How to Use Heuristics for Differential Privacy

Seth Neel* Aaron Roth† Zhiwei Steven Wu‡

November 20, 2018

Abstract

We develop theory for using *heuristics* to solve computationally hard problems in differential privacy. Heuristic approaches have enjoyed tremendous success in machine learning, for which performance can be empirically evaluated. However, privacy guarantees cannot be evaluated empirically, and must be proven — without making heuristic assumptions. We show that learning problems over broad classes of functions — those that have polynomially sized universal identification sets — can be solved privately and efficiently, assuming the existence of a non-private oracle for solving the same problem. Our first algorithm yields a privacy guarantee that is contingent on the correctness of the oracle. We then give a reduction which applies to a class of heuristics which we call *certifiable*, which allows us to convert oracle-dependent privacy guarantees to worst-case privacy guarantee that hold even when the heuristic standing in for the oracle might fail in adversarial ways. Finally, we consider classes of functions for which both they and their dual classes have small universal identification sets. This includes most classes of simple boolean functions studied in the PAC learning literature, including conjunctions, disjunctions, parities, and discrete halfspaces. We show that there is an efficient algorithm for privately constructing synthetic data for any such class, given a non-private learning oracle. This in particular gives the first oracle-efficient algorithm for privately generating synthetic data for contingency tables. The most intriguing question left open by our work is whether or not *every problem* that can be solved differentially privately can be privately solved with an oracle-efficient algorithm. While we do not resolve this, we give a barrier result that suggests that any generic oracle-efficient reduction must fall outside of a natural class of algorithms (which includes the algorithms given in this paper).

*Wharton Statistics Department, University of Pennsylvania. Email: sethneel@wharton.upenn.edu. Supported in part by an NSF Graduate Research Fellowship

†Department of Computer and Information Sciences, University of Pennsylvania. Email: aaroth@cis.upenn.edu. Supported in part by the Sloan foundation, the DARPA Brandeis project, and NSF awards 1253345 and 1513694.

‡Computer Science and Engineering Department, University of Minnesota. Email: zsw@umn.edu

Contents

1 Introduction

Differential privacy is compatible with a tremendous number of powerful data analysis tasks, including essentially any statistical learning problem [KLN+11, CMS11, BST14] and the generation of synthetic data consistent with exponentially large families of statistics [BLR13, RR10, HR10, GRU12, NTZ13]. Unfortunately, it is also beset with a comprehensive set of computational hardness results. Of course, it inherits all of the computational hardness results from the (non-private) agnostic learning literature: for example, even the simplest learning tasks — like finding the best conjunction or linear separator to approximately minimize classification error — are hard [FGKP09, FGRW12, DOSW11]. In addition, tasks that are easy absent privacy constraints can become hard when these constraints are added. For example, although information theoretically, it is possible to privately construct synthetic data consistent with all d-way marginals for d-dimensional data, privately constructing synthetic data for even 2-way marginals is computationally hard [UV10]. These hardness results extend even to providing numeric answers to more than quadratically many statistical queries [Ull16].

How should we proceed in the face of pervasive computational hardness? We might take inspiration from machine learning, which has not been slowed, despite the fact that its most basic problems (e.g. learning linear separators) are already hard even to approximate. Instead, the field has employed heuristics with tremendous success — including exact optimization of convex surrogate loss functions (as in the case of SVMs), decision tree heuristics, gradient based methods for differentiable but non-convex problems (as in back-propogation for training neural networks), and integer programming solvers (as in recent work on interpretable machine learning [UR16]). Other fields such as operations research similarly have developed sophisticated heuristics including integer program solvers and SAT solvers that are able to routinely solve problems that are hard in the worst case.

The case of private data analysis is different, however. If we are only concerned with performance (as is the case for most machine learning and combinatorial optimization tasks), we have the freedom to try different heuristics, and evaluate our algorithms in practice. Thus the design of heuristics that perform well in practice can be undertaken as an empirical science. In contrast, differential privacy is an inherently worst-case guarantee that cannot be evaluated empirically (see [GM18] for lower bounds for black-box testing of privacy definitions).

In this paper, we build a theory for how to employ *non-private* heuristics (of which there are many, benefitting from many years of intense optimization) to solve computationally hard problems in differential privacy. Our goal is to guide the design of practical algorithms about which we can still prove theorems:

1. We will aim to prove accuracy theorems *under the assumption that our heuristics solve some non-private problem optimally*. We are happy to make this assumption when proving our accuracy theorems, because accuracy is something that can be empirically evaluated on the datasets that we are interested in. An assumption like this is also necessary, because we are designing algorithms for problems that are computationally hard in the worst case. However:

2. We aim to prove that our algorithms are differentially private in the worst case, even under the assumption that our heuristics might fail in an adversarial manner.

1.1 Overview of Our Results

Informally, we give a collection of results showing the existence of *oracle-efficient* algorithms for privately solving learning and synthetic data generation problems defined by discrete classes of functions $\mathcal{Q}$ that have a special (but common) combinatorial structure. One might initially ask whether it is possible to give a direct reduction from a non-private but efficient algorithm for solving a learning problem to an efficient private algorithm for solving the same learning problem *without requiring any special structure at all*. However, this is impossible, because there are classes of functions (namely those that have finite VC-dimension but infinite Littlestone dimension) that are known to be learnable absent the constraint of privacy, but are not privately learnable in an information-theoretic sense [BNSV15, ALMM18]. The main question we leave open is whether *being information theoretically learnable under the constraint of differential privacy* is sufficient for oracle-efficient private learning. We give a barrier result suggesting that it might not be.

Before we summarize our results in more detail, we give some informal definitions.

1.1.1 Definitions

We begin by defining the kinds of *oracles* that we will work with, and end-goals that we will aim for. We will assume the existence of oracles for (non-privately) solving learning problems: for example, an oracle which can solve the empirical risk minimization problem for discrete linear threshold functions. Because ultimately oracles will be implemented using heuristics, we consider two types of oracles:

1. *Certifiable* heuristic oracles might fail, but when they succeed, they come with a certificate of success. Many heuristics for solving integer programs are certifiable, including cutting planes methods and branch and bound methods. SAT Solvers (and any other heuristic for solving a decision problem in NP) are also certifiable.

2. On the other had, some heuristics are *non-certifiable*. These heuristics might produce incorrect answers, without any indication that they have failed. Support vector machines and logistic regression are examples of non-certifiable heuristic oracles for learning linear threshold functions.

We define an oracle-efficient *non-robustly* differentially private algorithm to be an algorithm that runs in polynomial time in all relevant parameters given access to an oracle for some problem, and has an accuracy guarantee and a differential privacy guarantee which may both be *contingent* on the guarantees of the oracle — i.e. if the oracle is replaced with a heuristic, the algorithm may no longer be differentially private. Although in certain situations (e.g when we have very high confidence that our heuristics actually do succeed on all instances we will ever encounter) it might be acceptable to have a privacy guarantee that is contingent on having an infallible oracle, we would much prefer a privacy guarantee that held in the worst case. We say that an oracle-efficient algorithm is *robustly* differentially private if its privacy guarantee is not contingent on the behavior of the oracle, and holds in the worst case, even if an adversary is in control of the heuristic that stands in for our oracle.

1.1.2 Learning and Optimization

Our first result is a reduction from efficient non-private learning to efficient private learning over any class of functions $\mathcal{Q}$ that has a small universal identification set [GKS93]. A universal identification set of size m is a set of m examples such that the labelling of these examples by a function $q \in \mathcal{Q}$ is enough to uniquely identify q. Equivalently, a universal identification set can be viewed as a *separator set* [SKS16]: for any pair of functions $q \neq q' \in \mathcal{Q}$, there must be some example x in the universal identification set such that $q(x) \neq q(x')$. We will use these terms interchangeably throughout the paper. We show that if $\mathcal{Q}$ has a universal identification set of size m, then given an oracle which solves the empirical risk minimization problem (non-privately) over $\mathcal{Q}$, there is an ϵ-differentially private algorithm with additional running time scaling linearly with m and error scaling linearly with m^2/ϵ that solves the private empirical risk minimization problem over $\mathcal{Q}$. The error can be improved to $O(m^{1.5}\sqrt{\log 1/\delta}/\epsilon)$, while satisfying (ϵ, δ)-differential privacy. Many well studied discrete concept classes $\mathcal{Q}$ from the PAC learning literature have small universal identification sets. For example, in d dimensions, boolean conjunctions, disjunctions, parities, and halfspaces defined over the hypercube have universal identification sets of size d. This means that for these classes, our oracle-efficient algorithm has error that is larger than the generic optimal (and computationally inefficient) learner from [KLN+11] by a factor of $O(\sqrt{d})$. Other classes of functions also have small universal identification sets — for example, decision lists have universal identification sets of size d^2.

The reduction described above has the disadvantage that not only its accuracy guarantees — but also its proof of privacy — depend on the oracle correctly solving the empirical risk minimization problem it is given; it is *non-robustly* differentially private. This shortcoming motivates our main technical result: a generic reduction that takes as input any oracle-efficient non-robustly differentially private algorithm (i.e. an algorithm whose privacy proof might depend on the proper functioning of the oracle) and produces an oracle-efficient *robustly* differentially private algorithm, *whenever the oracle is implemented with a certifiable heuristic*. As discussed above, this class of heuristics includes the integer programming algorithms used in most commercial solvers. In combination with our first result, we obtain robustly differentially private oracle-efficient learning algorithms for conjunctions, disjunctions, discrete halfspaces, and any other class of functions with a small universal identification set.

1.1.3 Synthetic Data Generation

We then proceed to the task of constructing synthetic data consistent with a class of queries $\mathcal{Q}$. Following [HRU13, GGAH+14], we view the task of synthetic data generation as the process of computing an equilibrium of a particular zero sum game played between a data player and a query player. In order to compute this equilibrium, we need to be able to instantiate two objects in an oracle-efficient manner:

1. a private *learning* algorithm for $\mathcal{Q}$ (this corresponds to solving the best response problem for the "query player"), and

2. a *no-regret learning algorithm* for a dual class of functions $\mathcal{Q}_{\text{dual}}$ that results from swapping the role of the data element and the query function (this allows the "data player" to obtain a diminishing regret bound in simulated play of the game).

The no-regret learning algorithm need not be differentially private. From our earlier results, we are able to construct an oracle-efficient robustly differentially private learning algorithm for $\mathcal{Q}$ whenever it has a small universal identification set. On the other hand, Syrgkanis et al. [SKS16] show how to obtain an oracle-efficient no regret learning algorithm for a class of functions under the same condition. Hence, we obtain an oracle-efficient robustly differentially private synthetic data generation algorithm for any class of functions $\mathcal{Q}$ for which both $\mathcal{Q}$ and $\mathcal{Q}_{\text{dual}}$ have small universal identification sets. Fortunately, this is the case for many interesting classes of functions, including boolean disjunctions, conjunctions, discrete halfspaces, and parity functions. The result is that we obtain oracle-efficient algorithms for generating private synthetic data for all of these classes. We note that the oracle used by the data player need not be certifiable.

1.1.4 A Barrier Result

Finally, we exhibit a barrier to giving oracle-efficient private learning algorithms for *all* classes of functions $\mathcal{Q}$ known to be privately learnable. We identify a class of private learning algorithms called *perturbed empirical risk minimizers* (pERMs) which output the query that *exactly* minimizes some perturbation of their empirical risk on the dataset. This class of algorithms includes the ones we give in this paper, as well as many other differentially private learning algorithms, including the exponential mechanism and report-noisy-min. We show that any private pERM can be efficiently used as a no-regret learning algorithm with regret guarantees that depend on the scale of the perturbations it uses. This allows us to reduce to a lower bound on the running time of oracle-efficient online learning algorithms due to Hazan and Koren [HK16]. The result is that there exist finite classes of queries $\mathcal{Q}$ such that any oracle-efficient differentially private pERM algorithm must introduce perturbations that are polynomially large in the size of $|Q|$, whereas any such class is information-theoretically privately learnable with error that scales only with $\log|\mathcal{Q}|$.

The barrier implies that *if* oracle-efficient differentially private learning algorithms are as powerful as inefficient differentially private learning algorithms, then these general oracle efficient private algorithms must not be perturbed empirical risk minimizers. We conjecture that the set of problems solvable by oracle-efficient differentially private learners is strictly smaller than the set of problems solvable information theoretically under the constraint of differential privacy, but leave this as our main open question.

1.2 Additional Related Work

Conceptually, the most closely related piece of work is the "DualQuery" algorithm of [GGAH+14], which in the terminology of our paper is a robustly private oracle-efficient algorithm for generating synthetic data for k-way marginals for constant k. The main idea in [GGAH+14] is to formulate the private optimization problem that needs to be solved so that the only computationally hard task is one that does not depend on private data. There are other algorithms that can straightforwardly be put into this framework, like the projection algorithm from [NTZ13]. This approach immediately makes the privacy guarantees independent of the correctness of the oracle, but significantly limits the algorithm design space. In particular, the DualQuery algorithm (and the oracle-efficient version of the projection algorithm from [NTZ13]) has running time that is proportional to $|\mathcal{Q}|$, and so can only handle polynomially sized classes of queries (which is why k needs to be held constant). The main contribution of our paper is to be able to handle private optimization problems in which the hard computational step is *not* independent of the private

data. This is significantly more challenging, and is what allows us to give oracle-efficient robustly private algorithms for constructing synthetic data for exponentially large families $\mathcal{Q}$. It is also what lets give oracle-efficient private *learning* algorithms over exponentially large $\mathcal{Q}$ for the first time.

A recent line of work starting with the "PATE" algorithm [PAE+16] together with more recent theoretical analyses of similar algorithms by Dwork and Feldman, and Bassily, Thakkar, and Thakurta [DF18, BTT18] can be viewed as giving oracle-efficient algorithms for an easier learning task, in which the goal is to produce a finite number of private *predictions* rather than privately output the model that makes the predictions. These can be turned into oracle efficient algorithms for outputting a private model *under the assumption* that the mechanism has access to an additional source of unlabeled data drawn from the same distribution as the private data, but that does not need privacy protections. In this setting, there is no need to take advantage of any special structure of the hypothesis class $\mathcal{Q}$, because the information theoretic lower bounds on private learning proven in [BNSV15, ALMM18] do not apply. In contrast, our results apply without the need for an auxiliary source of non-private data.

Privately producing *contingency tables*, and synthetic data that encode them — i.e. the answers to statistical queries defined by conjunctions of features — has been a key challenge problem in differential privacy at least since [BCD+07]. Since then, a number of algorithms and hardness results have been given [UV10, GHRU13, KRSU10, TUV12, HRS12, FK14, CTUW14]. This paper gives the first oracle-efficient algorithm for generating synthetic data consistent with a full contingency table, and the first oracle-efficient algorithm for answering arbitrary conjunctions to near optimal error.

Technically, our work is inspired by Syrgkanis et al. [SKS16] who show how a small separator set (equivalently a small universal identification set) can be used to derive oracle-efficient no-regret algorithms in the contextual bandit setting. The small separator property has found other uses in online learning, including in the oracle-efficient construction of nearly revenue optimal auctions [DHL+17]. Hazan and Koren [HK16] show lower bounds for oracle-efficient no-regret learning algorithms in the experts setting, which forms the basis of our barrier result. More generally, there is a rich literature studying oracle-efficient algorithms in machine learning [BDH+05, BBB+08, BILM16] and optimization [BTHKM15] as a means of dealing with worst-case hardness, and more recently, for machine learning subject to fairness constraints [ABD+18, KNRW18, AIK18].

We also make crucial use of a property of differentially private algorithms, first shown by [CLN+16]: That when differentially private algorithms are run on databases of size n with privacy parameter $\epsilon \approx 1/\sqrt{n}$, then they have similar output distributions when run on datasets that are *sampled from the same distribution*, rather than just on neighboring datasets. In [CLN+16], this was used as a tool to show the existence of *robustly generalizing* algorithms (also known as *distributionally private* algorithms in [BLR13]). We prove a new variant of this fact that holds when the datasets are not sampled i.i.d. and use it for the first time in an analysis to prove differential privacy. The technique might be of independent interest.

2 Preliminaries

2.1 Differential Privacy Tools

Let $\mathcal{X}$ denote a d-dimensional data domain (e.g. $\mathbb{R}^d$ or $\{0,1\}^d$). We write n to denote the size of a dataset S. We call two *data sets* $S, S' \in \mathcal{X}^n$ *neighbors* (written as $S \sim S'$) if S can be derived from S' by replacing a single data point with some other element of $\mathcal{X}$.

Definition 1 (Differential Privacy [DMNS06, DKM+06])**.** *Fix $\varepsilon, \delta \geq 0$. A randomized algorithm $A : \mathcal{X}^* \to \mathcal{O}$ is (ε, δ)-differentially private if for every pair of neighboring data sets $S \sim S' \in \mathcal{X}^*$, and for every event $\Omega \subseteq \mathcal{O}$:*

$$\Pr[A(S) \in \Omega] \leq \exp(\varepsilon) \Pr[A(S') \in \Omega] + \delta.$$

Differentially private computations enjoy two nice properties:

Theorem 1 (Post Processing [DMNS06, DKM+06])**.** *Let $A : \mathcal{X}^* \to \mathcal{O}$ be any (ε, δ)-differentially private algorithm, and let $f : \mathcal{O} \to \mathcal{O}'$ be any function. Then the algorithm $f \circ A : \mathcal{X}^* \to \mathcal{O}'$ is also (ε, δ)-differentially private.*

Post-processing implies that, for example, every *decision* process based on the output of a differentially private algorithm is also differentially private.

Theorem 2 (Basic Composition [DMNS06, DKM+06])**.** *Let $A_1 : \mathcal{X}^* \to \mathcal{O}$, $A_2 : \mathcal{O} \times \mathcal{X}^* \to \mathcal{O}'$ be such that A_1 is $(\varepsilon_1, \delta_1)$-differentially private, and $A_2(o, \cdot)$ is $(\varepsilon_2, \delta_2)$-differentially private for every $o \in \mathcal{O}$. Then the algorithm $A : \mathcal{X}^* \to \mathcal{O}'$ defined as $A(x) = A_2(A_1(x), x)$ is $(\varepsilon_1 + \varepsilon_2, \delta_1 + \delta_2)$-differentially private.*

The Laplace distribution plays a fundamental role in differential privacy. The Laplace Distribution centered at 0 with scale b is the distribution with probability density function $\text{Lap}(z|b) = \frac{1}{2b} e^{-\frac{|z|}{b}}$. We write $X \sim \text{Lap}(b)$ when X is a random variable drawn from a Laplace distribution with scale b. Let $f : \mathcal{X}^n \to \mathbb{R}^k$ be an arbitrary function. The ℓ_1 *sensitivity* of f is defined to be $\Delta_1(f) = \max_{S \sim S'} \|f(S) - f(S')\|_1$. The *Laplace mechanism* with parameter ε simply adds noise drawn independently from $\text{Lap}(\frac{\Delta_1(f)}{\varepsilon})$ to each coordinate of $f(S)$.

Theorem 3 ([DMNS06])**.** *The Laplace mechanism is ε-differentially private.*

2.2 Statistical Queries and Separator Sets

We study learning (optimization) and synthetic data generation problems for statistical queries defined over a data universe $\mathcal{X}$. A statistical query over $\mathcal{X}$ is a function $q : \mathcal{X} \to \{0,1\}$. A statistical query can represent, e.g. any binary classification model or the binary loss function that it induces. Given a dataset $S \in \mathcal{X}^n$, the value of a statistical query q on S is defined to be $q(S) = \frac{1}{n} \sum_{i=1}^{n} q(S_i)$. In this paper, we will generally think about query classes $\mathcal{Q}$ that represent standard *hypothesis classes* from learning theory – like conjunctions, disjunctions, halfspaces, etc.

In this paper, we will make crucial use of *universal identification sets* for classes of statistical queries. Universal identification sets are equivalent to *separator sets*, defined (in a slightly more general form) in [SKS16].

Definition 2 ([GKS93, SKS16])**.** *A set $U \subseteq \mathcal{X}$ is a* universal identification set *or* separator set *for a class of statistical queries $\mathcal{Q}$ if for every pair of distinct queries $q, q' \in \mathcal{Q}$, there is an $x \in U$ such that:*

$$q(x) \neq q(x')$$

If $|U| = m$, then we say that $\mathcal{Q}$ has a separator set of size m.

Many classes of statistical queries defined over the boolean hypercube have separator sets of size proportional to their VC-dimension. For example, boolean conjunctions, disjunctions, halfspaces defined over the hypercube, and parity functions in d dimensions all have separator sets of size d. When we solve learning problems over these classes, we will be interested in the set of queries that define the 0/1 loss function over these classes: but as we observe in Appendix A, if a hypothesis class has a separator set of size m, then so does the class of queries representing the empirical loss for functions in that hypothesis class.

2.3 Learning and Synthetic Data Generation

We study private learning as empirical risk minimization (the connection between in-sample risk and out-of-sample risk is standard, and follows from e.g. VC-dimension bounds [KV94] or directly from differential privacy (see e.g. [BST14, DFH+15])). Such problems can be cast as finding a function q in a class $\mathcal{Q}$ that minimizes $q(S)$, subject to differential privacy (observe that the empirical risk of a hypothesis is a statistical query — see Appendix A). We will therefore study minimization problems over classes of statistical queries generally:

Definition 3. *We say that a randomized algorithm $M : \mathcal{X}^n \to \mathcal{Q}$ is an (α, β)-minimizer for $\mathcal{Q}$ if for every dataset $S \in \mathcal{X}^n$, with probability $1 - \beta$, it outputs $M(S) = q$ such that:*

$$q(S) \leq \arg\min_{q^* \in \mathcal{Q}} q^*(S) + \alpha$$

Synthetic data generation, on the other hand, is the problem of constructing a *new* dataset $\hat{S}$ that approximately agrees with the original dataset with respect to a fixed set of statistical queries:

Definition 4. *We say that a randomized algorithm $M : \mathcal{X}^n \to \mathcal{X}^*$ is an (α, β)-accurate synthetic data generation algorithm for $\mathcal{Q}$ if for every dataset $S \in \mathcal{X}^n$, with probability $1 - \beta$, it outputs $M(S) = \hat{S}$ such that for all $q \in \mathcal{Q}$:*

$$|q(S) - q(\hat{S})| \leq \alpha$$

2.4 Oracles and Oracle Efficient Algorithms

We discuss several kinds of oracle-efficient algorithms in this paper. It will be useful for us to study oracles that solve weighted generalizations of the minimization problem, in which each datapoint $x_i \in S$ is paired with a real-valued weight w_i. In the literature on oracle-efficiency in machine learning, these are widely employed, and are known as *cost-sensitive classification oracles*. Via a simple translation and re-weighting argument, they are no more powerful than unweighted minimization oracles, but are more convenient to work with.

Definition 5. *A weighted optimization oracle for a class of statistical queries $\mathcal{Q}$ is a function $\mathcal{O}^* : (\mathcal{X} \times \mathbb{R})^* \to \mathcal{Q}$ takes as input a weighted dataset $WD \in (\mathcal{X} \times \mathbb{R})^*$ and outputs a query $q = \mathcal{O}^*(WD)$ such*

that

$$q \in \operatorname*{argmin}_{q^* \in \mathcal{Q}} \sum_{(x_i, w_i) \in WD} w_i q^*(x_i).$$

In this paper, we will study algorithms that have access to weighted optimization oracles for learning problems that are computationally hard. Since we do not believe that such oracles have worst-case polynomial time implementations, in practice, we will instantiate such oracles with heuristics that are not guaranteed to succeed. There are two failure modes for a heuristic: it can fail to produce an output at all, or it can output an incorrect query. The distinction can be important. We call a heuristic that might fail to produce an output, but never outputs an incorrect solution a certifiable heuristic optimization oracle:

Definition 6. *A certifiable heuristic optimization oracle for a class of queries $\mathcal{Q}$ is a polynomial time algorithm $\mathcal{O} : (\mathcal{X} \times \mathbb{R})^* \to (\mathcal{Q} \cup \perp)$ that takes as input a weighted dataset $WD \in (\mathcal{X} \times \mathbb{R})^*$ and either outputs $\mathcal{O}(WD) = q \in \operatorname{argmin}_{q^* \in \mathcal{Q}} \sum_{(x_i, w_i) \in WD} w_i q^*(x_i)$ or else outputs $\perp$ ("Fail"). If it outputs a statistical query q, we say the oracle has succeeded.*

In contrast, a heuristic optimization oracle (that is not certifiable) has no guarantees of correctness. Without loss of generality, such oracles never need to return "Fail" (since they can always instead output a default statistical query in this case).

Definition 7. *A (non-certifiable) heuristic optimization oracle for a class of queries $\mathcal{Q}$ is an arbitrary polynomial time algorithm $M : (\mathcal{X} \times \mathbb{R})^* \to \mathcal{Q}$. Given a call to the oracle defined by a weighted dataset $WD \in (\mathcal{X} \times \mathbb{R})^*$ we say that the oracle has succeeded on this call up to error α if it outputs a query q such that $\sum_{(x_i, w_i) \in WD} w_i q(x_i) \leq \min_{q^* \in \mathcal{Q}} \sum_{(x_i, w_i) \in WD} w_i q^*(x_i) + \alpha$. If it succeeds up to error 0, we just say that the heuristic oracle has succeeded. Note that there may not be any efficient procedure to determine whether the oracle has succeeded up to error α.*

We say an algorithm $\mathcal{A}_{\mathcal{O}}$ is (certifiable)-oracle dependent if throughout the course of its run it makes a series of (possibly adaptive) calls to a (certifiable) heuristic optimization oracle $\mathcal{O}$. An oracle-dependent algorithm $\mathcal{A}_{\mathcal{O}}$ is *oracle equivalent* to an algorithm $\mathcal{A}$ if given access to a perfect optimization oracle $\mathcal{O}^*$, $\mathcal{A}_{\mathcal{O}^*}$ induces the same distribution on outputs as $\mathcal{A}$. We now state an intuitive lemma (that could also be taken as a more formal definition of *oracle equivalence*). See the Appendix for a proof.

Lemma 1. *Let $\mathcal{A}_{\mathcal{O}}$ be a certifiable-oracle dependent algorithm that is oracle equivalent to $\mathcal{A}$. Then for any fixed input dataset S, there exists a coupling between $\mathcal{A}(S)$ and $\mathcal{A}_{\mathcal{O}}(S)$ such that $\Pr[\mathcal{A}_{\mathcal{O}}(S) = a | \mathcal{A}_{\mathcal{O}}(S) \neq \perp] = \Pr[\mathcal{A}(S) = a | \mathcal{A}_{\mathcal{O}}(S) \neq \perp]$.*

We will also discuss differentially private heuristic optimization oracles, in order to state additional consequences of our construction in Section 4. Note that because differential privacy precludes exact computations, differentially private heuristic oracles are necessarily non-certifiable, and will never succeed up to error 0.

Definition 8. *A weighted (ϵ, δ)-differentially private (α, β)-accurate learning oracle for a class of statistical queries $\mathcal{Q}$ is an (ϵ, δ) differentially private algorithm $\mathcal{O} : (\mathcal{X} \times \mathbb{R})^* \to C$ that takes as input a*

weighted dataset $WD \in (\mathcal{X} \times \mathbb{R})^*$ *and outputs a query* $q_{priv} \in \mathcal{Q}$ *such that with probability* $1 - \beta$*:*

$$\sum_{(x_i,w_i)\in WD} w_i q_{priv}(x_i) - \operatorname*{argmin}_{q^*\in C} \sum_{(x_i,w_i)\in WD} w_i q^*(x_i) \leq \alpha$$

We say that an algorithm is *oracle-efficient* if given access to an oracle (in this paper, always a weighted optimization oracle for a class of statistical queries) it runs in polynomial time in the length of its input, and makes a polynomial number of calls to the oracle. In practice, we will be interested in the performance of oracle-efficient algorithms when they are instantiated with heuristic oracles. Thus, we further require oracle-efficient algorithms to halt in polynomial time even when the oracle fails. When we design algorithms for optimization and synthetic data generation problems, their (α, β)-accuracy guarantees will generally rely on all queries to the oracle succeeding (possibly up to error $O(\alpha)$). If our algorithms are merely *oracle equivalent* to differentially private algorithms, then their privacy guarantees depend on the correctness of the oracle. However, we would prefer that the *privacy* guarantee of the algorithm not depend on the success of the oracle. We call such algorithms *robustly* differentially private.

Definition 9. *An oracle-efficient algorithm* M *is* (ϵ, δ)*-robustly differentially private if it satisfies* (ϵ, δ)*-differential privacy even under worst-case performance of a heuristic optimization oracle. In other words, it is differentially private for every heuristic oracle* $\mathcal{O}$ *that it might be instantiated with.*

We write that an oracle efficient algorithm is non-robustly differentially private to mean that it is oracle equivalent to a differentially private algorithm.

3 Oracle Efficient Optimization

In this section, we show how weighted optimization oracles can be used to give differentially private oracle-efficient optimization algorithms for many classes of queries with performance that is worse only by a $\sqrt{d}$ factor compared to that of the (computationally inefficient) exponential mechanism. The first algorithm we give is not robustly differentially private — that is, its differential privacy guarantee relies on having access to a perfect oracle. We then show how to make that algorithm (or any other algorithm that is oracle equivalent to a differentially private algorithm) robustly differentially private when instantiated with a certifiable heuristic optimization oracle.

3.1 A (Non-Robustly) Private Oracle Efficient Algorithm

In this section, we give an oracle-efficient (non-robustly) differentially private optimization algorithm that works for any class of statistical queries that has a small separator set. Intuitively, it is attempting to implement the "Report-Noisy-Min" algorithm (see e.g. [DR14]), which outputs the query q that minimizes a (perturbed) estimate $\hat{q}(S) \equiv q(S) + Z_q$ where $Z_q \sim \text{Lap}(1/\varepsilon)$ for each $q \in \mathcal{Q}$. Because Report-Noisy-Min samples an independent perturbation for each query $q \in \mathcal{Q}$, it is inefficient: its run time is linear in $|\mathcal{Q}|$. Our algorithm – "Report Separator-Perturbed Min" (RSPM) – instead augments the dataset S in a way that implicitly induces perturbations of the query values $q(S)$. The perturbations are no longer independent across queries, and so to prove privacy, we need to use the structure of a separator set.

The algorithm is straightforward: it simply augments the dataset with one copy of each element of the separator set, each with a weight drawn independently from the Laplace distribution.

All original elements in the dataset are assigned weight 1. The algorithm then simply passes this weighted dataset to the weighted optimization oracle, and outputs the resulting query. The number of random variables that need to be sampled is therefore now equal to the size of the separator set, instead of the size of $\mathcal{Q}$. The algorithm is closely related to a no-regret learning algorithm given in [SKS16] — the only difference is in the magnitude of the noise added, and in the analysis, since we need a substantially stronger form of stability.

Report Separator-Perturbed Min (RSPM)
Given: A separator set $U = \{e_1, \ldots, e_m\}$ for a class of statistical queries $\mathcal{Q}$, a weighted optimization oracle $\mathcal{O}^*$ for $\mathcal{Q}$, and a privacy parameter ϵ.
Input: A dataset $S \in \mathcal{X}^n$ of size n.
Output: A statistical query $q \in \mathcal{Q}$.

Sample $\eta_i \sim Lap(m/\epsilon)$ for $i \in \{1, \ldots, m\}$
Construct a weighted dataset WD of size $n + m$ as follows:

$$WD(S, \eta) = \{(x_i, 1) : x_i \in S\} \cup \{(e_i, \eta_i) : e_i \in U\}$$

Output $q = \mathcal{O}^*(WD(S, \eta))$.

It is thus immediate that the Report Separator-Perturbed Min algorithm is oracle-efficient whenever the size of the separator set m is polynomial: it simply augments the dataset with a single copy of each of m separator elements, makes m draws from the Laplace distribution, and then makes a single call to the oracle:

Theorem 4. *The Report Separator-Perturbed Min algorithm is oracle-efficient.*

The accuracy analysis for the Report Separator-Perturbed Min algorithm is also straightforward, and follows by bounding the weighted sum of the additional entries added to the original data set.

Theorem 5. *The Report Separator-Perturbed Min algorithm is an (α, β)-minimizer for $\mathcal{Q}$ for:*

$$\alpha = \frac{2m^2 \log(m/\beta)}{\epsilon n}$$

Proof. Let q' be the query returned by RSPM, and let q^* be the true minimizer $q^* = \arg\min_{q \in \mathcal{Q}} q^*(S)$. Then we show that with probability $1 - \beta, q'(S) \leq q^*(S) + \alpha$. By the CDF of the Laplace distribution and a union bound over the m random variables η_i, we have that with probability $1 - \beta$:

$$\forall i,\ |\eta_i| \leq \frac{m \log(m/\beta)}{\epsilon}.$$

Since for every query q, $q(e_i) \in [0, 1]$, this means that with probability $1 - \beta$, $q'(WD) \geq q'(S) - m \cdot \frac{m \log(m/\beta)}{\epsilon n}$. Similarly $q^*(WD) \leq q^*(S) + m \cdot \frac{m \log(m/\beta)}{\epsilon n}$. Combining these bounds gives:

$$q'(S) \leq q'(WD) + m^2 \frac{\log(m/\beta)}{\epsilon n} \leq q^*(WD) + m^2 \frac{\log(m/\beta)}{\epsilon n} \leq q^*(S) + \frac{2m^2 \log(m/\beta)}{\epsilon n}$$

as desired, where the second inequality follows because by definition, q' is the true minimizer on the weighted dataset WD. □

Remark 1. *We can bound the expected error of RSPM using Theorem 5 as well. If we denote the error of RSPM by E, we've shown that for all β, $\Pr\left[E \geq \frac{2m^2 \log(m/\beta)}{\epsilon n}\right] \leq \beta$. Thus $\Pr\left[\frac{\epsilon n E}{2m^2} - \log m \geq \log(1/\beta)\right] \leq \beta$ for all β. Let $\tilde{E} = \max(0, \frac{\epsilon n E}{2m^2} - \log m)$. Since $\tilde{E}$ is non-negative:*

$$\mathbb{E}\left[\tilde{E}\right] = \int_0^\infty \Pr\left[\tilde{E} \geq t\right] \leq \int_0^\infty e^{-t} = 1.$$

Hence $\frac{\epsilon n \mathbb{E}[E]}{2m^2} - \log m \leq \mathbb{E}\left[\tilde{E}\right] \leq 1$, and so $\mathbb{E}[E] \leq \frac{2m^2}{\epsilon n}(1 + \log m)$.

The privacy analysis is more delicate, and relies on the correctness of the oracle.

Theorem 6. *If $\mathcal{O}^*$ is a weighted optimization oracle for $\mathcal{Q}$, then the Report Separator-Perturbed Min algorithm is ϵ-differentially private.*

Proof. We begin by introducing some notation. Given a weighted dataset $WD(S, \eta)$, and a query $q \in \mathcal{Q}$, let $q(S, \eta) = q(S) + \sum_{e_i \in U} q(e_i)\eta_i$ be the value when q is evaluated on the weighted dataset given the realization of the noise η. To allow us to distinguish queries that are output by the algorithm on different datasets and different realizations of the perturbations, write $\mathcal{Q}(S, \eta) = \mathcal{O}^*(WD(S, \eta))$. Fix any $q \in \mathcal{Q}$, and define:

$$\mathcal{E}(q, S) = \{\eta : \mathcal{Q}(S, \eta) = q\}$$

to be the event defined on the perturbations η that the mechanism outputs query q. Given a fixed $q \in \mathcal{Q}$ we define a mapping $f_q(\eta) : \mathbb{R}^m \to \mathbb{R}^m$ on noise vectors as follows:

1. If $q(e_i) = 1, f_q(\eta)_i = \eta_i - 1$
2. If $q(e_i) = 0, f_q(\eta)_i = \eta_i + 1$

Equivalently, $f_q(\eta)_i = \eta_i + (1 - 2q(e_i))$.

We now make a couple of observations about the function f_q.

Lemma 2. *Fix any $\hat{q} \in \mathcal{Q}$ and any pair of neighboring datasets S, S'. Let $\eta \in \mathcal{E}(\hat{q}, S)$ be such that $\hat{q}$ is the unique minimizer $\hat{q} \in \inf_{q \in \mathcal{Q}} q(S, \eta)$. Then $f_{\hat{q}}(\eta) \in \mathcal{E}(\hat{q}, S')$. In particular, this implies that for any such η:*

$$\mathbb{1}(\eta \in \mathcal{E}(\hat{q}, S)) \leq \mathbb{1}(f_{\hat{q}}(\eta) \in \mathcal{E}(\hat{q}, S'))$$

Proof. For this argument, it will be convenient to work with *un-normalized* versions of our queries, so that $q(S) = \sum_{x_i \in S} q(x_i)$ — i.e. we do not divide by the dataset size n. Note that this change of normalization does not change the identity of the minimizer. Under this normalization, the queries q are now 1-sensitive, rather than $1/n$ sensitive.

Recall that $\mathcal{Q}(S, \eta) = \hat{q}$. Suppose for point of contradiction that $\mathcal{Q}(S', f_{\hat{q}}(\eta)) = \tilde{q} \neq \hat{q}$. This in particular implies that $\tilde{q}(S', f_{\hat{q}}(\eta)) \leq \hat{q}(S', f_{\hat{q}}(\eta))$.

We first observe that $\hat{q}(S', \eta) - \tilde{q}(S', \eta) < 1$. This follows because:

$$\tilde{q}(S', \eta) \geq \tilde{q}(S, \eta) - 1 > \hat{q}(S, \eta) - 1 \tag{1}$$

Here the first inequality follows because the un-normalized queries q are 1-sensitive, and the second follows because $\hat{q} \in \arg\min_{q \in \mathcal{Q}} q(S, \eta)$ is the unique minimizer.

Next, we write:

$$\tilde{q}(S', f_{\hat{q}}(\eta)) - \hat{q}(S', f_{\hat{q}}(\eta)) = \tilde{q}(S', \eta) - \hat{q}(S', \eta) + \sum_{i=1}^{m} (\tilde{q}(e_i) - \hat{q}(e_i))(f_{\hat{q}}(\eta_i) - \eta_i)$$

Consider each term in the final sum: $(\tilde{q}(e_i) - \hat{q}(e_i))(f_{\hat{q}}(\eta_i) - \eta_i)$. Observe that by construction, each of these terms is non-negative: Clearly if $\tilde{q}(e_i) = \hat{q}(e_i)$, then the term is 0. Further, if $\tilde{q}(e_i) \neq \hat{q}(e_i)$, then by construction, $(\tilde{q}(e_i) - \hat{q}(e_i))(f_{\hat{q}}(\eta_i) - \eta_i) = 1$. Finally, by the definition of a separator set, we know that there is at least one index i such that $\tilde{q}(e_i) \neq \hat{q}(e_i)$. Thus, we can conclude:

$$\tilde{q}(S', f_{\hat{q}}(\eta)) - \hat{q}(S', f_{\hat{q}}(\eta)) \geq \tilde{q}(S', \eta) - \hat{q}(S', \eta) + 1 > 0$$

where the final inequality follows from applying inequality 1. But rearranging, this means that $\hat{q}(S', f_{\hat{q}}(\eta)) < \tilde{q}(S', f_{\hat{q}}(\eta))$, which contradicts the assumption that $\mathcal{Q}(S', f_{\hat{q}}(\eta)) = \tilde{q}$. □

Let p denote the probability density function of the joint distribution of the Laplace random variables η, and by abuse of notation also of each individual η_i.

Lemma 3. *For any $r \in \mathbb{R}^m, q \in \mathcal{Q}$:*

$$p(\eta = r) \leq e^{\epsilon} p(\eta = f_q(r))$$

Proof. For any index i and $z \in \mathbb{R}$, we have $p(\eta_i = z) = \frac{\epsilon}{2m} e^{-|z|\epsilon/m}$. In particular, if $|x - y| \leq 1$, $p(\eta_i = y) \leq e^{\epsilon/m} p(\eta_i = x)$. Since for all i and $r \in \mathbb{R}^m$ $|f_q(r)_i - r_i| \leq 1$, we have:

$$\frac{p(\eta = f_q(r))}{p(\eta = r)} = \prod_{i=1}^{m} \frac{p(\eta_i = f_q(r)_i)}{p(\eta_i = r_i)} \leq \prod_{i=1}^{m} e^{\epsilon/m} = e^{\epsilon}.$$

□

Lemma 4. *Fix any class of queries $\mathcal{Q}$ that has a finite separator set $U = \{e_1, \ldots, e_m\}$. For every dataset S there is a subset $B \subseteq \mathbb{R}^m$ such that:*

1. $\Pr[\eta \in B] = 0$ *and*
2. *On the restricted domain $\mathbb{R}^m \setminus B$, there is a unique minimizer $q' \in \arg\min_{q \in \mathcal{Q}} q(S, \eta)$*

Proof. Let:

$$B = \{\eta : \left| \arg\min_{q \in \mathcal{Q}} (q(S) + \sum_{i=1}^{m} \eta_i q(e_i)) \right| > 1\}$$

be the set of η values that do *not* result in unique minimizers q'.
Because $\mathcal{Q}$ is a finite set[1], by a union bound it suffices to show that for any two distinct queries $q_1, q_2 \in \mathcal{Q}$,

$$\Pr_{\eta} \left[q_1(S) + \sum_{i=1}^{m} \eta_i q_1(e_i) = q_2(S) + \sum_{i=1}^{m} \eta_i q_2(e_i) \right] = 0.$$

[1] Any class of queries $\mathcal{Q}$ with a separator set of size m can be no larger than 2^m.

This follows from the continuity of the Laplace distribution. Let i be any index such that $q_1(e_i) \neq q_2(e_i)$ (recall that by the definition of a separator set, such an index is guaranteed to exist). For any fixed realization of $\{\eta_j\}_{j\neq i}$, there is a single value of η_i that equalizes $q_1(S,\eta)$ and $q_2(S,\eta)$. But any single value is realized with probability 0.

□

We now have enough to complete the proof. We have for any query $\hat{q}$:

$$\begin{aligned}
\Pr[RSPM(S) = \hat{q}] &= \Pr[\eta \in \mathcal{E}(\hat{q}, S)] \\
&= \int_{\mathbb{R}^m} p(\eta)\mathbb{1}(\eta \in \mathcal{E}(\hat{q}, S))d\eta \\
&= \int_{\mathbb{R}^m \setminus B} p(\eta)\mathbb{1}(\eta \in \mathcal{E}(\hat{q}, S))d\eta && B \text{ has 0 measure. (Lemma 4)} \\
&\leq \int_{\mathbb{R}^m \setminus B} p(\eta)\mathbb{1}(f_{\hat{q}}(\eta) \in \mathcal{E}(\hat{q}, S'))d\eta && \text{Lemma 4} \implies \text{Lemma 2} \\
&\leq \int_{\mathbb{R}^m \setminus B} e^{\epsilon} p(f_{\hat{q}}(\eta))\mathbb{1}(f_{\hat{q}}(\eta) \in \mathcal{E}(\hat{q}, S'))d\eta && \text{Lemma 3} \\
&\leq \int_{\mathbb{R}^m \setminus f_{\hat{q}}(B)} e^{\epsilon} p(\eta)\mathbb{1}(\eta \in \mathcal{E}(\hat{q}, S'))\left|\frac{\partial f_{\hat{q}}}{\partial \eta}\right| d\eta && \text{Change of variables } \eta \to f_{\hat{q}}(\eta) \\
&= \int_{\mathbb{R}^m} e^{\epsilon} p(\eta)\mathbb{1}(\eta \in \mathcal{E}(\hat{q}, S'))d\eta && f_{\hat{q}}(B) \text{ has 0 measure, } \left|\frac{\partial f_{\hat{q}}}{\partial \eta}\right| = 1 \\
&= e^{\epsilon} \Pr[\eta \in \mathcal{E}(\hat{q}, S')] \\
&= e^{\epsilon} \Pr[RSPM(S') = \hat{q}]
\end{aligned}$$

□

In Appendix B, we give a somewhat more complicated analysis to show that by using Gaussian perturbations rather than Laplace perturbations, it is possible to improve the accuracy of the RSPM algorithm by a factor of $\sqrt{m}$, at the cost of satisfying (ϵ, δ)-differential privacy:

Theorem 7. *The Gaussian RSPM algorithm is (ϵ, δ)-differentially private, and is an oracle-efficient (α, β)-minimizer for any class of functions $\mathcal{Q}$ that has a universal identifications sequence of size m for:*

$$\alpha = O\left(\frac{m\sqrt{m\ln(m/\beta)\ln(1/\delta)}}{\varepsilon n}\right)$$

See Appendix B for the algorithm and its analysis.

It is instructive to compare the accuracy that we can obtain with oracle-efficient algorithms to the accuracy that can be obtained via the (inefficient, and generally optimal) exponential mechanism based generic learner from [KLN+11]. The existence of a universal identification set for $\mathcal{Q}$ of size m implies $|\mathcal{Q}| \leq 2^m$ (and for many interesting classes of queries, including conjunctions, disjunctions, parities, and discrete halfspaces over the hypercube, this is an equality — see Appendix A). Thus, the exponential-mechanism based learner from [KLN+11] is (α, β)-accurate for:

$$\alpha = O\left(\frac{m + \log(1/\beta)}{\epsilon n}\right).$$

Comparing this bound to ours, we see that we can obtain oracle-efficiency at a cost of roughly a factor of $\sqrt{m}$ in our error bound. Whether or not this cost is necessary is an interesting open question.

We can conclude that for a wide range of hypothesis classes $\mathcal{Q}$ including boolean conjunctions, disjunctions, decision lists, discrete halfspaces, and several families of circuits of logarithmic depth (see Appendix A) there is an oracle-efficient differentially private learning algorithm that obtains accuracy guarantees within small polynomial factors of the optimal guarantees of the (inefficient) exponential mechanism.

3.2 A Robustly Differentially Private Oracle-Efficient Algorithm

The RSPM algorithm is not *robustly* differentially private, because its privacy proof depends on the oracle succeeding. This is an undesirable property for RSPM and other algorithms like it, because we do not expect to have access to *actual* oracles for hard problems even if we expect that there are certain families of problems for which we can reliably solve typical instances[2]. In this section, we show how to remedy this: we give a black box reduction, starting from a (non-robustly) differentially private algorithm $\mathcal{A}_{\mathcal{O}}$ that is implemented using a *certifiable* heuristic[3] oracle $\mathcal{O}$, and producing a robustly differentially private algorithm $\tilde{\mathcal{A}}_{\mathcal{O}}$ for solving the same problem. $\tilde{\mathcal{A}}_{\mathcal{O}}$ will be (ϵ, δ)-differentially private for a parameter δ that we may choose, and will have a factor of roughly $\tilde{O}(1/\delta)$ running time overhead on top of $\mathcal{A}_{\mathcal{O}}$. So if $\mathcal{A}_{\mathcal{O}}$ is oracle efficient, so is $\tilde{\mathcal{A}}_{\mathcal{O}}$ whenever the chosen value of $\delta \geq 1/\text{poly}(n)$. If the oracle never fails, then we can prove utility guarantees for it when $\mathcal{A}_{\mathcal{O}}$ has such guarantees, since it just runs $\mathcal{A}_{\mathcal{O}}$ (using a smaller privacy parameter) on a random sub-sample of the original dataset. But the privacy guarantees hold even in the worst case of the behavior of the oracle. We call this reduction the *Private Robust Subsampling Meta Algorithm* or **PRSMA**.

3.2.1 Intuition and Proof Outline

Before we describe the analysis of **PRSMA**, a couple of remarks are helpful in order to set the stage.

1. At first blush, one might be tempted to assert that if an oracle-efficient non-robustly differentially private algorithm is implemented using a certifiable heuristic oracle, then it will sample from a differentially private distribution *conditioned on the event that the heuristic oracle doesn't fail*. But a moment's thought reveals that this isn't so: the possibility of failures both on the original dataset S and on the (exponentially many) neighboring datasets S' can substantially change the probabilities of arbitrary events Ω, and how these probabilities differ between neighboring datasets.

[2]There may be situations in which it is acceptable to use non robustly differentially private oracle-efficient algorithms — for example, if the optimization oracle is so reliable that it has never been observed to fail on the domain of interest. But robust differential privacy provides a worst-case guarantee which is preferable.

[3]We recall that heuristics for solving integer programs (such as cutting planes methods, branch and bound, and branch and cut methods, as implemented in commercial solvers) and SAT solvers are certifiable.

Private Robust Subsampling Meta Algorithm (PRSMA)

Given: Privacy parameters $\epsilon, \delta \geq 0$ and an oracle-efficient differentially private algorithm $\mathcal{A}^{\epsilon}_{\mathcal{O}} : \mathcal{X}^n \to \mathcal{M}$, implemented with a certifiable heuristic oracle $\mathcal{O}$.

Input: A dataset $S \in \mathcal{X}^n$ of size n.

Output: An output $m \in \mathcal{M}$ or $\perp$ ("Fail").

1: Randomly partition S into $K = \frac{1}{\epsilon}(1 + \log(\frac{2}{\delta}))$ equally sized datasets $\{S_i\}_{i=1}^K$. (If n is not divisible by K, first discard $n \mod K$ elements at random.)
2: **for** $i = 1 \ldots K$ **do**
3: Set $o_i = PASS$
4: **for** $t = 1 \ldots \frac{\log(K/\delta)}{\delta}$ **do**
5: Compute $\mathcal{A}^{\epsilon'}_{\mathcal{O}}(S_i) = a_{it}$, where $\epsilon' = \frac{1}{\sqrt{8\frac{n}{K}\log(2K/\delta)}}$
6: If $a_{it} = \perp$, set $o_i = \perp$
7: **end for**
8: **end for**
9: Compute $T = \#\{o_i \neq \perp\}$. Let $\tilde{T} = T + z$, where $z \sim \mathrm{Lap}(\frac{1}{\epsilon})$.
10: Test if $\tilde{T} > \frac{1}{\epsilon}(1 + \log(\frac{1}{\delta}))$, if no output $\perp$ and halt. **Else:**
11: Sample a uniformly at random from $\{a_{it} : o_i \neq \perp\}$.
12: Output a.

2. Next, one might think of the following simple candidate solution: Run the algorithm $\mathcal{A}_{\mathcal{O}}(S)$ roughly $\tilde{O}(1/\delta)$ many times in order to check that the failure probability of the heuristic algorithm on S is $\ll \delta$, and then output a sample of $\mathcal{A}_{\mathcal{O}}(S)$ only if this is so. But this doesn't work either: the failure probability itself will change if we replace S with a neighboring dataset S', and so this won't be differentially private. In fact, there is no reason to think that the failure probability of $\mathcal{A}_{\mathcal{O}}$ will be a low sensitivity function of S, so there is no way to privately estimate the failure probability to non-trivial error.

It *is* possible to use the *subsample-and-aggregate* procedure of [NRS07] to randomly partition the dataset into K pieces S_i, and privately estimate on *how many* of these pieces $\mathcal{A}_{\mathcal{O}}(S_i)$ fails with probability $\ll \delta$. The algorithm can then then fail if this private count is not sufficiently large. In fact, this is the first thing that **PRSMA** does, in lines 1-10, setting $o_i = PASS$ for those pieces S_i such that it seems that the probability of failure is $\ll \delta$, and setting $o_i = \perp$ for the others.

But the next step of the algorithm is to randomly select one of the partition elements S_i amongst the set that passed the earlier test: i.e. amongst the set such that $o_i \neq \perp$ — and return one of the outputs a that had been produced by running $\mathcal{A}_{\mathcal{O}}(S_i)$. It is not immediately clear why this should be private, because *which* partition elements passed the test $\{i : o_i \neq \perp\}$ is not itself differentially private. Showing that this results in a differentially private output is the difficult part of the analysis.

To get an idea of the problem that we need to overcome, consider the following situation which our analysis must rule out: Fix a partition of the dataset $S_1, \ldots, S_K$, and imagine that each partition element passes: we have $o_i \neq \perp$ for all i. Now suppose that there is some event Ω such that $\Pr[\mathcal{A}_{\mathcal{O}}(S_1) \in \Omega] \geq 1/2$, but $\Pr[\mathcal{A}_{\mathcal{O}}(S_i) \in \Omega]$ is close to 0 for all $i \neq 1$. Since $K \approx 1/\epsilon$, and the final output is drawn from a uniformly random partition element, this means that **PRSMA** outputs an element of Ω with probability $\Omega(\epsilon)$. Suppose that on a neighboring dataset S', S_1 no longer passes

the test and has $o_1 = \perp$. Since it is no longer a candidate to be selected at the last step, we now have that on S', **PRSMA** outputs an element of Ω with probability close to 0. This is a violation of (ϵ, δ)-differential privacy for any non-trivial value of δ (i.e. $\delta \leq O(\epsilon)$).

The problem is that (fixing a partition of S into $S_1, \ldots, S_K$) moving to a neighboring dataset S' can potentially arbitrarily change the probability that any single element S_i survives to step 11 of the algorithm, which can in principle change the probability of arbitrary events Ω by an *additive* $\pm O(\epsilon)$ term, rather than a *multiplicative* $1 \pm O(\epsilon)$ *factor*.

Since we are guaranteed that (with high probability) if we make it to step 11 without failing, then at least $\Omega(1/\epsilon)$ elements S_i have survived with $o_i \neq \perp$, it would be sufficient for differential privacy if for every event Ω, the probabilities $\Pr[\mathcal{A}_{\mathcal{O}}(S_i) \in \Omega]$ were within a constant factor of each other, for all i. Then a change of whether a single partition element S_i survives with $o_i \neq \perp$ or not would only add or remove an ϵ *fraction* of the total probability mass on event Ω. While this seems like a "differential-privacy" like property, but it is not clear that the fact that $\mathcal{A}_{\mathcal{O}^*}$ is differentially private can help us here, because the partition elements S_i, S_j are not neighboring datasets — in fact, they are disjoint. But as we show, it does in fact guarantee this property *if* we set the privacy parameter ϵ' to be sufficiently small — to roughly $O(1/\sqrt{n/K})$ in step 5.

With this intuition setting the stage, the roadmap of the proof is as follows. For notational simplicity, we write $\mathcal{A}(\cdot)$ to denote $\mathcal{A}_{\mathcal{O}^*}(\cdot)$, the oracle-efficient algorithm when implemented with a perfect oracle.

1. We observe that ϵ-differential privacy implies that the log-probability of any event Ω when $\mathcal{A}(\cdot)$ is run on S_i changes by less than an additive factor of ϵ when an element of S_i is changed. We use a method of bounded differences argument to show that this implies that the log-probability density function concentrates around its expectation, where the randomness is over the subsampling of S_i from S. A similar result is proven in [CLN+16] to show that differentially private algorithms achieve what they call "perfect generalization." We need to prove a generalization of their result because in our case, the elements of S_i are not selected independently of one another. This guides our choice of ϵ' in step 5 of the algorithm. (Lemma 5)

2. We show that with high probability, for every S_i such that $o_i \neq \perp$ after step 10 of the algorithm, $\mathcal{A}_{\mathcal{O}}(S_i)$ fails with probability at most $O(\delta)$. By Lemma 1, this implies that it is δ-close in total variation distance to $\mathcal{A}(S_i)$.

3. We observe that fixing a partition, on a neighboring dataset, only one of the partition elements S_i changes — and hence changes its probability of having $o_i \neq \perp$. Since with high probability, conditioned on **PRSMA** not failing, $\Omega(1/\epsilon)$ partition elements survive with $o_i \neq \perp$, parts 1 and 2 imply that changing a single partition element S_i only changes the probability of realizing any outcome event by a *multiplicative* factor of $\approx 1 + \epsilon$.

3.2.2 The Main Theorem

Theorem 8. ***PRSMA*** *is (ϵ, δ) differentially private when given as input:*

1. *An oracle-efficient non-robustly differentially private algorithm $\mathcal{A}_{\mathcal{O}}$ implemented with a certifiable heuristic oracle $\mathcal{O}$, and*

2. *Privacy parameters (ϵ^*, δ^*) where $\epsilon^* = \frac{\epsilon}{62} \leq \frac{1}{2}$ and $\delta^* = \frac{\delta}{11} \leq \frac{1}{2}$.*

Proof. We analyze **PRSMA** with privacy parameters ϵ and δ, optimizing the constants at the end. Fix an input dataset S with $|S| = n$, and an adjacent dataset $S' \sim S$, such that without loss of generality S, S' differ in the element $x_1 \neq x_1'$. We denote the **PRSMA** routine with input $\mathcal{A}_{\mathcal{O}}$ and dataset S by $\mathcal{A}_{\mathcal{O}}^{prsm}(\mathcal{S})$. We first observe that:

$$\Pr[\mathcal{A}_{\mathcal{O}}^{prsm}(\mathcal{S}) = \perp] \leq e^{\epsilon} \Pr[\mathcal{A}_{\mathcal{O}}^{prsm}(\mathcal{S}') = \perp]$$

This is immediate since the indicator for a failure is a post-processing of the Laplace mechanism. Since x_1 can affect at most one oracle failure, T is 1-sensitive, and so publishing $\tilde{T} = T + \text{Lap}(\frac{1}{\epsilon})$ satisfies ϵ-differential privacy since it is an invocation of the Laplace Mechanism defined in Section 2.1. (This can also be viewed as an instantiation of the "sub-sample and aggregate procedure of [NRS07]).

We now proceed to the meat of the argument. To establish (ϵ, δ) differential privacy we must reason about the probability of arbitrary events $\Omega \subset \mathcal{M}$, rather than just individual outputs a. We want to show:

$$\Pr[\mathcal{A}_{\mathcal{O}}^{prsm}(\mathcal{S}) \in \Omega] \leq e^{\epsilon} \Pr[\mathcal{A}_{\mathcal{O}}^{prsm}(\mathcal{S}') \in \Omega] + \delta$$

We first fix some notation and define a number of events that we will need to reason about. Let:

- $\mathcal{P}_{split}^{S}$ be the uniform distribution over equal sized partitions of S that the datasets S_i are drawn from in line 1; i.e. $\mathcal{P}(S) \sim \mathcal{P}_{split}^{S}$, where $\mathcal{P}(S)$ is the partition of S into $\{S_i\}$.
- $\mathcal{A}$ denote $\mathcal{A}_{\mathcal{O}^*}$, our oracle-efficient algorithm when instantiated with a perfect oracle $\mathcal{O}^*$. i.e. $\mathcal{A}(S)$ is the ϵ-differentially private distribution that we ideally want to sample from.
- Z be the event that the Laplace noise z in step 10 of **PRSMA** has magnitude greater than $\frac{1}{\epsilon}(\log(\frac{2}{\delta}))$.
- $\mathcal{F} = \left\{\{o_i\} : |\{o_i : o_i \neq \perp\}| > \frac{1}{\epsilon}\right\}$. We will use $\mathbf{o}$ to denote a particular set $\{o_i\}$. Let $I_{pass}^{\mathbf{o}}$ be the set $\{i : o_i \neq \perp\}$. Given $\mathbf{o} \in \mathcal{F}$, let $|\mathbf{o}|$ denote $|I_{pass}^{\mathbf{o}}|$.
- E be the event that for all $i = 1 \ldots K$: $\Pr[\mathcal{A}_{\mathcal{O}}(S_i) = \perp] \geq \delta \Rightarrow o_i = \perp$.
- i^* denote the index i of the randomly chosen a_{it} in step 11 of **PRSMA**.
- Q be the event that the draw $\mathcal{P}(S) \sim \mathcal{P}_{split}^{S}$ is such that the probabilities of $\mathcal{A}$ outputting $a \in \Omega$ when run on any two $S_i, S_j \in \mathcal{P}(S)$ are within a multiplicative factor of 2. Lemma 5 formally defines Q and shows $\Pr[Q] \geq 1 - \delta$. Let S_Q denote the set of $\mathcal{P}(S)$ on which event Q holds.

We now bound the probabilities of several of these events. By the CDF of the Laplace distribution, we have $\Pr[Z] = \Pr\left[|z| > \frac{1}{\epsilon}\log(1/\delta)\right] = e^{-\epsilon \cdot \frac{1}{\epsilon}\log(1/\delta)} = \delta$, and by a union bound:

$$\Pr[E] \geq 1 - K \cdot (1-\delta)^{(\log(K/\delta)/\delta)} \geq 1 - K \cdot e^{-\delta \cdot \log(K/\delta)/\delta} = 1 - \delta.$$

Let $\mathcal{L}$ be the event $Z^c \cap E$. By the above calculation and another union bound, $\Pr[\mathcal{L}] \geq 1 - 2\delta$. Our proof now proceeds via a sequence of lemmas. All missing proofs appear in Appendix C. We first show that Q occurs with high probability.

Lemma 5. *Let $\mathcal{P}(S) \sim \mathcal{P}^S_{split}$. Let $\mathcal{A} : \mathcal{X}^l \to \mathcal{M}$ be an $(\epsilon', 0)$ differentially private algorithm, where: $\epsilon' = \frac{1}{\sqrt{8\frac{n}{K}\log(2K/\delta)}}$. Fix $\Omega \subset M$, and let $q_\Omega(S_i) = \log \Pr[\mathcal{A}(S_i) \in \Omega]$. Define Q to be the event*

$$Q = \{\mathcal{P}(S) : \max_{i,j \in 1\ldots K} |q_\Omega(S_i) - q_\Omega(S_j)| \leq 2\}.$$

Then over the random draw of $\mathcal{P}(S) \sim \mathcal{P}^S_{split}$, $\Pr[Q] \geq 1 - \delta$.

The proof relies on the fact that $q_\Omega(\cdot)$ is ϵ-Lipschitz. This result is similar to Theorem 5.4 in [CLN+16], although in our case sampling without replacement induces dependence among the elements in S_i, and thus we can't appeal to standard concentration inequalities for independent random variables. Instead we prove that elements sampled without replacement from a fixed set satisfy a type of negative dependence called *stochastic covering*, and are n/K-homogenous (supported on a set of size n/K), which are used to prove exponential concentration of Lipschitz functions in [PP14]. We defer the details to the Appendix.

To establish the theorem we want to show that given an adjacent database S' differing only in the first element from S, that

$$\frac{\Pr[\mathcal{A}^{prsm}_{\mathcal{O}}(\mathcal{S}) \in \Omega]}{\Pr[\mathcal{A}^{prsm}_{\mathcal{O}}(\mathcal{S}') \in \Omega]} \leq e^{\epsilon^*} + \frac{\delta^*}{\Pr[\mathcal{A}^{prsm}_{\mathcal{O}}(\mathcal{S}') \in \Omega]} \tag{2}$$

Our analysis will proceed by expanding the numerator by first conditioning on $\mathcal{L}$, and then on particular realizations of the partition $\mathcal{P}(S)$, and on a fixed realization of $\mathbf{o} = \{o_i\}$. We can also restrict our attention to only summing over $\mathcal{P}(S) \in S_Q$, by showing that the terms corresponding to $\mathcal{P}(S) \in S_Q^c$ contribute at most an additive factor of 2δ to the final probability. We will also only sum over $\mathbf{o} \in \mathcal{F}$ since conditioned on Z^c, which is implied by $\mathcal{L}$, these are the only $\mathbf{o}$ such that $\Pr[\mathcal{A}^{prsm}_{\mathcal{O}}(\mathcal{S}) \in \Omega | \mathbf{o}] \neq 0$.

Lemma 6.

$$\frac{\Pr[\mathcal{A}^{prsm}_{\mathcal{O}}(\mathcal{S}) \in \Omega]}{\Pr[\mathcal{A}^{prsm}_{\mathcal{O}}(\mathcal{S}') \in \Omega]} \leq \frac{\sum_{\mathcal{P}(S) \in S_Q, \boldsymbol{o} \in \mathcal{F}} \Pr[\mathcal{A}^{prsm}_{\mathcal{O}}(\mathcal{S}) \in \Omega | P(S), \boldsymbol{o}, \mathcal{L}] \Pr[\boldsymbol{o}, P(S) | \mathcal{L}] + 4\delta}{\Pr[\mathcal{A}^{prsm}_{\mathcal{O}}(\mathcal{S}') \in \Omega]}$$

The rest of the proof will consist of upper bounding the individual terms $\frac{\Pr[\mathcal{A}^{prsm}_{\mathcal{O}}(\mathcal{S}) \in \Omega | P(S), \mathbf{o}, \mathcal{L}] \Pr[\mathbf{o}, P(S) | \mathcal{L}]}{\Pr[\mathcal{A}^{prsm}_{\mathcal{O}}(\mathcal{S}') \in \Omega]}$. Lemma 7 is a tool used to prove Lemma 8, which upper bounds the numerator, and Lemma 9 lower bounds the denominator. The conclusion of the argument consists of manipulations to upper bound the ratio of these two bounds.

We first analyze the $\Pr[\mathcal{A}^{prsm}_{\mathcal{O}}(\mathcal{S}) \in \Omega | P(S), \mathbf{o}, \mathcal{L}]$ term, for $\mathcal{P}(S) \in S_Q, \mathbf{o} \in \mathcal{F}$. Conditioned on Z^c, if $\mathbf{o} \in \mathcal{F}$, then **PRSMA** passes the test in step 10 and outputs a randomly chosen $a_{it} : i \in I^{\mathbf{o}}_{pass}$. Fixing a sampled value $i^* \in I^{\mathbf{o}}_{pass}$, a_{i^*t} is distributed identically to $a \sim \mathcal{A}_{\mathcal{O}}(S_{i^*}) | \mathcal{A}_{\mathcal{O}}(S_{i^*}) \neq \perp$, since after conditioning on S_{i^*} each a_{i^*t} is drawn iid and the event $o_i \neq \perp$ does not depend on the sampled values a_{it}. In other words, $\Pr[\mathcal{A}^{prsm}_{\mathcal{O}}(\mathcal{S}) \in \Omega | i^* = i, P(S), \mathbf{o}] = \Pr[\mathcal{A}_{\mathcal{O}}(S_{i^*}) \in \Omega | \mathcal{A}_{\mathcal{O}}(S_{i^*}) \neq \perp]$, and so:

$$\Pr[\mathcal{A}^{prsm}_{\mathcal{O}}(\mathcal{S}) \in \Omega | P(S), \mathbf{o}, \mathcal{L}] = \frac{1}{|\mathbf{o}|} \sum_{i \in I^{\mathbf{o}}_{pass}} \Pr[\mathcal{A}_{\mathcal{O}}(S_i) \in \Omega | \mathcal{A}_{\mathcal{O}}(S_i) \neq \perp] \tag{3}$$

Substituting in 3 we have:

$$\sum_{\mathcal{P}(S)\in S_Q,\mathbf{o}\in\mathcal{F}} \Pr[\mathcal{A}_{\mathcal{O}}^{prsm}(\mathcal{S})\in\Omega|P(S),\mathbf{o},\mathcal{L}]\Pr[\mathbf{o},P(S)|\mathcal{L}] = \sum_{\mathcal{P}(S)\in S_Q,\mathbf{o}\in\mathcal{F}} \frac{1}{|\mathbf{o}|}\sum_{i\in I^{\mathbf{o}}_{pass}} \Pr[\mathcal{A}_{\mathcal{O}}(S_i)\in\Omega|\mathcal{A}_{\mathcal{O}}(S_i)\neq\perp]\Pr[\mathbf{o},P(S)|\mathcal{L}]$$

We now use the fact that $\mathcal{P}(S)\in S_Q$, to show that none of the terms $\Pr[\mathcal{A}_{\mathcal{O}}(S_i)\in\Omega|\mathcal{A}_{\mathcal{O}}(S_i)\neq\perp]$ in the right hand side of Equation 3 individually represent a substantial fraction of the total probability mass. Conditioning on $\mathcal{L}$ ensures that if $o_i\neq\perp$ then $\Pr[\mathcal{A}_{\mathcal{O}}(S_i)=\perp]\leq\delta$, which means $\mathcal{A}_{\mathcal{O}}(S_i)$ is δ-close in total variation distance to $\mathcal{A}(S_i)$. By Lemma 5, with high probability any $\mathcal{A}(S_i)$ is approximately equally likely to output an element in Ω, which allows us to bound the effect that changing a single data point (and hence a single S_i) can have on the probability of outputting an element in Ω.

Lemma 7. *Fix any $\mathbf{o}$, any $\mathcal{P}(S)\in S_Q$, and index $j\in I^{\mathbf{o}}_{pass}$, i.e. $o_j\neq\perp$. Then:*

$$\Pr[\mathcal{A}_{\mathcal{O}}(S_j)\in\Omega|\mathcal{A}_{\mathcal{O}}(S_j)\neq\perp,\mathcal{L}]\leq\frac{e^2}{(1-\delta)^2}\frac{1}{|\boldsymbol{o}|-1}\sum_{i\in I^{\mathbf{o}}_{pass},i\neq j}\Pr[\mathcal{A}_{\mathcal{O}}(S_i)\in\Omega|\mathcal{A}_{\mathcal{O}}(S_i)\neq\perp]+\frac{\delta e^2}{(1-\delta)}$$

Without loss of generality (up to renaming of partition elements), assume that the element on which S and S' differ falls into S_1. Now we break the summation over $\mathcal{O}$ into two pieces, depending on whether $o_1=\perp$ or $o_1\neq\perp$. We will use Lemma 7 to bound terms involving $\Pr[\mathcal{A}_{\mathcal{O}}(S_1)\in\Omega|\mathcal{A}_{\mathcal{O}}(S_1)\neq\perp,\mathcal{L}]$, since S_1 is the only partition where $S_i\neq S'_i$.

$$\sum_{P(S)\in S_Q}\Bigg(\sum_{\mathbf{o}\in\mathcal{F}}\Big(\frac{1}{|\mathbf{o}|}\sum_{i\in I^{\mathbf{o}}_{pass}}\Pr[\mathcal{A}_{\mathcal{O}}(S_i)\in\Omega|\mathcal{A}_{\mathcal{O}}(S_i)\neq\perp]\Big)\Pr[\mathbf{o}|P(S)]\Bigg)\Pr[P(S)]=$$

$$\sum_{P(S)\in S_Q}\Bigg(\sum_{\mathbf{o}\in\mathcal{F}:o_1\neq\perp}\Big(\frac{1}{|\mathbf{o}|}\sum_{i\in I^{\mathbf{o}}_{pass}}\Pr[\mathcal{A}_{\mathcal{O}}(S_i)\in\Omega|\mathcal{A}_{\mathcal{O}}(S_i)\neq\perp]\Big)\Pr[\mathbf{o}|P(S)]+$$

$$\sum_{\mathbf{o}\in\mathcal{F}:o_1=\perp}\Big(\frac{1}{|\mathbf{o}|}\sum_{i\in I^{\mathbf{o}}_{pass}}\Pr[\mathcal{A}_{\mathcal{O}}(S_i)\in\Omega|\mathcal{A}_{\mathcal{O}}(S_i)\neq\perp]\Big)\Pr[\mathbf{o}|P(S)\Bigg)\Pr[P(S)]$$

Lemma 8.

$$\sum_{\mathcal{P}(S)\in S_Q,\boldsymbol{o}\in\mathcal{F}}\Pr[\mathcal{A}_{\mathcal{O}}^{prsm}(\mathcal{S})\in\Omega|P(S),\boldsymbol{o},\mathcal{L}]\Pr[\boldsymbol{o},P(S)|\mathcal{L}]\leq$$

$$(1+\frac{e^2}{(1-\delta)^2(\frac{1}{\epsilon}-1)})\sum_{P(S)\in S_Q}\Bigg(\sum_{\boldsymbol{o}\in\mathcal{F}}\Big(\frac{1}{|\boldsymbol{o}|}\sum_{i\in I^{\mathbf{o}}_{pass},i\neq 1}\Pr[\mathcal{A}_{\mathcal{O}}(S_i)\in\Omega|\mathcal{A}_{\mathcal{O}}(S_i)\neq\perp]\Big)\Pr[\boldsymbol{o}|P(S)]\Bigg)\Pr[P(S)]+\frac{\epsilon\delta e^2}{1-\delta}$$

We now condition on a fixed partition $P(S')$ in the denominator as well. Given a fixed partition $P(S)$ of S, define the adjacent partition $\mathcal{P}(S')\sim\mathcal{P}(S)$ as the partition of an adjacent database S' such that for all $i\neq 1$, $S_i=S'_i$, and S_1,S'_1 differ only in $x_1\neq x'_1$, where x_1 is the differing element between S,S'. Let S'_Q be the set of $\mathcal{P}(S')$ adjacent to $\mathcal{P}(S)\in S_Q$, i.e. $S'_Q=\{\mathcal{P}(S'):\exists\,\mathcal{P}(S)\in S_Q,\mathcal{P}(S')\sim\mathcal{P}(S)\}$. We now lower bound the denominator in Lemma 6, which follows by conditioning on $\mathbf{o},\mathcal{P}(S')$, and then dropping some (non-negative) terms.

Lemma 9.

$$\Pr[\mathcal{A}_{\mathcal{O}}^{prsm}(\mathcal{S}') \in \Omega] \geq \sum_{P(S') \in S'_Q} \Big(\sum_{\boldsymbol{o} \in \mathcal{F}} \Big(\frac{1}{|\boldsymbol{o}|} \sum_{i \in I^{\boldsymbol{o}}_{pass}, i \neq 1} \Pr[\mathcal{A}_{\mathcal{O}}(S'_i) \in \Omega | \mathcal{A}_{\mathcal{O}}(S'_i) \neq \perp]\Big) \Pr[\boldsymbol{o}|P(S')]\Big) \Pr[P(S')]$$

Thus by Lemmas 8 and 9,

$$\frac{\Pr[\mathcal{A}_{\mathcal{O}}^{prsm}(\mathcal{S}) \in \Omega | P(S), \mathbf{o}, \mathcal{L}] \Pr[\mathbf{o}, P(S)|\mathcal{L}]}{\Pr[\mathcal{A}_{\mathcal{O}}^{prsm}(\mathcal{S}') \in \Omega]} \leq$$

$$\frac{(1 + \frac{e^2}{(1-\delta)^2(\frac{1}{\epsilon}-1)}) \sum_{P(S) \in S_Q} \Big(\sum_{\mathbf{o} \in \mathcal{F}} \Big(\frac{1}{|\mathbf{o}|} \sum_{i \in I^{\mathbf{o}}_{pass}, i \neq 1} \Pr[\mathcal{A}_{\mathcal{O}}(S_i) \in \Omega | \mathcal{A}_{\mathcal{O}}(S_i) \neq \perp] \Pr[\mathbf{o}|P(S)]\Big) \Pr[P(S)]\Big)}{\sum_{P(S') \in S'_Q} \Big(\sum_{\mathbf{o} \in \mathcal{F}} \Big(\frac{1}{|\mathbf{o}|} \sum_{i \in I^{\mathbf{o}}_{pass}} \Pr[\mathcal{A}_{\mathcal{O}}(S'_i) \in \Omega | \mathcal{A}_{\mathcal{O}}(S'_i) \neq \perp]\Big) \Pr[\mathbf{o}|P(S')]\Big) \Pr[P(S')]} + \frac{\frac{\epsilon\delta e^2}{1-\delta}}{\Pr[\mathcal{A}_{\mathcal{O}}^{prsm}(\mathcal{S}') \in \Omega]} \tag{4}$$

For all $\mathcal{P}(S), \mathcal{P}(S'), \Pr[\mathcal{P}(S)] = \Pr[\mathcal{P}(S')]$, and we can bound the ratio of the summations over S_Q, S'_Q by the supremum of the ratio. Hence (4) $\leq$

$$\sup_{P(S) \sim P(S')} \frac{(1 + \frac{e^2}{(1-\delta)^2(\frac{1}{\epsilon}-1)}) \sum_{\mathbf{o} \in \mathcal{F}} \Big(\frac{1}{|\mathbf{o}|} \sum_{i \in I^{\mathbf{o}}_{pass}, i \neq 1} \Pr[\mathcal{A}_{\mathcal{O}}(S_i) \in \Omega | \mathcal{A}_{\mathcal{O}}(S_i) \neq \perp] \Pr[\mathbf{o}|P(S)]\Big)}{\sum_{\mathbf{o} \in \mathcal{F}} \Big(\frac{1}{|\mathbf{o}|} \sum_{i \in I^{\mathbf{o}}_{pass}} \Pr[\mathcal{A}_{\mathcal{O}}(S'_i) \in \Omega | \mathcal{A}_{\mathcal{O}}(S'_i) \neq \perp] \Pr[\mathbf{o}|P(S')]\Big)} + \frac{\frac{\epsilon\delta e^2}{1-\delta}}{\Pr[\mathcal{A}_{\mathcal{O}}^{prsm}(\mathcal{S}') \in \Omega]} \tag{5}$$

Now since for all $i \neq 1, S_i = S'_i$, if we could control the ratio $\Pr[\mathbf{o}|\mathcal{P}(S)]/\Pr[\mathbf{o}|\mathcal{P}(S')]$ we would be done. But this ratio could potentially be unbounded, as $\Pr[\mathbf{o}_1|P(S)]$ could be nonzero, and the substitution of x'_1 for x_1 could force failure on the first partition S'_1, and so $\Pr[\mathbf{o}_1|P(S')] = 0$.

Given $\mathbf{o}$ let $\mathbf{o}_{-1}$ denote $\{o_2, \ldots o_K\}$. The remainder of the argument circumvents this obstacle by decomposing the outer summation over $\mathbf{o} \in \mathcal{F}$ into a summation over the indicators of all but the first failure event ($\mathbf{o}_{-1}$), and integrating out the probability of the first failure event (o_1) from the joint probability $\Pr[\mathbf{o}|P(S)]$. This trick will be applied in the numerator and denominator, with a slight difference corresponding to an upper and a lower bound respectively. See the end of Section **C** of the Appendix for details. Following the chain of inequalities, we finally obtain:

$$\frac{\Pr[\mathcal{A}_{\mathcal{O}}^{prsm}(\mathcal{S}) \in \Omega | P(S), \mathbf{o}, \mathcal{L}] \Pr[\mathbf{o}, P(S)|\mathcal{L}]}{\Pr[\mathcal{A}_{\mathcal{O}}^{prsm}(\mathcal{S}') \in \Omega]} \leq (1 + \frac{e^2}{(1-\delta)^2(\frac{1}{\epsilon}-1)}) \frac{1}{1-\epsilon} + \frac{\frac{\epsilon\delta e^2}{1-\delta}}{\Pr[\mathcal{A}_{\mathcal{O}}^{prsm}(\mathcal{S}') \in \Omega]},$$

which substituting into Lemma 6 gives:

$$\Pr[\mathcal{A}_{\mathcal{O}}^{prsm}(\mathcal{S}) \in \Omega] \leq (1 + \frac{e^2}{(1-\delta)^2(\frac{1}{\epsilon}-1)}) \frac{1}{1-\epsilon} \Pr[\mathcal{A}_{\mathcal{O}}^{prsm}(\mathcal{S}') \in \Omega] + 4\delta + \frac{\epsilon\delta e^2}{1-\delta}$$

For $\epsilon, \delta \leq 1/2$, $(1 + \frac{e^2}{(1-\delta)^2(\frac{1}{\epsilon}-1)}) \frac{1}{1-\epsilon} \leq e^{8e^2\epsilon + \epsilon + \epsilon^2}$, which establishes that **PRSMA** is $(8e^2\epsilon + \epsilon + \epsilon^2, 4\delta + \frac{\epsilon\delta e^2}{1-\delta})$ differentially private. Setting $\epsilon = \epsilon^*, \delta = \delta^*$ completes the proof.

□

We now turn to **PRSMA**'s accuracy guarantees. Note that when **PRSMA** starts with an algorithm $\mathcal{A}_{\mathcal{O}^*}$ instantiated with a perfect oracle $\mathcal{O}^*$, it with high probability outputs the result of running $\mathcal{A}_{\mathcal{O}^*}$ on a subsampled dataset S_i of size $n/K \approx \epsilon n$, with privacy parameter $\epsilon' = \frac{1}{\sqrt{8\frac{n}{K}\log(2K/\delta)}}$. In general, therefore, the accuracy guarantees of **PRSMA** depend on how robust the guarantees of $\mathcal{A}$ are to subsampling, which is typical of "Subsample and Aggregate" approaches, and also to its specific privacy-accuracy tradeoff. Learning algorithms are robust to sub-sampling however: below we derive an accuracy theorem for **PRSMA** when instantiated with our oracle-efficient RSPM algorithm.

Theorem 9. *Let $\mathcal{Q}$ a class of statistical queries with a separator set of size m. Let $\mathcal{A}_{\mathcal{O}^*}$ denote the RSPM algorithm with access to $\mathcal{O}^*$, a perfect weighted optimization oracle for $\mathcal{Q}$. Then **PRSMA** instantiated with $\mathcal{A}_{\mathcal{O}^*}$, run on a dataset S of size n, with input parameters ϵ and δ is an (α, β)-minimizer for any $\beta > \delta$ and*

$$\alpha \leq \tilde{O}\left(\frac{m^2 \log\left(\frac{m}{\beta-\delta}\right)\log(1/\delta) + \sqrt{\log(1/\delta)\log\left(\frac{|\mathcal{Q}|}{\beta-\delta}\right)}}{\sqrt{n\epsilon}}\right),$$

where the $\tilde{O}$ hides logarithmic factors in $\frac{1}{\epsilon}, \log(\frac{1}{\delta})$.

Proof. With probability at least $1-\delta$, **PRSMA** outputs the result of RSPM run on an n/K fraction of the dataset, with privacy parameter $\epsilon' = \frac{1}{\sqrt{8\frac{n}{K}\log(2K/\delta)}}$. We will condition on this event for the remainder of the proof, which occurs except with probability δ.

Let q^* denote the true minimizer on S. Let S_K denote the random subsample, and let q_K denote the true minimizer on S_K. By Theorem 5, we know that for any $\eta > 0$, with probability $1-\eta$, the error on S_K is bounded as follows:

$$\hat{q}(S_K) - q_K(S_K) \leq \frac{2m^2 \log(m/\eta)}{\epsilon' \frac{n}{K}}$$

We next bound $\max_{q\in\mathcal{Q}} q(S_K) - q(S)$, the maximum difference between the value that any query takes on S_K compared to the value that it takes on S. By a Chernoff bound for subsampled random variables (see e.g. Theorem 1.2 of [BM13]), for any $q \in \mathcal{Q}, t > 0$,

$$\Pr\left[q(S_K) - q(S) \geq t\right] \leq \exp\left(-2\frac{n}{K}t^2\right).$$

By a union bound over $\mathcal{Q}$, this means that with probability $1-\eta$,

$$\max_{q\in\mathcal{Q}} q(S_K) - q(S) \leq \sqrt{\frac{K}{2n}\log\left(\frac{2|\mathcal{Q}|}{\eta}\right)}$$

We now have all the ingredients to complete the bound:

$$\begin{aligned}\hat{q}(S) - q^*(S) &= [\hat{q}(S) - \hat{q}(S_K)] + [\hat{q}(S_K) - q^*(S_K)] + [q^*(S_K) - q^*(S)] \\ &\leq 2\max_{q\in\mathcal{Q}} |q(S_K) - q(S)| + \hat{q}(S_K) - q^*(S_K) \\ &= 2\max_{q\in\mathcal{Q}} |q(S_K) - q(S)| + \hat{q}(S_K) - q_K(S_K) + q_K(S_K) - q^*(S_K) \\ &\leq 2\max_{q\in\mathcal{Q}} |q(S_K) - q(S)| + \hat{q}(S_K) - q_K(S_K).\end{aligned}$$

By a union bound and the results above, we know that the righthand side is less than

$$2\sqrt{\frac{K}{2n}\log(\frac{2|\mathcal{Q}|}{\eta})} + \frac{2m^2\log(m/\eta)}{\epsilon'\frac{n}{K}},$$

with probability at least $1-\eta$. Substituting $\eta = \beta - \delta$, $\epsilon' = \frac{1}{\sqrt{8\frac{n}{K}\log(2K/\delta)}}$, $K = O(\frac{1}{\epsilon}(1+\log(2/\delta)))$ gives the desired result. □

We remark that we can convert this (α,β)-accuracy bound into a bound on the expected error using the same technique we used to compute the expected error of RSPM. The expected error of **PRSMA** with the above inputs is $\tilde{O}(\frac{2m^2(\log m+1)}{\sqrt{n\epsilon}} + \frac{\sqrt{(\log|\mathcal{Q}|+1)}}{\sqrt{n\epsilon}})$.

4 OracleQuery: Oracle-Efficient Private Synthetic Data Generation

We now apply the oracle-efficient optimization methods we have developed to the problem of generating *private synthetic data*. In particular, given a private dataset S and a query class $\mathcal{Q}$, we would like to compute a synthetic dataset $\hat{S}$ subject to differential privacy such that the error $\max_{q\in\mathcal{Q}} |q(\hat{S}) - q(S)|$ is bounded by some target parameter α. We provide a general algorithmic framework called OracleQuery for designing oracle-efficient algorithms. The crucial property of the query class we rely on to obtain oracle efficiency is *dual separability*, which requires both the query class and its dual class have separator sets (Definition 2). Informally, the dual of a query class $\mathcal{Q}$ is the query class $\mathcal{Q}_{\text{dual}}$ that results from swapping the role of the functions $q \in \mathcal{Q}$ and the data elements $x \in \mathcal{X}$. More formally:

Definition 10 (Dual class and dual separability)**.** *Fix a class of queries $\mathcal{Q}$. For every element x in $\mathcal{X}$, let $h_x \colon \mathcal{Q} \to \{0,1\}$ be defined such that $h_x(q) = q(x)$. The* dual class $\mathcal{Q}_{\text{dual}}$ *of $\mathcal{Q}$ is the set of all such functions defined by elements in $\mathcal{X}$:*

$$\mathcal{Q}_{\text{dual}} = \{h_x \mid x \in \mathcal{X}\}.$$

We say that the class $\mathcal{Q}$ is (m_1, m_2)-dually separable if there exists a separator set of size m_1 for $\mathcal{Q}$, and there exists a separator set of size m_2 for $\mathcal{Q}_{\text{dual}}$.

As we will show (see Appendix A), many widely studied query classes, including discrete halfspaces, conjunctions, disjunctions, and parities are dually separable, often with $m_1 = m_2 = d$ (in fact, many of these classes are *self-dual*, meaning $\mathcal{Q} = \mathcal{Q}_{\text{dual}}$). For any $q \in \mathcal{Q}$, define its negation $\neg q$ to be $\neg q(x) = 1 - q(x)$. Let $\neg\mathcal{Q} = \{\neg q \mid q \in \mathcal{Q}\}$ be the *negation of $\mathcal{Q}$*. It will simplify several aspects of our exposition to deal with classes that are closed under negation. For any class $\mathcal{Q}$,

define $\overline{\mathcal{Q}} = \mathcal{Q} \cup \neg\mathcal{Q}$ to be the closure of $\mathcal{Q}$ under negation. Note that whenever we have a weighted minimization oracle for $\mathcal{Q}$, we have one for $\neg\mathcal{Q}$ as well — simply by negating the weights. Further, if U is a separator set for $\mathcal{Q}$, it is also a separator set for $\neg\mathcal{Q}$. This implies that we also have oracle efficient learners for $\overline{\mathcal{Q}}$, since we can separately learn over $\mathcal{Q}$ and $\neg\mathcal{Q}$, and then privately take the minimum value query that results from the two procedures (using e.g. report-noisy-min [DR14]).

Before we give our algorithm and analysis, we state several consequences of our main theorem (that follow from instantiating it with different oracle-efficient learners).

Theorem 10. *Let $\mathcal{Q}$ be an (m_1, m_2)-dually separable query class. Then given access to a weighted minimization oracle $\mathcal{O}$ over the class $\mathcal{Q}_{\text{dual}}$ and a differentially private weighted minimization algorithm $\mathcal{O}_{\epsilon_0,\delta_0}$ for the class $\mathcal{Q}$ (with appropriately chosen privacy parameters ϵ_0 and δ_0), the algorithm* OracleQuery *is oracle-efficient, (ϵ, δ)-differentially private, and (α, β)-accurate with α depending on the instantiation of $\mathcal{O}_{\epsilon_0,\delta_0}$. If $\mathcal{O}_{\epsilon_0,\delta_0}$ is robustly differentially private, then so is* OracleQuery.

1. *If $\mathcal{O}_{\epsilon_0,\delta_0}$ is instantiated with the Gaussian RSPM algorithm, then*

$$\alpha \leq \tilde{O}\left(\frac{m_1^{3/2} m_2^{3/4} \sqrt{\log(m_1/\beta)\log|\mathcal{X}|\log(1/\delta)}}{n\epsilon}\right)^{1/2}$$

   In this case, OracleQuery *is oracle equivalent to a differentially private algorithm, but is not robustly differentially private.*

2. *If $\mathcal{O}_{\epsilon_0,\delta_0}$ is instantiated with the **PRSMA** algorithm (using the Laplace RSPM as $\mathcal{A}_{\mathcal{O}^*}$), then*

$$\alpha \leq \tilde{O}\left(\frac{(m_1^{4/3} + \log^{1/3}(|\mathcal{Q}|))m_2^{1/4}\log^{1/6}(|\mathcal{X}|)}{(n\varepsilon)^{1/3}}\right) \cdot \text{polylog}\left(\frac{1}{\beta - \delta}\right)$$

   as long as $\beta > \delta$. In this case, OracleQuery *is robustly differentially private.*

3. *If $\mathcal{O}_{\epsilon_0,\delta_0}$ is an (α_0, β_0)-accurate differentially private oracle with $\alpha_0 = O(\log(|\mathcal{Q}|/(\epsilon_0 n)))$, then*

$$\alpha \leq \tilde{O}\left(\frac{m_2^{3/4}\sqrt{\log|\mathcal{X}|\log(1/\delta)}\log(|\mathcal{Q}|/\beta)}{n\epsilon}\right)^{1/2}$$

   In this case, OracleQuery *is robustly differentially private.*

where the $\tilde{O}$ hides logarithmic factors in $\frac{1}{\delta}, \frac{1}{\beta}, m_1, m_2, n$ and $\log(|\mathcal{X}|)$.

A couple of remarks are in order.

Remark 2. *The first two bounds quoted in Theorem 10 result from plugging in constructions of oracle-efficient differentially private learners that we gave in Section 3. These constructions start with a* non-private *optimization oracle. The third bound quoted in Theorem 10 assumes the existence of a* differentially private *oracle with error bounds comparable to the (inefficient) exponential mechanism based learner of [KLN+11]. We don't know if such oracles can be constructed from non-private (exact) optimization oracles. But this bound is analgous to the bounds given in the non-private oracle-efficient learning literature. This literature gives constructions assuming the existence of perfect learning oracles, but in practice, these oracles are instantiated with heuristics like regression or support vector machines,*

which exactly optimize some convex surrogate loss function. This is often reasonable, because although these heuristics don't have strong worst-case guarantees, they often perform very well in practice. The same exercise makes sense for private problems: we can use a differentially private convex minimization algorithm to optimize a surrogate loss function (e.g. [CMS11, BST14]), and hope that it does a good job minimizing classification error in practice. It no longer makes sense to assume that the heuristic exactly solves the learning problem (since this is impossible subject to differential privacy) — instead, the analogous assumption is that it does as well as the best inefficient private learner.

Remark 3. *It is useful to compare the bounds we obtain to the best bounds that can be obtained with inefficient algorithms. To be concrete, consider the class of boolean conjunctions defined over the boolean hypercube $\mathcal{X} = \{0,1\}^d$ (see Appendix A), which are dually-separable with $m_1 = m_2 = d$. The best (inefficient) bounds for constructing synthetic data useful for conjunctions [HR10, GRU12] obtain error: $\alpha = O\left(\frac{\sqrt{\log|\mathcal{Q}|}(\log|\mathcal{X}|)^{1/4}}{\sqrt{\epsilon n}}\right)$. In the case of boolean conjunctions, $\log|\mathcal{X}| = \log|\mathcal{Q}| = d$, and so this bound becomes: $\alpha = O\left(\frac{d^{3/4}}{\sqrt{\epsilon n}}\right)$. In contrast, the three oracle efficient bounds given in Theorem 10, when instantiated for boolean conjunctions are:*

1. $\alpha = O\left(\frac{d^{11/8}}{\sqrt{\epsilon n}}\right)$,
2. $\alpha = O\left(\frac{d^{7/4}}{(\epsilon n)^{1/3}}\right)$, *and*
3. $\alpha = O\left(\frac{d^{9/8}}{\sqrt{\epsilon n}}\right)$

respectively. Therefore the costs in terms of error that we pay, in exchange for oracle efficiency are $d^{5/8}$, $\frac{d}{(\epsilon n)^{1/6}}$, and $d^{3/8}$ respectively.

We now give a brief overview of our construction before diving into the technical details.

Proof overview: We present our solution in three main steps.

1. We first revisit the formulation by [HRU13] that views the synthetic data generation problem as a zero-sum game between a *Data player* and a *Query player*. We leverage the fact that at any approximate equilibrium, the data player's mixed strategy (over $\mathcal{X}$) represents a good synthetic dataset S' with respect to $\mathcal{Q}$.
2. Using the seminal result of [FS96], we will compute the equilibrium for the zero-sum game by simulating *no-regret dynamics* between the two players: in rounds, the Data player plays according to an oracle-efficient online learning algorithm due to [SKS16], and the Query player best responds to the Data player by using a differentially private oracle efficient optimization algorithm. At the end of the dynamics, the average play of the Data player is an approximate minimax strategy for the game, and hence a good synthetic dataset.
3. We instantiate the private best response procedure of the Query player using different oracle-efficient methods, which we have derived in this paper, each of which gives different accuracy guarantees. Finally, we apply our result to several query classes of interest.

4.1 The Query Release Game

The query release game defined in [HRU13] involves a *Data player* and *Query player*. The data player has action set equal to the data universe $\mathcal{X}$ (or equivalently the dual class $\mathcal{Q}_{\text{dual}}$), while the query player has action set equal to the query class $\overline{\mathcal{Q}}$. Given a pair of actions $x \in \mathcal{X}$ and $q \in \overline{\mathcal{Q}}$, the payoff is defined to be:

$$A(x, q) = q(S) - q(x),$$

where S is the input private dataset. In the zero-sum game, the Data player will try minimize the payoff and the Query player will try to maximize the payoff. To play the game, each player chooses a *mixed strategy*, which is defined by a probability distribution over their action set. Let $\Delta(\mathcal{X})$ and $\Delta(\overline{\mathcal{Q}})$ denote the sets of *mixed strategies* of the Data player and Query player respectively. To simplify notation, we will write $A(\hat{S}, \cdot) = \mathbb{E}_{x \sim \hat{S}}[A(x, \cdot)]$ and $A(\cdot, W) = \mathbb{E}_{q \sim W}[A(\cdot, q)]$ for any $\hat{S} \in \Delta(\mathcal{X})$ and $W \in \Delta(\overline{\mathcal{Q}})$. By von Neumann's minimax theorem, there exists a value V such that

$$V = \min_{\hat{S} \in \Delta(\mathcal{X})} \max_{q \in \overline{\mathcal{Q}}} A(\hat{S}, W) = \max_{W \in \Delta(\overline{\mathcal{Q}})} \min_{x \in \mathcal{X}} A(x, W)$$

If both players are playing strategies that can guarantee a payoff value close to V, then we say that the pair of strategies form an approximate equilibrium.

Definition 11 (Approximate Equilibrium)**.** *For any $\alpha > 0$, a pair of strategies $\hat{S} \in \Delta(\mathcal{X})$ and $W \in \Delta(\overline{\mathcal{Q}})$ form an α-*approximate minimax equilibrium *if*

$$\max_{q \in \overline{\mathcal{Q}}} A(\hat{S}, q) \leq V + \alpha \qquad \text{and} \qquad \min_{x \in \mathcal{X}} A(W, x) \geq V - \alpha.$$

Hsu et al. [HRU13] show that the query release game has value $V = 0$ and that at any approximate equilibrium, the mixed strategy of the Data player provides accurate answers for all queries in $\overline{\mathcal{Q}}$.

Lemma 10 (Accuracy at equilibrium [HRU13])**.** *Let $(\hat{S}, W)$ be an α-approximate equilibrium of the query release game. Then for any $q \in \overline{\mathcal{Q}}$, $|q(S) - q(\hat{S})| \leq \alpha$.*

Therefore, the dataset represented by the distribution $\hat{S}$ (or that could be obtained by sampling from $\hat{S}$) is exactly the synthetic dataset we would like to compute, and hence the problem of privately computing synthetic data is reduced to the problem of differentially private equilibrium computation in the query release game.

4.2 Solving the Game with No-Regret Dynamics

To privately compute an approximate equilibrium of the game, we will simulate the following *no-regret dynamics* between the Data Player and the Query Player in rounds: In each round t, the Data player plays a distribution S^t according to a *no-regret* learning algorithm, and the Query player plays an approximate best-response to S^t. The following classical theorem of Freund and Schapire [FS96] (instantiated in our setting) shows that the average play of both players in this dynamic forms an approximate equilibrium.

Theorem 11 ([FS96])**.** *Let $S^1, S^2, \ldots, S^T \in \Delta(\mathcal{X})$ be a sequence of distributions played by the Data Player, and let $q^1, q^2, \ldots, q^T \in \overline{\mathcal{Q}}$ be the Query player's sequence of approximate best-responses against*

these distributions. Suppose that the regret *of the two players satisfy:*

$$\mathrm{Reg}_D(T) = \sum_{t=1}^{T} A(S^t, q^t) - \min_{x \in \mathcal{X}} \sum_{t=1}^{T} A(x, q^t) \leq \gamma_D T$$
$$\mathrm{Reg}_Q(T) = \max_{q \in \overline{\mathcal{Q}}} \sum_{t=1}^{T} A(S^t, q) - \sum_{t=1}^{T} A(S^t, q^t) \leq \gamma_Q T.$$

Let $\overline{S}$ be uniform mixture of the distributions $\{S^1, \ldots, S^T\}$ and $\overline{W}$ be the uniform distribution over $\{q^1, \ldots, q^T\}$. Then $(\overline{S}, \overline{W})$ is a $(\gamma_D + \gamma_Q)$-approximate minimax equilibrium of the game.

Now we will detail the no-regret algorithm for the Data player and the best-response method for the Query player, and provide the regret bounds γ_D and γ_Q.

No-Regret Algorithm for the Data Player. We start with the observation that the regret of the Data player is independent of the private data S, because

$$\mathrm{Reg}_D(T) = \sum_{t=1}^{T} \left(q^t(S) - q^t(S^t) \right) - \min_{x \in \mathcal{X}} \sum_{t=1}^{T} \left(q^t(S) - q^t(x) \right) = \sum_{t=1}^{T} \neg q^t(S^t) - \min_{x \in \mathcal{X}} \sum_{t=1}^{T} \neg q^t(x).$$

Therefore, it suffices to minimize regret with respect to the sequence of loss functions $\{\neg q^t(\cdot)\}$, while ignoring the private dataset S. We crucially rely on the fact that each q^t is computed by the Query player subject to differential privacy, and so the Data player's learning algorithm need not be differentially private: differential privacy for the overall procedure will follow from the post-processing guarantee of differential privacy. In particular, we will run an oracle-efficient algorithm Context-FTPL due to [SKS16], which is a variant of the "Follow-the-Perturbed-Leader" of [KV05] algorithm that performs perturbations using a separator set. We state its regret guarantee below. Because Context-FTPL need not be differentially private, it can be instantiated with an arbitrary heuristic oracle, that need not be either differentially private or certifiable.

Context-FTPL $(\mathcal{Q}_{\text{dual}}, \mu)$ Algorithm [SKS16]
Given: parameter μ, hypothesis class $\mathcal{Q}_{\text{dual}}$ (or equivalently $\mathcal{X}$), separator set $U \subset \mathcal{Q}$ for $\mathcal{Q}_{\text{dual}}$, weighted optimization oracle $\mathcal{O}$ for $\mathcal{Q}_{\text{dual}}$
Input: A sequence of queries $\{q^1, \ldots, q^T\}$ selected by the Query player.

1: **for** $t = 1 \ldots T$ **do**
2: Data player plays the distribution S^t such that each draw x generated as follows:
3: Draw a sequence (s, η_s), for $s \in U$, where $\eta_s \sim \mathrm{Lap}(\mu)$
4: Let $x = \mathrm{argmin}_{x \in \mathcal{X}} \sum_{\tau=1}^{t-1} h_x(\neg q^\tau) + \sum_{s \in S} \eta_s h_x(s)$ {Use non-private oracle $\mathcal{O}$}
5: **end for**

Theorem 12 (Follows from [SKS16])**.** *Suppose that U is a separator set for $\mathcal{Q}_{\text{dual}}$ of cardinality m_2. Then the Data player running Context-FTPL $(\mathcal{X}, \mu)$ with appropriately chosen μ has regret:*

$$\mathrm{Reg}_D(T) \leq O(m_2^{3/4} \sqrt{T \log |\mathcal{X}|})$$

Note that the algorithm Context-FTPL only provides sample access to each distribution S^t, but each draw from S^t can be computed using a single call to the oracle $\mathcal{O}$.

Approximate Best Response by the Query Player. At each round t, after the Data player chooses S^t, the Query player needs to approximately solve the following best-response problem:

$$\operatorname*{argmax}_{q\in\overline{\mathcal{Q}}} A(S^t, q) = \operatorname*{argmax}_{q\in\overline{\mathcal{Q}}} \Big(q(S) - q(S^t)\Big)$$

Unlike the problem faced by the Data player, this optimization problem directly depends on the private data S, so the best response needs to be computed privately. Since we only have sample access to the distribution S^t, the Query player will first draw N random examples from the distribution S^t, and we will the empirical distribution $\hat{S}^t$ over the sample as a proxy for S^t. Recall that $\overline{\mathcal{Q}} = \mathcal{Q} \bigcup \neg\mathcal{Q}$, so we will first approximately and differentially privately solve both of the following two problems separately:

$$\operatorname*{argmax}_{q\in\mathcal{Q}} \Big(q(S) - q(\hat{S}^t)\Big) \qquad \text{and} \qquad \operatorname*{argmax}_{q\in\neg\mathcal{Q}} \Big(q(S) - q(\hat{S}^t)\Big) \tag{6}$$

Note that the two problems are equivalent to the following problems respectively:

$$\operatorname*{argmin}_{q\in\mathcal{Q}} \Big(q(\hat{S}^t) - q(S)\Big) \qquad \text{and} \qquad \operatorname*{argmin}_{q\in\mathcal{Q}} \Big(q(S) - q(\hat{S}^t)\Big), \tag{7}$$

both of which are weighted optimization problems:

$$\operatorname*{argmin}_{q\in\mathcal{Q}} \frac{1}{n}\sum_{x_i\in S} w_i q(x_i) + \frac{1}{N}\sum_{x'_j\in\hat{S}^t} w'_j q(x'_j) \tag{8}$$

with weights w_i, w'_j taking values in $\{1, -1\}$. We will rely on a private weighted optimization algorithm $\mathcal{O}_{\epsilon_0,\delta_0}$ to compute two solutions q_1^t and q_2^t for the two problems in Equation (7) respectively. Finally, the Query player privately selects one of the queries using *report noisy max*—i.e. it first perturb the values of $A(\hat{S}^t, q_1^t)$ and $A(\hat{S}^t, q_2^t)$ with Laplace noise, and then select the query with higher noisy value. By bounding the errors from the sampling of $\hat{S}^t$, the private optimization oracle $\mathcal{O}_{\epsilon_0,\delta_0}$, and report noisy max, we can derive the following regret guarantee for the Query player.

Lemma 11. *Suppose that the oracle $\mathcal{O}_{\epsilon_0,\delta_0}$ succeeds in solving all problems it is presented with up to error at most α_0 except with probability β_0. Then with probability at least $1 - 3\beta_0 T$, the Query player has regret:*

$$\mathrm{Reg}_Q(T) \le T\, O\left(\alpha_0 + \frac{\log(1/\beta_0)}{n\epsilon_0}\right).$$

Proof. There are three potential sources of error at each round t. The first is the error introduced by solving our optimization problem over the proxy distribution $\hat{S}^t$ instead of S^t. By applying a

Private Best-Response (PBR)

Given: privacy parameters (ϵ_0, δ_0), accuracy parameters (α_0, β_0), a private weighted optimization algorithm $\mathcal{O}_{\epsilon_0,\delta_0}$.

Input: A private dataset S and the Data player's sequence of distributions $\{S^1, \ldots, S^T\}$.

1: **for** $t = 1 \ldots T$ **do**
2: Query player plays a query q^t as follows:
3: Draw N samples $x'_1, \ldots x'_N \sim_{i.i.d.} S^t$ with $N = \frac{2\log(2|\mathcal{Q}|/\beta_0)}{\alpha_0^2}$
4: Form the weighted dataset $WD^1 = \{(x_i, \frac{1}{n})\}_{x_i \in S} \cup \{x_j, \frac{-1}{N}\}_{j=1\ldots N}$
5: Form the weighted dataset $WD^2 = \{(x_i, \frac{-1}{n})\}_{x_i \in S} \cup \{x_j, \frac{1}{N}\}_{j=1\ldots N}$
6: Let $q_1^t = \mathcal{O}_{\epsilon_0,\delta_0}(WD^1)$ and $q_2^t = \neg\left(\mathcal{O}_{\epsilon_0,\delta_0}(WD^2)\right)$
7: Perturb payoffs: $\tilde{A}_1 = A(\hat{S}^t, q_1^t) + \mathrm{Lap}(1/(\epsilon_0 n))$ and $\tilde{A}_2 = A(\hat{S}^t, q_2^t) + \mathrm{Lap}(1/(\epsilon_0 n))$
8: **If** $\tilde{A}_1 > \tilde{A}_2$ **then** $q^t = q_1^t$ **else** $q^t = q_2^t$
9: **end for**

Chernoff bound and a union bound over all queries in $\mathcal{Q}$, we have with probability $1 - \beta_0$ that,

$$\text{for all } q \in \overline{\mathcal{Q}}, \qquad \left|q(S^t) - q(\hat{S}^t)\right| \leq \sqrt{\frac{2\log(2|\mathcal{Q}|/\beta_0)}{N}} \leq \alpha_0. \tag{9}$$

Next is the error introduced by the oracle. By our assumption on the oracle $\mathcal{O}_{\epsilon_0,\delta_0}$, we have except with probability $2\beta_0$ that

$$\left(q_1^t(S) - q_1^t(\hat{S}^t)\right) \geq \max_{q \in \mathcal{Q}}\left(q(S) - q(\hat{S}^t)\right) - \alpha_0 \qquad \text{and} \qquad \left(q_2^t(S) - q_2^t(\hat{S}^t)\right) \geq \max_{q \in \neg\mathcal{Q}}\left(q(S) - q(\hat{S}^t)\right) - \alpha_0$$

The two inequalities together imply that

$$\max_{q \in \{q_1^t, q_2^t\}}\left(q(S) - q(\hat{S}^t)\right) \geq \max_{q \in \neg\mathcal{Q}}\left(q(S) - q(\hat{S}^t)\right) - \alpha_0 \tag{10}$$

Finally, the Laplace noise used to privately select the best query amongst q_1^t and q_2^t introduces additional error. But by the accuracy guarantee of report noisy max [DR14] (which follows from the CDF of the Laplace distribution and a union bound over two samples from it) we know that with probability $1 - \beta_0$,

$$\left(q^t(S) - q^t(\hat{S}^t)\right) \geq \max_{q \in \{q_1^t, q_2^t\}}\left(q(S) - q(\hat{S}^t)\right) - \frac{2\log(2/\beta_0)}{n\epsilon_0} \tag{11}$$

Combining Equations (9) to (11) and applying a union bound, we have the following per-round guarantee: except with probability $3\beta_0$,

$$\left(q^t(S) - q^t(S^t)\right) \geq \max_{q \in \overline{\mathcal{Q}}}\left(q(S) - q(S^t)\right) - O\left(\alpha_0 + \frac{\log(1/\beta_0)}{n\epsilon_0}\right).$$

Finally, taking a union bound over all T steps recovers the stated regret bound. □

4.3 The Full Algorithm: OracleQuery

Our main algorithm OracleQuery first simulates the no-regret dynamics described above, and then constructs a synthetic dataset from the average distribution $\overline{S} = \frac{1}{T}\sum_{t\in[T]} S^t$ played by the Data player. Since we only have sampling access to each S^t, we will approximate $\overline{S}$ by the empirical distribution of a set of independent samples drawn from $\overline{S}$. As we show below, the sampling error will be on the same order as the regret as long as we take roughly $\log|\mathcal{Q}|/\alpha^2$ samples.

Lemma 12. *Suppose that $\overline{S}$ is an η-approximate minimax strategy for the query release game. Let $\{x'_1,\ldots,x'_N\}$ be a set of $N = \frac{2\log(2|\mathcal{Q}|/\beta)}{\alpha_0^2}$ samples drawn i.i.d. from $\overline{S}$, and $\hat{S}$ be the empirical distributions over the drawn samples. Then with probability $1-\beta$, $\hat{S}$ is an $(\eta+\alpha_0)$-approximate minimax strategy.*

Proof. By the definition of an η-approximate minimax strategy, we have

$$\max_{q\in\overline{\mathcal{Q}}} A(\overline{S},q) \le V + \eta = \eta.$$

By applying the Chernoff bound, we know that except with probability $\beta/|\mathcal{Q}|$, the following holds for each $q\in\mathcal{Q}$:

$$\left|A(\overline{S},q) - A(\hat{S},q)\right| \le \sqrt{\frac{2\log(2|\mathcal{Q}|/\beta)}{N}} = \alpha$$

Note that this implies $\left|A(\overline{S},\neg q) - A(\hat{S},\neg q)\right| \le \alpha$ as well. Then by taking a union bound over $\mathcal{Q}$, we know that $\left|A(\overline{S},q) - A(\hat{S},q)\right| \le \alpha$ holds for all $q\in\overline{\mathcal{Q}}$ with probability at least $1-\beta$. It follows that

$$\max_{q\in\overline{\mathcal{Q}}} A(\hat{S},q) \le \max_{q\in\overline{\mathcal{Q}}} A(\overline{S},q) + \alpha_0 \le \eta + \alpha_0,$$

which recovers the stated bound. □

The details of the algorithm are presented in Algorithm 1. To analyze the algorithm, we will start with establishing its privacy guarantee, which directly follows from the advanced composition of [DRV10], the fact that each call to PBR by the Query player satisfies $(3\varepsilon_0, 2\delta_0)$-differential privacy (with ϵ_0 and δ_0 set according to Algorithm 1), and the fact that the rest of the algorithm can be viewed as a post-processing of these calls.

Lemma 13 (Privacy of OracleQuery)**.** OracleQuery *is oracle equivalent to an (ϵ,δ)-differentially private algorithm. If $\mathcal{O}_{\epsilon_0,\delta_0}$ is robustly differentially private, then* OracleQuery *is (ϵ,δ)-robustly differentially private.*

Now to analyze the accuracy of OracleQuery, we will show that the average distribution $\overline{S}$ is part of an approximate minimax equilibrium, and so is its approximation $\hat{S}$.

Lemma 14 (Accuracy of OracleQuery)**.** *Suppose that $\mathcal{O}_{\epsilon_0,\delta_0}$ is a weighted (ϵ_0,δ_0)-differentially private (α_0,β_0) minimization oracle over the class $\mathcal{Q}$, where the parameters ϵ_0, δ_0, and β_0 are set according to Algorithm 1. Then* OracleQuery *is an (α,β)-accurate synthetic data generation algorithm for $\mathcal{Q}$ with*

$$\alpha \le O\left(\alpha_0 + m_2^{3/4}\sqrt{\frac{\log(|\mathcal{X}|)}{T}} + \frac{\log(1/\beta_0)}{n\epsilon_0}\right)$$

Algorithm 1 Oracle-Efficient Synthetic Data Release: OracleQuery

Given: Target privacy parameters $\varepsilon, \delta \in (0,1)$, a target failure probability β, a number of rounds T, accuracy parameters α_0, β_0, a weighted optimization oracle $\mathcal{O}$ for the class $\mathcal{Q}_{\text{dual}}$, a (ϵ_0, δ_0)-differentially private (α_0, β_0)-accurate minimization oracle $\mathcal{O}_{\epsilon_0,\delta_0}$ for class $\mathcal{Q}$ with parameters that satisfy

$$\epsilon_0 = \frac{\varepsilon}{\sqrt{24T\ln(2/\delta)}}, \qquad \delta_0 \leq \frac{\delta}{4T}, \qquad \beta_0 = \frac{\beta}{4T}$$

Input: A dataset $S \in \mathcal{X}^n$.

1: Initialize $q^0 \in \overline{\mathcal{Q}}$ to be an arbitrary query
2: Let $N_{\alpha_0} = \frac{2\log(8|\mathcal{Q}|/\beta)}{\alpha_0^2}$
3: **for** $t = 1 \dots T$ **do**
4: Let S^t be a distribution defined by the sampling algorithm Context-FTPL $(\mathcal{Q}_{\text{dual}}, \mathcal{O}, \{q^0, \dots, q^{(t-1)}\})$ {Data player's no-regret algorithm}
5: Let $q^t = \text{PBR}(\varepsilon_0, \delta_0, \alpha_0, \beta_0, S, \mathcal{O}_{\epsilon_0,\delta_0}, S^t)$ {Query player's best response}
6: **end for**
7: **for** $j = 1, \dots, N_{\alpha_0}$: **do**
8: Draw τ from Unif($[T]$) and then draw x'_j from distribution S^τ
9: **end for**
Output: the dataset $\hat{S} = \{x'_1, \dots, x'_{N_{\alpha_0}}\}$

Proof. First, we will show that the average distribution $\overline{S}$ from the no-regret dynamics is an η-approximate minimax strategy, with

$$\eta \leq O\left(\alpha_0 + m_2^{3/4}\sqrt{\frac{\log(|\mathcal{X}|)}{T}} + \frac{\log(1/\beta_0)}{n\epsilon_0}\right).$$

Recall that the average regret for the two players is bounded by

$$\text{Reg}_D(T)/T \leq O(m_2^{3/4}\sqrt{\log|\mathcal{X}|/T}) \qquad \text{and,} \qquad \text{Reg}_Q(T)/T \leq O\left(\alpha_0 + \frac{\log(1/\beta_0)}{n\epsilon_0}\right),$$

with probability at least $1 - 3T\beta_0$. Then by Theorem 11, we know that $\overline{S}$ is an η-approximate minimax strategy, with probability at least $1 - 3\beta/4$. Let us condition on this event. Lastly, by the setting of N_{α_0} in Algorithm 1 and Lemma 12, we know that $\hat{S}$ is a $(\eta + \alpha_0)$, except with probability $\beta/4$. Then the stated bound follows directly from a union bound. □

Finally, we will consider three different instantiations of $\mathcal{O}_{\epsilon_0,\delta_0}$. To optimize the error guarantee for each instantiation, we set the number of rounds T used in Algorithm 1 so that the regret of the Data player given by Theorem 12 is on the same order as the error of $\mathcal{O}_{\epsilon_0,\delta_0}$. We first consider a differentially private oracle that matches the error guarantees of the generic private learner from [KLN+11].

Corollary 1. *Suppose that $\mathcal{O}_{\epsilon_0,0}$ is an $(\epsilon_0, 0)$-differentially private (α_0, β_0)-accurate weighted minimiza-*

tion oracle, where α_0, β_0 *are such that* $\alpha_0 \leq O\left(\frac{\log(|\mathcal{Q}|/\beta_0)}{n\epsilon_0}\right)$. *Then* OracleQuery *with* $T = \left\lceil \frac{n\epsilon m_2^{3/4}\sqrt{\log(|\mathcal{X}|)}}{\log(|\mathcal{Q}|/\beta)\sqrt{\log(1/\delta)}} \right\rceil$ *is an* (α, β)*-accurate synthetic data generation algorithm for* $\mathcal{Q}$ *with*

$$\alpha \leq \tilde{O}\left(\frac{m_2^{3/4}\sqrt{\log|\mathcal{X}|\log(1/\delta)}\log(|\mathcal{Q}|/\beta)}{n\epsilon}\right)^{1/2}$$

where the $\tilde{O}$ *hides logarithmic factors in* m_2, n *and* $\log(|\mathcal{X}|)$.

Next, we will instantiate $\mathcal{O}_{\epsilon_0,\delta_0}$ with the RSPM algorithm. Note that with this choice, although OracleQuery is oracle equivalent to a differentially private algorithm, it is not robustly differentially private.

Corollary 2. *When* $\mathcal{O}_{\epsilon_0,\delta_0}$ *is instantiated with the Gaussian RSPM algorithm,* OracleQuery *with* $T = \left\lceil \frac{m_2^{3/4}\sqrt{\log(|\mathcal{X}|)}n\epsilon}{m_1^{3/2}\sqrt{\log(m_1/\beta)}\ln(1/\delta)} \right\rceil$ *is an* (α, β)*-accurate synthetic data generation algorithm for* $\mathcal{Q}$ *with*

$$\alpha \leq \tilde{O}\left(\frac{m_1^{3/2}m_2^{3/4}\sqrt{\log(m_1/\beta)\log|\mathcal{X}|}\log(1/\delta)}{n\epsilon}\right)^{1/2}$$

where the $\tilde{O}$ *hides logarithmic factors in* m_2, n *and* $\log(|\mathcal{X}|)$.

Finally, we instantiate the oracle $\mathcal{O}_{\epsilon_0,\delta_0}$ with the **PRSMA** algorithm that uses the Laplace RSPM algorithm with a certifiable heuristic oracle.

Corollary 3. *When* $\mathcal{O}_{\epsilon_0,\delta_0}$ *is instantiated with PRSMA algorithm (that internally uses Laplace RSPM as* $\mathcal{A}_{\mathcal{O}^*}$*), then for any* $\beta > \delta$, OracleQuery *with* $T = \left\lceil \left(\frac{m_2^{3/4}\sqrt{\log(|\mathcal{X}|)n\epsilon}}{m_1^2+\sqrt{\log(|\mathcal{Q}|)}}\right)^{4/3} \right\rceil$ *is an* (α, β)*-accurate synthetic data generation algorithm for* $\mathcal{Q}$ *with*

$$\alpha \leq O\left(\frac{m_2^{1/4}\log^{1/6}|\mathcal{X}|\left(m_1^{4/3}+\log^{1/3}(|\mathcal{Q}|)\right)}{(n\varepsilon)^{1/3}}\right)\text{polylog}\left(m_1 1/\delta, 1/\epsilon, \frac{1}{\beta-\delta}\right).$$

4.4 Example: Conjunctions

Here, we instantiate our bounds for a particular query class of interest: boolean conjunctions over the hypercube $\{0,1\}^d$. Constructing synthetic data for conjunctions (or equivalently producing a full *marginal table* for a dataset) has long been a challenge problem in differential privacy, subject to a long line of work [BCD+07, UV10, GHRU13, KRSU10, TUV12, HRS12, FK14, CTUW14], and is known to be computationally hard even for the special case of *two-way* conjunctions [UV10]. Our results in particular imply the first oracle efficient algorithm for generating synthetic data for all 2^d conjunctions. Other interesting classes satisfy all of the conditions needed for our synthetic data generation algorithm to apply, including disjunctions, parities, and discrete halfspaces — see Appendix A for details, and for how to allow negated variables in the class of conjunctions while preserving seperability.

Definition 12. *Given a subset of variables $S \subseteq [d]$, the boolean conjunction defined by S is the statistical query $q_S(x) = \wedge_{j \in S} x_j$. The set of boolean conjunctions $\mathcal{Q}_C$ defined over the hypercube $\mathcal{X} = \{0,1\}^d$ is:*

$$\mathcal{Q}_C = \{q_S \mid S \subseteq [d]\}$$

Boolean conjunctions are (d, d) dually-separable (see Appendix A for the separator set).

Thus, we can instantiate Theorem 10 with (e.g.) the Gaussian RSPM algorithm, and obtain an oracle-efficient algorithm for generating synthetic data for all 2^d conjunctions that outputs a synthetic dataset S' that satisfies:

$$\max_{q \in \mathcal{Q}_C} |q(S) - q(S')| \leq \tilde{O}\left(\frac{d^{11/8}}{\sqrt{\epsilon n}}\right)$$

5 A Barrier

In this paper, we give *oracle-efficient* private algorithms for learning and synthetic data generation for classes of queries $\mathcal{Q}$ that exhibit special structure: small universal identification sets. Because of information theoretic lower bounds for differentially private learning [BNSV15, ALMM18], we know that these results cannot be extended to *all* learnable classes of queries $\mathcal{Q}$. But can they be extended to all classes of queries that are information theoretically learnable subject to differential privacy? Maybe — this is the most interesting question left open by our work. But here, we present a "barrier" illustrating a difficulty that one would have to overcome in trying to prove this result. Our argument has three parts:

1. First, we observe a folklore connection between differentially private learning and online learning: any differentially private empirical risk minimization algorithm $\mathcal{A}$ for a class $\mathcal{Q}$ that *always outputs the exact minimizer of a data-independent perturbation of the empirical risks* can also be used as a no-regret learning algorithm, using the "follow the perturbed leader" analysis of Kalai and Vempala [KV05]. The per-round run-time of this algorithm is exactly equal to the run-time of $\mathcal{A}$.

2. Oracle-efficient no-regret learning algorithms are subject to a lower bound of Hazan and Koren [HK16], that states that even given access to an oracle which solves optimization problems over a set of experts $\mathcal{Q}$ in unit time, there exist finite classes $\mathcal{Q}$ such that obtaining non-trivial regret guarantees requires total running time larger than poly($|\mathcal{Q}|$). This implies a lower bound on the magnitude of the perturbations that an algorithm of the type described in (1) must use.

3. Finally, we observe for any finite class of hypotheses $\mathcal{Q}$, information theoretically, it is possible to solve the empirical risk minimization problem on a dataset of size T up to error $O(\frac{\log|\mathcal{Q}|}{\epsilon T})$ using the generic learner from [KLN+11]. This implies a separation between the kinds of algorithms described in 1), and the (non-efficiently) achievable information theoretic bounds consistent with differential privacy.

We emphasize that oracle efficient algorithms for learning over $\mathcal{Q}$ have access to a non-private oracle which exactly solves the learning problem over $\mathcal{Q}$ — not an NP oracle, for which the situation is different (see the discussion in Section 6).

First we define the class of mechanisms our barrier result applies to:

Definition 13. *We say that an (ϵ, δ)-differentially private learning algorithm $\mathcal{A} : \mathcal{X}^n \to \mathcal{Q}$ for $\mathcal{Q}$ is a perturbed Empirical Risk Minimizer (pERM) if there is some distribution $\mathcal{D}_{\epsilon,\delta}$ (defined independently of the data S) over perturbations $Z \in \mathbb{R}^{|\mathcal{Q}|}$ such that on input $S \in \mathcal{X}^n$, $\mathcal{A}$ outputs:*

$$\mathcal{A}(S) = \arg\min_{q \in \mathcal{Q}} \Big(n \cdot q(S) + Z_q \Big)$$

where $Z \sim \mathcal{D}_{\epsilon,\delta}$.

We note that many algorithms are pERM algorithms. The most obvious example is *report-noisy-min*, in which each $Z_q \sim \mathrm{Lap}(1/\epsilon)$ independently. The exponential mechanism instantiated with empirical loss as its quality score (i.e. the generic learner of [KLN+11]) is also a pERM algorithm, in which each Z_q is drawn independently from a Gumbel distribution [DR14]. But note that the coordinates Z_q need not be drawn independently: The oracle-efficient RSPM algorithm we give in Section 3 is also a pERM algorithm, in which the perturbations Z_q are introduced in an implicit (correlated) way by perturbing the dataset itself. And it is natural to imagine that many algorithms that employ weighted optimization oracles — which after all solve an exact minimization problem — will fall into this class. The expected error guarantees of these algorithms are proven by bounding $\mathbb{E}[\|Z\|_\infty]$, which is typically a tight bound.

We now briefly recall the online learning setting. Let $\mathcal{Q}$ be an arbitrary class of functions $q : \mathcal{X} \to [0,1]$. In rounds $t = 1, \ldots, T$, the learner selects a function $q^t \in \mathcal{Q}$, and an (adaptive) adversary selects an example $x^t \in \mathcal{X}$, as a function of the sequence $(q^1, x^1, \ldots, q^{t-1}, x^{t-1})$. The learner incurs a loss of $\ell^t = q^t(x^t)$. A standard objective is to minimize the *expected average regret*:

$$R(T) = \mathbb{E}\left[\frac{1}{T}\sum_{t=1}^{T} q^t(x^t)\right] - \min_{q \in \mathcal{Q}} \frac{1}{T}\sum_{t=1}^{T} q(x^t)$$

where the expectation is taken over the randomness of the learner. A weighted optimization oracle in the online learning setting is exactly the same thing as it is in our setting: Given a weighted dataset (S, w), it returns $\arg\min_{q \in \mathcal{Q}} \sum_{x_i \in S} w_i \cdot q(x_i)$.

A natural way to try to use a private learning algorithm in the online learning setting is just to run it at each round t on the dataset defined on the set of data points observed so far: $S_t = \{x^1, \ldots, x^{t-1}\}$.

Definition 14. *Follow the Private Leader, instantiated with $\mathcal{A}$, is the online learning algorithm that at every round t selects $q^t = \mathcal{A}(S_t)$.*

The follow the private leader algorithm instantiated with $\mathcal{A}$ has a controllable regret bound whenever $\mathcal{A}$ is a differentially private pERM algorithm. The following theorem is folklore, but follows essentially from the original analysis of "follow the perturbed leader" by Kalai and Vempala [KV05]. See e.g. the lecture notes from [RS17] or [ALMT17] for an example of this analysis cast in the language of differential privacy. We include a proof in Appendix D for completeness.

Theorem 13. *Let $\varepsilon, \delta \in (0,1)$ and let $\mathcal{A}$ be an (ϵ, δ) differentially private pERM algorithm for query class $\mathcal{Q}$, with perturbation distribution $\mathcal{D}_{(\varepsilon,\delta)}$. Then Follow the Private Leader instantiated with $\mathcal{A}$ has expected regret bounded by:*

$$R(T) \leq O\left(\epsilon + \delta + \frac{\mathbb{E}_{Z \sim \mathcal{D}_{\varepsilon,\delta}}[\|Z\|_\infty]}{T}\right)$$

Note that the regret is controlled by $\mathbb{E}_{Z\sim\mathcal{D}_{\varepsilon,\delta}}[\|Z\|_\infty]$, which also controls the error of $\mathcal{A}$ as a learning algorithm.

We wish to exploit a lower bound on the running time of oracle efficient online learners over arbitrary sets $\mathcal{Q}$ due to Hazan and Koren [HK16]:

Theorem 14 ([HK16])**.** *For every algorithm with access to a weighted optimization oracle $\mathcal{O}$, there exists a class of functions $\mathcal{Q}$ such that the algorithm cannot guarantee that its expected average regret will be smaller than* 1/16 *in total time less than $O\left(\sqrt{|\mathcal{Q}|}/\log^3(|\mathcal{Q}|)\right)$.*

Here, it is assumed that calls to the oracle $\mathcal{O}$ can be carried out in unit time, and *total* time refers to the cumulative time over all T rounds of interaction. Hence, if $\mathcal{A}$ is oracle-efficient — i.e. it runs in time $f(t) = \text{poly}(t, \log|\mathcal{Q}|)$ when given as input a dataset of size t, the *total* run time of follow the private leader instantiated with $\mathcal{A}$ is: $\sum_{t=1}^{T} f(t) \leq T \cdot f(T) = \text{poly}(T, \log|\mathcal{Q}|)$.

This theorem is almost what we want — except that the order of quantifiers is reversed. It in principle leaves open the possibility that for every class $\mathcal{Q}$, there is a different oracle efficient algorithm (tailored to the class) that can efficiently obtain low regret. After all, our RSPM algorithm is non-uniform in this way — for each new class of functions $\mathcal{Q}$, it must be instantiated with a separator set for that class.

Via a min-max argument together with an equilibrium sparsification technique, we can give a version of the lower bound of [HK16] that has the order of quantifiers we want — see Appendix D for the proof.

Theorem 15. *For any d, there is a fixed finite class of statistical queries $\mathcal{Q}$ of size $|\mathcal{Q}| = N = 2^d$ defined over a data universe of size $|\mathcal{X}| = O(N^5 \log^2 N)$ such that for every online learning algorithm with access to a weighted optimization oracle for $\mathcal{Q}$, it cannot guarantee that its expected average regret will be $o(1)$ in total time less than $\Omega(\sqrt{N}/\log^3(N))$.*

Theorem 15 therefore implies that follow the private leader, when instantiated with any oracle-efficient differentially private pERM algorithm $\mathcal{A}$ cannot obtain diminishing regret $R(T) = o(1)$ unless the number of rounds $T = \Omega(|Q|^c)$ for some $c > 0$. In combination with Theorem 13, this implies our barrier result:

Theorem 16. *Any oracle efficient (i.e. running in time $\text{poly}(n, \log|\mathcal{Q}|)$) (ϵ, δ)-differentially private pERM algorithm instantiated with a weighted optimization oracle for the query class $\mathcal{Q}$ defined in Theorem 15, with perturbation distribution $\mathcal{D}_{(\varepsilon,\delta)}$ must be such that for every $(\epsilon + \delta) = o(1)$:*

$$\mathbb{E}_{Z\sim\mathcal{D}_{(\varepsilon,\delta)}}[\|Z\|_\infty] \geq \Omega(|\mathcal{Q}|^c)$$

for some constant $c > 0$.

If the accuracy guarantee of $\mathcal{A}$ is proportional to $\mathbb{E}_{Z\sim\mathcal{D}_{(\varepsilon,\delta)}}[\|Z\|_\infty]$ (as it is for all pERM algorithms that we know of), this means that there exist finite classes of statistical queries $\mathcal{Q}$ such that no oracle-efficient algorithm can obtain non-trivial error unless the dataset size $n \geq \text{poly}(|\mathcal{Q}|)$. Of course, if $n \geq \text{poly}(|\mathcal{Q}|)$, then algorithms such as report-noisy-min and the exponential mechanism can be run in polynomial time.

This is in contrast with what we can obtain via the generic (inefficient) private learner of [KLN+11], which obtains expected error $O\left(\frac{\log|Q|}{\epsilon n}\right)$, which is non-trivial whenever $n = \Omega\left(\frac{\log|\mathcal{Q}|}{\epsilon}\right)$. Similarly, because we show in Theorem 15 that the hard class $\mathcal{Q}$ can be taken to have universe size

$\mathcal{X} = \text{poly}(\mathcal{Q})$, this means that information theoretically, it is even possible to privately solve the (harder) problem of α-accurate synthetic data for $\mathcal{Q}$ for $\alpha = O\left(\left(\frac{\log^2|\mathcal{Q}|}{\epsilon n}\right)^{1/3}\right)$ using the (inefficient) synthetic data generation algorithm of [BLR13]. This is non-trivial whenever $n = \Omega\left(\frac{\log^2|\mathcal{Q}|}{\epsilon}\right)$. In contrast, our barrier result states is that *if* there exists an oracle-efficient learner $\mathcal{A}$ for this class $\mathcal{Q}$ that has polynomially related sample complexity to what is obtainable absent a guarantee of oracle efficiency, then $\mathcal{A}$ must either:

1. Not be a pERM algorithm, or:
2. Have expected error that is $O\left(\frac{\text{poly}(\log \mathbb{E}_{Z\sim\mathcal{D}_{(\varepsilon,\delta)}}[\|Z\|_\infty])}{n}\right)$.

Condition 2. seems especially implausible, as for every pERM we are aware of, $\mathbb{E}_{Z\sim\mathcal{D}_{(\varepsilon,\delta)}}[\|Z\|_\infty]$ is a tight bound (up to log factors) on its expected error. In particular, this barrier implies that there is no oracle efficient algorithm for *sampling* from the exponential mechanism distribution used in the generic learner of [KLN+11] for arbitrary query classes $\mathcal{Q}$.

6 Conclusion and Open Questions

In this paper, we have initiated the systematic study of the power of *oracle-efficient* differentially private algorithms, and have made the distinction between oracle-dependent non-robust differential privacy and robust differential privacy. This is a new direction that suggests a number of fascinating open questions. In our opinion, the most interesting of these is:

> "Can every learning and synthetic data generation problem that is solvable subject to differential privacy be solved with an oracle-efficient (robustly) differentially private algorithm, with only a polynomial blow-up in sample complexity?"

It remains an open question whether or not finite Littlestone dimension characterizes private learnability (it is known that infinite Littlestone dimension precludes private learnability [ALMM18]) — and so one avenue towards resolving both open questions in the affirmative simultaneously would be to show that finite Littlestone dimension can be leveraged to obtain oracle-efficient differentially private learning algorithms.

However, because of our barrier result, we conjecture that the set of query classes that are privately learnable in an oracle-efficient manner is a *strict subset* of the set that are privately learnable. If this is so, can we precisely characterize this set? What is the right structural property, and is it more general than the sufficient condition of having small universal identification sets that we have discovered?

Even restricting attention to query classes with universal identification sets of size m, there are interesting quantitative questions. The Gaussian version of our RSPM algorithm efficiently obtains error that scales as $m^{3/2}$, but information-theoretically, it is possible to obtain error scaling only linearly with m. Is this optimal error rate possible to obtain in an oracle-efficient manner, or is the $\sqrt{m}$ error overhead that comes with our approach necessary for oracle efficiency?

Our **PRSMA** algorithm shows how to generically reduce from an oracle-dependent guarantee of differential privacy to a guarantee of robust differential privacy — *but at a cost*, both in terms of running time, and in terms of error. Are these costs necessary? Without further assumptions

on the construction of the oracle, it seems difficult to avoid the $O(1/\delta)$-overhead in running time, but perhaps there are natural assumptions that can be placed on the failure-mode of the oracle that can avoid this. It is less clear whether the error overhead that we introduce — by running the original algorithm on an ϵ fraction of the dataset, with a privacy parameter $\epsilon' \approx 1/\sqrt{\epsilon n}$ — is necessary. Doing this is a key feature of our algorithm and analysis, because we take advantage of the fact that differentially private algorithms are actually *distributionally private* when ϵ' is set this small — but perhaps it can be avoided entirely with a different approach.

Our barrier result takes advantage of a connection between differentially private learnability and online learnability. Because private pERM algorithms can be used efficiently as no-regret learning algorithms, they are subject to the lower bounds on oracle-efficient online learning proven in [HK16]. But perhaps the connection between differentially private learnability and online learnability runs deeper. Can *every* differentially private learning algorithm be used in a black box manner to efficiently obtain a no-regret learning algorithm? Note that it is already known that private learnability implies finite Littlestone dimension, so the open question here concerns whether there is an *efficient blackbox* reduction from private ERM algorithms to online learning algorithms. If true, this would convert our barrier for pERM algorithms into a full lower-bound for oracle-efficient private learning algorithms generally.

Finally, a more open ended question — that applies both to our work and to work on oracle efficiency in machine learning more generally — concerns how to refine the model of oracle efficiency. Ideally, the learning problems fed to the oracle should be "natural" — e.g. a small perturbation or re-weighting of the original (non-private) learning problem, as is the case for the algorithms we present in our paper. This is desirable because presumably we believe that the heuristics which can solve hard learning problems in practice work for "natural" instances, rather than arbitrary problems. However, the definition for oracle efficiency that we use in this paper allows for un-natural algorithms. For example, it is possible to show that the problem of sampling from the exponential mechanism of [MT07] defined by rational valued quality scores that are efficiently computable lies in $\mathbf{BPP}^{\mathbf{NP}}$ — in other words, the sampling can be done in polynomial time given access to an oracle for solving circuit-satisfiability problems[4]. This implies in particular, that there exists an oracle efficient algorithm (as we have defined them) for any NP hard learning problem — because the learning oracle can be used as an arbitrary **NP** oracle via gadget reductions[5]. The same logic implies that there are oracle efficient no-regret learning algorithms for any class of experts for which offline optimization is NP hard — because an NP oracle can be used to sample from the multiplicative weights distribution. But these kinds of gadget reductions seem to be an abuse of the model of oracle efficiency, which currently reduces to all of $\mathbf{BPP}^{\mathbf{NP}}$ when the given oracle is solving an NP hard problem[6]. Ambitiously, might there be a refinement of the

[4]This construction is due to Jonathan Ullman and Salil Vadhan (personal communication). It starts from the ability to sample uniformly at random amongst the set of satisfying assignments of an arbitrary polynomially sized boolean circuit given an NP oracle, using the algorithm of [BGP00]. For any distribution $\mathcal{P}$ such that there is a polynomially sized circuit C for which the relative probability mass on any discrete input x can be computed by $C(x)$, we can construct a boolean circuit C' that computes for bounded bit-length rational numbers w: $C'(x, w) = 1$ if $C(x) \geq w$. The marginal distribution on elements x when sampling uniformly at random from the satisfying assignments of this circuit is $\mathcal{P}$.

[5]Note that this procedure is not *robustly* differentially private, since sampling from the correct distribution occurs only if the oracle does not fail. But it could be fed into our **PRSMA** algorithm to obtain robust privacy. It also does not solve synthetic data generation oracle efficiently because the quality score used for synthetic data generation in [BLR13] is not computable by a polynomially sized circuit generally.

[6]This does not contradict the lower bound of [HK16] for oracle efficient online learning, or our barrier re-

model of oracle efficiency that requires one to prove a utility theorem along the following lines: assuming an oracle which can with high probability solve learning problems drawn from the actual data distribution, the oracle efficient algorithm will (with slightly lower probability) solve the private learning problem when the underlying instance is drawn from the same distribution. Theorems of this sort would be of great interest, and would (presumably) rule out "unnatural" algorithms relying on gadget reductions.

Acknowledgements We thank Michael Kearns, Adam Smith, Jon Ullman and Salil Vadhan for insightful conversations about this work.

References

[ABD+18] Alekh Agarwal, Alina Beygelzimer, Miroslav Dudík, John Langford, and Hanna M. Wallach. A reductions approach to fair classification. In Jennifer G. Dy and Andreas Krause, editors, *Proceedings of the 35th International Conference on Machine Learning, ICML 2018, Stockholmsmässan, Stockholm, Sweden, July 10-15, 2018*, volume 80 of *JMLR Workshop and Conference Proceedings*, pages 60–69. JMLR.org, 2018.

[AHK12] Sanjeev Arora, Elad Hazan, and Satyen Kale. The multiplicative weights update method: a meta-algorithm and applications. *Theory of Computing*, 8(1):121–164, 2012.

[AIK18] Daniel Alabi, Nicole Immorlica, and Adam Kalai. Unleashing linear optimizers for group-fair learning and optimization. In *Conference On Learning Theory*, pages 2043–2066, 2018.

[ALMM18] Noga Alon, Roi Livni, Maryanthe Malliaris, and Shay Moran. Private pac learning implies finite littlestone dimension. *arXiv preprint arXiv:1806.00949*, 2018.

[ALMT17] Jacob Abernethy, Chansoo Lee, Audra McMillan, and Ambuj Tewari. Online learning via differential privacy. *arXiv preprint arXiv:1711.10019*, 2017.

[BBB+08] Maria-Florina Balcan, Nikhil Bansal, Alina Beygelzimer, Don Coppersmith, John Langford, and Gregory B. Sorkin. Robust reductions from ranking to classification. *Machine Learning*, 72(1-2):139–153, 2008.

[BCD+07] Boaz Barak, Kamalika Chaudhuri, Cynthia Dwork, Satyen Kale, Frank McSherry, and Kunal Talwar. Privacy, accuracy, and consistency too: a holistic solution to contingency table release. In *Proceedings of the twenty-sixth ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems*, pages 273–282. ACM, 2007.

[BDH+05] Alina Beygelzimer, Varsha Dani, Thomas P. Hayes, John Langford, and Bianca Zadrozny. Error limiting reductions between classification tasks. In Luc De Raedt

sult/conjectured separation in the case of private learning algorithms. This is because oracles solving problems that don't have polynomial time algorithms, but *are not NP hard* cannot be used to encode the arbitrary circuit-SAT instances needed to implement an NP oracle.

and Stefan Wrobel, editors, *Machine Learning, Proceedings of the Twenty-Second International Conference (ICML 2005), Bonn, Germany, August 7-11, 2005*, volume 119 of *ACM International Conference Proceeding Series*, pages 49–56. ACM, 2005.

[BGP00] Mihir Bellare, Oded Goldreich, and Erez Petrank. Uniform generation of np-witnesses using an np-oracle. *Information and Computation*, 163(2):510–526, 2000.

[BILM16] Alina Beygelzimer, Hal Daumé III, John Langford, and Paul Mineiro. Learning reductions that really work. *Proceedings of the IEEE*, 104(1):136–147, 2016.

[BLR13] Avrim Blum, Katrina Ligett, and Aaron Roth. A learning theory approach to noninteractive database privacy. *Journal of the ACM (JACM)*, 60(2):12, 2013.

[BM13] R. Bardenet and O.-A. Maillard. Concentration inequalities for sampling without replacement. *ArXiv e-prints*, September 2013.

[BNSV15] Mark Bun, Kobbi Nissim, Uri Stemmer, and Salil Vadhan. Differentially private release and learning of threshold functions. In *IEEE 56th Annual Symposium on Foundations of Computer Science*, 2015.

[BST14] Raef Bassily, Adam Smith, and Abhradeep Thakurta. Private empirical risk minimization: Efficient algorithms and tight error bounds. In *Foundations of Computer Science (FOCS), 2014 IEEE 55th Annual Symposium on*, pages 464–473. IEEE, 2014.

[BTHKM15] Aharon Ben-Tal, Elad Hazan, Tomer Koren, and Shie Mannor. Oracle-based robust optimization via online learning. *Operations Research*, 63(3):628–638, 2015.

[BTT18] Raef Bassily, Om Thakkar, and Abhradeep Thakurta. Model-agnostic private learning via stability. *arXiv preprint arXiv:1803.05101*, 2018.

[CLN+16] Rachel Cummings, Katrina Ligett, Kobbi Nissim, Aaron Roth, and Zhiwei Steven Wu. Adaptive learning with robust generalization guarantees. In *Proceedings of the 29th Conference on Learning Theory, COLT 2016, New York, USA, June 23-26, 2016*, pages 772–814, 2016.

[CMS11] Kamalika Chaudhuri, Claire Monteleoni, and Anand D Sarwate. Differentially private empirical risk minimization. *Journal of Machine Learning Research*, 12(Mar):1069–1109, 2011.

[CTUW14] Karthekeyan Chandrasekaran, Justin Thaler, Jonathan Ullman, and Andrew Wan. Faster private release of marginals on small databases. In *Proceedings of the 5th conference on Innovations in theoretical computer science*, pages 387–402. ACM, 2014.

[DF18] Cynthia Dwork and Vitaly Feldman. Privacy-preserving prediction. *arXiv preprint arXiv:1803.10266*, 2018.

[DFH+15] Cynthia Dwork, Vitaly Feldman, Moritz Hardt, Toniann Pitassi, Omer Reingold, and Aaron Leon Roth. Preserving statistical validity in adaptive data analysis. In *Proceedings of the forty-seventh annual ACM symposium on Theory of computing*, pages 117–126. ACM, 2015.

[DHL+17] Miroslav Dudík, Nika Haghtalab, Haipeng Luo, Robert E Schapire, Vasilis Syrgkanis, and Jennifer Wortman Vaughan. Oracle-efficient online learning and auction design. In *Foundations of Computer Science (FOCS), 2017 IEEE 58th Annual Symposium on*, pages 528–539. IEEE, 2017.

[DKM+06] Cynthia Dwork, Krishnaram Kenthapadi, Frank McSherry, Ilya Mironov, and Moni Naor. Our data, ourselves: Privacy via distributed noise generation. In *Annual International Conference on the Theory and Applications of Cryptographic Techniques*, pages 486–503. Springer, 2006.

[DMNS06] Cynthia Dwork, Frank McSherry, Kobbi Nissim, and Adam Smith. Calibrating noise to sensitivity in private data analysis. In *Proceedings of the Third Conference on Theory of Cryptography*, TCC'06, pages 265–284, Berlin, Heidelberg, 2006. Springer-Verlag.

[DOSW11] Ilias Diakonikolas, Ryan O'Donnell, Rocco A Servedio, and Yi Wu. Hardness results for agnostically learning low-degree polynomial threshold functions. In *Proceedings of the twenty-second annual ACM-SIAM symposium on Discrete algorithms*, pages 1590–1606. Society for Industrial and Applied Mathematics, 2011.

[DR14] Cynthia Dwork and Aaron Roth. The algorithmic foundations of differential privacy. *Found. Trends Theor. Comput. Sci.*, 9(3–4):211–407, August 2014.

[DRV10] Cynthia Dwork, Guy N. Rothblum, and Salil Vadhan. Boosting and differential privacy. In *Proceedings of the 2010 IEEE 51st Annual Symposium on Foundations of Computer Science*, FOCS '10, pages 51–60, Washington, DC, USA, 2010. IEEE Computer Society.

[FGKP09] Vitaly Feldman, Parikshit Gopalan, Subhash Khot, and Ashok Kumar Ponnuswami. On agnostic learning of parities, monomials, and halfspaces. *SIAM Journal on Computing*, 39(2):606–645, 2009.

[FGRW12] Vitaly Feldman, Venkatesan Guruswami, Prasad Raghavendra, and Yi Wu. Agnostic learning of monomials by halfspaces is hard. *SIAM Journal on Computing*, 41(6):1558–1590, 2012.

[FK14] Vitaly Feldman and Pravesh Kothari. Learning coverage functions and private release of marginals. In *Conference on Learning Theory*, pages 679–702, 2014.

[FS96] Yoav Freund and Robert E. Schapire. Game theory, on-line prediction and boosting. In *Proceedings of the Ninth Annual Conference on Computational Learning Theory, COLT 1996, Desenzano del Garda, Italy, June 28-July 1, 1996.*, pages 325–332, 1996.

[GGAH+14] Marco Gaboardi, Emilio Jesús Gallego-Arias, Justin Hsu, Aaron Roth, and Zhiwei Steven Wu. Dual query: Practical private query release for high dimensional data. *CoRR*, abs/1402.1526, 2014.

[GHRU13] Anupam Gupta, Moritz Hardt, Aaron Roth, and Jonathan Ullman. Privately releasing conjunctions and the statistical query barrier. *SIAM Journal on Computing*, 42(4):1494–1520, 2013.

[GKS93] Sally A Goldman, Michael J Kearns, and Robert E Schapire. Exact identification of read-once formulas using fixed points of amplification functions. *SIAM Journal on Computing*, 22(4):705–726, 1993.

[GM18] Anna Gilbert and Audra McMillan. Property testing for differential privacy. *arXiv preprint arXiv:1806.06427*, 2018.

[GRU12] Anupam Gupta, Aaron Roth, and Jonathan Ullman. Iterative constructions and private data release. In *Theory of cryptography conference*, pages 339–356. Springer, 2012.

[HK16] Elad Hazan and Tomer Koren. The computational power of optimization in online learning. In *Proceedings of the 48th Annual ACM SIGACT Symposium on Theory of Computing, STOC 2016, Cambridge, MA, USA, June 18-21, 2016*, pages 128–141, 2016.

[HR10] Moritz Hardt and Guy N. Rothblum. A multiplicative weights mechanism for privacy-preserving data analysis. In *Proceedings of the 2010 IEEE 51st Annual Symposium on Foundations of Computer Science*, FOCS '10, pages 61–70, Washington, DC, USA, 2010. IEEE Computer Society.

[HRS12] Moritz Hardt, Guy N Rothblum, and Rocco A Servedio. Private data release via learning thresholds. In *Proceedings of the twenty-third annual ACM-SIAM symposium on Discrete Algorithms*, pages 168–187. Society for Industrial and Applied Mathematics, 2012.

[HRU13] Justin Hsu, Aaron Roth, and Jonathan Ullman. Differential privacy for the analyst via private equilibrium computation. In *Proceedings of the Forty-fifth Annual ACM Symposium on Theory of Computing*, STOC '13, pages 341–350, New York, NY, USA, 2013. ACM.

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

[KNRW18] Michael J. Kearns, Seth Neel, Aaron Roth, and Zhiwei Steven Wu. Preventing fairness gerrymandering: Auditing and learning for subgroup fairness. In Jennifer G. Dy and Andreas Krause, editors, *Proceedings of the 35th International Conference on Machine Learning, ICML 2018, Stockholmsmässan, Stockholm, Sweden, July 10-15, 2018*, volume 80 of *JMLR Workshop and Conference Proceedings*, pages 2569–2577. JMLR.org, 2018.

[KRSU10] Shiva Prasad Kasiviswanathan, Mark Rudelson, Adam Smith, and Jonathan Ullman. The price of privately releasing contingency tables and the spectra of random matrices with correlated rows. In *Proceedings of the forty-second ACM symposium on Theory of computing*, pages 775–784. ACM, 2010.

[KV94] Michael J Kearns and Umesh Vazirani. *An introduction to computational learning theory*. MIT press, 1994.

[KV05] Adam Kalai and Santosh Vempala. Efficient algorithms for online decision problems. *Journal of Computer and System Sciences*, 71(3):291–307, 2005.

[LM00] B. Laurent and P. Massart. Adaptive estimation of a quadratic functional by model selection. *Ann. Statist.*, 28(5):1302–1338, 10 2000.

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

[NRS07] Kobbi Nissim, Sofya Raskhodnikova, and Adam Smith. Smooth sensitivity and sampling in private data analysis. In *Proceedings of the thirty-ninth annual ACM symposium on Theory of computing*, pages 75–84. ACM, 2007.

[NTZ13] Aleksandar Nikolov, Kunal Talwar, and Li Zhang. The geometry of differential privacy: the sparse and approximate cases. In *Proceedings of the forty-fifth annual ACM symposium on Theory of computing*, pages 351–360. ACM, 2013.

[PAE+16] Nicolas Papernot, Martín Abadi, Ulfar Erlingsson, Ian Goodfellow, and Kunal Talwar. Semi-supervised knowledge transfer for deep learning from private training data. *arXiv preprint arXiv:1610.05755*, 2016.

[PP14] ROBIN PEMANTLE and YUVAL PERES. Concentration of lipschitz functionals of determinantal and other strong rayleigh measures. *Combinatorics, Probability and Computing*, 23(1):140?160, 2014.

[RR10] Aaron Roth and Tim Roughgarden. Interactive privacy via the median mechanism. In *Proceedings of the forty-second ACM symposium on Theory of computing*, pages 765–774. ACM, 2010.

[RS17] Aaron Roth and Adam Smith. Lecture 15: Algorithmic foundations of adaptive data analysis. *https://adaptivedataanalysis.files.wordpress.com/2017/11/lect15.pdf*, 2017.

[SKS16] Vasilis Syrgkanis, Akshay Krishnamurthy, and Robert E. Schapire. Efficient algorithms for adversarial contextual learning. *CoRR*, abs/1602.02454, 2016.

[TUV12] Justin Thaler, Jonathan Ullman, and Salil Vadhan. Faster algorithms for privately releasing marginals. In *International Colloquium on Automata, Languages, and Programming*, pages 810–821. Springer, 2012.

[Ull16] Jonathan Ullman. Answering n^2+o(1) counting queries with differential privacy is hard. *SIAM Journal on Computing*, 45(2):473–496, 2016.

[UR16] Berk Ustun and Cynthia Rudin. Supersparse linear integer models for optimized medical scoring systems. *Machine Learning*, 102(3):349–391, 2016.

[UV10] John Ullman and Salil P. Vadhan. Pcps and the hardness of generating private synthetic data. *IACR Theory of Cryptography Conference (TCC)*, 2010.

A Examples of Separator Sets

A.1 Separator Sets for Empirical Loss Queries

In this section we show how to construct a separator set for a family of empirical loss queries defined over a hypothesis class, given a separator set for the corresponding hypothesis class. Formally, let us consider the data domain $\mathcal{X}$ to be the set of labelled examples $\mathcal{X}_A \times \{0,1\}$, where $\mathcal{X}_A$ is the domain of *attribute vectors*. Let $\mathcal{H}$ be a hypothesis class, with each hypothesis $h : \mathcal{X}_a \to \{0,1\}$ mapping attribute vectors to binary labels. Let $U_{\mathcal{H}}$ be a separator set for $\mathcal{H}$ such that for any pair of distinct hypotheses $h, h' \in \mathcal{H}$, there is a $u \in U_{\mathcal{H}}$ such that $h(u) \neq h'(u)$.

For every $h \in \mathcal{H}$, let $q_h \colon \mathcal{X} \to \{0,1\}$ be the *loss query* corresponding to h such that $q_h((x,y)) = \mathbb{1}[h(x) \neq y]$. Define the query class $\mathcal{Q}_{\mathcal{H}}$ to be $\{q_h \mid h \in \mathcal{H}\}$, and let $U = \{(u,0) \mid u \in U_{\mathcal{H}}\}$. Solving the learning problem over $\mathcal{H}$ corresponds to solving the minimization problem over $\mathcal{Q}_{\mathcal{H}}$.

Claim 1. *The set U is a separator set for the query class $\mathcal{Q}_{\mathcal{H}}$.*

Proof. Let $q_h, q_{h'} \in \mathcal{Q}_{\mathcal{H}}$ be a pair of distinct queries. Since $U_{\mathcal{H}}$ is a separator set for $\mathcal{H}$, there exists an element $u \in U_{\mathcal{H}}$ such that $h(u) \neq h(u')$. As a result,

$$q_h((u,0)) = h(u) \neq h(u') = q_{h'}((u,0)).$$

Therefore, U is a separator set for $\mathcal{Q}_{\mathcal{H}}$. □

Thus, if a hypothesis class $\mathcal{H}$ has a separator set of size m, so does the set of queries $\mathcal{Q}_{\mathcal{H}}$ representing the empirical loss of the hypotheses $h \in \mathcal{H}$.

A.2 Separator Sets for Common Hypothesis Classes

In this section we provide some examples of hypothesis classes with small separator sets. We say that a class of queries $\mathcal{Q}$ is self-dual of $\mathcal{Q} = \mathcal{Q}_{\text{dual}}$.

A.2.1 Conjunctions, Disjunctions, and Parities

We begin with some easy but important cases that can be verified by inspection:

Fact 1. *Let $\mathcal{X}_A = \{0,1\}^d$. For every $j \in [d]$, let $e_j \in \mathcal{X}_A$ be a boolean vector that has 1 in the j-th coordinate and 0 in all others, and let $\overline{e}_j \in \mathcal{X}_A$ be the vector that has 0 in the j-th coordinate and 1 in all others. Let $U = \{e_j \mid j \in [d]\}$ and $\overline{U} = \{\overline{e}_j \mid j \in [d]\}$. Then for the following hypothesis classes:*

- *$\overline{U}$ is a separator set for conjunctions over $\mathcal{X}_A$: $\{\wedge_{j \in S} x_j \mid S \subseteq [d]\}$;*
- *U is a separator set for disjunctions over $\mathcal{X}_A$: $\{\vee_{j \in S} x_j \mid S \subseteq [d]\}$;*
- *U is a separator set for parities over $\mathcal{X}_A$: $\{\oplus_{j \in S} x_j \mid S \subseteq [d]\}$.*

Hence, each of these classes has a separator set of size d, equal to the data dimension. Moreover, each of these classes is self-dual.

Remark 4. *Note here we have defined* monotone *conjunctions, disjunctions, and parities — i.e. in which the literals cannot appear negated. Up to a factor of 2 in the dimension, this is without loss of generality, since we can add d extra coordinates to each example which by convention will encode the negation of each of the values in the first d coordinates. This allows us to handle non-monotone conjunctions, disjunctions, and parities as well.*

A.2.2 Discrete Halfspaces

Discrete halfspaces are a richer set of hypotheses that generalize both conjunctions and disjunctions. Let $\mathcal{X}_A = B^d$ for some set $B \subseteq [-1, 1]$. For example we could allow real valued features by letting $B = [-1, 1]$, or we could take some discretization. We will assume that $0 \in B$. Halfspaces themselves will be defined with respect to vectors of weights w that are discretized to lie in some finite set $w \in \mathcal{V}^d$, for $\mathcal{V} \subseteq [-1, 1]$. Here we could take $|\mathcal{V}| = 2$ by requiring that the *weights* be defined over the hypercube ($\mathcal{V} = \{-1, 1\}$), or we could allow finer discretization. It is important that $\mathcal{V}$ be finite.

Definition 15 (Halfspace Query)**.** *Given a weight vector $w \in \mathcal{V}^d$, the halfspace query parameterized by w is defined to be $q_w(x) = \mathbf{1}\{w \cdot x \geq 1\}$. Let $\mathcal{Q}_\mathcal{V} = \{q_w : w \in \mathcal{V}^d\}$.*

The value of "1" used as the intercept is arbitrary, and can is set without loss of generality at the cost of 1 extra data dimension.

Lemma 15. *$\mathcal{Q}_\mathcal{V}$ has a separator set of size $(|\mathcal{V}| - 1)d$. In particular, if weights are defined over the hypercube ($\mathcal{V} = \{-1, 1\}$), then the separator set is of size d.*

Proof. Suppose the elements of $\mathcal{V}$ are $x_1 < x_2 < \ldots < x_{|\mathcal{V}|}$, which without loss of generality are distinct. We construct a separator set of size $(|\mathcal{V}| - 1)d$ as follows. Let $c_1, \ldots c_{|\mathcal{V}|-1}$ be a sequence such that c_v lies in $[\frac{1}{x_{v+1}}, \frac{1}{x_v})$. Define the vector $s_{jv} \in \mathcal{X}$ to take value c_v in coordinate j, and 0 elsewhere. We claim that $U = \{s_{jv}\}_{j=1,\ldots d, v=1\ldots|\mathcal{V}|-1}$ is a separator set for $\mathcal{Q}_\mathcal{V}$. Let $q_{w_1} \neq q_{w_2} \in \mathcal{Q}_\mathcal{V}$. Since $w_1 \neq w_2$ they must differ in some coordinate, call it k. Let $w_{1,k} = x_l, q_{2,k} = x_m$, and without loss of generality assume $x_m > x_l$. Then by construction of $\{c_v\}$ there exists c_v such that $c_v \geq \frac{1}{x_m}$, but $c_v < \frac{1}{x_l}$. We therefore have that $w_1 \cdot s_{kv} = x_l \cdot c_v < 1$, whereas $w_2 \cdot s_{kv} = x_m \cdot c_v \geq 1$, and hence $q_{w_1}(s_{kv}) = 0 \neq 1 = q_{w_2}(s_{kv})$. □

Finally, we note that the dual of $\mathcal{Q}_\mathcal{V}$ is $\mathcal{Q}_\mathcal{B}$ — and so if $B = \mathcal{V}$, the set of halfspace queries $\mathcal{Q}_\mathcal{V}$ is self-dual.

A.2.3 Decision Lists

For simplicity, in this section we discuss *monotone* decision lists, in which variables cannot be negated — but as we have already remarked, this is without loss of generality up to a factor of 2 in the dimension. Here we define the general class of k-decision lists: 1-decision lists (often just referred to as decision lists) are a restricted class of binary decision tree in which one child of every internal vertex must be a leaf. k-decision lists are a generalization in which each branching decision can depend on a conjunction of k variables.

Definition 16. *A monotone k-decision list over $\mathcal{X}_A = \{0, 1\}^d$ is defined by an ordered sequence $L = (c_1, b_1), \ldots, (c_l, b_l)$ and a bit b, in which each c_i is a monotone conjunction of at most k literals, and each $b_i \in \{0, 1\}$. Given a pair (L, b), the decision list $q_{L,b}(x)$ computes as follows: it outputs $q_{L,b}(x) = b_j$ where*

j is the minimum index in L satisfying $c_j(a) = 1$. If c_j is the first conjunction that x satisfies in the definition of $q_{L,b}(x)$ we say that x binds at c_j. If no such index exists then $q_{L,b}(x) = b$, and we say $q_{L,b}(x)$ does not bind. k-decision lists are strict generalizations of k-DNF and k-CNF formulae.

Lemma 16. *The class of k-decision lists has a separator set of size $\leq \sum_{j=0}^{2k} \binom{d}{j}$. In particular, 1-decision lists have a separator set of size $O(d^2)$.*

Proof. Let MC_k denote the set of monotone conjunctions of $\leq k$ literals over $\mathcal{X}_{\mathcal{A}}$. For any two s, l in MC_k, let e_{sl} denote the element $a \in \mathcal{X}_A$ such that all literals appearing in either s or l are set to 1, and all others are set to 0. Define $U = \{e_{sl} : s, l \in MC_k\}$. Since each e_{sl} corresponds to setting between 0 and $2k$ of the d variables to 1, there are $\sum_{j=0}^{2k} \binom{d}{j}$ elements in U.
Now let $(L_1, b_1) \neq (L_2, b_2)$ be two distinct k-decision lists. Since the two decision lists are distinct there must exist $x \in \mathcal{X}_A$ such that $q_{(L_1,b_1)}(x) \neq q_{(L_2,b_2)}(x)$. Let c_1, c_2 be the conjunctions on which $(L_1, b_1), (L_2, b_2)$ bind on x respectively. (If L_i does not bind on x, set c_i to be the empty conjunction). Define $e_{c_1,c_2} \in U$ as above. We claim that $q_{(L_1,b_1)}(e_{c_1,c_2}) = q_{(L_1,b_1)}(x) \neq q_{(L_2,b_2)}(x) = q_{(L_2,b_2)}(e_{c_1,c_2})$, and hence $e_{c_1,c_2} \in U$ distinguishes (L_1, b_1) from (L_2, b_2), which proves the claim. The key fact that $q_{(L_i,b_i)}(e_{c_1,c_2}) = q_{(L_i,b_i)}(x)$ follows from the fact that (L_i, b_i) still binds at c_i (or does not bind at all) on input x, and it can't bind earlier since any monotone conjunction satisfied by e_{c_1,c_2} is satisfied by x. □

A.2.4 Other Classes of Functions

In this section, we have exhibited simple constructions of small universal identification sets for conjunctions, disjunctions, parities, discrete halfspaces, and k-decision lists. This is not an exhaustive enumeration of such classes — we give these as examples of the most frequently studied classes of boolean functions in the PAC learning literature. However, short universal identification sets for other classes of functions are known. For example:

Theorem 17 ([GKS93])**.** *There exist polynomially sized universal identification sets for the following two classes of circuits:*

1. *Logarithmic depth read-once majority formulas, and*
2. *Logarithmic depth read-once positive NAND formulas.*

B RSPM with Gaussian Perturbations

We now present a Gaussian variant of the RSPM algorithm.

Theorem 18 (Utility)**.** *The Gaussian RSPM algorithm is an oracle-efficient (α, β)-minimizer for $\mathcal{Q}$ for:*

$$\alpha = O\left(\frac{m\sqrt{m\ln(2m/\beta)\ln(1/\delta)}}{\varepsilon n}\right)$$

Proof. Note that for each noise variable η_i, we have the following tail bound:

$$\Pr[|\eta_i| \geq t] \leq 2\exp\left(-\frac{t^2}{2\sigma^2}\right)$$

Gaussian RSPM

Given: A separator set $U = \{e_1, \ldots, e_m\}$ for a class of statistical queries $\mathcal{Q}$ and a weighted optimization oracle $\mathcal{O}^*$ for $\mathcal{Q}$, privacy parameters ϵ and $\delta \in (0, 1/e)$, and $\sigma = \frac{3.5\sqrt{m \ln(1/\delta)}}{\epsilon}$.

Input: A dataset $S \in \mathcal{X}^n$ of size n.

Output: A statistical query $q \in \mathcal{Q}$.

Sample independently $\eta_i \sim \mathcal{N}(0, \sigma^2)$ for $i \in \{1, \ldots, m\}$

Construct a weighted dataset WD of size $n + m$ as follows:

$$WD(S, \eta) = \{(x_i, 1) : x_i \in S\} \cup \{(e_i, \eta_i) : e_i \in U\}$$

Output $q = \mathcal{O}(WD(S, \eta))$.

Taking a union bound, we have with probability at least $1 - \beta$ that

$$\max_i |\eta_i| \leq \sqrt{2 \ln(2m/\beta)}\sigma$$

Then the proof follows from the same reasoning in the proof for Theorem 5. □

Theorem 19 (Privacy)**.** *If $\mathcal{O}^*$ is a weighted optimization oracle for $\mathcal{Q}$, then the Gaussian Report Separator-Perturbed Min algorithm is (ϵ, δ)-differentially private.*

Proof. In the following, we will inherit the notation from Section 3. We will denote the output by the algorithm on input dataset S under realizations of the perturbations η as $\mathcal{Q}(S, \eta) = \mathcal{O}^*(WD(S, \eta))$. For any $q \in \mathcal{Q}$, let $\mathcal{E}(q, S) = \{\eta : \mathcal{Q}(S, \eta) = q\}$. We will use the same mapping $f_q(\eta) : \mathbb{R}^m \to \mathbb{R}^m$ as defined in Section 3. We will write P_G to denote the pdf of the distribution $\mathcal{N}(0, \sigma^2)$. We will also again define the set B as the set of η for which there are multiple minimizers $\hat{q}$. For every $\eta \in \mathbb{R}^m \setminus B$, let $\hat{q}_\eta$ be the unique minimizer. Note that Lemmas 2 and 4 both hold in our setting, and in particular, Lemma 4 holds because of the continuity of the Gaussian distribution.

Similar to the standard analysis for the Gaussian mechanism [DR14], we will leverage the fact that the distribution $\mathcal{N}(0, \sigma^2 I)$ is independent of the orthonormal basis from which its constituent normals are drawn, so we have the freedom to choose the underlying basis without changing the distribution. For any $r \in \mathbb{R}^m$ and any $q \in \mathcal{Q}$, fix such a basis $b_1, \ldots, b_m$ such that b_1 is parallel to $v = f_q(r) - r$. A random draw η from $\mathcal{N}(0, \sigma^2 I)$ can be realized by the following process: first draw signed lengths $\lambda_i \sim \mathcal{N}(0, \sigma^2)$, for $i \in [m]$, then define $\eta^{[i]} = \lambda_i b_i$, and finally let $\eta = \sum_{i=1}^m \eta^{[i]}$. For each $i \in [m]$, let $r^{[i]}$ be the projection of r onto the direction of b_i, which gives $r = \sum_{i=1}^m r^{[i]}$.

Lemma 17. *Suppose that $\sigma > 1$. For any $r \in \mathbb{R}^m$, $q \in \mathcal{Q}$,*

$$P_G(r) \leq \exp\left(\frac{1}{2\sigma^2}\left(m + 2\sqrt{m}\|r^{[1]}\|_2\right)\right) P_G(f_q(r)).$$

Proof. Note that for any $r \in \mathbb{R}^m$, we have

$$P_G(r) = \frac{1}{(2\pi)^{m/2}\sigma^m} \exp\left(-\frac{\|r\|_2^2}{2\sigma^2}\right).$$

We will write $\nu = f_q(r) - r$. It follows that

$$\frac{P_G(r)}{P_G(f_q(r))} = \exp\left(\frac{1}{2\sigma^2}\left(\|r+\nu\|_2^2 - \|r\|_2^2\right)\right)$$

Now we can write

$$\|r+\nu\|_2^2 = \|\nu + r^{[1]}\|_2^2 + \sum_{i=1}^{m} \|r^{[i]}\|_2^2, \qquad \|r\|_2^2 = \sum_{i=1}^{m} \|r^{[i]}\|_2^2.$$

It follows that

$$\|r^{[1]}+\nu\|_2^2 - \|r^{[1]}\|_2^2 = \|\nu\|_2^2 + 2\|\nu\|_2\,\|r^{[1]}\|_2 \le m + 2\sqrt{m}\|r^{[1]}\|_2$$

This means

$$\frac{P_G(r)}{P_G(f_q(r))} \le \exp\left(\frac{1}{2\sigma^2}\left(m + 2\sqrt{m}\|r^{[1]}\|_2\right)\right),$$

which completes the proof. □

To finish up the privacy analysis, note that by Lemma 17, the ratio $P_G(\eta)/P_G(f_q(\eta))$ is bounded by $\exp(\varepsilon)$, as long as $\|\eta^{[1]}\|_2 < \sigma^2\varepsilon/\sqrt{m} - \sqrt{m}/2$. Now we will bound the probability that the random vector $\eta^{[1]}$ has norm exceeding this bound. First, observe that $\|\eta^{[1]}\|_2 = |\lambda_1|$, where λ_1 is a random draw from the distribution $\mathcal{N}(0,\sigma^2)$. Since λ_1^2 is a χ^2 random variable with degree of freedom 1, we can apply the following tail bound [LM00]: for any $t > 0$,

$$\Pr[\lambda_1^2 \ge \sigma^2\left(\sqrt{2t}+1\right)^2] \le \exp(-t)$$

which can be further simplied to

$$\Pr[\|\eta^{[1]}\|_2 \ge \sigma\left(\sqrt{2t}+1\right)] \le \exp(-t)$$

In other words, for any $\delta \in (0, 1/e)$, with probability at least $1-\delta$, we have

$$\|\eta^{[1]}\|_2 < \sigma(\sqrt{2\ln(1/\delta)}+1) \equiv \Lambda$$

It follows that $\Lambda \le \sigma^2\varepsilon/\sqrt{m} - \sqrt{m}/2$, as long as $\sigma = \frac{c\sqrt{m\ln(1/\delta)}}{\varepsilon}$ for any $c \ge 3.5$. We will use this value of σ for the remainder of the analysis. Now let $L = \{\eta \in \mathbb{R}^m \mid \|\eta^{[1]}\|_2 > \Lambda\}$, then we know that

$\Pr[\eta \in L] < \delta$. Let $S \subset \mathcal{Q}$ be a subset of queries. It follows that:

$$\begin{aligned}
\Pr\left[\eta \in \bigcup_{\hat{q}\in S} \mathcal{E}(\hat{q}, S)\right] &= \int_{\mathbb{R}^m} P_G(\eta) \mathbb{1}\left(\eta \in \bigcup_{\hat{q}\in S} \mathcal{E}(\hat{q}, S)\right) d\eta \\
&= \int_{(\mathbb{R}^m \setminus B)\setminus L} P_G(\eta) \mathbb{1}\left(\eta \in \bigcup_{\hat{q}\in S} \mathcal{E}(\hat{q}, S)\right) d\eta + \int_L P_G(\eta) \mathbb{1}\left(\eta \in \bigcup_{\hat{q}\in S} \mathcal{E}(\hat{q}, S)\right) d\eta \\
&\leq \int_{(\mathbb{R}^m \setminus B)\setminus L} P_G(\eta) \mathbb{1}\left(\eta \in \bigcup_{\hat{q}\in S} \mathcal{E}(\hat{q}, S)\right) d\eta + \delta \\
&= \sum_{\hat{q}\in S} \int_{\mathbb{R}^m \setminus (B\cup L)} P_G(\eta) \mathbb{1}\left(\eta \in \mathcal{E}(\hat{q}, S)\right) d\eta + \delta \\
&\leq \sum_{\hat{q}\in S} \int_{\mathbb{R}^m \setminus (B\cup L)} P_G(\eta) \mathbb{1}\left(f_{\hat{q}}(\eta) \in \mathcal{E}(\hat{q}, S')\right) d\eta + \delta && (\textit{Lemma } 2) \\
&\leq \sum_{\hat{q}\in S} \int_{\mathbb{R}^m \setminus (B\cup L)} \exp(\varepsilon) P_G(f_{\hat{q}}(\eta)) \mathbb{1}\left(f_{\hat{q}}(\eta) \in \mathcal{E}(\hat{q}, S')\right) d\eta + \delta && (\textit{Lemma } 17) \\
&= \sum_{\hat{q}\in S} \int_{\mathbb{R}^m \setminus (f_{\hat{q}}(B)\cup f_{\hat{q}}(L))} \exp(\varepsilon) P_G(\eta) \mathbb{1}\left(\eta \in \mathcal{E}(\hat{q}, S')\right) \left|\frac{\partial f_{\hat{q}}}{\partial \eta}\right| d\eta + \delta \\
&\leq \exp(\varepsilon) \sum_{\hat{q}\in S} \int_{\mathbb{R}^m} P_G(\eta) \mathbb{1}\left(\eta \in \mathcal{E}(\hat{q}, S')\right) d\eta + \delta && \left(\forall \hat{q}, \left|\frac{\partial f_{\hat{q}}}{\partial \eta}\right| = 1\right) \\
&= \exp(\varepsilon) \Pr\left[\eta \in \bigcup_{\hat{q}\in S} \mathcal{E}(\hat{q}, S')\right] + \delta
\end{aligned}$$

This completes the proof. □

C Proofs and Details for Theorem 8

Lemma 1. *Let $\mathcal{A}_{\mathcal{O}}$ be a certifiable-oracle dependent algorithm that is oracle equivalent to $\mathcal{A}$. Then for any fixed input dataset S, there exists a coupling between $\mathcal{A}(S)$ and $\mathcal{A}_{\mathcal{O}}(S)$ such that* $\Pr[\mathcal{A}_{\mathcal{O}}(S) = a | \mathcal{A}_{\mathcal{O}}(S) \neq \perp] = \Pr[\mathcal{A}(S) = a | \mathcal{A}_{\mathcal{O}}(S) \neq \perp]$.

Proof. We can assume without loss of generality that $\mathcal{A}_{\mathcal{O}}$ and $\mathcal{A}$ draw all their randomness up front in the form of a random seed η, and are then a deterministic function of the random seed and the input dataset. By definition, during the run of $\mathcal{A}_{\mathcal{O}}$, the algorithm generates a (possibly randomized, and possibly adaptively chosen) sequence of inputs to the optimization oracle $\mathcal{O}$, $\{wd_1, \ldots wd_m\}$, where each wd_i is a weighted dataset. We denote the output of the i^{th} optimization problem by o_i. After the m^{th} optimization problem, $\mathcal{A}_{\mathcal{O}}$ outputs a deterministic outcome $a = h(o_1, o_2, \ldots o_m)$. Given access to a perfect optimization oracle $\mathcal{O}^*$, $\mathcal{A}_{\mathcal{O}^*}$ is simply $\mathcal{A}$ – this is the definition of oracle equivalence. We construct a coupling between an algorithm $\mathcal{M}$ and $\mathcal{A}_{\mathcal{O}}$, and then argue running $\mathcal{M}$ is the same as running $\mathcal{A}$:

Input: A dataset S, random seed η, heuristic oracle $\mathcal{O}$, perfect oracle $\mathcal{O}^*$.
Output: values $M(S,\eta), \mathcal{A}(S,\eta)$

Run $\mathcal{A}_{\mathcal{O}}(\eta, S)$ - generating the first optimization problem wd_1.
for $i = 1 \dots m$ **do**
  Compute $o_i = \mathcal{O}(wd_i)$
  if $o_i = \perp$ **then**
    Output $\mathcal{A}_{\mathcal{O}}(S,w) = \perp$
    Set $o_i = \mathcal{O}^*(wd_i)$
  end if
  Generate wd_{i+1} adaptively as a function of previous outputs $(o_i, o_{i-1}, \dots o_1)$
end for
Output $M(S,\eta) = h(o_1, \dots o_m) = a$
if $\mathcal{A}_{\mathcal{O}}(S,\eta) = \perp$ has not been output **then**
  Output $\mathcal{A}_{\mathcal{O}}(S,w) = a$
end if

The procedure starts by generating a random seed η and initializing a run of $\mathcal{A}_{\mathcal{O}}(\eta, S)$ - generating the first optimization problem wd_1. If the oracle $\mathcal{O}$ fails on input $wd_1, \mathcal{A}_{\mathcal{O}}$ outputs $\perp$. In this case the next optimization input wd_2 is generated as a function of the output of the perfect oracle $\mathcal{O}^*(wd_1)$. If it succeeds, we simply generate the next output as a function of $\mathcal{O}(wd_1)$ (which is the same as $\mathcal{O}^*(wd_1)$ by definition of certifiability). This process continues until we solve the m^{th} optimization problem, and output $M(\eta, S), \mathcal{A}_{\mathcal{O}}(\eta, S)$ as described above. Now it is clear that if the oracle doesn't fail, we generate the same output a for $\mathcal{A}_{\mathcal{O}}$ and $\mathcal{M}$. No matter whether or not $\mathcal{O}$ fails, $\mathcal{M}$ has output that corresponds to perfectly solving the optimization problems generated with input S, η, and so it is equivalent to running $\mathcal{A}$. Moreover, whenever the oracle does not fail, $\mathcal{A}$ and $\mathcal{M}$ have the same output, by construction. This completes the proof. □

Definition 17 ([PP14]). *Let $(X_1, \dots, X_n) \subset \{0,1\}^n$ be an ensemble of n $\{0,1\}$-valued random variables. We say that $(X_1, \dots X_n)$ satisfy the stochastic covering property, if for any $I \subset [n]$, $J = [n] \backslash I$, and $a \geq a' \in \{0,1\}^{|I|}$, where $\geq$ denotes coordinate-wise dominance, such that $\|a' - a\|_1 = 1$, there is a coupling v of the distributions μ, μ' on $(X_k)_{k \in J}$ conditioned on $(X_k)_{k \in I} = a$ or $(X_k)_{k \in I} = a'$ respectively, such that $v(x,y) = 0$ unless $x \leq y$ and $\|x - y\|_1 \leq 1$.*

Lemma 18. *Given a set $|S| = n$, subsample $k \leq n$ elements without replacement. Let $X_i \in \{0,1\}$ be 1 if element $x_i \in S$ is subsampled, else 0. Then $(X_1, \dots X_n)$ satisfy the stochastic covering property.*

Proof. For $a \in \{0,1\}^{|I|}$ let $|a|$ be the number of 1's in a. Then the distribution of $x = X_J | a$ corresponds to subsampling $k - |a|$ elements from $(x_k)_{k \in J}$, and the distribution of $y = X_J | a'$ corresponds to subsampling $k - |a| + 1$ elements from $(x_k)_{k \in J}$ without replacement. To establish the stochastic covering property, we exhibit a coupling v of x, y:
To generate y subsample $k - |a| + 1$ elements from $(x_k)_{k \in J}$ without replacement. Let x be the first $k - |a|$ such elements subsampled. Both x, y constructed as such have the correct marginal distributions, and by construction $x \leq y, \|x - y\|_1 = 1$ always. □

Definition 18. *$(X_1, \dots, X_n) \subset \{0,1\}^n$ are k-homogenous if* $\Pr[\sum_{i=1}^n X_i = k] = 1$.

Theorem 20 (Theorem 3.1 in [PP14])**.** *Let $(X_1,\dots X_n) \in \{0,1\}$ be k-homogenous random variables satisfying the stochastic covering property. Let $f : \{0,1\}^n \to \mathbb{R}$ be an ϵ–Lipschitz function, and let $\mu = \mathbb{E}[f(X_1,\dots X_n)]$. Then for any $t > 0$:*

$$\Pr[|f(X_1,\dots X_n) - \mu| \geq t] \leq 2e^{\frac{-t^2}{8\epsilon^2 k}}$$

Lemma 5. *Let $\mathcal{P}(S) \sim \mathcal{P}^S_{split}$. Let $\mathcal{A} : \mathcal{X}^l \to \mathcal{M}$ be an $(\epsilon', 0)$ differentially private algorithm, where: $\epsilon' = \frac{1}{\sqrt{8\frac{n}{K}\log(2K/\delta)}}$. Fix $\Omega \subset M$, and let $q_\Omega(S_i) = \log \Pr[\mathcal{A}(S_i) \in \Omega]$. Define Q to be the event*

$$Q = \{\mathcal{P}(S) : \max_{i,j \in 1\dots K} |q_\Omega(S_i) - q_\Omega(S_j)| \leq 2\}.$$

Then over the random draw of $\mathcal{P}(S) \sim \mathcal{P}^S_{split}$, $\Pr[Q] \geq 1 - \delta$.

Proof. Fix any index k of the partition. Let $\{X_i\}_{i=1}^n$ be the indicator random random variables indicating that element i in S, is included in S_k. Since S_k is entirely determined by $\{X_i\}$, we can write $q_\Omega(S_k)$ as a function of $\{X_i\}$, e.g. $q_\Omega(X_1,\dots X_n)$. Moreover, by definition of ϵ-differential privacy, q_Ω is ϵ-Lipschitz, i.e. for any X_i, X_i':

$$|q_\Omega(X_1,\dots X_i,\dots X_n) - q_\Omega(X_1,\dots X_i',\dots X_n)| \leq \epsilon$$

By Lemma 18 proven in the Appendix, $(X_1,\dots X_n)$ satisfy what is called the *stochastic covering property*, a type of negative dependence. Since $|S_k| = n/K$, $(X_1,\dots X_n)$ are n/K-homogenous. Thus by Theorem 1 of [PP14], with probability $1 - \delta/K$:

$$|q_\Omega(S_k) - \mathop{\mathbb{E}}_{S_k \sim \mathcal{P}^S_{split}}[q_\Omega(S_k)]| \leq \epsilon' \sqrt{8\frac{n}{K}\log(2K/\delta)} = 1$$

So by a union bound this holds for all $k = 1\dots K$ with probability at least $1 - \delta$. Since for all $i, j, \mathbb{E}_{S_i \sim \mathcal{P}^S_{split}}[q_\Omega(S_i)] = \mathbb{E}_{S_j \sim \mathcal{P}^S_{split}}\left[q_\Omega(S_j)\right]$, by the triangle inequality for all $i, j, |q_\Omega(S_i) - q_\Omega(S_j)| \leq 2$ with probability $1 - \delta$, as desired.

□

Lemma 6.

$$\frac{\Pr[\mathcal{A}^{prsm}_{\mathcal{O}}(\mathcal{S}) \in \Omega]}{\Pr[\mathcal{A}^{prsm}_{\mathcal{O}}(\mathcal{S}') \in \Omega]} \leq \frac{\sum_{\mathcal{P}(S) \in S_Q, \boldsymbol{o} \in \mathcal{F}} \Pr[\mathcal{A}^{prsm}_{\mathcal{O}}(\mathcal{S}) \in \Omega | P(S), \boldsymbol{o}, \mathcal{L}] \Pr[\boldsymbol{o}, P(S) | \mathcal{L}] + 4\delta}{\Pr[\mathcal{A}^{prsm}_{\mathcal{O}}(\mathcal{S}') \in \Omega]}$$

Proof. Conditioning on $\mathcal{L}$ and using $\Pr[\mathcal{L}] \geq 1 - 2\delta$ we have:

$$\frac{\Pr[\mathcal{A}^{prsm}_{\mathcal{O}}(\mathcal{S}) = a]}{\Pr[\mathcal{A}^{prsm}_{\mathcal{O}}(\mathcal{S}') \in \Omega]} \leq \frac{2\delta + \Pr\left[\mathcal{A}^{prsm}_{\mathcal{O}}(\mathcal{S}) \in \Omega | \mathcal{L}\right]}{\Pr[\mathcal{A}^{prsm}_{\mathcal{O}}(\mathcal{S}') \in \Omega]}$$

Expanding $\Pr\left[\mathcal{A}^{prsm}_{\mathcal{O}}(\mathcal{S}) \in \Omega | \mathcal{L}\right]$ by conditioning on $\mathcal{F}, \mathcal{P}(S)$ and using the law of total probability

we have:

$$\frac{\Pr[\mathcal{A}_{\mathcal{O}}^{prsm}(\mathcal{S}) = a|\mathcal{L}]}{\Pr[\mathcal{A}_{\mathcal{O}}^{prsm}(\mathcal{S}') \in \Omega]} = \frac{\sum_{\mathcal{P}(S),\mathbf{o}\in\mathcal{F}} \Pr[\mathcal{A}_{\mathcal{O}}^{prsm}(\mathcal{S}) \in \Omega|P(S),\mathbf{o},\mathcal{L}]\Pr[\mathbf{o},P(S)|\mathcal{L}]}{\Pr[\mathcal{A}_{\mathcal{O}}^{prsm}(\mathcal{S}') \in \Omega]} = \tag{12}$$

Separating the summation in the numerator over $\mathcal{P}(S)$ into S_Q, S_Q^c we have:

$$\sum_{\mathcal{P}(S),\mathbf{o}\in\mathcal{F}} \Pr[\mathcal{A}_{\mathcal{O}}^{prsm}(\mathcal{S}) \in \Omega|P(S),\mathbf{o},\mathcal{L}]\Pr[\mathbf{o},P(S)|\mathcal{L}] = \sum_{\mathcal{P}(S)\in S_Q,\mathbf{o}\in\mathcal{F}} \Pr[\mathcal{A}_{\mathcal{O}}^{prsm}(\mathcal{S}) \in \Omega|P(S),\mathbf{o},\mathcal{L}]\Pr[\mathbf{o},P(S)|\mathcal{L}] +$$

$$\sum_{\mathcal{P}(S)\in S_Q^c,\mathbf{o}\in\mathcal{F}} \Pr[\mathcal{A}_{\mathcal{O}}^{prsm}(\mathcal{S}) \in \Omega|P(S),\mathbf{o},\mathcal{L}]\Pr[\mathbf{o},P(S)|\mathcal{L}]$$

Rewriting the second term,

$$\sum_{\mathcal{P}(S)\in S_Q^c,\mathbf{o}\in\mathcal{F}} \Pr[\mathcal{A}_{\mathcal{O}}^{prsm}(\mathcal{S}) \in \Omega|P(S),\mathbf{o},\mathcal{L}]\Pr[\mathbf{o},P(S)|\mathcal{L}] =$$

$$\sum_{\mathcal{P}(S)\in S_Q^c} (\sum_{\mathbf{o}\in\mathcal{F}} \Pr[\mathcal{A}_{\mathcal{O}}^{prsm}(\mathcal{S}) \in \Omega|P(S),\mathbf{o},\mathcal{L}]\Pr[\mathbf{o},|P(S),\mathcal{L}])\Pr[P(S)|\mathcal{L}] =$$

$$\sum_{\mathcal{P}(S)\in S_Q^c} \Pr[\mathcal{A}_{\mathcal{O}}^{prsm}(\mathcal{S}) \in \Omega|P(S),\mathcal{L}])\Pr[P(S)|\mathcal{L}] \leq \sum_{\mathcal{P}(S)\in S_Q^c} \Pr[P(S)|\mathcal{L}] = \Pr[Q^c|\mathcal{L}]$$

The first equality follows from the fact that $\Pr[\mathbf{o},\mathcal{P}(S)|\mathcal{L}] = \Pr[\mathbf{o}|\mathcal{P}(S),\mathcal{L}]\Pr[\mathcal{P}(S)|\mathcal{L}]$, and the second equality follows from the law of total probability. Since $\Pr[Q^c] \leq \delta$, and $\Pr[\mathcal{L}] \geq 1-2\delta$, $\Pr[Q^c|\mathcal{L}] \leq \delta/(1-2\delta) \leq 2\delta$, for $\delta \leq 1/4$. Thus

$$\frac{\Pr[\mathcal{A}_{\mathcal{O}}^{prsm}(\mathcal{S}) = a|\mathcal{L}]}{\Pr[\mathcal{A}_{\mathcal{O}}^{prsm}(\mathcal{S}') \in \Omega]} \leq \frac{\sum_{\mathcal{P}(S)\in S_Q,\mathbf{o}\in\mathcal{F}} \Pr[\mathcal{A}_{\mathcal{O}}^{prsm}(\mathcal{S}) \in \Omega|P(S),\mathbf{o},\mathcal{L}]\Pr[\mathbf{o},P(S)|\mathcal{L}] + 2\delta}{\Pr[\mathcal{A}_{\mathcal{O}}^{prsm}(\mathcal{S}') \in \Omega]}$$

Combining this bound, with the bound $\frac{\Pr[\mathcal{A}_{\mathcal{O}}^{prsm}(\mathcal{S})=a]}{\Pr[\mathcal{A}_{\mathcal{O}}^{prsm}(\mathcal{S}')\in\Omega]} \leq \frac{2\delta+\Pr[\mathcal{A}_{\mathcal{O}}^{prsm}(\mathcal{S})\in\Omega|\mathcal{L}]}{\Pr[\mathcal{A}_{\mathcal{O}}^{prsm}(\mathcal{S}')\in\Omega]}$, establishes the result.

□

Lemma 7. *Fix any $\boldsymbol{o}$, any $\mathcal{P}(S) \in S_Q$, and index $j \in I^{\mathbf{o}}_{pass}$, i.e. $o_j \neq \perp$. Then:*

$$\Pr[\mathcal{A}_{\mathcal{O}}(S_j) \in \Omega|\mathcal{A}_{\mathcal{O}}(S_j) \neq \perp,\mathcal{L}] \leq \frac{e^2}{(1-\delta)^2}\frac{1}{|\boldsymbol{o}|-1}\sum_{i\in I^{\mathbf{o}}_{pass},i\neq j} \Pr[\mathcal{A}_{\mathcal{O}}(S_i) \in \Omega|\mathcal{A}_{\mathcal{O}}(S_i) \neq \perp] + \frac{\delta e^2}{(1-\delta)}$$

Proof of Lemma 7. By Equation 3, we know:

$$\Pr[\mathcal{A}_{\mathcal{O}}^{prsm}(\mathcal{S}) \in \Omega|P(S),\mathbf{o},\mathcal{L}] = \frac{1}{|\mathbf{o}|}\sum_{i\in I^{\mathbf{o}}_{pass}} \Pr[\mathcal{A}_{\mathcal{O}}(S_i) = a|\mathcal{A}_{\mathcal{O}}(S_i) \neq \perp]$$

We also know that since we've conditioned on $\mathcal{L}$ (and hence on E), for each $i \in I^{\mathbf{o}}_{pass}$ on the RHS

of the above equation, $\Pr[\mathcal{A}_{\mathcal{O}}(S_i) = \perp] \leq \delta$. By Lemma 1, we know that there exists a coupling between $\mathcal{A}_{\mathcal{O}}(S_i)$ and $\mathcal{A}(S_i)$ such that $\Pr[\mathcal{A}_{\mathcal{O}}(S_i) = a | \mathcal{A}_{\mathcal{O}}(S_i) \neq \perp] = \Pr[\mathcal{A}(S_i) = a | \mathcal{A}_{\mathcal{O}}(S_i) \neq \perp]$.

By the law of total probability:

$$\Pr[\mathcal{A}(S_i) = a] = \Pr[\mathcal{A}(S_i) = a | \mathcal{A}_{\mathcal{O}}(S_i) \neq \perp] \Pr[\mathcal{A}_{\mathcal{O}}(S_i) \neq \perp] + \Pr[\mathcal{A}(S_i) = a | \mathcal{A}_{\mathcal{O}}(S_i) = \perp] \Pr[\mathcal{A}_{\mathcal{O}}(S_i) = \perp]$$

Then $\Pr[\mathcal{A}(S_i) = a | \mathcal{A}_{\mathcal{O}}(S_i) \neq \perp] \Pr[\mathcal{A}_{\mathcal{O}}(S_i) \neq \perp] \leq \Pr[\mathcal{A}(S_i) = a] \implies \Pr[\mathcal{A}(S_i) = a | \mathcal{A}_{\mathcal{O}}(S_i) \neq \perp] \leq \frac{1}{1-\delta} \Pr[\mathcal{A}(S_i) = a]$, and similarly $\Pr[\mathcal{A}(S_i) = a | \mathcal{A}_{\mathcal{O}}(S_i) \neq \perp] \geq \Pr[\mathcal{A}(S_i) = a] - \delta$.

Since we've shown that each of the conditional probabilities $\Pr[\mathcal{A}_{\mathcal{O}}(S_i) = a | \mathcal{A}_{\mathcal{O}}(S_i) \neq \perp]$ is close to $\Pr[\mathcal{A}_{\mathcal{O}}(S_i) = a]$, since we assume $\mathcal{P}(S) \in S_Q$, we know they are close to each other. Using the inequalities above:

$$\Pr[\mathcal{A}(S_j) = a | o_j \neq \perp] \leq \frac{1}{1-\delta} \Pr[\mathcal{A}(S_j) = a] \leq \frac{e^2}{1-\delta} \Pr[\mathcal{A}(S_i) = a] \leq \frac{e^2}{1-\delta} \left(\frac{\Pr[\mathcal{A}(S_i) = a | o_i \neq \perp]}{1-\delta} + \delta \right),$$

where the middle inequality follows from the definition of S_Q. Summing both sides over $i : o_i \neq \perp, i \neq j$ and rearranging gives the desired result. □

Lemma 8.

$$\sum_{\mathcal{P}(S) \in S_Q, \boldsymbol{o} \in \mathcal{F}} \Pr[\mathcal{A}_{\mathcal{O}}^{prsm}(\mathcal{S}) \in \Omega | P(S), \boldsymbol{o}, \mathcal{L}] \Pr[\boldsymbol{o}, P(S) | \mathcal{L}] \leq$$

$$(1 + \frac{e^2}{(1-\delta)^2(\frac{1}{\epsilon} - 1)}) \sum_{P(S) \in S_Q} \left(\sum_{\boldsymbol{o} \in \mathcal{F}} \Big(\frac{1}{|\boldsymbol{o}|} \sum_{i \in I^{\boldsymbol{o}}_{pass}, i \neq 1} \Pr[\mathcal{A}_{\mathcal{O}}(S_i) \in \Omega | \mathcal{A}_{\mathcal{O}}(S_i) \neq \perp] \Big) \Pr[\boldsymbol{o} | P(S)] \right) \Pr[P(S)] + \frac{\epsilon \delta e^2}{1-\delta}$$

Proof. Denote $\sum_{P(S) \in S_Q} \left(\sum_{\mathbf{o} \in \mathcal{F} : o_1 \neq \perp} \Big(\frac{1}{|\mathbf{o}|} \sum_{i \in I^{\mathbf{o}}_{pass}} \Pr[\mathcal{A}_{\mathcal{O}}(S_i) = a | \mathcal{A}_{\mathcal{O}}(S_i) \neq \perp] \Big) \Pr[\mathbf{o} | P(S)] \right)$, by $(\star)$. By Lemma 7, $(\star) \leq$

$$\sum_{P(S) \in S_Q} \left(\sum_{\mathbf{o} \in \mathcal{F} : o_1 \neq \perp} \Big(\frac{\delta e^2}{(1-\delta)|\mathbf{o}|} + (1 + \frac{e^2}{(|\mathbf{o}|-1)(1-\delta)^2}) \cdot \frac{1}{|\mathbf{o}|} \sum_{i \in I^{\mathbf{o}}_{pass}, i \neq 1} \Pr[\mathcal{A}_{\mathcal{O}}(S_i) = a | \mathcal{A}_{\mathcal{O}}(S_i) \neq \perp, i^* = i] \Big) \Pr[\mathbf{o} | P(S)] \right) \Pr[P(S)]$$

Pulling out the $\frac{\delta e^2}{(1-\delta)|\mathbf{o}|}$, we see that $\sum_{\mathbf{o} \in \mathcal{F} : o_1 \neq \perp} \frac{\delta e^2}{(1-\delta)|\mathbf{o}|} \Pr[\mathbf{o} | P(S)] \leq \sum_{\mathbf{o} \in \mathcal{F}} \frac{\delta e^2}{(1-\delta)} \cdot \epsilon \Pr[\mathbf{o} | P(S)] = \frac{\epsilon \delta e^2}{(1-\delta)}$.

Here we've used the fact that $|\mathbf{o}| > \frac{1}{\epsilon}$. Similarly, $\sum_{P(S) \in S_Q} \frac{\epsilon \delta e^2}{(1-\delta)} \Pr[P(S)] \leq \frac{\epsilon \delta e^2}{(1-\delta)}$. Applying to $(\star)$, we get:

$$(\star) \leq \sum_{P(S) \in S_Q} \left(\sum_{\mathbf{o} \in \mathcal{F} : o_1 \neq \perp} \Big((1 + \frac{e^2}{(1-\delta)^2(\frac{1}{\epsilon} - 1)}) \cdot \frac{1}{|\mathbf{o}|} \sum_{i \in I^{\mathbf{o}}_{pass}, i \neq 1} \Pr[\mathcal{A}_{\mathcal{O}}(S_i) = a | \mathcal{A}_{\mathcal{O}}(S_i) \neq \perp, i^* = i] \Big) \Pr[\mathbf{o} | P(S)] \right) \Pr[P(S)] + \frac{\epsilon \delta e^2}{1-\delta} \tag{13}$$

Applying this upper bound on ($\star$) gives:

$$\sum_{\mathcal{P}(S)\in S_Q, \mathbf{o}\in\mathcal{F}} \Pr[\mathcal{A}_{\mathcal{O}}^{prsm}(\mathcal{S})\in\Omega|P(S),\mathbf{o},\mathcal{L}]\Pr[\mathbf{o},P(S)|\mathcal{L}] \leq$$

$$\sum_{P(S)\in S_Q}\Bigg(\Big(\sum_{\mathbf{o}\in\mathcal{F}:o_1\neq\perp}\Big((1+\frac{e^2}{(1-\delta)^2(\frac{1}{\epsilon}-1)})\cdot\frac{1}{|\mathbf{o}|}\sum_{i\in I^{\mathbf{o}}_{pass},i\neq 1}\Pr[\mathcal{A}_{\mathcal{O}}(S_i)=a|\mathcal{A}_{\mathcal{O}}(S_i)\neq\perp,i^*=i]\Big)\Pr[\mathbf{o}|P(S)]\Big)+$$

$$\sum_{\mathbf{o}\in\mathcal{F}:o_1=\perp}\Big(\frac{1}{|\mathbf{o}|}\sum_{i\in I^{\mathbf{o}}_{pass}}\Pr[\mathcal{A}_{\mathcal{O}}(S_i)=a|\mathcal{A}_{\mathcal{O}}(S_i)\neq\perp,i^*=i]\Pr[\mathbf{o}|P(S))\Big)\Bigg)\Pr[P(S)]\leq$$

$$(1+\frac{e^2}{(1-\delta)^2(\frac{1}{\epsilon}-1)})\sum_{P(S)\in S_Q}\Bigg(\sum_{\mathbf{o}\in\mathcal{F}}\Big(\frac{1}{|\mathbf{o}|}\sum_{i\in I^{\mathbf{o}}_{pass},i\neq 1}\Pr[\mathcal{A}_{\mathcal{O}}(S_i)=a|\mathcal{A}_{\mathcal{O}}(S_i)\neq\perp,i^*=i]\Big)\Pr[\mathbf{o}|P(S)]\Bigg)\Pr[P(S)]$$

□

End of proof of Theorem 8.

Proof. First we rewrite the numerator:

$$\sum_{\mathbf{o}\in\mathcal{F}}\Big(\frac{1}{|\mathbf{o}|}\sum_{i\in I^{\mathbf{o}}_{pass},i\neq 1}\Pr[\mathcal{A}_{\mathcal{O}}(S_i)\in\Omega|\mathcal{A}_{\mathcal{O}}(S_i)\neq\perp]\Pr[\mathbf{o}|P(S)]\Big)=$$

$$\sum_{\mathbf{o}_{-1}\in\mathcal{F}}\sum_{i\in I^{\mathbf{o}}_{pass},i\neq 1}\Bigg(\Big(\frac{1}{|\mathbf{o}_{-1}|}\Pr[\mathcal{A}_{\mathcal{O}}(S_i)\in\Omega|\mathcal{A}_{\mathcal{O}}(S_i)\neq\perp]\Pr[o_1=\perp,\mathbf{o}_{-1}|P(S)]+$$

$$\frac{1}{|\mathbf{o}_{-1}+1|}\Pr[\mathcal{A}_{\mathcal{O}}(S_i)\in\Omega|\mathcal{A}_{\mathcal{O}}(S_i)\neq\perp]\Pr[o_1=1,\mathbf{o}_{-1}|P(S)]\Big)\Bigg)\leq$$

$$\sum_{\mathbf{o}_{-1}\in\mathcal{F}}\sum_{i\in I^{\mathbf{o}}_{pass},i\neq 1}\Big(\frac{1}{|\mathbf{o}_{-1}|-1}\Pr[\mathcal{A}_{\mathcal{O}}(S_i)\in\Omega|\mathcal{A}_{\mathcal{O}}(S_i)\neq\perp]\Pr[\mathbf{o}_{-1}|P(S)]\Big),$$

where we've used the fact that $\Pr[o_1=\perp,\mathbf{o}_{-1}|P(S)]+\Pr[o_1\neq\perp,\mathbf{o}_{-1}|P(S)]=\Pr[\mathbf{o}_{-1}|P(S)]$. Similarly, we can use this same trick to lower bound the denominator:

$$\sum_{\mathbf{o}\in\mathcal{F}}\sum_{i\in I^{\mathbf{o}}_{pass}}(\frac{1}{|\mathbf{o}|}\Pr[\mathcal{A}_{\mathcal{O}}(S'_i)\in\Omega|\mathcal{A}_{\mathcal{O}}(S'_i)\neq\perp]\Pr[\mathbf{o}|P(S')])\geq$$

$$\sum_{\mathbf{o}_{-1}\in\mathcal{F}}\sum_{i\in I^{\mathbf{o}}_{pass},i\neq 1}(\frac{1}{|\mathbf{o}_{-1}|}\Pr[\mathcal{A}_{\mathcal{O}}(S'_i)\in\Omega|\mathcal{A}_{\mathcal{O}}(S'_i)\neq\perp]\Pr[o_1\neq\perp,\mathbf{o}_{-1}|P(S')]+$$

$$\frac{1}{|\mathbf{o}_{-1}|-1}\Pr[\mathcal{A}_{\mathcal{O}}(S'_i)\in\Omega|\mathcal{A}_{\mathcal{O}}(S'_i)\neq\perp]\Pr[o_1=\perp,\mathbf{o}_{-1}|P(S')])\geq$$

$$\sum_{\mathbf{o}_{-1}\in\mathcal{F}}\sum_{i\in I^{\mathbf{o}}_{pass},i\neq 1}\Big(\frac{1}{|\mathbf{o}_{-1}|}\Pr[\mathcal{A}_{\mathcal{O}}(S'_i)\in\Omega|\mathcal{A}_{\mathcal{O}}(S'_i)\neq\perp]\Pr[\mathbf{o}_{-1}|P(S')]\Big)$$

Substituting these inequalities into (5), we get that (5) $\leq$

$$\sup_{P(S)\sim P(S')} \frac{(1+\frac{e^2}{(1-\delta)^2(\frac{1}{\epsilon}-1)})\sum_{\mathbf{o}_{-1}\in\mathcal{F}}\sum_{i\in I^{\mathbf{o}}_{pass},i\neq 1}\left(\frac{1}{|\mathbf{o}_{-1}|-1}\Pr[\mathcal{A}_{\mathcal{O}}(S_i)\in\Omega|\mathcal{A}_{\mathcal{O}}(S_i)\neq\perp]\Pr[\mathbf{o}_{-1}|P(S)]\right)}{\sum_{\mathbf{o}_{-1}\in\mathcal{F}}\sum_{i\in I^{\mathbf{o}}_{pass},i\neq 1}\left(\frac{1}{|\mathbf{o}_{-1}|}\Pr[\mathcal{A}_{\mathcal{O}}(S'_i)\in\Omega|\mathcal{A}_{\mathcal{O}}(S'_i)\neq\perp]\Pr[\mathbf{o}_{-1}|P(S')]\right)}+\frac{\frac{\epsilon\delta e^2}{1-\delta}}{\Pr[\mathcal{A}^{prsm}_{\mathcal{O}}(\mathcal{S}')\in\Omega]}\leq$$

$$(1+\frac{e^2}{(1-\delta)^2(\frac{1}{\epsilon}-1)})\cdot\sup_{P(S)\sim P(S'),\mathbf{o}_{-1},i\neq 1}\frac{\frac{1}{|\mathbf{o}_{-1}|-1}\Pr[\mathcal{A}_{\mathcal{O}}(S_i)\in\Omega|\mathcal{A}_{\mathcal{O}}(S_i)\neq\perp]\Pr[\mathbf{o}_{-1}|P(S)]}{\frac{1}{|\mathbf{o}_{-1}|}\Pr[\mathcal{A}_{\mathcal{O}}(S'_i)\in\Omega|\mathcal{A}_{\mathcal{O}}(S'_i)\neq\perp]\Pr[\mathbf{o}_{-1}|P(S')]}+\frac{\frac{\epsilon\delta e^2}{1-\delta}}{\Pr[\mathcal{A}^{prsm}_{\mathcal{O}}(\mathcal{S}')\in\Omega]}$$

Since $S_i = S'_i$ for all $i \neq 1$, this reduces to:

$$(1+\frac{e^2}{(1-\delta)^2(\frac{1}{\epsilon}-1)})\cdot\sup_{\mathbf{o}_{-1}}\frac{|\mathbf{o}_{-1}|}{|\mathbf{o}_{-1}|-1}+\frac{\frac{\epsilon\delta e^2}{1-\delta}}{\Pr[\mathcal{A}^{prsm}_{\mathcal{O}}(\mathcal{S}')\in\Omega]}\leq(1+\frac{e^2}{(1-\delta)^2(\frac{1}{\epsilon}-1)})\frac{1}{1-\epsilon}+\frac{\frac{\epsilon\delta e^2}{1-\delta}}{\Pr[\mathcal{A}^{prsm}_{\mathcal{O}}(\mathcal{S}')\in\Omega]}$$

since $|\mathbf{o}_{-1}| \geq \frac{1}{\epsilon}$ by definition.

Following the chain of inequalities back to their genesis, we finally obtain:

$$\frac{\Pr[\mathcal{A}^{prsm}_{\mathcal{O}}(\mathcal{S})\in\Omega|P(S),\mathbf{o},\mathcal{L}]\Pr[\mathbf{o},P(S)|\mathcal{L}]}{\Pr[\mathcal{A}^{prsm}_{\mathcal{O}}(\mathcal{S}')\in\Omega]}\leq(1+\frac{e^2}{(1-\delta)^2(\frac{1}{\epsilon}-1)})\frac{1}{1-\epsilon}+\frac{\frac{\epsilon\delta e^2}{1-\delta}}{\Pr[\mathcal{A}^{prsm}_{\mathcal{O}}(\mathcal{S}')\in\Omega]},$$

which substituting into Lemma 6 gives:

$$\Pr[\mathcal{A}^{prsm}_{\mathcal{O}}(\mathcal{S})\in\Omega]\leq(1+\frac{e^2}{(1-\delta)^2(\frac{1}{\epsilon}-1)})\frac{1}{1-\epsilon}\Pr[\mathcal{A}^{prsm}_{\mathcal{O}}(\mathcal{S}')\in\Omega]+4\delta+\frac{\epsilon\delta e^2}{1-\delta}$$

For $\epsilon, \delta \leq 1/2$, $(1+\frac{e^2}{(1-\delta)^2(\frac{1}{\epsilon}-1)})\frac{1}{1-\epsilon}\leq e^{8e^2\epsilon+\epsilon+\epsilon^2}$, which establishes that **PRSMA** is $(8e^2\epsilon+\epsilon+\epsilon^2, 4\delta+\frac{\epsilon\delta e^2}{1-\delta})$ differentially private. Setting $\epsilon=\epsilon^*, \delta=\delta^*$ completes the proof. □

D Proofs from Section 5

Theorem 13. *Let $\varepsilon, \delta \in (0,1)$ and let $\mathcal{A}$ be an (ϵ,δ) differentially private pERM algorithm for query class $\mathcal{Q}$, with perturbation distribution $\mathcal{D}_{(\varepsilon,\delta)}$. Then Follow the Private Leader instantiated with $\mathcal{A}$ has expected regret bounded by:*

$$R(T)\leq O\left(\epsilon+\delta+\frac{\mathbb{E}_{Z\sim\mathcal{D}_{\varepsilon,\delta}}[\|Z\|_\infty]}{T}\right)$$

Proof. This theorem is folklore, and this proof is adapted from the lecture notes of [RS17]. We introduce some notation. First, write $\ell^t \in \mathbb{R}^{|\mathcal{Q}|}$ to denote the "loss vector" faced by the algorithm at round t, with value $q_i(x^t)$ in coordinate i. Write $\ell^{1:t}$ to denote the summed vector $\ell^{1:t} = \sum_{j=1}^{t}\ell^j$. Write $M : \mathbb{R}^d \to \mathbb{R}^d$ to denote the function such that $M(v)_{i^*} = 1$ where $i^* = \arg\min_i v_i$ and $M(v)_i = 0$ otherwise. In this notation, at each round t, "Follow the Leader" obtains loss $M(\ell^{1:t-1})\cdot\ell^t$ and

"Follow the Private Leader" obtains loss $M(\ell^{1:t-1} + Z^t) \cdot \ell^t$. At the end of play, at time T, the best query q in hindsight obtains cumulative loss $M(\ell^{1:T}) \cdot \ell^{1:T}$.

The proof of this theorem will go through a thought experiment. Consider an imaginary algorithm called "be the leader", which at round t plays according to $M(\ell^{1:t})$. We will first show that this imaginary algorithm obtains loss that is only lower than that of the best action in hindsight.

Lemma 19 ([KV05])**.**

$$\sum_{i=1}^{T} M(\ell^{1:t}) \cdot \ell^t \leq M(\ell^{1:T}) \cdot \ell^{1:T}$$

Proof. This follows by a simple induction on T. For $T = 1$, it holds with equality. Now assume it holds for general T – we show it holds for the next time step:

$$\sum_{i=1}^{T+1} M(\ell^{1:t}) \cdot \ell^t \leq M(\ell^{1:T}) \cdot \ell^{1:T} + M(\ell^{1:T+1}) \cdot \ell^{T+1} \leq M(\ell^{1:T+1}) \cdot \ell^{1:T} + M(\ell^{1:T+1}) \cdot \ell^{T+1} = M(\ell^{1:T+1}) \cdot \ell^{1:T+1}$$

□

Recall that private pERM algorithms operate by sampling a perturbation vector $Z^t \sim \mathcal{D}_{\epsilon,\delta}$ at each round. Next, we show that "be the private leader", which at round t plays according to $M(\ell^{1:t} + Z^t)$, doesn't do much worse.

Lemma 20 ([KV05])**.** *For any set of loss vectors $\ell^1, \ldots, \ell^T$ and any set of perturbation vectors $Z^0 \equiv 0, Z^1, \ldots, Z^T$:*

$$\sum_{t=1}^{T} M(\ell^{1:t} + Z^t) \cdot \ell^t \leq M(\ell^{1:T}) \cdot \ell^{1:T} + 2 \sum_{t=1}^{T} \|Z^t - Z^{t-1}\|_\infty$$

Proof. Define $\hat{\ell}^t = \ell^t + Z^t - Z^{t-1}$. Note that $\hat{\ell}^{1:t} = \ell^{1:t} + Z^t$, since the sum telescopes. Thus, we can apply Lemma 19 on the sequence $\hat{\ell}$ to conclude:

$$\begin{aligned}
\sum_{t=1}^{T} M(\ell^{1:t} + Z^t) \cdot (\ell^t + Z^t - Z^{t-1}) &\leq M(\ell^{1:T} + Z^T) \cdot (\ell^{1:T} + Z^T) \\
&\leq M(\ell^{1:T}) \cdot (\ell^{1:T} + Z^T) \\
&= M(\ell^{1:T}) \cdot \ell^{1:T} + \sum_{t=1}^{T} M(\ell^{1:T}) \cdot (Z^t - Z^{t-1})
\end{aligned}$$

Subtracting from both sides, we have:

$$\sum_{t=1}^{T} M(\ell^{1:t} + Z^t) \cdot \ell^t \leq M(\ell^{1:T}) \cdot \ell^{1:T} + \sum_{t=1}^{T} (M(\ell^{1:T}) - M(\ell^{1:t} + Z^t)) \cdot (Z^t - Z^{t-1}) \leq M(\ell^{1:T}) \cdot \ell^{1:T} + \sum_{t=1}^{T} 2\|Z^t - Z^{t-1}\|_\infty$$

□

We will use this lemma and a trick to compute the expected regret of "be the private leader". Since expectations distribute over sums, we have:

$$\mathbb{E}[\sum_{t=1}^{T} M(\ell^{1:t} + Z^t) \cdot \ell^t] = \sum_{t=1}^{T} \mathbb{E}[M(\ell^{1:t} + Z^t) \cdot \ell^t]$$

Hence, the expectation remains unchanged in the thought experiment under which the perturbation is not resampled at every step, and instead $Z^1 = \ldots = Z^t \sim \mathcal{D}_{\epsilon,\delta}$. Applying Lemma 20 to this version of be the private leader, we obtain:

$$\mathbb{E}[\sum_{t=1}^{T} M(\ell^{1:t} + Z^t) \cdot \ell^t] \leq M(\ell^{1:T}) \cdot \ell^{1:T} + 2\mathbb{E}[\|Z^1\|_\infty]$$

Finally, we use the fact that the algorithm is (ϵ, δ)-differentially private, and the difference between "follow the private leader" and "be the private leader" amounts to running a differentially private algorithm on one of two datasets. For any (ϵ, δ) differentially private algorithm $\mathcal{A} : \mathcal{X}^* \to R$, for any function $f : R \to [0, T]$, and for any pair of neighboring datasets S, S' we have that:

$$\mathbb{E}[f(\mathcal{A}(S))] \leq e^\epsilon \mathbb{E}[f(\mathcal{A}(S'))] + \delta T.$$

We can therefore conclude that for each t:

$$\mathbb{E}[M(\ell^{1:t-1} + Z^t) \cdot \ell^t] \leq e^\epsilon \mathbb{E}[M(\ell^{1:t} + Z^t) \cdot \ell^t] + \delta t$$

Combining this bound with the regret bound we have proven for "be the private leader" yields:

$$\sum_{t=1}^{T} \mathbb{E}[M(\ell^{1:t-1} + Z^t) \cdot \ell^t] \leq e^\epsilon \mathbb{E}[\sum_{t=1}^{T} M(\ell^{1:t} + Z^t) \cdot \ell^t] + \delta T \leq e^\epsilon \left(M(\ell^{1:T}) \cdot \ell^{1:T} + 2\mathbb{E}[\|Z^1\|_\infty]\right) + \delta T \leq$$

$$(1 + 2\epsilon)\left(M(\ell^{1:T}) \cdot \ell^{1:T} + 2\mathbb{E}[\|Z^1\|_\infty]\right) + \delta T \leq M(\ell^{1:T}) \cdot \ell^{1:T} + (2 + 4\epsilon)\mathbb{E}[\|Z^1\|_\infty] + 2\epsilon T + \delta T$$

Dividing by T yields the theorem. □

Theorem 15. *For any d, there is a fixed finite class of statistical queries $\mathcal{Q}$ of size $|\mathcal{Q}| = N = 2^d$ defined over a data universe of size $|\mathcal{X}| = O(N^5 \log^2 N)$ such that for every online learning algorithm with access to a weighted optimization oracle for $\mathcal{Q}$, it cannot guarantee that its expected average regret will be $o(1)$ in total time less than $\Omega(\sqrt{N}/\log^3(N))$.*

Proof. We start by quoting the main ingredient proven in [HK16] that goes into their lower bound:

Theorem 21 ([HK16] Theorem 4)**.** *For every $N = 2^d$, and for every randomized algorithm for the players in the game with access to a best-response oracle, there is an $N \times N$ game with payoffs taking values in $\{0, 1/4, 3/4, 1\}$ such that with probability $2/3$ the players have not converged to a $1/4$-approximate min-max equilibrium until at least $\Omega(\sqrt{N}/\log^3(N))$ time.*

We start by using the Yao min-max principle to reverse the order of quantifiers: Theorem 21 also implies that there is a fixed *distribution* over $N \times N$ games that is hard in expectation for *every*

algorithm. However, because we will be interested in the support size of this distribution, we go into a bit more detail in how we apply the min-max principle.

Consider a "meta game" defined by an (infinite) matrix M, with rows i indexed by the $L = 4^{N^2}$ $N \times N$ zero-sum games G_i taking values in $\{0, 1/4, 3/4, 1\}$, and columns indexed by algorithms A_j instantiated with best-response oracles designed to play zero-sum games. $M(i, j)$ will encode the expected running time before algorithm A_j when used to play game G_i converges to a value that is within $1/4$ of the equilibrium value of game G_i. Let the row player (the "lower bound" player) be the maximization player in the zero sum game defined by M, and let the column player (the "algorithm player") be the minimization player. As stated, the entries in M can take unboundedly large values — but observe that there is a simple modification to the game that allows us to upper bound the entries in M by $O(N \log N)$. This is because there exists an algorithm A (the multiplicative weights algorithm — see e.g. [AHK12]) that can be used to play any $N \times N$ game and converge to a 1/4-approximate equilibrium after time at most $O(N \log N)$ (running in time $O(N)$ per iteration for $O(\log N)$ iterations). It is also possible to check whether a pair of distributions form a 1/4 approximate equilibrium with two calls to a best response oracle. Hence, we can take any algorithm A_i and modify it so that it converges to a $1/4$-approximate equilibrium after at most $O(N \log N)$ time. We simply halt the algorithm after $O(N \log N)$ time if it has not yet converged, and run the multiplicative weights algorithm A. Theorem 21 implies that the value of this modified game M is at least $\Omega(\sqrt{N}/\log^3(N))$.

We now observe that the "meta-game" M has an $O(1)$-approximate max-min strategy for the lower bound player that has support size at most $O(N^4 \log^2 N)$. To see this, consider the following constructive approach to computing an approximate equilibrium: simulate play of the game in rounds. Let the lower bound player sample a game G^t at each round t using the multiplicative weights distribution over her L actions, and let the algorithm player best respond at each round to the lower bound player's distribution. By construction, the algorithm player has 0 regret, whereas the lower bound player has regret $O(\sqrt{\log L/T} \cdot N \log N)$ after T rounds (since the entries of M are bounded between 0 and $O(N \log N)$) This corresponds to $O(1)$ regret after $T = O(\log L \cdot N^2 \log^2 N) = O(N^4 \log^2 N)$ many rounds. By Theorem 11, the empirical distribution over these $O(N^4 \log^2 N)$ many games G^t forms an $O(1)$-approximate max-min strategy. Thus we have proven:

Corollary 4. *For every $N = 2^d$, there is a fixed set $H \subseteq \{0, 1/4, 3/4, 1\}^{N \times N}$ of $N \times N$ games, of size $|H| = O(N^4 \log^2 N)$ such that for every randomized algorithm A for players in a game with access to a best response oracle, there is a game $G \in H$ such that with probability $2/3$, the players have not converged to a $1/4$-approximate min-max equilibrium until at least $\Omega(\sqrt{N}/\log^3(N))$ time.*

Now consider the $N \times O(N^5 \log^2 N)$ matrix R that results from stacking the matrices in H. Identify the N rows with a query class $\mathcal{Q}$ of size N, indexed by functions $q \in \mathcal{Q}$. Identify the columns with a data universe $\mathcal{X}$ of size $|\mathcal{X}| = O(N^5 \log^2 N)$ indexed by $x \in \mathcal{X}$, and define the queries such that $q(x) = R(q, x)$ for each $q \in \mathcal{Q}, x \in \mathcal{X}$. Observe that any no-regret algorithm with action set $\mathcal{Q}$ that can obtain $o(1)$ regret against an adversary who is constrained to play loss vectors in $\mathcal{X}$ can be used to compute an $o(1)$ approximate equilibrium strategy for any game in H (together with a single call to a best-response oracle per round by his oppoinent, by Theorem 11). Thus we can conclude that no algorithm can guarantee to get $o(1)$ regret over $\mathcal{Q}$ until at least $\Omega(\sqrt{N}/\log^3(N))$ time. □

Theorem 16. *Any oracle efficient (i.e. running in time poly$(n, \log|\mathcal{Q}|)$) (ϵ, δ)-differentially private pERM algorithm instantiated with a weighted optimization oracle for the query class $\mathcal{Q}$ defined in The-*

orem 15, with perturbation distribution $\mathcal{D}_{(\varepsilon,\delta)}$ must be such that for every $(\epsilon+\delta)=o(1)$:

$$\mathbb{E}_{Z\sim\mathcal{D}_{(\varepsilon,\delta)}}[\|Z\|_\infty]\geq\Omega(|\mathcal{Q}|^c)$$

for some constant $c>0$.

Proof. For any such pERM algorithm $\mathcal{A}$, Let $B=\mathbb{E}_{Z\sim\mathcal{D}_{(\varepsilon,\delta)}}[\|Z\|_\infty]$. We know from Theorem 13 that follow the private leader instantiated with $\mathcal{A}$ obtains regret $o(1)$ whenever $T=\omega(B)$, for any $\epsilon+\delta=o(1)$. Since $\mathcal{A}$ is oracle efficient (i.e. runs in time $\text{poly}(t,\log|\mathcal{Q}|)$), the total running time needed to obtain diminishing regret is $\sum_{t=1}^{T}\text{poly}(t,\log|\mathcal{Q}|)=\text{poly}(B,\log|\mathcal{Q}|)$. We know from Theorem 15 that to guarantee diminishing regret over $\mathcal{Q}$, the total running time must be at least $\Omega(\sqrt{|\mathcal{Q}|}/\log^3(|\mathcal{Q}|))$. Thus we must have that $B=\text{poly}(|\mathcal{Q}|)$ — i.e. $B=\Omega(|Q|^c)$ for some $c>0$. □