is sufficiently large, this information is enough to recover a vector \( \tilde{x} \) that has much larger correlation with \( x \) than any of the vectors \( y^1, \ldots, y^k \) have with \( x \). By setting the parameters appropriately, we will obtain a contradiction to differential privacy.

**Theorem 5.5.2** (Lower Bound for Correlated Vector Queries). For every \( 0 < \alpha < 1/2 \), there is a \( k = O(1/\alpha^2) \) such that for every sufficiently large \( n \in \mathbb{N} \), there is no \((1, 1/20)\)-differentially private algorithm that takes a dataset \( x \in \{\pm 1\}^n \) and is \( 1/100 \)-accurate for \( k \) adaptive queries from \( Q_{\text{corr}}^{n, \alpha, k} \).

We remark that the value of \( k \) in our lower bound is optimal up to constants, as there is a \((1, 1/20)\)-differentially private algorithm that can answer \( k = \Omega(1/\alpha^2) \) adaptively-chosen queries of

\(^6\)We use the following statement of Hoeffding’s Inequality: if \( Z_1, \ldots, Z_n \) are independent \( \pm 1 \)-valued random variables, and \( Z = \sum_{i=1}^n Z_i \), then

\[
\Pr \left[ \left| Z - \mathbb{E}[Z] \right| > C\sqrt{n} \right] \leq 2e^{-c^2/2}
\]
this sort. The algorithm simply answers each query with an independent invocation of randomized response. Randomized response is $O(\alpha)$-differentially private for each query, and we can invoke the adaptive composition theorem [47,55] to argue differential privacy for $k = \Omega(1/\alpha^2)$-queries.

Before proving Theorem 5.5.2, we state and prove the combinatorial lemma that forms the foundation of our lower bound.

**Lemma 5.5.3 (Reconstruction Lemma).** Fix parameters $0 \leq a, b \leq 1$. Let $x \in \{\pm 1\}^n$ and $y^1, \ldots, y^k \in \{\pm 1\}^n$ be vectors such that

$$\forall 1 \leq j \leq k \quad \langle y^j, x \rangle \geq an$$

$$\forall 1 \leq j < j' \leq k \quad |\langle y^j, y^{j'} \rangle| \leq bn.$$

Then, if we let $\bar{x} = \text{sign}(\sum_{j=1}^k y^j) \in \{\pm 1\}^n$ be the coordinate-wise majority of $y^1, \ldots, y^k$, we have

$$\langle \bar{x}, x \rangle \geq \left( 1 - \frac{2}{a^2 k} - \frac{2(b - a^2)}{a^2} \right)n.$$

**Proof of Lemma 5.5.3.** Let

$$\bar{y} = \frac{1}{k} \sum_{j=1}^k y^j \in [-1,1]^n.$$

By linearity, $\langle \bar{y}, x \rangle \geq an$ and

$$\|\bar{y}\|_2^2 = \frac{1}{k^2} \sum_{j,j'=1}^k \langle y^j, y^{j'} \rangle \leq \frac{1}{k^2} (kn + (k^2 - k)bn) \leq \left( \frac{1}{k} + b \right)n.$$

Define a random variable $W \in [-1,1]$ to be $x_i \bar{y}_i$ for a uniformly random $i \in [n]$. Then

$$E[W] = \frac{1}{n} \langle x, \bar{y} \rangle \geq a \quad \text{and} \quad E[W^2] = \frac{1}{n} \sum_{i=1}^n x_i^2 \bar{y}_i^2 = \frac{1}{n} \|\bar{y}\|_2^2 \leq \frac{1}{k} + b$$

By Chebyshev’s inequality,

$$\Pr[W \leq 0] \leq \Pr \left[ |W - E[W]| \geq a \right] \leq \frac{\text{Var}[W]}{a^2} = \frac{E[W^2] - E[W]^2}{a^2} \leq \frac{1}{k} + b - \frac{a^2}{a^2}.$$

Meanwhile,

$$\Pr[W \leq 0] = \frac{1}{n} \sum_{i=1}^n I[x_i \bar{y}_i \leq 0] \geq \frac{1}{n} \sum_{i=1}^n I[\text{sign}(\bar{y}_i) \neq x_i] = \frac{1}{2} - \frac{1}{2n} \langle \text{sign}(\bar{y}), x \rangle.$$
Thus we conclude

$$\langle \text{sign}(\overline{y}), x \rangle \geq n - 2n \Pr[W \leq 0] \geq n - 2n \left( \frac{\frac{1}{k} + b - a^2}{a^2} \right)$$

To complete the proof, we rearrange terms and note that $\text{sign}(\overline{y}) = \text{sign}(\sum_{j=1}^{k} y^j)$.

Now we are ready to prove our lower bound for algorithms that answer adaptively-chosen correlated vector queries.

**Proof of Theorem 5.5.2.** We will show that the output $y^1, \ldots, y^k$ of any algorithm $M$ that takes a dataset $x \in \{\pm 1\}^n$ and answers $k = 100/\alpha^2$ adaptively-chosen correlated vector queries can be used to find a vector $\tilde{x} \in \{\pm 1\}^n$ such that $\langle \tilde{x}, x \rangle > n/2$. In light of Lemma 5.5.3, this vector will simply be $\tilde{x} = \text{sign}(\sum_{j=1}^{k} y^j)$. We will then invoke the following elementary fact that differentially private algorithms do not admit this sort of reconstruction of their input dataset.

**Fact 5.5.4.** For every sufficiently large $n \in \mathbb{N}$, there is no $(1, 1/20)$-differentially private algorithm $M : \{\pm 1\}^n \rightarrow \{\pm 1\}^n$ such that for every $x \in \{\pm 1\}^n$, with probability at least $99/100$, $\langle M(x), x \rangle > n/2$.

The attack works as follows. For $j = 1, \ldots, k$, define the set $V_j = \{y^1, \ldots, y^{j-1}\}$ and ask the query $q_{V_j}(x) \in Q^{n,\alpha,k}_{\text{corr}}$ to obtain some vector $y^j$. Since $M$ is assumed to be accurate for $k$ adaptively-chosen queries, with probability $99/100$, we obtain vectors $y^1, \ldots, y^k \in \{\pm 1\}^n$ such that

$$\forall 1 \leq j \leq k \quad \langle y^j, x \rangle \geq \langle \alpha x, x \rangle - |\langle y - \alpha x, x \rangle|$$

$$\geq \alpha n - \frac{\alpha^2 n}{100}$$

$$\geq an.$$
where $a = 99\alpha/100$ and $b = 51\alpha^2/50$. Thus, by Lemma 5.5.3, if $\tilde{x} = \text{sign}(\sum_{j=1}^{k} y^j)$, and $k = 100/\alpha^2$, we have

$$\langle \tilde{x}, x \rangle \geq \left( 1 - \frac{2}{a^2 k} - \frac{2(b-a^2)}{a^2} \right) n$$

$$= \left( 1 - \frac{2}{99\alpha/100)^2 k} - \frac{2(51\alpha^2/50 - (99\alpha/100)^2)}{(99\alpha/100)^2} \right) n$$

$$= \left( 1 - \frac{2(100/99)^2}{100} - 2 \frac{(51/50) - (99/100)^2}{(99/100)^2} \right) n$$

$$\geq 0.89n \geq n/2.$$ 

By Fact 5.5.4, this proves that $M$ cannot be $(1, 1/20)$-differentially private. 

## 5.6 Threshold Queries

First we define threshold queries, which are a family of statistical queries.

**Definition 5.6.1.** Let $\text{Thresh}_X$ denote the class of threshold queries over a totally ordered domain $X$. That is, $\text{Thresh}_X = \{c_x : x \in X\}$ where $c_x : X \to \{0, 1\}$ is defined by $c_x(y) = 1$ iff $y \leq x$.

### 5.6.1 Separation for Pure Differential Privacy

In this section, we show that the sample complexity of answering adaptively-chosen thresholds can be exponentially larger than that of answering thresholds offline.

**Proposition 5.6.2 ([31,48,49]).** Let $X$ be any totally ordered domain. Then there exists a $(\varepsilon, 0)$-differentially private mechanism $M$ that, given $D \in X^n$, gives $\alpha$-accurate answers to $k$ offline queries.
from $\text{Thresh}_X$ for

$$n = O \left( \min \left\{ \frac{\log k + \log^2(1/\alpha)}{\alpha \varepsilon}, \frac{\log^2 k}{\alpha \varepsilon} \right\} \right)$$

On the other hand, we show that answering $k$ adaptively-chosen threshold queries can require sample complexity as large as $\Omega(k) – an exponential gap. Note that this matches the upper bound given by the Laplace mechanism [47].

**Proposition 5.6.3.** Answering $k$ adaptively-chosen threshold queries on $[2^{k-1}]$ to accuracy $\alpha$ subject to $\varepsilon$-differential privacy requires sample complexity $n = \Omega(k/\alpha \varepsilon)$.

The idea for the lower bound is that an analyst may adaptively choose $k$ threshold queries to binary search for an “approximate median” of the dataset. However, a packing argument shows that locating an approximate median requires sample complexity $\Omega(k)$.

**Definition 5.6.4** (Approximate Median). Let $X$ be a totally ordered domain, $\alpha > 0$, and $D = (x_1, \ldots, x_n) \in X^n$. We call $y \in X$ an $\alpha$-approximate median of $D$ if

$$\frac{1}{n} |\{i \in [n] : x_i \leq y\}| \geq \frac{1}{2} - \alpha \quad \text{and} \quad \frac{1}{n} |\{i \in [n] : x_i \geq y\}| \geq \frac{1}{2} - \alpha.$$  

Proposition 5.6.3 is obtained by combining Lemmas 5.6.5 and 5.6.6 below.

**Lemma 5.6.5.** Suppose $M$ answers $k = \lceil 1 + \log_T T \rceil$ adaptively-chosen queries from $\text{Thresh}_{[T]}$ with $\varepsilon$-differential privacy and $(\alpha, \beta)$-accuracy. Then there exists an $\varepsilon$-differentially private $M' : [T]^n \rightarrow [T]$ that computes an $\alpha$-approximate median with probability at least $1 - \beta$.

Proof. The algorithm $M'$, formalized in Figure 5.6, uses $M$ to perform a binary search.

<table>
<thead>
<tr>
<th>Input: $D \in X^n$.</th>
</tr>
</thead>
<tbody>
<tr>
<td>$M$ is given $D$.</td>
</tr>
<tr>
<td>Initialize $\ell_1 = 0$, $u_1 = T$, and $j = 1$.</td>
</tr>
<tr>
<td>While $u_j - \ell_j &gt; 1$ repeat:</td>
</tr>
<tr>
<td>Let $m_j = \lceil (u_j + \ell_j)/2 \rceil$.</td>
</tr>
<tr>
<td>Give $M$ the query $c_{m_j} \in \text{Thresh}_{[T]}$ and obtain the answer $a_j \in [0,1]$.</td>
</tr>
<tr>
<td>If $a_j \geq \frac{1}{2}$, set $(\ell_{j+1}, u_{j+1}) = (\ell_j, m_j)$; otherwise set $(\ell_{j+1}, u_{j+1}) = (m_j, u_j)$.</td>
</tr>
<tr>
<td>Increment $j$.</td>
</tr>
<tr>
<td>Output $u_j$.</td>
</tr>
</tbody>
</table>

**Figure 5.6:** $M' : X^n \rightarrow X$
We have $u_1 - \ell_1 = T$ and, after every query $j$, $u_{j+1} - \ell_{j+1} \leq \lceil (u_j - \ell_j)/2 \rceil$. Since the process stops when $u_j - \ell_j = 1$, it is easy to verify that $M'$ makes at most $\lceil 1 + \log_2(T - 1) \rceil$ queries to $M$.

Suppose all of the answers given by $M$ are $\alpha$-accurate. This happens with probability at least $1 - \beta$. We will show that, given this, $M'$ outputs an $\alpha$-approximate median, which completes the proof.

We claim that $c_{u_j}(D) \geq \frac{1}{2} - \alpha$ for all $j$. This is easily shown by induction. The base case is $c_T(D) = 1 \geq \frac{1}{2} - \alpha$. At each step either $u_{j+1} = u_j$ (in which case the induction hypothesis can be applied) or $u_{j+1} = m_j$; in the latter case our accuracy assumption gives

$$c_{u_{j+1}}(D) = c_{m_j}(D) \geq a_j - \alpha \geq \frac{1}{2} - \alpha.$$

We also claim that $c_{\ell_j}(D) < \frac{1}{2} + \alpha$ for all $j$. This follows from a similar induction and completes the proof.

**Lemma 5.6.6.** Let $M : [T]^n \rightarrow [T]$ be an $\varepsilon$-differentially private algorithm that computes an $\alpha$-approximate median with confidence $1 - \beta$. Then

$$n \geq \Omega \left( \log T + \log(1/\beta) \over \alpha \varepsilon \right).$$

**Proof.** Let $m = \lceil (1 - \alpha)n \rceil - 1$. For each $t \in [T]$, let $D^t \in [T]^n$ denote the dataset containing $m$ copies of $1$, $m$ copies of $T$, and $n - 2m$ copies of $t$. Then for each $t \in [T]$,

$$\Pr[M(D^t) = t] \geq 1 - \beta.$$

On the other hand, by the pigeonhole principle, there must exist $t_* \in [T - 1]$ such that

$$\Pr[M(D^T) = t_*) \leq \Pr[M(D^T) \in [T - 1]] \leq \beta \over T - 1.$$

The inputs $D^T$ and $D^{t_*}$ differ in at most $n - 2m \leq 2\alpha n + 2$ entries. By group privacy,

$$1 - \beta \leq \Pr[M(D^{t_*}) = t_*) \leq e^{\varepsilon(2\alpha n + 2)} \Pr[M(D^T) = t_*] \leq e^{\varepsilon(2\alpha n + 2)} \over T - 1.$$

Rearranging these inequalities gives

$$O(\varepsilon \alpha n) \geq e^{\varepsilon(2\alpha n + 2)} \geq \log \left( \frac{(1 - \beta)(T - 1)}{\beta} \right) \geq \Omega(\log(T/\beta)),$$

135
which yields the result.

Remark 5.6.7. Proposition 5.6.3 can be extended to online non-adaptive queries, which yields a separation between the online non-adaptive and offline models for pure differential privacy and threshold queries.

The key observation behind Remark 5.6.7 is that, while Lemma 5.6.5 in general requires making adaptive queries, for the inputs $D^t \in [T]^n$ ($t \in [T]$) used in Lemma 5.6.6 the queries are “predictable.” In particular, on input $D^t$, the algorithm $M'$ from the proof of Lemma 5.6.5 will (with probability at least $1 - \beta$) always make the same sequence queries. This allows the queries to be specified in advance in a non-adaptive manner. More precisely, we can produce an algorithm $M'_t$ that produces non-adaptive online queries by simulating $M'$ on input $D^t$ and using those queries. Given the answers to these online non-adaptive queries, $M'_t$ can either accept or reject its input depending on whether the answers are consistent with the input $D^t$; $M'_t$ will accept $D^t$ with high probability and reject $D^{t'}$ for $t' \neq t$ with high probability. The proof of Lemma 5.6.6 can be carried out using $M'_t$ instead of $M'$ at the end.

5.6.2 The BetweenThresholds Algorithm

The key technical novelty behind our algorithm for answering adaptively-chosen threshold queries is a refinement of the “AboveThreshold” algorithm [54, §3.6], which underlies the ubiquitous “sparse vector” technique [48, 50, 76, 105].

The sparse vector technique addresses a setting where we have a stream of $k$ (adaptively-chosen) low-sensitivity queries and a threshold parameter $t$. Instead of answering all $k$ queries accurately, we are interested in answering only the ones that are above the threshold $t$ – for the remaining queries, we only require a signal that they are below the threshold. Intuitively, one would expect to only pay in privacy for the queries that are actually above the threshold. And indeed, one can get away with sample complexity proportional to the number of queries that are above the threshold, and to the logarithm of the total number of queries.

We extend the sparse vector technique to settings where we demand slightly more information about each query beyond whether it is below a single threshold. In particular, we set two thresholds $t_u < t_l$, and for each query, release a signal as to whether the query is below the lower threshold,
above the upper threshold, or between the two thresholds.

As long as the thresholds are sufficiently far apart, whether (the noisy answer to) a query is below the lower threshold or above the upper threshold is stable, in that it is extremely unlikely to change on neighboring datasets. As a result, we obtain an \((\varepsilon, \delta)\)-differentially private algorithm that achieves the same accuracy guarantees as the traditional sparse vector technique, i.e. sample complexity proportional to \(\log k\).

Our algorithm is summarised by the following theorem.\footnote{In Theorem 5.6.8, only one threshold is allowed. However, our algorithm is more general and permits the setting of two thresholds. We have chosen this statement for simplicity.}

**Theorem 5.6.8.** Let \(\alpha, \beta, \varepsilon, \delta, t \in (0, 1)\) and \(n, k \in \mathbb{N}\) satisfy

\[
\frac{1}{\alpha \varepsilon} \max \{12 \log(30 / \varepsilon \delta), 16 \log((k + 1) / \beta)\}.
\]

Then there exists a \((\varepsilon, \delta)\)-differentially private algorithm that takes as input \(D \in \mathcal{X}^n\) and answers a sequence of adaptively-chosen queries \(q_1, \cdots, q_k : \mathcal{X}^n \to [0, 1]\) of sensitivity \(1/n\) with \(a_1, \cdots, a_{\leq k} \in \{L, R, \top\}\) such that, with probability at least \(1 - \beta\),

- \(a_j = L \implies q_j(D) \leq t\),
- \(a_j = R \implies q_j(D) \geq t\), and
- \(a_j = \top \implies t - \alpha \leq q_j(D) \leq t + \alpha\).

The algorithm may halt before answering all \(k\) queries; however, it only halts after outputting \(\top\).

Our algorithm is given in Figure 5.7. The analysis is split into Lemmas 5.6.9 and 5.6.10.

**Lemma 5.6.9** (Privacy for \(\text{BetweenThresholds}\)). Let \(\varepsilon, \delta \in (0, 1)\) and \(n \in \mathbb{N}\). Then \(\text{BetweenThresholds}\) (Figure 5.7) is \((\varepsilon, \delta)\)-differentially private for any adaptively-chosen sequence of queries as long as the gap between the thresholds \(t_\ell, t_u\) satisfies

\[
t_u - t_\ell \geq \frac{12}{\varepsilon n} (\log(10 / \varepsilon) + \log(1/\delta) + 1).
\]

**Lemma 5.6.10** (Accuracy for \(\text{BetweenThresholds}\)). Let \(\alpha, \beta, \varepsilon, t_\ell, t_u \in (0, 1)\) and \(n, k \in \mathbb{N}\) satisfy

\[
n \geq \frac{8}{\alpha \varepsilon} (\log(k + 1) + \log(1/\beta)).
\]
Input: $D \in \mathcal{X}^n$.
Parameters: $\varepsilon, t_\ell, t_u \in (0, 1)$ and $n, k \in \mathbb{N}$.
Sample $\mu \sim \text{Lap}(2/\varepsilon n)$ and initialize noisy thresholds $\hat{t}_\ell = t_\ell + \mu$ and $\hat{t}_u = t_u - \mu$.

For $j = 1, 2, \ldots, k$:
- Receive query $q_j : \mathcal{X}^n \to [0, 1]$.
- Set $c_j = q_j(D) + \nu_j$ where $\nu_j \sim \text{Lap}(6/\varepsilon n)$.
  - If $c_j < \hat{t}_\ell$, output $L$ and continue.
  - If $c_j > \hat{t}_u$, output $R$ and continue.
  - If $c_j \in [\hat{t}_\ell, \hat{t}_u]$, output $\top$ and halt.

Figure 5.7: BetweenThresholds

Then, for any input $D \in \mathcal{X}^n$ and any adaptively-chosen sequence of queries $q_1, q_2, \ldots, q_k$, the answers $a_1, a_2, \ldots, a_{\leq k}$ produced by BetweenThresholds (Figure 5.7) on input $D$ satisfy the following with probability at least $1 - \beta$. For any $j \in [k]$ such that $a_j$ is returned before BetweenThresholds halts,

- $a_j = L \implies q_j(D) \leq t_\ell + \alpha,$
- $a_j = R \implies q_j(D) \geq t_u - \alpha,$ and
- $a_j = \top \implies t_\ell - \alpha \leq q_j(D) \leq t_u + \alpha.$

Combining Lemmas 5.6.9 and 5.6.10 and setting $t_\ell = t - \alpha/2$ and $t_u = t + \alpha/2$ yields Theorem 5.6.8.

Proof of Lemma 5.6.9. Our analysis is an adaptation of Dwork and Roth’s [54, §3.6] analysis of the AboveThreshold algorithm. Recall that a transcript of the execution of BetweenThresholds is given by $a \in \{L, R, \top\}^*$. Let $\mathcal{M} : \mathcal{X}^n \to \{L, R, \top\}^*$ denote the function that simulates BetweenThresholds interacting with a given adaptive adversary (cf. Figure 5.3) and returns the transcript.

Let $S \subset \{L, R, \top\}^*$ be a set of transcripts. Our goal is to show that for neighboring datasets $D \sim D'$,

$$\Pr[\mathcal{M}(D) \in S] \leq e^\delta \Pr[\mathcal{M}(D') \in S] + \delta.$$ 

Let

$$z^* = \frac{1}{2}(t_u - t_\ell) - \frac{6}{\varepsilon n} \log(10/\varepsilon) - 1/n \geq \frac{2}{\varepsilon n} \log(1/\delta).$$
Our strategy will be to show that as long as the noise value $\mu$ is under control, in particular if $\mu \leq z^*$, then the algorithm behaves in essentially the same way as the standard AboveThreshold algorithm. Meanwhile, the event $\mu > z^*$ which corresponds to the (catastrophic) event where the upper and lower thresholds are too close or overlap, happens with probability at most $\delta$.

The following claim reduces the privacy analysis to examining the probability of obtaining any single transcript $a$:

**Claim 5.6.11.** Suppose that for any transcript $a \in \{L, R, \top\}^*$, and any $z \leq z^*$, that

$$
\Pr [\mathcal{M}(D) = a | \mu = z] \leq e^{\varepsilon/2} \Pr [\mathcal{M}(D') = a | \mu = z + 1/n].
$$

Then $\mathcal{M}$ is $(\varepsilon, \delta)$-differentially private.

**Proof.** By properties of the Laplace distribution, since $\mu \sim \text{Lap}(2/\varepsilon n)$, for any $z \in \mathbb{R}$, we have

$$
\Pr [\mu = z] \leq e^{\varepsilon/2} \Pr [\mu = z + 1/n],
$$

and

$$
\Pr [\mu > z^*] = \frac{1}{2} e^{-\varepsilon z^*/2} \leq \delta.
$$

Fix a set of transcripts $S$. Combining these properties allows us to write

$$
\Pr [\mathcal{M}(D) \in S] = \int_{z^*}^{\infty} \Pr [\mathcal{M}(D) \in S | \mu = z] \Pr [\mu = z] \, dz
$$

$$
\leq \left( \int_{z^*}^{\infty} \Pr [\mathcal{M}(D) \in S | \mu = z] \Pr [\mu = z] \, dz \right) + \Pr [\mu > z^*]
$$

$$
\leq \left( e^{\varepsilon/2} \int_{-\infty}^{z^*} \Pr [\mathcal{M}(D') \in S | \mu = z + 1/n] \Pr [\mu = z + 1/n] \, dz \right) + \delta
$$

$$
\leq e^\varepsilon \int_{-\infty}^{z^*} \Pr [\mathcal{M}(D') \in S | \mu = z + 1/n] \Pr [\mu = z + 1/n] \, dz
$$

$$
\leq e^\varepsilon \Pr [\mathcal{M}(D') \in S] + \delta
$$

Returning to the proof of Lemma 5.6.9, fix a transcript $a \in \{L, R, \top\}^*$. Our goal is now to show
that \( \mathcal{M} \) satisfies the hypotheses of Claim 5.6.11, namely that for any \( z \leq z^* \),

\[
\Pr[\mathcal{M}(D) = a|\mu = z] \leq e^{5/2}\Pr[\mathcal{M}(D') = a|\mu = z + 1/n].
\]  

(5.4)

For some \( k \geq 1 \), we can write the transcript \( a \) as \( (a_1, a_2, \ldots, a_k) \), where \( a_j \in \{L, R\} \) for each \( j < k \), and \( a_k = \top \).

For convenience, let \( A = \mathcal{M}(D) \) and \( A' = \mathcal{M}(D') \). We may decompose

\[
\Pr[\mathcal{M}(D) = a|\mu = z] = \Pr[(\forall j < k, A_j = a_j) \land q_k(D) + \nu_k \in [\hat{t}_\ell, \hat{t}_u]|\mu = z]
\]

\[
= \Pr[(\forall j < k, A_j = a_j)|\mu = z] \cdot \Pr[q_k(D) + \nu_k \in [\hat{t}_\ell, \hat{t}_u]|\mu = z \land (\forall j < k, A_j = a_j)].
\]  

(5.5)

We upper bound each factor on the right-hand side separately.

**Claim 5.6.12.**

\[
\Pr[(\forall i < k, A_i = a_i)|\mu = z] \leq \Pr[(\forall i < k, A'_i = a_i)|\mu = z + 1/n]
\]

*Proof.* For fixed \( z \), let \( A_z(x) \) denote the set of noise vectors \( (\nu_1, \ldots, \nu_{k-1}) \) for which \( (A_1, \ldots, A_{k-1}) = (a_1, \ldots, a_{k-1}) \) when \( \nu = z \). We claim that as long as \( z \leq z^* \), then \( A_z(D) \subseteq A_{z+1/n}(D') \). To argue this, let \( (\nu_1, \ldots, \nu_{k-1}) \in A_z(D) \). Fix an index \( j \in \{1, \ldots, k - 1\} \) and suppose \( a_j = L \). Then \( q_j(D) + \nu_j < t_\ell + z \), but since \( q_j \) has sensitivity \( 1/n \), we also have \( q_j(D') + \nu_j < t_\ell + (z + 1/n) \).

Likewise, if \( a_j = R \), then \( q_j(x) + \nu_j > t_u - z \), so

\[
q_j(D') + \nu_j > t_u - z - 1/n \geq t_\ell + (z + 1/n)
\]

as long as \( z \leq z^* \leq \frac{1}{2}(t_u - t_\ell) - 1/n \). (This ensures that \( \mathcal{M}(D') \) does not output L on the first branch of the “if” statement, and proceeds to output R.)

Since \( A_z(D) \subseteq A_{z+1/n}(D') \), this proves that

\[
\Pr[(\forall i < k, A_i = a_i)|\mu = z] = \Pr[(\nu_1, \ldots, \nu_{k-1}) \in A_z(D)]
\]

\[
\leq \Pr[(\nu_1, \ldots, \nu_{k-1}) \in A_{z+1/n}(D')]
\]

\[
= \Pr[(\forall i < k, A'_i = a_i)|\mu = z + 1/n].
\]

\[
\square
\]

Given Claim 5.6.12, all that is needed to prove (5.4) and, thereby, prove Lemma 5.6.9 is to bound
the second factor in (5.5) — that is, we must only show that

\[
\Pr \left[ q_k(D) + \nu_k \in [\hat{t}, \hat{t}_u] | \mu = z \land (\forall j < k, A_j = a_j) \right] \\
\leq e^{\varepsilon/2} \Pr \left[ q_k(D') + \nu_k \in [\hat{t}, \hat{t}_u] | \mu = z + 1/n \land (\forall j < k, A_j' = a_j) \right].
\]

Let \( \Delta = (q_k(D') - q_k(D)) \in [-1/n, 1/n] \). Then

\[
\Pr \left[ q_k(D) + \nu_k \in [\hat{t}, \hat{t}_u] | \mu = z \land (\forall j < k, A_j = a_j) \right] \\
= \Pr \left[ t + z \leq q_k(D) + \nu_k \leq t_u - z \right] \\
= \Pr \left[ t + z + \Delta \leq q_k(D') + \nu_k \leq t_u - z + \Delta \right] \\
= \Pr \left[ t + (z + 1/n) + (\Delta - 1/n) \leq q_k(D') + \nu_k \leq t_u - (z + 1/n) + (\Delta + 1/n) \right] \\
= \Pr \left[ q_k(D') + \nu_k \in [\hat{t}, \hat{t}_u + \Delta + 1/n] | \mu = z + 1/n \right] \\
\leq e^{\varepsilon/2} \Pr \left[ q_k(D') + \nu_k \in [\hat{t}, \hat{t}_u + \Delta + 1/n] | \mu = z + 1/n \right] \\
= e^{\varepsilon/2} \Pr \left[ q_k(D') + \nu_k \in [\hat{t}, \hat{t}_u] | \mu = z + 1/n \land (\forall j < k, A_j' = a_j) \right]
\]

where the last inequality follows from Claim 5.6.13 below (setting \( \eta = 2/n, \lambda = 6/\varepsilon n, [a, b] = [\hat{t}, \hat{t}_u] \), and \([a', b'] = [\hat{t} + \Delta - 1/n, \hat{t}_u + \Delta + 1/n]\) and the fact that \( z = z^* = \frac{1}{2}(t_u - t) - \frac{6}{\varepsilon n} \log(10/\varepsilon) - 1/n \) implies

\[
b - a = \hat{t}_u - \hat{t} = t_u - t - 2\mu \geq \frac{12}{\varepsilon n} \log \left( \frac{10}{\varepsilon} \right) \geq 2\lambda \log \left( \frac{1}{1 - e^{-\varepsilon/6}} \right)
\]

whenever \( 0 \leq \varepsilon \leq 1 \).

**Claim 5.6.13.** Let \( \nu \sim Lap(\lambda) \) and let \([a, b], [a', b'] \subset \mathbb{R}\) be intervals satisfying \([a, b] \subset [a', b']\). If \( \eta \geq (b' - a') - (b - a) \), then

\[
\Pr \left[ \nu \in [a', b'] \right] \leq \frac{e^{\eta/\lambda}}{1 - e^{-(b-a)/2\lambda}} \cdot \Pr \left[ \nu \in [a, b] \right].
\]

**Proof.** Recall that the probability density function of the Laplace distribution is given by \( f_{\lambda}(x) = \frac{1}{2\lambda} e^{-|x|/\lambda} \). There are four cases to consider: In the first case, \( a < b \leq 0 \). In the second case, \( a < 0 < b \) with \( |a| \leq |b| \). In the third case, \( 0 \leq a < b \). Finally, in the fourth case, \( a < 0 < b \) with \( |a| \geq |b| \). Since the Laplace distribution is symmetric, it suffices to analyze the first two cases.
Case 1: Suppose $a < b \leq 0$. Then

$$
\begin{align*}
\Pr [\nu \in [a', b']] &\leq \Pr [\nu \in [a, b]] + \int_b^{b+\eta} \frac{1}{2\lambda} e^{x/\lambda} dx \\
&= \frac{1}{2} \left( e^{(b+\eta)/\lambda} - e^{a/\lambda} \right) \\
&= \frac{1}{2} \cdot \left( \frac{e^{\eta/\lambda} - e^{(a-b)/\lambda}}{1 - e^{(a-b)/\lambda}} \right) \cdot (e^{b/\lambda} - e^{a/\lambda}) \\
&= \left( \frac{e^{\eta/\lambda} - e^{-(b-a)/\lambda}}{1 - e^{-(b-a)/\lambda}} \right) \cdot \Pr [\nu \in [a, b]].
\end{align*}
$$

Case 2: Suppose $a < 0 < b$ and $|a| \leq |b|$. Note that this implies $b \geq (b-a)/2$. Then

$$
\begin{align*}
\Pr [\nu \in [a', b']] &\leq \Pr [\nu \in [a, b]] + \eta \cdot \frac{1}{2\lambda} e^{a/\lambda} \\
&\leq \Pr [\nu \in [a, b]] \left( 1 + \frac{\eta}{2\lambda} \frac{e^{a/\lambda}}{\Pr [\nu \in [0, b]]} \right) \\
&= \Pr [\nu \in [a, b]] \left( 1 - e^{-b/\lambda} + \frac{\eta}{2\lambda} \frac{e^{a/\lambda}}{1 - e^{-b/\lambda}} \right) \\
&\leq \Pr [\nu \in [a, b]] \frac{1 + \eta/\lambda}{1 - e^{-b/\lambda}} \\
&\leq \Pr [\nu \in [a, b]] \frac{e^{\eta/\lambda}}{1 - e^{-(b-a)/2\lambda}}.
\end{align*}
$$

\qed

Proof of Lemma 5.6.10. We claim that it suffices to show that with probability at least $1 - \beta$ we have

$$
\forall 1 \leq j \leq k \quad |\nu_j| + |\mu| \leq \alpha.
$$

To see this, suppose $|\nu_j| + |\mu| \leq \alpha$ for every $j$. Then, if $a_j = L$, we have

$$
c_j = q_j(D) + \nu_j < \hat{t}_\ell = t_\ell + \mu, \quad \text{whence} \quad q_j(D) < t_\ell + |\mu| + |\nu_j| \leq t_\ell + \alpha.
$$

Similarly, if $a_j = R$, then

$$
c_j = q_j(D) + \nu_j > \hat{t}_u = t_u - \mu, \quad \text{whence} \quad q_j(D) > t_u - (|\mu| + |\nu_j|) \geq t_u - \alpha.
$$

142
Finally, if \( a_j = \top \), then
\[
c_j = q_j(D) + \nu_j \in [\hat{t}_\ell, \hat{t}_u] = [t_\ell + \mu, t_u - \mu], \quad \text{whence} \quad t_\ell - \alpha \leq q_j(D) \leq t_u + \alpha.
\]

We now show that indeed \(|\nu_j| + |\mu| \leq \alpha\) for every \( j \) with high probability. By tail bounds for the Laplace distribution,
\[
\Pr[|\mu| > \alpha/4] = \exp\left(-\frac{\varepsilon \alpha n}{8}\right) \quad \text{and} \quad \Pr[|\nu_j| > 3\alpha/4] = \exp\left(-\frac{\varepsilon \alpha n}{8}\right)
\]
for all \( j \). By a union bound,
\[
\Pr[|\mu| > \alpha/4] \lor \exists j \in [k] \ |\nu_j| > 3\alpha/4 \leq (k + 1) \cdot \exp\left(-\frac{\varepsilon \alpha n}{8}\right) \leq \beta,
\]
as required.

5.6.3 The Online Interior Point Problem

Our algorithm extends the results of Chapter 4 showing how to reduce the problem of privately releasing thresholds to the much simpler interior point problem. By analogy, our algorithm for answering adaptively-chosen thresholds relies on solving multiple instances of an online variant of the interior point problem in parallel. In this section, we present the OIP problem and give an \((\varepsilon, \delta)\)-differentially private solution that can handle \( k \) adaptively-chosen queries with sample complexity \( O(\log k) \). Our OIP algorithm is a direct application of the \textit{BetweenThresholds} algorithm from Section 5.6.2.

\textbf{Definition 5.6.14} (Online Interior Point Problem). An algorithm \( M \) solves the \textit{Online Interior Point (OIP) Problem} for \( k \) queries with confidence \( \beta \) if, when given as input any private dataset \( D = (x_1, \ldots, x_n) \in [0, 1]^n \) and any adaptively-chosen sequence of real numbers \( y_1, \ldots, y_k \in [0, 1] \), with probability at least \( 1 - \beta \) it produces a sequence of answers \( a_1, \ldots, a_k \in \{L, R\} \) such that
\[
\forall j \in \{1, 2, \cdots, k\} \quad y_j < \min_{i \in [n]} x_i \implies a_j = L, \quad y_j \geq \max_{i \in [n]} x_i \implies a_j = R.
\]
(If \( \min_{i \in [n]} x_i \leq y_j < \max_{i \in [n]} x_i \), then \( M \) may output either symbol \( L \) or \( R \).)

\textbf{Proposition 5.6.15}. The algorithm in Figure 5.8 is \((\varepsilon, \delta)\)-differentially private and solves the OIP...
Input: Dataset \( D \in [0,1]^n \).
Initialize a \texttt{BetweenThresholds} instance (Figure 5.7) \( B \) on dataset \( D \) with thresholds \( t_\ell = \frac{1}{3} \), \( t_u = \frac{2}{3} \).
For \( j = 1, 2, \ldots, k \):
Receive query \( y_j \in [0,1] \).
If \( B \) already halted on some query \( q_{y^*} \), output \( L \) if \( y_j < y^* \) and output \( R \) if \( y_j \geq y^* \).
Otherwise, give \( B \) the query \( c_{y_j} \in \text{Thresh}[0,1] \).
If \( B \) returns \( \top \), output \( R \). Otherwise, output the answer produced by \( B \).

\textbf{Figure 5.8: Online Interior Point Algorithm}

\textit{Problem with confidence} \( \beta \) as long as
\[
n \geq \frac{36}{\varepsilon \alpha} (\log(k + 1) + \log(1/\beta) + \log(10/\varepsilon) + \log(1/\delta) + 1).
\]

\textit{Proof.} Privacy follows immediately from Lemma 5.6.9, since Algorithm 5.8 is obtained by post-processing Algorithm 5.7, run using thresholds with a gap of size 1/3.

To argue utility, let \( \alpha = 1/3 \) so that
\[
n \geq \frac{8}{\varepsilon \alpha} (\log(k + 1) + \log(1/\beta)).
\]
By Lemma 5.6.10, with probability at least \( 1 - \beta \), the following events occur:

- If the \texttt{BetweenThresholds} instance \( B \) halts when it is queried on \( c_{y^*} \), then \( \min_{i \in [n]} x_i \leq y^* < \max_{i \in [n]} x_i \).
- If \( B \) has not yet halted and \( y_j < \min_{i \in [n]} x_i \), its answer to \( c_{y_j} \) is \( L \).
- If \( B \) has not yet halted and \( y_j \geq \max_{i \in [n]} x_i \), its answer to \( c_{y_j} \) is \( R \).

Thus, if \( B \) has not yet halted, the answers provided are accurate answers for the OIP Problem. On the other hand, when \( B \) halts, it has successfully identified an “interior point” of the dataset \( D \), i.e. a \( y^* \) such that \( \min_{i \in [n]} x_i \leq y^* < \max_{i \in [n]} x_i \). Thus, for any subsequent query \( y \), we have that
\[
y < \min_{i \in [n]} x_i \implies y < y^*,
\]
so Algorithm 5.8 correctly outputs \( L \). Similarly,
\[
y \geq \max_{i \in [n]} x_i \implies y \geq y^*,
\]
so Algorithm 5.8 correctly outputs $R$ on such a query.

### 5.6.4 Releasing Adaptive Thresholds with Approximate Differential Privacy

We are now ready to state our reduction from releasing thresholds to solving the OIP Problem.

**Theorem 5.6.16.** If there exists an $(\varepsilon, \delta)$-differentially private algorithm solving the OIP problem for $k$ queries with confidence $\alpha \beta / 8$ and sample complexity $n'$, then there is a $(4\varepsilon, (1 + e^\varepsilon)\delta)$-differentially private algorithm for releasing $k$ threshold queries with $(\alpha, \beta)$-accuracy and sample complexity

$$n = \max \left\{ \frac{6n'}{\alpha}, \frac{24 \log^{2.5}(4/\alpha) \cdot \log(2/\beta)}{\alpha \varepsilon} \right\}.$$

Combining this reduction with our algorithm for the OIP Problem (Proposition 5.6.15) yields:

**Corollary 5.6.17.** There is an $(\varepsilon, \delta)$-differentially private algorithm for releasing $k$ adaptively-chosen threshold queries with $(\alpha, \beta)$-accuracy for

$$n = O\left(\frac{\log k + \log^{2.5}(1/\alpha) + \log(1/\beta \varepsilon)}{\alpha \varepsilon}\right).$$

**Proof of Theorem 5.6.16.** Our algorithm and its analysis follow the reduction of Bun et al. [27] for reducing the (offline) query release problem for thresholds to the offline interior point problem.

Let $T$ be an $(\varepsilon, \delta)$-differentially private algorithm solving the OIP Problem with confidence $\alpha \beta / 8$ and sample complexity $n'$. Without loss of generality, we may assume that $T$ is differentially private in “add-or-remove-an-item sense”—i.e. if $D \in [0, 1]^*$ and $D'$ differs from $D$ up to the addition or removal of a row, then for every adversary $B$ and set $S$ of outcomes of the interaction between $B$ and $T$, we have $\Pr\left[\text{Adaptive}_{B_{\varepsilon, \delta}}(D) \in S\right] \leq e^\varepsilon \Pr\left[\text{Adaptive}_{B_{\varepsilon, \delta}}(D') \in S\right] + \delta$. Moreover, $T$ provides accurate answers to the OIP Problem with probability at least $1 - \alpha \beta / 8$ whenever its input is of size at least $n'$. To force an algorithm $T$ to have these properties, we may pad any dataset of size less than $n'$ with an arbitrary fixed element. On the other hand, we may subsample the first $n'$ elements from any dataset with more than this many elements.

Consider the algorithm $\text{AdaptiveThresholds}_T$ in Figures 5.9 and 5.10.
Input: Dataset $D \in [0,1]^n$.
Parameter: $\alpha \in (0,1)$.
Let $(D^{(1)}, \ldots, D^{(M)}) \leftarrow \kappa \text{Partition}(D, \alpha)$.
Initialize an instance of the OIP algorithm $T^{(m)}$ on each chunk $D^{(m)} \in [0,1]^*$, for $m \in [M]$.
For each $j = 1, \ldots, k$:
  - Receive query $c_{y_j} \in \text{Thresh}_{[0,1]}$.
  - Give query $y_j \in [0,1]$ to every OIP instance $T^{(m)}$, receiving answers $a^{(1)}_j, \ldots, a^{(M)}_j \in \{L,R\}$.
  - Return $a_j = \frac{1}{M} \cdot \left| \left\{ m \in [M] : a^{(m)}_j = R \right\} \right|$.

Figure 5.9: AdaptiveThresholds$_T$

Input: Dataset $D \in [0,1]^n$.
Parameter: $\alpha \in (0,1)$.
Output: (Random) partition $(D^{(1)}, \ldots, D^{(M)}) \in ([0,1]^*)^M$ of $D$, where $2/\alpha \leq M < 4/\alpha$.
Let $M = 2^{\lceil \log_2(2/\alpha) \rceil}$.
Sort $D$ in nondecreasing order $x_1 \leq x_2 \leq \ldots \leq x_n$.
For each $0 \leq \ell \leq \log_2 M$ and $s \in \{0,1\}^\ell$, sample $\nu_s \sim \text{Lap}((\log_2 M)/\varepsilon)$ independently.
For each $1 \leq m \leq M - 1$, let $\eta_m = \sum_{s \in P(m)} \nu_s$, where $P(m)$ is the set of all prefixes of the binary representation of $m$.
Let $t_0 = 1, t_1 = \left\lfloor \frac{n}{M} + \eta_1 \right\rfloor, \ldots, t_m = \left\lfloor \frac{m \cdot n}{M} + \eta_m \right\rfloor, \ldots, t_M = n + 1$.
Let $D^{(m)} = (x_{t_{m-1}}, \ldots, x_{t_m-1})$ for all $m \in [M]$.

Figure 5.10: Partition

The proof of Theorem 5.6.16 relies on the following two claims about the Partition subroutine, both of which are implicit in [27, Appendix C] and are based on ideas of Dwork et al. [48]. Claim 5.6.18 shows that for neighboring datasets $D \sim D'$, the behaviors of the Partition subroutine on $D$ and $D'$ are “similar” the following sense: for any fixed partition of $D$, one is roughly as likely (over the randomness of the partition algorithm) to obtain a partition of $D'$ that differs on at most two chunks, where the different chunks themselves differ only up to the addition or removal of a single
item. This will allow us to show that running $M$ parallel copies of the OIP algorithm on the chunks remains roughly $(\varepsilon, \delta)$-differentially private. Claim 5.6.19 shows that, with high probability, each chunk is simultaneously large enough for the corresponding OIP algorithm to succeed, but also small enough so that treating all of the elements in a chunk as if they were the same element still permits us to get $\alpha$-accurate answers to arbitrary threshold queries.

**Claim 5.6.18.** Fix neighboring datasets $D, D' \in [0, 1]^n$. Then there exists a (measurable) bijection $\varphi : \mathbb{R}^{2M} \rightarrow \mathbb{R}^{2M}$ with the following properties:

1. Let $z \in \mathbb{R}^{2M}$ be any noise vector. Let $D^{(1)}, \ldots, D^{(M)}$ denote the partition of $D$ obtained with random noise set to $\nu = z$. Similarly, let $D'^{(1)}, \ldots, D'^{(M)}$ denote the partition of $D'$ obtained under noise $\nu = \varphi(z)$. Then there exist indices $i_1, i_2$ such that: 1) For $i \in \{i_1, i_2\}$, the chunks $D^{(i)}$ and $D'^{(i)}$ differ up to the addition or removal of at most one item and 2) For every index $i \notin \{i_1, i_2\}$, we have $D^{(i)} = D'^{(i)}$.

2. For every noise vector $z \in \mathbb{R}^{2M}$, we have $\Pr [\nu = \varphi(z)] \leq e^{2\varepsilon} \Pr [\nu = z]$.

**Claim 5.6.19.** With probability at least $1 - \beta/2$, we have that $|t_m - m \cdot n/M| \leq \alpha n/24$ for all $m \in [M]$.

**Privacy of Algorithm 5.9.** We first show how to use Claim 5.6.18 to show that Algorithm 5.9 is differentially private. Fix an adversary $A$, and let $B = \text{Adaptive}_{A} \rightarrow \text{AdaptiveThresholds}_{T}$ simulate the interaction between $A$ and Algorithm 5.9. Let $S$ be a subset of the range of $B$. Then, by Property (1) of Claim 5.6.18 and group privacy, we have that for any $z \in \mathbb{R}^{2M}$:

$$\Pr [B(D) \in S|\nu = z] \leq e^{2\varepsilon} \Pr [B(D') \in S|\nu = \varphi(z)] + (1 + e^\varepsilon)\delta.$$
By Property (2) of Claim 5.6.18, we also have \( \Pr[\nu = z] \leq e^{2\varepsilon} \Pr[\nu = \varphi(z)] \) for every \( z \in \mathbb{R}^{2M} \). Therefore,

\[
\Pr[B(D) \in S] = \int_{\mathbb{R}^{2M}} \Pr[B(D) \in S|\nu = z] \cdot \Pr[\nu = z] \, dz \\
\leq \int_{\mathbb{R}^{2M}} (e^{2\varepsilon} \Pr[B(D') \in S|\nu = \varphi(z)] + (1 + e^{\varepsilon}) \delta) \cdot \Pr[\nu = z] \, dz \\
\leq (1 + e^{\varepsilon}) \delta + \int_{\mathbb{R}^{2M}} e^{2\varepsilon} \Pr[B(D') \in S|\nu = \varphi(z)] \cdot e^{2\varepsilon} \Pr[\nu = \varphi(z)] \, dz \\
\leq (1 + e^{\varepsilon}) \delta + e^{4\varepsilon} \Pr[B(D') \in S].
\]

Hence, \( B \) is \( (e^{4\varepsilon}, (1 + e^{\varepsilon}) \delta) \)-differentially private, as claimed.

**Accuracy of Algorithm 5.9.** We now show how to use Claim 5.6.19 to show that Algorithm 5.9 produces \((\alpha, \beta)\)-accurate answers. By a union bound, the following three events occur with probability at least \( 1 - \beta \):

1. For all \( m \in [M] \), \( \frac{m}{M} - \frac{t_m}{n} \leq \frac{\alpha}{6} \).
2. Every chunk \( D^{(m)} \) has size \( |D^{(m)}| = t_m - t_{m-1} \in [\alpha n/6, 2\alpha n/3] \).
3. Every instance of \( T \) succeeds.

Now we need to show that if these three events occur, we can produce \( \alpha \)-accurate answers to every threshold query \( c_{y_1}, \ldots, c_{y_k} \). Write the sorted input dataset as \( x_1 \leq x_2 \leq \ldots \leq x_n \). We consider two cases for the \( j \)th query: As our first case, suppose \( x_n \leq y_j \). Then for every chunk \( D^{(m)} \), we have \( \max\{D^{(m)}\} \leq y_j \). Then the success condition of \( T^{(m)} \) guarantees that \( a_j^{(m)} = R \). Thus, the answer \( a_j = 1 \) is (exactly) accurate for the query \( c_j \).

As our second case, let \( i \) be the smallest index for which \( x_i > y_j \), and suppose the item \( x_i \) is in some chunk \( D^{(m_i)} \). Note that this means that the true answer to the query \( c_{y_j} \) is \( (i - 1)/n \) and that \( t_{m_i-1} \leq i \leq t_{m_i} - 1 \). Then again, for every \( m < m_i \) we have \( \max\{D^{(m)}\} \leq y_j \), so every such \( T^{(m)} \) instance yields \( a_j^{(m)} = R \). Thus,

\[
a_j = \frac{1}{M} \cdot \left| \left\{ m \in [M] : a_j^{(m)} = R \right\} \right| \geq \frac{m_i - 1}{M} \geq \frac{t_{m_i}}{n} - \frac{\alpha}{6} - \frac{\alpha}{2} \geq \frac{(i-1)}{n} - \alpha,
\]

since \( M \geq 2/\alpha \).
On the other hand, for every $m > m_i$, we have $\min\{D^{(m)}\} > y_j$, so every such $T^{(m)}$ instance instead yields $a_j^{(m)} = L$. 

$$a_j \leq \frac{m_i}{M} \leq \frac{t_{m_i}}{n} + \frac{\alpha}{6} \leq \frac{t_{m_i-1} + 2\alpha n/3}{n} + \frac{\alpha}{6} \leq \frac{i}{n} + \frac{2\alpha}{3} + \frac{\alpha}{6} \leq \frac{i - 1}{n} + \alpha,$$

since $n \geq 6/\alpha$. □
Chapter 6

Simultaneous Private Learning of Multiple Concepts

In this chapter, we study the direct-sum problem for differentially private PAC learning. That is, we ask whether the resources needed to privately PAC learn $k$ different concept classes “add up,” or whether they do not need to scale significantly with $k$. In our setting, individual examples are drawn from domain $\mathcal{X}$ and labeled by $k$ unknown concepts $(c_1, \ldots, c_k)$ taken from a concept class $\mathcal{C} = \{c : \mathcal{X} \to \{0, 1\}\}$, i.e., each example is of the form $(x, y_1, \ldots, y_k)$, where $x \in \mathcal{X}$ and $y_i = c_i(x)$. The goal of a multi-learner is to output $k$ hypotheses $(h_1, \ldots, h_k)$ that generalize the input examples while preserving the privacy of individuals.

The direct-sum problem has its roots in complexity theory, and is a basic problem for many algorithmic tasks. It also has implications for the practical use of differential privacy. Consider, for instance, a hospital that collects information about its patients and wishes to use this information for medical research. The hospital records for each patient a collection of attributes such as age, sex, and the results of various diagnostic tests (for each patient, these attributes make up a point $x$ in some domain $\mathcal{X}$) and, for each of $k$ diseases, whether the patient suffers from the disease (the $k$ labels $(y_1, \ldots, y_k)$). Based on this collection of data, the hospital researchers wish to learn good predictors for the $k$ diseases. One option for the researchers is to perform each of the learning tasks on a fresh sample of patients, hence enlarging the number of patient examples needed (i.e. the sample complexity) by a factor of $k$, which can be very costly.
The problem of learning multiple concepts simultaneously (without privacy) has been considered before. Motivated by the problem of bridging computational learning and reasoning, Valiant [118] observed that (without privacy) multiple concepts can be learned from a common dataset in a data efficient manner. In particular, the sample complexity that is necessary and sufficient for performing the $k$ learning tasks is actually fully characterized by the VC dimension of the concept class $C$—it is independent of the number of learning tasks $k$. In this work, we set out to examine if the situation is similar when the learning is performed with differential privacy. Interestingly, we see that with differential privacy the picture is quite different, and in particular, the required number of examples can grow polynomially in $k$.

At first glance, private multi-learning appears to be similar to the query release problem. As we saw in Chapter 2, it is possible to answer an exponential number of counting queries on a dataset [14,76,105]. In particular, it is possible to generate with differential privacy a dataset $\hat{D}$ such that the average value of $c$ on $D$ approximates the average of $c$ on $\hat{D}$ for every $c \in C$ simultaneously. The sample complexity required, i.e., the size of the dataset $D$, to perform this sanitization is only logarithmic in $|C|$. Results of this flavor suggest that we can also learn exponentially many concepts simultaneously. However, we give negative results showing that this is not the case, and that multi-learning can have significantly higher sample complexity than query release.

### 6.1 Results and Techniques

Prior work on privately learning the simple concept classes Point$_X$ (of functions that evaluate to 1 on exactly one point of their domain $X$ and to 0 otherwise) and Thresh$_X$ (of functions that evaluate to 1 on a prefix of the domain $X$ and to 0 otherwise) has demonstrated a rather complex picture, depending on whether learners are proper or improper, and whether learning is performed with pure or approximate differential privacy [5,7,8,27]. We analyze the sample complexity of multi-learning of these simple concept classes, as well as general concept classes. We also consider the class Parity$_d$ of parity functions, but in this case we restrict our attention to examples drawn uniformly at random from $\{0,1\}^d$. We examine both proper and improper PAC and agnostic learning under pure and approximate differential privacy. For ease of reference, we include tables with our results in Section 6.1.1, where we omit the dependency on the privacy and accuracy parameters.
Techniques for private $k$-learning. Composition theorems for differential privacy show that the sample complexity of learning $k$ concepts simultaneously is at most a factor of $k$ larger than the sample complexity of learning one concept (and may be reduced to $\sqrt{k}$ for approximate differential privacy). Unfortunately, privately learning one concept from a concept class $C$ can sometimes be quite costly, requiring much higher sample complexity than VC($C$) which is needed to learn nonprivately. Building on techniques of Beimel, Nissim, and Stemmer [9], we show that the multiplicative dependence on $k$ can always be reduced to the VC-dimension of $C$, at the expense of producing a one-time sanitization of the dataset.

**Theorem 6.1.1** (Informal). Let $C$ be a concept class for which there is pure differentially private sanitizer for $C^\oplus = \{f \oplus g : f, g \in C\}$ with sample complexity $m$. Then there is a pure differentially private agnostic $k$-learner for $C$ with sample complexity $O(m + k \cdot \text{VC}(C))$.

Similarly, if $C^\oplus$ has an approximate differentially private sanitizer with sample complexity $m$, then there is an approximate differentially private agnostic $k$-learner for $C$ with sample complexity $O(m + \sqrt{k} \cdot \text{VC}(C))$.

Recall from Chapter 2 that the best known general-purpose sanitizers require sample complexity $m = O(\text{VC}(C) \log |\mathcal{X}|)$ for pure differential privacy [14] and $m = O(\log |C| \sqrt{\log |\mathcal{X}|})$ for approximate differential privacy [76]. However, for specific concept classes (such as Point$_\mathcal{X}$ and Thresh$_\mathcal{X}$), the sample complexity of sanitization can be much lower.

Applying Theorem 6.1.1 with $k = 1$ immediately shows that the sample complexity of privately sanitizing $C^\oplus$ is at least as large as the sample complexity of learning a single concept from $C$. We do not know of a general relationship in the reverse direction. A negative answer to the following question would imply that the upfront sample complexity cost of $m$ in Theorem 6.1.1 is not significant.

**Open Problem 6.1.2** (Private learning vs. sanitization). Is there a concept class $C$ for which sanitizing $C^\oplus$ (or even $C$ itself) with approximate (resp. pure) differential privacy requires asymptotically larger sample complexity than PAC learning $C$ with approximate (resp. pure) differential privacy?

In the case of approximate differential privacy, the sample complexity of $k$-learning can be even lower than what is achievable with our generic learner. Using stability-based arguments, we show that point functions and parities under the uniform distribution can be PAC $k$-learned with sample complexity $O(\text{VC}(C))$ – independent of the number of concepts $k$ (see Theorems 6.3.12 and 6.3.11).
Lower bounds. In light of the above results, one might hope to be able to reduce the dependence on $k$ further, or to eliminate it entirely (as is possible in the case of non-private learning). We show that this is not possible, even for the simplest of concept classes. In the case of pure differential privacy, a packing argument [5, 61, 78] shows that any non-trivial concept class requires sample complexity $\Omega(k)$ to privately $k$-learn (Theorem 6.5.1). For approximate differential privacy, we use fingerprinting codes [21] to show that unlike points and parities, threshold functions require sample complexity $\tilde{\Omega}(k^{1/3})$ to PAC learn privately (Corollary 6.4.5). Moreover, any non-trivial concept class requires sample complexity $\Omega(\sqrt{k})$ to privately learn in the agnostic model (Theorem 6.4.8). In the case of point functions, this matches the upper bound achievable by our generic learner.

Our lower bounds show that dependences on $\text{poly}(k)$ and $\text{VC}(C)$ are separately necessary to multi-learn a concept class, while our upper bounds require sample complexity at least $\sqrt{k} \cdot \text{VC}(C)$. We leave it as an open question to determine whether a dependence on $\text{poly}(k) \cdot \text{VC}(C)$ is also necessary.

Open Problem 6.1.3 (Multiplicative lower bounds for multi-learning). Is there a family of concept classes $C$ with VC dimension $d = \omega(1)$ that requires sample complexity $\Omega(\text{poly}(k) \cdot d)$ to multi-learn with $(\varepsilon, \delta)$-differential privacy?

We highlight a few of the main takeaways from our results:

A complex answer to the direct sum question. Our upper bounds show that solving $k$ learning problems simultaneously can require substantially lower sample complexity than solving the problems individually. On the other hand, our lower bounds show that a significant dependence on $k$ is generally necessary.

Separation between private PAC and private agnostic learning. Non-privately, the sample complexities of PAC and agnostic learning are of the same order (differing only in the dependency in the accuracy parameters). Beimel et al. [9] showed that this is also the case with differentially private learning (of one concept). Our results on learning point functions show that private PAC and agnostic multi-learning can be substantially different (even for learning up to constant error). In the case of approximate differential privacy, $O(1)$ sample suffice to PAC-learn multiple point functions. However, $\tilde{\Omega}(\sqrt{k})$ samples are needed to learn $k$ points agnostically.
Separation between improper learning with approximate differential privacy and non-private learning. In Chapter 4, we saw that the sample complexity of learning one threshold function with approximate differential privacy exceeds the VC dimension, but only in the case of proper learning. Thus it remains possible that improper learning with approximate differential privacy can match the sample complexity of non-private learning. While we do not address this question directly, we exhibit a separation for multi-learning. In particular, learning $k$ thresholds with approximate differential privacy requires $\tilde{\Omega}(k^{1/3})$ samples, even improperly, while $O(1)$ samples suffices non-privately.

6.1.1 Tables of results

The following tables summarize the results of this work. In the tables below $C$ is a class of concepts (i.e., predicates) defined over domain $X$. Sample complexity upper and lower bounds is given in terms of $|C|$ and $|X|$. Note that for Point$_X$, Thresh$_X$, and Parity$_d$ we have $|C| = \Theta(|X|)$.

Where not explicitly noted, upper bounds hold for the setting of agnostic learning and lower bounds are for the (potentially easier) setting of PAC learning. Similarly, where not explicitly noted, upper bounds are for proper learning and lower bounds are for the (less restrictive) setting of improper learning. For simplicity, these tables hide constant and logarithmic factors, as well as dependencies on the learning and privacy parameters.

Multi-learning with pure differential privacy.

Upper bounds:

<table>
<thead>
<tr>
<th>$C$</th>
<th>PAC learning</th>
<th>Agnostic learning</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>proper</td>
<td>improper</td>
</tr>
<tr>
<td>Point$_X$</td>
<td>$k + \log</td>
<td>C</td>
</tr>
<tr>
<td>Thresh$_X$</td>
<td>$k + \log</td>
<td>C</td>
</tr>
<tr>
<td>General</td>
<td>$\min{k \log</td>
<td>C</td>
</tr>
<tr>
<td>Parity$_d$ (uniform)</td>
<td>$k \log</td>
<td>C</td>
</tr>
</tbody>
</table>

Lower bounds:
<table>
<thead>
<tr>
<th>( \mathcal{C} )</th>
<th>PAC learning (proper and improper)</th>
<th>Agnostic learning (proper and improper)</th>
<th>References</th>
</tr>
</thead>
<tbody>
<tr>
<td>Point( \mathcal{X} )</td>
<td>( k + \log</td>
<td>\mathcal{C}</td>
<td>)</td>
</tr>
<tr>
<td>Thresh( \mathcal{X} )</td>
<td>( k + \log</td>
<td>\mathcal{C}</td>
<td>)</td>
</tr>
<tr>
<td>Parity(_d) (uniform)</td>
<td>( k \log</td>
<td>\mathcal{C}</td>
<td>)</td>
</tr>
</tbody>
</table>

Multi-learning with approximate differential privacy.

**Upper bounds:**

<table>
<thead>
<tr>
<th>( \mathcal{C} )</th>
<th>PAC learning (proper and improper)</th>
<th>Agnostic learning (proper and improper)</th>
<th>References</th>
</tr>
</thead>
<tbody>
<tr>
<td>Point( \mathcal{X} )</td>
<td>( 1 ) (Thm. 6.3.12)</td>
<td>( \sqrt{k} ) (Cor. 6.3.8)</td>
<td></td>
</tr>
<tr>
<td>Thresh( \mathcal{X} )</td>
<td>( 2^{\log^*</td>
<td>\mathcal{X}</td>
<td>} + \sqrt{k} ) (Cor. 6.3.9)</td>
</tr>
<tr>
<td>General ( \mathcal{C} )</td>
<td>( \min \left{ \sqrt{k} \log</td>
<td>\mathcal{C}</td>
<td>, \sqrt{k} \text{VC}(\mathcal{C}) + \log</td>
</tr>
<tr>
<td>Parity(_d) (uniform)</td>
<td>( \log</td>
<td>\mathcal{C}</td>
<td>) (Thm. 6.3.11)</td>
</tr>
</tbody>
</table>

**Lower bounds:**

<table>
<thead>
<tr>
<th>( \mathcal{C} )</th>
<th>PAC learning proper</th>
<th>improper</th>
<th>Agnostic learning proper</th>
<th>improper</th>
<th>References</th>
</tr>
</thead>
<tbody>
<tr>
<td>Point( \mathcal{X} )</td>
<td>( 1 )</td>
<td></td>
<td>( \sqrt{k} )</td>
<td></td>
<td>Cor. 6.4.11</td>
</tr>
<tr>
<td>Thresh( \mathcal{X} )</td>
<td>( \log^*</td>
<td>\mathcal{X}</td>
<td>+ k^{1/3} )</td>
<td>( k^{1/3} )</td>
<td>( \log^*</td>
</tr>
<tr>
<td>Parity(_d) (uniform)</td>
<td>( \log</td>
<td>\mathcal{C}</td>
<td>)</td>
<td></td>
<td>( \sqrt{k} + \log</td>
</tr>
</tbody>
</table>

**6.2 Model and Preliminaries**

We recall and extend standard definitions from learning theory and differential privacy.

**6.2.1 Multi-learners**

**Definition 6.2.1** (Multi-labeled dataset). A *k-labeled* dataset over a domain \( \mathcal{X} \) is a dataset \( S \in (\mathcal{X} \times \{0,1\}^k)^* \). That is, \( S \) contains \( |S| \) elements from \( \mathcal{X} \), each concatenated with \( k \) binary labels.
Let $A : (X \times \{0, 1\}^k)^n \rightarrow (2^X)^k$ be an algorithm that operates on a $k$-labeled dataset and returns $k$ hypotheses. Let $C$ be a concept class over a domain $X$ and let $\mathcal{H}$ be a hypothesis class over $X$. We now give a generalization of the notion of PAC learning [117] to multi-labeled datasets (the standard PAC definition is obtained by setting $k = 1$):

**Definition 6.2.2 (PAC Multi-Learner).** Algorithm $A$ is an $(\alpha, \beta)$-PAC $k$-learner for concept class $C$ using hypothesis class $\mathcal{H}$ with sample complexity $n$ if for every distribution $D$ over $X$ and for every fixture of $(c_1, \ldots, c_k)$ from $C$, given a $k$-labeled dataset as an input $S = ((x_i, c_1(x_i), \ldots, c_k(x_i)))_{i=1}^n$ where each $x_i$ is drawn i.i.d. from $D$, algorithm $A$ outputs $k$ hypotheses $(h_1, \ldots, h_k)$ from $\mathcal{H}$ satisfying

$$\Pr \left[ \max_{1 \leq j \leq k} (\text{err}_D(c_j, h_j)) > \alpha \right] \leq \beta.$$ 

The probability is taken over the random choice of the examples in $S$ according to $D$ and the coin tosses of the learner $A$. If $\mathcal{H} \subseteq C$ then $A$ is called a proper learner; otherwise, it is called an improper learner.

**Definition 6.2.3 (Agnostic PAC Multi-Learner).** Algorithm $A$ is an $(\alpha, \beta)$-PAC agnostic $k$-learner for $C$ using hypothesis class $\mathcal{H}$ and sample complexity $n$ if for every distribution $P$ over $X \times \{0, 1\}^k$, given a $k$-labeled dataset $S = ((x_i, y_{1,i}, \ldots, y_{k,i}))_{i=1}^n$ where each $k$-labeled sample $(x_i, y_{1,i}, \ldots, y_{k,i})$ is drawn i.i.d. from $P$, algorithm $A$ outputs $k$ hypotheses $(h_1, \ldots, h_k)$ from $\mathcal{H}$ satisfying

$$\Pr \left[ \max_{1 \leq j \leq k} \left( \text{err}_P(h_j) - \min_{c \in C} \text{err}_P(c) \right) > \alpha \right] \leq \beta,$$

where $P_j$ is the marginal distribution of $P$ on the examples and the $j^{\text{th}}$ label. The probability is taken over the random choice of the examples in $S$ according to $P$ and the coin tosses of the learner $A$. If $\mathcal{H} \subseteq C$ then $A$ is called a proper learner; otherwise, it is called an improper learner.

### 6.2.2 The Sample Complexity of Multi-Learning

Without privacy considerations, the sample complexities of PAC and agnostic learning are essentially characterized by VC dimension. We state these characterizations in the context of multi-learning.
The Vapnik-Chervonenkis Dimension

Classical results in statistical learning theory show that the generalization error of a hypothesis $H$ and its empirical error (observed on a large enough sample) are similar.

**Definition 6.2.4 (Empirical Error).** Let $S = ((x_i, y_i))_{i=1}^n \in (X \times \{0, 1\})^n$ be a labeled sample from $X$.

The empirical error of a hypothesis $h : X \rightarrow \{0, 1\}$ w.r.t. $S$ is defined as $\text{err}_S(h) = \frac{1}{n} |\{i : h(x_i) \neq y_i\}|$.

Let $D \in X^n$ be a (unlabeled) sample from $X$ and let $c : x \rightarrow \{0, 1\}$ be a concept. The empirical error of hypothesis $h : X \rightarrow \{0, 1\}$ w.r.t. $c$ and $D$ is defined as $\text{err}_D(c, h) = \frac{1}{n} |\{i : h(x_i) \neq c(x_i)\}|$.

**Theorem 6.2.5 (VC-Dimension Generalization Bound, e.g. [15]).** Let $D$ and $C$ be, respectively, a distribution and a concept class over a domain $X$, and let $c \in C$. For a sample $S = ((x_i, c(x_i)))_{i=1}^n$ containing $n \geq \frac{64}{\alpha} \left(\text{VC}(C) \ln(\frac{64}{\alpha}) + \ln(\frac{4}{\beta})\right)$ and the $x_i$ are drawn i.i.d. from $D$, it holds that

$$\Pr \left[ \exists h \in C \text{ s.t. } \text{err}_D(h, c) > \alpha \land \text{err}_S(h) \leq \frac{\alpha}{2} \right] \leq \beta.$$ 

This generalization argument extends to the setting of agnostic learning, where a hypothesis with small empirical error might not exist.

**Theorem 6.2.6 (VC-Dimension Agnostic Generalization Bound, e.g. [1, 2]).** Let $H$ be a concept class over a domain $X$, and let $P$ be a distribution over $X \times \{0, 1\}$. For a sample $S = ((x_i, y_i))_{i=1}^n$ containing $n \geq \frac{64}{\alpha^2} \left(\text{VC}(H) \ln(\frac{64}{\alpha}) + \ln(\frac{4}{\beta})\right)$ i.i.d. elements from $P$, it holds that

$$\Pr \left[ \exists h \in H \text{ s.t. } |\text{err}_P(h) - \text{err}_S(h)| > \alpha \right] \leq \beta.$$ 

Using theorems 6.2.9 and 6.2.6, an upper bound of $O(\text{VC}(C))$ on the sample complexity of learning a concept class $C$ follows by reduction to the empirical learning problem. The goal of empirical learning is similar to that of PAC learning, except accuracy is measured only with respect to a fixed input dataset. Theorems 6.2.9 and 6.2.6 state that when an empirical learner is run on sufficiently many samples, it is also accurate with respect to a distribution on inputs.

**Definition 6.2.7 (Empirical Learner).** Algorithm $A$ is an $(\alpha, \beta)$-accurate empirical $k$-learner for a concept class $C$ using hypothesis class $H$ with sample complexity $n$ if for every collection of concepts $(c_1, \ldots, c_k)$ from $C$ and dataset $S = ((x_i, c_1(x_i), \ldots, c_k(x_i)))_{i=1}^n \in (X \times \{0, 1\}^k)^n$, algorithm
\( \mathcal{A} \) outputs \( k \) hypotheses \((h_1, \ldots, h_k)\) from \( \mathcal{H} \) satisfying
\[
\Pr \left[ \max_{1 \leq j \leq k} \left( \text{err}_{S|_j}(h_j) \right) > \alpha \right] \leq \beta,
\]
where \( S|_j = ((x_i, c_j(x_i)))_{i=1}^n \). The probability is taken over the coin tosses of \( \mathcal{A} \).

**Definition 6.2.8 (Agnostic Empirical Learner).** Algorithm \( \mathcal{A} \) is an agnostic \((\alpha, \beta)\)-accurate empirical \( k \)-learner for a concept class \( \mathcal{C} \) using hypothesis class \( \mathcal{H} \) with sample complexity \( n \) if for every dataset \( S = ((x_i, y_{1,i}, \ldots, y_{k,i}))_{i=1}^n \in (\mathcal{X} \times \{0, 1\})^n \), algorithm \( \mathcal{A} \) outputs \( k \) hypotheses \((h_1, \ldots, h_k)\) from \( \mathcal{H} \) satisfying
\[
\Pr \left[ \max_{1 \leq j \leq k} \left( \text{err}_{S|_j}(h_j) - \min_{c \in \mathcal{C}} \left( \text{err}_{S|_j}(c) \right) \right) > \alpha \right] \leq \beta,
\]
where \( S|_j = ((x_i, y_{j,i}))_{i=1}^n \). The probability is taken over the coin tosses of \( \mathcal{A} \).

**Theorem 6.2.9.** Let \( \mathcal{A} \) be an \((\alpha, \beta)\)-accurate empirical \( k \)-learner for a concept class \( \mathcal{C} \) (resp. agnostic empirical \( k \)-learner) using hypothesis class \( \mathcal{H} \). Then \( \mathcal{A} \) is also a \((2\alpha, \beta + \beta')\)-accurate PAC learner for \( \mathcal{C} \) when given at least \( \max\{n, \frac{32}{\alpha}(\text{VC} (\mathcal{H} \oplus \mathcal{C}) \log(32/\alpha) + \log(8/\beta')) \} \) samples (resp. \( \max\{n, \frac{64}{\alpha^2}(\text{VC} (\mathcal{H}) \log(6/\alpha) + \log(8k/\beta')) \} \) samples). Here, \( \mathcal{H} \oplus \mathcal{C} = \{h \oplus c : h \in \mathcal{H}, c \in \mathcal{C} \} \).

**Proof.** We begin with the non-agnostic case. Let \( \mathcal{A} \) be an \((\alpha, \beta)\)-accurate empirical \( k \)-learner for \( \mathcal{C} \). Let \( \mathcal{D} \) be a distribution over the example space \( \mathcal{X} \). Let \( S \) be a random i.i.d. sample of size \( m \) from \( \mathcal{D} \). The generalization bound for PAC learning (Theorem 6.2.5) states that if \( m \geq \frac{32}{\alpha} (d \log(32/\alpha) + \log(8/\beta')) \), then
\[
\Pr[\exists c \in \mathcal{C}, h \in \mathcal{H} : \text{err}_S(c, h) \leq \alpha \land \text{err}_D(c, h) > 2\alpha] \leq \beta',
\]
where \( d = \text{VC} (\mathcal{H} \oplus \mathcal{C}) \). The result follows by a union bound over the failure probability of \( \mathcal{A} \) and the failure of generalization.

Now we turn to the agnostic case. Let \( \mathcal{A} \) be an agnostic \((\alpha, \beta)\)-accurate empirical \( k \)-learner for \( \mathcal{C} \). Fix an index \( j \in [k] \), and let \( \mathcal{P}_j \) be a distribution over \( \mathcal{X} \times \{0, 1\} \). Let \( S \) be a random i.i.d. sample of size \( m \) from \( \mathcal{P}_j \). Then generalization for agnostic learning (Theorem 6.2.6) yields
\[
\Pr[\exists h \in \mathcal{H} : |\text{err}_S(h) - \text{err}_{\mathcal{P}_j}(h)| > \alpha] \leq \frac{\beta'}{k},
\]
for \( m \geq \frac{64}{\alpha^2}(\text{VC} (\mathcal{H}) \log(6/\alpha) + \log(8k/\beta')) \). The result follows by a union bound over the failure probability of \( \mathcal{A} \) and the failure of generalization for each of the indices \( j = 1, \ldots, k \). 

\[\Box\]
Applying the above theorem in the special case where \( A \) finds the concept \( c \in \mathcal{C} \) that minimizes the empirical error on its given sample, we obtain the following sample complexity upper bound for proper multi-learning.

**Corollary 6.2.10.** Let \( \mathcal{C} \) be a concept class with VC dimension \( d \). There exists an \((\alpha, \beta)\)-accurate proper PAC \( k \)-learner for \( \mathcal{C} \) using \( O\left(\frac{1}{\alpha^2}(d\log(1/\alpha) + \log(1/\beta))\right) \) samples. Moreover, there exists an \((\alpha, \beta)\)-accurate proper agnostic PAC \( k \)-learner for \( \mathcal{C} \) using \( O\left(\frac{1}{\alpha^2}(d\log(1/\alpha) + \log(k/\beta))\right) \) samples.

**Proof.** For the non-agnostic case, we simply let \( A \) be the \((0,0)\)-accurate empirical learner that outputs any vector of hypotheses that is consistent with its given examples (one is guaranteed to exist, since the target concept satisfies this condition). The claim follows from Theorem 6.2.9 noting that \( \text{VC}(\mathcal{C} \oplus \mathcal{C}) = O(\text{VC}(\mathcal{C})) \).

For the agnostic case, consider the algorithm \( A \) that on input \( S \) outputs hypotheses \((h_1, \ldots, h_k)\) that minimize the quantities \( \text{err}_S(h_j) \). Applying the agnostic generalization bound [1], this is an \((\alpha/2, \beta/2)\)-accurate agnostic empirical learner given \( O\left(\frac{1}{\alpha^2}(d\log(1/\alpha) + \log(k/\beta))\right) \) samples. The claim then follows from Theorem 6.2.9. \( \Box \)

It is known that even for \( k = 1 \), the sample complexities of PAC and agnostic learning are at least \( \Omega(\text{VC}(\mathcal{C})/\alpha) \) and \( \Omega(\text{VC}(\mathcal{C})/\alpha^2) \), respectively. Therefore, the above sample complexity upper bound is tight up to logarithmic factors.

We recall a few specific concept classes which will play an important role in this work.

- **Point\( _X \):** Let \( \mathcal{X} \) be any domain. The class of **point functions** is the set of all concepts that evaluate to 1 on exactly one element of \( \mathcal{X} \), i.e. \( \text{Point}_X = \{ c_x : x \in \mathcal{X} \} \) where \( c_x(y) = 1 \) iff \( y = x \). The VC dimension of Point\( _X \) is 1 for any \( \mathcal{X} \).

- **Thresh\( _X \):** Let \( \mathcal{X} \) be any totally ordered domain. The class of **threshold functions** takes the form \( \text{Thresh}_X = \{ c_x : x \in \mathcal{X} \} \) where \( c_x(y) = 1 \) iff \( y \leq x \). The VC dimension of Thresh\( _X \) is 1 for any \( \mathcal{X} \).

- **Parity\( _d \):** Let \( \mathcal{X} = \{0, 1\}^d \). The class of **parity functions** on \( \mathcal{X} \) is given by \( \text{Parity}_d = \{ c_x : x \in \mathcal{X} \} \) where \( c_x(y) = \langle x, y \rangle \mod 2 \). The VC dimension of Parity\( _d \) is \( d \).
In this work, we focus our study of the concept class \( \text{Parity}_d \) on the problem of learning parities under the uniform distribution. The PAC and agnostic learning problems are defined as before, except we only require a learner to be accurate when the marginal distribution on examples is the uniform distribution \( U_d \) over \( \{0, 1\}^d \).

**Definition 6.2.11 (PAC Learning \( \text{Parity}_d \) under Uniform).** Algorithm \( A \) is an \((\alpha, \beta)\)-PAC \( k \)-learner for \( \text{Parity}_d \) using hypothesis class \( \mathcal{H} \) and sample complexity \( n \) if for every fixed \((c_1, \ldots, c_k)\) from \( \mathcal{C} \), given a \( k \)-labeled dataset as an input \( S = (\{(x_i, c_1(x_i), \ldots, c_k(x_i))\})_{i=1}^n \) where each \( x_i \) is drawn i.i.d. from \( U_d \), algorithm \( A \) outputs \( k \) hypotheses \((h_1, \ldots, h_k)\) from \( \mathcal{H} \) satisfying
\[
\Pr \left[ \max_{1 \leq j \leq k} \left( \text{err}_{U_d}(c_j, h_j) \right) > \alpha \right] \leq \beta.
\]

**Definition 6.2.12 (Agnostically Learning \( \text{Parity}_d \) under Uniform).** Algorithm \( A \) is an \((\alpha, \beta)\)-PAC agnostic \( k \)-learner for \( \text{Parity}_d \) using hypothesis class \( \mathcal{H} \) and sample complexity \( n \) if for every distribution \( \mathcal{P} \) over \( \{0, 1\}^d \times \{0, 1\}^k \), with marginal distribution \( U_d \) over the data universe \( \{0, 1\}^d \), given a \( k \)-labeled dataset \( S = (\{(x_i, y_{1,i}, \ldots, y_{k,i})\})_{i=1}^n \) where each \( k \)-labeled sample \((x_i, y_{1,i}, \ldots, y_{k,i})\) is drawn i.i.d. from \( \mathcal{P} \), algorithm \( A \) outputs \( k \) hypotheses \((h_1, \ldots, h_k)\) from \( \mathcal{H} \) satisfying
\[
\Pr \left[ \max_{1 \leq j \leq k} \left( \text{err}_{\mathcal{P}}(h_j) - \min_{c \in \mathcal{C}} \left( \text{err}_{\mathcal{P}}(c) \right) \right) > \alpha \right] \leq \beta,
\]
where \( \mathcal{P}_j \) is the marginal distribution of \( \mathcal{P} \) on the examples and the \( j \)th label.

### 6.2.3 Improved Private Sanitizers

We give a new sanitizer for point functions with essentially optimal sample complexity, which improves and simplifies a result of [8].

**Proposition 6.2.13.** There exists an \((\alpha, \beta)\)-accurate and \((\varepsilon, \delta)\)-differentially private sanitizer for \( \text{Point}_X \) with sample complexity
\[
n = O \left( \frac{\log(1/\alpha \beta \delta)}{\alpha \varepsilon} \right).
\]

**Proof.** To give a \((2\alpha, \beta)\)-accurate sanitizer, it suffices to produces, for each point function \( c_x \), an approximate answer \( a_x \in [0, 1] \) with \( |a_x - c_x| \leq \alpha \). This is because given these approximate answers, one can reconstruct a dataset \( \hat{D} \) of size \( O(1/\alpha) \) with \( |c_x(\hat{D}) - a_x| \leq \alpha \) for every \( x \in X \).

The algorithm for producing the answers \( a_x \) is as follows.
Algorithm 6 Query release for Point \( X \)

**Input:** Privacy parameters \((\varepsilon, \delta)\), dataset \( D \in X^n \)

For each \( x \in X \), do the following:

1. If \( c_x(D) \leq \frac{\alpha}{4} \), release \( a_x = 0 \)

2. Let \( \hat{a}_x = c_x(D) + \text{Lap}(2/\varepsilon n) \)

3. If \( \hat{a}_x \leq \frac{\alpha}{2} \), release \( a_x = 0 \)

4. Otherwise, release \( a_x = \hat{a}_x \)

First, we argue that Algorithm 6 is \((\varepsilon, \delta)\)-differentially private. Below, we write \( X \approx_{(\varepsilon, \delta)} Y \) to denote the fact that for every measurable set \( S \) in the union of the supports of \( X \) and \( Y \), we have \( \Pr[X \in S] \leq e^{\varepsilon} \Pr[Y \in S] + \delta \).

Let \( D \sim D' \) be adjacent datasets of size \( n \), with \( x \in D \) replaced by \( x' \in D' \). Then the output distribution of the mechanism differs only on its answers to the queries \( c_x \) and \( c_{x'} \). Let us focus on \( c_x \). If both \( c_x(D) \leq \alpha/4 \) and \( c_x(D') \leq \alpha/4 \), then the mechanism always releases 0 for both queries. If both \( c_x(D) > \alpha/4 \) and \( c_x(D') > \alpha/4 \), then \( a_x(D) \approx_{(\varepsilon/2, 0)} a_x(D') \) by properties of the Laplace mechanism. Finally, if \( c_x(D) > \alpha/4 \) but \( c_x(D') \leq \alpha/4 \), then \( c_x(D') = 0 \) with probability 1. Moreover, we must have \( \text{Point}_{X'}(D) \leq \alpha/4 + 1/n \), so

\[
\Pr[a_x(D) = 0] \geq \Pr[\text{Lap}(2/\varepsilon n) < \alpha/4 - 1/n] = 1 - \frac{1}{2} \exp(-\varepsilon n\alpha/8 + \varepsilon/2) \geq 1 - \delta/2.
\]

So in this case, \( a_x(D) \approx_{(0, \delta/2)} a_x(D') \). Therefore, we conclude that overall \( a_x(D) \approx_{(\varepsilon/2, \delta/2)} a_x(D') \).

An identical argument holds for \( a_{x'} \), so the mechanism is \((\varepsilon, \delta)\)-differentially private.

Now we argue that the answers \( a_x \) are accurate. First, the answers are trivially \( \alpha \)-accurate for all queries \( c_x \) on which \( c_x(D) \leq \alpha/4 \). For each of the remaining queries, it is \( \alpha \)-accurate with probability at least

\[
\Pr[|\text{Lap}(2/\varepsilon n)| < \alpha/2] = 1 - \exp(-\varepsilon n\alpha/4) \geq 1 - \frac{\alpha \beta}{4}.
\]

Taking a union bound over the at most \( 4/\alpha \) queries with \( \text{Point}_{X'}(D) > \alpha/4 \), we conclude that the mechanism is \( \alpha \)-accurate for all queries with probability at least \( 1 - \beta \).
Recall that, improving on work of Beimel et al. [8], in Section 4.4 we gave a sanitizer for threshold functions with sample complexity roughly $2^{|X|} \log^* |X|$.

**Proposition 6.2.14.** There exists an $(\alpha, \beta)$-accurate and $(\varepsilon, \delta)$-differentially private sanitizer for $\text{Thresh}_X$ with sample complexity

$$n = O \left( \frac{1}{\alpha \varepsilon} \cdot 2^{|X|} \cdot \log^* |X| \cdot \log \left( \frac{\log^* |X|}{\varepsilon \delta} \right) \cdot \log(1/\beta) \cdot \log^{2.5}(1/\alpha) \right).$$

### 6.2.4 Private learners and multi-learners

Generalizing on the concept of private learners [81], we say that an algorithm $A$ is $(\alpha, \beta, \varepsilon, \delta)$-private PAC $k$-learner for $C$ using $\mathcal{H}$ if $A$ is $(\alpha, \beta)$-PAC $k$-learner for $C$ using $\mathcal{H}$, and $A$ is $(\varepsilon, \delta)$-differentially private (similarly with agnostic private PAC $k$-learners). We omit the parameter $k$ when $k = 1$ and the parameter $\delta$ when $\delta = 0$.

For the case $k = 1$, we have a generic construction with sample complexity proportional to $\log |C|$: 

**Theorem 6.2.15 ([81]).** Let $C$ be a concept class, and $\alpha, \beta, \varepsilon > 0$. There exists an $(\alpha, \beta, \varepsilon)$-private agnostic proper learner for $C$ with sample complexity $O \left( (\log |C| + \log 1/\beta) (1/((\alpha \varepsilon) + 1/\alpha^2)) \right)$.

Beimel, Nissim, and Stemmer [9] gave a generic transformation from data sanitization to private learning, which generally gives improved sample complexity upper bounds.

**Theorem 6.2.16 ([9]).** Suppose there exists an $(\alpha, \beta)$-accurate and $(\varepsilon, \delta)$-differentially private sanitizer for $C^\circ$ with sample complexity $m$. Then there exists a proper $(2\alpha, 2\beta)$-PAC and $(\varepsilon + \varepsilon', \delta')$-differentially private learner for $C$ with sample complexity

$$O \left( m + \frac{\text{VC}(C)}{\alpha^2 \varepsilon'} \log \left( \frac{1}{\alpha} \right) + \frac{1}{\alpha \varepsilon'} \log \left( \frac{1}{\beta} \right) \right).$$

A number of works [5, 7, 8, 27, 63] have established sharper upper and lower bounds for learning the specific concept classes $\text{Point}_X$ and $\text{Thresh}_X$. In the case of pure differential privacy, $\text{Point}_X$ requires $\Theta(\log |X|)$ samples to learn properly [5], but can be learned improperly with $O(1)$ samples. On the other hand, the class of threshold functions $\text{Thresh}_X$ require $\Omega(\log |X|)$ samples to learn, even improperly [63]. In the case of approximate differential privacy, $\text{Point}_X$ and $\text{Thresh}_X$ can be learned properly with sample complexities $O(1)$ [8] and $\tilde{O}(2^{|X|})$ [27], respectively. Moreover, properly learning threshold functions requires sample complexity $\Omega(\log^* |X|)$. 162
### 6.2.5 Private PAC learning vs. Empirical Learning

We saw by Theorem 6.2.9 that when an empirical $k$-learner $A$ for a concept class $C$ is run on a random sample of size $\Omega(VC(C))$, it is also a (agnostic) PAC $k$-learner. In particular, if an empirical $k$-learner $A$ is differentially private, then it also serves as a differentially private (agnostic) PAC $k$-learner.

Generalizing a result of [27], the next theorem shows that the converse is true as well: a differentially private (agnostic) PAC $k$-learner yields a private empirical $k$-learner with only a constant factor increase in the sample complexity.

**Theorem 6.2.17.** Let $\varepsilon \leq 1$. Suppose $A$ is an $(\varepsilon, \delta)$-differentially private $(\alpha, \beta)$-accurate (agnostic) PAC $k$-learner for a concept class $C$ with sample complexity $n$. Then there is an $(\varepsilon, \delta)$-differentially private $(\alpha, \beta)$-accurate (agnostic) empirical $k$-learner $\tilde{A}$ for $C$ with sample complexity $m = 9n$. Moreover, if $A$ is proper, then so is the resulting empirical learner $\tilde{A}$.

**Proof.** We give the proof for the agnostic case; the non-agnostic case is argued identically, and is immediate from [27]. To construct the empirical learner $\tilde{A}$, we use the fact that the given learner $A$ performs well on any distribution over labeled examples – in particular, it performs well on the uniform distribution over rows of the input dataset to $\tilde{A}$. Consider a dataset $S = ((x_i, y_{1,i}, \ldots, y_{k,i}))_{i=1}^m \in (\mathcal{X} \times \{0, 1\}^k)^m$. On input $S$, define $\tilde{A}$ by sampling $n$ rows from $S$ (with replacement), and outputting the result of running $A$ on the sample. Let $S$ denote the uniform distribution over the rows of $S$, and let $S_j$ be its marginal distribution which is uniform over $S_{ij} = ((x_i, y_{j,i}))_{i=1}^m$. Then sampling $n$ rows from $S$ is equivalent to sampling $n$ rows i.i.d. from $S$. Hence, if $(h_1, \ldots, h_k)$ is the output of $A$ on the subsample, we have

$$\Pr \left[ \max_{1 \leq j \leq k} \left( \text{err}_{S_j}(h_j) - \min_{c \in C} \text{err}_{S_j}(c) \right) > \alpha \right] = \Pr \left[ \max_{1 \leq j \leq k} \left( \text{err}_{S_j}(h_j) - \min_{c \in C} \text{err}_{S_j}(c) \right) > \alpha \right] \leq \beta.$$  

To show that $\tilde{A}$ remains $(\varepsilon, \delta)$-differentially private, we apply the secrecy of the sample lemma (Lemma 2.3.10 from Section 2.3.3), which shows that the sampling procedure does not hurt privacy. This completes the proof. 

□
6.3 Upper Bounds on the Sample Complexity of Private Multi-Learners

6.3.1 Generic Construction

In this section we present the following general upper bounds on the sample complexity of private $k$-learners.

**Theorem 6.3.1.** Let $C$ be a finite concept class, and let $k \geq 1$. There exists a proper agnostic $(\alpha, \beta, \varepsilon)$-private PAC $k$-learner for $C$ with sample complexity

$$O_{\alpha, \beta, \varepsilon} \left( k \cdot \log k + \min \left\{ k \cdot \log |C|, (k + \log |\mathcal{X}|) \cdot \text{VC}(C) \right\} \right),$$

and there exists a proper agnostic $(\alpha, \beta, \varepsilon, \delta)$-private PAC $k$-learner for $C$ with sample complexity

$$O_{\alpha, \beta, \varepsilon, \delta} \left( \sqrt{k} \cdot \log k + \min \left\{ \sqrt{k} \cdot \log |C|, (\sqrt{k} + \log |\mathcal{X}|) \cdot \text{VC}(C), \sqrt{k} \cdot \text{VC}(C) + \sqrt{\log |\mathcal{X}|} \cdot \log |C| \right\} \right).$$

The straightforward approach for constructing a private $k$-learner for a class $C$ is to separately apply a (standard) private learner for $C$ for each of the $k$ target concepts. Using composition theorem 3.3.3 to argue the overall privacy guarantee of the resulting learner, we get the following observation.

**Observation 6.3.2.** Let $C$ be a concept class and let $k \geq 1$. If there is an $(\alpha, \beta, \varepsilon, \delta)\text{-PAC learner for } C$ with sample complexity $n$, then

- There is an $(\alpha, k\beta, k\varepsilon, k\delta)\text{-PAC } k\text{-learner for } C$ with sample complexity $n$.
- There is an $(\alpha, k\beta, O(\sqrt{k \log (\frac{1}{\beta})} + ke^2), O(k\delta))\text{-PAC } k\text{-learner for } C$ with sample complexity $n$.

Moreover, if the initial learner is proper and/or agnostic, then so is the resulting learner.

In cases where sample efficient private PAC learners exist, it might be useful to apply Observation 6.3.2 in order to obtain a private $k$-learner. For example, Beimel et al. [5, 7] gave an improper agnostic $(\alpha, \beta, \varepsilon)$-PAC learner for Point$_\mathcal{X}$ with sample complexity $O_\alpha(\frac{1}{\varepsilon} \log \frac{1}{\beta})$. Using Observation 6.3.2 yields the following corollary.

**Corollary 6.3.3.** There exists an improper agnostic $(\alpha, \beta, \varepsilon)$-PAC $k$-learner for Point$_\mathcal{X}$ with sample complexity $O_{\alpha, \beta, \varepsilon}(k \log k)$. 

164
For a general concept class $C$, we can use Observation 6.3.2 with the generic construction of Theorem 6.2.15, stating that for every concept class $C$ there exists a private agnostic proper learner $A$ that uses $O(\log |C|)$ labeled examples.

**Corollary 6.3.4.** Let $C$ be a concept class, and $\alpha, \beta, \epsilon > 0$. There exists an $(\alpha, \beta, \epsilon)$-private agnostic proper $k$-learner for $C$ with sample complexity $O_{\alpha, \beta, \epsilon}(k \cdot \log |C| + k \cdot \log k)$. Moreover, there exists an $(\alpha, \beta, \epsilon, \delta)$-private agnostic proper $k$-learner for $C$ with sample complexity $O_{\alpha, \beta, \epsilon, \delta}(\sqrt{k} \cdot \log |C| + \sqrt{k} \cdot \log k)$.

**Example 6.3.5.** There exists a proper agnostic $(\alpha, \beta, \epsilon)$-PAC $k$-learner for Parity$_d$ with sample complexity $O_{\alpha, \beta, \epsilon}(kd + k \log k)$.

As we will see in Section 6.5, the bounds of Corollary 6.3.3 and Example 6.3.5 on the sample complexity of $k$-learning $\text{Point}_X$ and Parity$_d$ are tight (up to logarithmic factors). That is, with pure differential privacy, the direct sum gives (roughly) optimal bounds for improperly learning $\text{Point}_X$, and for (properly or improperly) learning $\text{Parity}_d$. This is not the case for learning $\text{Thresh}_X$ or for properly learning $\text{Point}_X$.

In order to avoid the factor $k \log |C|$ (or $\sqrt{k} \log |C|$) in Corollary 6.3.4, we now show how an idea used in [9] (in the context of semi-supervised learning) can be used to construct sample efficient private $k$-learners. In particular, this construction will achieve tight bounds for learning $\text{Thresh}_X$ and for properly learning $\text{Point}_X$ under pure-differential privacy.

Fix a concept class $C$, target concepts $c_1, \ldots, c_k \in C$, and a $k$-labeled dataset $S$ (we use $D$ to denote the unlabeled portion of $S$). For every $1 \leq j \leq k$, the goal is to identify a hypothesis $h_j \in C$ with low $\text{err}_D(h_j; c_j)$ (such a hypothesis also has good generalization). Beimel et al. [9] observed that given a sanitization $\hat{D}$ of $D$ w.r.t. $C^\oplus = \{f \oplus g : f, g \in C\}$, for every $f, g \in C$ it holds that

$$\text{err}_D(f, g) = \frac{1}{|D|}|\{x \in D : (f \oplus g)(x) = 1\}| \approx \frac{1}{|D|}|\{x \in \hat{D} : (f \oplus g)(x) = 1\}| = \text{err}_{\hat{D}}(f, g).$$

Hence, a hypothesis $\mathcal{H}$ with low $\text{err}_{\hat{D}}(h, c_j)$ also has low $\text{err}_D(h, c_j)$ and vice versa. Let $\mathcal{H}$ be the set of all dichotomies over $\hat{D}$ realized by $C$. Note that $\exists f^*_j \in \mathcal{H}$ that agrees with $c_j$ on $\hat{D}$, i.e., $\exists f^*_j \in \mathcal{H}$ s.t. $\text{err}_{\hat{D}}(f^*_j, c_j) = 0$, and hence $\text{err}_D(f^*_j, c_j)$ is also low. The thing that works in our favor here is that $\mathcal{H}$ is small – at most $2^{|\hat{D}|} \leq 2^{Y_{\text{VC}}(C)}$ – and hence choosing a hypothesis out of $\mathcal{H}$ is easy. Therefore, for every $j$ we can use the exponential mechanism to identify a hypothesis $h_j \in \mathcal{H}$ with
low \text{err}_D(h_j, c_j).

Lemma 6.3.6. Let \( C \) be a concept class, and \( \alpha, \beta, \epsilon, \delta > 0 \). There exists an \((\alpha, \beta, \epsilon)\)-private agnostic \( k \)-learner for \( C \) with sample complexity \( O_{\alpha, \beta, \epsilon}(\text{VC}(C) \cdot \log |X| + k \cdot \text{VC}(C) + k \cdot \log k) \). Moreover, there exists an \((\alpha, \beta, \epsilon, \delta)\)-private agnostic \( k \)-learner for \( C \) with sample complexity \( O_{\alpha, \beta, \epsilon, \delta}(\min\{\text{VC}(C) \cdot \log |X|, \log |C| \cdot \sqrt{\log |X|}\} + \sqrt{k} \cdot \text{VC}(C) + \sqrt{k} \cdot \log k) \).

Lemma 6.3.6 follows from the following lemma.

Lemma 6.3.7. Let \( \epsilon' > 0 \) and let \( A \) be an \((\frac{\alpha}{\epsilon'}, \frac{\beta}{\epsilon'})\)-accurate \((\epsilon, \delta)\)-private sanitizer for \( C \) with sample complexity \( m \). Then there is an \((\alpha, \beta)\)-PAC agnostic \( k \)-learner for \( C \) with sample complexity

\[
O \left( m + \frac{\text{VC}(C)}{\alpha^2 \epsilon^2} \log \left( \frac{1}{\alpha} \right) + \frac{1}{\alpha \epsilon} \log \left( \frac{k}{\beta} \right) + \frac{1}{\alpha^2} \text{VC}(C) \log \left( \frac{k}{\alpha \beta} \right) \right).
\]

Moreover, it is both \((\epsilon + k \epsilon', \delta)\) and \((\epsilon + \sqrt{2k \ln(1/\delta)} \epsilon' + 2k \epsilon'^2, 2\delta)\)-differentially private.

Using Lemma 6.3.7 with the generic sanitizer of Theorem 2.2.8 or Theorem 2.2.9 results in Lemma 6.3.6.

**Algorithm 7** *GenericLearner*

**Input:** Concept class \( C \), privacy parameters \( \epsilon', \epsilon, \delta \), and a \( k \)-labeled dataset \( S = (x_i, y_{i,1}, \ldots, y_{i,k})_{i=1}^n \). We use \( D = (x_i)_{i=1}^n \) to denote the unlabeled portion of \( S \).

**Used Algorithm:** An \((\frac{\alpha}{\epsilon'}, \frac{\beta}{\epsilon'})\)-accurate \((\epsilon, \delta)\)-private sanitizer for \( C \) with sample complexity \( m \).

1. Initialize \( H = \emptyset \).
2. Construct an \((\epsilon, \delta)\)-private sanitization \( \tilde{D} \) of \( D \) w.r.t. \( C \), where \( |\tilde{D}| = O \left( \frac{\text{VC}(C)}{\alpha^2} \log \left( \frac{1}{\alpha} \right) \right) = O \left( \frac{\text{VC}(C)}{\alpha^2} \log \left( \frac{1}{\alpha} \right) \right) \).
3. Let \( B = \{b_1, \ldots, b_{|B|}\} \) be the set of all points appearing at least once in \( \tilde{D} \).
4. For every \((z_1, \ldots, z_{|B|}) \in \Pi_C(B) = \{(c(b_1), \ldots, c(b_{|B|})) : c \in C\}, \) add to \( \mathcal{H} \) an arbitrary concept \( c \in C \) s.t. \( c(b_{1}) = z_{1} \) for every \( 1 \leq \ell \leq |B| \).
5. For every \( 1 \leq j \leq k \), use the exponential mechanism with privacy parameter \( \epsilon' \) to choose and return a hypothesis \( h_j \in \mathcal{H} \) with (approximately) minimal error on the examples in \( S \) w.r.t. their \( j^{\text{th}} \) label.

**Proof of Lemma 6.3.7.** The proof is via the construction of *GenericLearner* (algorithm 7). Note that *GenericLearner* only accesses \( S \) via a sanitizer (on Step 2) and using the exponential mechanism (on Step 5). Composition theorem 3.3.3 state that *GenericLearner* is both \((\epsilon + k \epsilon', \delta)\)-differentially
private and \((\epsilon + \sqrt{2k\ln(1/\delta)\epsilon^2} + 2k\epsilon^2, 2\delta)\)-differentially private. We, thus, only need to prove that with high probability the learner returns \(\alpha\)-good hypotheses.

Fix a distribution \(\mathcal{P}\) over \(X \times \{0, 1\}^k\), and let \(\mathcal{P}_j\) denote the marginal distribution of \(\mathcal{P}\) on the examples and the \(j\)th label. Let \(S\) consist of examples \((x_i, y_{i,1}, \ldots, y_{i,k}) \sim \mathcal{P}\). We use \(D = (x_i)_{i=1}^n\) to denote the unlabeled portion of \(S\), and use \(S_{|j} = ((x_i, y_{j,i}))_{i=1}^n\) to denote a dataset containing the examples in \(S\) together with their \(j\)th label. Define the following three events:

\[E_1: \text{For every } f, h \in \mathcal{C} \text{ it holds that } |\text{err}_D(f, h) - \text{err}_{\mathcal{B}}(f, h)| \leq \frac{2\alpha}{5}.\]

\[E_2: \text{For every } f \in \mathcal{C} \text{ and for every } 1 \leq j \leq k \text{ it holds that } |\text{err}_{S_{|j}}(f) - \text{err}_{\mathcal{P}_j}(f)| \leq \frac{\alpha}{5}.\]

\[E_3: \text{For every } 1 \leq j \leq k, \text{ the hypothesis } h_j \text{ chosen by the exponential mechanism is such that } \text{err}_{S_{|j}}(h_j) \leq \frac{\alpha}{5} + \min_{f \in \mathcal{H}} \left\{\text{err}_{S_{|j}}(f)\right\}.\]

We first argue that when these three events happen algorithm \(\text{GenericLearner}\) returns good hypotheses. Fix \(1 \leq j \leq k\), and let \(c_j^* = \arg\min_{f \in \mathcal{C}} \{\text{err}_{\mathcal{P}_j}(f)\}\). We denote \(\Delta = \text{err}_{\mathcal{P}_j}(c_j^*)\). We need to show that if \(E_1 \cap E_2 \cap E_3\) occurs, then the hypothesis \(h_j\) returned by \(\text{GenericLearner}\) is s.t. \(\text{err}_{\mathcal{P}_j}(h_j) \leq \alpha + \Delta\).

For every \((y_1, \ldots, y_{|B|}) \in \Pi_C(B)\), algorithm \(\text{GenericLearner}\) adds to \(\mathcal{H}\) a hypothesis \(f\) s.t. \(\forall 1 \leq \ell \leq |B|, f(b_{\ell}) = y_{\ell}\). In particular, \(\mathcal{H}\) contains a hypothesis \(h_j^*\) s.t. \(h_j^*(x) = c_j^*(x)\) for every \(x \in B\), that is, a hypothesis \(h_j^*\) s.t. \(\text{err}_{\mathcal{B}}(h_j^*, c_j^*) = 0\). As event \(E_1\) has occurred we have that this \(h_j^*\) satisfies \(\text{err}_D(h_j^*, c_j^*) \leq \frac{2\alpha}{5}\). Using the triangle inequality (and event \(E_2\)) we get that this \(h_j^*\) satisfies \(\text{err}_{S_{|j}}(h_j^*) \leq \text{err}_D(h_j^*, c_j^*) + \text{err}_{S_{|j}}(c_j^*) \leq \frac{3\alpha}{5} + \Delta\). Thus, event \(E_3\) ensures that algorithm \(\text{GenericLearner}\) chooses (using the exponential mechanism) a hypothesis \(h_j \in \mathcal{H}\) s.t. \(\text{err}_{S_{|j}}(h_j) \leq \frac{4\alpha}{5} + \Delta\). Event \(E_2\) ensures, therefore, that this \(h_j\) satisfies \(\text{err}_{\mathcal{P}_j}(h_j) \leq \alpha + \Delta\). We will now show \(E_1 \cap E_2 \cap E_3\) happens with high probability.

Standard arguments in learning theory state that (w.h.p.) the empirical error on a (large enough) random sample is close to the generalization error. Specifically, by setting \(n \geq O\left(\frac{1}{\alpha^2} \text{VC(}\mathcal{C}\text{)} \log \left(\frac{k}{\alpha^2}\right)\right)\), Theorem 6.2.6 ensures that Event \(E_2\) occurs with probability at least \((1 - \frac{2}{5}\beta)\).

Assuming that \(n \geq m\) (the sample complexity of the sanitizer used in Step 5), with probability
at least \((1 - \frac{\beta}{3})\) for every \((h \oplus f) \in \mathcal{C}^\oplus\) (i.e., for every \(h, f \in \mathcal{C}\)) it holds that

\[
\frac{\alpha}{5} \geq |Q_{(h \oplus f)}(D) - Q_{(h \oplus f)}(\tilde{D})| = \left| \frac{|\{x \in D : (h \oplus f)(x) = 1\}|}{|D|} - \frac{|\{x \in \tilde{D} : (h \oplus f)(x) = 1\}|}{|\tilde{D}|} \right|
\]

\[
= \left| \frac{|\{x \in D : h(x) \neq f(x)\}|}{|D|} - \frac{|\{x \in \tilde{D} : h(x) \neq f(x)\}|}{|\tilde{D}|} \right|
\]

\[
= |\text{err}_D(h, f) - \text{err}_{\tilde{D}}(h, f)|.
\]

Event \(E_1\) occurs therefore with probability at least \((1 - \frac{\beta}{3})\).

The exponential mechanism ensures that the probability of event \(E_3\) is at least \(1 - k|H| \cdot \exp(-\epsilon'm/10)\) (see Proposition 2.1.4). Note that \(\log |H| \leq |B| \leq |\tilde{D}| = O\left(\frac{\text{VC}(\mathcal{C}) \log(\frac{1}{\alpha})}{\alpha^2}\right)\). Therefore, for \(n \geq O\left(\frac{\text{VC}(\mathcal{C}) \log(\frac{1}{\alpha}) + \frac{1}{\alpha^2} \log(\frac{\delta}{\beta})}{\alpha^2}\right)\), Event \(E_3\) occurs with probability at least \((1 - \frac{\beta}{5})\).

All in all, setting \(n \geq O\left(m + \frac{\text{VC}(\mathcal{C})}{\alpha^2} \log(\frac{1}{\alpha}) + \frac{1}{\alpha^2} \log(k) + \frac{1}{\alpha^2} \text{VC}(\mathcal{C}) \log(\frac{k}{\alpha^2})\right)\), ensures that the probability of \(\text{GenericLearner}\) failing is at most \(\beta\).

Theorem 6.3.1 now follows by combining Lemma 6.3.6 and Corollary 6.3.4.

For certain concept classes, there are sanitizers with substantially lower sample complexity than the generic sanitizers. Combining Lemma 6.3.6 with Proposition 6.2.13, we obtain:

**Corollary 6.3.8.** There is an \((\alpha, \beta)\)-PAC agnostic \(k\)-learner for \(\text{Point}_X\) with sample complexity

\[
O\left(\frac{\log(1/\alpha \beta \delta)}{\alpha\epsilon} + \frac{\log(1/\alpha)}{\alpha^3 \epsilon'} + \frac{\log(k/\beta)}{\alpha \epsilon'} + \frac{\log(k/\alpha \beta)}{\alpha^2}\right).
\]

Moreover, it is both \((\epsilon + k^2 \epsilon', \delta)\) and \((\epsilon + \sqrt{2k \log(1/\delta) \epsilon' + 2k^2 \delta}, 2\delta)\)-differentially private.

Similarly, combining Lemma 6.3.6 with Proposition 6.2.14, we obtain:

**Corollary 6.3.9.** There is an \((\alpha, \beta)\)-PAC agnostic \(k\)-learner for \(\text{Thresh}_X\) with sample complexity

\[
O\left(\frac{2^{\log^* |\mathcal{X}|} \cdot \log^* |\mathcal{X}| \cdot \log \left(\frac{\log^* |\mathcal{X}|}{\alpha \epsilon}\right)}{\alpha \epsilon} \cdot \log(1/\beta) \cdot \log^{2.5}(1/\alpha) + \frac{\log(1/\alpha)}{\alpha^3 \epsilon'} + \frac{\log(k/\beta)}{\alpha \epsilon'} + \frac{\log(k/\alpha \beta)}{\alpha^2}\right).
\]

Moreover, it is both \((\epsilon + k^2 \epsilon', \delta)\) and \((\epsilon + \sqrt{2k \log(1/\delta) \epsilon' + 2k^2 \delta}, 2\delta)\)-differentially private.
6.3.2 Upper Bounds for Approximate Private Multi-Learners

In this section we give two examples of cases where the sample complexity of private \( k \)-learning is of the same order as that of non-private \( k \)-learning (the sample complexity does not depend on \( k \)). Our algorithms are \((\varepsilon, \delta)\)-differentially private, and rely on stability arguments: the identity of the best \( k \) concepts, as an entire vector, is unlikely to change on nearby \( k \)-labeled datasets. Hence, it can be released privately.

The main technical tool we use is the \( A_{\text{dist}} \) algorithm of Smith and Thakurta [113]. Our discussion follows the treatment of [8].

Recall that a quality function \( q : X^* \times F \rightarrow \mathbb{N} \) defines an optimization problem over the domain \( X^* \) and a finite solution set \( F \): Given a dataset \( S \in X^* \), find \( f \in F \) that (approximately) maximizes \( q(S, f) \). The sensitivity of a quality function, \( \Delta q \), is the maximum over all \( f \in F \) of the sensitivity of the function \( q(\cdot, f) \). The algorithm \( A_{\text{dist}} \) privately identifies the exact maximizer as long as it is sufficiently stable.

**Algorithm 8** \( A_{\text{dist}} \)

**Input:** Privacy parameters \( \varepsilon, \delta \), dataset \( S \in X^* \), sensitivity-1 quality function \( q \)

1. Let \( f_1, f_2 \in F \) be the highest scoring and second-highest scoring solutions to \( q(S, \cdot) \), respectively.
2. Let \( \text{gap} = q(S, f_1) - q(S, f_2) \), and \( \widehat{\text{gap}} = \text{gap} + \text{Lap}(1/\varepsilon) \).
3. If \( \widehat{\text{gap}} < \frac{1}{\varepsilon} \log \frac{1}{\delta} \), output \( \perp \). Otherwise, output \( f_1 \).

**Proposition 6.3.10** (Properties of \( A_{\text{dist}} \) [113]).

1. Algorithm \( A_{\text{dist}} \) is \((\varepsilon, \delta)\)-differentially private.

2. When run on a dataset \( S \) with \( \text{gap} > \frac{1}{\varepsilon} \log \frac{1}{\delta} \), Algorithm \( A_{\text{dist}} \) outputs the highest scoring solution \( f_1 \) with probability at least \( 1 - \beta \).

Learning Parities under the Uniform Distribution

**Theorem 6.3.11.** For every \( k, d \) there exists an \((\alpha=0, \beta, \varepsilon, \delta)\)-PAC (non-agnostic) \( k \)-learner for Parity\(_d\) under the uniform distribution with sample complexity \( O(d \log(\frac{1}{\beta}) + \frac{1}{\varepsilon} \log(\frac{1}{\delta})) \).
Recall that (even without privacy constraints) the sample complexity of PAC learning Parity\(_d\) under the uniform distribution is \(\Omega(d)\). Hence the sample complexity of privately \(k\)-learning Parity\(_d\) (non-agnostically) under the uniform distribution is of the same order as that of non-private \(k\)-learning.

For the intuition behind Theorem 6.3.11, let \(c_1, \ldots, c_k\) denote the \(k\) target concepts, and consider the quality function \(q(D, (h_1, \ldots, h_k)) = \max_{1 \leq j \leq k} \{\text{err}_D(h_j, c_j)\}\). On a large enough sample \(D\) we expect that \(q(D, (h_1, \ldots, h_k)) \approx \frac{1}{2}\) for every \((h_1, \ldots, h_k) \neq (c_1, \ldots, c_k)\), while \(q(D, (c_1, \ldots, c_k)) = 0\).

The \(k\) target concepts can hence be privately identified (exactly) using stability techniques.

In order to make our algorithm computationally efficient, we apply the “subsample and aggregate” idea of Nissim et al. [97]. We divide the input sample into a small number of subsamples, use Gaussian elimination to (non-privately) identify a candidate hypothesis vector on each subsample, and then select from these candidates privately.

**Algorithm 9 ParityLearner**

**Input:** Parameters \(\epsilon, \delta\), and a \(k\)-labeled dataset \(S\) of size \(n = O(\frac{d^2}{\epsilon} \log(\frac{1}{\delta}))\).

**Output:** Hypotheses \(h_1, \ldots, h_k\).

1. Split \(S\) into \(m = O(\frac{1}{\epsilon} \log(\frac{1}{\delta}))\) disjoint samples \(S_1, \ldots, S_m\) of size \(O(d)\) each. Initiate \(Y\) as the empty multiset.

2. For every \(1 \leq t \leq m\):
   
   (a) For every \(1 \leq j \leq k\) try to use Gaussian elimination to identify a parity function \(y_j\) that agrees with the labels of the \(j\)th column of \(S_t\).
   
   (b) If a parity is identified for every \(j\), then set \(Y = Y \cup \{(y_1, \ldots, y_k)\}\). Otherwise set \(Y = Y \cup \{\bot\}\).

3. Use algorithm \(\mathcal{A}_{\text{dist}}\) with privacy parameters \(\epsilon, \delta\) to choose and return a vector of \(k\) parity functions \((h_1, \ldots, h_k) \in (\text{Parity}_d)^k\) with a large number of appearances in \(Y\).

**Proof of Theorem 6.3.11.** The proof is via the construction of ParityLearner (algorithm 9). First note that changing a single input element in \(S\) can change (at most) one element of \(Y\). Hence, applying (the \((\epsilon, \delta)\)-private) algorithm \(\mathcal{A}_{\text{dist}}\) on \(Y\) preserves privacy (applying ParityLearner on neighboring inputs amounts to executing \(\mathcal{A}_{\text{dist}}\) on neighboring inputs).

Now fix \(k\) target concepts \(c_1, \ldots, c_k \in \text{Parity}_d\) and let \(S\) be a random \(k\)-labeled dataset containing \(n\) i.i.d. elements from the uniform distribution \(U_d\) over \(X = \{0, 1\}^d\), each labeled by \(c_1, \ldots, c_k\). Observe that (for every \(1 \leq t \leq m\)) we have that \(S_t\) contains i.i.d. elements from \(U_d\) labeled by \(c_1, \ldots, c_k\). We use \(D_t\) to denote the unlabeled portion of \(S_t\). Standard arguments in learning theory
(cf. Theorem 6.2.5) state that for \(|S_t| \geq O(d)|
\[
\Pr \left[ \exists h, f \in \text{Parity}_d \text{ s.t. } \text{err}_{U_d}(h, f) \geq \frac{1}{4} \wedge \text{err}_{D_t}(h, f) \leq \frac{1}{40} \right] \leq \frac{1}{8}.
\]

The above inequality holds, in particular, for every hypothesis \(h \in \text{Parity}_d\) and every target concept \(c_j\), and hence,

\[
\Pr \left[ \exists h \in \text{Parity}_d \text{ and } j \text{ s.t. } \text{err}_{U_d}(h, c_j) \geq \frac{1}{4} \wedge \text{err}_{D_t}(h, c_j) \leq \frac{1}{40} \right] \leq \frac{1}{8}.
\]

Recall that under the uniform distribution, the only \(h \in \text{Parity}_d\) s.t. \(\text{err}_{U_d}(h, c_j) \neq \frac{1}{2}\) is \(c_j\) itself, and hence

\[
\Pr \left[ \exists h \in \text{Parity}_d \text{ and } j \text{ s.t. } h \neq c_j \wedge \text{err}_{D_t}(h, c_j) \leq \frac{1}{40} \right] \leq \frac{1}{8}.
\]

So, for every \(1 \leq t \leq m\), with probability 7/8 we have that for every label column \(j\) the only hypothesis with empirical error less than \(\frac{1}{40}\) on \(S_t\) is the \(j^{th}\) target concept itself (with empirical error 0). In such a case, step 2a (Gaussian elimination) identifies exactly the vector of \(k\) target concepts \((c_1, \ldots, c_k)\). Since \(m \geq O(\log(\frac{1}{\beta}))\), the Chernoff bound ensures that except with probability \(\beta/2\), the vector \((c_1, \ldots, c_k)\) is identified in at least 3/4 of the iterations of step 2. Assuming that this is the case, the vector \((c_1, \ldots, c_k)\) appears in \(Y\) at least \(3m/4\) times, while every other vector can appear at most \(m/4\) times. Provided that \(m \geq O(\frac{1}{\epsilon} \log(\frac{1}{\beta}))\), algorithm \(A_{\text{dist}}\) ensures that the \(k\) target concepts are chosen with probability \(1 - \beta/2\).

All in all, algorithm \(\text{ParityLearner}\) identifies the \(k\) target concepts (exactly) with probability \(1 - \beta\), provided that \(n \geq O(\frac{d}{\epsilon} \log(\frac{1}{\beta}))\).

\[\square\]

**Learning Points**

We next show that the class of \(\text{Point}_\mathcal{X}\) can be (non-agnostically) \(k\)-learned using constant sample complexity, matching the non-private sample complexity.

**Theorem 6.3.12.** For every domain \(\mathcal{X}\) and every \(k \in \mathbb{N}\) there exists an \((\alpha, \beta, \epsilon, \delta)\)-PAC (non-agnostic) \(k\)-learner for \(\text{Point}_\mathcal{X}\) with sample complexity \(O(\frac{1}{\alpha \epsilon} \log(\frac{1}{\alpha \delta}))\).

The proof is via the construction of Algorithm 10. The algorithm begins by privately identifying (using sanitization) a set of \(O(1/\alpha)\) “heavy” elements in the input dataset, appearing \(\Omega(\alpha)\) times. The \(k\) labels of such a heavy element can be privately identified using stability arguments (since their
duplicity in the dataset is large). The labels of a “non-heavy” element can be set to 0 since a target concept can evaluate to 1 on at most one such non-heavy element, in which case the error is small.

**Notation.** We use $\#_S(x)$ to denote the duplicity of a domain element $x$ in a dataset $S$. For a distribution $\mu$ we denote $\mu(x) = \Pr[\tilde{x} \sim \mu \mid \tilde{x} = x]$.

**Algorithm 10 PointLearner**

**Input:** Privacy parameters $\epsilon, \delta$, and a $k$-labeled dataset $S = (x_i, y_i, \ldots, y_i, k)_{i=1}^n$. We use $D = (x_i)_{i=1}^n$ to denote the unlabeled portion of $S$.

**Output:** Hypotheses $h_1, \ldots, h_k$.

1. Let $\hat{D} \in \mathcal{X}^m$ be an $(\frac{\epsilon}{2}, \frac{\delta}{2})$-private $(\frac{\alpha}{15}, \frac{\beta}{4})$-accurate sanitization of $D$ w.r.t. Point$_X$ (e.g., using Proposition 6.2.13).

2. Let $G = \{ x \in \mathcal{X} : \frac{1}{m} \#_D(x) \geq \alpha/15 \}$ be the set of all “$\frac{\alpha}{15}$-heavy” domain elements w.r.t. the sanitization $\hat{D}$. Note that $|G| \leq 15/\alpha$.

3. Let $q$ be the quality function that on input a $k$-labeled dataset $S$, a domain element $x$, and a binary vector $\vec{v} \in \{0, 1\}^k$, returns the number of appearances of $(x, \vec{v})$ in $S$. That is, $q(S, x, (v_1, \ldots, v_k)) = |\{i : x_i = x \wedge y_i = v_1 \wedge \ldots \wedge y_i = v_k\}|$.

4. Use algorithm $A_{\text{dist}}$ with privacy parameters $\frac{\epsilon}{2}, \frac{\delta}{2}$ to choose a set of vectors $V = \{\vec{v}_x \in \{0, 1\}^k : x \in G\}$ maximizing $Q(S, V) = \min_{\vec{v}_x \in V} \{q(S, x, \vec{v}_x)\}$. That is, we use algorithm $A_{\text{dist}}$ to choose a set of $|G|$ vectors – a vector $\vec{v}_x$ for every $x \in G$ -- such that the minimal number of appearances of an entry $(x, \vec{v}_x)$ in the dataset $S$ is maximized.

5. For $1 \leq j \leq k$: If the $j$th entry of every $\vec{v}_x \in V$ is 0, then set $h_j \equiv 0$. Otherwise, let $x$ be s.t. $\vec{v}_x \in V$ has 1 as its $j$th entry, and define $h_j : \mathcal{X} \rightarrow \{0, 1\}$ as $h_j(y) = 1$ iff $y = x$.

6. Return $h_1, \ldots, h_k$.

**Proof.** The proof is via the construction of PointLearner (algorithm 10). First note the algorithm only access the input dataset using sanitization on step 1, and using algorithm $A_{\text{dist}}$ on step 4. By composition theorem 3.3.3, algorithm PointLearner is $(\epsilon, \delta)$-differentially private.

Let $\mu$ be a distribution over $\mathcal{X}$, and let $c_1, \ldots, c_k \in $ Point$_X$ be the fixed target concepts. Consider the execution of PointLearner on a dataset $S = (x_i, y_i, \ldots, y_i, k)_{i=1}^n$ sampled from $\mu$ and labeled by $c_1, \ldots, c_k$. We use $D$ to denote the unlabeled portion of $S$, $\hat{D}$ for the sanitization of $D$ constructed on step 1, and write $m = |\hat{D}|$. Define the following good events.

$E_1$: For every $x \in \mathcal{X}$ s.t. $\mu(x) \geq \alpha$ it holds that $\frac{1}{n} \#_S(x) \geq \alpha/10$.

$E_2$: For every $x \in \mathcal{X}$ we have that $|\frac{1}{m} \#_\hat{D}(x) - \frac{1}{n} \#_S(x)| \leq \alpha/30$. 

172
$E_3$: Algorithm $A_{dist}$ returns a vector set $V$ s.t. $q(S, x, v_x) \geq 1$ for every $x \in G$.

We now argue that when these three events happen algorithm $PointLearner$ returns good hypotheses. First, observe that the set $G$ contains every element $x$ s.t. $\mu(x) \geq \alpha$: Let $x$ be s.t. $\mu(x) \geq \alpha$. As event $E_1$ has occurred, we have that $\frac{1}{n}\#S(x) \geq \alpha/10$. As event $E_2$ has occurred, we have that $\frac{1}{m}\#D(x) \geq \alpha/15$, and therefore $x \in G$.

Note that if $q(S, x, \bar{v}) \geq 1$ then the example $x$ is labeled as $\bar{v}$ by the target concepts. Thus, as event $E_3$ has occurred, for every $\bar{v}_x \in V$ it holds that $\bar{v}_x = (c_1(x), \ldots, c_k(x))$. Now let $h_j$ be the $j^{th}$ returned hypothesis. We next show that $h_j$ is $\alpha$-good. If $h \neq 0$, then let $x$ be the unique element s.t. $h_j(x) = 1$, and note that (according to step 5) the $j^{th}$ entry of $\bar{v}_x$ is 1, and hence, $c_j(x) = 1$. So $h_j = c_j$ (since $c_j$ is a concept in $Point(X)$).

If $h_j \equiv 0$ then the $j^{th}$ entry of every $\bar{v}_x \in V$ is 0. Note that in such a case $h_j$ only errs on the unique element $x$ s.t. $c_j(x) = 1$, and it suffices to show that $\mu(x) < \alpha$. Assume towards contradiction that $\mu(x) \geq \alpha$. As before, event $E_1 \cap E_2$ implies that $x \in G$. As event $E_3$ has occurred, we also have that $\bar{v}_x \in V$ s.t. $q(S, x, \bar{v}_x) \geq 1$, and the example $x$ is labeled as $\bar{v}_x$ by the target concepts. This contradicts the assumption that the $j^{th}$ entry of $\bar{v}_x \in V$ is 0.

Thus, whenever $E_1 \cap E_2 \cap E_3$ happens, algorithm $PointLearner$ returns $\alpha$-good hypotheses. We will now show $E_1 \cap E_2 \cap E_3$ happens with high probability. Provided $n \geq O(\frac{1}{\alpha} \log(\frac{1}{\alpha}))$, event $E_2$ is guaranteed to hold with all but probability $\beta/4$ by the utility properties of the sanitizer used on step 1. See Proposition 6.2.13.

Theorem 6.2.5 (VC bound) ensures that event $E_1$ holds with probability $1 - \beta/4$, provided that $n \geq O(\frac{1}{\alpha} \log(\frac{1}{\alpha}))$. To see this, let $z \equiv 0$ denote the constant 0 hypothesis, and consider the class $C = Point(X) \cup \{z\}$. Note that VC($C$) = 1. Hence, Theorem 6.2.9 states that, with all but probability $1 - \beta/4$, for every $c \in Point(X)$ s.t. $\text{err}_\mu(c, z) \geq \alpha$ it holds that $\text{err}_D(c, z) \geq \alpha/10$. That is, with all but probability $1 - \beta/4$, for every $x \in X$ s.t. $\mu(x) \geq \alpha$ it holds that $\frac{1}{n}\#D(x) = \frac{1}{n}\#S(x) \geq \alpha/10$.

Before analyzing event $E_3$, we show that if $E_2$ occurs, then every $x \in G$ is s.t. $\#S(x) \geq \alpha/30$. Let $x \in G$, that is, $x$ s.t. $\frac{1}{m}\#D(x) \geq \alpha/15$. Assuming event $E_2$ has occurred, we therefore have that $\frac{1}{n}\#S(x) \geq \alpha/30$. So every $x \in G$ appears in $S$ at least $\alpha n/30$ times with the labels $(c_1(x), \ldots, c_k(x)) \triangleq \bar{c}(x)$. Thus, $q(S, x, \bar{c}(x)) \geq \alpha n/30$. In addition, for every $\bar{v} \neq \bar{c}(x)$ it holds that $q(S, x, \bar{v}) = 0$, since every appearance of the example $x$ is labeled by the target concepts. Hence,
provided that \( n \geq O\left( \frac{1}{\alpha \epsilon \log \left( \frac{1}{\alpha \beta} \right) \epsilon} \right) \), algorithm \( \mathcal{A}_{\text{dist}} \) ensures that event \( E_3 \) happens with probability at least \( 1 - \beta/2 \).

Overall, \( E_1 \cap E_2 \cap E_3 \) happens with probability at least \( 1 - \beta \). \( \square \)

### 6.4 Approximate Privacy Lower Bounds from Fingerprinting Codes

In this section, we show how fingerprinting codes can be used to obtain \( \text{poly}(k) \) lower bounds against privately learning \( k \) concepts, even for very simple concept classes. Recall that a (fully-collusion-resistant) fingerprinting code is a scheme for distributing codewords \( c_1, \ldots, c_n \) to \( n \) users that can be uniquely traced back to each user. Moreover, if any group of users combines its codewords into a pirate codeword \( c' \), then the pirate codeword can still be traced back to one of the users who contributed to it. Of course, without any assumption on how the pirates can produce their combined codeword, no secure tracing is possible. To this end, the pirates are constrained according to a marking assumption, which asserts that the combined codeword must agree with at least one of the pirates' codeword in each position. Namely, at an index \( j \) where \( c_{ij} = b \) for every \( i \in b \), the pirates are constrained to output \( c' \) with \( c'_{j} = b \) as well.

To illustrate our technique, we start with an informal discussion of how the original Boneh-Shaw fingerprinting code yields an \( \tilde{\Omega}(k^{1/3}) \) sample complexity lower bound for multi-learning threshold functions. For parameters \( n \) and \( k \), the \((n, k)\)-Boneh-Shaw codebook is a matrix \( C \in \{0, 1\}^{n\times k} \), whose rows \( c_i \) are the codewords given to users \( i = 1, \ldots, n \). The codebook is built from a number of highly structured columns, where a "column of type \( i \)" consists of \( n \) bits where the first \( i \) bits are set to \( 1 \) and the last \( n - i \) bits are set to \( 0 \). For \( i = 1, \ldots, n - 1 \), each column of type \( i \) is repeated a total of \( k/(n - 1) \) times, and the codebook \( C \) is obtained as a random permutation of these \( k \) columns. The security of the Boneh-Shaw code is a consequence of the secrecy of this random permutation. If a coalition of pirates is missing the codeword of user \( i \), then it is unable to distinguish columns of type \( i - 1 \) from columns of type \( i \). Hence, if a pirate codeword is too consistent with a user \( i \)'s codeword in both the columns of type \( i - 1 \) and the columns of type \( i \), a tracing algorithm can reasonably conclude that user \( i \) contributed to it. Boneh and Shaw showed that such a code is indeed secure for \( k = \tilde{O}(n^2) \).

To see how this fingerprinting code gives a lower bound for multi-learning thresholds, consider
thresholds over the data universe \( \mathcal{X} = \{1, \ldots, |\mathcal{X}|\} \) for \(|\mathcal{X}| \geq n \). The key observation is that each column of the Boneh-Shaw codebook can be obtained as a labeling of the examples \(1, \ldots, n\) by a threshold concept. Namely, a column of type \(i\) is the labeling of \(1, \ldots, n\) by the concept \(c_i\). Now suppose a coalition of users \(T \subseteq [n]\) constructs a dataset \(S\) where each row is an example \(i \in T\) together with the labels \(c_{i1}, \ldots, c_{ik}\) coming from the codeword given to user \(i\). Let \((h_1, \ldots, h_k)\) be the hypotheses produced by running a threshold multi-learner on the dataset. If every user has a bit \(b\) at index \(j\) of her codeword, then the hypothesis produced by the learner must also evaluate to \(b\) on most of the examples. Thus, the empirical averages of the hypotheses \((h_1, \ldots, h_k)\) on the examples can be used to obtain a pirate codeword satisfying the marking assumption. The security of the fingerprinting code, i.e. the fact that this codeword can be traced back to a user \(i \in T\), implies that the learner cannot be differentially private. Hence, \(n\) samples is insufficient for privately learning \(k = \tilde{O}(n^3)\) threshold concepts, giving a sample complexity lower bound of \(\tilde{\Omega}(k^{1/3})\).

The lower bounds in this section are stated for empirical learning, but extend to PAC learning by Theorem 6.2.17. We also remark that they hold against the relaxed privacy notion of label privacy, where differential privacy only needs to hold with respect to changing the labels of one example.

### 6.4.1 Lower Bound for Improper PAC Learning

Recall from Section 2.6 that an \((n, k)\)-fingerprinting code consists of a pair of randomized algorithms \((Gen, Trace)\). The parameter \(n\) is the number of users supported by the fingerprinting code, and \(k\) is the length of the code. The codebook generator \(Gen\) produces a codebook \(C \in \{0, 1\}^{n \times k}\). Each row \(c_i \in \{0, 1\}^k\) of \(C\) is the codeword of user \(i\). For a subset \(T \subseteq [n]\), we let \(C_T\) denote the set \(\{c_i : i \in T\}\) of codewords belonging to users in \(T\). The accusation algorithm \(Trace\) takes as input a pirate codeword \(c'\) and accuses some \(i \in [n]\) (or \(\bot\) if it fails to accuse any user). Our lower bounds for multi-learning follow from constructions of fingerprinting codes with additional structural properties.

**Definition 6.4.1.** Let \(\mathcal{C}\) be a concept class over a domain \(\mathcal{X}\). An \((n, k)\)-fingerprinting code \((Gen, Trace)\) is *compatible with concept class \(\mathcal{C}\)* if there exist \(x_1, \ldots, x_n \in \mathcal{X}\) such that for every codebook \(C\) in the support of \(Gen\), there exist concepts \(c_1, \ldots, c_k\) such that \(c_{ij} = c_j(x_i)\) for every \(i = 1, \ldots, n\) and \(j = 1, \ldots, k\).

**Theorem 6.4.2.** Suppose there exists an \((n, k)\)-fingerprinting code compatible with a concept class \(\mathcal{C}\)
with security $\xi$. Let $\alpha \leq 1/3$, $\beta, \varepsilon > 0$, and $\delta < \frac{1-\xi-\beta}{n} - e^\varepsilon \xi$. Then every (improper) $(\alpha, \beta)$-accurate and $(\varepsilon, \delta)$-differentially private empirical $k$-learner for $C$ requires sample complexity greater than $n$.

The proof of Theorem 6.4.2 follows the ideas sketched above.

**Proof.** Let $(Gen, Trace)$ be an $(n, k)$-fingerprinting code compatible with the concept class $C$, and let $x_1, \ldots, x_n \in \mathcal{X}$ be its associated universe elements. Let $D = (x_1, \ldots, x_n)$ and let $A$ be an $(\alpha, \beta)$-accurate empirical $k$-learner for $C$ with sample complexity $n$. We will use $A$ to design an adversary $A_{FP}$ against the fingerprinting code.

Let $T \subseteq [n]$ be a coalition of users, and consider a codebook $C \leftarrow_r Gen$. The adversary strategy $A_{FP}(C_T)$ begins by constructing a labeled dataset $S = (S_i)_{i=1}^n$ by setting $S_i = (x_i, c_{i1}, \ldots, c_{ik})$ for each $i \in T$ and to a nonce row for $i \notin T$. It then runs $A(S)$ obtaining hypotheses $(h_1, \ldots, h_k)$. Finally, it computes for each $j = 1, \ldots, k$ the averages

$$h_j(D) = \frac{1}{n} \sum_{i=1}^n h_j(x_i)$$

and produces a pirate word $c'$ by setting each $c'_j$ to the value of $a_j$ rounded to 0 or 1.

Now consider the coalition $T = [n]$. Since the fingerprinting code is compatible with $C$, each column $(c_{1j}, \ldots, c_{nj}) = (c_j(x_1), \ldots, c_j(x_n))$ for some concept $c_j \in C$. Thus, if the hypotheses $(h_1, \ldots, h_k)$ are $\alpha$-accurate for $(c_1, \ldots, c_k)$ on $S$, then $c' \in F(C_T) = F(C)$. Therefore, by the completeness property of the code and the $(\alpha, \beta)$-accuracy of $A$, we have

$$\Pr[Trace(A_{FP}(C)) \neq \bot] \geq 1 - \xi - \beta.$$ 

In particular, there exists an $i^*$ for which

$$\Pr[Trace(A_{FP}(C)) = i^*] \geq \frac{1 - \xi - \beta}{n}.$$ 

On the other hand, by the soundness property of the code,

$$\Pr[Trace(A_{FP}(C_{\bot})) = i^*] \leq \xi.$$ 

Thus, $A$ cannot be $(\varepsilon, \delta)$-differentially private whenever

$$\frac{1 - \xi - \beta}{n} > e^\varepsilon \cdot \xi + \delta.$$
Remark 6.4.3. If we additionally assume that there exists an element \( x_0 \in \mathcal{X} \) with \( c_1(x_0) = c_2(x_0) = \ldots = c_k(x_0) \), then we can use a “padding” argument to obtain a stronger lower bound of \( n/3\alpha \). More specifically, suppose \( c_1(x_0) = \ldots = c_k(x_0) = 0 \). We pad the dataset \( S \) constructed above with \((1/3\alpha - 1)n\) copies of the junk row \((x_0, 0, \ldots, 0)\). Now if a hypothesis \( \mathcal{H} \) is \( \alpha \)-accurate for a 0-marked column, it’s empirical average will be at most \( \alpha \). On the other hand, an \( \alpha \)-accurate hypothesis for a 1-marked column will have empirical average at least \( 2\alpha \). Since there is a gap between these two quantities, a pirate algorithm can still turn an accurate vector of \( k \) hypotheses into a feasible codeword.

As observed earlier, the \((n, k)\)-Boneh-Shaw code is compatible with the concept class \( \text{Thresh}_\mathcal{X} \) for any \(|\mathcal{X}| \geq n\). Thus, instantiating Theorem 6.4.2 (and Remark 6.4.3) with the Boneh-Shaw code yields a lower bound for \( k \)-learning thresholds.

Lemma 6.4.4 ([21]). Let \( \mathcal{X} \) be a totally ordered domain with \(|\mathcal{X}| \geq n\) for some \( n \in \mathbb{N} \). Then there exists an \((n, k)\)-fingerprinting code compatible with the concept class \( \text{Thresh}_\mathcal{X} \) with security \( \xi \) as long as \( k \geq 2n^3 \log(2n/\xi) \).

Corollary 6.4.5. Every improper \((\alpha, \beta)\)-accurate and \((\varepsilon = O(1), \delta = o(1/n))\)-differentially private empirical \( k \)-learner for \( \text{Thresh}_\mathcal{X} \) requires sample complexity \( \min\{|\mathcal{X}|, \tilde{\Omega}(k^{1/3}/\alpha)\} \).

Discussion. Compatibility with a concept class is an interesting measure of the complexity of a fingerprinting code which warrants further attention. Peikert, shelat, and Smith [100] showed that structural constraints (related to compatibility) on a fingerprinting code give a lower bound on its length beyond the general lower bound of \( k = \tilde{\Omega}(n^2) \) for arbitrary fingerprinting codes. In particular, they showed that the length \( k = \tilde{O}(n^3) \) of the Boneh-Shaw code is essentially tight for the “multiplicity paradigm”, where a codebook is a random permutation of a fixed set of columns, each repeated the same number of times. We take this as evidence that our \( \tilde{\Omega}(k^{1/3}) \) lower bound for \( \text{Thresh}_\mathcal{X} \) cannot be improved via compatible fingerprinting codes. However, closing the gap between our lower bound and the upper bound of roughly \( \sqrt{k} \) remains an intriguing open question.
Open Problem 6.4.6 (Privately learning $k$ thresholds). What is the sample complexity of privately learning $k$ threshold functions simultaneously with $(\varepsilon, \delta)$-differential privacy? In particular, is the dependence on $k$ closer to $k^{1/3}$ or to $\sqrt{k}$?

A natural avenue for obtaining stronger poly($k$) lower bounds for private $k$-learning is to identify compatible fingerprinting codes with shorter length. Tardos [112] showed the existence of an $(n, k)$-fingerprinting code of optimal length $k = \tilde{O}(n^2)$ (see Proposition 6.4.10). The construction of his code differs significantly from multiplicity paradigm: for each column $j$ of the Tardos code, a bias $p_j \in (0, 1)$ is sampled from a fixed distribution, and then each bit of the column is sampled i.i.d. with bias $p_j$. Hence, the columns of the Tardos code are supported on all bit vectors in $\{0, 1\}^n$. This means that for a concept class $C$ to be compatible with the $(n, k)$-Tardos code, it must be the case that $\text{VC}(C) \geq n$. Thus, the lower bound one obtains against $k$-learning $C$ only matches the lower bound for PAC learning $C$ (without privacy). It would be very interesting to construct a fingerprinting code of optimal length $k = \tilde{O}(n^2)$ with substantially fewer than $2^n$ column types (and hence compatible with a concept class of VC-dimension smaller than $n$).

Open Problem 6.4.7 (Optimal structured fingerprinting codes). Is there an $(n, k)$-fingerprinting code of optimal length $k = \tilde{O}(n^2)$ that is supported on $2^{o(n)}$ column types? If not, is there a general tradeoff between fingerprinting code length and the number of column types needed to achieve such length?

6.4.2 Lower Bound for Agnostic Learning

In the agnostic learning model, a learner has to perform well even when the columns of a multi-labeled dataset do not correspond to any concept. This allows us to apply the argument of Theorem 6.4.2 without the constraint of compatibility. The result is that any fingerprinting code, in particular one with optimal length, gives an agnostic learning lower bound for any non-trivial concept class.

Theorem 6.4.8. Suppose there exists an $(n, k)$-fingerprinting code with security $\xi$. Let $C$ be a concept class with at least two distinct concepts. Let $\alpha \leq 1/3$, $\beta, \varepsilon > 0$, and $\delta < \frac{1-\xi-\beta}{n} - \varepsilon^2 \xi$. Then every (improper) agnostic $(\alpha, \beta)$-accurate and $(\varepsilon, \delta)$-differentially private empirical $k$-learner for $C$ requires sample complexity greater than $n$. 
Proof. The proof follows in much the same way as that of Theorem 6.4.2. Let \((Gen, Trace)\) be an \((n, k)\)-fingerprinting code, and let \(x \in \mathcal{X}\) be such that there exist \(c_0, c_1 \in \mathcal{C}\) with \(c_0(x) = 0\) and \(c_1(x) = 1\). Let \(\mathcal{A}\) be an agnostic \((\alpha, \beta)\)-accurate empirical \(k\)-learner for \(\mathcal{C}\) with sample complexity \(n\). Define \(A\) the fingerprinting code adversary \(A_{FP}\) just as in Theorem 6.4.2. Namely, \(A_{FP}\) constructs examples of the form \((x, w_{i1}, \ldots, w_{ij})\) with the available rows of the fingerprinting code, runs \(\mathcal{A}\) on the result, and returns the rounded empirical averages of the \(k\) resulting hypotheses.

To show that \(\mathcal{A}\) cannot be \((\varepsilon, \delta)\)-differentially private, it suffices to show that if \(\mathcal{A}\) produces accurate hypotheses \(h_1, \ldots, h_k\), then the pirate codeword produced by \(A_{FP}\) is feasible. To see this, suppose \(h_1, \ldots, h_k\) are accurate, i.e.

\[
\max_{1 \leq i \leq k} \left( \text{err}_{S_i}(h_j) - \min_{c \in \mathcal{C}} \left( \text{err}_{S_i}(c) \right) \right) \leq \alpha.
\]

Let column \(j\) of the codebook \(C\) be 0-marked, i.e. \(c_{ij} = 0\) for all \(i \in [n]\). Recall that \(c_0(x) = 0\), and hence \(\text{err}_{S_i}(c_0) = 0\). Therefore, since hypothesis \(h_j\) is \(\alpha\)-accurate, we have \(\text{err}_{S_i}(h_j) \leq \alpha\). This implies that bit \(c_j'\) of the pirate codeword is 0. An identical argument shows that the bits of the pirate codeword in the 1-marked columns are also 1. Thus, if \(\mathcal{A}\) produces accurate hypotheses, the pirate codeword produced by \(A_{FP}\) is feasible. The rest of the argument in the proof of Theorem 6.4.2 completes the proof.

\[\square\]

Remark 6.4.9. Just as in Remark 6.4.3, a padding argument shows how to obtain a lower bound of \(n/3\alpha\) under some additional assumptions on \(\mathcal{C}\), e.g. if the distinct concepts also share a common point \(x'\) with \(c_0(x') = c_1(x')\).

Proposition 6.4.10 ([112]). For \(n \in \mathbb{N}\) and \(\xi \in (0, 1)\), there exists an \((n, k)\)-fingerprinting code with security \(\xi\) as long as \(k = O(n^2 \log(n/\xi))\).

Corollary 6.4.11. Every improper agnostic \((\alpha, \beta)\)-accurate and \((\varepsilon = O(1), \delta = o(1/n))\)-differentially private empirical \(k\)-learner for Point\(\mathcal{X}\), Thresh\(\mathcal{X}\), Parity\(\alpha\) requires sample complexity \(\min\{|\mathcal{X}|, \Omega(k^{1/2})\}\).

The same proof yields a lower bound for agnostically learning parities under the uniform distribution.

Proposition 6.4.12. Suppose there exists an \((n, k)\)-fingerprinting code with security \(\xi\). Let \(\alpha \leq 1/6, \beta > 0\) and \(d = \log n\). Then every (improper) agnostic \((\alpha, \beta, \varepsilon = O(1), \delta = o(1/n))\)-PAC \(k\)-learner
for Parity_d requires sample complexity Ω(n).

Proof sketch. By Lemma 6.2.17, it is enough to rule out a private empirical learner for a dataset whose n examples are the distinct binary strings in \{0,1\}^d. To do so, we follow the proof of Theorem 6.4.8, highlighting the changes that need to be made. First, we let c_0 be the all-zeroes concept, and let c_1 be an arbitrary other parity function. Second, \mathcal{A}_{FP} instead constructs examples of the form (x_i, c_1, \ldots, c_k) where x_i is the ith binary string. Finally, when converting the hypotheses (h_1, \ldots, h_k) into a feasible codeword, we instead set c_0^j to 0 if h_j(D) ≤ α, and set c_0^j to 1 if h_j(D) ≥ 1/2 - α. This works because, while err_S(c_0^j) = 0 with respect to 0-marked columns, any concept (and in particular, c_1) has error 1/2 with respect to 1-marked columns.

6.5 Examples where the Direct Sum is Optimal

In this section we show several examples for cases where the direct sum is (roughly) optimal. As we saw in Section 6.4, with (ε, δ)-differential privacy, every non-trivial agnostic k-learner requires sample complexity Ω(√k). We can prove a similar result for ε-private learners, that holds even for non-agnostic learners:

Theorem 6.5.1. Let \mathcal{C} be any non-trivial concept class over a domain \mathcal{X} (i.e., |\mathcal{C}| ≥ 2). Every proper or improper (α, β = 1/2, ε)-private PAC k-learner for \mathcal{C} requires sample complexity Ω(k/ε).

In [5, 7, 8], Beimel et al. presented an agnostic proper learner for Point_\mathcal{X} with sample complexity \tilde{O}(α, β, ε, δ(1)) under (ε, δ)-privacy, and an agnostic improper learner for Point_\mathcal{X} with sample complexity \tilde{O}(α, β, ε, δ(1)) under ε-privacy. Hence, using Observation 6.3.2 (direct sum) with their results yields an (α, β, ε, δ)-PAC agnostic proper k-learner for Point_\mathcal{X} with sample complexity \tilde{O}(α, β, ε, δ(\sqrt{k})), and an (α, β, ε)-PAC agnostic improper k-learner for Point_\mathcal{X} with sample complexity \tilde{O}(α, β, ε(k)). As supported by our lower bounds (Corollary 6.4.11 and Theorem 6.5.1), those learners have roughly optimal sample complexity (ignoring the dependency in α, β, ε, δ and logarithmic factors in k).

Proof of Theorem 6.5.1. The proof is based on a packing argument [5, 78]. Let x ∈ \mathcal{X} and f, g ∈ \mathcal{C} be s.t. f(x) ≠ g(x). Let μ denote the constant distribution over \mathcal{X} giving probability 1 to the point x. Note that err_μ(f, g) = 1. Moreover, observe that for every concept \mathcal{H}, if err_μ(h, f) < 1 then h(x) = f(x), and similarly with h, g.
Let \( \mathcal{A} \) be an \((\alpha, \beta, \epsilon)\)-private PAC \( k \)-learner for \( C \) with sample complexity \( n \). For every choice of \( k \) target functions \((c_1, \ldots, c_k) = \bar{c} \in \{f, g\}^k\), let \( S_{\bar{c}} \) denote the \( k \)-labeled dataset containing \( n \) copies of the point \( x \), each of which is labeled by \( c_1, \ldots, c_k \). Without loss of generality, we can assume that on such datasets \( \mathcal{A} \) returns hypotheses in \( \{f, g\} \) (since under \( \mu \) we can replace an arbitrarily chosen hypothesis \( \mathcal{H} \) with \( f \) if \( f(x) = h(x) \) or with \( g \) if \( g(x) = h(x) \)). Therefore, by the utility properties of \( \mathcal{A} \), for every \( \bar{c} = (c_1, \ldots, c_k) \in \{f, g\}^k \) we have that \( \Pr_{\bar{c}}[\mathcal{A}(S_{\bar{c}}) = (c_1, \ldots, c_k)] \geq \frac{1}{2} \). By changing the dataset \( S_{\bar{c}} \) to \( S_{\bar{c}}' \) one row at a time while applying the differential privacy constraint, we see that

\[
\Pr_{\bar{c}}[\mathcal{A}(S_{\bar{c}}) = (c_1', \ldots, c_k')] \geq \frac{1}{2} e^{-en}.
\]

Since the above inequality holds for every two datasets \( S_{\bar{c}} \) and \( S_{\bar{c}}' \), we get

\[
\frac{1}{2} \geq \Pr_{\bar{c}}[\mathcal{A}(S_{\bar{c}}) \neq (c_1, \ldots, c_k)] \geq (2^k - 1)\frac{1}{2} e^{-en}.
\]

Solving for \( n \), this yields \( n = \Omega(k/\epsilon) \).

**Remark 6.5.2.** The above proof could easily be strengthened to show that \( n = \Omega(\frac{k}{\alpha \epsilon}) \), provided that \( C \) contains two concepts \( f, g \) s.t. \( \exists x, y \in \mathcal{X} \) for which \( f(x) \neq g(x) \) and \( f(y) = g(y) \).

The following lemma shows that the sample complexities of properly and improperly learning parities under the uniform distribution are the same. Thus, for showing lower bounds, it is without loss of generality to consider proper learners.

**Lemma 6.5.3.** Let \( \alpha < 1/4 \). Let \( \mathcal{A} \) be a (possibly improper) \((\alpha, \beta, \epsilon, \delta)\)-PAC \( k \)-learner for Parity\(_d\) under the uniform distribution with sample complexity \( n \). Then there exists a proper \((\alpha', 0, \beta, \epsilon, \delta)\)-PAC \( k \)-learner \( \mathcal{A}' \) for Parity\(_d\) (under the uniform distribution) with sample complexity \( n \).

**Proof.** The algorithm \( \mathcal{A}' \) runs \( \mathcal{A} \) and “rounds” each hypothesis \( h_j \) produced to the nearest parity function. That is, it outputs \((h'_1, \ldots, h'_k)\) where \( h'_j \) is a parity function that minimizes \( \Pr_{x \sim \mathcal{U}_d}[h'_j(x) \neq h_j(x)] \). Since this is just post-processing of the differentially private algorithm \( \mathcal{A} \), the proper learner \( \mathcal{A} \) remains \((\epsilon, \delta)\)-differentially private.

Now suppose \((h_1, \ldots, h_k)\) is \( \alpha \)-accurate for parity functions \((c_1, \ldots, c_k)\) on the uniform distribution.
Then for each $j$, 
\[
\Pr_{x \sim U_d} [h'_j(x) \neq c_j(x)] \leq \Pr_{x \sim U_d} [h'_j(x) \neq h_j(x)] + \Pr_{x \sim U_d} [h_j(x) \neq c_j(x)] \\
\leq 2 \Pr_{x \sim U_d} [h_j(x) \neq c_j(x)] \\
\leq 2\alpha.
\]

Hence, $\text{err}_{U_d}(h'_j, c_j) < 1/2$. Since the error of any parity function from $c_j$ (other than $c_j$ itself) is exactly 1/2 under the uniform distribution, we conclude that $(h'_1, \ldots, h'_k)$ is in fact 0-accurate for $(c_1, \ldots, c_k)$.

**Theorem 6.5.4.** Let $\alpha < \frac{1}{4}$. Every $(\alpha, \beta=\frac{1}{2}, \epsilon)$-PAC $k$-learner for Parity$_d$ (under the uniform distribution) requires sample complexity $\Omega(kd/\epsilon)$.

As we saw in Example 6.3.5, applying direct sum for $k$-learning parities results in a proper agnostic $(\alpha, \beta, \epsilon)$-PAC $k$-learner for Parity$_d$ with sample complexity $O_{\alpha,\beta,\epsilon}(kd + k\log k)$. As stated by Theorem 6.5.4, this is the best possible (ignoring logarithmic factors and the dependency in $\alpha, \beta, \epsilon$).

*Proof of Theorem 6.5.4.* The proof is based on a packing argument [5,78]. Let $\mathcal{A}$ be an $(\alpha, \beta, \epsilon)$-PAC $k$-learner for Parity$_d$ with sample complexity $n$. By Lemma 6.5.3, we may assume $\mathcal{A}$ is proper and learns the hidden concepts exactly.

For every choice of $k$ parity functions $(c_1, \ldots, c_k) = \vec{c} \in \text{Parity}_d^k$, let $S_{\vec{c}}$ denote a random $k$-labeled dataset containing $n$ i.i.d. elements from $U_d$, each labeled by $(c_1, \ldots, c_k)$. By the utility properties of $\mathcal{A}$ we have that $\Pr_{U_d, \mathcal{A}}[\mathcal{A}(S_{\vec{c}}) = \vec{c}] \geq \frac{1}{2}$. In particular, for every $\vec{c} \in \text{Parity}_d^k$ there exists a dataset $D_{\vec{c}}$ labeled by $\vec{c}$ s.t. $\Pr_{\mathcal{A}}[\mathcal{A}(S_{\vec{c}}) = \vec{c}] \geq \frac{1}{2}$. By changing the dataset $D_{\vec{c}}$ to $D_{\vec{\vec{c}}}$ one row at a time while applying the differential privacy constraint, we see that 
\[
\Pr_{\mathcal{A}}[\mathcal{A}(D_{\vec{c}}) = \vec{c}] \geq \frac{1}{2}e^{-\epsilon n}.
\]

Since the above inequality holds for every two datasets $D_{\vec{c}}$ and $D_{\vec{\vec{c}}}$, we get 
\[
\frac{1}{2} \geq \Pr_{\mathcal{A}}[\mathcal{A}(D_{\vec{c}}) \neq \vec{c}] \geq (|\text{Parity}_d|^k - 1) \frac{1}{2}e^{-\epsilon n}.
\]

Solving for $n$, this yields $n = \Omega(kd/\epsilon)$. □
Chapter 7

Computational Complexity of Private PAC Learning

Chapters 4 and 6 described situations where some concept classes require more samples to learn privately than they require to learn without privacy. In this chapter, we address the complementary question of whether there is also a \textit{computational} price of differential privacy for learning tasks, for which much less is known. The initial work of Kasiviswanathan et al. [81] identified the basic question of whether any efficiently PAC learnable concept class is also efficiently privately learnable, but only limited progress has been made on this question since then [5, 96].

The main result of this chapter is a negative answer to this question. We exhibit a concept class that is efficiently PAC learnable, but under plausible cryptographic assumptions cannot be learned efficiently and privately. To prove this result, we establish a connection between private learning and \textit{order-revealing encryption}. We construct a new order-revealing encryption scheme with strong correctness properties that may be of independent learning-theoretic and cryptographic interest.

7.1 Results and Techniques

Kasiviswanathan et al. [81] gave a generic “Private Occam’s Razor” algorithm, showing that any concept class $\mathcal{C}$ can be privately (properly) learned using $O(\log |\mathcal{C}|)$ samples. Unfortunately, this algorithm runs in time $\Omega(|\mathcal{C}|)$, which is exponential in the description size of each concept. With an eye toward designing efficient private learners, Blum et al. [13] made the powerful observation
that any efficient learning algorithm in the statistical queries (SQ) framework of Kearns [84] can be efficiently simulated with differential privacy. Moreover, Kasiviswanathan et al. [81] showed that the efficient learner for the concept class of parity functions based on Gaussian elimination can also be implemented efficiently with differential privacy. These two techniques – SQ learning and Gaussian elimination – are essentially the only methods known for computationally efficient PAC learning. The fact that these can both be implemented privately led Kasiviswanathan et al. [81] to ask whether all efficiently learnable concept classes could also be efficiently learned with differential privacy.

Beimel et al. [5] made partial progress toward this question in the special case of pure differential privacy with proper learning, showing that the sample complexity of efficient learners can be much higher than that of inefficient ones. Specifically, they showed that assuming the existence of pseudorandom generators with exponential stretch, there exists for any $\ell(d) = \omega(\log d)$ a concept class over $\{0, 1\}^d$ for which every efficient proper private learner requires $\Omega(d)$ samples, but an inefficient proper private learner only requires $O(\ell(d))$ examples. Nissim [96] strengthened this result substantially for “representation learning,” where a proper learner is further restricted to output a canonical representation of its hypothesis. He showed that, assuming the existence of one-way functions, there exists a concept class that is efficiently representation learnable, but not efficiently privately representation learnable (even with approximate differential privacy). With Nissim’s kind permission, we give the details of this construction in Section 7.5.

Despite these negative results for proper learning, one might still have hoped that any efficiently learnable concept class could be efficiently improperly learned with privacy. Indeed, a number of works have shown that, especially with differential privacy, improper learning can be much more powerful than proper learning. For instance, Beimel et al. [5] showed that under pure differential privacy, the simple class of Point functions (indicators of a single domain element) requires $\Omega(d)$ samples to privately learn properly, but only $O(1)$ samples to privately learn improperly. Moreover, computational separations are known between proper and improper learning even without privacy considerations. Pitt and Valiant [101] showed that unless $\text{NP} = \text{RP}$, $O(1)$-term DNF are not efficiently properly learnable, but they are efficiently improperly learnable [117].

Under plausible cryptographic assumptions, we resolve the question of Kasiviswanathan et al. [81] in the negative, even for improper learners. The assumption we need is the existence of “strongly
correct” order-revealing encryption (ORE) schemes, described in Section 7.1.2.

**Theorem 7.1.1 (Informal).** Assuming the existence of strongly correct ORE, there exists an efficiently computable concept class EncThresh that is efficiently PAC learnable, but not efficiently learnable by any \((\varepsilon, \delta)\)-differentially private algorithm.

We stress that this result holds even for improper learners and for the relaxed notion of approximate differential privacy. However, as our evidence for the existence of strongly correct ORE is still far from definitive, it remains an interesting open problem to reduce the cryptographic assumptions needed to achieve a separation as in Theorem 7.1.1 (See Section 7.1.2 for a discussion of the assumptions we need).

**Open Problem 7.1.2** (Hardness of private PAC learning under weaker assumptions). Can we separate efficient PAC learning and efficient private PAC learning under standard cryptographic assumptions?

We remark that cryptography has played a major role in shaping our understanding of the computational complexity of learning in a number of models (e.g. [85, 86, 108, 117]). It has also been used before to show separations between what is efficiently learnable in different models (e.g. [12, 109]).

### 7.1.1 Our Techniques

We give an informal overview of the construction and analysis of the concept class EncThresh.

We first describe the concept class of thresholds Thresh and its simple PAC learning algorithm. Consider the domain \([N] = \{1, \ldots, N\}\). Given a number \(t \in [N]\), a threshold concept \(c_t\) is defined by \(c_t(x) = 1\) if and only if \(x \leq t\). The concept class of thresholds admits a simple and efficient proper PAC learning algorithm \(L_{\text{Thresh}}\). Given a sample \(\{(x_1, c_t(x_1)), \ldots, (x_n, c_t(x_n))\}\) labeled by an unknown concept \(c_t\), the learner \(L_{\text{Thresh}}\) identifies the largest positive example \(x_{i^*}\) and outputs the hypothesis \(h = c_{x_{i^*}}\). That is, \(L_{\text{Thresh}}\) chooses the threshold concept that minimizes the empirical error on its sample. To achieve a small constant error on any underlying distribution on examples, it suffices to take \(n = O(1)\) samples. Moreover, this learner can be modified to guarantee differential privacy by instead randomly sampling a threshold hypothesis with probability that decays exponentially in the
empirical error of the hypothesis [81,91]. The sampling can be performed in polynomial time, and requires only a modest blow-up in the learner’s sample complexity.

A simple but important observation about $L_{\text{Thresh}}$ – which, crucially, is not true of the differentially private version – is that it is completely oblivious to the actual numeric values of its examples, or even to the fact that the domain is $[N]$. In fact, $L_{\text{Thresh}}$ works equally well on any totally-ordered domain on which it can efficiently compare examples. In an extreme case, the learner $L_{\text{Thresh}}$ still works when its examples are encrypted under an order-revealing encryption (ORE) scheme, which intuitively guarantees that $L_{\text{Thresh}}$ is able to learn the order of its examples, but nothing else about them. Up to small technical modifications, our concept class EncThresh is exactly the class Thresh where examples are encrypted under an ORE scheme.

For EncThresh to be efficiently PAC learnable, it must be learnable even under distributions that place arbitrary weight on examples corresponding to invalid ciphertexts. To this end, we require a “strong correctness” condition on our ORE scheme. The strong correctness condition ensures that all ciphertexts, even those that are not obtained as encryptions of messages, can be compared in a consistent fashion. This condition is not met by current constructions of ORE, and one of the technical contributions of this work is a generic transformation from weakly correct ORE schemes to strongly correct ones.

While a learner similar to $L_{\text{Thresh}}$ is able to efficiently PAC learn the concept class EncThresh, we argue that it cannot do so while preserving differential privacy with respect to its examples. Intuitively, the security of the ORE scheme ensures that essentially the only thing a learner for EncThresh can do is output a hypothesis that compares an example to one it already has. We make this intuition precise by giving an algorithm that traces the hypothesis output by any efficient learner back to one of the examples used to produce it. This formalization builds conceptually on the connection between differential privacy and traitor-tracing schemes (see Section 7.1.3), but requires new ideas to adapt to the PAC learning model.

7.1.2 Order-Revealing Encryption

Motivated by the task of answering range queries on encrypted databases, an order-revealing encryption (ORE) scheme [16,18] is a special type of symmetric key encryption scheme where it is possible to publicly sort ciphertexts according to the order of the plaintexts. More precisely, the
plaintext space of the scheme is the set of integers \([N] = \{1, \ldots, N\}\), and in addition to the \textit{private} encryption and decryption procedures \textbf{Enc}, \textbf{Dec}, there is a public comparison procedure \textbf{Comp} that takes as input two ciphertexts, and reveals the order of the corresponding plaintexts. The notion of \textit{best-possible semantic security}, defined in Boneh et al. [18], intuitively captures the requirement that, given a collection of ciphertexts, no information about the plaintexts is learned, \textit{except} for the ordering.

**Known constructions of order-revealing encryption.** Relatively few constructions of order-revealing encryption are known, and all constructions are currently based on strong assumptions. Order-revealing encryption can be seen as a special case of 2-input \textit{functional encryption}, also known as \textit{property preserving encryption} [99]. In such a scheme, there are several functions \(f_1, \ldots, f_k\), and given two ciphertexts \(c_0, c_1\) encrypting \(m_0, m_1\), it is possible to learn \(f_i(m_0, m_1)\) for all \(i \in [k]\). General \textit{multi-input} functional encryption schemes can be obtained from indistinguishability obfuscation [68] or multilinear maps [18]. It is also possible to build ORE from \textit{single-input} functional encryption with function privacy, which means that \(f\) is kept secret. Such schemes can be built from regular single-input schemes without function privacy [24], and such single-input schemes can also be built from obfuscation [65] or multilinear maps [66].

It is known that the forms of functional encryption discussed above actually imply obfuscation [11], meaning that all the assumptions from which we can currently build order-revealing encryption imply obfuscation. However, we stress that ORE appears to be much, much weaker than obfuscation or functional encryption: only a single, very simple functionality is supported, namely comparison. In particular the functionality does not support evaluating cryptographic primitives on the plaintext, a feature required of essentially all of the interesting applications of obfuscation/functional encryption. Therefore, we conjecture that ORE can actually be based on significantly weaker assumptions:

**Open Problem 7.1.3 (Existence of ORE schemes).** Can the existence of secure order-revealing encryption schemes be based on cryptographic assumptions that are weaker than obfuscation or those based on multilinear maps?

One way or another, it is important to resolve the status of ORE relative to obfuscation and other

\footnote{More generally, any totally-ordered plaintext space can be considered}
strong primitives: if ORE can be based on mild assumptions, it would strengthen our impossibility result (potentially resolving Open Problem 7.1.2), and likely lead to more efficient ORE constructions that can actually be used in practice. If ORE actually implies obfuscation or other similarly strong primitives, then ORE could be a path to building more efficient obfuscation with better security. Our work demonstrates that, in addition to having real-world practical motivations, ORE is also an interesting theoretical object.

Unfortunately, the above constructions are not quite sufficient for our purposes. The issue arises from the fact that our learner needs to work for any distribution on ciphertexts, even distributions whose support includes malformed ciphertexts. Unfortunately, previous constructions only achieve a weak form of correctness, which guarantees that encrypting two messages and then comparing the ciphertexts using \texttt{Comp} produces the same result (with overwhelming probability) as comparing the plaintexts directly. This requirement only specifies how \texttt{Comp} works on valid ciphertexts, namely actual encryptions of messages. Moreover, correctness is only guaranteed for these messages with overwhelming probability, meaning even some valid ciphertexts may cause \texttt{Comp} to misbehave.

For our learner, this weak form of correctness means, for some distributions that place significant weight on bad ciphertexts, the comparison procedure is completely useless, and thus the learner will fail for these distributions.

We therefore need a stronger correctness guarantee. We need that, for any two ciphertexts, the comparison procedure is consistent with decrypting the two ciphertexts and comparing the resulting plaintexts. This correctness guarantee is meaningful even for improperly generated ciphertexts.

We note that none of the existing constructions of order-revealing encryption outlined above satisfy this stronger notion. For the obfuscation-based schemes, ciphertexts consist of obfuscated programs. In these schemes, it is easy to describe invalid ciphertexts where the obfuscated program performs incorrectly, causing the comparison procedure to output the wrong result. In the multilinear map-based schemes, the underlying instantiation use current “noisy” multilinear maps, such as [64]. An invalid ciphertext could, for example, have too much noise, which will cause the comparison procedure to behave unpredictably.

**Obtaining strong correctness.** We first argue that, for all existing ORE schemes, the scheme can be modified so that \texttt{Comp} is correct for all valid ciphertexts. We then give a generic conversion
from any ORE scheme with weakly correct comparison, including the tweaked existing schemes, into a strongly correct scheme. We simply modify the ciphertext by adding a non-interactive zero-knowledge (NIZK) proof that the ciphertext is well-formed, with the common reference string added to the public comparison key. Then the decryption and comparison procedures check the proof(s), and only output the result (either decryption or comparison) if the proof(s) are valid. The (computational) zero-knowledge property of the NIZK implies that the addition of the proof to the ciphertext does not affect security. Meanwhile, NIZK soundness implies that any ciphertext accepted by the decryption and comparison procedures must be valid, and the weak correctness property of the underlying ORE implies that for valid ciphertexts, decryption and comparison are consistent. The result is that comparisons are consistent with decryption for all ciphertexts, giving strong correctness.

As we need strong correctness for every ciphertext, even hard-to-generate ones, we need the NIZK proofs to have perfect soundness, as opposed to computational soundness. Such NIZK proofs were built in [71].

We note also that the conversion outlined above is not specific to ORE, and applies more generally to functional encryption schemes.

7.1.3 Related Work

Hardness of Private Query Release. The results presented in Chapter 2 showed that an arbitrary sequence of counting queries can be answered accurately with differential privacy even when $k$ is exponential in the dataset size $n$. Unfortunately, all of these algorithms that are capable of answering more than $n^2$ queries are inefficient, running in time exponential in the dimensionality of the data. Moreover, several works [22, 50, 115] have gone on to show that this inefficiency is likely inherent.

These computational lower bounds for private query release rely on a connection between the hardness of private query release and traitor-tracing schemes, which was first observed by Dwork et al. [50]. Traitor-tracing schemes were introduced by Chor, Fiat, and Naor [35] to help digital content producers identify pirates as they illegally redistribute content. Traitor-tracing schemes are conceptually analogous to the example reidentification scheme we use to obtain our hardness result for private learning. Instantiating this connection with the traitor-tracing scheme of Boneh, Sahai,
and Waters [20], which relies on certain assumptions in bilinear groups, Dwork et al. [50] exhibited a family of \(2^{\tilde{O}(\sqrt{n})}\) queries for which no efficient algorithm can produce a data structure which could be used to answer all queries in this family. Very recently, Boneh and Zhandry [22] constructed a new traitor-tracing scheme based on indistinguishability obfuscation that yields the same infeasibility result for a family of \(n \cdot 2^{O(d)}\) queries on records of size \(d\). Extending this connection, Ullman [115] constructed a specialized traitor-tracing scheme to show that no efficient private algorithm can answer more than \(\tilde{O}(n^2)\) arbitrary queries that are given as input to the algorithm.

Dwork et al. [50] also showed strong lower bounds against private algorithms for producing synthetic data. Synthetic data generation algorithms produce a new “fake” dataset, whose rows are of the same type as those in the original dataset, with the promise that the answers to some restricted set of queries on the synthetic dataset well-approximate the answers on the original dataset. Assuming the existence of one-way functions, Dwork et al. [50] exhibited an efficiently computable collection of queries for which no efficient private algorithm can produce useful synthetic data. Ullman and Vadhan [116] refined this result to hold even for extremely simple classes of queries.

Nevertheless, the restriction to synthetic data is significant to these results, and they do not rule out the possibility that other privacy-preserving data structures can be used to answer large families of restricted queries. In fact, when the synthetic data restriction is lifted, there are algorithms (e.g. [32, 52, 77, 114]) that answer queries from certain exponentially large families in subexponential time. One can view the problem of synthetic data generation as analogous to proper learning. In both cases, placing natural syntactic restrictions on the output of an algorithm may in fact come at the expense of utility or computational efficiency.

Efficiency of SQ Learning. Feldman and Kanade [62] addressed the question of whether information-theoretically efficient SQ learners – i.e., those making polynomially many queries – could be made computationally efficient. One of their main negative results showed that unless \(\text{NP} = \text{RP}\), there exists a concept class with polynomial query complexity that is not efficiently SQ learnable. Moreover, this concept class is efficiently PAC learnable, which suggests that the restriction to SQ learning can introduce an inherent computational cost.

We show that the concept class EncThresh can be learned (inefficiently) with polynomially many statistical queries. The result of Blum et al. [13] discussed above, showing that SQ learning algorithms
can be efficiently simulated by differentially private algorithms, thus shows that EncThresh also separates SQ learners making polynomially many queries from computationally efficient SQ learners.

**Corollary 7.1.4 (Informal).** Assuming the existence of strongly correct ORE, the concept class EncThresh is efficiently PAC learnable and has polynomial SQ query complexity, but is not efficiently SQ learnable.

While our proof relies on much stronger hardness assumptions than does [62], it reveals ORE as a new barrier to efficient SQ learning. As discussed in more detail in Section 7.3.3, even though their result is about computational hardness, Feldman and Kanade’s choice of a concept class relies crucially on the fact that parities are hard to learn in the SQ model even information-theoretically. By contrast, our concept class EncThresh is computationally hard to SQ learn for a reason that appears fundamentally different than the information-theoretic hardness of SQ learning parities.

**Learning from Encrypted Data.** Several works have developed schemes for training, testing, and classifying machine learning models over encrypted data (e.g. [23,69]). In a model use case, a client holds a sensitive dataset, and uploads an encrypted version of the dataset to a cloud computing service. The cloud service then trains a model over the encrypted data and produces an encrypted classifier it can send back to the client, ideally without learning anything about the examples it received. The notion of privacy afforded to the individuals in the dataset here is complementary to differential privacy. While the cloud service does not learn anything about the individuals in the dataset, its output might still depend heavily on the data of certain individuals.

In fact, our non-differentially private PAC learner for the class EncThresh exactly performs the task of learning over encrypted data, producing a classifier without learning anything about its examples beyond their order (this addresses the difficulty of implementing comparisons from prior work [69]). Thus one can interpret our results as showing that not only are these two notions of privacy for machine learning training complementary, but that they may actually be in conflict. Moreover, the strong correctness guarantee we provide for ORE (which applies more generally to multi-input functional encryption) may help enable the theoretical study of learning from encrypted data in other PAC-style settings.
7.2 Preliminaries

7.2.1 PAC Learning and Private PAC Learning

For each $k \in \mathbb{N}$, let $X_k$ be an instance space (such as $\{0,1\}^k$), where the parameter $k$ represents the size of the elements in $X_k$. Let $C_k$ be a set of boolean functions $\{c : X_k \to \{0,1\}\}$. The sequence $(X_1,C_1),(X_2,C_2),\ldots$ represents an infinite sequence of learning problems defined over instance spaces of increasing dimension. We will generally suppress the parameter $k$, and refer to the problem of learning $C$ as the problem of learning $C_k$ for every $k$.

**Definition 7.2.1** (Efficient PAC Learning [117]). A PAC learner $L$ is efficient if it runs in time polynomial in the size parameter $k$, the representation size of the target concept $c$, and the accuracy parameters $1/\alpha$ and $1/\beta$. Note that a necessary (but not sufficient) condition for $L$ to be efficient is that its sample complexity $n$ is polynomial in the learning parameters.

The technical object that we will use to show our hardness results for differential privacy is what we call an example reidentification scheme. It is analogous to the hard-to-sanitize database distributions [50,116] used in prior works to prove hardness results for private query release, but is adapted to the setting of computational learning. In the first step, an algorithm $Gen_{ex}$ chooses a concept and a sample $S$ labeled according to that concept. In the second step, a learner $L$ receives either the sample $S$ or the sample $S_{-i}$ where an appropriately chosen example $i$ is replaced by a junk example, and learns a hypothesis $h$. Finally, an algorithm $Trace_{ex}$ attempts to use $h$ to identify one of the rows given to $L$. If $Trace_{ex}$ succeeds at identifying such a row with high probability, then it must be able to distinguish $L(S)$ from $L(S_{-i})$, showing that $L$ cannot be differentially private. We formalize these ideas below.

**Definition 7.2.2.** An $(\alpha,\xi)$-example reidentification scheme for a concept class $C$ consists of a pair of algorithms, $(Gen_{ex}, Trace_{ex})$ with the following properties.

$Gen_{ex}(k,n)$ Samples a concept $c \in C_k$ and an associated distribution $D$. Draws i.i.d. examples $x_1,\ldots,x_n \leftarrow D$, and a fixed value $x_0$. Let $S$ denote the labeled sample $((x_1,c(x_1)),\ldots,(x_n,c(x_n)))$, and for any index $i \in [n]$, let $S_{-i}$ denote the sample with the pair $(x_i,c(x_i))$ replaced with $(x_0,c(x_0))$. 

192
\text{Trace}_{\text{ex}}(h)$ takes state shared with $\text{Gen}_{\text{ex}}$ as well as a hypothesis $h$ and identifies an index in $[n]$ (or $\perp$ if none is found).

The scheme obeys the following “completeness” and “soundness” criteria on the ability of $\text{Trace}_{\text{ex}}$ to identify an example given to a learner $L$.

**Completeness.** A good hypothesis can be traced to some example. That is, for every efficient learner $L$,

$$\Pr[\text{err}_D(c, h) \leq \alpha \land \text{Trace}_{\text{ex}}(h) = \perp] \leq \xi.$$  

Here, the probability is taken over $h \leftarrow \pi L(S)$ and the coins of $\text{Gen}_{\text{ex}}$ and $\text{Trace}_{\text{ex}}$.

**Soundness.** For every efficient learner $L$, $\text{Trace}_{\text{ex}}$ cannot trace $i$ from the sample $S_{\neg i}$. That is, for all $i \in [n]$,

$$\Pr[\text{Trace}_{\text{ex}}(h) = i] \leq \xi$$ 

for $h \leftarrow \pi L(S_{\neg i})$.

We may sometimes relax the completeness condition to hold only under certain restrictions on $L$’s output (e.g. $L$ is a proper learner or $L$ is a representation learner). In this case, we say the $(\text{Gen}_{\text{ex}}, \text{Trace}_{\text{ex}})$ is an example reidentification scheme for (properly, representation) learning a class $\mathcal{C}$.

**Theorem 7.2.3.** Let $(\text{Gen}_{\text{ex}}, \text{Trace}_{\text{ex}})$ be an $(\alpha, \xi)$-example reidentification scheme for a concept class $\mathcal{C}$. Then for every $\beta > 0$ and polynomial $n(h)$, there is no efficient $(\varepsilon, \delta)$-differentially private $(\alpha, \beta)$-PAC learner for $\mathcal{C}$ using $n$ samples when

$$\delta < \left( \frac{1 - \beta - \xi}{n} \right) - e^\varepsilon \xi.$$ 

In a typical setting of parameters, we will take $\alpha, \beta, \varepsilon = O(1)$ and $\delta, \xi = o(1/n)$, in which case the inequality in Theorem 7.2.3 will be satisfied for sufficiently large $n$.

**Proof.** Suppose instead that there were a computationally efficient $(\varepsilon, \delta)$-differentially private $(\alpha, \beta)$-PAC learner $L$ for $\mathcal{C}$ using $n$ samples. Then there exists an $i \in [n]$ such that $\Pr[\text{Trace}_{\text{ex}}(L(S)) =
\[ i \geq (1 - \beta - \xi)/n. \] However, since \( L \) is differentially private,

\[
\Pr[\text{Trace}_{\text{ex}}(L(S_{-i})) = i] \geq e^{-\epsilon} \left( \frac{1 - \beta - \xi}{n} - \delta \right) > \xi(n),
\]

which contradicts the soundness of \((\text{Gen}_{\text{ex}}, \text{Trace}_{\text{ex}})\).

\[ \square \]

### 7.2.2 Order-Revealing Encryption

**Definition 7.2.4.** An Order-Revealing Encryption (ORE) scheme is a tuple \((\text{Gen}, \text{Enc}, \text{Dec}, \text{Comp})\) of algorithms where:

- \(\text{Gen}(1^\lambda, 1^\ell)\) is a randomized procedure that takes as inputs a security parameter \(\lambda\) and plaintext length \(\ell\), and outputs a secret encryption/decryption key \(sk\) and public parameters \(\text{pars}\).

- \(\text{Enc}(sk, m)\) is a potentially randomized procedure that takes as input a secret key \(sk\) and a message \(m \in \{0, 1\}^\ell\), and outputs a ciphertext \(c\).

- \(\text{Dec}(sk, c)\) is a deterministic procedure that takes as input a secret key \(sk\) and a ciphertext \(c\), and outputs a plaintext message \(m \in \{0, 1\}^\ell\) or a special symbol \(\bot\).

- \(\text{Comp}(\text{pars}, c_0, c_1)\) is a deterministic procedure that “compares” two ciphertexts, outputting either “\(>\)”, “\(<\)”, “\(=\)”, or \(\bot\).

**Correctness.** An ORE scheme must satisfy two separate correctness requirements:

- **Correct Decryption:** This is the standard notion of correctness for an encryption scheme, which says that decryption succeeds. We will only consider *strongly* correct decryption, which requires that decryption *always* succeeds. For all security parameters \(\lambda\) and message lengths \(\ell\),

\[
\Pr[\text{Dec}(sk, \text{Enc}(sk, m) ) = m : (sk, \text{pars}) \leftarrow \text{Gen}(1^\lambda, 1^\ell)] = 1.
\]

- **Correct Comparison:** We require that the comparison function succeeds. We will consider two notions, namely *strong* and *weak*. In order to define these notions, we first define two auxiliary functions:

- \(\text{Comp}_{\text{plain}}(m_0, m_1)\) is just the plaintext comparison function. That is, for \(m_0 < m_1\),

\[
\text{Comp}_{\text{plain}}(m_0, m_1) = "<", \text{Comp}_{\text{plain}}(m_1, m_0) = ">", \text{and} \text{Comp}_{\text{plain}}(m_0, m_0) = "=
\]

---

194
– Comp\textsubscript{ciph}(sk, c_0, c_1) is a ciphertext comparison function which uses the secret key. If first computes \( m_b = \text{Dec}(sk, c_b) \) for \( b = 0, 1 \). If either \( m_0 = \bot \) or \( m_1 = \bot \) (in other words, if either decryption failed), then Comp\textsubscript{ciph} outputs \( \bot \). If both \( m_0, m_1 \neq \bot \), then the output is Comp\textsubscript{plain}(m_0, m_1).

Now we define our comparison correctness notions:

– **Weakly Correct Comparison**: This informally requires that comparison is consistent with encryption. For all security parameters \( \lambda \), message lengths \( \ell \), and messages \( m_0, m_1 \in \{0, 1\}^\ell \),

\[
\Pr \left[ \begin{array}{c}
\text{Comp}(\text{pars}, c_0, c_1) = \text{Comp}_{\text{plain}}(m_0, m_1) : (sk, \text{pars}) \leftarrow \text{Gen}(1^\lambda, 1^\ell) \\
\quad c_b \leftarrow \text{Enc}(sk, m_b)
\end{array} \right] = 1.
\]

In particular, for correctly generated ciphertexts, Comp never outputs \( \bot \).

– **Strongly Correct Comparison**: This informally requires that comparison is consistent with decryption. For all security parameters \( \lambda \), message lengths \( \ell \), and ciphertexts \( c_0, c_1 \),

\[
\Pr \left[ \begin{array}{c}
\text{Comp}(\text{pars}, c_0, c_1) = \text{Comp}_{\text{ciph}}(sk, c_0, c_1) : (sk, \text{pars}) \leftarrow \text{Gen}(1^\lambda, 1^\ell)
\end{array} \right] = 1.
\]

**Security.** For security, we will consider a relaxation of the “best possible” security notion of Boneh et al. [18]. Namely, we only consider static adversaries that submit all queries at once. “Best possible” security is a modification of the standard notion of CPA security for symmetric key encryption to block trivial attacks. That is, since the comparison function always leaks the order of the plaintexts, the left and right sets of challenge messages must have the same order. In our relaxation where all challenge messages are queried at once, we can therefore assume without loss of generality that the left and right sequences of messages are sorted in ascending order. For simplicity, we will also disallow the adversary from querying on the same message more than once. This gives the following definition:

**Definition 7.2.5.** An ORE scheme (Gen, Enc, Dec, Comp) is **statically secure** if, for all efficient adversaries \( A \), \(|\Pr[W_0] - \Pr[W_1]|\) is negligible, where \( W_b \) is the event that \( A \) outputs 1 in the following experiment:
• \(A\) produces two message sequences \(m_1^{(L)} < m_2^{(L)} < \ldots < m_q^{(L)}\) and \(m_1^{(R)} < m_2^{(R)} < \ldots < m_q^{(R)}\)

• The challenger runs \((sk, pars) \leftarrow \text{Gen}(1^\lambda, 1^\ell)\). It then responds to \(A\) with \(pars\), as well as \(c_1, \ldots, c_q\) where

\[
c_i = \begin{cases} 
\text{Enc}(sk, m_i^{(L)}) & \text{if } b = 0 \\
\text{Enc}(sk, m_i^{(R)}) & \text{if } b = 1 
\end{cases}
\]

• \(A\) outputs a guess \(b'\) for \(b\).

We also consider a weaker definition, which only allows the sequences \(m_i^{(L)}\) and \(m_i^{(R)}\) to differ at a single point:

**Definition 7.2.6.** An ORE scheme \((\text{Gen, Enc, Dec, Comp})\) is *statically single-challenge secure* if, for all efficient adversaries \(A\), \(|\Pr[W_0] - \Pr[W_1]|\) is negligible, where \(W_0\) is the event that \(A\) outputs 1 in the following experiment:

• \(A\) produces a sequence of messages \(m_1 < m_2 < \ldots < m_q\), and challenge messages \(m_L, m_R\) such that \(m_i < m_L < m_R < m_{i+1}\) for some \(i \in [q-1]\).

• The challenger runs \((sk, pars) \leftarrow \text{Gen}(1^\lambda, 1^\ell)\). It then responds to \(A\) with \(pars\), as well as \(c_1, \ldots, c_q\) where \(c_i = \text{Enc}(sk, m_i)\) and

\[
c^* = \begin{cases} 
\text{Enc}(sk, m_L) & \text{if } b = 0 \\
\text{Enc}(sk, m_R) & \text{if } b = 1 
\end{cases}
\]

• \(A\) outputs a guess \(b'\) for \(b\).

We now argue that these two definitions are equivalent up to some polynomial loss in security.

**Theorem 7.2.7.** \((\text{Gen, Enc, Dec, Comp})\) is *statically secure* if and only if it is *statically single-challenge secure*.

**Proof.** We prove that single-challenge security implies many-challenge security through a sequence of hybrids. Each hybrid will only differ in the messages \(m_i\) that are encrypted, and each adjacent hybrid will only differ in a single message. The first hybrid will encrypt \(m_i^{(L)}\), and the last hybrid will encrypt \(m_i^{(R)}\). Thus, by applying the single-challenge security for each hybrid, we conclude that the first and last hybrids are indistinguishable, thus showing many-challenge security.
**Hybrid $j$ for $j \leq q$.**

$$m_i = \begin{cases} 
\min(m_i^{(L)}, m_i^{(R)}) & \text{if } i \leq j \\
m_i^{(L)} & \text{if } i > j 
\end{cases}$$

First, notice that all the $m_i$ are in order since both sequences $m_i^{(L)}$ and $m_i^{(R)}$ are in order. Second, the only difference between Hybrid $(j - 1)$ and Hybrid $j$ is that $m_j = m_j^{(L)}$ in Hybrid $(j - 1)$ and $m_j = \min(m_j^{(L)}, m_j^{(R)})$ in Hybrid $j$. Thus, single-challenge security implies that each adjacent hybrid is indistinguishable. Moreover, for $j$ where $m_j^{(L)} < m_j^{(R)}$, the two hybrids are actually identical.

**Hybrid $j$ for $j > q$.**

$$m_i = \begin{cases} 
\min(m_i^{(L)}, m_i^{(R)}) & \text{if } i \leq 2q - j \\
m_i^{(R)} & \text{if } i > 2q - j 
\end{cases}$$

Again, notice that all the $m_i$ are in order. Moreover, the only different between Hybrid $(2q - j)$ and Hybrid $(2q - j + 1)$ is that $m_j = \min(m_j^{(L)}, m_j^{(R)})$ in Hybrid $(2q - j)$ and $m_j = m_j^{(R)}$ in Hybrid $(2q - j + 1)$. Thus, single-challenge security implies that each adjacent hybrid is indistinguishable. Moreover, for $j$ where $m_j^{(L)} > m_j^{(R)}$, the two hybrids are actually identical.

\[\square\]

### 7.3 EncThresh and its Learnability

Let $(\text{Gen}, \text{Enc}, \text{Dec}, \text{Comp})$ be a statically secure ORE scheme with strongly correct comparison. We define a concept class EncThresh, which intuitively captures the class of threshold functions where examples are encrypted under the ORE scheme. Throughout this discussion, we will take $N = 2^t$ and regard the plaintext space of the ORE scheme to be $[N] = \{1, \ldots, N\}$. Ideally we would like, for each threshold $t \in [N + 1]$ and each $(\text{sk}, \text{pars}) \leftarrow \text{Gen}(1^\lambda)$, to define a concept

$$f_{1,t,\text{sk},\text{pars}}(c) = \begin{cases} 
1 & \text{if } \text{Dec}_{\text{sk}}(c) < t \\
0 & \text{otherwise.} 
\end{cases}$$

However, we need to make a few technical modifications to ensure that EncThresh is efficiently PAC learnable.
1. In order for the learner to be able to use the comparison function $\text{Comp}$, it must be given the public parameters $\text{pars}$ generated by the ORE scheme. We address this in the natural way by attaching a set of public parameters to each example. Moreover, we define $\text{EncThresh}$ so that each concept is supported on the single set of public parameters that corresponds to the secret key used for encryption and decryption.

2. Only a subset of binary strings form valid $(\text{sk}, \text{pars})$ pairs that are actually produced by $\text{Gen}$ in the ORE scheme. To represent concepts, we need a reasonable encoding scheme for these valid pairs. The encoding scheme we choose is the polynomial-length sequence of random coin tosses used by the algorithm $\text{Gen}$ to produce $(\text{sk}, \text{pars})$.

We now formally describe the concept class $\text{EncThresh}$. Each concept is parameterized by a string $r$, representing the coin tosses of the algorithm $\text{Gen}$, and a threshold $t \in [N + 1]$ for $N = 2^\ell$. In what follows, let $(\text{sk}^r, \text{pars}^r)$ be the keys output by $\text{Gen}(1^\lambda, 1^\ell)$ when run on the sequence of coin tosses $r$. Let

$$
 f_{t,r}(\text{pars}, c) = \begin{cases} 
 1 & \text{if } (\text{pars} = \text{pars}^r) \land (\text{Dec}(\text{sk}^r, c) \neq \bot) \land (\text{Dec}(\text{sk}^r, c) < t) \\
 0 & \text{otherwise.}
\end{cases}
$$

Notice that given $t$ and $r$, the concept $f_{t,r}$ can be efficiently evaluated. The description length $k$ of the instance space $X_k = \{0, 1\}^k$ is polynomial in the security parameter $\lambda$ and plaintext length $\ell$.

### 7.3.1 An Efficient PAC Learner for $\text{EncThresh}$

We argue that $\text{EncThresh}$ is efficiently PAC learnable by formalizing the argument from the introduction. Because we need to include the ORE public parameters in each example, the PAC learner $L$ (Algorithm 13) for $\text{EncThresh}$ actually works in two stages. In the first stage, $L$ determines whether there is significant probability mass on examples corresponding to some public parameters $\text{pars}$. Recall that each concept in $\text{EncThresh}$ is supported on exactly one such set of parameters. If there is no significant mass on any $\text{pars}$, then the all-zeroes hypothesis is a good hypothesis. On the other hand, if there is a heavy set of parameters, the learner $L$ applies $\text{Comp}$ using those parameters to learn a good comparator.
**Theorem 7.3.1.** Let $\alpha, \beta > 0$. There exists a PAC learning algorithm $L$ for the concept class EncThresh achieving error $\alpha$ and confidence $1 - \beta$. Moreover, $L$ is efficient (running in time polynomial in the parameters $k, 1/\alpha, \log(1/\beta)$).

**Algorithm 11** Learner $L$ for EncThresh

1. Request examples $\{((\text{pars}_1, c_1, b_1), \ldots, (\text{pars}_n, c_n, b_n))\}$ for $n = \lfloor \log(1/\beta)/\alpha \rfloor$.
2. Identify an $i$ for which $b_i = 1$ and set $\text{pars}^* = \text{pars}_i$; if no such $i$ exists, return $h \equiv 0$.
3. Let $G = \{j : \text{pars}_j = \text{pars}^*, b_j = 1\}$. Let $j^* \in G$ be an index with $\text{Comp}(\text{pars}^*, c, c_j^*) \in \{<, =, \perp\}$ for all $j \in G$.
4. Return $h$ defined by

$$h(\text{pars}, c) = \begin{cases} 1 & \text{if } (\text{pars} = \text{pars}^*) \land (\text{Comp}(\text{pars}^*, c, c_j^*) \in \{<, =\}) \\ 0 & \text{otherwise.} \end{cases}$$

**Proof.** Fix a target concept $f_{t,r} \in \text{EncThresh}_k$ and a distribution $D$ on examples. First observe that the learner $L$ always outputs a hypothesis with one-sided error, i.e. we always have $h \leq f_{t,r}$ pointwise. Also observe that $f_{t',r} \leq f_{t,r}$ pointwise for any $t' < t$. These both follow from the strong correctness of the ORE scheme. Let $(\text{sk}^*, \text{pars}^*)$ denote the keys output by $\text{Gen}(1^\lambda, 1^t)$ when run on the sequence of coin tosses $r$. Let POS denote the set of examples $(\text{pars}, c)$ on which $f_{t,r}(\text{pars}, c) = 1$.

We divide the analysis of the learner into two cases based on the weight $D$ places on POS.

**Case 1:** $D$ places weight at least $\alpha$ on POS. Define $\hat{t} \in [N + 1]$ as the largest $\hat{t} \leq t$ such that $\text{err}_D(f_{\hat{t},r}, f_{t,r}) \geq \alpha$. Such a $\hat{t}$ is guaranteed to exist since $f_{0,r}$ is the all-zeros function, and therefore $\text{err}_D(f_{0,r}, f_{t,r})$ is equal to the weight $D$ places on POS, which is at least $\alpha$.

Suppose $f_{t+1,r} \leq h$ pointwise. Since $h$ has one-sided error (that is, $h \leq f_{t,r}$ pointwise), we have $\text{err}_D(f_{t+1,r}, f_{t,r}) = \text{err}_D(f_{t+1,r}, h) + \text{err}_D(h, f_{t,r})$, or

$$\text{err}_D(h, f_{t,r}) = \text{err}_D(f_{t+1,r}, f_{t,r}) - \text{err}_D(f_{t+1,r}, h) \leq \text{err}_D(f_{t+1,r}, f_{t,r}) < \alpha.$$

Therefore, it suffices to show that $f_{t+1,r} \leq h$ with probability at least $1 - \beta$. This is guaranteed as long as $L$ receives a sample $(\text{pars}^*, c_i)$ with $\hat{t} \leq \text{Dec}(\text{sk}^*, c_i) < t$. In other words, $f_{t,r}(\text{pars}^*, c_i) = 1$ and $f_{t,r}(\text{pars}^*, c_i) = 0$. Since $f_{t,r} \leq f_{t,r}$ pointwise, such samples exactly account for the error between $f_{t,r}$ and $f_{t,r}$. Thus since $\text{err}_D(f_{t,r}, f_{t,r}) \geq \alpha$, for each $i$ it must be that $\hat{t} \leq \text{Dec}(\text{sk}^*, c_i) < t$ with
probability at least $\alpha$. The learner $L$ therefore receives some sample $c_i$ with $\hat{t} \leq \text{Dec}({sk^T}, c_i) < t$ with probability at least $1 - (1 - \alpha)^n \geq 1 - \beta$ (since we took $n \geq \log(1/\beta)/\alpha$).

**Case 2:** $D$ places less than $\alpha$ weight on POS. Then the identically zero hypothesis has error at most $\alpha$, so the claim holds because $0 \leq h \leq f_{t,r}$.

### 7.3.2 Hardness of Privately Learning EncThresh

We now prove the hardness of privately learning EncThresh by constructing an example reidentification scheme for this concept class. Recall that an example reidentification scheme consists of two algorithms, $\text{Gen}_{\text{ex}}$, which selects a distribution, a concept, and examples to give to a learner, and $\text{Trace}_{\text{ex}}$ which attempts to identify one of the examples the learner received.

Our example reidentification scheme yields a hard distribution even for *weak-learning*, where the error parameter $\alpha$ is taken to be inverse-polynomially close to $1/2$.

**Theorem 7.3.2.** Let $\gamma(n)$ and $\xi(n)$ be noticeable functions. Let $(\text{Gen}, \text{Enc}, \text{Dec}, \text{Comp})$ be a statically single-challenge secure ORE scheme. Then there exists an (efficient) $(\alpha = \frac{1}{2} - \gamma, \xi)$-example reidentification scheme $(\text{Gen}_{\text{ex}}, \text{Trace}_{\text{ex}})$ for the concept class EncThresh.

We start with an informal description of the scheme $(\text{Gen}_{\text{ex}}, \text{Trace}_{\text{ex}})$. The algorithm $\text{Gen}_{\text{ex}}$ sets up the parameters of the ORE scheme, chooses the “middle” threshold concept corresponding to $t = N/2$, and sets the distribution on examples to be encryptions of uniformly random messages (together with the correct public parameters needed for comparison). Let $m_1 < m_2 < \ldots < m_n$ denote the sorted sequence of messages whose encryptions make up the sample produced by $\text{Gen}_{\text{ex}}$ (with overwhelming probability, they are indeed distinct). We can thus break the plaintext space up into buckets of the form $B_i = [m_i, m_{i+1})$. Suppose $L$ is a (weak) learner that produces a hypothesis $h$ with advantage $\gamma$ over random guessing. Such a hypothesis $h$ must be able to distinguish encryptions of messages $m \leq t$ from encryptions of messages $m > t$ with advantage $\gamma$. Thus, there must be a pair of adjacent buckets $B_{i-1}, B_i$ for which $h$ can distinguish encryptions of messages from $B_{i-1}$ from encryptions from $B_i$ with advantage $\frac{\gamma}{n}$.

This observation leads to a natural definition for $\text{Trace}_{\text{ex}}$: locate a pair of adjacent buckets $B_{i-1}, B_i$ that $h$ distinguishes, and output the identity $i$ of the example separating those buckets.
Completeness of the resulting scheme, i.e. the fact that some example is reidentified when \( L \) succeeds, follows immediately from the preceding discussion. We argue soundness, i.e. that an example absent from \( L \)'s sample is not identified, by reducing to the static security of the ORE scheme. The intuition is that if \( L \) is not given example \( i \), then it should not be able to distinguish encryptions from bucket \( B_{i-1} \) from encryptions from bucket \( B_i \).

To make the security reduction somewhat more precise, suppose for the sake of contradiction that there is an efficient algorithm \( L \) that violates the soundness of \((\text{Gen}_{\text{ex}}, \text{Trace}_{\text{ex}})\) with noticeable probability \( \xi \). That is, there is some \( i \) such that even without example \( i \), the algorithm \( L \) manages to produce (with probability \( \xi \)) a hypothesis \( h \) that distinguishes \( B_{i-1} \) from \( B_i \). A natural first attempt to violate the security of the ORE is to construct an adversary that challenges on the message sequences \( m_1 < \ldots < m_{i-1} < m_i^{(L)} < m_{i+1}, \ldots, m_n \) and \( m_1 < \ldots < m_{i-1} < m_i^{(R)} < m_{i+1} < \ldots < m_n \), where \( m_i^{(L)} \) is randomly chosen from \( B_{i-1} \) and \( m_i^{(R)} \) is randomly chosen from \( B_i \). Then if \( h \) can distinguish \( B_{i-1} \) from \( B_i \), the adversary can distinguish the two sequences. Unfortunately, this approach fails for a somewhat subtle reason. The hypothesis \( h \) is only guaranteed to distinguish \( B_{i-1} \) from \( B_i \) with probability \( \xi \). If \( h \) fails to distinguish the buckets – or distinguishes them in the opposite direction – then the adversary’s advantage is lost.

To overcome this issue, we instead rely on the security of the ORE for sequences that differ on two messages. For the “left” challenge, our adversary samples two messages from the same randomly chosen bucket, \( B_{i-1} \) or \( B_i \) (in addition to requesting encryptions of \( m_1, \ldots, m_{i-1}, m_i, \ldots, m_n \)). For the “right” challenge, it samples one message from each bucket \( B_{i-1} \) and \( B_i \). Let \( c^0 \) and \( c^1 \) be the ciphertexts corresponding to the challenge messages. If \( h \) agrees on \( c^0 \) and \( c^1 \), then this suggests the messages are from the same bucket, and the adversary should guess “left”. On the other hand, if \( h \) disagrees on \( c^0 \) and \( c^1 \), then the adversary should guess “right”. If \( h \) distinguishes the buckets \( B_{i-1} \) and \( B_i \), this adversary does strictly better than random guessing. On the other hand, even if \( h \) fails to distinguish the buckets, the adversary does at least as well as random guessing. So overall, it still has a noticeable advantage at the ORE security game.

We now give the formal proof of Theorem 7.3.2.

\textbf{Proof.} We construct an example reidentification scheme for EncThresh as follows. The algorithm \( \text{Gen}_{\text{ex}} \) fixes the threshold \( t = N/2 \) and samples \((\text{sk}^r, \text{pars}^r) \leftarrow \text{Gen}(1^\lambda, 1^\xi)\), yielding a concept \( f_{1,f} \). Let
\( \mathcal{D} \) be the distribution \((\text{pars}^*, \text{Enc}(\text{sk}^r, m))\) for uniformly random \(m \in [N]\). Let \(m'_1, \ldots, m'_n \leftarrow_r [N]\), and let \(m_1 \leq \ldots \leq m_n\) be the result of sorting the \(m'_i\). Let \(m_0 = 0\) and \(m_{n+1} = N\). Since \(n = \text{poly}(k) \ll N\), these random messages will be well-spaced. In particular, with overwhelming probability, \(|m_{i+1} - m_i| > 1\) for every \(i\), so we assume this is the case in what follows. \(\text{Gen}_{\text{ex}}\) then sets the samples to be \((x_1 = (\text{pars}^r, \text{Enc}(\text{sk}^r, m'_1)), \ldots, x_n = (\text{pars}^r, \text{Enc}(\text{sk}^r, m'_n)))\). Let \(x_0 = (\text{pars}^r, \text{Enc}(\text{sk}^r, m_0))\) be a “junk” example.

The algorithm \(\text{Trace}_{\text{ex}}\) creates buckets \(B_i = [m_i, m_{i+1})\). For each \(i\), let

\[
p_i = \frac{1}{K} \sum_{j=1}^{K} h(x_j)
\]

where \(x_j = (\text{pars}^r, \text{Enc}(\text{sk}^r, m_j))\) for i.i.d. \(m_1, \ldots, m_K \leftarrow_r B_i\). Then \(|\hat{p}_i - p_i| \leq \frac{\gamma}{n}\) for every \(i\) with probability at least \(1 - \xi/4\).

\[\text{Lemma 7.3.3.} \quad \text{Let } K = \frac{8\gamma^2 \log(9n/\xi)}{\gamma^2}. \text{ For each } i = 0, \ldots, n, \text{ let}\]

\[
\hat{p}_i = \frac{1}{K} \sum_{j=1}^{K} h(x_j)
\]

\[\text{Proof.} \quad \text{By a Chernoff bound, the probability that any given } \hat{p}_i \text{ deviates from } p_i \text{ by more than } \frac{\gamma}{n} \text{ is at most } 2 \exp(-K\gamma^2/8n^2) \leq \frac{\xi}{4(n+1)}. \text{ The lemma follows by a union bound.} \]

We first verify completeness for this scheme. Let \(L\) be a learner for \(\text{EncThresh}\) using \(n\) examples.

If the hypothesis \(h\) produced by \(L\) is \((\frac{1}{2} - 2\gamma)\)-good, then there exists \(i_0 < i_1\) such that \(p_{i_0} - p_{i_1} \geq 2\gamma\). If this is the case, then there must be an \(i\) for which \(p_{i-1} - p_i \geq \frac{2\gamma}{n}\). Then with probability all but \(\xi(n)/2\) over the estimates \(\hat{p}_i\), we have \(\hat{p}_{i-1} - \hat{p}_i \geq \frac{\gamma}{n}\), so some index is accused.

Now we verify soundness. Fix a PPT \(L\), and let \(j^* \in [n]\). Suppose \(L\) violates the soundness of the scheme with respect to \(j^*\), i.e.

\[
\Pr_{h \leftarrow_r L(S_{j^*}), \text{coins of } \text{Gen}_{\text{ex}}} [\text{Trace}_{\text{ex}}(h) = j^*] > \xi.
\]

We will use \(L\) to construct an adversary \(\mathcal{A}\) for the ORE scheme that succeeds with noticeable
advantage. It suffices to build an adversary for the static (many-challenge) security of ORE, with Theorem 7.2.7 showing how to convert it to a single-challenge adversary. This many-challenge success probability is:

$$p = \sum_{j=1}^{m} \text{advantage.}$$

It suffices to build an adversary for the static (many-challenge) security of ORE, with a random guess whenever the messages it samples are not well-spaced.)

**Algorithm 12** ORE adversary $A$

1. Sample $m'_1, \ldots, m'_n \leftarrow [N]$, and let $m_1 \leq \ldots \leq m_n$ be the result of sorting the $m'_j$. Let $\pi$ be the permutation on $\{1, \ldots, n\}$ such that $m_{\pi(j)} = m'_j$. Let $m_0 = 0$. Let $i^* = \pi(j^*)$ so that $m_{i^*} = m'_j$.

2. Construct pairs $(m^0_L, m^1_L)$ and $(m^0_R, m^1_R)$ as follows. Let $B_0 = (m_{i^*-1}, m_{i^*})$ and $B_1 = (m_{i^*}, m_{i^*+1})$. Sample $m^0_L \leq m^1_L$ at random from the same $B_j$, for a random choice of $j \in \{0, 1\}$. Sample $m^0_R \leftarrow_R B_0$ and $m^1_R \leftarrow_R B_1$.

3. Challenge on the pair of sequences $m_0, m_1, \ldots, m_{i^*-1}, m^1_L, m^2_L, m_{i^*}, \ldots, m_n$ and $m_0, m_1, \ldots, m_{i^*-1}, m^1_R, m^2_R, m_{i^*}, \ldots, m_n$, receiving ciphertexts $c_1, \ldots, c^0_{i^*}, c^1_{i^*}, \ldots, c_n$. For $j \neq j^*$, let $c'_j = c_{\pi(j)}$ so that $c'_j$ is an encryption of $m'_j$.

4. Set $t = N/2$ and let

$$S_{-j^*} = \{(\text{pars}^r, c'_1, \chi(m'_1 \leq t)), \ldots, (\text{pars}^r, c'_{j^*-1}, \chi(m'_{j^*-1} \leq t)), (\text{pars}^r, c_0, 1), (\text{pars}^r, c'_{j^*+1}, \chi(m'_{j^*+1} \leq t)), \ldots, (\text{pars}^r, c_n, \chi(m'_n \leq t))\}$$

$$= \{(\text{pars}^r, c_{\pi(1)}, \chi(m_{\pi(1)} \leq t)), \ldots, (\text{pars}^r, c_{\pi(j^*-1)}, \chi(m_{\pi(j^*-1)} \leq t)), (\text{pars}^r, c_0, 1), (\text{pars}^r, c_{\pi(j^*+1)}, \chi(m_{\pi(j^*+1)} \leq t)), \ldots, (\text{pars}^r, c_{\pi(n)}, \chi(m_{\pi(n)} \leq t))\}$$

Obtain $h \leftarrow_R L(S_{-j^*})$.

5. Guess $b' = 0$ if $h(\text{pars}^r, c^0_{i^*}) = h(\text{pars}^r, c^1_{i^*})$. Otherwise guess $b' = 1$.

Let $i^*$ be such that $m_{i^*} = m'_j$. With probability at least $\xi$ over the parameters $(sk^r, \text{pars}^r)$, the choice of messages, the choice of the hypothesis $h$, and the coins of Trace$_{ex}$, there is a gap $p_{i^*+1} - p_{i^*} \geq \frac{\gamma}{n}$. Hence, by Lemma 7.3.3, there is a gap $p_{i^*+1} - p_{i^*} \geq \frac{\gamma}{2n}$ with probability at least $\frac{\xi}{2}$.

We now calculate the advantage of the adversary $A$. Fix a hypothesis $h$. For notational simplicity, let $p = p_{i^*-1}$ and let $q = p_{i^*}$. Let $y_0 = h(\text{pars}^r, c^0_{i^*})$ and $y_1 = h(\text{pars}^r, c^1_{i^*})$. Then the adversary’s success probability is:

203
\[
\begin{align*}
\Pr[b' = b] &= \frac{1}{2} \left( \Pr[y_0 = y_1 | b = 0] + \Pr[y_0 \neq y_1 | b = 1] \right) \\
&= \frac{1}{2} \left( \frac{1}{2} (p^2 + (1-p)^2 + q^2 + (1-q)^2) + (1 - pq - (1-p)(1-q)) \right) \\
&= \frac{1}{2} + \frac{1}{2} (p - q)^2.
\end{align*}
\]

Thus if \( p - q \geq \frac{3}{2n} \), then the adversary’s advantage is at least \( \frac{3}{4n^2} \). On the other hand, even for arbitrary values of \( p, q \), the advantage is still nonnegative. Therefore, the advantage of the strategy is at least \( \frac{3}{8n} - \text{negl}(k) \) (the \( \text{negl}(k) \) term coming from the assumption that the \( m'_i \) sampled where distinct), which is a noticeable function of the parameter \( k \). This contradicts the static security of the ORE scheme.

\[
\square
\]

### 7.3.3 The SQ Learnability of EncThresh

The statistical query (SQ) model is a natural restriction of the PAC model by which a learner is able to measure statistical properties of its examples, but cannot see the individual examples themselves. We recall the definition of an SQ learner.

**Definition 7.3.4** (SQ learning [84]). Let \( c : X \to \{0,1\} \) be a target concept and let \( \mathcal{D} \) be a distribution over \( X \). In the SQ model, a learner is given access to a statistical query oracle \( \text{STAT}(c, \mathcal{D}) \). It may make queries to this oracle of the form \( (\psi, \tau) \), where \( \psi : X \times \{0,1\} \to \{0,1\} \) is a query function and \( \tau \in (0,1) \) is an error tolerance. The oracle \( \text{STAT}(c, \mathcal{D}) \) responds with a value \( v \) such that \( |v - \Pr_{x \in \mathcal{D}}[\psi(x, c(x)) = 1]| \leq \tau \). The goal of a learner is to produce, with probability at least \( 1 - \beta \), a hypothesis \( h : X \to \{0,1\} \) such that \( \text{err}_{\mathcal{D}}(c, h) \leq \alpha \). The query functions must be efficiently evaluable, and the tolerance \( \tau \) must be lower bounded by an inverse polynomial in \( k \) and \( 1/\alpha \).

The query complexity of a learner is the worst-case number of queries it issues to the statistical query oracle. An SQ learner is efficient if it also runs in time polynomial in \( k, 1/\alpha, 1/\beta \).

Feldman and Kanade [62] investigated the relationship between query complexity and computational complexity for SQ learners. They exhibited a concept class \( \mathcal{C} \) which is efficiently PAC learnable
and SQ learnable with polynomially many queries, but assuming \( \text{NP} \neq \text{RP} \), is not efficiently SQ learnable. Concepts in this concept class take the form

\[
g_{\phi, y}(x, x') = \begin{cases} 
\text{PAR}_y(x') & \text{if } x = \phi \\
0 & \text{otherwise.}
\end{cases}
\]

Here, \( \text{PAR}_y(x') \) is the inner product of \( y \) and \( x' \) modulo 2. The concept class \( C \) consists of \( g_{\phi, y} \) where \( \phi \) is a satisfiable 3-CNF formula and \( y \) is the lexicographically first satisfying assignment to \( \phi \). The efficient PAC learner for parities based on Gaussian elimination shows that \( C \) is also efficiently PAC learnable. It is also (inefficiently) SQ learnable with polynomially many queries: either the all-zeros hypothesis is good, or an SQ learner can recover the formula \( \phi \) bit-by-bit and determine the satisfying assignment \( y \) by brute force. On the other hand, because parities are information-theoretically hard to SQ learn, the satisfying assignment \( y \) remains hidden to an SQ learner unless it is able to solve 3-SAT.

In this section, we show that the concept class EncThresh shares these properties with \( C \). Namely, we know that EncThresh is efficiently PAC learnable and because it is not efficiently privately learnable, it is not efficiently SQ learnable [13]. We can also show that EncThresh has an SQ learner with polynomial query complexity. Making this observation about EncThresh is of interest because the hardness of SQ learning EncThresh does not seem to be related to the (information-theoretic) hardness of SQ learning parities.

**Proposition 7.3.5.** The concept class EncThresh is (inefficiently) SQ learnable with polynomially many queries.

As with \( C \) there are two cases. In the first case, the target distribution places nearly zero weight on examples with \( \text{pars} = \text{pars}^r \), and so the all-zeros hypothesis is good. In the second case, the target distribution places noticeable weight on these examples, and our learner can use statistical queries to recover the comparison parameters \( \text{pars}^r \) bit-by-bit. Once the public parameters are recovered, our learner can determine a corresponding secret key by brute force. Lemma 7.3.6 below shows that any corresponding secret key – even one that is not actually \( \text{sk}^r \) – suffices. The learner can then use binary search to determine the threshold value \( t \).

**Proof.** Let \( f_{t, r} \) be the target concept, \( \mathcal{D} \) be the target distribution, and \( \alpha \) be the target error rate.
With the statistical query \((x \times b \mapsto b, \alpha/4)\), we can determine whether the all-zeros hypothesis is accurate. That is, if we receive a value that is less than \(\alpha/2\), then \(\Pr_{x \in D}[f_{t,r}(x) = 1] \leq \alpha\). If not, then we know that \(\Pr_{x \in D}[f_{t,r}(x) = 1] \geq \alpha/4\), so \(D\) places significant weight on examples prefixed with \(\text{pars}'\). Suppose now that we are in the latter case.

Let \(m = |\text{pars}|.\) For \(i = 1, \ldots, m\), define \(\psi_i(\text{pars}, c, b) = 1\) if \(\text{pars}_i = 1\) and \(b = 1\), and \(\psi_i(\text{pars}, c, b) = 0\) otherwise. Then by asking the queries \((\psi_i, \alpha/16)\), we can determine each bit \(\text{pars}'_i\) of \(\text{pars}'\).

Now by brute force search, we determine a secret key \(\text{sk}\) for which \((\text{sk}, \text{pars}') \in \text{Range}(\text{Gen})\). The recovered secret key \(\text{sk}\) may not necessarily be the same as \(\text{sk}'\). However, the following lemma shows that \(\text{sk}\) and \(\text{sk}'\) are functionally equivalent:

**Lemma 7.3.6.** Suppose \((\text{Gen}, \text{Enc}, \text{Dec}, \text{Comp})\) is a strongly correct ORE scheme. Then for any pair \((\text{sk}_1, \text{pars}), (\text{sk}_2, \text{pars}) \in \text{Range}(\text{Gen})\), we have that \(\text{Dec}_{\text{sk}_1}(c) = \text{Dec}_{\text{sk}_2}(c)\) for all ciphertexts \(c\).

With the secret key \(\text{sk}\) in hand, we now conduct a binary search for the threshold \(t\). Recall that we have an estimate \(v\) for the weight that \(f_{t,r}\) places on positive examples, i.e. \(|v - \Pr_{x \in D}[f_{t,r}(x) = 1]| \leq \alpha/4\). Starting at \(t_1 = N/2\), we issue the query \((\varphi_1, \alpha/4)\) where \(\varphi_1(\text{pars}, c, b) = 1\) iff \(\text{pars} = \text{pars}'\) and \(\text{Dec}(\text{sk}, c) < t\). Let \(h_{t_1}\) denote the hypothesis

\[
h_{t_1}(\text{pars}, c) = \begin{cases} 
1 & \text{if } (\text{pars} = \text{pars}') \land (\text{Dec}(\text{sk}, c) \neq \bot) \land (\text{Dec}(\text{sk}, c) < t_1) \\
0 & \text{otherwise.}
\end{cases}
\]

Thus, the query \((\varphi_1, \alpha/4)\) approximates the weight \(h_{t_1}\) places on positive examples. Let the answer to this query be \(v_1\). If \(|v_1 - v| \leq \alpha/2\), then we can halt and output the good hypothesis \(h_{t_1}\). Otherwise, if \(v_1 < v - \alpha/2\), we set the next threshold to \(t_2 = 3N/4\), and if \(v_1 > v + \alpha/2\), we set the next threshold to \(t_2 = N/4\). We recurse up to \(\log N = \ell = \text{poly}(k)\) times, yielding a good hypothesis for \(f_{t,r}\). \(\square\)

**Proof of Lemma 7.3.6.** Suppose the lemma is not true. First suppose that there exists a ciphertext \(c\) such that \(\text{Dec}(\text{sk}_1, c) = p_1 < p_2 = \text{Dec}(\text{sk}_2, c)\). Let \(c' \in \text{Enc}(\text{sk}_1, p_2)\). Then by strong correctness applied to the parameters \((\text{sk}_1, \text{pars})\), we must have \(\text{Comp}(\text{pars}, c, c') = "<"\). Now by strong correctness applied to \((\text{sk}_2, \text{pars})\), we must have \(\text{Dec}(\text{sk}_2, c') > p_2\). Thus, \(p_1 < \text{Dec}(\text{sk}_1, c') = p_2 < \text{Dec}(\text{sk}_2, c')\). Repeating this argument, we obtain a contradiction because the message space is finite.
Now suppose instead that there is a ciphertext $c$ for which $\text{Dec}(\text{sk}_1, c) = p \in [N]$, but $\text{Dec}(\text{sk}_2, c) = \perp$. Let $c' \in \text{Enc}(\text{sk}_1, p')$ for some $p' > p$. Then $\text{Comp}(\text{pars}, c, c') = "<"$ by strong correctness applied to $(\text{pars}, \text{sk}_1)$. But $\text{Comp}(\text{pars}, c, c') = "\perp"$ by strong correctness applied to $(\text{pars}, \text{sk}_2)$, again yielding a contradiction.

### 7.4 Constructing Strongly Correct ORE Schemes

We now explain how to obtain ORE with strongly correct comparison, as all prior ORE schemes only satisfy the weaker notion of correctness. The lack of strong correctness is easiest to see with the scheme of Boneh et al. [18]. The protocol is built from current multilinear map constructions, which are noisy. If the noise terms grow too large, the correctness of the multilinear map is not guaranteed. The comparison function in [18] is computed by performing multilinear operations, and for correctly generated ciphertexts, the operations will give the right answer. However, there exist ciphertexts, namely those with very large noise, for which the comparison function gives an incorrect output. The result is that the comparison operation is not guaranteed to be consistent with decrypting the ciphertexts and comparing the plaintexts.

As described in the introduction, we give a generic conversion from any ORE scheme with weakly correct comparison into a strongly correct scheme. We simply modify the encryption algorithm by adding a non-interactive zero-knowledge (NIZK) proof that the resulting ciphertext is well-formed. Then the decryption and comparison procedures check the proof(s), and only output a non-$\perp$ result (either decryption or comparison) if the proof(s) are valid.

**Instantiating our scheme.** In our construction, we need the (weak) correctness of the underlying ORE scheme to hold with probability one. However, the existing protocols only have correctness with overwhelming probability, so some minor adjustments need to be made to the protocols. This is easiest to see in the ORE scheme of Boneh et al. [18]. The Boneh et al. scheme uses noisy multilinear maps [64] which may introduce errors. Therefore, the protocol described in [18] only achieves the (weak) correctness property with overwhelming probability, whereas we will require (weak) correctness with probability 1 for the conversion. However, it is straightforward to generate the parameters for the protocol in such a way as to completely eliminate errors. Essentially, the
parameters in the protocol have an error term that is generated by a (discrete) Gaussian distribution, which has unbounded support. Instead, we truncate the Gaussian, resulting in a noise distribution with bounded support. By truncating sufficiently far from the center, the resulting distribution is also statistically close to the full Gaussian, so security of the protocol with truncated noise follows from the security of the protocol with un-truncated noise. By truncating the noise distribution, it is straightforward to set parameters so that no errors can occur.

It is similarly straightforward to modify current obfuscation candidates, which are also built from multilinear maps, to obtain perfect (weak) correctness by truncating the noise distributions. Thus, our scheme has instantiations using multilinear maps or iO.

7.4.1 Conversion from Weakly Correct ORE

We describe our generic conversion from an order-revaling encryption scheme with weak correctness using NIZKs. We will need the following additional tools:

**Perfectly binding commitments.** A perfectly binding commitment $\text{Com}$ is a randomized algorithm with two properties. The first is perfect binding, which states that if $\text{Com}(m; r) = \text{Com}(m'; r')$, then $m = m'$. The second requirement is computational hiding, which states that the distributions $\text{Com}(m)$ and $\text{Com}(m')$ are computationally indistinguishable for any messages $m, m'$. Such commitments can be built, say, from any injective one-way function.

**Perfectly sound NIZK.** A NIZK protocol consists of three algorithms:

- **Setup($1^k$)** is a randomized algorithm that outputs a common reference string $\text{crs}$.
- **Prove($\text{crs}, x, w$)** takes as input a common reference string $\text{crs}$, an NP statement $x$, and a witness $w$, and produces a proof $\pi$.
- **Ver($\text{crs}, x, \pi$)** takes as input a common reference string $\text{crs}$, statement $x$, and a proof $\pi$, and outputs either accept or reject.

We make three requirements for a NIZK:
• **Perfect Completeness.** For all security parameters $\lambda$ and any true statement $x$ with witness $w$,$$
abla \Pr[\text{Ver}(\text{crs}, x, \pi) = \text{accept} : \text{crs} \leftarrow \text{Setup}(1^\lambda); \pi \leftarrow \text{Prove}(\text{crs}, x, w)] = 1.$$

• **Perfect Soundness.** For all security parameters $\lambda$, any false statement $x$ and any (invalid) proof $\pi$,$$
abla \Pr[\text{Ver}(\text{crs}, x, \pi) = \text{accept} : \text{crs} \leftarrow \text{Setup}(1^\lambda)] = 0.$$

• **Computational Zero Knowledge.** There exists a simulator $S_1, S_2$ such that for any computationally bounded adversary $A$, the quantity

$$\| \Pr[A^{\text{Prove} \cdot \text{crs} \cdot \cdot}(\text{crs}) = 1 : \text{crs} \leftarrow \text{Setup}(1^\lambda)]$$

$$- \Pr[A^{\text{Sim} \cdot \text{crs} \cdot \cdot \cdot}(\text{crs}) = 1 : (\text{crs}, \tau) \leftarrow S_1(1^\lambda)] \|$$

is negligible, where $\text{Sim}(\text{crs}, \tau, x, w)$ outputs $S_2(\text{crs}, \tau, x)$ if $w$ is a valid witness for $x$, and $\text{Sim}(\text{crs}, \tau, x, w) = \bot$ if $w$ is invalid.

NIZKs satisfying these requirements can be built from bilinear maps [71].

**The Construction**

We now give our conversion from weak correctness to strong correctness. Let $(\text{Setup}, \text{Prove}, \text{Ver})$ be a perfectly sound NIZK and $(\text{Gen'}, \text{Enc'}, \text{Dec'}, \text{Comp'})$ and ORE with weakly correct comparison. We will assume that $\text{Enc'}$ is deterministic; if not, we can derandomize $\text{Enc'}$ using a pseudorandom function.

Let $\text{Com'}$ be a perfectly binding commitment. We construct a new ORE scheme $(\text{Gen}, \text{Enc}, \text{Dec}, \text{Comp})$ with strongly correct comparison:

- **Gen$(1^\lambda, 1^\ell)$**: run $(sk', pars') \leftarrow \text{Gen}'(1^\lambda, 1^\ell)$. Let $\sigma = \text{Com}(sk; r)$ for randomness $r$, and run $\text{crs} \leftarrow \text{Setup}(1^\lambda)$. Then the secret key is $sk = (sk', r, \text{crs})$ and the public parameters are $pars = (pars', \sigma, \text{crs})$.

- **Enc$(sk, m)$**: Compute $c' = \text{Enc'}(sk', m)$. Let $x_{c'}$ be the statement $\exists \hat{m}, \hat{sk'}, \hat{r} : \sigma = \text{Com}(\hat{sk'}, \hat{r}) \land c' = \text{Enc'}(\hat{sk'}, \hat{m})$. Run $\pi_{c'} = \text{Prove}(\text{crs}, x_{c'}, (m, sk', r))$. Output the ciphertext $c = (c', \pi_{c'})$. 209
• Dec(sk, c): Write \( c = (c', \pi_c) \). If \( \text{Ver}(\text{crs}, x_{c'}, \pi_{c'}) = \text{reject} \), output \( \perp \). Otherwise, output \( m = \text{Dec}'(sk', c') \).

• Comp(pars, c_0, c_1): Write \( c_b = (c'_b, \pi_{c'_b}) \) and \( \text{pars} = (\text{pars}', \sigma, \text{crs}) \). If \( \text{Ver}(\text{crs}, x_{c'_b}, \pi_{c'_b}) = \text{reject} \) for either \( b = 0, 1 \), then output \( \perp \). Otherwise, output \( \text{Comp}'(\text{pars}', c'_0, c'_1) \).

Correctness. Notice that, for each plaintext \( m \), the ciphertext component \( c' = \text{Enc}'(sk', m) \) is the unique value such that \( \text{Dec}(sk, (c', \pi)) = m \) for some proof \( \pi \). Moreover, the completeness of the zero knowledge proof implies that \( \text{Enc}(sk, m) \) outputs a valid proof. Decryption correctness follows.

For strong comparison correctness, consider two ciphertexts \( c_0, c_1 \) where \( c_b = (c'_b, \pi_{c'_b}) \). Suppose both proofs \( \pi_{c'_b} \) are valid, which means that verification passes when running Comp and so \( \text{Comp}(\text{pars}, c_0, c_1) = \text{Comp}'(\text{pars}', c'_0, c'_1) \). Verification also passes when decrypting \( c_b \), and so \( \text{Dec}(sk, c_b) = \text{Dec}'(sk', c'_b) \).

Since the proofs are valid, \( c'_b = \text{Enc}'(sk', m_b) \) for some \( m_b \) for both \( b = 0, 1 \). The weak correctness of comparison for \( (\text{Gen}', \text{Enc}', \text{Dec}', \text{Comp}') \) implies that \( \text{Comp}'(\text{pars}', c'_0, c'_1) = \text{Comp}_{\text{plain}}(m_0, m_1) \). The decryption correctness of \( (\text{Gen}', \text{Enc}', \text{Dec}', \text{Comp}') \) then implies that \( \text{Dec}(sk', c'_b) = m_b \), and therefore \( \text{Dec}(sk, c_b) = m_b \). Thus \( \text{Comp}_{\text{ciph}}(sk, c_0, c_1) = \text{Comp}_{\text{plain}}(m_0, m_1) \). Putting it all together, \( \text{Comp}(\text{pars}, c_0, c_1) = \text{Comp}_{\text{ciph}}(sk, c_0, c_1) \), as desired.

Now suppose one of the proofs \( \pi_{c'_b} \) are invalid. Then \( \text{Comp}(\text{pars}, c_0, c_1) = \perp \) and \( \text{Dec}(sk, c_b) = \perp \). This means \( \text{Comp}_{\text{ciph}}(sk, c_0, c_1) = \perp = \text{Comp}(\text{pars}, c_0, c_1) \), as desired.

Security. To prove security, we first use the zero-knowledge simulator to simulate the proofs \( \pi_c' \) without using a witness (namely, the secret decryption key). Then we use the hiding property of the commitment to replace \( \sigma \) with a commitment to 0. At this point, the entire game can be simulated using an \( \text{Enc}' \) oracle, and so the security reduces to the security of \( \text{Enc}' \).

**Theorem 7.4.1.** If \( (\text{Gen}', \text{Enc}', \text{Dec}', \text{Comp}') \) is a (statically) secure ORE, \( (\text{Setup}, \text{Prove}, \text{Ver}) \) is computationally zero knowledge, and \( \text{Com} \) is computationally hiding, then \( (\text{Gen}, \text{Enc}, \text{Dec}, \text{Comp}) \) is a statically secure ORE.

**Proof.** We will prove security through a sequence of hybrids. Let \( \mathcal{A} \) be an adversary with advantage \( \epsilon \) in breaking the static security of \( (\text{Gen}, \text{Enc}, \text{Dec}, \text{Comp}) \).
Hybrid 0. This is the real experiment, where \( \sigma \leftarrow \text{Com}(sk) \), \( \text{crs} \leftarrow \text{Setup}(1^\lambda) \), and the proofs \( \pi_{c'} \) are answered using \text{Prove} and valid witnesses. \( \mathcal{A} \) has advantage \( \epsilon \) in distinguishing the left and right ciphertexts.

Hybrid 1. This is the same as Hybrid 0, except that \( \text{crs} \) is generated as \( (\text{crs}, \tau) \leftarrow \mathcal{S}_1(1^\lambda) \), and all proofs are generated using \( \mathcal{S}_2(\text{crs}, \tau, \cdot) \). The zero knowledge property of \((\text{Setup}, \text{Prove}, \text{Ver})\) shows that this is indistinguishable from Hybrid 0.

Hybrid 2. This is the same as Hybrid 1, except that \( \sigma \leftarrow \text{Com}(0) \). Since the randomness for computing \( \sigma \) is not needed for simulation, this change is undetectable using the hiding of \text{Com}.

Thus the advantage of \( \mathcal{A} \) in Hybrid 2 is at least \( \epsilon - \text{negl} \) for some negligible function \text{negl}. Now consider the following adversary \( c\mathcal{B} \) that attempts to break the security of \((\text{Gen}', \text{Enc}', \text{Dec}', \text{Comp}')\). \( \mathcal{B} \) simulates \( \mathcal{A} \), and forwards the message sequences \( m_1^{(L)} < m_2^{(L)} < \ldots < m_q^{(L)} \) and \( m_1^{(R)} < m_2^{(R)} < \ldots < m_q^{(R)} \) produced by \( \mathcal{A} \) to its own challenger. In response, it receives \( \text{pars}' \), and ciphertexts \( c_i' \), where \( c_i' \) encrypts either \( m_i^{(L)} \) if \( b = 0 \) or \( m_i^{(R)} \) if \( b = 1 \), for a random bit \( b \) chosen by the challenger.

\( \mathcal{B} \) now generates \( \sigma \leftarrow \text{Com}(0) \), \( (\text{crs}, \tau) \leftarrow \mathcal{S}_1(1^\lambda) \), and lets \( \text{pars} = (\text{pars}', \sigma, \text{crs}) \). It also computes \( \pi_{c_i'} \leftarrow \mathcal{S}_2(\text{crs}, \tau, x_{c_i'}) \), and defines \( c_i = (c_i', \pi_{c_i'}) \), and gives \( \text{pars} \) and the \( c_i \) to \( \mathcal{A} \). Finally when \( \mathcal{A} \) outputs a guess \( b' \) for \( b \), \( \mathcal{B} \) outputs the same guess \( b' \).

We see that the view of \( \mathcal{A} \) as a subroutine of \( \mathcal{B} \) is exactly the same view as in Hybrid 2. Thus, \( b' = b \) with probability at least \( \epsilon - \text{negl} \). The security of \((\text{Gen}', \text{Enc}', \text{Dec}', \text{Comp}')\) implies that this quantity, and hence \( \epsilon \), must be negligible. Thus \( \mathcal{A} \) must have negligible advantage in breaking the security of \((\text{Gen}, \text{Enc}, \text{Dec}, \text{Comp})\).

\( \square \)

7.5 Hardness of Representation Learning

In this section, we show how to construct a concept class \text{ValidSig} that separates efficient representation learning from efficient private representation learning, assuming only the existence of one-way functions. Here by “representation learning” we mean a restricted form of proper learning where a learner must output a particular representation (i.e. encoding) of a hypothesis \( h \) in the concept class
As with proper learning, this is a natural syntactic restriction to place on a learner: for instance, if one wants to learn linear threshold functions (LTF), it makes sense to require a learner to produce the actual coefficients of an LTF, rather than an arbitrary circuit that happens to compute an LTF. In fact, our notion of representation learning is often taken as the definition of proper learning. However, we choose to distinguish the two definitions because, whereas the \( \text{NP} \)-hardness results of e.g. [101] hold even for our more permissive notion of proper learning, the result in this section holds only for representation learning.

The construction is based on the following elegant idea due to Kobbi Nissim [96]. Suppose \( H : D \rightarrow R \) is a cryptographic hash function with the property that given \( x_1, \ldots, x_n \) with \( y = H(x_1) = \ldots = H(x_n) \), it is infeasible for an efficient adversary to find another \( x \) for which \( H(x) = y \).

Consider the concept class \( \text{HashPoint} \), consisting of the concepts

\[
f_{x'}(x) = \begin{cases} 1 & \text{if } H(x) = H(x') \\ 0 & \text{otherwise.} \end{cases}
\]

for every \( x \in R \). The representation of a concept \( f_x \) is the point \( x \). The concept class \( \text{HashPoint} \) is very easy to learn (by representation) without privacy: a learner can identify any positive example \( x_i \) and output the representation \( x_i \). Since \( H(x_i) = H(x) \), the concept \( f_{x_i} \) is actually equal to the target concept \( f_x \). On the other hand, a learner that identifies an index \( x^* \) for which \( f_{x^*} = f_x \) cannot be differentially private, since the security of the hash function means it is infeasible to produce such an \( x^* \) that is not present in the sample.

Note that this argument breaks down if one tries to show that \( \text{HashPoint} \) is not privately properly learnable. While it is infeasible to privately produce a representation \( x^* \) for which \( f_{x^*} \) is a good hypothesis, the hypothesis \( h(x) = \chi(H(x) = h(x_i)) \) is equal as a function to every good \( f_{x^*} \). Moreover, this hypothesis can be constructed privately as long as the sample contains sufficiently many positive examples.

We make this discussion formal by constructing a concept class \( \text{ValidSig} \), based on super-secure digital signature schemes, which can be constructed from one-way functions. Our use of signatures to derive hardness results for private proper learning is very analogous to prior hardness results for synthetic data generation [50,116].
Definition 7.5.1. A digital signature scheme is a triple of algorithms \((\text{Gen}, \text{Sign}, \text{Ver})\) where

- \(\text{Gen}(1^\lambda)\) produces a key pair \((sk, vk)\).
- \(\text{Sign}(sk, m)\) takes the private signing key \(sk\) and a message \(m \in \{0, 1\}^*\) and produces a signature \(\sigma\) for the message \(m\).
- \(\text{Ver}(vk, m, \sigma)\) takes the public verification key \(vk\), a message \(m\), and a signature \(\sigma\), and (deterministically) outputs a bit indicating whether \(\sigma\) is a valid signature for \(m\).

The correctness property of a digital signature scheme is that for every \((sk, vk) \leftarrow_r \text{Gen}(1^\lambda)\), every message \(m \in \{0, 1\}^*\), and every signature \(\sigma \leftarrow_r \text{Sign}(sk, m)\), we have \(\text{Ver}(vk, m, \sigma) = 1\).

Definition 7.5.2. A digital signature scheme is super-secure under adaptive chosen-plaintext attacks if all efficient adversaries \(A\) win the following weak forgery game with negligible probability:

- The challenger samples \((sk, vk) \leftarrow_r \text{Gen}(1^\lambda)\).
- The adversary \(A\) is given \(vk\) and oracle access to \(\text{Sign}(sk, \cdot)\). It adaptively queries the signing oracle, obtaining a sequence of message-signature pairs \(A\). It then outputs a forgery \((m^*, \sigma^*)\).
- The value of the game is 1 iff \(\text{Ver}(vk, m^*, \sigma^*) = 1\) and \((m^*, \sigma^*) \notin A\).

It is known that super-secure digital signature schemes can be constructed from one-way functions [67, 83, 94, 103].

We now describe our concept class \(\text{ValidSig}\). Let \((\text{Gen}, \text{Sign}, \text{Ver})\) be a super-secure digital signature scheme. We define a concept class \(\text{ValidSig}\) as follows. Fix the message length \(\ell\). For every \((vk, m, \sigma)\) with \(m \in \{0, 1\}^\ell\) and \(\text{Ver}(vk, m, \sigma) = 1\), define the concept

\[
f_{vk, m, \sigma}(vk', m', \sigma') = \begin{cases} 
1 & \text{if } (vk = vk') \land (\text{Ver}(vk, m', \sigma') = 1) \\
0 & \text{otherwise.}
\end{cases}
\]

For convenience, we also include the all-zeroes hypothesis in \(\text{ValidSig}\), with representation \(\bot\).

Theorem 7.5.3. Let \(\alpha, \beta > 0\). There exists a proper PAC learning algorithm \(L\) for the concept class \(\text{ValidSig}\) achieving error \(\alpha\) and confidence \(1 - \beta\). Moreover, \(L\) is efficient (running in time polynomial in the parameters \(k, 1/\alpha, \log(1/\beta)\)).
Algorithm 13 Learner $L$ for $\text{ValidSig}$

1. Request examples $\{(vk_1, m_1', \sigma'_1, b_1), \ldots, (vk_n, m_n', \sigma'_n, b_n)\}$ for $n = \lceil \log(1/\beta)/\alpha \rceil$.

2. Identify an $i$ for which $b_i = 1$ and return the representation $(vk_i', m_i', \sigma'_i)$. If no such $i$ exists, return $\bot$ representing the all-zeroes hypothesis.

Proof. Fix a target concept $f_{vk, m, \sigma} \in \text{ValidSig}_k$ and a distribution $D$ on examples. Let $\text{POS}$ denote the set of examples $(vk', m', \sigma')$ on which $f_{vk, m, \sigma}(vk', m', \sigma') = 1$. We divide the analysis of the learner into three cases based on the weight $D$ places on the sets $\text{POS}$.

Case 1: $D$ places at least $\alpha$ weight on $\text{POS}$. Then $L$ receives a positive example with probability at least $1 - (1 - \alpha)^n \geq 1 - \beta$, and is thus able to identify a concept that equals the target concept.

Case 2: $D$ places less than $\alpha$ weight on $\text{POS}$. If $L$ gets a positive example, then the analysis of Case 1 applies. Otherwise, the all-zeroes hypothesis is $\alpha$-good.

We now prove the hardness of properly privately learning $\text{ValidSig}$ by constructing an example reidentification scheme for properly learning this concept class. Our example reidentification scheme yields a hard distribution even when the error parameter $\alpha$ is taken to be inverse-polynomially close to 1.

Theorem 7.5.4. Let $\gamma(n)$ and $\xi(n)$ be noticeable functions. Let $(\text{Gen}, \text{Sign}, \text{Ver})$ be a super-secure digital signature scheme. Then there exists an (efficient) $(\alpha = 1 - \gamma, \xi)$-example reidentification scheme $(\text{Gen}_{\text{ex}}, \text{Trace}_{\text{ex}})$ for representation learning the concept class $\text{ValidSig}$.

Proof. We construct an example reidentification scheme for $\text{ValidSig}$ as follows. The algorithm $\text{Gen}_{\text{ex}}$ samples $(sk, vk) \leftarrow \text{Gen}(1^\lambda)$, a message $m \in \{0, 1\}^\ell$, and a signature $\sigma \leftarrow \text{Sign}(sk, m)$, yielding a concept $f_{vk, m, \sigma}$. Let $D$ be the distribution of $(vk, m, \text{Sign}(sk, m))$ for random $m \leftarrow \{0, 1\}^\ell$. $\text{Gen}_{\text{ex}}$ then samples $x_0, x_1, \ldots, x_n$ i.i.d. from $D$. Given a representation $(vk^*, m^*, \sigma^*)$, the algorithm $\text{Trace}_{\text{ex}}$ simply identifies an index $i$ for which $x_i = (vk^*, m^*, \sigma^*)$, and outputs $\bot$ if none is found.

We first verify completeness. Let $L$ be a learner for $\text{ValidSig}$ using $n$ examples. If the representation $(vk^*, m^*, \sigma^*)$ produced by $L$ represents an $(1 - \gamma)$-good hypothesis, then it must be the case that
\( \text{vk}^* = \text{vk} \) and \( \text{Ver}(\text{vk}, m^*, \sigma^*) = 1 \). Thus, if \( L \) violates the completeness condition, it can be used to construct the weak forgery adversary \( A \) (Figure 14) that succeeds with noticeable probability \( \xi \).

**Algorithm 14** Weak forgery adversary \( A \)

1. Query the signing oracle on random messages \( m'_1, \ldots, m'_n \leftarrow \{0, 1\}^\ell \), obtaining signatures \( \sigma'_1, \ldots, \sigma'_n \).
2. Run \( L \) on the labeled examples \( ((\text{vk}, m'_1, \sigma'_1), 1), \ldots, ((\text{vk}, m'_n, \sigma'_n), 1) \), obtaining a representation \( (m^*, \sigma^*) \).
3. Output the forgery \( (m^*, \sigma^*) \).

Now we verify soundness. Observe that for any \( i \), the sample \( S_{-i} \) contains no information about message \( m_i \). Therefore, the learner has a \( 2^{-\ell} = \text{negl}(k) \) probability at producing a representation containing message \( m_i \), proving soundness.

\[ \square \]
Chapter 8

Conclusions

Our study of the complexity of differential privacy reveals aspects of the rich interplay between privacy, learning theory, computational complexity, and cryptography. We summarize some of the main themes in this thesis, as well as point out a few additional open problems.

Cryptography vs. Differential Privacy. Cryptography, especially through secure content-distribution schemes, has played a major role in the theory of computational lower bounds for differential privacy [22,50,115]. One of the main contributions of this thesis shows that fingerprinting codes, which often form the information-theoretic hardcore of such schemes, are themselves useful for establishing sample complexity lower bounds. Beyond the applications to lower bounds for attribute means and $k$-way marginals (Chapter 3, [111]), prefix queries (Chapter 5), and multi-learning (Chapter 6) presented in this thesis, fingerprinting codes have been applied to other problems in differential privacy, including PCA [57] and convex empirical risk minimization [4]. It remains an intriguing question to understand the exact nature of the relationship between (generalized) fingerprinting codes and lower bounds for differential privacy (see Open Problem 3.2.14), or to develop new methods for constructing generalized fingerprinting codes.

The other noteworthy appearance of cryptography in this thesis is through our use of order-revealing encryption to prove computational lower bounds for private PAC learning. While we suspect the assumptions needed for our separation are overkill (see Open Problems 7.1.2 and 7.1.3), order-revealing encryption provides a new and intuitive source of hardness for computational tasks. We are confident that additional tools from modern cryptography will be brought to bear on
differential privacy.

In this vein, it is worth pointing out that the impact of cryptography on differential privacy is not limited to the theory of lower bounds. A complementary line of work has demonstrated promising positive uses of cryptography for differential privacy as well. In particular, by relaxing the guarantee of differential privacy to hold only against computationally bounded adversaries, algorithms can use tools from secure multiparty computation \cite{6,90,92} to achieve improved accuracy for distributed problems. There is still much in this space left for us to understand, especially in terms of what advantages cryptography can confer in the traditional client-server model \cite{25,70}.

**The Curious Complexity of Threshold Functions.** Many of the new lower bounds presented in this thesis are witnessed by the simple class of one-dimensional thresholds. In Chapter 4, we saw that they are already powerful enough to separate proper private PAC learning from non-private learning. They also suffice to give a dramatic separation between private and non-private multi-learning (Chapter 6), and lay the information-theoretic foundation for our computational hardness result for private PAC learning (Chapter 7).

One of the most surprising aspects of these lower bounds is the gap between threshold functions and the simpler point functions. Both classes have VC dimension 1, and are hence “easy” to learn non-privately, in either the sense of query release/distribution-learning or in the sense of PAC learning. However, all of our lower bounds leverage the fact that threshold functions hold a non-trivial combinatorial structure that is not captured by VC dimension alone. Can we get a handle on what this structure is, and use it to understand the complexity of other families of queries? In particular, can we identify more generic combinatorial properties of a query class $Q$ that suffice to prove lower bounds (perhaps along the lines of the Ramsey-theoretic lower bound in Section 4.2.2)?

More ambitiously:

**Open Problem 8.0.5 (Combinatorial characterization of sample complexity).** Characterize the sample complexity of releasing an arbitrary family of counting queries $Q$ with $(\varepsilon, \delta)$-differential privacy and $(\alpha, \beta, \gamma)$-accuracy (in terms of the combinatorial properties of $Q$).

Similarly, characterize the sample complexity of PAC learning a concept class $C$ with $(\varepsilon, \delta)$-differential privacy.

Recall that the line of work on the geometry of differential privacy \cite{93,95} resolves this question.
up to \(\text{poly}(d, \log |\mathcal{Q}|)\) factors. A characterization, up to constant factors, of the sample complexity required to learn a concept class with pure \((\varepsilon, 0)\)-differential privacy exists based on a quantity called the \textit{probabilistic representation dimension} \cite{7}, or equivalently randomized one-way communication complexity \cite{63}. These works exploit a beautiful duality between packing lower bounds and the universality of the exponential mechanism for pure differential privacy. In contrast, no candidate universal mechanism for approximate differential privacy is known.
Bibliography


225


