UNIVERSITY OF CALIFORNIA SAN DIEGO

Algorithmic Stability for Responsible Computing

A dissertation submitted in partial satisfaction of the
requirements for the degree Doctor of Philosophy

in

Computer Science

by

Jessica Sorrell

2022

University of California San Diego

2022

DEDICATION

This work is dedicated to my father, Bruce Sorrell, my mother, Gaynor Sorrell, and my sister, Melissa Sorrell.

EPIGRAPH

“Some things you’ll do for money, and some you’ll do for fun, but the things you do for love are going to come back to you one by one.”

- The Mountain Goats

TABLE OF CONTENTS

LIST OF ALGORITHMS

ACKNOWLEDGEMENTS

Thank you to my advisors, Russell Impagliazzo and Daniele Micciancio. I am grateful for their guidance, patience, and encouragement, without which this dissertation would not have been possible.

Thank you to my collaborators Matilda Backendal, Mihir Bellare, Mark Bun, Marco Carmosino, Ilias Diakonikolas, Daniel Kane, Baiyu Li, Toniann Pitassi, Mark Schultz, Jiahao Sun, and Christos Tzamos. I learned a great deal from all of you.

Thank you to my friends and colleagues Vivek Arte, Daniel Beaglehole, Michael Borkowski, Wei Dai, Hannah Davis, Anant Dhayal, Greg Fields, Jiawei Gao, Nicholas Genise, Ken Hoover, Max Hopkins, Joseph Jaeger, Amy Kanne, Shauna Kravec, Lane Lawley, Guarav Mahajan, Nathaniel Mathews, Sam McGuire, Ivan Mikhailin, Sasank Mouli, Tamalika Mukherjee, Ruth Ng, Anthony Ostuni, Laura Shea, Adam Suhl, Igors Stepanovs, Michael Walter, Psi Vesely, and the many others omitted. It has been a joy to work alongside you all.

Thank you to my committee members Mihir Bellare, Jelena Bradic, and Kamalika Chaudhuri for their guidance and thoughtful discussions over the years.

Thank you to those who helped me find my way after many years of being lost: Anurag Agarwal, Alex Avila, David Barth-Hart, Nick Bryan, Rick Cheehy, Anna Fiorucci, Bill Harris, and David Ross. Thank you for believing in me.

I cannot fully express in this document the gratitude I feel for my family. Thank you for your patience, love, and support. I know I did not make it easy.

Chapter 2, in full, is a reprint of the material as it appears in Conference on Learning Theory 2022. Bun, Mark; Carmosino, Marco; Sorrell, Jessica. “Efficient, Noise-Tolerant, and Private Learning via Boosting”. The dissertation author was a primary investigator and author of this material.

Chapter 3, in full, has been submitted for publication of the material as it may appear in Advances in Cryptology - CRYPTO 2022. Li, Baiyu; Micciancio, Daniele; Schultz, Mark; Sorrell, Jessica. “Securing Approximate Homomorphic Encryption Using Differential Privacy”. The dissertation author was a primary investigator and author of this material.

Chapter 4, in full, is based on the material as it appears in Symposium on Theory of Computing. Impagliazzo, Russell; Lei, Rex; Pitassi, Toniann; Sorrell, Jessica. “Reproducibility in Learning”. The dissertation author was a primary investigator and author of this material.

VITA

2015	Bachelor of Science, Applied Mathematics, Rochester Institute of Technology
2020	Master of Science, Computer Science, University of California San Diego
2022	Doctor of Philosophy, Computer Science, University of California San Diego

ABSTRACT OF THE DISSERTATION

Algorithmic Stability for Responsible Computing

by

Jessica Sorrell

Doctor of Philosophy in Computer Science

University of California San Diego, 2022

Professor Russell Impagliazzo, Co-Chair
Professor Daniele Micciancio, Co-Chair

Algorithmic stability is a measure of how much the output of an algorithm changes in response to small changes to its inputs. Various notions of stability give rise to desirable algorithmic properties, such as generalization in the case of uniform stability for learning algorithms. Another notion of stability, differential privacy, guarantees that the output of an algorithm will not change too much when any one element of its input sample is exchanged for another. This notion ensures that the output of an algorithm cannot “leak” too much information about any given element of its input, ensuring privacy for individuals who may contribute their data to the input of the algorithm.

This dissertation develops new stable algorithms for promoting reliable and secure computation. We develop a new framework for generically constructing differentially private learning algorithms via boosting. We show how to use a variant of the standard notion of differential privacy to achieve stronger security guarantees for approximate fully homomorphic encryption. Finally, we develop a new stability notion for randomized algorithms called reproducibility, which guarantees that the output of an algorithm remains unchanged with high probability when its input is entirely redrawn from the same underlying distribution. We design algorithms for fundamental statistical tasks that achieve this new notion, and explore connections to related notions of stability.

Introduction

Applications necessitating the collection and analysis of large quantities of data are proliferating, from training neural networks to assist with medical diagnostics to wastewater surveillance for tracking the spread of COVID-19. When we deploy algorithms for these applications, we have a (sometimes legal) obligation to manage collected data responsibly and to guarantee our algorithms meet all desiderata arising from the context of the application. Such guarantees are especially critical in high-stakes algorithm-assisted decision-making, or when computing on sensitive personal data, but they can be challenging to provide given the complexity of the algorithms and models deployed for these purposes.

This dissertation studies the design of algorithms that enable responsible use of data via provable guarantees of broadly desirable properties. In particular, we focus on the properties of privacy, security, and reproducibility. Privacy establishes that an individual will not have their information shared, in whole or in part, with other parties without the consent of that individual. Security guarantees that our data is not subject to unauthorized access by potentially malicious parties. Reproducibility ensures that the output of an algorithm is actually capturing the relevant features of the data on which it was run, rather than spurious correlations in that data.

These varied properties address different concerns with the responsible use and storage of data, but the techniques used to obtain them are in fact quite similar. Each of the algorithms presented in this dissertation achieves its provable guarantees via some notion of stability. An algorithm is stable if its output does not change too much given small changes to its input. In this dissertation, we study several notions of stability and their applications in trustworthy computing.

Differential Privacy.

The notions of privacy adopted throughout this dissertation are all variants of differential privacy. Differential privacy was first introduced by Dwork et al. [2006] to formalize the requirement that an algorithm run on a database should not reveal too much information about any individual row of that database. In other words, the output of a differentially private algorithm should not look too different when your data is included in its input compared to when it is excluded. These algorithms are by necessity randomized, and so the informal aim that the output not "look too different" is made rigorous by requiring that the distributions over outputs of the algorithm should be close in statistical distance when a single row is replaced with a new one.

Fully-Homomorphic Encryption.

Differential privacy as described above is a valuable notion for capturing privacy concerns arising from published findings. It guarantees that even if the result of some computation is made public, whether that result is a language model or the conclusion of a scientific study, that these results should not reveal whether or not your data was included in the computation. However, there are other privacy concerns relevant to data sharing that it cannot address.

Consider the case of genome-wide association studies (GWAS), which study large samples of genomic data to identify correlations between genetic variants and observable traits (e.g., a disease). Such studies have the potential to significantly improve our understanding of disease genetics, but they require collection of whole genome data from individuals who may be understandably reluctant to share this information with another party. *Fully-homomorphic encryption* (FHE) is a cryptographic tool tailored to preserving privacy in these settings. FHE schemes satisfy standard cryptographic notions of security, but are equipped with additional subroutines that allow an untrusted party to carry out computation on encrypted data. The resulting ciphertexts may then later be decrypted to obtain the result of the computation, without the untrusted party ever having access to the plaintext data. In the case of GWAS studies, FHE would enable study participants to encrypt genomic data and upload it to a cloud database, on which GWAS studies could then be performed homomorphically.

FHE is not itself an algorithmic stability notion, but in this dissertation we show how to use techniques from differential privacy to secure one class of FHE schemes. For this class of schemes, we show that making the decryption function more stable (with respect to one feature of its input) allows the scheme to satisfy a stronger and more useful definition of security.

Reproducibility.

Reproducibility was introduced in Impagliazzo et al. [2022] as a formal property of randomized algorithms. An algorithm is *reproducible* if running the algorithm on two different samples from the same distribution produces the same result with high probability. Informally, this property guarantees that an algorithm is not sensitive to resampling, and so results obtained using one data set can be reproduced with high probability on other training sets from the same population. This ensures, for example, that published results can be verified by other researchers using their own data sets. In this dissertation, we present our definition of reproducibility and gave the first reproducible algorithms for fundamental tasks in machine learning.

Organization.

In Chapter 1, we introduce the technical preliminaries that we will use throughout this work.

In Chapter 2, we develop a framework for generically constructing private and noise tolerant learning algorithms via boosting. Boosting is a commonly-adopted approach in machine learning for improving the accuracy of a weak learner [1] in order to obtain a classifier with arbitrarily low error. Our intuition tells us that it should be much easier to construct a weak learner for a given classification task than a highly accurate one, but boosting algorithms show us that these objectives are in fact equivalent.

Beyond mere accuracy, what other properties might we want from a boosting algorithm? When using boosting to train a classifier on sensitive information, we may also want to guarantee that the output of our boosting algorithm does not reveal too much information about any single data point in its training set. This specification is formalized by the notion of *differential privacy*. In addition, due to the practical challenges of obtaining accurately labeled data, we may also want to guarantee

[1] A weak learner is only required to output a classifier that is somewhat correlated with the true labels of the data.

that our classifier will still have low error even if some of its training data has been mislabeled. In these cases, it is not sufficient to construct a noise-tolerant or differentially private weak learner and apply a generic boosting algorithm to improve its accuracy. Boosting algorithms will typically fail to preserve these additional desirable properties of their weak learners, and so new algorithmic ideas are required to guarantee that classifiers obtained by boosting satisfy their specifications.

Our framework for designing boosting algorithms preserves differential privacy of their weak learners. We give natural and sufficient conditions under which a boosting algorithm can be made private, and demonstrate the usefulness of our framework by applying it to the task of privately learning large-margin halfspaces. Our boosted learner matches the sample complexity of the state of the art algorithm for the same problem Nguyễn et al. [2019], while also tolerating random classification noise [2]. The material presented in this chapter originally appeared in Bun et al. [2020a].

In Chapter 3, we show how to use a variant of the standard notion of differential privacy to achieve stronger security guarantees for approximate fully homomorphic encryption. The primary barrier to adoption of FHE in practice is its current inefficiency. All known FHE schemes induce a significant computational overhead that makes them less appealing in practice than they might be given their numerous applications in secure computing. At a very high level, this overhead follows from the fact that known constructions of FHE are "noisy," that is their security relies on adding noise to the plaintext message during encryption. If this noise is small, then decryption will recover the correct message with high probability, but homomorphic computations cause this error to grow. To perform arbitrary homomorphic computations, an additional step must be introduced to reduce the error of intermediate ciphertexts, otherwise the results of the homomorphic computation will be incorrect. This additional step is known as *bootstrapping* and is typically the main source of inefficiency in FHE schemes.

One approach to improving the efficiency of FHE is to study *approximate FHE schemes*. Approximate FHE schemes adopt a weaker notion of correctness that only guarantees the decrypted

[2]Random classification noise is a model of label noise for binary classification in which every element of the data domain has the same probability η of having its label flipped.

result of homomorphic computation is approximately correct. Such a relaxation of correctness may be sufficient for applications that only require bounded precision, and allowing modest amounts of noise to accumulate during computation reduces the frequency with which bootstrapping is required. The first approximate FHE scheme was introduced in Cheon et al. [2017], and has garnered much interest from FHE researchers. Unfortunately, the FHE scheme introduced in Cheon et al. [2017] does not satisfy reasonable notions of security that we should expect in FHE applications, and attacks on this scheme were demonstrated in Li and Micciancio [2021]. In Chapter 3, we show that techniques from differential privacy can be used to secure general approximate FHE schemes. In particular, we show how to use the Gaussian mechanism to secure the FHE scheme of Cheon et al. [2017], alongside a lower bound proving that our parameters for securing their scheme are almost tight. The material presented in this chapter originally appeared in Li et al. [2022].

In Chapter 4, we develop a new stability notion for randomized algorithms called *reproducibility*. Reproducibility guarantees that the output of an algorithm remains unchanged with high probability when its input is entirely redrawn from the same underlying distribution. We design algorithms for fundamental statistical tasks that achieve this new notion, and explore connections to related notions of stability. The material presented in this chapter originally appeared in Impagliazzo et al. [2022].

Chapter 1

Preliminaries

In this chapter, we introduce definitions and lemmas that will be used throughout this dissertation.

1.1 Probability

We will frequently rely on the following notions of similarity between distributions.

Definition 1.1.1 (Kullback-Leibler Divergence)**.** Let μ_1 and μ_2 be bounded measures over the same domain X. The *Kullback-Leibler divergence* between μ_1 and μ_2 is defined as:

$$\mathrm{KL}(\mu_1 \parallel \mu_2) = \sum_{x \in X} \mu_1(x) \log\left(\frac{\mu_1(x)}{\mu_2(x)}\right) + \mu_2(x) - \mu_1(x)$$

Definition 1.1.2 (Statistical Distance)**.** The *statistical distance* between two distributions Y and Z, denoted $\Delta(Y,Z)$, is defined as:

$$\Delta(Y,Z) = \max_S |\mathbf{Pr}[Y \in S] - \mathbf{Pr}[Z \in S]|$$

The α-Rényi divergence has a parameter $\alpha \in (1,\infty)$ which allows it to interpolate between KL-divergence at $\alpha = 1$ and max-divergence at $\alpha = \infty$.

Definition 1.1.3 (Rényi Divergence)**.** Let P and Q be probability distributions on Ω. For $\alpha \in (1,\infty)$, we define the Rényi Divergence of order α between P and Q as:

$$D_\alpha(P \parallel Q) = \frac{1}{\alpha - 1} \log\left(\mathbf{E} x \sim Q\left(\frac{P(x)}{Q(x)}\right)^{\alpha}\right)$$

In Chapter 3, we will need the following properties and notation for KL divergence.

Lemma 1.1.4 (Properties of the KL Divergence, Theorem 2.2 of Polyanskiy and Wu [2014]). *The KL divergence satisfies*

1. Sub-Additivity for Joint Distributions*: If $(\mathscr{X}_0, \mathscr{X}_1)$ and $(\mathscr{Y}_0, \mathscr{Y}_1)$ are pairs of (possibly dependent) random variables, then*

$$\begin{aligned} D((\mathscr{X}_0, \mathscr{X}_1)||(\mathscr{Y}_0, \mathscr{Y}_1)) &\leq \mathbb{E}_{x \sim \mathscr{X}_0}[D((\mathscr{X}_1 \mid x)||(\mathscr{Y}_1 \mid x))] + D(\mathscr{X}_0||\mathscr{Y}_0) \\ &\leq \max_x D((\mathscr{X}_1 \mid x)||(\mathscr{Y}_1 \mid x)) + D(\mathscr{X}_0||\mathscr{X}_1), \end{aligned}$$

2. Data Processing Inequality*: For any (potentially randomized) function f, for any two distributions $\mathscr{P}, \mathscr{Q}$, $D(f(\mathscr{P})||f(\mathscr{Q})) \leq D(\mathscr{P}||\mathscr{Q})$, and*

3. Pinsker's Inequality*: $\Delta(\mathscr{P}, \mathscr{Q}) \leq \sqrt{D(\mathscr{P}||\mathscr{Q})/2}$.*

We introduce the following notation to more compactly bound the divergence between vectors of random variables.

Definition 1.1.5. Let $\mathscr{X} = (\mathscr{X}_i)_{i=1}^n, \mathscr{Y} = (\mathscr{Y}_i)_{i=1}^n$ be two lists of discrete random variables over the support $\prod_{i=1}^n X_i \subseteq \mathbb{R}^n$, and δ any divergence. We define the *vector divergence* $\hat{\delta}(\mathscr{X}||\mathscr{Y})$ to be the non-negative real vector $(v_1, \ldots, v_n) \in \mathbb{R}^n_{\geq 0}$ with coordinates $v_i = \max_a \delta([\mathscr{X}_i \mid \mathscr{X}_{<i} = a]||[\mathscr{Y}_i \mid \mathscr{Y}_{<i} = a])$.

In this notation, sub-additivity of the KL divergence (for example) can be written as $D(\mathscr{X}||\mathscr{Y}) \leq \|\widehat{D}(\mathscr{X}||\mathscr{Y})\|_1$.

Our lower bound of Section 3.4.3 will require the following bound on statistical distance between centered Gaussians.

Lemma 1.1.6 (Theorem 1.3 Devroye et al. [2018])**.** *Let $\sigma_0, \sigma_1 > 0$. Then*

$$\Delta(\mathcal{N}(0,\sigma_0^2), \mathcal{N}(0,\sigma_1^2)) \geq \frac{1}{200} \min\left\{1, \frac{|\sigma_0^2 - \sigma_1^2|}{\sigma_0^2}\right\}. \tag{1.1}$$

1.2 Learning

We work in the **P**robably **A**pproximately **C**orrect (**PAC**) setting [Valiant, 1984b]. Our learning algorithms *probably* learn a hypothesis which *approximately* agrees with an unknown target concept. We denote by $\mathscr{X}$ the domain of examples, and for the remainder of this dissertation consider only the Boolean classification setting where labels are always ± 1.

Definition 1.2.1 (PAC Learning)**.** A hypothesis class $\mathscr{H}$ is (α, β)*-PAC learnable* if there exists a sample bound $n_{\mathscr{H}} : (0,1)^2 \to \mathbb{N}$ and a learning algorithm $\mathscr{A}$ such that: for every $\alpha, \beta \in (0,1)$ and for every distribution D over $\mathscr{X} \times \{\pm 1\}$, running $\mathscr{A}$ on $n \geq n_{\mathscr{H}}(\alpha, \beta)$ i.i.d. samples from D will with probability at least $(1 - \beta)$ return a hypothesis $h : \mathscr{X} \to \{\pm 1\}$ such that:

$$\mathbf{Pr}_{(x,y)\sim D}\left[h(x) = y\right] \geq 1 - \alpha.$$

PAC learners guarantee strong generalization to unseen examples. Several of the algorithms presented in this dissertation will construct PAC learners by boosting weak learners — which need only beat random guessing on any distribution over the training set.

Definition 1.2.2 (Weak Learning)**.** Let $S \subset (\mathscr{X} \times \{\pm 1\})^n$ be a training set of size n. Let D be a distribution over $[n]$. A *weak learning algorithm* with *advantage* γ takes (S, D) as input and outputs a function $h : \mathscr{X} \to [-1, 1]$ such that:

$$\frac{1}{2} \sum_{j=1}^{n} D(j) |h(x_j) - y_j| \leq \frac{1}{2} - \gamma$$

1.3 Privacy

Two datasets $S, S' \in X^n$ are said to be *neighboring* (denoted $S \sim S'$) if they differ by at most a single element. Differential privacy requires that analyses performed on neighboring datasets have "similar" outcomes. Intuitively, the presence or absence of a single individual in the dataset should not impact a differentially private analysis "too much." We formalize this below.

Definition 1.3.1 (Differential Privacy)**.** A randomized algorithm $\mathscr{M} : X^n \to \mathscr{R}$ is (ε, δ)-differentially private if for all measurable $T \subseteq \mathscr{R}$ and all neighboring datasets $S \sim S' \in X^n$, we have

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

In our analyses, it will sometimes be more useful to work with the notion of (zero-)concentrated differential privacy, which bounds higher moments of privacy loss than normal differential privacy.

Definition 1.3.2 (Zero Concentrated Differential Privacy (zCDP))**.** A randomized algorithm $\mathscr{M} : X^n \to \mathscr{R}$ satisfies ρ-zCDP if for all neighboring datasets $S \sim S' \in X^n$ and all $\alpha > 1$, we have $D_\alpha(\mathscr{M}(S) \parallel \mathscr{M}(S')) \leq \rho\alpha$, where $D_\alpha(\cdot \parallel \cdot)$ denotes the Rényi divergence of order α.

This second notion will often be more convenient to work with, because it tightly captures the privacy guarantee of Gaussian noise addition and of composition:

Lemma 1.3.3 (Tight Composition for zCDP, Bun and Steinke [2016])**.** *If $\mathscr{M}_1 : X^n \to \mathscr{R}_1$ satisfies ρ_1-zCDP, and $\mathscr{M}_2 : (X^n \times \mathscr{R}_1) \to \mathscr{R}_2$ satisfies ρ_2-zCDP, then the composition $\mathscr{M} : X^n \to \mathscr{R}_2$ defined by $\mathscr{M}(S) = \mathscr{M}_2(S, \mathscr{M}_1(S))$ satisfies $(\rho_1 + \rho_2)$-zCDP.*

zCDP can be converted into a guarantee of (ε, δ)-differential privacy.

Lemma 1.3.4 (zCDP $\implies$ DP, Bun and Steinke [2016])**.** *Let $\mathscr{M} : X^n \to \mathscr{R}$ satisfy ρ-zCDP. Then for every $\delta > 0$, we have that $\mathscr{M}$ also satisfies (ε, δ)-differential privacy for $\varepsilon = \rho + 2\sqrt{\rho \log(1/\delta)}$.*

The following lemma will let us bound Rényi divergence between related Gaussians:

Lemma 1.3.5 (Rényi Divergence Between Spherical Gaussians; folklore, see Bun and Steinke [2016])**.** *Let* $z, z' \in \mathbb{R}^d$*,* $\sigma \in \mathbb{R}$*, and* $\alpha \in [1, \infty)$*. Then*

$$D_\alpha(\mathcal{N}(z, \sigma^2 I_d) \parallel \mathcal{N}(z', \sigma^2 I_d)) = \frac{\alpha \|z - z'\|_2^2}{2\sigma^2}.$$

Finally, ρ-zCDP is closed under post-processing, just like standard DP.

Lemma 1.3.6 (Post-processing zCDP, Bun and Steinke [2016])**.** *Let* $\mathcal{M} : X^n \to R_1$ *and* $f : R_1 \to R_2$ *be randomized algorithms. Suppose* $\mathcal{M}$ *satisfies* ρ*-zCDP. Define* $\mathcal{M}' : X^n \to R_2$ *by* $\mathcal{M}'(x) = f(M(x))$*. Then* $\mathcal{M}'$ *satisfies* ρ*-zCDP.*

We now state the generalization properties of differentially private algorithms that select statistical queries, which count the fraction of examples satisfying a predicate. Let X be an underlying population, and denote by D a distribution over X.

Definition 1.3.7 (Statistical Queries)**.** A statistical query q asks for the expectation of some function on random draws from the underlying population. More formally, let $q : X \to [0, 1]$ and then define the statistical query *based on q* (abusing notation) as the following, on a sample $S \in X^n$ and the population, respectively:

$$q(S) = \frac{1}{|S|} \sum_{x \in S} q(x) \qquad \text{and} \qquad q(D) = \mathbf{E} x \sim D q(x)$$

In the case of statistical queries, dependence of accuracy on the sample size is good enough to obtain interesting sample complexity bounds from privacy alone. These transfer theorems have recently been improved, bringing "differential privacy implies generalization" closer to practical utility by decreasing the very large constants from prior work [Jung et al., 2019]. As our setting is asymptotic, we employ a very convienent (earlier) transfer lemma of Bassily et al. [2016].

Theorem 1.3.8 (Privacy $\implies$ Generalization of Statistical Queries, Bassily et al. [2016])**.** *Let* $0 < \varepsilon < 1/3$*, let* $0 < \delta < \varepsilon/4$ *and let* $n \geq \log(4\varepsilon/\delta)/\varepsilon^2$*. Let* $\mathcal{M} : X^n \to Q$ *be* (ε, δ)*-differentially* *private, where Q is the set of statistical queries $q : X \to \mathbb{R}$. Let D be a distribution over X, let $S \leftarrow_{R} D^n$ be an i.i.d. sample of n draws from D, and let $q \leftarrow_{R} \mathscr{M}(S)$. Then:*

$$\mathbf{Pr}_{q,S}\left[|q(S) - q(D)| \geq 18\varepsilon\right] \leq \delta/\varepsilon.$$

Chapter 2

Private Boosting

2.1 Introduction

Boosting is a fundamental technique in both the theory and practice of machine learning for converting weak learning algorithms into strong ones. Given a sample S of n labeled examples drawn i.i.d. from an unknown distribution, a weak learner is guaranteed to produce a hypothesis that can predict the labels of fresh examples with a noticeable advantage over random guessing. The goal of a boosting algorithm is to convert this weak learner into a strong learner: one which produces a hypothesis with classification error close to zero.

A typical boosting algorithm — e.g., the AdaBoost algorithm of Freund and Schapire [1997] — operates as follows. In each of rounds $t = 1, \ldots, T$, the boosting algorithm selects a distribution D_t over S and runs the weak learner on S weighted by D_t, producing a hypothesis h_t. The history of hypotheses $h_1, \ldots, h_t$ is used to select the next distribution D_{t+1} according to some update rule (e.g., the multiplicative weights update rule in AdaBoost). The algorithm terminates either after a fixed number of rounds T, or when a weighted majority of the hypotheses $h_1, \ldots, h_T$ is determined to have sufficiently low error.

In many situations, it is desirable for the distributions D_t to be *smooth* in the sense that they do not assign too much weight to any given example, and hence do not deviate too significantly from the uniform distribution. This property is crucial in applications of boosting to noise-tolerant learning [Domingo and Watanabe, 2000, Servedio, 2003b], differentially private learning [Dwork et al., 2010], and constructions of hard-core sets in complexity theory [Impagliazzo, 1995, Barak et al., 2009a]. Toward the first of these applications, Servedio [2003b] designed a smooth boosting algorithm (`SmoothBoost`) suitable for PAC learning in spite of malicious noise. In this model of learning, up to an η fraction of the sample S could be corrupted in an adversarial fashion before being presented to the learner [Valiant, 1985]. Smooth boosting enables a weak noise-tolerant learner to be converted into a strong noise-tolerant learner. Intuitively, the smoothness property is necessary to prevent the weight placed on corrupted examples in S from exceeding the noise-tolerance of the weak learner. The round complexity of smooth boosting was improved by Barak et al. [2009a] to match that of the AdaBoost algorithm by combining the multiplicative weights update rule with Bregman projections onto the space of smooth distributions.

Smoothness is also essential in the design of boosting algorithms which guarantee *differential privacy* [Dwork et al., 2006], a mathematical definition of privacy for statistical data analysis. Kasiviswanathan et al. [2011] began the systematic study of PAC learning with differential privacy. Informally, a (randomized) learning algorithm is differentially private if the distribution on hypotheses it produces does not depend too much on any one of its input samples. Again, it is natural to design "private boosting" algorithms which transform differentially private weak learners into differentially private strong learners. In this context, smoothness is important for ensuring that each weighted input sample does not have too much of an effect on the outcomes of any of the runs of the weak learner. A private smooth boosting algorithm was constructed by Dwork et al. [2010], who augmented the AdaBoost algorithm with a private weight-capping scheme which can be viewed as a Bregman projection.

2.1.1 Contributions

Simple and Modular Private Boosting.

Our main result is a framework for private boosting which simplifies and generalizes the private boosting algorithm of Dwork et al. [2010]. We obtain these simplifications by sidestepping a technical issue confronted by Dwork et al. [2010]. Their algorithm maintains two elements of state from round to round: the history of hypotheses $H = h_1, \ldots, h_t$ and auxiliary information regarding each previous distribution $D_1, \ldots, D_t$, which is used to enforce smoothness. They remark: "[this algorithm] raises the possibility that adapting an existing or future smooth boosting algorithm to preserve privacy might yield a simpler algorithm."

We realize exactly this possibility by observing that most smooth boosting algorithms have effectively *stateless* strategies for re-weighting examples at each round. By definition, a boosting algorithm must maintain some history of hypotheses. Therefore, re-weighting strategies that can be computed using only the list of hypotheses require no auxiliary information. Happily, most smooth boosting algorithms define such hypothesis-only re-weighting strategies. Eliminating auxiliary state greatly simplifies our analysis, and implies natural conditions under which existing smooth boosting algorithms could be easily privatized.

Our main algorithm is derived from that of Barak et al. [2009a], which we call `BregBoost`. Their algorithm alternates between mutiplicative re-weighting and Bregman projection: the multiplicative update reflects current performance of the learner, and the Bregman projection ensures that `BregBoost` is smooth. Unfortunately, a naïve translation of `BregBoost` into our framework would Bregman project more than once per round. This maintains correctness, but ruins privacy. Inspired by the private optimization algorithms of Hsu et al. [2013, 2014] we give an alternative analysis of `BregBoost` that requires only a *single* Bregman projection at each round. The need for "lazy" Bregman projections emerges naturally by applying our template for private boosting to `BregBoost`, and results in a private boosting algorithm with optimal round complexity: `LazyBregBoost`. This method of lazy projection [see Rakhlin, 2009, for an exposition] has appeared in prior works about differential privacy [Hsu et al., 2013, 2014], but not in the context of designing boosting algorithms.

Application: Privately Learning Large-Margin Halfspaces.

A halfspace is a function $f : \mathbb{R}^d \to \{-1, 1\}$ of the form $f(x) = \operatorname{sign}(u \cdot x)$ for some vector $u \in \mathbb{R}^d$. Given a distribution D over the unit ball in $\mathbb{R}^d$, the *margin* of f with respect to D is the infimum of $|u \cdot x|$ over all x in the support of D. Learning large-margin halfspaces is one of the central problems of learning theory. A classic solution is given by the Perceptron algorithm, which is able to learn a τ-margin halfspace to classification error α using sample complexity $O(1/\tau^2\alpha)$

independent of the dimension d. Despite the basic nature of this problem, it was only in very recent work of Nguyễn et al. [2019] that dimension-independent sample complexity bounds were given for *privately* learning large-margin halfspaces. In that work, they designed a learning algorithm achieving sample complexity $\tilde{O}(1/\tau^2\alpha\varepsilon)$ for τ-margin halfspaces with $(\varepsilon, 0)$-differential privacy, and a computationally efficient learner with this sample complexity for (ε, δ)-differential privacy. Both of their algorithms use dimensionality reduction (i.e., the Johnson-Lindenstrauss lemma) to reduce the dimension of the data from d to $O(1/\tau^2)$. One can then learn a halfspace by privately minimizing the hinge-loss on this lower dimensional space.

Meanwhile, one of the first applications of smooth boosting was to the study of noise-tolerant learning of halfspaces. Servedio [2003b] showed that smooth boosting can be used to design a (non-private) algorithm with sample complexity $\tilde{O}(1/\tau^2\alpha^2)$ which, moreover, tolerates an $\eta = O(\tau\alpha)$ rate of malicious noise. Given the close connection between smooth boosting and differential privacy, it is natural to ask whether private boosting can also be used to design a learner for large-margin halfspaces. Note that while one could pair the private boosting algorithm of Dwork, Rothblum, and Vadhan with our differentially private weak learner for this application, the resulting hypothesis would be a majority of halfspaces, rather than a single halfspace. Like Nguyễn et al. [2019] we address *proper* differentially-private learning of large-margin halfspaces where the hypothesis is itself a halfspace and not some more complex Boolean device.

We use our framework for private boosting to achieve a proper halfspace learner with sample complexity $\tilde{O}\left(\frac{1}{\varepsilon\alpha\tau^2}\right)$ when (ε, δ)-DP is required. Our learner is simple, efficient, and automatically tolerates random classification noise [Angluin and Laird, 1987] at a rate of $O(\alpha\tau)$. That is, we recover the sample complexity of Nguyễn et al. [2019] using a different algorithmic approach while also tolerating noise. Additionally, our efficient algorithm guarantees *zero-concentrated* differential privacy [Bun and Steinke, 2016], a stronger notion than (ε, δ)-DP. In this dissertation we phrase all guarantees as (ε, δ)-DP to facilitate comparison of sample bounds.

Theorem 2.1.1 (Informal, Fat-Shattering Application to Large-Margin Halfspaces). *Given $n = \tilde{O}\left(\frac{1}{\varepsilon\alpha\tau^2}\right)$ samples from a distribution D supported by a τ-margin halfspace u subject to $O(\alpha\tau)$-*

rate random label noise, our learning algorithm is (ε,δ)*-DP and outputs with probability* $(1-\beta)$ *a halfspace that* α*-approximates u over D.*

Furthermore, it may be interesting that we can also obtain non-trivial sample bounds for the same problem using *only* differential privacy. The analysis of Nguyễn et al. [2019] uses the VC dimension of halfspaces and the analyses of Servedio [2003b] and Theorem 2.1.1 above both use the fat-shattering dimension of halfspaces to ensure generalization. We can instead use the generalization properties of differential privacy to prove the following (in Section 2.7).

Theorem 2.1.2 (Informal, Privacy-Only Application to Large-Margin Halfspaces)**.** *Given* $n=\tilde{O}\left(\frac{1}{\varepsilon\alpha\tau^2}+\frac{1}{\alpha^2\tau^2}+\varepsilon^{-2}+\alpha^{-2}\right)$ *samples from a distribution D supported by a* τ*-margin halfspace u subject to* $O(\alpha\tau)$*-rate random label noise, our learning algorithm is* (ε,δ)*-DP and outputs with probability* $(1-\beta)$ *a halfspace that* α*-approximates u over D.*

Intuitively, the fat-shattering argument has additional "information" about the hypothesis class and so can prove better bounds. However, the argument based only on differential privacy would apply to *any* hypothesis class with a differentially private weak learner. So, we present a template for generalization of boosting in Sections 2.3.3 and 2.3.4 which relies *only* on the learner's privacy guarantees.

2.2 Private Boosting Preliminaries

In this section, we fix notation and definitions for describing measures and associated operations.

For a finite set X, let $\mathscr{U}(X)$ be the uniform distribution over X.

Definition 2.2.1 (Bounded Measures)**.** A *bounded measure* on domain X is a function $\mu : X \to [0,1]$.

Density: $d(\mu) = \mathbf{E}x \sim \mathscr{U}(X)\mu(x)$ — the "relative size" of a measure in X.

Absolute Size: $|\mu| = \sum_{x\in X} \mu(x)$

Induced Distribution: $\hat{\mu}(x) = \mu(x)/|\mu|$ — the distribution obtained by normalizing a measure

A measure is "nice" if it is simple and efficient to sample from the associated distribution. If a measure has high enough density, then rejection sampling will be efficient. So, the set of *high density* measures is important, and will be denoted by:

$$\Gamma_\kappa = \{\mu \mid d(\mu) \geq \kappa\}.$$

To maintain the invariant that we only call weak learners on measures of high density, we use Bregman projections onto the space of high density measures.

Definition 2.2.2 (Bregman Projection)**.** Let $\Gamma \subseteq \mathbb{R}^{|S|}$ be a non-empty closed convex set of measures over S. The *Bregman projection* of $\tilde{\mu}$ onto Γ is defined as:

$$\Pi_\Gamma \tilde{\mu} = \arg\min_{\mu \in \Gamma} \mathrm{KL}(\mu \parallel \tilde{\mu})$$

Bregman projections have the following desirable property:

Theorem 2.2.3 (Bregman, 1967)**.** *Let $\tilde{\mu}, \mu$ be measures such that $\mu \in \Gamma$. Then,*

$$\mathrm{KL}(\mu \parallel \Pi_\Gamma \tilde{\mu}) + \mathrm{KL}(\Pi_\Gamma \tilde{\mu} \parallel \tilde{\mu}) \leq \mathrm{KL}(\mu \parallel \tilde{\mu}). \textit{ In particular, } \mathrm{KL}(\mu \parallel \Pi_\Gamma \tilde{\mu}) \leq \mathrm{KL}(\mu \parallel \tilde{\mu}).$$

Barak, Hardt, and Kale gave the following characterization of Bregman projections onto the set of κ-dense measures, which we will also find useful.

Lemma 2.2.4 (Bregman Projection onto Γ_κ is Capped Scaling Barak et al. [2009a])**.** *Let Γ denote the set of κ-dense measures. Let $\tilde{\mu}$ be a measure such that $|\tilde{\mu}| < \kappa n$, and let $c \geq 1$ be the smallest constant such that the measure μ, where $\mu(i) = \min\{1, c \cdot \tilde{\mu}(i)\}$, has density κ. Then $\Pi_\Gamma \tilde{\mu} = \mu$.*

2.3 Abstract Boosting

In Section 2.3.2, we give sufficient conditions for private boosting, using a natural decomposition that applies to many boosting algorithms. In Sections 2.3.3 and 2.3.4 we use the decomposition to give templates of sample complexity bounds and noise-tolerant generalization guarantees for private boosted classifiers that use only the algorithmic stability imposed by differential privacy. We instantiate those templates in Section 2.7, to construct a noise-tolerant PAC learner for large-margin halfspaces.

2.3.1 Boosting Schemas

A boosting algorithm repeatedly calls a *weak* learning algorithm, aggregating the results to produce a final hypothesis that has good training error. Each call to the weak learner re-weights the samples so that samples predicted poorly by the hypothesis collection so far are given more "attention" (probability mass) by the weak learner in subsequent rounds. Thus boosting naturally decomposes into two algorithmic parts: the weak learner `WkL` and the re-weighting strategy `NxM`.

Below, we describe boosting formally using a "helper function" to iterate weak learning and re-weighting. Crucially, we avoid iterating over any information regarding the intermediate weights; the entire state of our schema is a list of hypotheses. This makes it easy to apply a privacy-composition theorem to any boosting algorithm where `NxM` and `WkL` satisfy certain minimal conditions, elaborated later. Much of the complexity in the analysis of private boosting by Dwork et al. [2010] was due to carefully privatizing auxiliary information about sample weights; we avoid that issue entirely. So, many smooth boosting algorithms could be easily adapted to our framework.

We denote by $\mathscr{H}$ the hypotheses used by the weak learner, by S an i.i.d. sample from the target distribution D, by T the number of rounds, by $\mathfrak{M}$ the set of bounded measures over S, and by $\mathscr{D}(S)$ the set of distributions over S.

Algorithm 1. `Boost`. In: $S \in X^n$, $T \in \mathbb{N}$

```
$H \leftarrow \{\}$
**for** $t = 1$ to $T$ **do**
    $H \leftarrow \texttt{Iter}(S, H)$
$\hat{f}(x) \leftarrow \frac{1}{T}\sum_{i=1}^{T} h_i(x)$
**return** $\operatorname{sign}(\hat{f}(x))$
```

Algorithm 2. `Iter`. In: $S \in X^n$, $H \in \mathscr{H}^*$

```
$\mu \leftarrow \texttt{NxM}(S, H)$
$h \leftarrow \texttt{WkL}(S, \mu)$
**return** $H \cup \{h\}$
// Add $h$ to list of hypotheses
```

2.3.2 Ensuring Private Boosting

Under what circumstances will boosting algorithms using this schema guarantee differential privacy? Since we output (with minimal post-processing) a collection of hypotheses from the weak learner, it should at least be the case that the weak learning algorithm `WkL` is itself differentially private. In fact, we will need that the output distribution on hypotheses of a truly private weak learner does not vary too much if it is called with both similar *samples* and similar *distributional targets*.

Definition 2.3.1 (Private Weak Learning)**.** A weak learning algorithm $\texttt{WkL} : S \times \mathscr{D}(S) \to \mathscr{H}$ satisfies (ρ, s)-zCDP if for all neighboring samples $S \sim S' \in (\mathscr{X}^n \times \{\pm 1\})$, all $\alpha > 1$, and any pair of distributions $\hat{\mu}, \hat{\mu}'$ on X such that $\Delta(\hat{\mu}, \hat{\mu}') < s$, we have:

$$D_\alpha(\texttt{WkL}(S, \hat{\mu}) \parallel \texttt{WkL}(S', \hat{\mu}')) \leq \rho\alpha$$

This makes it natural to demand that neighboring samples induce similar measures. Formally:

Definition 2.3.2 (ζ-Slick Measure Production)**.** A measure production algorithm $\texttt{NxM} : S \times \mathscr{H} \to \mathfrak{M}$ is called ζ-slick if, for all neighboring samples $S \sim S' \in (\mathscr{X}^n \times \{\pm 1\})$ and for all sequences of hypotheses $H \in \mathscr{H}^*$, letting $\hat{\mu}$ and $\hat{\mu}'$ be the distributions induced by $\texttt{NxM}(S, H)$ and $\texttt{NxM}(S', H)$ respectively, we have:

$$\Delta(\hat{\mu}, \hat{\mu}') \leq \zeta$$

It is immediate that a single run of `Iter` is private if it uses `NxM` and `WkL` procedures that are appropriately slick and private, respectively. Suppose `WkL` is (ρ_W, ζ)-zCDP and `NxM` is ζ-slick. By composition, `Iter` run using these procedures is ρ_W-zCDP. Finally, observe that `Boost` paired with a private weak learner and slick measure production is $T\rho_W$-zCDP, because the algorithm simply composes T calls to `Iter` and then post-processes the result.

2.3.3 Template: Privacy $\implies$ Boosting Generalizes

We now outline how to use the generalization properties of differential privacy to obtain a PAC learner from a private weak learner, via boosting. Recall that the fundamental boosting theorem is a *round bound:* after a certain number of rounds, boosting produces a collection of weak hypotheses that can be aggregated to predict the *training data* well.

Theorem 2.3.3 (Template for a Boosting Theorem)**.** *Fix* `NxM`*. For any weak learning algorithm* `WkL` *with advantage γ, running* `Boost` *using these concrete subroutines terminates in at most $T(\gamma, \alpha, \beta)$ steps and outputs (with probability at least $1-\beta$) a hypothesis H such that:*

$$\mathbf{Pr}_{(x,y)\sim S}[H(x) \neq y] \leq \alpha$$

We can capture the training error of a learning algorithm using a statistical query. For any hypothesis H, define:

$$\mathrm{err}_H(x,y) \mapsto \begin{cases} 1 & \text{if } H(x) \neq y \\ 0 & \text{otherwise} \end{cases}$$

Denoting by D the target distribution, PAC learning demands a hypothesis such that $\mathrm{err}_H(D) \leq \alpha$, with high probability. If the boosting process is differentially private, the generalization properties of differential privacy ensure that $\mathrm{err}_H(S)$ and $\mathrm{err}_H(D)$ are very close with high probability. Thus, boosting private weak learners can enforce low test error. We elaborate below.

Theorem 2.3.4 (Abstract Generalization)**.** *Let* `WkL` *be a (ρ, ζ)-zCDP weak learner with advantage γ. Suppose* `NxM` *is ζ-slick and enjoys round-bound T with error α and failure probability β. Denote by $\mathscr{M}'$ the algorithm* `Boost` *run using* `WkL` *and* `NxM`*. Let $\varepsilon = O(\sqrt{\rho T \log(1/\delta)})$ and suppose $n \geq \Omega(\log(\varepsilon/\delta)/\varepsilon^2)$. Then, with probability at least $1-\beta-\delta/\varepsilon$ over $S \sim_{iid} D^n$ and the internal randomness of $\mathscr{M}'$, the hypothesis output by $\mathscr{M}'$* generalizes *to D:*

$$\mathbf{Pr}_{(x,y)\sim D}[H(x) \neq y] \leq \alpha.$$

Proof (sketch). By the round-bound and inspection of Algorithm 1, $\mathscr{M}'$ simply composes T calls to `Iter` and post-processes. So by zCDP composition (Lemma 1.3.3) we know that $\mathscr{M}'$ is ρT-zCDP. This can be converted to an (ε, δ)-DP guarantee on $\mathscr{M}'$ for any $\delta > 0$ (Lemma 1.3.4).

For the sake of analysis, define a new mechanism $\mathscr{M}$ that runs $\mathscr{M}'(S)$ to obtain H and then outputs the statistical query err_H. This is just post-processing, so $\mathscr{M}$ is also (ε, δ)-DP. Thus, given enough samples, the conversion of privacy into generalization for statistical queries applies to $\mathscr{M}$:

$$\mathbf{Pr}_{S\sim D^n}[|\mathrm{err}_H(S) - \mathrm{err}_H(D)| \geq 18\varepsilon] \leq \delta/\varepsilon \text{ (Theorem 1.3.8)} .$$

By the guarantee of the round-bound, $\mathrm{err}_H(S) \leq \alpha$ with probability at least $1-\beta$. Therefore,

$$\mathbf{Pr}_{S\sim D^n}\left[\mathbf{Pr}_{(x,y)\sim D}[H(x) \neq y] \leq \alpha + 18\varepsilon\right] \leq \delta/\varepsilon + \beta.$$

□

Observe that we require privacy both for privacy's sake and for the generalization theorem. Whichever requirement is more stringent will dominate the sample complexity of any algorithm so constructed.

2.3.4 Template: Privacy $\implies$ Noise-Tolerant Generalization

Suppose now that there is some kind of interference between our learning algorithm and the training examples. For example, this could be modeled by random classification noise with rate η [Angluin and Laird, 1987]. This altered setting violates the preconditions of the DP to generalization transfer. A noised sample is *not* drawn i.i.d. from D and so the differential privacy of $\mathscr{M}$ is not sufficient to guarantee generalization of the "low training error" query err_H as defined above.

To get around this issue, we fold a noise model into the generalization-analysis mechanism. Define an alternative *noised* mechanism $\mathscr{M}_\eta$ (Algorithm 3) atop any $\mathscr{M}$ that outputs a "test error" query, and apply "DP to Generalization" on $\mathscr{M}_\eta$ instead. Suppose that $\mathscr{M}$ is differentially private, and the underlying learning algorithm $\mathscr{A}$ run by $\mathscr{M}$ tolerates noise at rate η. Then, if $\mathscr{M}_\eta$ is DP, we can generalize the noise-tolerance of $\mathscr{A}$.

Algorithm 3. $\mathscr{M}_\eta$ for RCN. Input: $S \in X^n, S \sim D^n$

```
$\forall i \in [n]$ $\mathsf{F}_i \leftarrow 1$
$\forall i \in [n]$ $\mathsf{F}_i \leftarrow -1$ with probability $\eta$
$\tilde{y}_i \leftarrow y_i \mathsf{F}_i$
$\tilde{S} \leftarrow \{(x_i, \tilde{y}_i)\}$
$\mathrm{err}_H \leftarrow \mathscr{M}(\tilde{S})$
**return** $\mathrm{err}_H$
```

At least for random classification noise, $\mathscr{M}_\eta$ does indeed maintain privacy. Observe that for a fixed noise vector $\mathbf{N}$, $\mathscr{M}_\eta$ run on neighboring data sets S and S' will run $\mathscr{M}$ with neighboring datasets $\tilde{S}$ and $\tilde{S}'$, and therefore the output distributions over queries will have bounded distance. Since the noise is determined independent of the sample, this means that $\mathscr{M}_\eta$ inherits the differential privacy of $\mathscr{M}$, and therefore satisfies the conditions of Theorem 1.3.8. So the resulting learner still generalizes.

This trick could handle much harsher noise models. For instance, each example selected for noise could be arbitrarily corrupted instead of given a flipped label. But we seem unable to capture fully malicious noise: an adversary viewing the whole sample could compromise privacy and so generalization. Thus, the "effective noise model" implicit above seems to distinguish between adversaries who have a global versus local view of the "clean" sample. This seems a natural division; we hope that future work will explore the expressiveness of this noise model.

2.4 Concrete Boosting via Lazy Bregman Projection

We instantiate the framework above. This requires a "Next Measure" routine (`LB-NxM`, Algorithm 4) a Boosting Theorem (Theorem 2.4.1) and a slickness bound (Lemma 2.4.2).

2.4.1 Measure Production Using Lazy Dense Multiplicative Weights

Our re-weighting strategy combines multiplicative weights with Bregman projections. In each round, we compute the collective margin on each example. Then, we multiplicative-weight the examples according to error: examples predicted poorly receive more weight. Finally, to ensure that no example receives too much weight, we Bregman-project the resulting measure into the space Γ of κ-dense measures. We call this strategy "lazy" because projection happens only *once* per round.

Algorithm 4. `LB-NxM(`κ, λ`)`: Lazy-Bregman Next Measure

```
Parameters: $\kappa \in (0,1)$, desired density of output measures; $\lambda \in (0,1)$, learning rate
Input: $S$, the sample; $H = \{h_1, \dots, h_t\}$, a sequence of hypotheses
Output: A measure over $[n]$, $n = |S|$
  $\mu_1(i) \leftarrow \kappa \ \forall i \in [n]$   /* Initial measure is uniformly $\kappa$ */
  for $j \in [t]$ do
    $\ell_j(x_i) \leftarrow 1 - \frac{1}{2}|h_j(x_i) - y_i| \ \forall i \in [n]$   /* Compute error of each hypothesis */
  $\tilde{\mu}_{t+1}(i) \leftarrow e^{-\lambda \sum_{j=1}^{t} \ell_j(x_i)} \mu_1(i) \ \forall i \in [n]$
  $\mu_{t+1} \leftarrow \Pi_\Gamma(\tilde{\mu}_{t+1})$
  return $\hat{\mu}_{t+1}$
```

`LB-NxM` is typed correctly for substitution into the `Boost` algorithm above; the measure is computed using *only* a sample and current list of hypotheses. Thus, `LazyBregBoost` = `Boost(LB-NxM)` admits a simple privacy analysis as in Section 2.3.2.

2.4.2 Boosting Theorem for Lazy Bregman Projection

Given a weak learner that beats random guessing, running `LazyBregBoost` yields low training error after a bounded number of rounds; we prove this in Section 2.8. Our argument adapts the well-known reduction from boosting to iterated play of zero-sum games [Freund and Schapire, 1996] for hypotheses with real-valued outputs. For completeness, we also give a self-contained analysis of the iterated-play strategy corresponding to `LB-NxM` in Section 2.9. Similar strategies are used by other differentially-private algorithms [Hsu et al., 2013, 2014] and their properties are known to follow from results in online convex optimization [Shalev-Shwartz, 2012]. However, to our knowledge an explicit proof for the "lazy" variant above does not appear in the literature; so we include one in Section 2.9. Overall, we have the follwing:

Theorem 2.4.1 (Lazy Bregman Round-Bound)**.** *Suppose we run* `Boost` *with* `LB-NxM(`$\kappa, \gamma/4$`)` *on a sample $S \subset \mathscr{X} \times \{\pm 1\}$ using any real-valued weak learner with advantage γ for $T \geq \frac{16 \log(1/\kappa)}{\gamma^2}$ rounds. Let $H : \mathscr{X} \to [-1, 1]$ denote the final, aggregated hypothesis. The process has:*

Good Margin: *H mostly agrees with the labels of S.*

$$\mathbf{Pr}_{(x,y)\sim S}\left[yH(x) \leq \gamma\right] \leq \kappa$$

Smoothness: *Every distribution $\hat{\mu}_t$ supplied to the weak learner has $\hat{\mu}_t(i) \leq \frac{1}{\kappa n}\ \forall i$*

2.4.3 Slickness Bound for Lazy Bregman Projection

`LB-NxM` is "lazy" in the sense that Bregman projection occurs only once per round, after all the multiplicative updates. The projection step is *not* interleaved between multiplicative updates. This is necessary to enforce slickness, which we require for privacy as outlined in Section 2.3.2.

Lemma 2.4.2 (Lazy Bregman Slickness)**.** The dense measure update rule `LB-NxM` (Algorithm 4) is ζ-slick for $\zeta = 1/\kappa n$.

Proof of Lemma 2.4.2. Let $\tilde{\mu}, \tilde{\mu}'$ be the unprojected measures produced at the end of the outermost loop of `NxM`, when `NxM` is run with the sequence of hypotheses $H = \{h_1, \ldots, h_T\}$, and on neighboring datasets $S \sim S'$. Let i be the index at which S and S' differ, and note that $\tilde{\mu}(j) = \tilde{\mu}'(j)$ for all $j \neq i$.

Let $\tilde{\mu}_0$ denote the measure with $\tilde{\mu}_0(j) = \tilde{\mu}(j) = \tilde{\mu}'(j)$ for all $j \neq i$, and $\tilde{\mu}_0(i) = 0$. Take Γ to be the space of κ-dense measures, and let $\mu_0 = \Pi_\kappa \tilde{\mu}_0$ and $\mu = \Pi_\kappa \tilde{\mu}$ denote the respective projected measures. We will show that $SD(\hat{\mu}_0, \hat{\mu}) \leq 1/\kappa n$, which is enough to prove the claim by the triangle inequality. (Note that $|\mu_0| = |\mu| = \kappa n$, which follows from Lemma 2.2.4 and the observation that $|\tilde{\mu}_0| \leq |\tilde{\mu}| \leq \kappa n$. Moreover, $\mu_0(j) \geq \mu(j)$ for every $j \neq i$.)

We calculate

$$\begin{aligned}
\sum_{j=1}^{n} |\mu_0(j) - \mu(j)| &= |\mu(i)| + \sum_{j \neq i} |\mu_0(j) - \mu(j)| \\
&\leq 1 + \sum_{j \neq i} \mu_0(j) - \mu(j) \\
&= 1 + |\mu_0| - (|\mu| - \mu(i)) \\
&\leq 1 + |\mu_0| - |\mu| + 1 \\
&= 2,
\end{aligned}$$

since μ and μ_0 have density κ. Hence,

$$\begin{aligned}\Delta(\hat{\mu}, \hat{\mu_0}) &= \frac{1}{2}\sum_{i=1}^{n}\left|\frac{\mu(i)}{|\mu|} - \frac{\mu_0(i)}{|\mu_0|}\right| \\ &= \frac{1}{2\kappa n}\sum_{i=1}^{n}|\mu(i) - \mu_0(i)| \\ &\leq \frac{1}{\kappa n}.\end{aligned}$$

□

2.5 Application: Learning Halfspaces with a Margin

2.5.1 Learning Settings

We first assume realizability by a large-margin halfspace. Let u be an unknown unit vector in $\mathbb{R}^d$, and let D be a distribution over examples from the ℓ_2 unit ball $B_d(1) \subset \mathbb{R}^d$. Further suppose that D is τ-good for u, meaning $|u \cdot x| \geq \tau$ for all x in the support of D. A PAC learner is given access to n i.i.d. labeled samples from D, honestly labeled by u.

A noise-tolerant learner is given access to a *label noise* example oracle with noise rate η, which behaves as follows. With probability $1-\eta$, the oracle returns a clean example $(x, \text{sign}(u \cdot x))$ for $x \sim D$. With probability η, the oracle returns an example with the label flipped: $(x, -\text{sign}(u \cdot x))$ for $x \sim D$. Given access to the noisy example oracle, the goal of a leaner is to output a hypothesis $h : B_d \to \{-1, 1\}$ which α-approximates u under D, i.e., $\mathbf{Pr}_{x \sim D}[h(x) \neq \text{sign}(u \cdot x)] \leq \alpha$ [Angluin and Laird, 1987].

Servedio [2003b] showed that smooth boosting can be used to solve this learning problem under the (more demanding) *malicious noise* rate $\eta = O(\alpha\tau)$ using sample complexity $n = \tilde{O}(1/(\tau\alpha)^2)$. We apply the Gaussian mechanism to his weak learner to construct a differentially private weak learner, and then boost it while preserving privacy. Our (best) sample complexity bounds then follow by appealing to the fat-shattering dimension of bounded-norm halfspaces in Section 2.5.4. Slightly worse bounds proved using only differential privacy are derived in Section 2.7.

2.5.2 Weak Halfspace Learner: Centering with Noise

The noise-tolerant weak learner for halfspaces was $\mathrm{WL}(S, \hat{\mu})$ which outputs the hypothesis $h(x) = z \cdot x$ where

$$z = \sum_{i=1}^{n} \hat{\mu}(j) \cdot y_i \cdot x_i.$$

The accuracy of this learner is given by the following theorem:

Theorem 2.5.1 (Servedio [2003b]). *Let $\hat{\mu}$ be a distribution over $[n]$ such that $L_\infty(\hat{\mu}) \leq 1/\kappa n$. Suppose that at most ηn examples in S do not satisfy the condition $y_i \cdot (u \cdot x_i) \geq \tau$ for $\eta \leq \kappa\tau/4$. Then* $\mathrm{WL}(S, \hat{\mu})$ *described above returns a hypothesis $h : B_d \to [-1, 1]$ with advantage at least $\tau/4$ under $\hat{\mu}$.*

We apply the Gaussian mechanism to Servedio's weak learner to obtain $\widehat{\mathrm{WL}}(S, \hat{\mu}, \sigma)$ which outputs $h(x) = \hat{z} \cdot x$ for

$$\hat{z} = \sum_{i=1}^{n} \hat{\mu}(j) \cdot y_i \cdot x_i + \nu,$$

with noise $\nu \sim \mathcal{N}(0, \sigma^2 I_d)$. We get a similar advantage bound, now trading off with privacy.

Theorem 2.5.2 (Private Weak Halfspace Learner). *Let $\hat{\mu}$ be a distribution over $[n]$ such that $L_\infty(\hat{\mu}) \leq 1/\kappa n$. Suppose that at most ηn examples in S do not satisfy the condition $y_i \cdot (u \cdot x_i) \geq \tau$ for $\eta \leq \kappa\tau/4$. Then we have:*

1. ***Privacy:*** $\widehat{\mathrm{WL}}(S, \hat{\mu}, \sigma)$ *satisfies (ρ, s)-zCDP for $\rho = \frac{2(1/\kappa n + s)^2}{\sigma^2}$.*

2. ***Advantage:*** *There is a constant c such that for any $\xi > 0$,* $\widehat{\mathrm{WL}}(S, \hat{\mu}, \sigma)$ *returns a hypothesis $h : B_d \to [-1, 1]$ that, with probability at least $1 - \xi$, has advantage at least $\tau/4 - c\sigma\sqrt{\log(1/\xi)}$ under $\hat{\mu}$.*

Proof. We begin with the proof of privacy. Let $\hat{\mu}_1, \hat{\mu}_2$ be κ-smooth distributions over $[n]$ with statistical distance $\Delta(\hat{\mu}_1, \hat{\mu}_2) \leq s$. Let $S \sim S'$ be neighboring datasets with $\{(x_i, y_i)\} = S \setminus S'$ and

$\{(x'_i, y'_i)\} = S' \setminus S$. Then we have

$$\begin{aligned}
\|\hat{z}_{S,\hat{\mu}_1} - \hat{z}_{S',\hat{\mu}_2}\|_2 &= \|\hat{\mu}_1(i) y_i \cdot x_i - \hat{\mu}_2(i) y'_i \cdot x'_i + \sum_{\substack{j=1 \\ j \neq i}}^{n} (\hat{\mu}_1(j) - \hat{\mu}_2(j)) y_j \cdot x_j\|_2 \\
&\leq \|\hat{\mu}_1(i) y_i \cdot x_i\|_2 + \|\hat{\mu}_2(i) y'_i \cdot x'_i\|_2 + \sum_{\substack{j=1 \\ j \neq i}}^{n} \|(\hat{\mu}_1(j) - \hat{\mu}_2(j)) y_j \cdot x_j\|_2 \\
&= \hat{\mu}_1(i) + \hat{\mu}_2(i) + \sum_{\substack{j=1 \\ j \neq i}}^{n} |(\hat{\mu}_1(j) - \hat{\mu}_2(j))| \\
&\leq 2\hat{\mu}_2(i) + \sum_{j=1}^{n} |\hat{\mu}_1(j) - \hat{\mu}_2(j)| \\
&\leq 2(1/\kappa n + s).
\end{aligned}$$

Then Lemma 1.3.5 gives us that

$$D_\alpha\left(\widehat{\mathrm{WL}}(S, \hat{\mu}_1, \sigma) \,\middle\|\, \widehat{\mathrm{WL}}(S', \hat{\mu}_2, \sigma)\right) \leq \frac{2\alpha(1/\kappa n + s)^2}{\sigma^2}$$

and therefore $\widehat{\mathrm{WL}}(S, \hat{\mu}_1, \sigma)$ satisfies (ρ, s)-zCDP for $\rho = \frac{2(1/\kappa n + s)^2}{\sigma^2}$.

Building on Servedio's result, we now give the advantage lower bound. Servedio's argument shows that the advantage of $\widehat{\mathrm{WL}}(S, \hat{\mu}, \sigma)$ is at least $\hat{z} \cdot u/2 = z \cdot u/2 + \nu \cdot u/2$. Since ν is a spherical Gaussian and u is a unit vector, we have that for any $\xi > 0$,

$$\mathbf{Pr}[|\nu \cdot u| \geq c\sigma\sqrt{\log(1/\xi)}] \leq \xi.$$

□

2.5.3 Strong Halfspace Learner: Boosting

Putting all the pieces together, we run `Boost` using the private weak halfspace learner (Theorem 2.5.2) and lazy-Bregman measures (Theorem 2.4.1). Via the composition theorem for differential privacy, we get a privacy guarantee for the terminal hypothesis as outlined in Section 2.3.2. Finally, we use the fat shattering dimension to ensure that this hypothesis generalizes.

Algorithm 5. Strong Halfspace Learner, via Boosting (`HS-StL`)

```
Input: Sample: $S$; Parameters: $(\alpha,\beta)$-PAC, $(\varepsilon,\delta)$-DP, $\tau$-margin
Output: A hypothesis $H$
    $\sigma \leftarrow \tau/8c\sqrt{\log\left(\frac{3072\log(1/\kappa)}{\beta\tau^2}\right)}$
    $T \leftarrow 1024\log(1/\kappa)/\tau^2$
    $H \leftarrow$ Boost run with LB-NxM($\kappa := (\alpha/4), \lambda := (\tau/8)$) and $\widehat{\mathrm{WL}}(\cdot,\cdot,\sigma)$ for $T$ rounds
```

2.5.4 Generalization via fat-shattering dimension.

Following the analysis of Servedio [2003b], we can show that with high probability the hypothesis output by our halfspace learner will generalize, even for a sample drawn from a distribution with random classification noise at rate $O(\alpha\tau)$. The proof of generalization goes by way of fat-shattering dimension. Using an argument nearly identical to that of Servedio [2000], we can bound the fat-shattering dimension of our hypothesis class. This bound, along with the guarantee of Theorem 2.4.1 that our final hypothesis will have good margin on a large fraction of training examples, allows us to apply the following generalization theorem of Bartlett and Shawe-Taylor, which bounds the generalization error of the final hypothesis.

Theorem 2.5.3 (Bartlett and Shawe-Taylor [1998])**.** *Let $\mathscr{H}$ be a family of real-valued hypotheses over some domain $\mathscr{X}$, let D be a distribution over labeled examples $\mathscr{X} \times \{-1,1\}$. Let $S = \{(x_1,y_1),\ldots,(x_n,y_n)\}$ be a sequence of labeled examples drawn from D, and let $h(x) = sign(H(x))$ for some $H \in \mathscr{H}$. If h has margin less than γ on at most k examples in S, then with probability at least $1-\delta$ we have that*

$$\mathbf{Pr}_{(x,y)\sim D}[h(x) \neq y] \leq \frac{k}{n} + \sqrt{\frac{2d}{n}\ln(34en/d)\log(578n) + \ln(4/\delta)}$$

where $d = fat_F(\gamma/16)$ is the fat-shattering dimension of $\mathscr{H}$ with margin $\gamma/16$.

In order to meaningfully apply the above theorem, we will need to bound the fat-shattering dimension of our hypothesis class $\mathscr{H}$. Our bound (proved in Section 2.6) follows from the analysis of Servedio [2000], but given that our hypothesis class is not exactly that analyzed in Servedio [2000], the bound holds only when the noise added to the hypotheses at each round of boosting does not increase the ℓ_2 norm of the final hypothesis by too much.

Lemma 2.5.4 (Fuzzy Halfspace Fat Shattering Dimension)**.** With probability $1-\frac{\beta}{3}$, after $T = \frac{1024\log(1/\kappa)}{\tau^2}$ rounds of boosting, Algorithm 5 outputs a hypothesis in a class with fat-shattering dimension $fat_{\mathscr{H}}(\gamma) \leq 4/\gamma^2$.

With the above bound on fat-shattering dimension, we may prove the following properties of `HS-StL`.

Theorem 2.5.5 (Private Learning of Halfspaces)**.** *The `HS-StL` procedure, is a (ε,δ)-Differentially Private (α,β)-strong PAC learner for τ-margin halfspaces, tolerating random classification noise at rate $O(\alpha\tau)$, with sample complexity*

$$n = \Omega\left(\underbrace{\frac{\sqrt{\log(1/\alpha)\log(1/\delta)\log(\log(1/\alpha)/\beta\tau^2)}}{\varepsilon\alpha\tau^2}}_{privacy} + \underbrace{\frac{\log(1/\tau\alpha)\log(1/\alpha)}{\alpha^2\tau^2} + \frac{\log(1/\beta)}{\kappa\tau}}_{accuracy}\right)$$

Proof. We begin by calculating the cumulative zCDP guarantee of `HS-StL`. First, by the privacy bound for $\widehat{\mathrm{WL}}$ (Theorem 2.5.2), we know that a single iteration of `Boost` is ρ-zCDP for $\rho = \frac{8}{(\kappa n\sigma)^2}$. Furthermore, by tight composition for zCDP (Lemma 1.3.3) and our setting of T, `HS-StL` is ρ_T-zCDP where:

$$\rho_T = O\left(\frac{\log(1/\kappa)}{(\kappa n\sigma\tau)^2}\right).$$

Denote by ε and δ the parameters of approximate differential privacy at the final round T of `HS-StL`. Now we convert from zero-concentrated to approximate differential privacy, via Lemma 1.3.4: for all $\delta > 0$, if $\varepsilon > 3\sqrt{\rho_T\log(1/\delta)}$, then `HS-StL` is (ε,δ)-DP. So, for a given target ε and δ, taking

$$n \in O\left(\frac{\sqrt{\log(1/\kappa)\log(1/\delta)\log(\log(1/\kappa)/\beta\tau)}}{\varepsilon\kappa\tau^2}\right)$$

will ensure the desired privacy.

We now turn to bounding the probability of events that could destroy good training error.

Too Many Corrupted Samples. Our proof of $\widehat{\mathrm{WL}}$'s advantage required that fewer than $\kappa\tau n/4$ examples are corrupted. At noise rate $\eta \leq \kappa\tau/8$, we may use a Chernoff bound to argue that the probability of exceeding this number of corrupted samples is at most $\beta/3$, by taking $n > \frac{24\log(3/\beta)}{\kappa\tau}$.

Gaussian Mechanism Destroys Utility. The Gaussian noise injected to ensure privacy could destroy utility for a round of boosting. Our setting of σ simplifies the advantage of $\widehat{\mathrm{WL}}$ to $\gamma(\tau,\sigma) = \tau/8$ with all but probability $\xi = \frac{\beta\tau^2}{3072\log(1/\kappa)}$. Then we have that with probability $(1-\xi)^T \geq 1-\frac{\beta}{3}$, every hypothesis output by $\widehat{\mathrm{WL}}$ satisfies the advantage bound $\gamma \geq \tau/8$. Therefore, by Theorem 2.4.1, HS-StL only fails to produce a hypothesis with training error less than κ with probability $\beta/3$.

We now consider events that cause generalization to fail.

Final hypothesis $H \notin \mathscr{H}$. The Gaussian noise added to ensure privacy could cause the final hypothesis H to fall outside the class $\mathscr{H} = \{f(x) = z \cdot x : \|z\|_2 \leq 2\}$, for which we have a fat-shattering dimension bound. The probability of this event, however, is already accounted for by the probability that the Gaussian Mechanism destroys the weak learner's utility, as both failures follow from the Gaussian noise exceeding some ℓ_2 bound. The failures that affect utility are a superset of those that affect the fat-shattering bound, and so the $\beta/3$ probability of the former subsumes the probability of the latter.

Failure internal to generalization theorem. Theorem 2.5.3 gives a generalization guarantee that holds only with some probability. We denote the probability of this occurrence by β_1.

If none of these failures occur, it remains to show that we can achieve accuracy α. From Lemma 2.8.5, we have that H will have margin $\gamma = \tau/8$ on all but a κ fraction of the examples, some of which may have been corrupted. We assume the worst case – that H is correct on all corrupted examples. We have already conditioned on the event that fewer than $\kappa\tau n/4$ examples have been corrupted, and so we may then conclude that H has margin less than γ on at most a 2κ fraction of the uncorrupted examples. Then if we set $\kappa = \alpha/4$ and take

$$n \in O\left(\frac{\log(1/\alpha\gamma)\log(1/\alpha)}{\alpha^2\tau^2}\right),$$

then so long as $e^{-\alpha^2} < \beta_1 < \beta/3$, we can apply Theorem 2.5.3 to conclude that

$$\mathbf{Pr}_{S\sim D^n}\left[\mathbf{Pr}_{(x,y)\sim D}[H(x) \neq y] < \alpha\right] \geq 1-\beta.$$

$\square$

2.6 Fuzzy Halfspaces have Bounded Fat-Shattering Dimension

We recall and prove Lemma 2.5.4.

Lemma 2.5.4 (Fuzzy Halfspace Fat Shattering Dimension)**.** With probability $1-\frac{\beta}{3}$, after $T = \frac{1024\log(1/\kappa)}{\tau^2}$ rounds of boosting, Algorithm 5 outputs a hypothesis in a class with fat-shattering dimension $fat_{\mathscr{H}}(\gamma) \leq 4/\gamma^2$.

This follows from the lemmas below due to Servedio, Bartlett, and Shawe-Taylor.

Lemma 2.6.1 (Servedio [2000])**.** *If the set $\{x_1,\ldots,x_n\}$ is γ-shattered by $\mathscr{H} = \{f(x) = z\cdot x : \|z\|_2 \leq 2\}$, then every $b \in \{-1,1\}^n$ satisfies*

$$\Big\|\sum_{i=1}^{n} b_i x_i\Big\|_2 \geq \gamma n/2.$$

Lemma 2.6.2 (Bartlett and Shawe-Taylor [1998])**.** *For any set $\{x_1,\ldots,x_n\}$ with each $x_i \in \mathbb{R}^d$ and $\|x_i\|_2 \leq 1$, then there is some $b \in \{-1,1\}^n$ such that $\big\|\sum_{i=1}^n b_i x_i\|_2 \leq \sqrt{n}$.*

Proof. We begin by showing that, with high probability, the hypothesis output by Algorithm 5 is in the class $\mathscr{H} = \{f(x) = z \cdot x : \|z\|_2 \leq 2\}$. To bound the ℓ_2 norm of z, we observe that

$$\|z\|_2 = \frac{1}{T}\Big\|\sum_{t=1}^{T} \hat{z}_t\Big\|_2 \leq \frac{1}{T}\sum_{t=1}^{T}\Big\|\sum_{i=1}^{n} \hat{\mu}_t(i) y_i x_i\Big\|_2 + \|\nu_t\|_2 = 1 + \frac{1}{T}\sum_{t=1}^{T}\|\nu_t\|_2$$

where $\hat{z}_t$ denotes the weak learner hypothesis at round t of boosting, and ν_t denotes the Gaussian vector added to the hypothesis at round t. Letting $\sigma = \tau/8c\sqrt{\log\left(\frac{3072\log(1/\kappa)}{\beta\tau^2}\right)}$ for a constant c, it follows that with probability at least $1 - \frac{\beta\tau^2}{3072\log(1/\kappa)} = 1 - \frac{\beta}{3T}$, a given ν_t has $\|\nu_t\|_2 \leq \tau/8 < 1$, and therefore with probability $(1-\frac{\beta}{3T})^T \geq (1-\frac{\beta}{3})$, $\frac{1}{T}\sum_{t=1}^{T}\|\nu_t\|_2 < 1$, and so $\|z\|_2 \leq 2$. Therefore with all but probability $\beta/3$, the hypothesis output by Algorithm 5 is in the class $\mathscr{H}$.

From Lemma 2.6.1, it cannot be the case that a set $\{x_1, \ldots, x_n\}$ is γ-shattered by $\mathscr{H}$ if there exists a $b \in \{-1,1\}^n$ such that

$$\Big\|\sum_{i=1}^{n} b_i x_i\Big\|_2 < \gamma n/2.$$

At the same time, it follows from Lemma 2.6.2 that if $n > 4/\gamma^2$, such a $b \in \{-1,1\}^n$ must exist. Therefore the fat-shattering dimension of $\mathscr{H}$ at margin γ is $fat_{\mathscr{H}}(\gamma) \leq 4/\gamma^2$. Since our final hypothesis is in $\mathscr{H}$ with probability $1-\beta/3$, our claim holds. □

2.7 Privacy-Only Noise-Tolerant Sample Bound for Large-Margin Halfspaces

We state and prove the formal version of Theorem 2.1.2.

Theorem 2.7.1 (Learning Halfspaces Under Random Label Noise). *The* `HS-StL` *procedure is a (ε,δ)-Differentially Private (α,β)-strong PAC learner for τ-margin halfspaces tolerating random label noise at rate $\eta = O(\alpha\tau)$ with sample complexity*

$$n = \tilde{\Omega}\left(\underbrace{\frac{1}{\varepsilon\alpha\tau^2}}_{\substack{\textit{privacy}\\ \textit{(Claim 2.7.2)}}} + \underbrace{\frac{1}{\alpha^2\tau^2}}_{\textit{accuracy}} + \underbrace{\frac{1}{\varepsilon^2} + \frac{1}{\alpha^2}}_{\substack{\textit{generalization}\\ \textit{(Claim 2.7.5)}}}\right)$$

Proof. Denote by ε_T and δ_T the parameters of approximate differential privacy at the final round T of HS-StL, and by the H the output hypothesis of HS-StL. We proceed as follows.

1. Given enough samples, HS-StL is differentially private. (Claim 2.7.2)
2. Random Label Noise at rate $\eta = O(\alpha\tau)$ will (w.h.p.) not ruin the sample. (Claim 2.7.3)
3. The Gaussian Mechanism will (w.h.p.) not ruin the weak learner. (Claim 2.7.4)
4. Given enough samples, training error is (w.h.p.) close to test error. (Claim 2.7.5)
5. Given enough samples, HS-StL (w.h.p.) builds a hypothesis with low test error.

For the remainder of this proof, fix the settings of all parameters as depicted in HS-StL (Algorithm 5). We reproduce them here:

$$\kappa \leftarrow \alpha/4 \tag{2.1}$$

$$\sigma \leftarrow \tau/8c\sqrt{\log\left(\frac{3072\log(1/\kappa)}{\beta\tau^2}\right)} \tag{2.2}$$

Claim 2.7.2 (Enough Samples $\implies$ HS-StL is Differentially Private)**.** *For every $\delta_T > 0$, we have:*

$$n > \tilde{O}\left(\frac{1}{\varepsilon_T\alpha\tau^2}\right) \implies \textit{HS-StL is } (\varepsilon_T, \delta_T)\textit{-DP}$$

Proof. By the privacy bound for $\widehat{\mathrm{WL}}$ (Theorem 2.5.2), we know that a single iteration of Boost is ρ-zCDP for $\rho = \frac{8}{(\kappa n\sigma)^2}$. Then, Boost runs for $T = \frac{1024\log(1/\kappa)}{\tau^2}$ rounds. So, by tight composition for zCDP (Lemma 1.3.3), HS-StL is ρ_T-zCDP where:

$$\rho_T = O\left(\frac{\log(1/\kappa)}{(\kappa n\sigma\tau)^2}\right)$$

Now we convert from zero-concentrated to approximate differential privacy, via Lemma 1.3.4: if $\varepsilon_T < 3\sqrt{\rho_T\log(1/\delta_T)}$, then HS-StL is $(\varepsilon_T, \delta_T)$-DP for all $\delta_T > 0$. We re-arrange to bound n.

$$\rho_T < O\left(\frac{\varepsilon_T^2}{\log(1/\delta_T)}\right)$$

Unpacking ρ_T we get:

$$\frac{\log(1/\kappa)}{(\kappa n\sigma\tau)^2} < O\left(\frac{\varepsilon_T^2}{\log(1/\delta_T)}\right)$$

This will hold so long as:

$$n > \Omega\left(\frac{\sqrt{\log(1/\kappa)\log(1/\delta_T)}}{\kappa\sigma\tau\varepsilon_T}\right)$$

Substituting the settings of σ and κ from `HS-StL`, we obtain:

$$n > \Omega\left(\frac{\sqrt{\log(1/\alpha)\log(1/\delta_T)\log(\log(1/\alpha)/\beta\tau^2)}}{\varepsilon_T\alpha\tau^2}\right)$$

□

We next consider the two events that could destroy good training error.

Too Many Corrupted Samples Noise could corrupt so many samples that the weak learner fails. Under an approprite noise rate, this is unlikely. We denote this event by BN (for "bad noise").

Gaussian Mechanism Destroys Utility The Gaussian noise injected to ensure privacy could destroy utility for a round of boosting. We denote this event by BG (for "bad Gaussian").

Both events are unlikely, under the settings of `HS-StL`.

Claim 2.7.3 (Hopelessly Corrupted Samples are Unlikely)**.** *Let* $\mathsf{F}_1,\ldots,\mathsf{F}_n$ *indicate the event "label i was flipped by noise," and denote by* $\mathsf{F} = \sum_{i=1}^n \mathsf{F}_i$ *the number of such corrupted examples. Under the settings of `HS-StL` and noise rate* $\eta = \alpha\tau/32$*, we have:*

$$n > \frac{96\ln(4/\beta)}{\alpha\tau} \implies \mathbf{Pr}[\mathsf{BN}] = \mathbf{Pr}[\mathsf{F} > \kappa n] \leq \beta/4$$

Proof. At noise rate η, we have $\mathbb{E}[\mathsf{F}] = n\eta$. From the definitions and Theorem 2.5.2,

$$\mathbf{Pr}[\mathsf{BN}] = \mathbf{Pr}\left[\mathsf{F} \geq \frac{\alpha\tau}{16}n\right]$$

We apply the following simple Chernoff bound: $\forall\delta \geq 1$

$$\Pr[\mathsf{F} \geq (1+\delta)\mathbb{E}[\mathsf{F}]] \leq \exp(-\mathbb{E}[\mathsf{F}]\,\delta/3)$$

Substituting with $\delta = 1$:

$$\Pr[\mathsf{F} \geq 2\eta n] \leq \exp(-(\eta n)/3)$$

Noise rate $\eta = \alpha\tau/32$ gives the appropriate event above:

$$\Pr[\mathsf{F} \geq 2\eta n] = \Pr\left[\mathsf{F} \geq \frac{\alpha\tau}{16}n\right] \leq \exp(-(\alpha\tau n)/96)$$

Constraining the above probability to less than β/k_1 for any constant $k_1 > 1$ we solve to obtain:

$$n > \frac{96\ln(k_1/\beta)}{\alpha\tau}$$

$\square$

Claim 2.7.4 (Bad Gaussians are Unlikely)**.** *Let BG_i indicate the event that the ith call to the weak learner fails to have advantage at least $\tau/8$. Under the settings of* `HS-StL`*:*

$$\mathbf{Pr}[\mathsf{BG}] = \mathbf{Pr}[\exists i\ \mathsf{BG}_i] \leq \beta/2$$

Proof. Our setting of σ simplifies the advantage of $\widehat{\mathrm{WL}}$ to $\gamma(\tau,\sigma) = \tau/8$ with all but probability $\xi = \frac{\beta\tau^2}{1024\log(1/\kappa)}$. Then, by the round bound for `LB-NxM` (Theorem 2.4.1), `Boost` will terminate after $T = \frac{8\log(1/\kappa)}{\gamma^2} = \frac{512\log(1/\kappa)}{\tau^2}$ rounds, and so we have that with probability $(1-\xi)^T \geq 1 - \frac{\beta}{2}$ every hypothesis output by $\widehat{\mathrm{WL}}$ satisfies the advantage bound.

$\square$

To enforce generalization, we capture both training and test error for any hypothesis H with a statistical query that indicates misclassification. Evaluated over the sample it is the training error of H, and evaluated over the population it is the test error of H.

$$\mathrm{err}_H(x,y) \mapsto \begin{cases} 1 & \text{if } yH(x) \leq 0 \\ 0 & \text{otherwise} \end{cases}$$

Claim 2.7.5 (Enough Samples $\implies$ Good Generalization)**.** *If* $0 < \varepsilon_T < 1/3$ *and* $0 < \delta_T < \varepsilon_T/4$

$$n \geq \tilde{\Omega}\left(\frac{1}{\varepsilon_T^2}\right) \implies \Pr_{S \sim D^n}\left[\mathrm{err}_H(D) \geq \mathrm{err}_H(S) + 18\varepsilon_T\right] \leq \delta_T/\varepsilon_T$$

Proof. Define (for analysis only) a procedure `HS-StL-test`, which outputs the "error" statistical query. That is, letting $H = $ `HS-StL`(S) where $S \sim D^n$, `HS-StL-test` prints err_H. Thus, `HS-StL-test` is a mechanism for selecting a statistical query.

Because `HS-StL-test` is simple post-processing, it inherits the privacy of `HS-StL`. Since we select err_H privately, it will (by Theorem 1.3.8) be "similar" on the sample S (training error) and the population D (test error). Ignoring over-estimates of test error and observing the sample bounds of Theorem 1.3.8, this gives the claim.

□

Claim 2.7.6 (Low Training Error is Likely)**.** *Given* $\neg\mathsf{GB}$ *and* $\neg\mathsf{BN}$*, we have* $\mathrm{err}_H(S) \leq \alpha/2$.

Proof. Let $\tilde{S}$ denote the noised sample, and let S_C and S_D be the "clean" and "dirty" subsets of examples, respectively.

Given $\neg\mathsf{GB}$ and $\neg\mathsf{BN}$, the weak learning assumption holds on every round. So, by Theorem 2.4.1, the boosting algorithm will attain low training error $\mathrm{err}_H(\tilde{S}) = \kappa$. This is not yet enough to imply low test error, because $\tilde{S} \not\sim_{iid} D$. So we bound $\mathrm{err}_H(S)$ using $\mathrm{err}_H(\tilde{S})$. Suppose that the noise affects training in the worst possible way: H fits *every* flipped label, so H gets *every* example in S_D wrong. Decompose and bound $\mathrm{err}_H(S)$ as follows:

$$\begin{aligned}
\mathrm{err}_H(S) &= \sum_{(x,y)\in S} \mathrm{err}_H(x,y) \\
&= \sum_{(x,y)\in S_C} \mathrm{err}_H(x,y) + \sum_{(x,y)\in S_D} \mathrm{err}_H(x,y) \\
&\leq \kappa + |S_D| && \text{Boosting Theorem, worst case fit} \\
&\leq \kappa + \frac{\alpha\tau}{16} && \neg\mathsf{BN}
\end{aligned}$$

Because the sample is from the unit ball, we have $\tau \in (0,1)$. Therefore, it is always the case that $\frac{\alpha\tau}{16} < \frac{\alpha}{4}$. So $\mathrm{err}_H(S) \leq \kappa + \alpha/4 \leq \alpha/2$, concluding proof of the above claim. $\square$

It remains only to select ε_T and δ_T so that the claims above may be combined to conclude low test error with high probability. Recall that our objective is sufficient privacy to simultaneously:

1. Ensure that `HS-StL` is (ε,δ)-DP
2. Apply DP to generalization transfer (Theorem 1.3.8) for good test error.

Both these objectives impose constraints on ε_T and δ_T. The requirement that the algorithm is desired to be (ε,δ)-DP in particular forces ε_T to be smaller than ε and δ_T to be smaller than δ. The transfer theorem is slightly more subtle; to PAC-learn, we require:

$$\mathbf{Pr}_{S\sim D^n}[\mathrm{err}_H(D) \geq \alpha] \leq \beta$$

While Claim 2.7.5 gives us that

$$\Pr_{S\sim D^n}[\mathrm{err}_H(D) \geq \mathrm{err}_H(S) + 18\varepsilon_T] \leq \delta_T/\varepsilon_T$$

So, accounting for both privacy and accuracy, we need $\varepsilon_T < \phi = \min(\varepsilon, \alpha/36)$. We can select any $\delta_T < \min(\delta,\ \phi\beta/4)$ to ensure that $\delta_T/\varepsilon_T < \beta/2$. By substituting the different realizations of these 'min' operations into Claims 2.7.2 and 2.7.5 we obtain the sample bound.

Finally, observe that with these settings we can union bound the probability of BN and BG and the event that generalization fails with β, as required for PAC learning. But it follows from the claims above that if $\neg\mathsf{BN}$ and $\neg\mathsf{BG}$ and a good transfer all occur, then the ouput hypothesis H has test error less than α, concluding the argument.

□

2.8 Smooth Boosting via Games

Here, we prove our round-bound and final margin guarantee for `LazyBregBoost`. The proof is a reduction to approximately solving two-player zero-sum games. We introduce the basic elements of game theory, then outline and excute the reduction. Overall, we recall and prove Theorem 2.4.1:

Theorem 2.4.1 (Lazy Bregman Round-Bound). *Suppose we run `Boost` with `LB-NxM`$(\kappa, \gamma/4)$ on a sample $S \subset \mathscr{X} \times \{\pm 1\}$ using any real-valued weak learner with advantage γ for $T \geq \frac{16 \log(1/\kappa)}{\gamma^2}$ rounds. Let $H : \mathscr{X} \to [-1,1]$ denote the final, aggregated hypothesis. The process has:*

Good Margin: *H mostly agrees with the labels of S.*

$$\mathbf{Pr}_{(x,y)\sim S}\left[yH(x) \leq \gamma\right] \leq \kappa$$

Smoothness: *Every distribution $\hat{\mu}_t$ supplied to the weak learner has $\hat{\mu}_t(i) \leq \frac{1}{\kappa n}\ \forall i$*

2.8.1 Two-Player Zero-Sum Games

A two player game can be described by a matrix, where the *rows* are indexed by "row player" strategies $\mathscr{P}$, the *columns* are indexed by "column player" strategies $\mathscr{Q}$, and each entry (i,j) of the matrix is the *loss* suffered by the row player when row strategy $i \in \mathscr{P}$ is played against column strategy $j \in \mathscr{Q}$. Such a game is *zero-sum* when the colum player is given as a reward the row player's loss. Accordingly, the row player should minimize and the column player should maximize.

A single column or row is called a *pure strategy.* To model Boosting, we imagine players who can randomize their actions. So the fundamental objects are *mixed strategies:* distributions P over the rows and Q over the columns. Playing "according to" a mixed strategy means sampling from the distribution over pure strategies and playing the result. When two mixed strategies are played against each other repeatedly, we can compute the *expected loss* of P vs. Q playing the game M:

$$M(P,Q) = \underbrace{\sum_{i,j \in \mathscr{P} \times \mathscr{Q}} P(i)M(i,j)Q(j)}_{\text{(i)}} = \underbrace{\sum_{j \in \mathscr{Q}} M(P,j)Q(j)}_{\text{(ii)}} = \underbrace{\sum_{i \in \mathscr{P}} P(i)M(i,Q)}_{\text{(iii)}} \tag{2.3}$$

"Iterated play" pits the row player against an arbitrary environment represented by the column player. At each round, both the row player and column player choose strategies P_t and Q_t respectively. The expected loss of playing Q_t against each *pure* row strategy is revealed to the row player. Then, the row player suffers the *expected loss* of P_t vs. Q_t. This set-up is depicted by Algorithm 6. Good row player strategies have *provably bounded regret* — they do not suffer much more loss than the best possible *fixed* row player strategy in hindsight during iterated play.

Algorithm 6. Iterated Play

```
[Input: $T$ the number of rounds to play for
Output: Total expected row player cost incurred]
  for $t = 1$ to $T$ do
    $P_t \leftarrow$ Row player choice of mixed strategies, seeing $\ell_1, \ldots, \ell_{t-1}$
    $Q_t \leftarrow$ Column player choice of mixed strategies, seeing $P_t$
    $\ell_t(i) \leftarrow M(i, Q_t)\ \forall i$   /* Reveal loss on each pure row strategy */
    $C \leftarrow C + M(P_t, Q_t)$   /* Accumulate total loss */
```

Here, we reduce boosting to the "Lazy Dense Multiplicative Updates" row player strategy (Algorithm 7) which enjoys bounded regret (Lemma 2.8.1, proved in Section 2.9 for completeness) and two other helpful properties:

Simple State: The only state is all previous loss vectors and step count so far; this enables privacy.

Single Projection: It Bregman-projects just once per round; this enforces slickness.

Lemma 2.8.1 (Lazy Dense Updates Regret Bound)**.** *Let Γ be the set of κ-dense measures. Set $\mu_1(i) = \kappa$ for every i. Then for all $\mu \in \Gamma$ we have the following regret bound.*

$$\frac{1}{T}\sum_{t=1}^{T} M(\hat{\mu}_t, Q_t) \leq \frac{1}{T}\sum_{t=1}^{T} M(\hat{\mu}, Q_t) + \lambda + \frac{\mathrm{KL}(\mu \parallel \mu_1)}{\lambda \kappa |\mathscr{P}| T}$$

Algorithm 7. Lazy Dense Update Strategy (LDU)

```
Input: $\mathscr{P}$, a set of pure row-player strategies, learning rate $\lambda$, losses $\ell_1, \dots, \ell_T$
Output: A measure over $\mathscr{P}$
  for $i \in \mathscr{P}$ do
    $\mu_1(i) \leftarrow \kappa$
  for $i \in \mathscr{P}$ do
    $\tilde{\mu}_{T+1}(i) \leftarrow e^{-\lambda \sum_{t=1}^{T} \ell_t(i)} \mu_1(i)$
  $\mu_{T+1} \leftarrow \Pi_\Gamma \tilde{\mu}_T$
```

2.8.2 Reducing Boosting to a Game

We present a reduction from boosting real-valued weak learners to approximately solving iterated games, following Freund and Schapire [1996]. To prove the necessary round-bound (Theorem 2.4.1), we do the following:

1. **Create a Game.** The meaning of "advantage" given by a Weak Learning assumption (Definition 1.2.2) naturally induces a zero-sum game where pure row strategies are points in the sample S and pure column strategies are hypotheses in $\mathscr{H}$. The Booster will play mixed row strategies by weighting the sample and the weak learner will play pure column strategies by returning a single hypothesis at each round.

2. **Weak Learning $\implies$ Booster Loss Lower-Bound.** Weak Learners have some advantage in predicting with respect to *any* distribution on the sample. Thus, the particular sequence of distributions played by *any* Booster must incur at least some loss.

3. **Imagine Pythia, a Prophetic Booster.** Given perfect foreknowledge of how the weak learner will play, what is the best Booster strategy? Create a "prescient" Boosting strategy $P^\star$ which concentrates measure on the "worst" examples $(x, y) \in S$ for the *final* hypothesis H at each round.

4. **How Well Does Pythia Play?** Upper-bound the total loss suffered by a Booster playing $P^\star$ each round.

5. **Solve for T:** Recall that we want the Booster to *lose*. That is, we want an accurate ensemble

of hypotheses. Combining the upper and lower bounds above with the regret bound, we solve for a number of rounds to "play" (Boost) for such that the size of the set of "worst" examples shrinks to a tolerable fraction of the sample. This gives the round-bound (Theorem 2.4.1).

Create a Game

Our game is a continuous variant of the Boolean "mistake matrix" [Freund and Schapire, 1996]. Let $\mathscr{H} \subseteq \{B_2(1) \to [-1,1]\}$ be a set of bounded $\mathbb{R}$-valued hypothesis functions on the unit ball. These functions will be the pure column player strategies. Now let $S = (x_1, y_1), \ldots, (x_n, y_n)$ be a set of points where $y_i \in \{\pm 1\}$ and $x_i \in B_2(1)$. These points will be the pure row player strategies. Having chosen the strategies, all that remains is to define the entries of a matrix. To cohere with the definition of weak learning for real-valued functions, we define the following game of *soft punishments*:

$$M_S^{\mathscr{H}} := M_S^{\mathscr{H}}(i,h) = 1 - \frac{1}{2}|h(x_i) - y_i|$$

Notice the quantification here: we can define the soft punishment game for any sample and any set of hypotheses. We will omit $\mathscr{H}$ and S when fixed by context. This is a simple generalization of the game from Freund and Schapire [1996] which assigns positive punishment 1 to correct responses, and 0 to incorrect responses. Since we work with real-valued instead of Boolean predictors, we alter the game to scale row player loss by how "confident" the hypothesis is on an input point.

Weak Learning $\implies$ Booster Loss Lower-Bound.

Mixed strategies for the booster (row player) are just distributions over the sample. So the accuracy assumption on the weak learner `WkL`, which guarantees advantage on *every* distribution, induces a lower bound on the total loss suffered by *any* booster playing against `WkL`. Recall that losses are measured with respect to *distributions* over strategies, not *measures*. So, below we normalize any measure to a distribution "just before" calculating expected loss

Lemma 2.8.2 (Utility of Weak Learning)**.** *For any sequence of T booster mixed strategies $(\mu_1, \ldots, \mu_T)$, suppose the sequence of column point strategies $h_1, \ldots, h_T$ is produced by a weak learner that has* *advantage γ. Then:*

$$\sum_{t=1}^{T} M(\hat{\mu}_t, h_t) \geq {}^{T}\!/\!_{2} + T\gamma$$

Proof.

$$\begin{aligned}
\sum_{t=1}^{T} M(\hat{\mu}_t, h_t) &= \sum_{t=1}^{T} \mathbf{E}\, i \sim \hat{\mu}_t M(i, h_t) && \text{unroll def 2.3 } (iii) \\
&= \sum_{t=1}^{T} \mathbf{E}\, i \sim \hat{\mu}_t 1 - {}^{1}\!/\!_{2}|h_t(x_i) - y_i| && \text{re-arrange ``advantage''} \\
&= \sum_{t=1}^{T} {}^{1}\!/\!_{2} + \mathbf{E}\, i \sim \hat{\mu}_t {}^{1}\!/\!_{2} - {}^{1}\!/\!_{2}|h_t(x_i) - y_i| && \text{linearity of } \mathbb{E} \\
&= {}^{T}\!/\!_{2} + \sum_{t=1}^{T} \mathbf{E}\, i \sim \hat{\mu}_t {}^{1}\!/\!_{2} h_t(x_i) y_i && \text{distributing summations} \\
&\geq {}^{T}\!/\!_{2} + T\gamma && \text{by Weak Learning Assumption}
\end{aligned}$$

□

Imagine Pythia, a Prophetic Booster.

How should a booster play if she knows the future? Suppose Pythia knows exactly which hypotheses $h_1, \dots, h_T$ the weak learner will play, but is restricted to playing the same fixed κ-dense strategy for all T rounds. Intuitively, she should assign as much mass as possible to points of S where the combined hypothesis $H = (1/T)\sum_{t \in [T]} h_t$ is incorrect, and then assign remaining mass to points where H is correct but uncertain. We refer to this collection of points as B, the set of "bad" points for $h_1, \dots, h_T$. We formalize this strategy as Algorithm 8.

The prophetic booster Pythia plays the uniform measure on a set B of "bad" points selected by Algorithm 8, normalized to a distribution. That is:

$$P^{\star}(i) = \begin{cases} 1/|B| & \text{if } i \in B \\ 0 & \text{otherwise} \end{cases}$$

It is important to observe that if i is outside of the "bad set" B, we know H has "good" margin

Algorithm 8. Pythia

```
Input: $S$ a sample with $|S| = n$; $H$ a combined hypothesis; $\kappa$ a target density
Output: Distribution $P^\star$ over $[n]$; Minimum margin $\theta_T$
  $B \leftarrow \{i \in [1,n] \mid y_i H(x_i) < 0\}$    /* Place all mistakes in $B$ */
  Sort $[1,n] \setminus B$ by margin of $H$ on each point
  $\theta_T \leftarrow 0$
  while $|B| < \kappa n$ do
    Add minimum margin element $i$ of $[1,n] \setminus B$ to $B$
    Update $\theta_T$ to margin of $H$ on $(x_i, y_i)$
  $P^\star \leftarrow$ the uniform distribution over $B$
  Output $P^\star, \theta_T$
```

on (x_i, y_i). To quantify this, observe that for all $i \in B$, H has margin at most θ_T on (x_i, y_i).

Proposition 2.8.3 (Bad Margin in Bad Set). *For every $i \in B$, we know $\sum_{t=1}^{T} y_i h(x_i) \leq T\theta_T$*

Proof.

$$\begin{aligned} i \in B \implies y_i H(x_i) &\leq \theta_T && \text{inspection of Pythia, above} \\ \frac{y_i}{T}\sum_{t=1}^{T} h_t(x_i) &\leq \theta_T && \text{unroll } H \\ \sum_{t=1}^{T} y_i h_t(x_i) &\leq T\theta_T && \text{re-arrange} \end{aligned}$$

□

How Well Does Pythia Play?

Here, we calculate the utility of foresight — an upper-bound on the loss of $P^\star$. Suppose H is the terminal hypothesis produced by the boosting algorithm. We substitute $P^\star$ into the definition of expected loss for $M_S^{\mathscr{H}}$ (soft punishments) and relate the margin on the bad set for H to the cumulative loss of H, giving the following lemma.

Lemma 2.8.4 (Excellence of Pythia). *Let S be a sample, $(h_1, \ldots, h_T) \in \mathscr{H}^T$ a sequence of hypotheses, $H = (1/T)\sum_{i=1}^{T} h_i$, and $\kappa \in [0, 1/2]$ a density parameter. Let $P^\star, \theta_H =$* Pythia(S, H, κ). *Then:*

$$\sum_{t=1}^{T} M(P^\star, h_t) \leq (T/2) + (T\theta_H)/2$$

We require a simple fact about advantages. Since $h(x) \in [-1,+1]$ and $y \in \{\pm 1\}$, we know:

$$\begin{aligned} yh(x) &= 1 - |h(x) - y| \\ \implies (1/2)(yh(x)) &= (1/2) - (1/2)|h(x) - y| \end{aligned}$$

The entries of the soft punishments matrix can also be re-written by formatting advantage as above. For $i \in [1,n]$ and $h \in \mathscr{H}$ we have:

$$\mathrm{M}(i,h) = (1/2) + (1/2)(y_i h(x_i)) \tag{2.4}$$

Proof. We manipulate the total regret of $P^\star$ towards getting an upper-bound in terms of the minimum margin of H and number of rounds played.

$$\begin{aligned}
\sum_{t=1}^{T} M(P^\star, h_t) &= \sum_{t=1}^{T}\sum_{i=1}^{n} P^\star(i)\cdot \mathrm{M}(i,h_t) && \text{Part (iii) of Expected Loss (Definition 2.3)} \\
&= \sum_{t=1}^{T}\sum_{i\in B} P^\star(i)\cdot \mathrm{M}(i,h_t) && \text{Restrict sum — } P^\star(i)=0 \text{ outside } B \\
&= \frac{1}{|B|}\sum_{t=1}^{T}\sum_{i\in B} \mathrm{M}(i,h_t) && \text{Factor out } P^\star(i) \text{ — constant by definition} \\
&= \frac{1}{|B|}\sum_{i\in B}\sum_{t=1}^{T} ((1/2) + (1/2)h_t(x_i)y_i) && \text{Equation 2.4 about } M_S^{\mathscr{H}} \text{ entries} \\
&= \frac{1}{|B|}\left((|B|T)/2 + (1/2)\sum_{i\in B}\sum_{t=1}^{T} h_t(x_i)y_i\right) && \text{Algebra} \\
&\le \frac{1}{|B|}\left((|B|T)/2 + (1/2)\sum_{i\in B} T\theta\right) && \text{Bad margin in } B \text{ (Proposition 2.8.3)} \\
&= (T/2) + (T\theta)/2 && \text{Evaluate \& re-arrange}
\end{aligned}$$

□

Solve for T.

We now have an upper bound on the loss incurred by a prescient booster, and a lower bound on the loss to *any* booster under the weak learning assumption. This allows us to "sandwich" the performance of boosting according to the lazy dense updates (LDU, Algorithm 7) strategy between these two extremes, because LDU has good performance relative to *any* fixed strategy (Lemma 2.8.1, proved in Section 2.9). This sandwich gives a relationship between the number of rounds T and the margin of the final hypothesis, which we now solve for the number of rounds necessary to boost using LDU to obtain a "good" margin on "many" samples.

Lemma 2.8.5. *Let S be a sample of size n, let μ_t be the measure produced at round t by $\mathtt{NxM}(S, H_{t-1})$ playing the Lazy Dense Update Strategy of Algorithm 7, and let h_t be the hypothesis output by $\widehat{\mathtt{WkL}}(S, \hat{\mu}_{t-1}, \sigma)$ at round t. Then after $T \geq \frac{16\log(1/\kappa)}{\gamma^2}$ rounds of $\mathtt{Iter}$, the hypothesis $H_T(x) = \frac{1}{T}\sum_{t=1}^{T} h_t(x)$ has margin at least γ on all but κn many samples.*

Proof. Denote by $\mathscr{U}(B)$ the uniform measure (all $x \in B$ assigned a weight of 1) on the bad set B discovered by Pythia. Combining the regret bound comparing LDU to fixed Pythia with the lower bound on loss that comes from the weak learner assumption, we have, overall:

$$\frac{T}{2} + T\gamma \underbrace{\leq}_{\substack{\text{Weak Learning} \\ \text{(Lemma 2.8.2)}}} \sum_{t=1}^{T} \mathrm{M}(\hat{\mu}_t, h_t) \underbrace{\leq}_{\substack{\text{Regret Bound} \\ \text{(Lemma 2.8.1)}}} \sum_{t=1}^{T} \mathrm{M}(P^\star, h_t) + \lambda T + \frac{\mathrm{KL}(\mathscr{U}(B) \parallel \mu_1)}{\kappa n \lambda}$$

Apply Lemma 2.8.4, replacing prescient Boosting play with the upper bound on loss we obtained:

$$T\gamma \leq \sum_{t=1}^{T} \mathrm{M}(\hat{\mu}_t, h_t) \leq \frac{T\theta_H}{2} + \lambda T + \frac{\mathrm{KL}(\mathscr{U}(B) \parallel \mu_1)}{\kappa n \lambda}$$

Let's compute the necessary KL-divergence, recalling that $|\mathscr{U}(B)| = |\mu_1| = \kappa n$:

$$\begin{aligned}
\mathrm{KL}(\mathscr{U}(B) \parallel \mu_1) &= \sum_{x \in B} \mathscr{U}(B)(x) \log\left(\frac{\mathscr{U}(B)(x)}{\mu_1(x)}\right) - |\mathscr{U}(B)| + |\mu_1| \\
&= \sum_{x \in B} \log\left(\frac{1}{\kappa}\right) \\
&= \kappa n \log\left(\frac{1}{\kappa}\right)
\end{aligned}$$

Substituting into the above and dividing through by T, we have:

$$\gamma \le \frac{\theta_H}{2} + \lambda + \frac{\log(1/\kappa)}{T\lambda}$$

Which, setting

$$T = \frac{16 \log\left({}^{1}\!/\!{}_{\kappa}\right)}{\gamma^2} \text{ and } \lambda = \gamma/4$$

implies

$$\theta_H \ge \gamma.$$

This is the margin bound we claimed, for every $(x, y) \notin B$. $\square$

2.9 Bounded Regret for Lazily Projected Updates

A "lazy" multiplicative weights strategy that, at each round, projects only *once* into the space of dense measures is presented below. Here, we prove that this strategy (Algorithm 9) has bounded regret relative to any fixed dense strategy.

Algorithm 9. Lazy Dense Update Process

```
Input: $\mathscr{P}$, a set of pure row-player strategies, learning rate $\lambda$, losses $\ell_1, \ldots, \ell_T$
Output: A measure over $P$
  for $x \in \mathscr{P}$ do
    $\mu_1(x) \leftarrow \kappa$
  for $x \in \mathscr{P}$ do
    $\tilde{\mu}_{T+1}(x) \leftarrow e^{-\lambda \sum_{t=1}^{T} \ell_t(x)} \mu_1(x)$
  $\mu_{T+1} \leftarrow \Pi_\Gamma \tilde{\mu}_T$
```

To analyze the regret of a row player playing the strategy of Algorithm 9, we will need the following definition.

Definition 2.9.1 (Strong convexity)**.** A differentiable function $f : \mathbb{R}^n \to \mathbb{R}$ is α-strongly convex on $X \subset \mathbb{R}^n$ with respect to the ℓ_p norm if for all $x, y \in X$

$$\langle \nabla f(x) - \nabla f(y), x - y \rangle \geq \alpha \|x - y\|_p^2.$$

If f is twice differentiable, then f is α-strongly convex on X with respect to the ℓ_p norm if for all $x \in X$, $y \in \mathbb{R}^n$

$$y^T (\nabla^2 f(x)) y \geq \alpha \|y\|_p^2.$$

We now proceed to analyze Algorithm 9 by arguments closely following those found in Chapter 2 of Rakhlin's notes on online learning [Rakhlin, 2009]. First, we show this update rule minimizes a sequential regularized loss over *all* dense measures.

Lemma 2.9.2. *Let $M(\hat{\mu}, Q_t) = \mathbb{E}_{i \sim \hat{\mu}}[\ell_t(i)]$. Let Γ be the set of κ-dense measures, and let μ_1 be the uniform measure over $\mathscr{P}$ of density κ ($\mu_1(x) = \kappa$ for all $x \in P$). Then for all $T \geq 1$, the measure μ_{T+1} produced by Algorithm 9 satisfies*

$$\mu_{T+1} = \operatorname*{arg\,min}_{\mu \in \Gamma} \left[\lambda |\mu| \sum_{t=1}^{T} M(\hat{\mu}, Q_t) + \mathrm{KL}(\mu \parallel \mu_1) \right]$$

Proof. Suppose towards contradiction that there exists $\mu \in \Gamma$ such that

$$\lambda |\mu| \sum_{t=1}^{T} M(\hat{\mu}, Q_t) + \mathrm{KL}(\mu \parallel \mu_1) < \lambda |\mu_{T+1}| \sum_{t=1}^{T} M(\hat{\mu}_{T+1}, Q_t) + \mathrm{KL}(\mu_{T+1} \parallel \mu_1).$$

Then it must be the case that

$$\begin{aligned}
\mathrm{KL}(\mu_{T+1} \parallel \mu_1) - \mathrm{KL}(\mu \parallel \mu_1) &> \lambda \sum_{t=1}^{T} \langle \mu - \mu_{T+1}, \ell_t \rangle \\
&= \langle \mu - \mu_{T+1}, \sum_{t=1}^{T} \lambda \ell_t \rangle \\
&= \sum_i (\mu(i) - \mu_{T+1}(i)) \log \tfrac{\mu_1(i)}{\tilde{\mu}_{T+1}(i)} \\
&= \sum_i \mu(i) \left(\log \tfrac{\mu(i)}{\tilde{\mu}_{T+1}} - \log \tfrac{\mu(i)}{\mu_1(i)} \right) - \sum_i \mu_{T+1}(i) \left(\log \tfrac{\mu_{T+1}(i)}{\tilde{\mu}_{T+1}(i)} - \log \tfrac{\mu_{T+1}(i)}{\mu_1(i)} \right) \\
&= \mathrm{KL}(\mu \parallel \tilde{\mu}_{T+1}) - \mathrm{KL}(\mu \parallel \mu_1) - \mathrm{KL}(\mu_{T+1} \parallel \tilde{\mu}_{T+1}) + \mathrm{KL}(\mu_{T+1} \parallel \mu_1).
\end{aligned}$$

Under our assumption, then, it must be the case that $0 > \mathrm{KL}(\mu \parallel \tilde{\mu}_{T+1}) - \mathrm{KL}(\mu_{T+1} \parallel \tilde{\mu}_{T+1})$, but μ_{T+1} was defined to be the κ-dense measure that minimized the KL divergence from $\tilde{\mu}_{T+1}$, and so we have a contradiction. □

Using Lemma 2.9.2, we can now show the following regret bound for a row player that, at each round, "knows" the column strategy Q_t that it will play against that round.

Lemma 2.9.3. *Let Γ be the set of κ-dense measures. Let μ_1 be the uniform measure over $\mathscr{P}$ of density κ ($\mu_1(x) = \kappa$ for all $x \in \mathscr{P}$), and let $|\mathscr{P}| = n$. Then for all $\mu \in \Gamma$ we have that*

$$\sum_{t=1}^{T} M(\hat{\mu}_{t+1}, Q_t) \leq \sum_{t=1}^{T} M(\hat{\mu}, Q_t) + \frac{\mathrm{KL}(\mu \parallel \mu_1)}{\lambda \kappa n}$$

Proof. For $T = 0$, this follows immediately from the definition of μ_1.

Assume now that for any $\mu \in \Gamma$ that

$$\sum_{t=1}^{T-1} M(\hat{\mu}_{t+1}, Q_t) \leq \sum_{t=1}^{T-1} M(\hat{\mu}, Q_t) + \frac{\mathrm{KL}(\mu \parallel \mu_1)}{\lambda \kappa n},$$

and in particular

$$\sum_{t=1}^{T-1} M(\hat{\mu}_{t+1}, Q_t) \leq \sum_{t=1}^{T-1} M(\hat{\mu}_{T+1}, Q_t) + \frac{\mathrm{KL}(\mu \parallel \mu_1)}{\lambda \kappa n}.$$

It follows that

$$\begin{aligned}
\sum_{t=1}^{T} M(\hat{\mu}_{t+1}, Q_t) &= \sum_{t=1}^{T-1} M(\hat{\mu}_{t+1}, Q_t) + M(\hat{\mu}_{T+1}, Q_t) \\
&= \sum_{t=1}^{T-1} M(\hat{\mu}_{t+1}, Q_t) + \sum_{t=1}^{T} M(\hat{\mu}_{T+1}, Q_t) - \sum_{t=1}^{T-1} M(\hat{\mu}_{T+1}, Q_t) \\
&\le \sum_{t=1}^{T} M(\hat{\mu}_{T+1}, Q_t) + \frac{\mathrm{KL}(\mu_{T+1} \parallel \mu_1)}{\lambda \kappa n} \\
&\le \sum_{t=1}^{T} M(\hat{\mu}, Q_t) + \frac{\mathrm{KL}(\mu \parallel \mu_1)}{\lambda \kappa n},
\end{aligned}$$

for all $\mu \in \Gamma$, since $\mu_{T+1} = \arg\min_{\mu \in \Gamma}[\lambda|\mu| \sum_{t=1}^{T} M(\hat{\mu}, Q_t) + \mathrm{KL}(\mu \parallel \mu_1)]$. □

To show a regret bound for the lazy dense update rule of Algorithm 9, we now need only relate the lazy-dense row player's loss to the loss of the row player with foresight. To prove this relation, we will need the following lemma showing strong convexity of $R(\mu) = \mathrm{KL}(\mu \parallel \mu_1)$ on the set of κ-dense measures.

Lemma 2.9.4. *The function $R(\mu) = \mathrm{KL}(\mu \parallel \mu_1)$ is $(1/\kappa n)$-strongly convex over the set of measures with density no more than κ, with respect to the ℓ_1 norm.*

Proof. Let μ be a measure of density $d \le \kappa$, and let $x = (\frac{1}{\mu(1)}, \ldots, \frac{1}{\mu(n)})$. The Hessian of $R(\mu)$ is $\nabla^2 R(\mu) = I_n x$. Therefore, for all $y \in \mathbb{R}^n$,

$$\begin{aligned}
y^T \nabla^2 R(\mu) y &= \sum_i \frac{y_i^2}{\mu(i)} \\
&= \frac{1}{|\mu|} \sum_i \mu(i) \sum_i \frac{y_i^2}{\mu(i)} && \text{multiply by 1} \\
&\ge \frac{1}{|\mu|} \left(\sum_i \sqrt{\mu(i)} \frac{y_i}{\sqrt{\mu(i)}} \right)^2 && \text{Cauchy-Schwarz} \\
&= \frac{1}{|\mu|} \|y\|_1^2
\end{aligned}$$

Therefore $R(\mu)$ is strongly convex on the given domain, for the given norm. □

We can now relate the losses $M(\hat{\mu}_T, Q_t)$ and $M(\hat{\mu}_{T+1}, Q_t)$.

Lemma 2.9.5. *Let $R(\mu) = \mathrm{KL}(\mu \parallel \mu_1)$, which is $1/\kappa n$-strongly convex with respect to the ℓ_1 norm on Γ, the set of κ-dense measures. Then for all $T \geq 1$*

$$M(\hat{\mu}_T, Q_t) - M(\hat{\mu}_{T+1}, Q_t) \leq \lambda$$

Proof. We first note that $M(\hat{\mu}_T, Q_t) - M(\hat{\mu}_{T+1}, Q_t) = \frac{1}{\kappa n}\langle \mu_T - \mu_{T+1}, \ell_T \rangle$. So it suffices to show that

$$\sum_i (\mu_T(i) - \mu_{T+1}(i))\ell_T(i) \leq \kappa n \lambda$$

which, because $\ell_T(i) \in [0,1]$, is implied by $\|\mu_t(i) - \mu_{t+1}(i)\|_1 \leq \kappa n \lambda$.

Our strong convexity assumption on R and an application of Bregman's theorem (Theorem 2.2.3) give us that

$$\begin{aligned}
\tfrac{1}{\kappa n}\|\mu_T - \mu_{T+1}\|_1^2 &\leq \langle \nabla R(\mu_T) - \nabla R(\mu_{T+1}), \mu_T - \mu_{T+1} \rangle \\
&= \mathrm{KL}(\mu_T \parallel \mu_{T+1}) + \mathrm{KL}(\mu_{T+1} \parallel \mu_T) \\
&\leq \mathrm{KL}(\mu_T \parallel \tilde{\mu}_{T+1}) - \mathrm{KL}(\mu_{T+1} \parallel \tilde{\mu}_{T+1}) + \mathrm{KL}(\mu_{T+1} \parallel \tilde{\mu}_T) - \mathrm{KL}(\mu_T \parallel \tilde{\mu}_T) \\
&= \sum_i \mu_T(i)\left(\log \frac{\mu_T(i)}{\tilde{\mu}_{T+1}(i)} - \log \frac{\mu_T(i)}{\tilde{\mu}_T(i)}\right) - \sum_i \mu_{T+1}(i)\left(\log \frac{\mu_{T+1}(i)}{\tilde{\mu}_{T+1}(i)} - \log \frac{\mu_{T+1}(i)}{\tilde{\mu}_T(i)}\right) \\
&= \sum_i \mu_T(i)\left(\log \frac{\tilde{\mu}_T(i)}{\tilde{\mu}_{T+1}(i)}\right) - \sum_i \mu_{T+1}(i)\left(\log \frac{\tilde{\mu}_T(i)}{\tilde{\mu}_{T+1}(i)}\right) \\
&= \langle \mu_T - \mu_{T+1}, \lambda \ell_{T+1} \rangle \\
&\leq \|\mu_T - \mu_{T+1}\|_1 \|\lambda \ell_{T+1}\|_\infty
\end{aligned}$$

Therefore

$$\tfrac{1}{\kappa n}\|\mu_T - \mu_{T+1}\|_1 \leq \lambda \|\ell_{T+1}\|_\infty \leq \lambda$$

and so

$$\|\mu_T - \mu_{T+1}\|_1 \leq \lambda \kappa n.$$

□

We are now ready to prove the regret bound for the lazy dense update process of Algorithm 9, stated earlier in a simplified form as Lemma 2.8.1.

Lemma 2.9.6. *Let Γ be the set of κ-dense measures. Let μ_1 be the uniform measure over $\mathscr{P}$ of density κ ($\mu_1(x) = \kappa$ for all $x \in \mathscr{P}$), and let $|\mathscr{P}| = n$. Then for all $\mu \in \Gamma$ we have the following regret bound.*

$$\frac{1}{T}\sum_{t=1}^{T} M(\hat{\mu}_t, Q_t) \leq \frac{1}{T}\sum_{t=1}^{T} M(\hat{\mu}, Q_t) + \lambda + \frac{\mathrm{KL}(\mu \parallel \mu_1)}{\lambda \kappa n T}$$

Proof. From Lemma 2.9.3, we have that

$$\frac{1}{T}\sum_{t=1}^{T} M(\hat{\mu}_{t+1}, Q_t) \leq \frac{1}{T}\sum_{t=1}^{T} M(\hat{\mu}, Q_t) + \frac{\mathrm{KL}(\mu \parallel \mu_1)}{\lambda \kappa n T}$$

Adding $\frac{1}{T}\sum_{t=1}^{T} M(\hat{\mu}_t, Q_t)$ to both sides of the inequality and rearranging gives

$$\frac{1}{T}\sum_{t=1}^{T} M(\hat{\mu}_t, Q_t) \leq \frac{1}{T}\sum_{t=1}^{T} M(\hat{\mu}, Q_t) + \frac{1}{T}\sum_{t=1}^{T} (M(\hat{\mu}_t, Q_t) - M(\hat{\mu}_{t+1}, Q_t)) + \frac{\mathrm{KL}(\mu \parallel \mu_1)}{\lambda \kappa n T}.$$

We may then apply Lemma 2.9.5 to conclude

$$\frac{1}{T}\sum_{t=1}^{T} M(\hat{\mu}_t, Q_t) \leq \frac{1}{T}\sum_{t=1}^{T} M(\hat{\mu}, Q_t) + \lambda + \frac{\mathrm{KL}(\mu \parallel \mu_1)}{\lambda \kappa n T}.$$

□

2.10 Previous Work on Algorithmic Stability and Boosting

There are many complementary explanations for generalization of boosting: VC theory, greedy optimization of exponential loss on the training set [Breiman, 1999], entropy projection [Kivinen and Warmuth, 1999], and effect of boosting on the distribution of margins in the training set [Schapire et al., 1997]. Each genre of analysis represents a rich study of some aspect of boosting.

Groundbreaking work of Bousquet and Elisseeff [2002] showed that "stable" learning algorithms algorithms generalize well to unseen samples, *without* appealing to combinatorial (e.g., Rademacher complexity) bounds on the underlying hypothesis class. Intuitively, a *stable* learning algorithm does not change the output hypothesis "too much" when the training set changes by a small amount.

Because so many ideas in learning theory have been successfully applied to boosting, it is natural to ask if algorithmic stability can be used to analyze boosting as well. Below we list some work along these lines. Overall, it does not appear to have been nearly as successful as other techniques — possibly because the work so far has only studied AdaBoost, which is far more sensitive to noise and outliers than the smooth boosting studied here. That there works were able to obtain any kind of stability for AdaBoost is surprising.

In this work, we are at least able to use a *very strong* algorithmic stability condition (differential privacy) to prove generalization theorems for *smooth* boosting procedures and obtain sample bounds that are comparable to those from other theories. This suggests that we could work to relax these assumptions to approach a stability analysis of real-world boosting "from the top down."

- Gao and Zhou [2010] is able to show that AdaBoost (as a classifier) satisfies a new notion of stability. No concrete sample bounds are derived from the analysis.

- Kutin and Niyogi [2001] obtains stability bounds for AdaBoost as a regressor. Stability bounds from VC-dimension d of the base class are linear in d, their bounds are exponential in d. *However*, their bounds still apply when VC-dimension is infinite. No concrete sample bounds are derived from the analysis.

- Kutin and Niyogi [2002] studies relationships between various notions of algorithmic stability, and lists obtaining stability results for boosting algorithms as an interesting open question.

- "Stability and Convergence Trade-off of Iterative Optimization Algorithms" asks for such trade-offs for boosting algorithms. They do not obtain any such trade-offs for boosting-style iterated optimization.

- “Bagging Regularizes” notes that bagging can *improve* the stability of less-stable weak learners. They ask if the same can be proved for boosting. We are far from this: privacy degrades in our analysis of boosting. But their analysis uses a sub-sampling trick to improve stability; if weak learners are on disjoint subsets then only one weak hypothesis can be changed by a perturbation in the training set, thus the ensemble is more stable than the single base learner. Can we analyses boosting in terms of KL divergence between intermediate distributions? IE: the larger the divergence between intermediate distributions, the more disjoint in the weak learners the impact of a perturbation of the training set is.

This chapter, in full, is a reprint of the material as it appears in Conference on Learning Theory 2022. Bun, Mark; Carmosino, Marco; Sorrell, Jessica. “Efficient, Noise-Tolerant, and Private Learning via Boosting”. The dissertation author was the primary investigator and author of this material.

Chapter 3

Using Privacy to Secure Approximate Fully-Homomorphic Encryption

3.1 Introduction

Fully homomorphic encryption (FHE) on *approximate* numbers, proposed by Cheon, Kim, Kim and Song in Cheon et al. [2017], has attracted much attention in the past few years as a method to improve the efficiency of computing on encrypted data in a wide range of applications (like privacy preserving machine learning) where approximate results are acceptable Cheon et al. [2018d,c,b,e,a], Han et al. [2019], Park et al. [2019]. The CKKS scheme Cheon et al. [2017], just like most other (homomorphic) encryption schemes based on lattices, can be proved to satisfy the well established security notion of *indistinguishability under chosen plaintext attack* (IND-CPA) Goldwasser and Micali [1984] under widely accepted complexity assumptions, like the average-case hardness of the *Learning With Errors (LWE)* problem or the worst-case complexity of computational problems on (algebraic) point lattices Regev [2009], Lyubashevsky et al. [2013], Peikert et al. [2017], Peikert [2009].

Recently Li and Micciancio Li and Micciancio [2021] have shown that the traditional formulation of IND-CPA security is inadequate to capture security of approximate encryption against passive attacks, and demonstrated that the CKKS scheme is susceptible to a very efficient total key recovery attack, mounted by a passive adversary. The problem highlighted in Li and Micciancio [2021] is not with the IND-CPA security definition per se, which remains a good and well accepted definition for exact FHE schemes, but with the specifics of approximate decryption, which may inadvertently leak information about the secret key even when used by honest parties. The work Li and Micciancio [2021] also proposes a new, enhanced formulation of IND-CPA security (called IND-CPA$^{\mathsf{D}}$, or IND-CPA *with decryption oracles*), which properly captures the capabilities of a passive attacker against an *approximate* FHE scheme, and is equivalent to the standard notion of IND-CPA security for encryption schemes with exact decryption. The work Li and Micciancio [2021] also suggested some practical countermeasures to avoid their attack, and all major open source libraries implementing CKKS (e.g., SEAL, HElib, PALISADE, Lattigo) included similar countermeasures shortly after the results in Li and Micciancio [2021] were made public. However, neither Li and Micciancio [2021] nor any of these libraries present a solution that provably achieves the IND-CPA$^{\mathsf{D}}$ security definition proposed in Li and Micciancio [2021], leaving it as an open problem.

3.1.1 Contributions

In this chapter we show how to achieve IND-CPA$^{\mathsf{D}}$ security in a provable way. More specifically, we present a general technique to transform any approximate FHE scheme satisfying the (weak) IND-CPA security notion into one achieving the strong IND-CPA$^{\mathsf{D}}$ security definition proposed in Li and Micciancio [2021]. We then demonstrate how to apply the technique to the specific case of the CKKS scheme, which is the most prominent example of approximate homomorphic encryption.

Our technique works by combining a given (approximate) FHE scheme with another fundamental tool from the cryptographers' toolbox: differential privacy. The construction is very simple and intuitive: given an approximate FHE scheme (like CKKS), we modify the decryption function by post-processing its output (the decrypted message) with a properly chosen differentially private mechanism. Using differential privacy to limit the key leakage of approximate decryption is a fairly natural idea, and it is essentially the intuition behind the practical countermeasures proposed in Li and Micciancio [2021] and implemented by the libraries. But formally analyzing the method and provably achieving IND-CPA$^{\mathsf{D}}$ security raises a number of technical challenges:

- The Hamming metric, commonly used to define and analyze differentially private mechanisms, is not well suited to the setting of (lattice based) homomorphic encryption.
- Similarly, the Laplace noise commonly used and studied in the standard setting of differential privacy is not a good match for our target application, as it is both associated with the wrong norm (ℓ_1, rather than ℓ_2 or ℓ_∞), and has heavier tails than, e.g., the Gaussian distribution, and so will give worse bounds on the error introduced by post-processing.
- Formally proving the security of our construction requires a careful definition of what it means for an FHE scheme to be *approximate*. Previous works Cheon et al. [2017], Li and Micciancio [2021] simply defined approximate FHE as an encryption scheme which *does not* satisfy the correctness requirement

$$\mathsf{Dec}(\mathsf{Eval}(f, \mathsf{Enc}(m_1), \ldots, \mathsf{Enc}(m_k))) = f(m_1, \ldots, m_k) \tag{3.1}$$

  without imposing any specific limitation on how a scheme may deviate from it.
- Perturbing the output of the decryption function with a differentially private mechanism comes at the cost of lowering the output quality, making the result of the (already approximate) decryption function even less accurate, highlighting the necessity of carefully tuning the amount of noise added.
- The minimal security level considered acceptable by applications in practice typically depends on whether the cryptographic primitive is statistically secure (against computationally unbounded adversaries) or computationally secure (in which case a higher security margin is advisable to anticipate possible algorithmic or implementation improvements in the attacks.) Our application of statistical security tools (differential privacy) to encryption seems to require the instantiation of statistical security with the high security parameters of a computational encryption scheme.

In order to address the above obstacles, we

- provide a general definition of differential privacy, parameterized by an arbitrary norm, and then instantiate it with the Euclidean norm for the case of lattice-based encryption;
- employ a differentially private mechanism (for the Euclidean norm) based on Gaussian noise, which blends well with the probability distributions used in lattice cryptography;
- give formal definitions of *approximate* FHE, which provide precise guarantees on the output quality of the (approximate) decryption function. In fact, we identify two possible definitions, based on what we call *static* and *dynamic* noise estimates, and show that they result in quite different security properties (more on this below);
- use KL-divergence and other probabilistic tools to carefully calibrate the mechanism noise to the output quality, showing that $\Theta(\kappa)$ bits of noise are required to formally achieve κ-bit $\mathsf{IND\text{-}CPA^D}$ security;
- present and use a finer grained definition of bit-security that distinguishes between a computational security parameter c and a statistical one s, which can be set to a lower value than c (more on this below).

We first elaborate on our definition of *approximate* FHE. Previous works Cheon et al. [2017], Li and Micciancio [2021] did not include a precise definition of what it means for an encryption scheme (or decryption function) to be approximate, because the quality of the approximation (and more generally, the definition of the decryption function itself) does not impact the $\mathsf{IND\text{-}CPA}$ security of a scheme. This is contrasted with our work, where bounding the approximation quality of the decryption function plays a critical role in our analysis. Generally speaking, an approximate FHE scheme provides a guarantee (upper bound) on how much the output of the decryption function $\mathsf{Dec}(\mathsf{Eval}(f, \mathsf{Enc}(m_1), \ldots, \mathsf{Enc}(m_k)))$ may deviate from the output of the computation $f(m_1, \ldots, m_k)$. We distinguish two types of approximate FHE:

- Approximate FHE with *static* noise estimates, where this bound can be publicly computed as a function of the homomorphic computation f performed on the input ciphertexts. This is, for example, the type of noise estimates used in the HElib library HElib.

- Approximate FHE with *dynamic* noise estimates, where this bound is computed by the decryption function Dec using also the input ciphertext and the secret key. An ingenious method for dynamic noise estimation has been proposed by the PALISADE library PALISADE.

Most of our results, like our general framework based on differential privacy and a provably $\mathsf{IND\text{-}CPA}^{\mathsf{D}}$ secure variant of the CKKS approximate FHE scheme, are in the setting of static noise estimates. In this setting, we are able to establish the security of our generic construction (Theorem 3.3.3), and provide precise security guarantees for the modified approximate FHE scheme, showing that if the original scheme is κ-bit $\mathsf{IND\text{-}CPA}$ secure, then combining it with an appropriate differentially private mechanism achieves $\kappa - 8$ bits of security against the stronger $\mathsf{IND\text{-}CPA}^{\mathsf{D}}$ security definition, losing only 8 bits of security (Theorem 3.3.3). The amount of noise required to achieve this result is quantified by the notion of ρ-KLDP (Kullback-Leibler Differential Privacy), for a sufficiently small value of ρ. Our analysis is nearly tight for the CKKS scheme, in the sense that if one uses a substantially smaller amount of noise, we are able to exhibit an attack that breaks $\mathsf{IND\text{-}CPA}^{\mathsf{D}}$ security (Theorem 3.4.5).

3.2 FHE Preliminaries

We recall some notions and known results.

3.2.1 Bit Security

We use the notion of bit security from Micciancio and Walter [2018], which we briefly review below.

Definition 3.2.1 (Indistinguishability Game). Let $\{\mathscr{D}^0_\theta\}_\theta$, $\{\mathscr{D}^1_\theta\}_\theta$ be two distribution ensembles. The indistinguishability game is defined as follows: the challenger C chooses $b \leftarrow \mathscr{U}(\{0,1\})$. At any time after that the adversary A may send (adaptively chosen) query strings θ_i to C, and obtain samples $c_i \leftarrow \mathscr{D}^b_{\theta_i}$. The goal of the adversary is to output $b' = b$.

Definition 3.2.2 (Bit Security)**.** For any adversary A playing an indistinguishability game $\mathscr{G}$, we define its

- output probability as $\alpha^A = Pr[A \neq \perp]$, and its
- conditional success probability as $\beta^A = Pr[b' = b | A \neq \perp]$,

where the probabilities are taken over the randomness of the entire indistinguishability game (including the internal randomness of A). We also define A's

- conditional distinguishing advantage as $\delta^A = 2\beta^A - 1$, and
- the advantage of A as $\mathrm{adv}^A = \alpha^A(\delta^A)^2$.

The bit security of the indistinguishability game is $\min_A \log_2 \frac{T(A)}{\mathrm{adv}^A}$, where $T(A)$ is the running time of A.

As argued in Micciancio and Walter [2018], this is the correct way to define bit security for decision problems. Notice quadratic scaling with δ^A, rather than the linear scaling used for search problems. For additional motivation for the quadratic dependency, we note it matches known sample complexity lower bounds for distinguishing distributions that are close in the total variation distance, see Section 5.2 of Canonne [2020].

Lemma 3.2.3 (Lemma 2 of Micciancio and Walter [2018])**.** *Let $\mathscr{H}_i$ be k distributions and $\mathscr{G}_{i,j}$ be the indistinguishability game instantiated with $\mathscr{H}_i$ and $\mathscr{H}_j$. Let C be a fixed constant. Let $\varepsilon_{i,j} = \max_A \mathrm{adv}^A$ over all adversaries A against $\mathscr{G}_{i,j}$ with $T(A) \leq C$. Then $\varepsilon_{1,k} \leq 3k \sum_{i=1}^{k-1} \varepsilon_{i,i+1}$.*

The two distributions to be distinguished in a game $\mathscr{G}$ sometimes both post-process samples from some other probability ensamble $\mathscr{P}_\theta$. The following theorem bounds the loss of bit security of $\mathscr{G}$ if we replace $\mathscr{P}$ with another distribution $\mathscr{Q}$.

Theorem 3.2.4 (Theorem 8 of Micciancio and Walter [2018])**.** *Let $\mathscr{G}^{\mathscr{P}}$ be an indistinguishability game with black-box access to a probability ensemble $\mathscr{P}_\theta$. If $\mathscr{G}^{\mathscr{P}_\theta}$ is κ-bit secure, and $\max_\theta D(\mathscr{P}_\theta || \mathscr{Q}_\theta) \leq 2^{-\kappa+1}$, then $\mathscr{G}^{\mathscr{Q}_\theta}$ is $(\kappa - 8)$-bit secure.*

The aforementioned theorem is stated more generally in Micciancio and Walter [2018]. Our specialization of it requires that $\delta(\mathscr{P}||\mathscr{Q}) = \sqrt{D(\mathscr{P}||\mathscr{Q})/2}$ is what Micciancio and Walter [2018] calls a λ-efficient measure, which is implicit in Bai et al. [2018] and Pöppelmann et al. [2014].

We will need a few novel bounds on the quantities previously mentioned in this sub-section. These bounds are simplest to describe in terms of the following divergence.

Definition 3.2.5 (Bit Security Divergence)**.** Let $\mathscr{X}, \mathscr{Y}$ be random variables supported on X. The *bit security divergence* between $\mathscr{X}$ and $\mathscr{Y}$ is the quantity

$$\delta_{\mathsf{BS}}(\mathscr{X}, \mathscr{Y}) = \sup_{S \subseteq X} \frac{\mathbf{Pr}_{\mathscr{X}}[S] + \mathbf{Pr}_{\mathscr{Y}}[S]}{2} \Delta(\mathscr{X}|S, \mathscr{Y}|S)^2,$$

where $\mathscr{X}|S, \mathscr{Y}|S$ are the conditional distributions of $\mathscr{X}, \mathscr{Y}$, conditioned on the event S.

It is straightforward to verify that this is indeed a divergence, and moreover it is symmetric (which is why we write $\delta_{\mathsf{BS}}(\cdot,\cdot)$ rather than $\delta_{\mathsf{BS}}(\cdot||\cdot)$). It is not a metric, as the $O(k)$ factor in Lemma 3.2.3 is known to be tight, which is incompatible with $\delta_{\mathsf{BS}}(\cdot,\cdot)$ satisfying a triangle inequality.

$\delta_{\mathsf{BS}}(\cdot,\cdot)$ captures the advantage of an optimal (potentially computationally unbounded) adversary that aborts on the set S^c, and therefore can be seen as an extension of the standard total variation distance to the framework of Micciancio and Walter [2018]. We will need the following novel Pinsker-like bound on this quantity.

Lemma 3.2.6. *Let $\mathscr{X}, \mathscr{Y}$ be random variables supported on X. Then $\delta_{\mathsf{BS}}(\mathscr{X}, \mathscr{Y}) \leq D(\mathscr{X}||\mathscr{Y})/2$.*

We can use this to bound the advantage of *computationally unbounded* adversaries in the indistinguishability game.

Lemma 3.2.7. *Let $\mathscr{G}$ be the indistinguishability game instantiated with distribution ensembles $\{\mathscr{X}_\theta\}_\theta, \{\mathscr{Y}_\theta\}_\theta$, where $\theta \in \Theta$. Let $q \in \mathbb{N}$. Then, for any (potentially computationally unbounded) adversary A making at most q queries to its oracle, we have that*

$$\mathrm{adv}^A \leq \frac{q}{2} \max_{\theta \in \Theta} D(\mathscr{X}_\theta || \mathscr{Y}_\theta). \tag{3.2}$$

Proof. View an (adaptive) adversary as an arbitrary distribution on query-response pairs $\mathscr{X}_{\hat{\theta}} := ((\hat{\theta}_1, \mathscr{X}_{\hat{\theta}_1}), \dots, (\hat{\theta}_q, \mathscr{X}_{\hat{\theta}_q}))$ (and similarly for $\mathscr{Y}_{\hat{\theta}}$). We then have that

$$\mathrm{adv}^A \leq \delta_{\mathsf{BS}}(\mathscr{X}_{\hat{\theta}}, \mathscr{Y}_{\hat{\theta}}) \leq \frac{1}{2} D(\mathscr{X}_{\hat{\theta}}, \mathscr{Y}_{\hat{\theta}}) \leq \frac{1}{2} \left\| \widehat{D}(\mathscr{X}_{\hat{\theta}}, \mathscr{Y}_{\hat{\theta}}) \right\|_1 \leq \frac{q}{2} \max_{\theta \in \Theta} D(\mathscr{X}_\theta || \mathscr{Y}_\theta). \tag{3.3}$$

□ □

3.2.2 Fully Homomorphic Encryption

We briefly review definitions related to FHE. For simplicity, we focus on public-key setting. In all our definitions, we denote the security parameter using κ.

Definition 3.2.8 (FHE Scheme)**.** A (public-key) homomorphic encryption scheme with plaintext space $\mathscr{M}$, ciphertext space $\mathscr{C}$, public key space $\mathscr{P}\mathscr{K}$, secret-key space $\mathscr{S}\mathscr{K}$, and space of evaluatable circuits $\mathscr{L}$ is a tuple of four probabilistic polynomial-time algorithms

$$\mathsf{KeyGen} : 1^{\mathbb{N}} \to \mathscr{P}\mathscr{K} \times \mathscr{S}\mathscr{K}$$

$$\mathsf{Enc} : \mathscr{P}\mathscr{K} \times \mathscr{M} \to \mathscr{C}$$

$$\mathsf{Dec} : \mathscr{S}\mathscr{K} \times \mathscr{C} \to \mathscr{M}$$

$$\mathsf{Eval} : \mathscr{P}\mathscr{K} \times \mathscr{L} \times \mathscr{C} \to \mathscr{C}$$

Typically the public key naturally splits into two components, one used by Enc and one used by Eval. This separation is used to minimize the storage requirements of encryption (as the evaluation key is often quite large), and has no impact on security, so for simplicity we model both Enc and Eval as taking as input the same public key.

Standard FHE schemes are expected to satisfy the following notion of correctness.

Definition 3.2.9 (Correctness)**.** An FHE scheme $\Pi = (\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec}, \mathsf{Eval})$ is *correct* for some class of circuits $\mathscr{L}$ if for all $m_1, \dots, m_k \in \mathscr{M}$, for all $C \in \mathscr{L}$, for all $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{KeyGen}(1^\kappa)$, we have that

$$\mathsf{Dec}_{\mathsf{sk}}(\mathsf{Eval}_{\mathsf{pk}}(C, \mathsf{Enc}_{\mathsf{pk}}(m_1), \dots, \mathsf{Enc}_{\mathsf{pk}}(m_k))) = C(m_1, \dots, m_k). \tag{3.4}$$

One can relax the notion of correctness to *statistical* correctness, where the above identity only holds with high probability (over the random coins of Enc and Eval). We will not make a distinction between these two notions.

The work Cheon et al. [2017] introduced an "approximate" FHE scheme (CKKS), for which Equation (3.4) does not hold. The security implications of this relaxation are investigated in Li and Micciancio [2021], as discussed below. However, neither Cheon et al. [2017] nor Li and Micciancio [2021] provide a formal definition of an "approximate" FHE scheme, and instead simply drop the correctness requirement (3.4) without any further restriction. This is despite the CKKS scheme satisfying an approximate version of the correctness property of Equation (3.4).

The definition of *approximately correct* FHE scheme plays a fundamental role in our work. Informally, an approximately correct FHE scheme allows for meaningful, but inexact, computation on encrypted messages. To formalize the relaxed correctness requirements of an approximately correct FHE scheme, we first define the *ciphertext error*, which specifies the extent to which a homomorphic computation fails to be exact.

Definition 3.2.10 (Ciphertext Error)**.** Let $\Pi = (\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec}, \mathsf{Eval})$ be an FHE scheme with message space $\mathscr{M} \subseteq \widetilde{\mathscr{M}}$, which is a normed space with norm $\|\cdot\| : \widetilde{\mathscr{M}} \to \mathbb{R}_{\geq 0}$. For any ciphertext ct, secret key sk, and message m, the *ciphertext error* of $(\mathsf{ct}, m, \mathsf{sk})$ is defined to be

$$\mathsf{Error}(\mathsf{ct}, m, \mathsf{sk}) = \|\mathsf{Dec}_{\mathsf{sk}}(\mathsf{ct}) - m\|. \tag{3.5}$$

Typically, for some circuit $C \in \mathscr{L}$, key pair $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{KeyGen}(1^\kappa)$, and input values $m_1, \ldots, m_k \in \mathscr{M}$, one is interested in the quantity $\mathsf{Error}(\mathsf{ct}, m, \mathsf{sk})$ for

$$m = C(m_1, \ldots, m_k), \quad \text{and,} \quad \mathsf{ct} = \mathsf{Eval}_{\mathsf{pk}}(C, \mathsf{Enc}_{\mathsf{pk}}(m_1), \ldots, \mathsf{Enc}_{\mathsf{pk}}(m_k)),$$

i.e. where m and ct correspond to the same computation done on plaintexts and ciphertexts.

In this work we investigate two distinct correctness properties for approximate homomorphic encryption. The first is implicit in the literature on CKKS. We call this notion "static" because it does not rely on dynamic estimates of ciphertext error.

Definition 3.2.11 (Static Approximate Correctness)**.** Let Π be an FHE scheme with message space $\mathscr{M} \subseteq \widetilde{\mathscr{M}}$, which is a normed space with norm $\|\cdot\| : \widetilde{\mathscr{M}} \to \mathbb{R}_{\geq 0}$. Let $\mathscr{L}$ be a space of circuits, $\mathscr{L}_k \subseteq \mathscr{L}$ the subset of parity k circuits, and let $\mathsf{Estimate} : \bigsqcup_{k\in\mathbb{N}} \mathscr{L}_k \times \mathbb{R}^k_{\geq 0} \to \mathbb{R}_{\geq 0}$ be an efficiently computable function. We call the tuple $\tilde{\Pi} = (\Pi, \mathsf{Estimate})$ a *statically approximate* FHE scheme if for all $k \in \mathbb{N}$, for all $C \in \mathscr{L}_k$, for all $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{KeyGen}(1^\kappa)$, if $\mathsf{ct}_1, \ldots, \mathsf{ct}_k$ and $m_1, \ldots, m_k$ are such that $\mathsf{Error}(\mathsf{ct}_i, m_i, \mathsf{sk}) \leq t_i$, then

$$\mathsf{Error}(\mathsf{Eval}_{\mathsf{pk}}(C, \mathsf{ct}_1, \ldots, \mathsf{ct}_k), C(m_1, \ldots, m_k), \mathsf{sk}) \leq \mathsf{Estimate}(C, t_1, \ldots, t_k).$$

Note that the type signature $\bigsqcup_{k\in\mathbb{N}} \mathscr{L}_k \times \mathbb{R}^k_{\geq 0} \to \mathbb{R}_{\geq 0}$ encodes that $\mathsf{Estimate}$ takes as input a circuit C, and an error bound t_i for each of the k input wires to the circuit $C \in \mathscr{L}_k$. This correctness notion is "static" in the sense of static typing. In particular, $\mathsf{Estimate}$ only depends on

- the computation C to be done, and
- error bounds t_i for the inputs to the homomorphic computation.

All of these quantities are publicly computable given an abstract description of a computation, and (for non-adaptive computations) can even be precomputed (say by an FHE "compiler").

Generally $\mathsf{Estimate}(\cdot)$ either computes a (provable) worst-case bound on the error, or a (heuristic) average-case bound. This thesis assumes worst-case bounds. Approximate FHE schemes often require that all $m_1, \ldots, m_k$ are of bounded norm — this can be captured in the above definition by choosing $\mathscr{M}$ to be a set of bounded norm.

Security

We use the following security definition, proposed in Li and Micciancio [2021], which properly captures security of approximate FHE schemes against passive attacks.

Algorithm 10. Oracles for the $\mathsf{IND\text{-}CPA}^{\mathsf{D}}$ game.

```
initialization
    $(\mathsf{pk}, \mathsf{sk}) \leftarrow \mathsf{KeyGen}(1^{\kappa})$
global state
    $S \leftarrow \emptyset$
    $i \leftarrow 0$
$\mathsf{E}^b_{\mathsf{pk}}(m_0, m_1) :=$
    $\mathsf{ct} \leftarrow \mathsf{Enc}_{\mathsf{pk}}(m_b)$
    $S[i] \leftarrow (m_0, m_1, \mathsf{ct})$
    $i \leftarrow i + 1$
    return $\mathsf{ct}$
$\mathsf{H}^b_{\mathsf{pk}}(g, \mathbf{J} = (j_1, \ldots, j_k)) :=$
    $\mathsf{ct} \leftarrow \mathsf{Eval}_{\mathsf{pk}}(g, S[j_1].\mathsf{ct}, \ldots, S[j_k].\mathsf{ct})$
    $gm_0 \leftarrow g(S[j_1].m_0, \ldots, S[j_k].m_0)$
    $gm_1 \leftarrow g(S[j_1].m_1, \ldots, S[j_k].m_1)$
    $S[i] \leftarrow (gm_0, gm_1, \mathsf{ct})$
    $i \leftarrow i + 1$
    return $\mathsf{ct}$
$\mathsf{D}^b_{\mathsf{sk}}(i) :=$
    if $S[i].m_0 = S[i].m_1$
        return $\mathsf{Dec}_{\mathsf{sk}}(S[i].\mathsf{ct})$
    else
        return $\perp$
```

Definition 3.2.12 ($\mathsf{IND\text{-}CPA}^{\mathsf{D}}$ Security, Li and Micciancio [2021]). Let $\Pi = (\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec}, \mathsf{Eval})$ be a FHE scheme. We define the $\mathsf{IND\text{-}CPA}^{\mathsf{D}}$ game to be an indistinguishability game parameterized by distribution ensembles $\{(\mathsf{E}^b_\theta, \mathsf{H}^b_\theta, \mathsf{D}^b_\theta)\}_\theta$ for $b \in \{0,1\}$, where these oracles are the (stateful[1]) oracles given in Algorithm 10. The scheme Π is κ-bit *IND-CPA^D-secure* if for any A, we have that $\kappa \leq \log_2 \frac{T(A)}{\mathrm{adv}^A}$, where adv^A is as in Definition 3.2.2.

In Li and Micciancio [2021] it is also shown that for FHE schemes satisfying the standard correctness requirement (3.4), $\mathsf{IND\text{-}CPA}^{\mathsf{D}}$ security is equivalent to the traditional formulation of indistinguishability under chosen plaintext attack (IND-CPA), defined as follows.

Definition 3.2.13 (IND-CPA Security). Let $\Pi = (\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec}, \mathsf{Eval})$ be a FHE scheme. We define the IND-CPA game to be an indistinguishability game parameterized by distribution ensembles $\{\mathsf{E}^b_\theta\}_\theta$ for $b \in \{0,1\}$ of Algorithm 10. The scheme Π is κ-bit *IND-CPA-secure* if for any A,

[1]As a standard convention, if at any point in a game the adversary makes an invalid query (*e.g.*, a circuit g not supported by the scheme, or indices out of range), the oracle simply returns an error symbol $\perp$.

we have that $\kappa \leq \log_2 \frac{T(A)}{\mathrm{adv}^A}$, where adv^A is as in Definition 3.2.2.

We will additionally use weaker and stronger variants of $\mathsf{IND\text{-}CPA}^{\mathsf{D}}$, informally defined as follows:

- q-$\mathsf{IND\text{-}CPA}^{\mathsf{D}}$ security. This is the same as $\mathsf{IND\text{-}CPA}^{\mathsf{D}}$ security, but restricted to adversaries that make at most $q(\kappa)$ queries to oracle D.
- KR^{D} security, or security against key recovery attacks. Here we modify the $\mathsf{IND\text{-}CPA}^{\mathsf{D}}$ game by restricting[2] the E oracle to queries of the form $\mathsf{E}(m,m)$, and requiring the adversary to output (at the end of the attack) a secret key sk', rather than the bit b'. The attack is successful if $\mathsf{sk} = \mathsf{sk}'$.

KR^{D} security is implied by $\mathsf{IND\text{-}CPA}^{\mathsf{D}}$ security, but it is much weaker, and it is not generally considered a satisfactory notion of security. Here (as in Li and Micciancio [2021]), KR^{D} security is used exclusively to show that certain schemes are not secure, making the insecurity results stronger.

3.3 A Differentially Private Approach to $\mathsf{IND\text{-}CPA}^{\mathsf{D}}$ Security

In this section we investigate achieving q-$\mathsf{IND\text{-}CPA}^{\mathsf{D}}$ security for statically approximate, $\mathsf{IND\text{-}CPA}$-secure FHE schemes $\tilde{\Pi}$. Our approach is to post-process decryptions of $\tilde{\Pi}$ with an appropriate notion of differential privacy. The noise added by this differentially private mechanism will suffice to information-theoretically hide the ciphertext error, allowing us to reduce our analysis to the case of exact FHE, where $\mathsf{IND\text{-}CPA}$ and q-$\mathsf{IND\text{-}CPA}^{\mathsf{D}}$ security are equivalent.

3.3.1 Our Notion of Differential Privacy

Our notion of differential privacy is a generalization of the notion of Rényi differential privacy Mironov [2017] to different norms[3]. As the tightest bounds in our setting occur in the simplest[4]

[2]This is without loss of generality, as the only point of general queries $\mathsf{E}(m,m')$ is to get information correlated with the secret bit b, which the game does not depend on.

[3]In Differential Privacy, "adjacent" values are typically measured in the Hamming norm, while for our purposes the ℓ_2 and ℓ_∞ norms are of primary interest.

[4]There is an alternative simplification of the Rényi divergence when $\alpha = \infty$ known as the *max-log distance* Micciancio and Walter [2017] with desirable properties, for example it is a metric, similarly to the statistical distance.

case when $\alpha = 1$, we present things solely in terms of this Rényi divergence, *i.e.* the KL divergence.

Definition 3.3.1 (Norm KL Differential Privacy)**.** For $t \in \mathbb{R}_{\geq 0}$, let $M_t : B \to C$ be a family of randomized algorithms, where B is a normed space with norm $\|\cdot\| : B \to \mathbb{R}_{\geq 0}$. Let $\rho \in \mathbb{R}$ be a privacy bound. We say that the family M_t is ρ-KL differentially private (ρ-KLDP) if, for all $x, x' \in B$ with $\|x - x'\| \leq t$,

$$D(M_t(x)||M_t(x')) \leq \rho. \tag{3.6}$$

Note that our mechanism M depends on a bound on the distance $\|x - x'\| \leq t$, which it uses (internally) to set parameters to meet the desired privacy bound. In the most common case of Gaussian noise, it will use noise of standard deviation $\sigma = \Omega(2^{\kappa/2}t)$ to achieve κ-bit security (Corollary 3.3.8).

As $\|x - x'\| = \|x' - x\|$ is itself symmetric, our definition is invariant under replacing $D(\mathscr{D}_0||\mathscr{D}_1)$ with $\max(D(\mathscr{D}_0||\mathscr{D}_1), D(\mathscr{D}_1||\mathscr{D}_0))$, and is therefore implicitly dependent on this larger (symmetric) measure, although we do not make this explicit in our work.

Algorithm 11. The FHE Scheme $M[\tilde{\Pi}]$

```
$\mathsf{Enc}'_{\mathsf{pk}}(m) :=$
    $c \leftarrow \mathsf{Enc}_{\mathsf{pk}}(m)$
    **return** $\mathsf{ct} = (c, t_e)$
$\mathsf{Eval}'_{\mathsf{pk}}(C, \mathsf{ct}'_1, \ldots, \mathsf{ct}'_k) :=$
    $c \leftarrow \mathsf{Eval}_{\mathsf{pk}}(C, \mathsf{ct}_1.c, \ldots, \mathsf{ct}_k.c)$
    $t \leftarrow \mathsf{Estimate}(C, \mathsf{ct}_1.t, \ldots, \mathsf{ct}_k.t)$
    **return** $\mathsf{ct} = (c, t)$

$\mathsf{Dec}'_{\mathsf{sk}}(\mathsf{ct}) :=$
    **return** $M_{\mathsf{ct}.t}(\mathsf{Dec}_{\mathsf{sk}}(\mathsf{ct}.c))$
```

Definition 3.3.2. Let $\Pi = (\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec}, \mathsf{Eval})$ be an FHE scheme with plaintext space $\mathscr{M} \subseteq \widetilde{\mathscr{M}}$, where $\widetilde{\mathscr{M}}$ is a normed space with norm $\|\cdot\|$. Let $\mathsf{Estimate}$ be such that $\tilde{\Pi} = (\Pi, \mathsf{Estimate})$ is statically approximate, and let t_e be an upper bound on ciphertext errors of all fresh encryptions

As our bounds degrade linearly in α as $\alpha \to \infty$, this notion is unsuitable for our situation.

$\mathsf{Enc}_{\mathsf{pk}}(m)$ for all $m \in \mathscr{M}$. Let M_t be a ρ-KLDP mechanism on $\widetilde{\mathscr{M}}$. Define the FHE scheme $M[\tilde{\Pi}]$ that has an identical KeyGen algorithm to Π, with the modified algorithms $\mathsf{Enc}'_{\mathsf{pk}}, \mathsf{Eval}'_{\mathsf{pk}}$, and $\mathsf{Dec}'_{\mathsf{sk}}$ of Algorithm 11.

In the above definition of the scheme $M[\tilde{\Pi}]$, we use the "tagged ciphertext" notation $\mathsf{ct} = (c,t)$, where c is an ordinary ciphertext and t is an estimated ciphertext error upper bound. An initial estimation t_e is provided by the encryption algorithm, and the evaluation algorithm updates the error upper bound using $\mathsf{Estimate}(\cdot)$ such that the resulting scheme is a statically approximate FHE scheme.

Algorithm 12. The decryption oracle for the game $\mathscr{G}_1$ of Theorem 3.3.3.

```
$\mathsf{D}(i) :=$
  **if** $S[i].m_0 = S[i].m_1$
    $t_i \leftarrow S[i].\mathsf{ct}.t$
    **return** $M_{t_i}(S[i].m_0)$
  **else**
    **return** $\perp$
```

Theorem 3.3.3. *Let* $\Pi = (\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec}, \mathsf{Eval})$ *be an FHE scheme with plaintext space* $\mathscr{M} \subseteq \widetilde{\mathscr{M}}$*, where* $\widetilde{\mathscr{M}}$ *is a normed space with norm* $\|\cdot\|$*. Let* $\mathsf{Estimate}$ *be such that* $\tilde{\Pi} = (\Pi, \mathsf{Estimate})$ *is statically approximate. Let* $\kappa > 0$*, let* M_t *be a* ρ*-KLDP mechanism on* $\widetilde{\mathscr{M}}$ *where* $\rho \leq 2^{-\kappa-7}/q$*, and let* $q \in \mathbb{N}$*. If* Π *is* $(\kappa+8)$*-bit secure in the IND-CPA game, then* $M[\tilde{\Pi}]$ *is* κ*-bit secure in the q-IND-CPA*^{D} *game.*

Proof. The theorem follows from combining Lemma 3.2.7 and Theorem 3.2.4. □

3.3.2 Gaussian Mechanism

In this section, we present and analyze a differentially private mechanism M_t which simply adds Gaussian noise to its input.

Definition 3.3.4. Let $\mu \in \mathbb{Z}$, and $\sigma > 0$. The discrete Gaussian of parameters μ, σ (written $\mathscr{N}_{\mathbb{Z}}(\mu, \sigma^2)$) is the probability distribution supported on $\mathbb{Z}$ with p.m.f. $p(x) \propto \exp(-(x-\mu)^2/2\sigma^2)$.

It is known how to (with high probability) exactly sample from this distribution in constant time Canonne et al. [2020].

Proposition 3.3.5 (Prop. 5 of Canonne et al. [2020])**.** *Let $\sigma \in \mathbb{R}_{\geq 0}$, and let $\mu, \nu \in \mathbb{Z}$. Then:*

$$D(\mathscr{N}_{\mathbb{Z}}(\mu, \sigma^2) || \mathscr{N}_{\mathbb{Z}}(\nu, \sigma^2)) = \frac{(\nu-\mu)^2}{2\sigma^2}. \tag{3.7}$$

Definition 3.3.6. Let $\rho > 0$, and $n \in \mathbb{N}$. Define the *(discrete) Gaussian Mechanism $M_t : \mathbb{Z}^n \to \mathbb{Z}^n$* be the mechanism that, on input $x \in \mathbb{Z}^n$, outputs a sample from $\mathscr{N}_{\mathbb{Z}^n}(x, \frac{t^2}{2\rho} I_n)$.

Lemma 3.3.7. *For any $\rho > 0, n \in \mathbb{N}$, the Gaussian mechanism is ρ-KLDP.*

Proof. Let $\mathscr{X} = \mathscr{N}_{\mathbb{Z}^n}(x, \frac{t^2}{2\rho} I_n)$ and $\mathscr{Y} = \mathscr{N}_{\mathbb{Z}^n}(y, \frac{t^2}{2\rho} I_n)$. By sub-additivity of the KL divergence and Proposition 3.3.5, we have that $D(\mathscr{X}||\mathscr{Y}) \leq \|\widehat{D}(\mathscr{X}||\mathscr{Y})\|_1 = \frac{\rho}{t^2}\|x-y\|_2^2 \leq \rho$.

$\square$

Corollary 3.3.8. *Let $\Pi = (\mathsf{KeyGen}, \mathsf{Enc}, \mathsf{Dec}, \mathsf{Eval})$ be an FHE scheme with plaintext space $\mathscr{M} \subseteq \widetilde{\mathscr{M}}$, where $\widetilde{\mathscr{M}} \subseteq \mathbb{Z}^n$ is a normed space with norm $\|\cdot\|$. Let $\mathsf{Estimate}$ be such that $\tilde{\Pi} = (\Pi, \mathsf{Estimate})$ is a statically approximate FHE scheme. Let M_t be the Gaussian mechanism (with $\rho := 2^{-\kappa-7}/q$). If Π is $(\kappa+8)$-bit secure in the $\mathsf{IND\text{-}CPA}$ game, then $M[\tilde{\Pi}]$ is κ-bit secure in the q-$\mathsf{IND\text{-}CPA}^{\mathsf{D}}$ game.*

For $\rho := 2^{-\kappa-7}/q$, one can check that the Gaussian mechanism adds noise of standard deviation $8\sqrt{q2^{\kappa}}\mathsf{ct}.t$ to each coordinate, so one loses $\kappa/2+3+\log_2\sqrt{q}+\log_2\mathsf{ct}.t$ bits of precision. As the ciphertext already contains $\log_2\mathsf{ct}.t$ bits of noise, the *additional* precision lost by $M[\tilde{\Pi}]$ is $\kappa/2+\log_2\sqrt{q}+3$ bits.

Proof. This reduces to combining Lemma 3.3.7 with Theorem 3.3.3. The size of the Gaussian noise comes from $\rho = \mathsf{ct}.t^2/2\sigma^2 \iff \sigma = \frac{1}{\sqrt{2\rho}}\mathsf{ct}.t$. As we need that $\rho \leq 2^{-\kappa-7}/q$, it follows that $\sigma \geq 8\sqrt{q}2^{\kappa/2}\mathsf{ct}.t$.

□

This transformation does not explicitly depend on the underlying parameters of the particular implementation of approximate encryption (for example, the size of the LWE moduli one is working over, the dimension of the message space, etc.), and instead only implicitly depends on these quantities via the computation of the static ciphertext error bound. We caution that to apply this result to CKKS one needs to be slightly careful about the underlying norm one is working with, which we do later in Theorem 3.4.3.

3.4 Application to CKKS

Prior work of Li and Micciancio [2021] shows that the approximate FHE scheme of Cheon et al. [2017] does not satisfy $\mathsf{IND\text{-}CPA^D}$-security, even though it satisfies $\mathsf{IND\text{-}CPA}$-security. We refer the reader to Li and Micciancio [2021] for additional details, but at a high level they show that publishing the results of an approximate FHE computation under CKKS leaks information about the secret key, enabling a full key recovery attack in the case of trivial computation, and an attack against $\mathsf{IND\text{-}CPA^D}$-security for more general homomorphic computation. In this section, we apply Theorem 3.3.3 and Lemma 3.3.7 to give a modification of the CKKS decryption function that allows us to prove $\mathsf{IND\text{-}CPA^D}$-security of the modified scheme.

We use the results of Section 3.3 to show that post-processing the results of the CKKS decryption function with the Gaussian mechanism is sufficient to achieve $\mathsf{IND\text{-}CPA^D}$-security for the CKKS scheme, for large enough Gaussian noise (Section 3.4.2). We also prove a nearly matching lower bound on the Gaussian noise necessary to achieve $\mathsf{IND\text{-}CPA^D}$-security for the CKKS scheme (Section 3.4.3).

3.4.1 The CKKS Approximate FHE Scheme

We begin with a few mathematical preliminaries necessary to the CKKS scheme. For any positive integer N, let $\Phi_N(X) = \prod_{j \in \mathbb{Z}_N^*}(X - \omega_N^j)$ be the Nth cyclotomic polynomial, where $\omega_N = e^{2\pi \mathrm{i}/N} \in \mathbb{C}$ is the complex Nth principal root of unity, and $\mathbb{Z}_N^*$ is the group of invertible integers modulo N. We recall that $\Phi_N(X) \in \mathbb{Z}[X]$ is a monic polynomial of degree $n = \varphi(N) = |\mathbb{Z}_N^*|$ with integer coefficients. We denote by $\mathscr{R}^N = \mathbb{Z}[X]/(\Phi_N(X))$ the ring of integers of the number field $\mathbb{Q}[X]/(\Phi_N(X))$, omitting the superscript when it is clear from context. We use $\mathscr{R}_Q^N = \mathbb{Z}[X]/(Q, \Phi_N(X))$ to denote the ring of elements of $\mathscr{R}^N$ reduced modulo Q.

An element $a \in \mathbb{R}[X]/(\Phi_N(X))$ may be embedded into $\mathbb{C}^n$ under the *canonical embedding* $\tau(a)$ (typically defined over $\mathbb{Q}[X]/(\Phi_N(X))$, but naturally extending to $\mathbb{R}[X]/(\Phi_N(X))$). The map $\tau(a)$ takes a to the $n = \varphi(N)$ evaluations of a at the n roots of $\Phi_N(X)$. Notice that these n values come in conjugate pairs and can be identified as a vector in $\mathbb{C}^{n/2}$ via a projection $\pi : (z, \bar{z}) \mapsto z$. So, complex vectors in $\mathbb{C}^{n/2}$ are considered as messages in CKKS, and they are encoded to plaintext polynomials in $\mathscr{R}$ by composing π^{-1} and τ^{-1} together with a scaling factor; conversely, plaintexts are decoded using $\tau \circ \pi$. We define the *canonical embedding norm* $\|\cdot\|_\infty^{\mathsf{can}}$ of an element $a \in \mathbb{R}[X]/(\Phi_N(X))$ to be $\|a\|_\infty^{\mathsf{can}} = \|\tau(a)\|_\infty$. We will use this norm to track the ciphertext error of CKKS ciphertexts.

We now present the relevant subroutines of the CKKS FHE scheme. We omit many details of the CKKS scheme, and refer the reader to Cheon et al. [2017] for a more complete description. The CKKS scheme is parameterized by a plaintext dimension $n/2$ (typically a power-of-two), a ciphertext modulus Q, and a discrete Gaussian error distribution χ with standard deviation σ.

- $\mathsf{CKKS.KeyGen}(1^\kappa)$: Take $w = w(\kappa)$ and $p = p(\kappa, Q)$. To generate the secret key sk, sample $s \leftarrow \{s \in \{-1,0,1\}^n : |s|_0 = w\}$ and take $\mathsf{sk} = (1, s)$. To generate the public key pk, sample $a \leftarrow \mathscr{R}_Q$, $e \leftarrow \chi$, and take $\mathsf{pk} = (b = -as + e, a)$. To generate the evaluation key ek, sample $a' \leftarrow \mathscr{R}_{pQ}$, $e' \leftarrow \chi$, and take $\mathsf{ek} = (b', a')$ for $b' = -a's + e' + ps^2 \mod pQ$. Return $(\mathsf{sk}, \mathsf{pk}, \mathsf{ek})$.

- $\mathsf{CKKS.Encode}(\mathbf{x} \in \mathbb{C}^{n/2}; \Delta)$: Return $\lfloor \Delta \cdot \varphi^{-1}(\mathbf{x}) \rceil \in \mathscr{R}$.

- $\mathsf{CKKS.Enc}_{\mathsf{pk}}(m)$: Let T denote the distribution over $\{0, \pm 1\}^n$ induced by sampling each coordinate independently, drawing -1 with probability 1/4, 1 with probability 1/4, and 0 with probability 1/2. Sample $r \leftarrow T$, $e_0, e_1 \leftarrow \chi$, and return $r \cdot \mathsf{pk} + (m + e_0, e_1) \mod Q$.

- $\mathsf{CKKS.Add}(\mathbf{c}_0, \mathbf{c}_1 \in \mathscr{R}_Q)$: Return $\mathbf{c}_0 + \mathbf{c}_1 \mod Q$.

- $\mathsf{CKKS.Mult}_{\mathsf{ek}}(\mathbf{c}_0, \mathbf{c}_1 \in \mathscr{R}_Q)$: For $\mathbf{c}_0 = (b_0, a_0)$ and $\mathbf{c}_1 = (b_1, a_1)$, let $(b_2, a_2) = (b_0 b_1, a_0 b_1 + a_1 b_0) + \lfloor p^{-1} \cdot a_0 a_1 \cdot \mathsf{ek} \rceil \mod Q$. Return (b_2, a_2).

- $\mathsf{CKKS.Decode}(a \in \mathscr{R}; \Delta)$: Return $\varphi(\Delta^{-1} \cdot a) \in \mathbb{C}^{n/2}$.

- $\mathsf{CKKS.Dec}_{\mathsf{sk}}(\mathbf{c})$: For $\mathbf{c} = (b, a) \in \mathscr{R}_Q^2$, return $b + as \mod Q$.

Note that CKKS supports encryption and decryption of floating-point inputs by pre-processing encryption with $\mathsf{CKKS.Encode}$, and post-processing decryption with $\mathsf{CKKS.Decode}$. All intermediate operations are then done with integer arithmetic. To simplify exposition, we focus on these intermediate operations, and therefore restrict to the case of integer arithmetic.

We will need the following (standard) expressions for how the ciphertext error transforms during addition and multiplication.

Lemma 3.4.1 (Error Growth Cheon et al. [2017])**.** *Let $\mathbf{c}_0$ and $\mathbf{c}_1$ denote two CKKS ciphertexts, with $\mathbf{c}_0 = \mathsf{CKKS.Enc}_{\mathsf{pk}}(m_0)$ and $\mathbf{c}_1 = \mathsf{CKKS.Enc}_{\mathsf{pk}}(m_1)$ with errors e_0 and e_1 respectively. Then the ciphertext $\mathbf{c}_{\mathsf{Mult}} = \mathsf{CKKS.Mult}(\mathbf{c}_0, \mathbf{c}_1)$ has error $m_0 e_1 + m_1 e_0 + e_0 e_1 + e_{\mathsf{Mult}}$ for a term e_{Mult} that depends on the parameters of the CKKS instance (and the ciphertexts $\mathbf{c}_0, \mathbf{c}_1$). The ciphertext $\mathbf{c}_{\mathsf{Add}} = \mathsf{CKKS.Add}(\mathbf{c}_0, \mathbf{c}_1)$ has error $e_0 + e_1$.*

Certain authors have suggested various heuristics for analyzing e_{Mult}. We will find the following one useful for the analysis of the attack of Section 3.4.3.

Heuristic 1 (Appendix A.5 of Gentry et al. [2012])**.** Let w be the hamming weight of sk. Then e_{Mult} may be modeled as a random variable with mean zero and variance $O(wn)$.

The rest of our work will benefit from the following notation.

Definition 3.4.2. For $\sigma > 0$, let $\mathsf{S\text{-}CKKS}_\sigma$ be the CKKS encryption scheme, where one modifies decryption to compute $\mathsf{S\text{-}CKKS}_\sigma.\mathsf{Dec}_{\mathsf{sk}}(\mathsf{ct}) = \mathsf{CKKS.Dec}_{\mathsf{sk}}(\mathsf{ct}.c) + \mathcal{N}_{\mathbb{Z}^n}(0, \sigma^2 \mathsf{ct}.t^2 I_n)$.

3.4.2 $\mathsf{IND\text{-}CPA^D}$-Secure CKKS

It is straightforward to apply Corollary 3.3.8 to CKKS to obtain q-$\mathsf{IND\text{-}CPA^D}$ security.

Theorem 3.4.3. *For any $q \in \mathbb{N}$, if CKKS is $(\kappa+8)$-bit IND-CPA-secure, and $\sigma = 8\sqrt{qn}2^{\kappa/2}$, then $\mathsf{S\text{-}CKKS}_\sigma$ is κ-bit q-IND-CPA^D-secure, i.e. $\kappa/2+\tilde{O}(1)$ additional bits of Gaussian noise suffice to achieve q-IND-CPA^D security.*

Proof. This follows immediately from Corollary 3.3.8, (using the aforementioned inequality $\|m\|_\infty^{\mathsf{can}} \leq \sqrt{n}\,\|m\|_2$, as our analysis of the Gaussian mechanism uses an ℓ_2 norm bound). $\square$

3.4.3 Lower Bound for Gaussian Mechanism

Together, Lemma 3.3.7 and Theorem 3.3.3 give an upper bound on the amount of Gaussian noise required to achieve $\mathsf{IND\text{-}CPA^D}$-security for an $\mathsf{IND\text{-}CPA}$-secure approximate encryption scheme. In this subsection, we show that this upper bound is essentially tight for CKKS by demonstrating an attack against $\mathsf{IND\text{-}CPA^D}$ security for noticeably smaller Gaussian noise, *i.e.* analyzing $\mathsf{S\text{-}CKKS}_{\sigma_s}$ for sanitization noise $\sigma_s \ll 8\sqrt{qn}2^{\kappa/2}$. In what follows, recall that $n = \varphi(N)$, and w denotes the Hamming weight of the key sk.

Algorithm 13. Adversary $A(1^\kappa, \mathsf{pk}, \mathsf{ek})$

```
  for $i \in \{0, \dots, 44\}$ do
      $\mathsf{ct}_i \leftarrow \mathsf{E}_{\mathsf{pk}}(m_i^{(0)} = 0, m_i^{(1)} = B)$;
  for $i \in \{45, \dots, 59\}$ do
      $\mathsf{ct}_i \leftarrow \mathsf{E}_{\mathsf{pk}}(m_i^{(0)} = 0, m_i^{(1)} = -B)$;
  $\mathsf{ct}_{60} \leftarrow \mathsf{H}_{\mathsf{ek}}(g, \{0, \dots, 59\})$ for $g(x_0, \dots, x_{59}) = \sum_{i=0}^{29}(x_i \cdot x_{30+i})$
  $m' \leftarrow \mathsf{D}_{\mathsf{sk}}(60)$
  $V_0 = 30\sigma^4 + O(wn) + \sigma_s^2$                          Variance of $\tau(m')_0$ if $b = 0$
  $V_1 = 30\sigma^4 + 60B^2\sigma^2 + O(wn) + \sigma_s^2$          Variance of $\tau(m')_0$ if $b = 1$
  if $|\tau(m')_0| < \sqrt{\frac{\log(V_1/V_0)V_0V_1}{V_1 - V_0}}$ then
      return 0
  else
      return 1
```

At a high level, the adversary A will exploit the message-dependence of the S-CKKS error growth (Lemma 3.4.1) to design an H query such that the expected magnitude of the ciphertext error of ct_{60} is larger when $b = 1$ than when $b = 0$. The adversary A will then query D on this ciphertext, and choose its bit based on the size of the message m' it receives.

We will next show that the aforementioned adversary will have noticeable advantage unless σ_s is larger than σ (the standard deviation of the underlying RLWE error) by a factor super-polynomial in the security parameter.

Lemma 3.4.4. *Let $\sigma_s > 0$. Then there exists an adversary A against* $\mathsf{S\text{-}CKKS}_{\sigma_s}$ *in the IND-CPA^D such that* $\mathrm{adv}^A = \Omega\left(\frac{1}{\sigma_s^4 n^6}\right)$.

Proof. We first observe that the ciphertext $\mathsf{ct}_{60} = \mathsf{Eval}_{\mathsf{ek}}(g, \{0, \dots, 59\})$ is an approximate encryption of 0 both when $b = 0$ and $b = 1$ in the IND-CPA$^\mathsf{D}$ experiment. Therefore the decryption query made by A returns a value rather than $\perp$.

If $b = 0$, then because all ciphertexts ct_i encrypt messages $m_i = 0$, the message-dependent terms of the error growth from Lemma 3.4.1 are also 0, and so the ciphertext error of ct_{60} is $\sum_{i=0}^{29} e_{\mathsf{Mult}} + e_i e_{30+i}$, where e_i denotes the ciphertext error of ct_i. Recall that if error vectors e and e' have entries sampled from a discrete Gaussian with parameter σ, then each of the components of $\tau(ee')$ is distributed with mean 0 and variance σ^4. We can then use the Central Limit Theorem to approximate the distribution of the sum $\sum_{i=0}^{29} e_{\mathsf{Mult}} + e_i e_{30+i}$ as a Gaussian distribution with mean 0 and variance $30\sigma^4 + O(wn)$. Note that this approximation can be improved by increasing the number of terms in the sum to a larger constant. For the sake of concreteness we have designed the adversary such that there are 30 terms, as this is the value at which the Central Limit Theorem is empirically justified.

If $b = 1$, then the message-dependent terms of the error growth are significant, and the error of ct_{60} is

$$\sum_{i=0}^{14} (e_{\mathsf{Mult}} + e_i e_{30+i} + Be_i + Be_{30+i}) + \sum_{i=15}^{29} (e_{\mathsf{Mult}} + e_i e_{30+i} - Be_i + Be_{30+i}).$$

As in the case where $b = 0$, we will approximate this distribution as a Gaussian with mean 0. Though the error terms $e_i e_{30+i}$ and $Be_i + Be_{30+i}$ are not independent, they do have covariance 0, as do the terms $e_i e_{30+i}$ and $Be_{30+i} - Be_i$, and so we can approximate the sum of errors as being drawn from a discrete Gaussian distribution with mean 0 and variance $30\sigma^4 + 60B^2\sigma^2 + O(wn)$.

The adversary sees the result of post-processing the error term with the Gaussian mechanism, run with parameter σ_s, and then chooses its bit to return based on the absolute value of the first component $\tau(m')_0$ under the canonical embedding. When $b = 0$, this means the adversary sees a sample drawn from a distribution that is well-approximated by a centered Gaussian with variance $V_0 = 30\sigma^4 + O(wn) + \sigma_s^2 \mathsf{ct}.t^2$. When $b = 1$, however, the adversary sees a sample drawn from a distribution that is well-approximated by a Gaussian with the same mean, but larger variance $V_1 = 30\sigma^4 + 60B^2\sigma^2 + O(wn) + \sigma_s^2 \mathsf{ct}.t^2$. Let

$$x = \sqrt{\frac{\log(V_1/V_0)V_0V_1}{V_1 - V_0}}.$$

A straightforward calculation shows that for $|\tau(m')_0| < x$, m' is a more likely outcome when $b = 0$

than when $b = 1$, and when $|\tau(m')_0| \geq x$, m' is at least as likely when $b = 1$ as it is when $b = 0$. Then we have that the advantage of adversary A is approximately the total variation distance between a Gaussian with variance V_0 and a Gaussian with variance V_1. By Lemma 1.1.6, we have that

$$\Delta(\mathcal{N}(0,V_0), \mathcal{N}(0,V_1)) \geq \frac{1}{200}\frac{|V_0 - V_1|}{V_0} \in \Theta\left(\frac{B^2\sigma^2}{\sigma^4 + wn + \sigma_s^2 \mathsf{ct}.t^2}\right).$$

Recall that w is the hamming weight of the secret key sk, and so we have $w < n$. For security, we know that $\sqrt{n} < \sigma$, and so it follows that the advantage of our (non-aborting) adversary A against the $\mathsf{IND\text{-}CPA^D}$ security of CKKS is the *square* of the total variation distance, *i.e.* $\Theta\left(\frac{B^4\sigma^4}{(\sigma^4+\sigma_s^2\mathsf{ct}.t^2)^2}\right)$. Finally, note that for $\|e_i\|_\infty^{\mathsf{can}} < \sigma n$ holds with high probability, so $\mathsf{ct}.t \leq O(B\sigma n^{3/2})$ (where we pick up a $\sqrt{n}$ factor to convert to the ℓ_2 norm), and therefore the advantage of our adversary is $\Theta\left(\frac{B^4\sigma^4}{\sigma^8+\sigma_s^4\sigma^4B^4n^6}\right) = \Omega\left(\frac{1}{\sigma_s^4 n^6}\right)$.

$\square$

Theorem 3.4.5. *If* $\mathsf{S\text{-}CKKS}_{\sigma_s}$ *is* κ*-bit IND-CPA*^{D}*-secure, then* $\sigma_s = \Omega(2^{\kappa/4}/n^{3/2})$*, i.e. one must add at least* $\kappa/4 - \tilde{\Omega}(1)$ *bits of additional Gaussian noise.*

Proof. We have that $\kappa \leq \log_2 O\left(\frac{T(A)}{\mathrm{adv}^A}\right) \leq \log_2 O(\sigma_s^4 n^6) \implies \sigma_s \geq 2^{\kappa/4}/n^{3/2}$, and therefore $\kappa/4 - \log_2 \Omega(n^{3/2}) \leq \log_2 \sigma_s$. $\square$

We therefore see that while one can potentially improve on the parameters for post-processing noise given in Theorem 3.4.3, the main (exponential) term is tight to within constant factors.

This chapter, in full, has been submitted for publication of the material as it may appear in Advances in Cryptology - CRYPTO 2022. Li, Baiyu; Micciancio, Daniele; Schultz, Mark; Sorrell, Jessica. “Securing Approximate Homomorphic Encryption Using Differential Privacy”. The dissertation author was the primary investigator and author of this material.

Chapter 4

Reproducibility in Learning

4.1 Introduction

Reproducibility is vital to ensuring scientific conclusions are reliable, and researchers have an obligation to ensure that their results are replicable. In the last twenty years, lack of reproducibility has been a major issue in nearly all scientific areas of study. For example, a 2012 Nature article by Begley and Ellis reported that the biotechnology company Amgen was only able to replicate 6 out of 53 landmark studies in haematology and oncology Begley and Ellis [2012]. In a 2016 Nature article, Baker published a survey of 1500 researchers, reporting that 70% of scientists had tried and failed to replicate the findings of another researcher, and that 52% believed there is a significant crisis in reproducibility Baker [2016].

A key issue underlying the reproducibility crisis (as articulated in many articles, e.g., Ioannidis [2005]) is the fact that new data/publications are growing at an exponential rate, giving rise to an explosion of methods for data generation, screening, testing, and analysis, where, crucially, only the combinations producing the most significant results are reported. Such practices (also known as P-hacking, data dredging, and researcher degrees of freedom) can lead to erroneous findings that appear to be significant, but that don't hold up when other researchers attempt to replicate them. Identifying and mitigating these problems is quite subtle. First, is not easy to come up with an agreed-upon set of practices that guarantees reproducibility, and secondly, testing to determine whether or not a finding is statistically significant is a complex task.

Within the subfields of machine learning and data science, there are similar concerns about the reliability of published findings. The performance of models produced by machine learning algorithms may be affected by the values of random seeds or hyperparameters chosen during training, and performance may be brittle to deviations from the values disseminated in published results Henderson et al. [2017], Islam et al. [2017], Lucic et al. [2018]. To begin addressing concerns about reproducibility, several prominent machine learning conferences have begun hosting reproducibility workshops and holding reproducibility challenges, to promote best practices and encourage researchers to share the code used to generate their results Pineau et al. [2020].

In this work, we aim to initiate the study of reproducibility as a property of algorithms themselves, rather than the process by which their results are collected and reported. We define the following notion of reproducibility, which informally says that a randomized algorithm is reproducible if two distinct runs of the algorithm on two sets of samples drawn from the same distribution, with internal randomness fixed between both runs, produces the *same* output with high probability.

Definition 4.1.1 (Reproducibility)**.** Let D be a distribution over a universe $\mathscr{X}$, and let $\mathscr{A}$ be a randomized algorithm with sample access to D. $\mathscr{A}(\mathbf{s})$ is *ρ-reproducible* if

$$\mathbf{Pr}_{\mathbf{s}_1,\mathbf{s}_2,r}[\mathscr{A}(\mathbf{s}_1;r) = \mathscr{A}(\mathbf{s}_2;r)] \geq 1-\rho,$$

where $\mathbf{s}_1$ and $\mathbf{s}_2$ denote sequences of samples drawn i.i.d. from D, and r denotes a random binary string representing the internal randomness used by $\mathscr{A}$.

Our definition of reproducibility is inspired by the literature on pseudodeterministic algorithms, particularly the work of Grossman and Liu Grossman and Liu [2019] and Goldreich Goldreich [2019]. In the pseudodeterministic setting, the primary concern is reproducing the output of an algorithm given the same input, over different choices of the algorithm's internal randomness. Our notion (Definition 4.1.1) is more suitable for the setting of machine learning, where it is desirable to reproduce the exact same output of an algorithm (with high probability) over different sample sets drawn from a distribution D.

We observe the following key properties of Definition 4.1.1.

Stability. Reproducibility is a strong stability property that implies independent parties can replicate previous results with high probability, so long as the randomness used to achieve these results is made public. For researchers solving machine learning and data analysis tasks, reproducibility allows researchers to verify published results with high probability, as long as the datasets are drawn from the same distribution.

Generalization. Reproducibility implies generalization. A reproducible learning algorithm, with high probability, outputs a hypothesis h such that the difference between the risk of h and the empirical risk of h on the training set is small. Intuitively, reproducibilitiy implies that h is independent of the training set with high probability. Thus, a Hoeffding bound can be applied to bound the risk in terms of the empirical risk.

Privacy. Differential privacy (DP) is an important notion that requires small distance between the two distributions induced by an algorithm, when run on any two datasets that differ in a single element. Crucially, it asks for the guarantees in the *worst case over datasets*. Reproducible algorithms guarantee a different form of privacy: If $\mathscr{A}$ is reproducible, then what $\mathscr{A}$ learns (for example, a trained classifier) is almost always the same; thus, $\mathscr{A}$ is usually *independent* of the chosen training data. In this way, reproducible algorithms are prevented from memorizing anything that is specific to the training data, similar to differentially private algorithms. Reproducibility is weaker than differential privacy in the sense that reproducibility only applies to in-distribution samples, whereas differential privacy applies to *any* training set. On the other hand, reproducibility is stronger in the sense that its guarantee for in-distribution samples is global rather than local (for neighboring samples).

Testability. While differential privacy has become the standard for privacy-preserving computation, an important issue that is the subject of extensive research is testing and verifying differential privacy. As discussed in Gaboardi et al. [2020], DP-algorithms and their implementations are usually analyzed by hand, and proofs of differential privacy are often intricate and prone to errors. Implementing such an algorithm in practice often gives rise to DP leaks, due to coding errors or assumptions made in the proof that do not hold on finite computers (such as the ability to sample from continuous distributions). Moreover, the complexity of verifying differential privacy is hard. Verification in the black-box setting (where the auditor has oracle access to the learning algorithm) was recently shown to be infeasible, as low query complexity implies high values of the the privacy parameters ε and δ Gilbert and McMillan [2018]. In the white-box setting where $\mathscr{A}$ is given to the tester, Gaboardi et al. [2020] shows that testing for differential privacy is $coNP^{\#P}$-complete. This has led to an active research area aiming at developing automated as well as interactive testing and verification methods for differential privacy Narayan et al. [2015], Gaboardi et al. [2013], Reed and Pierce [2010], Albarghouthi and Hsu [2018], Barthe et al. [2015, 2021], Fredrikson and Jha [2014], Zhang and Kifer [2017]. In contrast, reproducibility is a form of privacy that can be *efficiently tested* in (randomized) polynomial time (in the dimension of the data universe and ρ).

4.1.1 Contributions

Reproducibility: Properties and Alternative Definitions

We discuss alternative definitions of reproducibility and show that they are all essentially equivalent. Then, we also prove some other nice properties of reproducible algorithms. (All formal statements and proofs are in Section 4.8.)

1. **Alternative Definitions and Amplification.** We start by discussing two alternative definitions of reproducibility and relate them to our definition. First, we can generalize the definition to include algorithms $\mathscr{A}$ that not only have access to internal randomness and to random samples from an underlying distribution D, but that also have access to extra non-random inputs. This more general definition captures both the original definition of pseudodeterministic algorithms as well as our definition of reproducible learning algorithms, and all of our results remain unchanged. Second, we discuss an alternative two-parameter definition, and show that the definitions are qualitatively the same. We show how to amplify the reproducibility parameter by a standard argument where the sample complexity is increased modestly.

2. **Public versus Private Randomness.** Recall that we define reproducibility as the probability that an algorithm returns the same answer when run twice using different random samples from *D* but the same internal randomness. In Grossman and Liu [2019], the authors define a related concept in which the internal randomness is divided into two pieces, public and private randomness, but the algorithm should return the same answer when just the public randomness is held fixed. We show that, without loss of generality, it suffices to use only public randomness.

3. **Reproducibility Implies Generalization.** Learning algorithms attempt to use finite samples to generate hypotheses on unknown, possibly complex distributions. The error of a hypothesis *h* on the underlying distribution is called the generalization error. A reproducible algorithm outputs the same hypothesis with high probability, and thus the algorithm seldom draws distinctions between specific samples and the entire distribution.

4. **Connections to Data Reuse.** We explore the connection between reproducible algorithms and the adaptive data analysis model discussed in Dwork et al. [2015b] and Dwork et al. [2015a]. We show that reproducible algorithms are strongly resilient against adaptive queries. Informally, with respect to reproducible algorithms, the sample complexity and accuracy of (reproducibly) answering *m* adaptively chosen queries behaves similarly to the sample complexity and accuracy of reproducibly answering *m nonadaptively* chosen queries.

Upper Bounds

Our main technical results are reproducible algorithms for some well-studied statistical query and learning problems that are used as building blocks in many other algorithms.

1. **Simulating SQ Algorithms.** In Section 4.3, we give a generic algorithm that reduces the problem of ρ-reproducibly estimating a single statistical query with tolerance τ and error δ to that of *nonreproducibly* estimating the same query within a smaller tolerance and error.

   Theorem 4.1.2 (Theorem 4.3.3, Restated)**.** *Let $\psi : \mathscr{X} \to \{0,1\}$ be a statistical query. Then the sample complexity of ρ-reproducibly estimating ψ within tolerance τ and error δ is at* *most the sample complexity of (nonreproducibly) estimating ψ within tolerance $\tau' = \tau\rho$ and error $\delta' = \tau\delta$.*

The basic idea is to obtain an estimate of the statistical query with a smaller tolerance τ' and then use a randomized rounding scheme where the interval $[0,1]$ is divided into intervals of size roughly τ/ρ. Then, every value in the interval is rounded to the midpoint of the region it occurs in. The partition into intervals is chosen with a random offset so that with high probability nearby points will lie in the same region.

2. **Heavy-hitters.** Using our simulation of SQ queries, in Section 4.4, we demonstrate the usefulness of reproducibility by giving a reproducible algorithm `rHeavyHitters` for identifying approximate v-heavy-hitters of a distribution, i.e. the elements in the support of the distribution with probability mass at least v.

   Lemma 4.1.3 (Lemma 4.4.3, Restated)**.** *For all $\varepsilon \in (0, 1/2)$, $v \in (\varepsilon, 1-\varepsilon)$, with probability at least $1-\rho$, $\texttt{rHeavyHitters}_{\rho,v,\varepsilon}$ is ρ-reproducible, and returns a list of v'-heavy-hitters for some $v' \in [v-\varepsilon, v+\varepsilon]$. Furthermore, the sample complexity is bounded by $\widetilde{O}(\rho^{-2})$.*

   The high level idea of our algorithm is to first draw sufficiently many samples, $\mathbf{s}_1$, $Q_1 = |\mathbf{s}_1|$, so that with high probability all heavy-hitters are in $\mathbf{s}_1$. In the second stage, we draw a fresh set $\mathbf{s}_2$ of Q_2 many samples and use them to empirically estimate the density of each element in $\mathbf{s}_1$, and remove those that aren't above the cutoff v', where v' is chosen randomly from $[v-\varepsilon, v+\varepsilon]$ to avoid boundary issues.

3. **Median Finding.** In Section 4.5, we design a reproducible algorithm for finding an approximate median in an arbitrary distribution over a finite domain. Approximate median finding is a fundamental statistical problem, and is also extensively studied in the privacy literature. Like in the setting of differential privacy, reproducible median finding is a key building block for making other algorithms reproducible. For example, using a known reduction from Bun et al.

[2015], a ρ-reproducible approximate median algorithm implies an algorithm of comparable sample complexity for reproducibly PAC-learning (one-dimensional) threshold functions.

Theorem 4.1.4 (Theorem 4.5.8, Restated)**.** *Let $\tau, \rho \in [0,1]$ and let $\delta = 1/3$. Let D be a distribution over $\mathscr{X}$, where $|\mathscr{X}| = 2^d$. Then* $\texttt{rMedian}_{\rho,d,\tau,\delta}$ *is ρ-reproducible, outputs a τ-approximate median of D with success probability $1-\delta$, and has sample complexity*

$$\tilde{\Omega}\left(\left(\frac{1}{\tau^2(\rho-\delta)^2}\right)\cdot\left(\frac{3}{\tau^2}\right)^{\log^*|\mathscr{X}|}\right)$$

To describe the key ideas in the algorithm, we first show how approximate-median finding is useful for turning many algorithms into reproducible ones. Consider any problem where the correct answers form an interval, and assume we start with a (not-necessarily) reproducible algorithm that is mildly accurate. Then we can run a reproducible approximate-median finding algorithm on the distribution of outputs of the original algorithm to construct a very accurate reproducible algorithm.

We will actually use this strategy recursively to reproducibly solve approximate median itself. Our algorithm recursively composes a mildly accurate reproducible median algorithm with a generic very accurate non-reproducible median algorithm. This recursive technique is inspired by, but simpler than, previous algorithms in the privacy literature Bun et al. [2015], Kaplan et al. [2020], and like these algorithms, the sample complexity of our algorithm has a non-constant but very slowly growing dependence on the domain size.

4. **Learning Halfspaces.** In Section 4.6, we obtain a reproducible algorithm `rHalfspaceWkL` for weakly learning halfspaces. In Section 4.7, we transform it into a reproducible strong learner by way of a reproducible boosting algorithm `rBoost`. We stress that our algorithms for halfspaces are reproducible in the stronger distribution-free setting.

Theorem 4.1.5 (Corollary 4.7.5, Restated)**.** *Let D be a distribution over $\mathbb{R}^d$, and let $f : \mathbb{R}^d \to \{\pm 1\}$ be a halfspace with margin τ in D. For all $\rho, \varepsilon > 0$. Algorithm* `rBoost` *run with weak* *learner* `rHalfspaceWkL` ρ*-reproducibly returns a hypothesis* $\mathbf{h}$ *such that, with probability at least* $1-\rho$*,* $\mathbf{Pr}_{\mathbf{x}\sim D}[\mathbf{h}(\mathbf{x}) = f(\mathbf{x})] \geq 1-\varepsilon$*. Furthermore, the overall sample complexity is* $\widetilde{O}\left(\frac{d^{10/9}}{\tau^{76/9}\rho^{20/9}\varepsilon^{28/9}}\right)$.

Our approximate median algorithm for answering statistical queries can be used to reproducibly learn 1-dimensional halfspaces using a known reduction from Bun et al. [2015]. At a high level, this algorithm started with an SQ algorithm with tight concentration/tolerance, and we then applied a randomized rounding scheme in order to argue that the reproducibility of the resulting algorithm. In order to reproducibly learn higher dimensional thresholds (halfspaces), we will follow a similar approach. We start with a simple weak learning algorithm for halfspaces Servedio [2002] that takes examples $(\mathbf{x}_i, y_i) \in \mathscr{X} \times \{\pm 1\}$, normalizes them, and returns the halfspace defined by vector $\sum_i \mathbf{x}_i \cdot y_i$. We show a concentration bound on the sum of normalized vectors from a distribution, and then argue that all vectors within the concentration bound are reasonable hypothesis with non-negligible advantage.

Our randomized rounding scheme is a novel application of the randomized rounding technique developed in the study of foams Kindler et al. [2012]. The concentration bound together with the foams rounding scheme Kindler et al. [2012] yields a reproducible halfspace weak learner. We then obtain our reproducible strong learner for halfspaces by combining it with a (new) reproducible boosting algorithm. Our algorithm is sample efficient but inefficient with respect to runtime, due to the inefficiency of the foams rounding scheme. We also give another randomized rounding procedure that gives a polynomial-time strong reproducible halfspace learner, but with polynomially larger sample complexity.

The Price of Reproducibility.

In Section 4.9 we ask what is the cost of turning a nonreproducible algorithm into a reproducible one. We first show that a τ-tolerant ρ-reproducible SQ algorithm $\mathscr{A}$ for ϕ implies a ρ-reproducible algorithm for the τ-coin problem: given samples from a p-biased coin with the promise that either $p \geq 1/2+\tau$ or $p \leq 1/2-\tau$, determine which is the case. Our main result in this section are nearly tight upper and lower bound bounds of $\Theta(\tau^{-2}\rho^{-2})$ on the sample complexity of ρ-reproducibly solving the τ-coin problem (for constant δ), and thus the same bounds for ρ-reproducibly answering SQ queries. On the other hand, it is well-known that the *nonreproducible* sample complexity of the τ-coin problem is $\Theta(\tau^{-2}\log(1/\delta))$ (see, e.g. Mousavi). So the cost of guaranteeing ρ-reproducibility for SQ queries is a factor of ρ^{-2}.

For upper bounds, our generic algorithm in Section 4.3 converts any SQ query into a reproducible one: if our end goal is a ρ-reproducible algorithm for estimating a statistical query with tolerance τ and error δ, then the sample complexity is at most the sample complexity of *nonreproducibly* answering the query to within tolerance τ', and success probability $1-\delta'$ where $\tau' = O(\tau\rho)$ and $\delta' = O(\delta\tau)$, which has sample complexity $O(\tau^{-2}\rho^{-2}\log(1/\delta'))$. The main result in this section is the following lower bound for ρ-reproducibly answering statistical queries.

Theorem 4.1.6 (Theorem 4.9.1, Restated)**.** *Let $\tau > 0$ and let $\delta < 1/16$. Any ρ-reproducible algorithm for solving the τ-coin coin problem with success probability at least $1-\delta$ requires sample complexity $\Omega(\tau^{-2}\rho^{-2})$.*

Related Work.

A subset of these results Impagliazzo et al. [2021] was presented at the TPDP 2021 workshop.

Our Definition 4.1.1 is inspired by the literature on pseudodeterministic algorithms Gat and Goldwasser [2011], Goldreich et al. [2013], Goldwasser and Grossman [2017], Goldwasser et al. [2018, 2019], Grossman and Liu [2019], Goldreich [2019]. In particular, Grossman and Liu [2019] and Goldreich [2019] define reproducibility in the context of pseudodeterminism. There, the input of a reproducible algorithm is a fixed string. In our setting, the input of a reproducible learning algorithm is a distribution, only accessible by randomly drawing samples.

Independently of our work, Ghazi et al. [2021] define a property equivalent to reproducibility, called "pseudo-global stability". Their (α,β)-accurate (η',ν')-pseudo-global stability definition is equivalent to the (η,ν)-reproducibility definition discussed in Section 4.8, except that pseudo-global stability includes explicit parameters for correctness and sample complexity. In Section 4.8, we show that these two definitions are equivalent to Definition 4.1.1 up to polynomial factors. Ghazi et al. [2021] gives pseudo-globally stable SQ algorithms, an amplification of the stability parameter, and an algorithm to find a heavy-hitter of a distribution. The authors use pseudo-global stability to show that classes with finite Littlestone dimension can be learned user-levelly privately, and they connect pseudo-global stability to approximate differential privacy. Pseudo-global stability is a generalization of global stability, introduced in Bun et al. [2020b]. Those authors use global stability as an intermediate step to show that classes with finite Littlestone dimension can be learned privately, and they show how global stability implies generalization.

Our work is related to other notions of stability in machine learning which, like our definition, are properties of learning algorithms. In the supervised learning setting, stability is a measure of how much the output of a learning algorithm changes when small changes are made to the input training set. An important body of work establishes strong connections between the stability of a learning algorithm and generalization Devroye and Wagner [1979a,b], Kearns and Ron [1999], Bousquet and Elisseeff [2002], Shalev-Schwartz et al. [2010]. Distributional notions of stability which remain stable under composition and postprocessing, were defined and shown to be closely connected to differential privacy and adaptive data analysis (e.g., Bassily et al. [2016], Dwork et al. [2015a]). In fact, the definition of differential privacy itself is a form of stability known as max-KL stability. Stability-based principles have also been explored in the context of unsupervised learning where model selection is a difficult problem since there is no ground truth. For example, a stable algorithm for clustering has the property that when the algorithm is applied to different data sets from the same distribution, it will yield similar outputs (e.g., von Luxburg [2010]).

In all of these settings, stability depends on how close the outputs are when the inputs are close; what varies is the particular measure of closeness in input and output space. For example, closeness in the output can be with respect to function or parameter space; for distributional stability close means that the output distributions are close with respect to some metric over distributions. Our definition of reproducibility can be viewed as an extreme form of stability where the output is required to be *identical* almost all of the time, and not just similar. Thus reproducibility enjoys many of the nice properties of stable algorithms (e.g., postprocessing, composition) but has the advantage of being far easier to verify.

Open Questions and Future Work

One motivation for examining reproducibility in algorithms is the "reproducibility crisis" in experimental science. Can we use reproducibility to create statistical methodologies that would improve reproducibility in published scientific work? A concrete step towards this would be to design reproducible hypothesis testing algorithms. We can view a null hypothesis as postulating that data will come from a specific distribution D, and want algorithms that accept with high probability if the data comes from D (or a "close" distribution) and reject with good probability if the data distribution is "far" from D. For example, the coin problem is a degenerate case in which the data are Boolean and the distance is the difference in the expected values. For different types of data and distance metrics, what is the optimal sample complexity of hypothesis testing, and how much more is that for reproducible hypothesis testing?

A related problem is that of learning under distributional shifts, or individual-based fair learning (where we want the learning algorithm to treat similar people similarly with respect to a similarity metric defining closeness). A key step in making algorithms reproducible is a randomized procedure to round the output of a standard empirical learner to a *single* hypothesis in a way that is independent of the underlying distribution. Can similar ideas be used to design learning algorithms robust to distributional shifts, or to give more informed performance metrics?

This work establishes that there exist reproducible algorithms for a variety of learning problems. However, we do not characterize exactly which learning algorithms can be made reproducible, or how reproducibility affects the required sample complexity. Is it possible to identify an invariant of concept classes which characterizes the complexity of reproducible learning, analogous to VC-dimension for PAC learning Vapnik and Chervonenkis [1971], representation dimension and one-way communication complexity for exact differential privacy Feldman and Xiao [2014], Beimel et al. [2013], and Littlestone Dimension for approximate differential privacy Bun et al. [2020b]? A specific problem of interest is that of learning linear functions over finite fields. If the data has full dimension, the function can be solved for uniquely; so, designing reproducible algorithms when the data does not form a basis seems interesting.

Also, we described the first reproducible boosting algorithm. Are there natural conditions under which a boosting algorithm can always be made reproducible? Are the sample complexity upper bounds we obtain for our applications tight or close to tight? In particular, is there a reproducible algorithm for approximate median that has only $\log^* |\mathscr{X}|$ dependence on the domain size?

Reproducibility provides a distinctive type of privacy. Except with the small probability ρ, a reproducible algorithm's outputs are a function entirely of the underlying distribution and the randomness of the algorithm, not the samples. Thus, a reproducible algorithm seldom leaks information about the specific input data. We borrowed techniques from the study of private data analysis and differential privacy, and we hope that future work will formalize connections between reproducibility and private data analysis. We also hope that some applications of differential privacy will also be achievable through reproducibility.

4.2 Concentration of Sum of Vectors

In this Section, we use Azuma's inequality to prove a concentration bound on the sum of vectors from a distribution. We will need this bound for the proof of Theorem 4.6.13.

Let D be a distribution on $\mathbb{R}^n$. Let $\mathbf{v} = \{\mathbf{v_1}, \ldots, \mathbf{v_T}\} \in D^T$ be a random sample of T vectors from D with the following properties:

1. $\mathbb{E}_{\mathbf{v}\in D^T}[\sum_{i=1}^T \mathbf{v_i}] - \mathbb{E}_{v\in D}[v] = 0$.

2. $\forall v \in D$, $||v||_2 \leq c$.

The following lemma shows that the length of $\mathbf{v^1} + \mathbf{v^2} + \ldots + \mathbf{v^T}$ is tightly concentrated.

Lemma 4.2.1. *Let $D, \mathbf{v} \in D^T$ satisfy properties (1) and (2) above, and let $\mathbf{v}^{\leq \mathbf{T}} = \sum_{i=1}^T \mathbf{v_i}$. Then for all $\Delta > 0$,*

$$\mathbf{Pr}_{\mathbf{v}}[||\mathbf{v}^{\leq \mathbf{T}}||_2 \geq \sqrt{T}(1+c/2) + \Delta] \leq e^{-\Delta^2/2c^2T}.$$

The intuition behind Lemma 4.2.1 is similar to the one-dimensional case, where D is a distribution over $(-1,1)$, $\mathbf{v} \in D^T$, and $\sum_{i=1}^T \mathbf{v_i}$ is concentrated around zero, with standard deviation

$\sqrt{T}$. Let $\mathbf{v^{\leq i}}$ denote $\sum_{i=1}^{i} \mathbf{v_i}$. In the one-dimensional case, we can prove concentration of $\mathbf{v^{\leq T}}$ via a Chernoff or martingale argument since the expected value of $\mathbf{v^{\leq i}}$ (the sum of the first i numbers) is equal to $\mathbf{v^{\leq i-1}}$. However for the higher dimensional case, $\mathbf{v^{\leq i}}$ is now the sum of the first i vectors, and it is in general not the case that the expected length of $\mathbf{v^{\leq i}}$ is equal or even not much larger than the length of $\mathbf{v^{\leq i-1}}$. However, if the length of $\mathbf{v^{\leq i-1}}$ is sufficiently large (greater than $\sqrt{T}$), then $\mathbb{E}[||\mathbf{v^{\leq i}}||_2 \mid \mathbf{v^{\leq i-1}}]$ can be upper bounded (approximately) by $||\mathbf{v^{\leq i-1}}||_2 + 1/\sqrt{T}$. Therefore, if we want to bound the probability that the length of $\mathbf{v^{\leq T}}$ is large (at least $\sqrt{T}+\Delta$), there must be some time t such that the vector $\mathbf{v^{\leq t}}$ is outside of the ball of radius $\sqrt{T}$ around the origin, and never returns. So we can bound the probability that $||\mathbf{v^{\leq T}}||_2 \geq \sqrt{t}+\Delta$, by considering the sequence of random variables $\mathbf{x^{\leq t}}, \ldots, \mathbf{x^{\leq T}}$ such that $\mathbf{x^{\leq t}}$ is equal to the length of $\mathbf{v^{\leq t}}$, and for each $t' \geq t$, $\mathbf{x^{\leq t'}}$ is the length of $\mathbf{v^{\leq t'}}$ minus a correction term (so that we can upper bound $\mathbb{E}[\mathbf{x^{\leq t'+1}} \mid \mathbf{x^{\leq t'}}]$ by $\mathbf{x^{\leq t'}}$.) We will show that $\mathbf{x^{\leq t}}, \ldots, \mathbf{x^{\leq T}}$ is a supermartingale where $|\mathbf{x^{\leq t'+1}} - \mathbf{x^{\leq t'}}|$ is bounded by a constant, and then the concentration inequality will follow from Azuma's Lemma.

Definition 4.2.2. Let D be a distribution over $\mathbb{R}^n$ satisfying the above two properties.

1. Let $\mathbf{v} = \{\mathbf{v_1}, \ldots, \mathbf{v_{T'}}\} \in D^{T'}$ be a sequence of $T' \leq T$ random variables, and let $\mathbf{v_0} \in \mathbb{R}^n$ have length $\sqrt{T}$. For $0 \leq i \leq T'$, let $\mathbf{v^{\leq i}} = \sum_{i=1}^{T'} \mathbf{v_i}$.

2. The *stopping time* $\tau \in [T']$ (with respect to $\{\mathbf{v^{\leq i}}\}$) is equal to:

$$\min\{\{i \in [T'] \mid ||\mathbf{v^{\leq i}}||_2 < \sqrt{T}\} \cup \{T'\}\}.$$

   That is, τ is the first time i such that the length of $\mathbf{v^{\leq i}}$ drops below $\sqrt{T} + \frac{i}{3\sqrt{T}}$ (and otherwise $\tau = T'$).

3. For each $i \in [T']$, we define the sequence of random variables $\mathbf{x^{\leq 0}}, \mathbf{x^{\leq 1}}, \mathbf{x^{\leq 2}}, \ldots, \mathbf{x^{\leq T'}}$ where $\mathbf{x^{\leq 0}} = ||\mathbf{v^0}||_2 = \sqrt{T}$, and for all $i \geq 1$, $\mathbf{x^{\leq i}}$ will be the adjusted length of the first i vectors,

$||\mathbf{v}^{\leq \mathbf{i}}||$ with stopping condition τ:

$$\mathbf{x}^{\leq \mathbf{i}} = \begin{cases} ||\mathbf{v}^{\leq \mathbf{i}}||_2 - \frac{ci}{2\sqrt{T}} & \text{if } \tau > i \\ \mathbf{x}^{\leq \tau} & \text{otherwise} \end{cases}$$

Claim 4.2.3. *The sequence of random variables $\mathbf{x}^{\leq \mathbf{1}}, \ldots, \mathbf{x}^{\leq \mathbf{T'}}$ is a supermartingale.*

Proof. We need to show that for every $i \in [T']$, $\mathbb{E}[\mathbf{x}^{\leq \mathbf{i}} \mid \mathbf{x}^{\leq \mathbf{i-1}}] \leq \mathbf{x}^{\leq \mathbf{i-1}}$. Fix $i \in [T']$; if $\tau \leq i-1$ then $\mathbf{x}^{\leq \mathbf{i}} = \mathbf{x}^{\leq \mathbf{i-1}}$ so the condition holds. Otherwise assume that $\tau \geq i$. Since

$$\mathbb{E}[\mathbf{x}^{\leq \mathbf{i}} \mid \mathbf{x}^{\leq \mathbf{i-1}}] = \mathbb{E}[\mathbf{x}^{\leq \mathbf{i}} \mid \mathbf{v}^{\leq \mathbf{i-1}}] = \mathbb{E}[||\mathbf{v}^{\leq \mathbf{i-1}} + \mathbf{v_i}||_2] - \frac{ci}{2\sqrt{T}}$$

and $\mathbf{x}^{\leq \mathbf{i-1}} = ||\mathbf{v}^{\leq \mathbf{i-1}}||_2 - \frac{c(i-1)}{2\sqrt{T}}$, it suffices to show that $\mathbb{E}[||\mathbf{v}^{\leq \mathbf{i-1}} + \mathbf{v_i}||_2 \leq ||\mathbf{v}^{\leq \mathbf{i-1}}||_2 + \frac{c}{2\sqrt{T}}$.

To prove this, we can write $\mathbf{v_i} = \mathbf{v_i^{||}} + \mathbf{v_i^{\perp}}$ where $\mathbf{v_i^{||}}$ is the component of $\mathbf{v_i}$ in the direction of $\mathbf{v}^{\leq \mathbf{i-1}}$, and $\mathbf{v_i^{\perp}}$ is the orthogonal component. Since the expected length of $\mathbf{v}^{\leq \mathbf{i-1}} + \mathbf{v_i^{||}}$ is equal to the length of $\mathbf{v}^{\leq \mathbf{i-1}}$ (by property 1), we just have to show that the expected length of $\mathbf{v}^{\leq \mathbf{i-1}} + \mathbf{v_i^{\perp}}$ is at most $\frac{c}{2\sqrt{T}}$. Since $\mathbf{v_i}$ has length at most c, so does $\mathbf{v_i^{\perp}}$, so we have:

$$\mathbb{E}[||\mathbf{v}^{\leq \mathbf{i-1}} + \mathbf{v_i^{\perp}}||_2] \leq (||\mathbf{v}^{\leq \mathbf{i-1}}||_2^2 + c)^{1/2} \leq \frac{c}{2\sqrt{T}}$$

where the last inequality holds since $\tau \geq i$ implies $||\mathbf{v}^{\leq \mathbf{i-1}}||_2 \geq \sqrt{T}$. □

Claim 4.2.4. *For all i, $|\mathbf{x}^{\leq \mathbf{i}} - \mathbf{x}^{\leq \mathbf{i-1}}| \leq c$.*

Proof. Since $\mathbf{v_i}$ has length at most c the absolute value of the difference between $||\mathbf{v}^{\leq \mathbf{i}}||_2$ and $||\mathbf{v}^{\leq \mathbf{i-1}}||_2$ is at most 2. The claim easily follows since $\mathbf{x}^{\leq \mathbf{i}} = ||\mathbf{v}^{\leq \mathbf{i}}|| + \frac{ci}{2\sqrt{T}}$. □

The above two Claims together with Azuma's inequality gives:

$$Pr[|\mathbf{x}^{\leq \mathbf{T'}} - \mathbf{x}^{\leq \mathbf{0}}| \geq \Delta] \leq e^{-\Delta^2/2c^2T}.$$

Proof. (of Lemma 4.2.1)

In order for $\mathbf{v}^{\leq \mathbf{T}}$ to have length at least $\sqrt{T}(1+c/2)+\Delta$, there must be some largest time $t \in [T]$ such that $||\mathbf{v}^{\leq \mathbf{t}}||_2 \in (\sqrt{T}, \sqrt{T}+1]$. That is, at all times $t' \geq t$ the vector $\mathbf{v}^{\leq \mathbf{t'}}$ is outside the ball of radius $\sqrt{T}$. Thus by the above argument, the random variables $\mathbf{x}^{\leq \mathbf{i}}{}_{i=t}^{T}$ are a supermartingale where the absolute value of the difference between successive variables is at most c, and by Azuma, $\mathbf{Pr}[\mathbf{x}^{\leq \mathbf{T}} \geq \sqrt{T}+\Delta]$ is at most $e^{-\Delta^2/2c^2T}$. Since $\mathbf{x}^{\leq \mathbf{T}} = ||\mathbf{v}^{\leq \mathbf{T}}||_2 - \frac{Tc}{2\sqrt{T}} = ||\mathbf{v}^{\leq \mathbf{T}}||_2 - \frac{\sqrt{T}c}{2}$, $\mathbf{Pr}[||\mathbf{v}^{\leq \mathbf{T}}||_2 \geq \sqrt{T}(1+c/2)+\Delta]$ is at most $e^{-\Delta^2/2c^2T}$. □

This gives the following corollary.

Corollary 4.2.5. *Let D be a distribution supported on the unit ball in d dimensions, and let f be a halfspace. Let S be a sample of T examples* $(\mathbf{x}_i, y_i)$ *drawn i.i.d. from D, and let* $\mathbf{z} = \sum_S \mathbf{x}_i \cdot y_i$*. Let* $a \in (0, 1/2)$*. Then* $\mathbf{Pr}_{S\sim D}\left[||\mathbf{z} - \mathbb{E}_{\mathbf{v}\sim D}\mathbf{v}|| \geq 4T^{1/2+a}\right] \leq e^{-T^{2a}/2}$.

Proof. In order to have D satisfy the properties (1) and (2) above, we must translate D by the expectation $\mathbb{E}_{\mathbf{v}\sim D}[\mathbf{v}]$. After this translation, the maximum length of a vector in the support is $c = 2$. Plugging in $\Delta = 2T^{1/2+a}$ and noting $2T^{1/2+a} \geq 2T^{1/2}$ yields the conclusion. □

4.3 Statistical Queries

We show how to use randomized rounding to reproducibly simulate any SQ oracle and therefore any SQ algorithm. The statistical query model introduced by Kearns [1998] is a restriction of the PAC-learning model introduced by Valiant [1984a]. We consider the statistical query oracle primarily in the context of unsupervised learning (e.g., see Feldman [2016]).

Definition 4.3.1 (Statistical query oracle)**.** Let $\tau \in [0,1]$ and $\phi : \mathscr{X} \to [0,1]$ be a query. Let D be a distribution over domain $\mathscr{X}$. A *statistical query oracle* for D, denoted $\mathscr{O}_D(\tau, \phi)$, takes as input a tolerance parameter τ and a query ϕ, and outputs a value v such that $|v - \mathbb{E}_{x\sim D}[\phi(x)]| \leq \tau$.

Definition 4.3.2 (Simulating a statistical query oracle)**.** Let $\delta \in [0,1]$ and τ, ϕ, D be as above. Let $\mathscr{O}_D$ be a statistical query oracle for D. Let $\mathbf{s}$ denote an i.i.d. sample drawn from D. We say that a routine STAT *simulates* $\mathscr{O}_D$ with failure probability δ if for all τ, δ, ϕ, there exists an $n_0 \in \mathbb{N}_+$ such that if $n > n_0$, $v \leftarrow \texttt{STAT}(\tau, \phi, \mathbf{s})$ satisfies $|v - \mathbb{E}_{x \sim D}[\phi(x)]| \leq \tau$ except with probability δ.

To denote a routine simulating a statistical query oracle for fixed parameters τ, ϕ, and (optionally) ρ, we write these parameters as subscripts.

Algorithm 14. $\texttt{rSTAT}_{\rho,\tau,\phi}(\mathbf{s})$
Parameters: τ - tolerance parameter
ρ - reproducibility parameter
ϕ: a query $X \to [0,1]$

```
1: $\alpha = \frac{2\tau}{\rho+1-2\delta}$
2: $\alpha_{\text{off}} \leftarrow_r [0, \alpha]$
3: Split $[0,1]$ in regions: $R = \{[0, \alpha_{\text{off}}), [\alpha_{\text{off}}, \alpha_{\text{off}} + \alpha), \ldots, [\alpha_{\text{off}} + i\alpha, \alpha_{\text{off}} + (i+1)\alpha), \ldots, [\alpha_{\text{off}} + k\alpha, 1)\}$
4: $v \leftarrow \frac{1}{|\mathbf{s}|} \sum_{x \in \mathbf{s}} \phi(x)$
5: Let $r_v$ denote the region in $R$ that contains $v$
6: return the midpoint of region $r_v$
```

Theorem 4.3.3 upper bounds the sample complexity of $\texttt{rSTAT}_{\tau,\rho,\phi}$. In Section 4.9, we show this upper bound is tight as a function of ρ.

Theorem 4.3.3 (rSTAT simulates a statistical query oracle)**.** *Let $\tau, \delta, \rho \in [0,1]$, $\rho > 2\delta$, and let* $\mathbf{s}$ *be a sample drawn i.i.d. from distribution D. Then if*

$$|\mathbf{s}| \in \tilde{O}\left(\frac{1}{\tau^2(\rho - 2\delta)^2}\right)$$

$\texttt{rSTAT}_{\rho,\tau,\phi}(\mathbf{s})$ *ρ-reproducibly simulates an SQ oracle $\mathscr{O}_{D,\tau,\phi}$ with failure rate δ.*

In Section 4.9, we will prove a near matching lower bound on the sample complexity of ρ-reproducibly estimating a statistical query with tolerance τ and success probability $1 - \delta$.

Proof. We begin by showing that $\texttt{rSTAT}_{\rho,\tau,\phi}$ simulates an SQ oracle $\mathscr{O}_{D,\tau,\phi}$ with failure rate δ.

Let $\tau' = \frac{\tau(\rho-2\delta)}{\rho+1-2\delta}$. Recall $\alpha \overset{\text{def}}{=} \frac{2\tau}{\rho+1-2\delta}$, so $\frac{2\tau'}{\alpha} = \rho - 2\delta$. A Chernoff bound gives that

$$\left| \frac{1}{|\mathbf{s}|} \sum_{x \in \mathbf{s}} \phi(x) - \mathop{\mathbb{E}}_{x \sim D} \phi(x) \right| \le \tau' = \frac{\tau(\rho - 2\delta)}{\rho + 1 - 2\delta}$$

except with failure probability δ, so long as $|\mathbf{s}| \ge \log(2/\delta)/(2\tau'^2)$. Outputting the midpoint of region r_v can further offset this result by at most $\alpha/2 = \frac{\tau}{\rho+1-2\delta}$. Therefore

$$|v - \mathbb{E}_{x\sim D}\phi(x)| \le \frac{\tau(\rho-2\delta)}{\rho+1-2\delta} + \frac{\tau}{\rho+1-2\delta} = \tau,$$

except with probability δ, so long as the sample $\mathbf{s}$ satisfies

$$\log(2/\delta)/(2\tau'^2) = \frac{\log(2/\delta)(\rho+1-2\delta)^2}{2\tau^2(\rho-2\delta)^2} \le \frac{4\log(2/\delta)}{2\tau^2(\rho-2\delta)^2} \le |\mathbf{s}|.$$

We now show that $\texttt{rSTAT}_{\rho,\tau,\phi}$ is ρ-reproducible by considering two invocations of $\texttt{rSTAT}_{\rho,\tau,\phi}$ with common randomness r on samples $\mathbf{s}_1, \mathbf{s}_2 \sim D$ respectively. The probability that either empirical estimate of $\mathbb{E}_{x\sim D}[\phi(x)]$ fails to satisfy tolerance τ is at most 2δ. Denote by v_1 and v_2 the values returned by the parallel runs $\texttt{rSTAT}(\mathbf{s}_1; r)$ and $\texttt{rSTAT}(\mathbf{s}_2; r)$ at line 4. Conditioning on success, values v_1 and v_2 differ by at most $2\tau'$. $\texttt{rSTAT}$ outputs different values for the two runs if and only if v_1 and v_2 are in different regions of R, determined by the common randomness r. This occurs if some region's endpoint is between v_1 and v_2; since α_{off} is chosen uniformly in $[0,\alpha]$, the probability that v_1 and v_2 land in different regions is at most $2\tau'/\alpha = \rho - 2\delta$. Accounting for the 2δ probability of failure to estimate $\mathbb{E}_{x\sim D}[\phi(x)]$ to within tolerance, $\texttt{rSTAT}_{\rho,\tau,\phi}(\mathbf{s})$ is ρ-reproducible. □

4.4 Heavy-hitters

Next, we present our reproducible approximate heavy-hitters algorithm, analyzing its sample complexity and reproducibility. We will use this algorithm as a subroutine in later algorithms such as in the approximate-median algorithm. Also, we will show how to use this algorithm to give a generic way to boost reproducibility from constant ρ to arbitrarily small ρ.

Definition 4.4.1 (Heavy-Hitter)**.** Let D be a distribution over $\mathscr{X}$. Then we say $x \in \mathscr{X}$ is a *v-heavy-hitter* of D if $\mathbf{Pr}_{x' \sim D}[x' = x] \geq v$.

Definition 4.4.2 ((Approximate) Heavy-Hitter Problem)**.** Let L_v be the set of $x \in \operatorname{supp}(D)$ that are v-heavy-hitters of D. Given sample access to D, output a set L satisfying $L_{v+\varepsilon} \subseteq L \subseteq L_{v-\varepsilon}$.

Let D be a distribution over $\mathscr{X}$. The following algorithm reproducibly returns a set of v'-heavy-hitters of D, where v' is a random value in $[v-\varepsilon, v+\varepsilon]$. Picking v' randomly allows the algorithm to, with high probability, avoid a situation where the cutoff for being a heavy-hitter (i.e. v') is close to the probability mass of any $x \in \operatorname{supp}(D)$.

Algorithm 15. `rHeavyHitters`$_{\rho,v,\varepsilon}$
Input: samples $\mathscr{X}_{\text{set}}$, S from distribution D over $\mathscr{X}$ plus internal randomness r
Parameters: Target reproducibility ρ, target range $[v-\varepsilon, v+\varepsilon]$
Output: List of v'-heavy-hitters of D, where $v' \in [v-\varepsilon, v+\varepsilon]$

```
$\mathscr{X}_{\text{set}} \leftarrow Q_1 \overset{\text{def}}{=} \frac{\ln(6/(\rho(v-\varepsilon)))}{v-\varepsilon}$ examples from $D$   /* Step 1: Find candidate heavy-hitters */
$S \leftarrow Q_2 \overset{\text{def}}{=} \frac{2^6 \ln(Q_1/\rho) \cdot Q_1^2}{(\rho\varepsilon)^2}$ fresh examples from $D$   /* Step 2: Estimate probabilities */
for all $x \in \mathscr{X}_{\text{set}}$ do
    $\widehat{p_x} \leftarrow \mathbf{Pr}_{x' \sim S}[x' = x]$   /* Estimate $p_x \overset{\text{def}}{=} \mathbf{Pr}_{x' \sim D}[x' = x]$ */
$v' \leftarrow_r [v-\varepsilon, v+\varepsilon]$ uniformly at random   /* Step 3: Remove non-$v'$-heavy-hitters */
Remove from $\mathscr{X}_{\text{set}}$ all $x$ for which $\widehat{p_x} < v'$.
return $\mathscr{X}_{\text{set}}$
```

Algorithm `rHeavyHitters` returns exactly the list of v'-heavy-hitters so long as the following hold:

1. In Step 1 of Algorithm 15, all $(v-\varepsilon)$-heavy-hitters of D are included in $\mathscr{X}_{\text{set}}$.

2. In Step 2, the probabilities $\widehat{p_x}$ for all $x \in \mathscr{X}_{\text{set}}$ are correctly estimated to within error $\rho\varepsilon/(3Q_1)$.

3. In Step 3, the randomly sampled v' does not fall within an interval of width $\rho\varepsilon/(3Q_1)$ centered on the true probability of a $(v-\varepsilon)$-heavy-hitter of D.

We show that these 3 conditions will hold with probability at least $1-\rho/2$, and so will hold for two executions with probability at least $1-\rho$.

Lemma 4.4.3. *For all $\varepsilon \in (0,1/2)$, $v \in (\varepsilon, 1-\varepsilon)$, with probability at least $1-\rho$,* `rHeavyHitters` *is reproducible, returns a list of v'-heavy-hitters for some $v' \in [v-\varepsilon, v+\varepsilon]$, and has sample complexity $\widetilde{O}\left(\frac{1}{\rho^2\varepsilon^2(v-\varepsilon)^2}\right)$.*

Proof. We say Step 1 of Algorithm 15 succeeds if all $(v-\varepsilon)$-heavy-hitters of D are included in $\mathscr{X}_{\text{set}}$ after Step 1. Step 2 succeeds if the probabilities for all $x \in \mathscr{X}_{\text{set}}$ are correctly estimated to within error $\rho\varepsilon/(3Q_1)$. Step 3 succeeds if the returned $\mathscr{X}_{\text{set}}$ is exactly the set of v'-heavy-hitters of D. Quantities Q_1 and Q_2 are defined in the pseudocode of Algorithm 15.

In Step 1, an individual $(v-\varepsilon)$-heavy-hitter is not included with probabilility at most $(1-v+\varepsilon)^{Q_1}$; union bounding over all $1/(v-\varepsilon)$ possible $(v-\varepsilon)$-heavy-hitters, Step 1 succeeds with probability at least $1-\frac{(1-v+\varepsilon)^{Q_1}}{v-\varepsilon} > 1-\rho/6$. Here, for clarity of presentation in the statement of Lemma 4.4.3, we make use of the inequality $v-\varepsilon < \ln(1/(1-v+\varepsilon))$.

By a Chernoff bound, each p_x is estimated to within error $\rho\varepsilon/(3Q_1)$ with all but probability $\rho/(6Q_1)$ in Step 2. Union bounding over all Q_1 possible $x \in \mathscr{X}_{\text{set}}$, Step 2 succeeds except with probability $\rho/6$.

Conditioned on the previous steps succeeding, Step 3 succeeds if the randomly chosen v' is not within $\rho\varepsilon/(3Q_1)$ of the true probability of any $x \in \mathscr{X}_{\text{set}}$ under distribution D. A v' chosen randomly from the interval $[v-\varepsilon, v+\varepsilon]$ lands in any given subinterval of width $\rho\varepsilon/(3Q_1)$ with probability $\rho/(6Q_1)$, and so by a union bound, Step 3 succeeds with probability at least $1-\rho/6$.

Therefore, Algorithm 15 outputs exactly the set of v'-heavy-hitters of D with probability at least $1-\rho/2$. If we consider two executions of Algorithm 15, both using the same shared randomness for chooosing v', output the set of v'-heavy-hitters of D with probability at least $1-\rho$, and so `rHeavyHitters` is ρ-reproducible.

The sample complexity is $Q_1 + Q_2 \in \tilde{\Omega}\left((\rho\varepsilon(v-\varepsilon))^{-2}\right)$. $\square$

Corollary 4.4.4. *If v and ε are constants, then* $\texttt{rHeavyHitters}_{\rho,v,\varepsilon}$ *has sample complexity* $\widetilde{O}\left(1/\rho^2\right)$.

Learning Heavy-hitters using Statistical Queries.

Next, we show that any statistical query algorithm for the v-heavy-hitters problem requires $\Omega(\log|\mathscr{X}|/\log(1/\tau))$ calls to the SQ oracle. Since Algorithm 15 has a sample complexity independent of the domain size, this implies a separation between reproducible problems and problems solvable using only SQ queries.

Consider the ensemble $\{D_x\}_{x\in\mathscr{X}}$ on $\mathscr{X}$, where distribution D_x is supported entirely on a single $x \in \mathscr{X}$.

Claim 4.4.5 (Learning Heavy-hitters using Statistical Queries)**.** *Any statistical query algorithm for the v-heavy-hitters problem on ensemble $\{D_x\}_{x\in\mathscr{X}}$ requires* $\Omega(\log|\mathscr{X}|/\log(1/\tau))$ *calls to the SQ oracle.*

Proof. An SQ algorithm for the v-heavy-hitters problem must, for each distribution D_x, output set $\{x\}$ with high probability. An SQ oracle is allowed tolerance τ in its response to statistical query ϕ. So, for any ϕ, there must be some distribution D_x for which the following holds: at least a τ-fraction of the distributions $D_{x'}$ in the ensemble satisfy $|\phi(x') - \phi(x)| \leq \tau$. Thus, in the worst case, any correct SQ algorithm can rule out at most a $(1-\tau)$-fraction of the distributions in the ensemble with one query. If $\mathscr{X}$ is finite, then an SQ algorithm needs at least $\log_{1/\tau}(|\mathscr{X}|)$ queries. $\square$

4.5 Approximate Median

In this section, we design a reproducible algorithm for finding an approximate median in an arbitrary distribution over a finite domain. In addition to being a significant problem in its own right, and one studied extensively in the privacy literature, this is a key sub-routine for making many algorithms reproducible. In particular, for any problem where the correct answers form an interval, and we have a (not-necessarily) reproducible algorithm that is correct strictly more than half the time, we can run the approximate median finding algorithm on the distribution of outputs of the original to construct a reliably correct and reproducible version. (In fact, we use this technique recursively within our reproducible median-finding algorithm itself. Our algorithm `rMedianOfMedians` composes a mildly accurate reproducible median algorithm with a generic very accurate non-reproducible median algorithm.) We use a recursive technique inspired by but simpler than previous algorithms in the privacy literature Bun et al. [2015], Kaplan et al. [2020], and like for these algorithms, the sample complexity of our algorithm has a non-constant but very slowly growing dependence on the domain size. We also can use this algorithm to reproducibly PAC-learn (one-dimensional) threshold functions, using a known reduction from Bun et al. [2015].

Definition 4.5.1 (τ-approximate median)**.** Let D be a distribution over a well-ordered domain $\mathscr{X}$. $x \in \mathscr{X}$ is a τ-approximate median of D if $\mathbf{Pr}_{x' \sim D}[x' \leq x] \geq 1/2 - \tau$ and $\mathbf{Pr}_{x' \sim D}[x' \geq x] \geq 1/2 - \tau$.

4.5.1 Reproducible Approximate Median Algorithm

In this section, we present a pseudocode description of our τ-approximate median algorithm `rMedian` (Algorithm 16), and prove the following theorem.

Theorem 4.5.2 (Reproducible Median)**.** *Let $\tau, \rho \in [0,1]$ and let $\delta = \rho/2$. Let D be a distribution over $\mathscr{X}$, where $|\mathscr{X}| = 2^d$. Then $\texttt{rMedian}_{\rho,d,\tau,\delta}$ (Algorithm 16) is ρ-reproducible, outputs a τ-approximate median of D with all but probability δ, and has sample complexity*

$$n \in \tilde{O}\left(\left(\frac{1}{\tau^2(\rho-\delta)^2}\right) \cdot \left(\frac{3}{\tau^2}\right)^{\log^* |\mathscr{X}|}\right)$$

As an introduction to the key ideas of Algorithm 16, we consider a weighted binary tree T based on distribution D. Each internal node has two edges (a 0-edge and a 1-edge). Root-to-leaf paths represent binary representations of numbers. The weight of each internal node v is the probability that its associated binary prefix (induced by the root-to-v path) appears in an element drawn from D. If within this tree we can find a node v with weight in $[1/4, 3/4]$, then we can use the associated prefix to return an approximate median of D with approximation parameter potentially much larger than τ.

To achieve a specified approximation parameter τ, rather than using D itself to construct the binary tree T, we will use a distribution D^m over medians of D. Specifically, we use a non-reproducible median algorithm to sample from τ-approximate medians of D. Identifying an approximate median of distribution D^m for even a very large approximation parameter then ensures we return a τ-approximate median of D.

The question remains of how to efficiently search T to find a node v of weight in $[1/4, 3/4]$ (under D^m). We perform this search recursively by using `rMedian` to find a prefix length ℓ such that the probability of sampling two elements from D^m agreeing on a prefix of length ℓ is large. We can then restrict our search for v to nodes near level ℓ in T (starting from the root). We apply the reproducible heavy-hitters algorithm `rHeavyHitters` to find high weight nodes near level ℓ of T, and then exhaustively search the list of heavy-hitters to find an appropriate v.

We use the following non-reproducible approximate median algorithm, that returns the median of its sample $\mathbf{s}$, as a subroutine of Algorithm 16.

Lemma 4.5.3 (Simple Median Algorithm)**.** *Let sample* $\mathbf{s}$ *be drawn from distribution D. Algorithm* `Median`$(\mathbf{s})$ *returns a* τ*-approximate median on D using* $|\mathbf{s}| = 3(1/2-\tau)\ln(2/\delta)/\tau^2$ *samples with success probability at least* $1-\delta$.

Proof. Algorithm `Median`$(\mathbf{s})$ fails when more than half of the elements in sample $\mathbf{s}$ are either i) smaller than the $(1/2-\tau)$-percentile element of D or ii) larger than the $(1/2+\tau)$-percentile element of D. Let event E_i denote the first case and event E_{ii} denote the second case. Since the elements in $\mathbf{s}$ are drawn i.i.d., the first event can be bounded by a Chernoff bound. Let X be a random variable denoting the number of elements in $\mathbf{s}$ that are smaller than the $(1/2-\tau)$-percentile element of D.

$$\begin{aligned}
\mathbf{Pr}[E_i] &= \mathbf{Pr}[X \geq (1+\tau/(1/2-\tau))\,\mathbb{E}[X]] \\
&\leq \exp(-(\tau/(1/2-\tau))^2\,\mathbb{E}[X]/3) \\
&\leq \exp\left(-\frac{\tau^2}{1/2-\tau}\frac{|\mathbf{s}|}{3}\right) = \exp(-\ln(2/\delta)) \\
&= \delta/2
\end{aligned}$$

The same argument can be used to bound the second event E_{ii}. By a union bound, the algorithm succeeds with probability at least $1-\delta$. $\square$

Before proceeding with the description of Algorithm 16, we fix some useful notation for its analysis.

- n_m - sample complexity of $\texttt{Median}_{\tau,\delta_0}$
- n_h - sample complexity of $\texttt{rHeavyHitters}_{\rho_0,\nu,\varepsilon}$
- n_{sq} - sample complexity of $\texttt{rSTAT}_{\tau,\rho_0,\phi}$
- n_d - sample complexity of $\texttt{rMedian}_{\rho,d,\tau,\delta}$
- D^m - Algorithm 16 takes as input a sample from distribution D over $\mathscr{X}$, where $|\mathscr{X}| = 2^d$. We use D^m to denote the distribution induced by sampling n_m examples from D, computing $\texttt{Median}_{\tau,\delta_0}$ on these examples, and returning the ouput
- $D_{\lceil \log d \rceil}$ - We use $D_{\lceil \log d \rceil}$ to denote the distribution induced by sampling 2 examples from D^m and returning the longest prefix ℓ on which the two medians agree. Note that this new distribution is over a new domain $\mathscr{X}'$ with $|\mathscr{X}'| = 2^{\lceil \log d \rceil} \in \Theta(d)$.
- $\rho_0 \in O(\rho / \log^* |X|)$
- $\delta_0 \in O\left((\frac{\delta}{n_h+n_{sq}})^{2\log^* |\mathscr{X}|} \cdot \left(\frac{\tau^2}{3}\right)^{2(\log^* |\mathscr{X}|)^2} \right)$

Algorithm 16. `rMedian`($\mathbf{s}$)
Input: $\mathbf{s}$ - a sample of n elements drawn i.i.d. from D
Parameters: ρ - target reproducibility parameter, d - specifies domain size $|\mathscr{X}| = 2^d$, τ - target accuracy of median, δ - target failure probability
Output: a τ-approximate median of D

```
1: if $d = 1$ then
2:     Let $\phi_0(x) = 1$ if $x = 0$, 0 o/w
3:     $p_0 \leftarrow \texttt{rSTAT}_{\rho_0,\tau/2,\phi_0}(\mathbf{s})$   /* Base case */
4:     If $p_0 \geq 1/2 - \tau/2$, return 0. Else, return 1.
5: Break $\mathbf{s}$ into $|\mathbf{s}|/n_m$ subsamples
6: Run $\texttt{Median}_{\tau,\delta_0}$ on each subsample to generate a new sample $\mathbf{m}$ of $\tau$-approximate medians of $D_d$
7: Pair up elements $\mathbf{m}_{2i}$ and $\mathbf{m}_{2i-1}$, for $i \in \{1, \cdots, |\mathbf{m}|/2\}$
8: For each pair $(\mathbf{m}_{2i}, \mathbf{m}_{2i-1})$, let $l_i$ denote the longest prefix on which they agree
9: Let $\mathbf{s}_{rm}$ denote the multiset of $l_i$'s
10: $\ell \leftarrow \texttt{rMedian}_{\rho,\lceil \log d \rceil,\tau,\delta}(\mathbf{s}_{rm})$
11: $\mathbf{s}_{h0}, \mathbf{s}_{h1} \leftarrow n_h$ new examples from $\mathbf{m}$ each
12: $\mathbf{s}_\ell \leftarrow \{x_{|\ell} : x \in \mathbf{s}_{h0}\}$   /* $\mathbf{s}_\ell$ is the set $\mathbf{s}_{h0}$ projected onto length $\ell$ prefixes */
13: $V \leftarrow \texttt{rHeavyHitters}_{\rho_0,v,\varepsilon}(\mathbf{s}_\ell)$, for $v = 5/16 + \tau$, $\varepsilon = 1/16$
14: if $\ell < d$ then
15:     $\mathbf{s}_{\ell+1} \leftarrow \{x_{|\ell+1} : x \in \mathbf{s}_{h1}\}$
16:     $V \leftarrow V \cup \texttt{rHeavyHitters}_{\rho_0,v,\varepsilon}(\mathbf{s}_{\ell+1})$   /* Find vertices at level $\ell$ and $\ell + 1$ with weight $\geq 1/4$ */
17: else
18:     return the first element of $V$
19: for $v \in V$ do
20:     Let $\phi_v(x) = 1$ if $x_{||v|} = v$, 0o/w
21:     $\mathbf{s}_q \leftarrow n_{sq}$ new examples from $\mathbf{m}$
22:     $p_v \leftarrow \texttt{rSTAT}_{\rho_0,\tau,\phi_v}(\mathbf{s}_q)$,  /* Query $D^m$ for probability $x \leq v||1\cdots1$ */
23:     if $1/4 \leq p_v \leq 3/4$ then
24:         $s \leftarrow v$   /* Find length $\ell$ prefix of weight in $[1/4 - \tau, 3/4 + \tau]$ */
25: $s_0 = s||0\cdots0$   /* $s_0$ is the prefix $s$ padded with 0's to length $d$ */
26: $s_1 = s||1\cdots1$   /* $s_1$ is the prefix $s$ padded with 1's to length $d$ */
27:
28: Let $\phi_{s_0}(x) = 1$ if $x \leq s_0$, 0o/w
29: $\mathbf{s}_{s_0} \leftarrow n_m$ new examples from $\mathbf{m}$
30: $p_{s_0} \leftarrow \texttt{rSTAT}_{\rho_0,\tau,\phi_{s_0}}(\mathbf{s}_{s_0})$
31: if $p_{s_0} \geq 1/8 - 2\tau$ then
32:     return $s_0$
33: else
34:     return $s_1$
```

Lemma 4.5.4 (Termination)**.** *Algorithm 16 terminates after $T = \log^* |\mathscr{X}|$ recursive calls.*

Proof. Algorithm 16 reaches its base case when invoked with parameter $d = 1$. At each successive recursive call (Line 10), the domain size 2^d is reduced to $2^{\lceil \log d \rceil} < 2d$, and so $d = 1$ after no more than $T = \log^* |\mathscr{X}|$ recursive calls. □

Lemma 4.5.5 (Sample Complexity)**.** *Let $\tau, \delta, \rho \in [0,1]$. Let D be a distribution over $\mathscr{X}$, with $|\mathscr{X}| = 2^d$. Then* $\texttt{rMedian}_{\rho,d,\tau,\delta}$ *has sample complexity*

$$n \in O\left(\left(\frac{1}{\tau^2(\rho-\delta)^2} \right) \cdot \left(\frac{3\log(2/\delta_0)}{\tau^2} \right)^{\log^* |\mathscr{X}|} \right)$$

Proof. We begin by arguing that, for $d > 1$, $\texttt{rMedian}_{\rho,d,\tau,\delta}$ has sample complexity $n_m(2n_{\lceil \log d \rceil} + n_h + 4n_{sq})$. First, observe that Line 6 of Algorithm 16 is the only line that uses the sample $\mathbf{s}$ directly, and it uses $\mathbf{s}$ to generate a sample $\mathbf{m}$ of size $|\mathbf{s}|/n_m$ from D^m. The remaining subroutines use subsamples from $\mathbf{m}$. Therefore, if the sample complexity of the remaining subroutines is bounded by some value N, then $\texttt{rMedian}_{\rho,d,\tau,\delta}$ will have sample complexity Nn_m. We now consider the sequence of subroutines and their respective complexities.

1. Line 10: $\texttt{rMedian}_{\rho,\lceil \log d \rceil,\tau,\delta}$ requires $n_{\lceil \log d \rceil}$ examples from $D_{\lceil \log d \rceil}$. Line 8 generates an example from $D_{\lceil \log d \rceil}$ from 2 examples from D^m, and so the call to $\texttt{rMedian}_{\rho,\lceil \log d \rceil,\tau,\delta}$ at Line 10 contributes $2n_{\lceil \log d \rceil}$ to the sample complexity.
2. Line 13 and Line 16: $\texttt{rHeavyHitters}_{\rho_0,v,\varepsilon}$ requires n_h examples from D^m
3. Line 19: the at most 3 calls to $\texttt{rSTAT}_{\rho_0,\tau,\phi_v}$ require $3n_{sq}$ examples from D^m
4. Line 30: $\texttt{rSTAT}_{\rho_0,\tau,\phi_{s_0}}$ requires n_{sq} examples from D^m

Therefore $\texttt{rMedian}_{\rho,d,\tau,\delta}$ uses $n = n_m(2n_{\lceil \log d \rceil} + 2n_h + 4n_{sq})$ examples from D.

In the base case, the entire contribution to the sample complexity comes from the call to $\texttt{Median}_{\tau,\delta_0}$, which requires n_m examples from D_1. Unrolling the recursion, we have

$$
\begin{aligned}
n &\in O\left((2n_m)^{\log^*|\mathscr{X}|}(n_h + n_{sq})\right)\\
&\in \tilde{O}\left(\left(\frac{1}{\tau^2(\rho-\delta)^2}\right)\cdot\left(\frac{3\log(2/\delta_0)}{\tau^2}\right)^{\log^*|\mathscr{X}|}\right).
\end{aligned}
$$

□

Lemma 4.5.6 (Accuracy)**.** *Let $\rho, \tau, \delta \in [0,1]$ and let n denote the sample complexity proved in Lemma 4.5.5. Let $\mathbf{s}$ be a sample of elements drawn i.i.d. from D such that $|\mathbf{s}| \in \Omega(n)$. Then* `rMedian`$(\mathbf{s})$ *returns a τ-approximate median of D except with probability δ.*

Proof. First, we prove that `rMedian`$(\mathbf{s})$ returns a τ-approximate median of D, conditioned on the success of all recursive calls and subroutines. We proceed inductively. In the base case we have that $|\mathscr{X}| = 2$, and therefore at least one of the two elements in $\mathscr{X}$ must be a τ-approximate median. The statistical query performed in line 6 of Algorithm 16 uses sample $\mathbf{s}$ to estimate the fraction of D_1 supported on 0, to within tolerance $\tau/2$, so long as $|\mathbf{s}| \geq n_m$. This holds from Lemma 4.5.5, and so a τ-approximate median for D_1 is returned in the base case.

It remains to show that if a τ-approximate median for $D_{\lceil \log d \rceil}$ is returned at Line 10 of Algorithm 16, that a τ-approximate median for D is returned. We first note that, except with probability $\delta_0 \cdot |\mathbf{s}|/n_m$, all elements of $\mathbf{m}$ are τ-approximate medians of D. To generate the sample supplied to `rMedian` at Line 13, we pair up the elements of $\mathbf{m}$ to obtain the $|\mathbf{s}|/(2n_m)$ l_i, which denote the longest prefix on which a pair of elements from $\mathbf{m}$ agree. Then $\mathbf{s}_{rm}$ constitutes a sample of size $n_{\lceil \log d \rceil}$ drawn i.i.d. from $D_{\lceil \log d \rceil}$ and by inductive assumption the call to `rMedian` at Line 10 returns a τ-approximate median of $D_{\lceil \log d \rceil}$. Therefore we have that $\mathbf{Pr}_{x_1,x_2\sim D^m}[x_{1|\ell} = x_{2|\ell}] \geq 1/2 - \tau$ and $\mathbf{Pr}_{x_1,x_2\sim D^m}[x_{1|\ell+1} = x_{2|\ell+1}] < 1/2 + \tau$. It follows that there must exist a prefix s of length ℓ such that $\mathbf{Pr}_{x\sim D^m}[x_{|\ell} = s] \geq 1/4$.

If $\ell = d$, then $x_{|\ell} = x$, and so any prefix s such that $\mathbf{Pr}_{x\sim D^m}[x_{|\ell} = s] \geq 1/4$ is a 3/8-median of D^m and therefore a τ-median of D. In this case s is returned at Line 18.

For the remainder of the proof, we assume $\ell < d$. We argue that there must exist a prefix s of length ℓ or $\ell+1$ for which $1/4 \leq \mathbf{Pr}_{x\sim D^m}[x_{||s|} = s] \leq 3/4$. We already have that there exists a prefix s of length ℓ such that $\mathbf{Pr}_{x\sim D^m}[x_{\ell} = s] \geq 1/4$. Suppose that $\mathbf{Pr}_{x\sim D^m}[x_{|\ell} = s] > 3/4$. Now suppose that one of $s||0$ or $s||1$ had probability greater than $3/4$ under D^m. Then it must be the case that $\mathbf{Pr}_{x_1,x_2\sim D^m}[x_{1|\ell+1} = x_{2|\ell+1}] > 9/16$, and so $\mathbf{Pr}_{\ell'\sim D_{\lceil \log d\rceil}}[\ell' \leq \ell] < 1-9/16 = 7/16$, contradicting that ℓ is a τ-approximate median of $D_{\lceil \log d\rceil}$. So both $s||0$ and $s||1$ must have probability less than 3/4 under D^m. Because s has probability at least 3/4, it follows that at least one of $s||0$ and $s||1$ must have probability at least 1/4 under D^m, and so we have that there exists a prefix s' of length $\ell+1$ such that $1/4 \leq \mathbf{Pr}_{x\sim D^m}[x_{|\ell+1} = s'] \leq 3/4$.

Now that we have the existence of such a prefix, we will argue that when the loop of Line 19 terminates, s is a prefix satisfying

$$1/4 - \tau \leq \mathop{\mathbf{Pr}}_{x_1\sim D^m}[x_{1|\ell} = s] \leq 3/4 + \tau.$$

Observe that the calls to `rHeavyHitters` at Line 13 and Line 16 identify a common prefix s such that $\mathbf{Pr}_{x_1\sim D^m}[x_{1|\ell} = s] \geq 1/4$. This follows from taking $v = 5/16$, $\varepsilon = 1/16$, and the fact that the sample $\mathbf{s}_{\ell}$ and $\mathbf{s}_{\ell+1}$ constitute i.i.d. samples of size n_h drawn from $D^m_{|\ell}$ and $D^m_{|\ell+1}$ respectively (where we use $D^m_{|\ell}$ to indicate the distribution induced by sampling from D^m and returning only the first ℓ bits). Then we have from the proof of Lemma 4.4.3 that all $v - \varepsilon = 1/4$-heavy hitters from $D^m_{|\ell}$ and $D^m_{|\ell+1}$ are contained in the set V. The loop beginning at Line 19 will use reproducible statistical queries to estimate the probability of each $v \in V$ under $D^m_{||v|}$. If the estimated probability $p_v \in [1/4, 3/4]$, then v is stored in s, and so the last such string visited by the loop is the value of s upon termination.

Now we show that if $s_0 = s||0\cdots 0$ is returned at Line 32, then it is a τ-approximate median of D, otherwise $s_1 = s||1\cdots 1$ is a τ-approximate median. Conceptually, we partition the domain $\mathscr{X}$ into three sets:

1. $C_{s_0} = \{x \in \mathscr{X} : x < s_0\}$
2. $C_s = \{x \in \mathscr{X} : s_0 \leq x \leq s_1\}$

3. $C_{s_1} = \{x \in \mathscr{X} : x > s_1\}$

Because s satisfies $1/4 - \tau \leq \mathbf{Pr}_{x \sim D^m}[x_{||s|} = s] \leq 3/4 + \tau$, it must be the case that D^m assigns probability mass at least $1/4 - \tau$ to the union $C_{s_0} \cup C_{s_1}$. Then it holds that at least one of C_{s_0} and C_{s_1} is assigned probability mass at least $1/8 - \tau/2$. The statistical query made at Line 30 estimates the probability mass assigned to C_{s_0} by D^m to within tolerance τ, so if s_0 is returned, it holds that $\mathbf{Pr}_{x \sim D^m}[x < s_0] \geq 1/8 - 3\tau$. Because we know $\mathbf{Pr}_{x \sim D^m}[x \in C_s] \geq 1/4 - \tau$, we then also have that $\mathbf{Pr}_{x \sim D^m}[x \geq s_0] \geq 1/4 - \tau$. Because D^m is a distribution over τ-approximate medians of D, we have that s_0 is a τ-approximate median of D as desired. If s_0 is not returned, then it must be the case that $\mathbf{Pr}_{x \sim D^m}[x > s_1] \geq 1/8 - 3\tau$, and a similar argument shows that s_1 must be a τ-approximate median of D.

Finally, we argue that all recursive calls and subroutines are successful, except with probability δ. Failures can occur exclusively at the following calls.

- Line 6: the $\log^* |\mathscr{X}| \cdot |\mathbf{s}|/(n_m)$ calls to $\texttt{Median}_{\tau,\delta_0}$
- Line 10: the $\log^* |\mathscr{X}|$ recursive calls to $\texttt{rMedian}_{\rho,\lceil \log d \rceil,\tau,\delta}$
- Line 13 and Line 16: the $2\log^* |\mathscr{X}|$ calls to $\texttt{rHeavyHitters}_{\rho_0,v,\varepsilon}$
- Line 19: the (at most) $4\log^* |\mathscr{X}|$ calls to $\texttt{rSTAT}_{\rho_0,\tau,\phi_v}$
- Line 30: the $\log^* |\mathscr{X}|$ calls to $\texttt{rSTAT}_{\rho_0,\tau,\phi_{s_0}}$

Calls to $\texttt{Median}_{\tau,\delta_0}$ dominate the total failure probability, and so taking $\delta_0 \in O(\frac{\delta}{|\mathbf{s}| \log^* |\mathscr{X}|})$ suffices to achieve failure probability δ. □

Lemma 4.5.7 (Reproducibility)**.** *Let $\rho, \tau, \delta \in [0,1]$ and let n denote the sample complexity proved in Lemma 4.5.5. Let* $\mathbf{s}$ *be a sample of $O(n)$ elements drawn i.i.d. from D. Then* $\texttt{rMedian}_{\rho,d,\tau,\delta}$ *is ρ-reproducible.*

Proof. We prove the lemma by inductive argument. First, we observe that reproducibility of the value returned in the base case depends only on the value $p_0 \leftarrow \texttt{rSTAT}_{\rho_0,\tau/2,\phi_0}(\mathbf{s})$ in Line 3. Therefore, reproducibility in the base case follows from the ρ_0-reproducibility of $\texttt{rSTAT}_{\rho_0,\tau/2,\phi_0}$.

We now argue that if the $i+1$th recursive call is ρ-reproducible, that the ith recursive call is $(\rho+5\rho_0)$-reproducible.

Two parallel executions of the ith level of recursion, given samples $\mathbf{s}_1$ and $\mathbf{s}_2$ drawn i.i.d. from the same distribution D, will produce the same output so long as the following values are the same:

1. $\ell \leftarrow \texttt{rMedian}_{\rho,d,\tau,\delta}(\mathbf{s}_{rm})$ at Line 10

2. $V \leftarrow \texttt{rHeavyHitters}_{\rho_0,v,\varepsilon}(\mathbf{s}_{\ell})$ at Line 13

3. $V \leftarrow V \cup \texttt{rHeavyHitters}_{\rho_0,v,\varepsilon}(\mathbf{s}_{\ell+1})$ at Line 16

4. $s \leftarrow \texttt{rSTAT}_{\rho_0,\tau,\phi_{s_0}}(\mathbf{s}_{meds})$ when the loop at Line 19 terminates

5. $p_{s_0} \leftarrow \texttt{rSTAT}_{\rho_0,\tau,\phi_{s_0}}(\mathbf{s}_{s_0})$ at Line 30

produce the same value. We have that 1 holds by inductive assumption.

Conditioning on 1, the calls to $\texttt{rHeavyHitters}_{\rho_0,v,\varepsilon}$ are made on samples drawn i.i.d. from the same distribution, and so the ρ_0-reproducibility of $\texttt{rHeavyHitters}_{\rho_0,v,\varepsilon}$ guarantees that V contains the same list of heavy-hitters in both runs except with probability 2ρ.

Conditioning on both 1 and 2, it follows that the loop at Line 19 iterates over the same strings V, and so both runs make the same sequence of statistical queries $\texttt{rSTAT}_{\tau,\rho_0,\phi_v}$. From conditioning on 2, and the values of v and ε, we have that $|V| \leq 3$, and so the ρ_0-reproducibility of $\texttt{rSTAT}_{\tau,\rho_0,\phi_{s_0}}$ gives us that sequence of values $p_v \leftarrow \texttt{rSTAT}_{\rho_0,\tau,\phi_v}(\mathbf{s}_q)$ is the same in both runs, except with probability 3_{ρ_0}.

Finally, conditioning on 1, 2, and 3, the values of s_0 and s_1 are the same across both runs, and so the same statistical query $\texttt{rSTAT}_{\rho_0,\tau,\phi_{s_0}}$ is made in both runs. Whether s_0 or s_1 is returned depends only on the value $r_{s_0} \leftarrow \texttt{rSTAT}_{\tau,\rho_0,\phi_{s_0}}(\mathbf{s}_{s_0})$, and so the ρ_0-reproducibility of $\texttt{rSTAT}_{\rho_0,\tau,\phi_{s_0}}$ gives us that the same string is returned by both executions. A union bound over all failures of reproducibility then gives us that the ith recursive call will be $(\rho+6\rho_0)$-reproducible.

From Lemma 4.5.4, we have that no more than $T = \log^*|\mathscr{X}|$ recursive calls are made by the algorithm. Therefore $\texttt{rMedian}_{\rho,d,\tau,\delta}$ is reproducible with parameter $\rho_0 + 5T\rho_0 \leq 6\rho_0 \log^*|\mathscr{X}| =$

ρ. $\square$

Theorem 4.5.8 then follows as a corollary of Lemma 4.5.5, Lemma 4.5.6, and Lemma 4.5.7.

Theorem 4.5.8 (Reproducible Median)**.** *Let $\tau, \rho \in [0,1]$ and let $\delta = \rho/2$. Let D be a distribution over $\mathscr{X}$, where $|\mathscr{X}| = 2^d$. Then* `rMedian`$_{\rho,d,\tau,\delta}$ *(Algorithm 16) is ρ-reproducible, outputs a τ-approximate median of D with all but probability δ, and has sample complexity*

$$n \in \tilde{O}\left(\left(\frac{1}{\tau^2(\rho-\delta)^2}\right) \cdot \left(\frac{3}{\tau^2}\right)^{\log^* |\mathscr{X}|}\right)$$

4.6 Learning Halfspaces

In Section 4.3, we saw how combining a concentration bound with a randomized rounding technique yielded a reproducible algorithm. Specifically, given a statistical query algorithm with an accuracy guarantee (with high probability) on the 1-dimensional space $[0,1]$, we can construct a reproducible statistical query algorithm using randomized rounding. By sacrificing a small amount of accuracy, our reproducible statistical query algorithm can decide on a canonical return value in $[0,1]$.

In this section, we extend this argument from $\mathbb{R}$ to $\mathbb{R}^d$, by way of an interesting application of a randomized rounding technique from the study of foams Kindler et al. [2012]. Algorithm 1 in Kindler et al. [2012] probabilistically constructs a tiling of $\mathbb{R}^d$ such that every point is rounded to a nearby integer lattice point. This tiling has an additional property that the probability that two points are not rounded to the same point by a constructed tiling is at most linear in their l_2 distance. In the usual PAC-learning setting, there is a simple weak learning algorithm for halfspaces that takes examples $(\mathbf{x}_i, y_i) \in \mathscr{X} \times \{\pm 1\}$, normalizes them, and returns the halfspace defined by vector $\sum_i \mathbf{x}_i \cdot y_i$ Servedio [2002]. We show a concentration bound on the sum of normalized vectors from a distribution, and then argue that all vectors within the concentration bound are reasonable hypotheses with non-negligible advantage. The combination of this concentration bound and the foam-based rounding scheme yields a reproducible halfspace weak learner `rHalfspaceWkL`.

However, constructing this foam-based rounding scheme takes expected time that is exponential in the dimension d. We give an alternative rounding scheme that randomly translates the integer lattice and rounds points to their nearest translated integer lattice point. This construction yields another reproducible halfspace weak learner $\texttt{rHalfspaceWkL}^{\text{box}}$ with roughly an additional factor of d in the sample complexity, but with polynomial runtime. In Section 4.7, we show how to combine these reproducible weak learners with a reproducible boosting algorithm, yielding an polynomial-time reproducible strong learner for halfspaces.

4.6.1 Reproducible Halfspace Weak Learner: An Overview

Let D be a distribution over $\mathbb{R}^d$, and let EX be an example oracle for D and f, where $f : \mathbb{R}^d \to \{\pm 1\}$ is a halfspace that goes through the origin. Let $||\mathbf{x}||$ denote the l_2 norm of vector $\mathbf{x}$. We assume that D satisfies a (worst-case) margin assumption with respect to f.

Definition 4.6.1. [Margin] Let D be a distribution over $\mathbb{R}^d$. We say D has margin τ_f with respect to halfspace $f(\mathbf{x}) \overset{\text{def}}{=} \text{sign}(\mathbf{w} \cdot \mathbf{x})$ if $\frac{\mathbf{x} \cdot f(\mathbf{x})}{||\mathbf{x}||} \cdot \frac{\mathbf{w}}{||\mathbf{w}||} \geq \tau_f$ for all $x \in \text{supp}(D)$. Additionally, we say D has (worst-case) margin τ if $\tau = \sup_f \tau_f$.

Our reproducible halfspace weak learner `rHalfspaceWkL` uses its input to compute an empirical estimation $\mathbf{z}$ of the expected vector $\mathbb{E}_{\mathbf{x} \sim D}[\mathbf{x} \cdot f(x)]$. Then, `rHalfspaceWkL` uses its randomness to construct a rounding scheme R via Algorithm `ConstructFoams`. R is used to round our (rescaled) empirical estimation $\mathbf{z}$, and the resulting vector defines the returned halfspace. The algorithm relies on the margin assumption to ensure that the weak learner's returned hypothesis is positively correlated with the true halfspace f.[1]

[1]The parameter a is a constant, but we leave it in variable form for convenience in the analysis; we choose $a = .05$ in this proof for clarity of presentation, but one could optimize the choice of a to yield a slightly better sample complexity.

Algorithm 17. `rHalfspaceWkL`$(\mathbf{s}; r)$
Parameters: ρ - desired reproducibility
d - dimension of halfspace
τ - assumed margin
a - a constant, $a = .05$
Input: A sample $\mathbf{s}$ of $m = \left(\frac{896\sqrt{d}}{\tau^2\rho}\right)^{1/(1/2-a)}$ examples $(\mathbf{x}_i, y_i)$ drawn i.i.d. from distribution D
Output: A hypothesis with advantage $\tau/4$ on D against f

```
$k \leftarrow \frac{1}{m}\frac{8\sqrt{d}}{\tau^2} = 8 \cdot \left(\frac{\rho}{896}\right)^{1/(1/2-a)} \left(\frac{\tau^2}{\sqrt{d}}\right)^{(1/2+a)/(1/2-a)}$    /* Scaling factor */
$\mathbf{z} \leftarrow \sum_S \frac{\mathbf{x}_i}{\|\mathbf{x}_i\|} \cdot y_i$
$R \leftarrow_r$ ConstructFoams$(d)$ (Algorithm 18)   /* Rounding scheme $R : \mathbb{R}^d \to \mathbb{Z}^d$ */
$\mathbf{w} \leftarrow R(k \cdot \mathbf{z})$
return Hypothesis $h(\mathbf{x}) \overset{\text{def}}{=} \frac{\mathbf{x}}{\|\mathbf{x}\|} \cdot \frac{\mathbf{w}}{\|\mathbf{w}\|}$
```

The subroutine `ConstructFoams` previously appeared as Algorithm 1 in Kindler et al. [2012]. For completeness, we include a description below (Algorithm 18).

Algorithm 18. `ConstructFoams`(d)
Input: dimension d
Output: rounding scheme $R : \mathbb{R}^d \to \mathbb{Z}^d$

```
Let $f : [0,1]^d \to \mathbb{R}$ s.t. $f(x_1, \dots, x_d) \overset{\text{def}}{=} \prod_{i=1}^d (2\sin^2(\pi x_i))$
Let all points in $\mathbb{R}^d$ be unassigned
for stage $t = 1, 2, \dots$ until all points are assigned do
    Uniformly at random sample $Z_t, H_t$ from $[0,1)^d \times (0, 2^d)$.
    Let droplet $D_i$ be the set of points $\{x | x \in -Z_t + [0,1)^d, f(x + Z_t) > H_t\}$.
    Let $R$ map all currently unassigned points in $D_i$ to $(0, 0, \dots, 0)$ and extend this assignment
    periodically to all integer lattice points.
return $R$
```

The following is the main result of this section.

Theorem 4.6.2. *Let D be a distribution over $\mathbb{R}^d$, and let $f : \mathbb{R}^d \to \{\pm 1\}$ be a halfspace with margin τ in D. Then* rHalfspaceWkL$(\mathbf{s}; r)$ *is a $(\rho, \tau/4, \rho/4)$-weak learner for halfspaces. That is, Algorithm 17 ρ-reproducibly returns a hypothesis h such that, with probability at least $1-\rho/2$, $\frac{1}{2}\mathbb{E}_{\mathbf{x}\sim D} h(\mathbf{x})f(\mathbf{x}) \geq \tau/4$, using a sample of size $m = \left(\frac{896\sqrt{d}}{\tau^2\rho}\right)^{20/9}$.*

Proof. **Correctness (Advantage):** We argue correctness in two parts. First, we show the expected weighted vector $\mathbb{E}_{\mathbf{x}\sim D}\left[\frac{\mathbf{x}\cdot f(\mathbf{x})}{\|\mathbf{x}\|}\right]$ defines a halfspace with good advantage (see Lemma 4.6.8), following the arguments presented in Theorem 3 of Servedio [2002]. Then, we argue that rounding the empirical weighted vector $\mathbf{z}$ in Algorithm 17 only slightly rotates the halfspace. By bounding the possible loss in advantage in terms of the amount of rotation (Lemma 4.6.9), we argue that the rounded halfspace $\mathbf{w}/|\mathbf{w}|$ also has sizable advantage.

By Lemma 4.6.8, the expected weighted vector $\mathbb{E}_{\mathbf{x}\sim D}\left[\frac{\mathbf{x}\cdot f(\mathbf{x})}{\|\mathbf{x}\|}\right]$ has advantage $\tau/2$ on D and f. The martingale-based concentration bound in Corollary 4.6.10 implies that the distance between $\mathbf{z}$ and $\mathbb{E}[\mathbf{z}] = m \cdot \mathbb{E}_{\mathbf{x}\sim D}\left[\frac{\mathbf{x}\cdot f(\mathbf{x})}{\|\mathbf{x}\|}\right]$ is less than $4m^{1/2+a}$ with probability at least $1 - e^{-m^{2a}/2}$ for any $a \in (0, 1/2)$ (chosen later). Then, the vector is scaled by k and rounded. Any rounding scheme R randomly generated by `ConstructFoams` always rounds its input to a point within distance $\sqrt{d}$ (Observation 4.6.4). Combining, the total distance between vectors $\frac{\mathbf{w}}{k\cdot\|\mathbb{E}[\mathbf{z}]\|}$ and $\frac{\mathbb{E}[\mathbf{z}]}{\|\mathbb{E}[\mathbf{z}]\|}$ is at most

$$\frac{4m^{1/2+a} + \sqrt{d}/k}{\|\mathbb{E}[\mathbf{z}]\|}.$$

As D has margin τ with respect to f, for all $\mathbf{x} \in \text{supp}(D)$, $\frac{\mathbf{x}}{\|\mathbf{x}\|} \cdot f(\mathbf{x})$ has length at least τ in the direction of the expected weighted vector $\mathbb{E}_{\mathbf{x}\sim D}\left[\frac{\mathbf{x}\cdot f(\mathbf{x})}{\|\mathbf{x}\|}\right]$.

Thus, $\|\mathbb{E}[\mathbf{z}]\| \geq \tau m$, and the above quantity is at most $\frac{4m^{1/2+a}+\sqrt{d}/k}{\tau m}$. Simplifying, $\frac{4m^{1/2+a}}{\tau m} = \frac{4}{\tau m^{1/2-a}} = \frac{4\tau}{896}\frac{\rho}{\sqrt{d}} < \tau/8$ and $\frac{\sqrt{d}/k}{\tau m} = \frac{\sqrt{d}}{\tau}\frac{\tau^2}{8\sqrt{d}} = \tau/8$. By applying Lemma 4.6.9 with $\theta = \tau/8 + \tau/8 = \tau/4$, we can conclude that h has advantage at least $\tau/2 - \tau/4 = \tau/4$, as desired.[2]

[2]A dedicated reader may notice that the scaling factor k is subconstant. A possible error may arise if the scaling factor is so small that the halfspace vector $\mathbf{z}\cdot k$ gets rounded to 0 by the rounding function R (constructed by `ConstructFoams`). Fortunately, with our choice of parameters, this turns out to not be an issue. The empirical vector sum $\mathbf{z}$ has norm at least $\tau \cdot m$, where τ is the margin size and m is the sample complexity. As we have chosen scaling factor k such that $m \cdot k = 8\sqrt{d}/\tau^2$, the input given to R has norm at least $8\sqrt{d}/\tau$. Every rounding function R constructed by

Reproducibility: Let $\mathbf{z}_1$ and $\mathbf{z}_2$ denote the empirical sums of vectors $\mathbf{x}_i y_i$ from two separate runs of `rHalfspaceWkL`. It suffices to show that the rounding scheme R constructed by `ConstructFoams` rounds $k\cdot\mathbf{z}_1$ and $k\cdot\mathbf{z}_2$ to the same vector $\mathbf{w}$ with high probability. The distance between $\mathbf{z}_1$ and $\mathbf{z}_2$ is at most $2\cdot 4m^{1/2+a}$ with probability at least $1-2e^{-m^{2a}/2}$, by Corollary 4.6.10, the triangle inequality, and a union bound. After scaling by k, this distance is at most $8km^{1/2+a}$. By Lemma 4.6.3, the probability that R does not round $k\cdot\mathbf{z}_1$ and $k\cdot\mathbf{z}_2$ to same integer lattice point is at most $7\cdot 8km^{1/2+a}$. Altogether, the reproducibility parameter is at most

$$2e^{-m^{2a}/2}+56km^{1/2+a}.$$

The second term satisfies $56km^{1/2+a}=448\cdot\frac{\rho}{896}\cdot 1=\rho/2$, and the first term $2e^{-m^{2a}/2}\leq\rho/2$ when $m\geq(2\ln(4/\rho))^{1/(2a)}$. So, as long as a is chosen such that $m=\left(\frac{896\sqrt{d}}{\tau^2\rho}\right)^{1/(1/2-a)}\geq(2\ln(4/\rho))^{1/(2a)}$, the algorithm is ρ-reproducible. This occurs if $\left(\frac{896}{\rho}\right)^{2a/(1/2-a)}\geq 2\ln(4/\rho)$, which is true for all values of $\rho\in(0,1)$ when $a=.05$.[3]

Failure rate: The algorithm succeeds when the martingale concentration bound holds. So, the failure probability of `rHalfspaceWkL` is at most $e^{-m^{2a}/2}\leq\rho/4$.

Sample complexity: Plugging in $a=.05$ in the expression $m=\left(\frac{896\sqrt{d}}{\tau^2\rho}\right)^{1/(1/2-a)}$ yields the conclusion. $\square$

4.6.2 Reproducible Weak Halfspace Learner – Definitions and Lemmas

Foams-Based Rounding Scheme from Kindler et al. [2012]

For completeness, we restate relevant results from Kindler et al. [2012] for our construction.

Lemma 4.6.3 (Combining Theorem 1 and Theorem 3 of Kindler et al. [2012]). *Let $R:\mathbb{R}^d\to\mathbb{Z}^d$ be the randomized rounding scheme constructed by Algorithm 18 (Algorithm 1 in Kindler et al. [2012]). Let $x,y\in\mathbb{R}^d$, and let $\varepsilon\overset{\text{def}}{=}d_{l_2}(x,y)$. Then $\mathbf{Pr}[R(x)=R(y)]\geq 1-O(\varepsilon)$, where the probability is over the randomness used in the algorithm.*

`ConstructFoams` rounds its input to a point at distance at most $\sqrt{d}$ away (Observation 4.6.4), so we can be sure that R never rounds our vector to the zero vector.

[3]The constant a can be improved slightly if a is chosen as a function of ρ.

Proof. Theorem 3 of Kindler et al. [2012] states that $f(\mathbf{x}) = \Pi_{i=1}^{d}(2\sin^2(\pi x_i))$ is a *proper* density function and $\int_{[0,1)^d} |\langle \nabla f, u\rangle| \leq 2\pi$ for all unit vectors u. Theorem 1 of Kindler et al. [2012] states the following. Let f be a proper density function, and points $x, y \in \mathbb{R}^d$ such that $y = x + \varepsilon \cdot u$, where $\varepsilon > 0$ and u is a unit vector. Let N denote the number of times the line segment $\overline{xy}$ crosses the boundary between different droplets (potentially mapping to the same integer lattice point) in an execution of `ConstructFoams`. Then $\mathbb{E}[N] \approx \varepsilon \cdot \int_{[0,1)^d]} |\langle \nabla f, u\rangle|$, where the $\approx$ notation is hiding a $W\varepsilon^2$ term, where $W > 0$ is a universal constant depending only on f. The authors refine this statement (Kindler et al. [2012], page 24) to show that the $W\varepsilon^2$ term can be made arbitrarily small. Combining, $\mathbb{E}[N] \leq 2\pi\varepsilon + W\varepsilon^2 < 6.3\varepsilon$. By Markov's inequality, $\mathbf{Pr}[N = 0] < 1 - 6.3\varepsilon$. □

Observation 4.6.4. `ConstructFoams` *always outputs a rounding scheme R with the following property: the (l_2) distance between any vector $\mathbf{v} \in \mathbb{R}^d$ and $R(\mathbf{v})$ is at most $\sqrt{d}$.*

This follows from noticing that R maps each coordinate of $\mathbf{v}$ to its floor or ceiling.

Theorem 4.6.5 (Runtime of `ConstructFoams`; Kindler et al. [2012], page 25)**.** *There are universal constants $1 < c < C$ such that Algorithm 18, when run with $f(\mathbf{x}) = \Pi_{i=1}^{d}(2\sin^2(\pi x_i))$, takes between c^d and C^d stages except with probability at most c^{-d}.*

Weak Learning Definitions

Definition 4.6.6 (Weak Learning Algorithm (in the Filtering Model))**.** Let $\mathscr{C}$ be a concept class of functions from domain $\mathscr{X}$ to $\{\pm 1\}$, and let $f \in \mathscr{C}$. Let D be a distribution over $\mathscr{X}$. Let `WkL` be an algorithm that takes as input a labeled sample $S = \{(x_i, f(x_i))\}_m$ drawn i.i.d. from D, and outputs a hypothesis $h : \mathscr{X} \to [-1, 1]$. Then `WkL` is a (γ, δ)-*weak learner* for $\mathscr{C}$ with sample complexity m if, for all f, D, with probability at least $1 - \delta$, `WkL`(S) outputs a hypothesis $h : \mathscr{X} \to [-1, 1]$ such that $\mathbb{E}_{x \sim D} f(x)h(x) \geq 2\gamma$, where S is a sample of size $|S| = m$ drawn i.i.d. from D.

We say a (γ, δ)-weak learner has *advantage* γ. Equivalently, if a hypothesis h satisfies $\frac{1}{2}\mathbb{E}_{x \sim D} f(x)h(x) \geq \gamma$, then we say h has advantage γ (on D and f).

Definition 4.6.7 (Reproducible Weak Learning Algorithm)**.** Algorithm `rWkL` is a (ρ, γ, δ)-weak learner if `rWkL` is ρ-reproducible and a (γ, δ)-weak learner.

Halfspaces and Their Advantage

Definition 4.6.1. [Margin] Let D be a distribution over $\mathbb{R}^d$. We say D has margin τ_f with respect to halfspace $f(\mathbf{x}) \overset{\text{def}}{=} \text{sign}(\mathbf{w}\cdot\mathbf{x})$ if $\frac{\mathbf{x}\cdot f(\mathbf{x})}{||\mathbf{x}||} \cdot \frac{\mathbf{w}}{||\mathbf{w}||} \geq \tau_f$ for all $x \in \text{supp}(D)$. Additionally, we say D has (worst-case) margin τ if $\tau = \sup_f \tau_f$.

Lemma 4.6.8 (Advantage of Expected Weighted Vector Hypothesis Servedio [2002])**.** *Let $f(\mathbf{x}) \overset{\text{def}}{=} \text{sign}(\mathbf{w}\cdot\mathbf{x})$ be a halfspace, and let D be a distribution over $\mathbb{R}^d$ with margin τ with respect to f. Let $\mathbf{z} = \mathbb{E}_{\mathbf{v}\sim D}\left[\frac{\mathbf{v}}{||\mathbf{v}||} f(\mathbf{v})\right]$. Then the hypothesis $h_{\mathbf{z}}(\mathbf{x}) = \frac{\mathbf{x}}{||\mathbf{x}||}\cdot\frac{\mathbf{z}}{||\mathbf{z}||}$ has advantage at least $\tau/2$.*

Proof. The advantage of $h_{\mathbf{z}}$ is $\frac{1}{2}\mathbb{E}_{\mathbf{x}\sim D}[h_{\mathbf{z}}(\mathbf{x})f(\mathbf{x})] = \frac{1}{2}\mathbb{E}_{\mathbf{x}\sim D}\left[\frac{\mathbf{x}}{||\mathbf{x}||}\cdot\frac{\mathbf{z}}{||\mathbf{z}||}\cdot f(\mathbf{x})\right] = \frac{\mathbf{z}\cdot\mathbf{z}}{2||\mathbf{z}||} = \frac{||\mathbf{z}||}{2} \geq \frac{\mathbf{z}\cdot\mathbf{w}}{2||\mathbf{w}||}$, by the Cauchy-Schwarz inequality. Vector $\mathbf{z}$ is a convex combination of $\frac{\mathbf{x}\cdot f(\mathbf{x})}{||\mathbf{x}||}$ terms, for $\mathbf{x} \in \text{supp}(D)$. By the margin assumption, $\frac{\mathbf{x}\cdot f(\mathbf{x})}{||\mathbf{x}||}\cdot\frac{\mathbf{w}}{||\mathbf{w}||} \geq \tau$ for all $x \in \text{supp}(D)$. Thus, $\frac{\mathbf{z}\cdot\mathbf{w}}{2||\mathbf{w}||} \geq \frac{\tau}{2}$.

□

Lemma 4.6.9 (Advantage of Perturbed Halfspaces)**.** *Consider a halfspace defined by unit vector $\mathbf{w}$, and let $h(\mathbf{x}) = \frac{\mathbf{x}}{||\mathbf{x}||}\cdot\mathbf{w}$. Assume h has advantage γ, i.e. $\frac{1}{2}\mathbb{E}_{x\sim D} f(x)h(x) \geq \gamma$. Let $\mathbf{u}$ be any vector such that $||\mathbf{u}|| \leq \theta$, where $\theta \in [0, \sqrt{3}/2)$. Let perturbed vector $\mathbf{w}' = \frac{\mathbf{w}+\mathbf{u}}{||\mathbf{w}+\mathbf{u}||}$, and let $h'(\mathbf{x}) = \frac{\mathbf{x}}{||\mathbf{x}||}\cdot\mathbf{w}'$. Then h' has advantage at least $\gamma - \theta$.*

Proof. First, we bound the maximum distance between $\mathbf{w}$ and $\mathbf{w}'$. Then, we apply Cauchy-Schwarz to bound the advantage loss. $\mathbf{w}'$ is constructed by perturbing $\mathbf{w}$ by a vector $\mathbf{u}$, and then normalizing to norm 1. $\mathbf{w}'$ is furthest away from $\mathbf{w}$ when the vector $\mathbf{w}'$ is tangent to the ball of radius $||\mathbf{u}||$ around $\mathbf{w}$. In this case, $||\mathbf{w}'-\mathbf{w}||^2 = (1-\sqrt{1-\theta^2})^2 + \theta^2 = 2 - 2\sqrt{1-\theta^2}$. Since $\theta < \sqrt{3}/2$, $2 - 2\sqrt{1-\theta^2} < 4\theta^2$. So, $||\mathbf{w}'-\mathbf{w}||^2 < 4\theta^2$. The advantage of h' is

$$\begin{aligned}\frac{1}{2}\mathbb{E}_{\mathbf{x}\sim D}\left[\frac{\mathbf{x}}{||\mathbf{x}||}\cdot\mathbf{w}'\cdot f(\mathbf{x})\right] &= \frac{1}{2}\mathbb{E}_{\mathbf{x}\sim D}\left[\frac{\mathbf{x}}{||\mathbf{x}||}\cdot(\mathbf{w}+(\mathbf{w}'-\mathbf{w}))\cdot f(\mathbf{x})\right] \\ &= \gamma + \frac{1}{2}(\mathbf{w}'-\mathbf{w})\cdot\mathbb{E}_{\mathbf{x}\sim D}\left[\frac{\mathbf{x}}{||\mathbf{x}||}\cdot f(\mathbf{x})\right]\end{aligned}$$

By Cauchy-Schwarz, the second term of the right-hand side has magnitude at most $\sqrt{4\theta^2\cdot 1}/2$, so the advantage of h' is at least $\gamma - \theta$. □

Concentration Bound on Sum of Normalized Vectors

Let D be a distribution on $\mathbb{R}^n$. Let $\mathbf{v} = \{\mathbf{v_1}, \ldots, \mathbf{v_T}\} \in D^T$ be a random sample of T vectors from D with the following properties:

1. $\mathbb{E}_{\mathbf{v} \in D^T}[\sum_{i=1}^T \mathbf{v_i}] - \mathbb{E}_{v \in D}[v] = 0$.

2. $\forall v \in D$, $||v||_2 \leq c$.

Lemma 4.2.1. *Let $D, \mathbf{v} \in D^T$ satisfy properties (1) and (2) above, and let $\mathbf{v}^{\leq \mathbf{T}} = \sum_{i=1}^T \mathbf{v_i}$. Then for all $\Delta > 0$,*

$$\mathbf{Pr_v}[||\mathbf{v}^{\leq \mathbf{T}}||_2 \geq \sqrt{T}(1 + c/2) + \Delta] \leq e^{-\Delta^2/2c^2T}.$$

For a proof, see Section 4.2.

Corollary 4.6.10. *Let D be a distribution supported on the unit ball in d dimensions, and let f be a halfspace. Let S be a sample of T examples $(\mathbf{x}_i, y_i)$ drawn i.i.d. from D, and let $\mathbf{z} = \sum_S \mathbf{x}_i \cdot y_i$. Let $a \in (0, 1/2)$. Then $\mathbf{Pr}_{S \sim D}\left[||\mathbf{z} - \mathbb{E}_{\mathbf{v} \sim D}\mathbf{v}|| \geq 4T^{1/2+a}\right] \leq e^{-T^{2a}/2}$.*

Proof. In order to have D satisfy the properties (1) and (2) above, we must translate D by the expectation $\mathbb{E}_{\mathbf{v} \sim D}[\mathbf{v}]$. After this translation, the maximum length of a vector in the support is $c = 2$. Plugging in $\Delta = 2T^{1/2+a}$ and noting $2T^{1/2+a} \geq 2T^{1/2}$ yields the conclusion. $\square$

4.6.3 Coordinate-Based Rounding Scheme

Algorithm `rHalfspaceStL` uses polynomial sample complexity and runs in polynomial time except for subroutine `ConstructFoams`, which runs in expected exponential time in the dimension d (Theorem 4.6.5). Next, we consider a simpler rounding scheme that rounds points coordinate-by-coordinate to a randomly shifted integer lattice. This rounding scheme requires tighter concentration bounds, resulting in approximately another factor of d in the sample complexity. In return, it can be constructed by `ConstructBoxes` and executed in linear time in sample complexity m and dimension d.

Algorithm 19. `ConstructBoxes`(d)
Input: dimension d
Output: rounding scheme $R : \mathbb{R}^d \to \mathbb{R}^d$

```
Uniformly at random draw $Z$ from $[0,1)^d$.
Let box $B$ be the set of points $\{x | \forall i \in [d], x_i \in [-1/2 + z_i, 1/2 + z_i)\}$
Let $R$ map all points in $B$ to point $Z$ and extend this assignment periodically by integer lattice points
return $R$
```

Lemma 4.6.11. *Let $R : \mathbb{R}^d \to \mathbb{R}^d$ be the randomized rounding scheme constructed by* `ConstructBoxes`. *Let $\mathbf{x}, \mathbf{y} \in \mathbb{R}^d$, and let $\varepsilon \overset{\text{def}}{=} d_{l_2}(\mathbf{x}, \mathbf{y})$. Then $\mathbf{Pr}[R(\mathbf{x}) = R(\mathbf{y})] \geq 1 - d\varepsilon$.*

Proof. We bound this probability by a crude l_2 to l_1 distance conversion. If $\mathbf{x}$ and $\mathbf{y}$ have l_2 distance ε, then the distance between x_i and y_i is at most ε for all coordinates $i \in [d]$. The i'th coordinate of $\mathbf{x}$ and $\mathbf{y}$ are not rounded to the same point with probability $|x_i - y_i|$. By a union bound, $\mathbf{Pr}[R(\mathbf{x}) = R(\mathbf{y})] \geq 1 - d\varepsilon$. □

Observation 4.6.12. `ConstructBoxes` *always outputs a rounding scheme R with the following property: the (l_2) distance between any vector $\mathbf{v} \in \mathbb{R}^d$ and $R(\mathbf{v})$ is at most $\sqrt{d}/2$.*

This follows from noticing that R maps each coordinate of $\mathbf{v}$ to value within distance $1/2$.

Reproducible Halfspace Weak Learner using Boxes

Algorithm 20. `rHalfspaceWkL`$^{\text{box}}(\mathbf{s};r)$
Parameters: desired reproducibility ρ, dimension d, assumed margin τ, constant $a = .1$
Input: A sample S of $m = \left(\frac{64d^{3/2}}{\tau^2\rho}\right)^{1/(1/2-a)}$ examples $(\mathbf{x}_i, y_i)$ drawn i.i.d. from distribution D
Output: A hypothesis with advantage $\gamma/4$ on D against f

```
$k \leftarrow \frac{1}{m}\frac{4\sqrt{d}}{\tau^2} = 4 \cdot \left(\frac{\rho \cdot \tau^{1+2a}}{64 \cdot d^{5/4+a/2}}\right)^{1/(1/2-a)}$   /* Scaling factor */
$\mathbf{z} \leftarrow \sum_S \frac{\mathbf{x}_i}{\|\mathbf{x}_i\|} \cdot y_i$
$R \leftarrow_r$ ConstructBoxes$(d)$ (Algorithm 19)   /* Rounding scheme $R : \mathbb{R}^d \to \mathbb{R}^d$ */
$\mathbf{w} \leftarrow R(k \cdot \mathbf{z})$
return Hypothesis $h(\mathbf{x}) \overset{\text{def}}{=} \frac{\mathbf{x}}{\|\mathbf{x}\|} \cdot \frac{\mathbf{w}}{\|\mathbf{w}\|}$
```

Theorem 4.6.13. *Let D be a distribution over $\mathbb{R}^d$, and let $f : \mathbb{R}^d \to \{\pm 1\}$ be a halfspace with margin τ in D. Then* `rHalfspaceWkL`$(\mathbf{s};r)$ *is a $(\rho, \tau/4, \rho/4)$-weak learner for halfspaces. That is, Algorithm 17 ρ-reproducibly returns a hypothesis h such that, with probability at least* $1-\rho/2$, $\mathbf{Pr}_{\mathbf{x}\sim D} h(\mathbf{x})f(\mathbf{x}) \geq \tau/4$, *using a sample of size* $m = \left(\frac{64d^{3/2}}{\tau^2\rho}\right)^{5/2}$.

Proof. **Correctness (Advantage):** The proof proceeds almost identically to the proof of Theorem 4.6.2. By Lemma 4.6.8, the expected weighted vector $\mathbb{E}_{\mathbf{x}\sim D}\left[\frac{\mathbf{x}\cdot f(\mathbf{x})}{\|\mathbf{x}\|}\right]$ has advantage $\tau/2$ on D and f. Any rounding scheme R randomly generated by `ConstructBoxes` always rounds its input to a point within distance $\sqrt{d}/2$, so the distance between vectors $\frac{\mathbf{w}}{k\cdot\|\mathbb{E}[\mathbf{z}]\|}$ and $\frac{\mathbb{E}[\mathbf{z}]}{\|\mathbb{E}[\mathbf{z}]\|}$ is at most

$$\frac{4m^{1/2+a} + \sqrt{d}/2k}{\tau m}.$$

Simplifying, $\frac{4m^{1/2+a}}{\tau m} = \frac{4}{\tau m^{1/2-a}} = \frac{4\tau}{64}\frac{\rho}{d^{3/2}} < \tau/8$ and $\frac{\sqrt{d}/2k}{\tau m} = \frac{\sqrt{d}}{2\tau}\frac{\tau^2}{4\sqrt{d}} = \tau/8$. By applying Lemma 4.6.9 with $\theta = \tau/8 + \tau/8$, we can conclude that h has advantage at least $\tau/2 - (\tau/8 + \tau/8) = \tau/4$, as desired.

Reproducibility: Let $\mathbf{z}_1$ and $\mathbf{z}_2$ denote the empirical sums of vectors $\mathbf{x}_i y_i$ from two separate runs of `rHalfspaceWkL`. It suffices to show that the rounding scheme R constructed by `ConstructBoxes` rounds $k \cdot \mathbf{z}_1$ and $k \cdot \mathbf{z}_2$ to the same vector $\mathbf{w}$ with high probability. The distance between $\mathbf{z}_1$ and $\mathbf{z}_2$ is at most $2 \cdot 4m^{1/2+a}$ with probability at least $1 - 2e^{-m^{2a}/2}$, by Corollary 4.6.10, the triangle inequality, and a union bound. After scaling by k, this distance is at most $8km^{1/2+a}$. By Lemma 4.6.11, the probability that R does not round $k \cdot \mathbf{z}_1$ and $k \cdot \mathbf{z}_2$ to same integer lattice point is at most $d \cdot 8km^{1/2+a}$. Altogether, the reproducibility parameter is at most

$$2e^{-m^{2a}/2} + 8dkm^{1/2+a}.$$

The second term satisfies $8dkm^{1/2+a} = 8d(km/m^{1/2-a}) = \rho/2$, and the first term $2e^{-m^{2a}/2} \leq \rho/2$ when $m \geq (2\ln(4/\rho))^{1/(2a)}$. So, as long as a is chosen such that $m = \left(\frac{64d^{3/2}}{\tau^2\rho}\right)^{1/(1/2-a)} \geq (2\ln(4/\rho))^{1/(2a)}$, the algorithm is ρ-reproducible. This occurs if $\left(\frac{64}{\rho}\right)^{2a/(1/2-a)} \geq 2\ln(4/\rho)$, which is true for all values of $\rho \in (0,1)$ when $a = .07$. For simpler constants, we use $a = .1$.

Failure rate: The algorithm succeeds when the martingale concentration bound holds. So, the failure probability of `rHalfspaceWkL` is at most $e^{-m^{2a}/2} \leq \rho/4$.

Sample complexity: Plugging in $a = .1$ in the expression $m = \left(\frac{64d^{3/2}}{\tau^2\rho}\right)^{1/(1/2-a)}$ yields the conclusion. □

4.7 Reproducible Boosting

In this section, we argue that a small modification of the boosting algorithm in Servedio [2003a] is a reproducible boosting algorithm. Given access to a reproducible weak learner, this boosting algorithm ρ-reproducibly outputs a hypothesis. Boosting algorithms are a natural candidate for constructing reproducible algorithms — many boosting algorithms in the standard PAC-setting are deterministic, and the final classifier returned is often a simple function of the weak learner hypotheses (e.g. a majority vote). Combining this reproducible boosting algorithm with our reproducible halfspace weak learner from Section 4.6 yields a reproducible strong learner for halfspaces.

Specifically, we modify the smooth boosting algorithm described in Servedio [2003a] in the batch setting, presenting it in the filtering setting Bradley and Schapire [2007]. This boosting algorithm has three main components, all of which can be made reproducible: (i) checking for termination (via a statistical query), (ii) running the weak learner (reproducible by assumption), and (iii) updating the weighting function (deterministic). The final classifier is a sum of returned weak learner hypotheses. With high probability over two runs, our boosting algorithm `rBoost` collects the exact same hypotheses $h_1, \ldots, h_T$ from its reproducible weak learner.

4.7.1 Reproducible Boosting Algorithm: An Overview

In smooth boosting algorithms, a "measure" function $\mu : \mathscr{X} \to [0,1]$ determines a reweighting of distribution D. The induced reweighted distribution, denoted D_μ, is defined by the probability density function $D_\mu(x) = \mu(x) \cdot D(x)/d(\mu)$, where $d(\mu)$ is a normalizing factor $\mathbb{E}_{x \sim D}\mu(x)$. We refer to $d(\mu)$ as the *density* of measure μ. A sample **s** is drawn from D_μ and passed to the weak learner `rWkL`. Sampling from D_μ using example oracle EX is done by rejection sampling — draw a sample (x,y) from EX and a random $b \in_r [0,1]$; if $r \leq \mu(x)$, keep (x,y); otherwise, reject x and loop until we keep (x,y). On expectation, we require $m/d(\mu)$ examples from D to sample m examples from D_μ.

At the beginning of the algorithm, $\mu(x) = 1$ for all $x \in \text{supp}(D)$. Weak learner hypotheses h_t are used to modify update μ (and thus D_μ) for future weak learner queries. The algorithm terminates when the density $d(\mu)$ drops below the desired accuracy parameter ε — at this point, the majority vote hypothesis $\mathbf{h} = \text{sign}(\sum_t h_t)$ has accuracy at least $1 - \varepsilon$ over D.

More specifically, we define $\mu_{t+1}(x) = M(g_t(x))$ using a base measure function $M : \mathbb{R} \to [0,1]$ and score function $g : \mathscr{X} \to \mathbb{R}$. As in Servedio [2003a], we use a capped exponential function as our base measure function $M(a) = \begin{cases} 1 & a \leq 0 \\ (1-\gamma)^{a/2} & a > 0 \end{cases}$. The score function is $g_t(x) = \sum_{i=1}^{t}(h_i(x)f(x) - \theta)$, where $\theta < \gamma$ is chosen as a function of γ.

Algorithm 21. $\texttt{rBoost}^{\texttt{rWkL}}(\mathbf{s}; r)$

Parameters: desired reproducibility ρ, accuracy ε, constant $\theta \stackrel{\text{def}}{=} \gamma/(2+\gamma)$, round complexity $T = O(1/\varepsilon\gamma^2)$

Input: Sample $\mathbf{s}$ of m examples $(\mathbf{x}_i, y_i)$ drawn i.i.d. from distribution D.

Weak learner rWkL with advantage γ and sample complexity $m_{\texttt{rWkL}}$.

Output: $\mathbf{h} = \text{sign}\left(\sum_{t=1}^{T} h_t\right)$.

```
$g_0(x) \stackrel{\text{def}}{=} 0$
$\mu_1(x) \stackrel{\text{def}}{=} M(g_0) = 1$   /* "Measure" function for reweighting */
$t \leftarrow 0$
**while** 1 **do**
    $t \leftarrow t+1$
    $D_{\mu_t}(x) \stackrel{\text{def}}{=} \mu_t(x) \cdot D(x)/d(\mu_t)$   /* Reweighted distribution */
    $\mathbf{s}_1 \leftarrow \widetilde{O}(m_{\texttt{rWkL}}/\varepsilon)$ fresh examples from $\mathbf{s}$
    $\mathbf{s}_{\texttt{rWkL}} \leftarrow \texttt{RejectionSampler}(\mathbf{s}_1, m_{\texttt{rWkL}}, \mu_t; r_1)$   /* Rejection sampling for rWkL */
    Hypothesis $h_t \leftarrow \texttt{rWkL}(\mathbf{s}_{\texttt{rWkL}}; r_2)$
    $g_t(x) \stackrel{\text{def}}{=} g_{t-1}(x) + h_t(x)f(x) - \theta$   /* Reweight distribution using $h_t$ */
    $\mu_{t+1}(x) \stackrel{\text{def}}{=} M(g_t(x))$
    $\mathbf{s}_2 \leftarrow \widetilde{O}\left(\frac{1}{\rho^2\varepsilon^3\gamma^2}\right)$ fresh examples from $\mathbf{s}$   /* Run rSTAT to check if $d(\mu_{t+1}) \leq \varepsilon$ */
    **if** $\texttt{rSTAT}_{\tau,\rho_0,\phi}(\mathbf{s}_2; r_3) \leq 2\varepsilon/3$ **then**   /* tolerance $\tau = \varepsilon/3$, reproducibility $\rho_0 = \rho/(3T)$ */
        Exit while loop   /* failure rate $\rho/(12T)$, query $\phi(x,y) = \mu(x)$ */
**return** $\mathbf{h} \leftarrow \text{sign}\left(\sum_t h_t\right)$
```

Algorithm 22. `RejectionSampler`$(\mathbf{s}_{\text{all}}, m_{\text{target}}, \mu; r)$
Input: sample $\mathbf{s}_{\text{all}}$ drawn i.i.d. from distribution D, target size of output sample $m_{\text{target}} \in [|\mathbf{s}_{\text{all}}|]$, and description of measure function $\mu : \mathscr{X} \to [0,1]$.
Output: $\perp$ or a sample $\mathbf{s}_{\text{kept}}$ of size $|\mathbf{s}_{\text{kept}}| = m_{\text{target}}$

```
$\mathbf{s}_{\text{kept}} \leftarrow \emptyset$
**for** $i = 1$ to $i = |\mathbf{s}_{\text{all}}|$ **do**
  Use randomness $r$ to randomly pick a $b \in [0,1]$
  **if** $\mu(x_i) \geq b$ **then**  /* Reject $(x_i, y_i)$ w. p. $1 - \mu(x)$ */
    $\mathbf{s}_{\text{kept}} \leftarrow \mathbf{s}_{\text{kept}} || (x_i, y_i)$   /* Add example $(x_i, y_i)$ to $\mathbf{s}_{\text{kept}}$ */
  **if** $|\mathbf{s}_{\text{kept}}| = m_{\text{target}}$ **then**
    **return** $\mathbf{s}_{\text{kept}}$
**return** $\perp$  /* Ran out of fresh samples in $\mathbf{s}_{\text{all}}$ */
```

A subtle note is that this boosting algorithm must precisely manage its sample $\mathbf{s}$ and random string r when invoking subroutines. In order to utilize the reproducibility of subroutines (e.g. `rWkL`), the boosting algorithm needs to ensure that it uses random bits from the same position in r. A first-come first-serve approach to managing r (i.e. each subroutine uses only the amount of randomness it needs) fails immediately for `rBoost` — the amount of randomness `RejectionSampler` needs is dependent on the sample, so the next subroutine (in this case, `rWkL`) may not be using the same randomness across two (same-randomness r) runs of `rBoost`.

If one can precisely upper bound the amount of randomness needed for each of L subroutines, then r can be split into chunks $r_1||r_2||\ldots||r_L$, avoiding any desynchronization issues. Alternatively, one can split r into L equally long random strings by only using bits in positions equivalent to l mod L for subroutine $l \in [L]$.

4.7.2 Analysis of `rBoost` (Algorithm 21)

As before, function f in concept class C is a function from domain $\mathscr{X}$ to $\{\pm 1\}$. D is a distribution over $\mathscr{X}$.

Theorem 4.7.1 (Reproducible Boosting)**.** *Let $\varepsilon > 0, \rho > 0$. Let* `rWkL` *be a $(\rho_r, \gamma, \delta_{\texttt{rWkL}})$-weak learner. Then* $\texttt{rBoost}^{\texttt{rWkL}}(\mathbf{s}; r)$ *is ρ-reproducible and with probability at least $1-\rho$, outputs a hypothesis $\mathbf{h}$ such that $\mathbf{Pr}_{x\sim D}[\mathbf{h}(x) = f(x)] \geq 1-\varepsilon$.* `rBoost` *runs for $T = O(1/(\varepsilon\gamma^2_{\texttt{rWkL}}))$ rounds and uses $\widetilde{O}\left(\frac{m_{\texttt{rWkL}(\rho/(6T))}}{\varepsilon^2\gamma^2} + \frac{1}{\rho^2\varepsilon^3\gamma^2}\right)$ samples, where the $\widetilde{O}$ notation hides $\log(1/(\rho\varepsilon\gamma^2))$ factors and $m_{\texttt{rWkL}(\rho/(6T))}$ denotes the sample complexity of* `rWkL` *with reproducibility parameter $\rho/(6T)$.*

For readability, we break the proof into components for round complexity, correctness, reproducibility, sample complexity, and failure probability.

Proof. **Round Complexity:** Theorem 3 in Servedio [2003a] gives a $T = O(1/(\varepsilon\gamma^2_{\texttt{rWkL}}))$ round complexity bound for this boosting algorithm in the batch setting. Analogous arguments hold in the filtering setting, so we defer to Servedio [2003a] for brevity.

Correctness: Similarly, since reproducible weak learner `rWkL` satisfies the definitions of a weak learner, the correctness arguments in Servedio [2003a] also hold. A small difference is the termination condition — rather than terminate when the measure satisfies $d(\mu) < \varepsilon$, our algorithm terminates when the density estimated by `rSTAT` is less than $2\varepsilon/3$. We run `rSTAT` on query $\phi(x) = \mu(x)$ with tolerance parameter $\varepsilon/3$. Thus, when the `rBoost` terminates, $d(\mu_t) < \varepsilon$.

Reproducibility: We show this boosting algorithm not only reproducibly outputs the same hypothesis $\mathbf{h}$, but that each returned weak learner hypothesis h_t is identical across two runs of the boosting algorithms (using the same randomness r) with high probability. The reweighted distribution D_{μ_t} depends only on the previous weak learner hypotheses $h_1, \ldots, h_{t-1}$, so the only possibilities for loss of reproducibility are: (i) returning $\perp$ while rejection sampling from D_μ; (ii) running the reproducible weak learner; and (iii) using a statistical query to decide to exit the while loop. We note that our choice of parameters adds non-reproducibilty at most $\rho/(3T)$ for each and apply a union bound over at most T rounds of boosting.

1. By Lemma 4.7.2, $O(\frac{m_{\texttt{rWkL}}}{\varepsilon} \cdot \log(T/\rho))$ examples suffice to guarantee `RejectionSampler` outputs $\perp$ with probability at most $\rho/(6T)$. Union bounding over two runs, this is at most $\rho/(3T)$.

2. By Lemma 4.7.4, running rWkL with reproducibility parameter $\rho/(6T)$ will add a $\rho/(3T)$ contribution to the non-reproducibility.

3. We run rSTAT with reproducibility parameter $\rho/(3T)$.

Sample Complexity: There are two contributions to the sample complexity: samples used for the weak learner rWkL, and samples used by rSTAT to estimate the density of measure μ_t. Fresh samples are used for each of T rounds of boosting. Together, by Theorem 4.3.3 and the definition of $\mathbf{s}_1$ (in Algorithm 21), the sample complexity is

$$O\left(T \cdot \left(\frac{m_{\texttt{rWkL}(\rho/(6T))}}{\varepsilon} \cdot \log(T/\rho) + \frac{\log(T/\rho)}{(\varepsilon^2)(\rho)^2}\right)\right) = \widetilde{O}\left(\frac{m_{\texttt{rWkL}(\rho/(6T))}}{\varepsilon^2\gamma^2} + \frac{1}{\rho^2\varepsilon^3\gamma^2}\right)$$

where the $\widetilde{O}$ notation hides $\log(1/(\rho\varepsilon\gamma^2))$ factors and $m_{\texttt{rWkL}(\rho/(6T))}$ denotes the sample complexity of rWkL with reproducibility parameter $\rho\varepsilon\gamma^2$.

Failure Probability: Assuming the weak learner returns correct hypotheses when it is reproducible, the boosting algorithm rBoost is correct when it is reproducible, so the failure probability is bounded above by ρ.[4] $\square$

4.7.3 Rejection Sampling Lemmas

Next, we show that reproducibility composes well with rejection sampling throughout the execution of rBoost.

Lemma 4.7.2 (Failure Rate of RejectionSampler)**.** *Let measure μ have density $d(\mu) \geq \varepsilon/3$. Let $\mathbf{s}_{all}$ be a sample drawn i.i.d. from distribution D. If $|\mathbf{s}_{all}| \geq \frac{24 m_{target}}{\varepsilon} \cdot \log(1/\delta)$, then* RejectionSampler$(\mathbf{s}_{all}, m_{target}, \mu$ *outputs $\perp$ with probability at most δ.*

Proof. The probability RejectionSampler outputs $\perp$ is precisely the probability a binomial random variable $X \sim B(|\mathbf{s}_{\text{all}}|, d(\mu))$ is at most m_{target}. By a Chernoff bound, $\mathbf{Pr}[X \leq (1-.5)|\mathbf{s}_{\text{all}}| \cdot d(\mu)] \leq \exp(-|\mathbf{s}_{\text{all}}| \cdot d(\mu)/8) \leq \exp(-|\mathbf{s}_{\text{all}}| \cdot \varepsilon/24)$. Thus, $\mathbf{Pr}[X \leq m_{\text{target}}] \leq \delta$. $\square$

[4]A more precise sample complexity statement in terms of the failure probability δ can be obtained by unboxing the error probabilities. The algorithm can fail if RejectionSampler outputs $\perp$, if rWkL fails, and if rSTAT fails. Bounding each of these quantities by $\delta/(3T)$ ensures that the rBoost has failure rate δ.

Remark 4.7.3. *The following is a justification of why we may assume $d(\mu) \geq \varepsilon/3$ in the previous lemma.*

When `RejectionSampler` *is first called in round* 1 *of* `rBoost`*,* $\mu(x) = 1$ *for all* x*, so* $d(\mu) = 1$*. In subsequent rounds* $t \geq 2$*,* `RejectionSampler` *is only called if, in previous round* $t-1$*,* `rSTAT` *estimated* $d(\mu)$ *to be at least* $2\varepsilon/3$*.* `rSTAT` *is run with tolerance* $\varepsilon/3$*, so* $d(\mu) \geq \varepsilon/3$ *whenever* `rSTAT` *succeeds. Whenever we apply the above lemma, we are assuming the success of previous subroutines (by keeping track of and union bounding over their error).*

The following lemma shows that rejection sampling before running a reproducible algorithm only increases the non-reproducibility ρ by a factor of 2. To be precise, we let p denote the probability the rejection sampler returns $\perp$. However, when we apply this Lemma in the proof of Theorem 4.7.1, we will have already accounted for this probability.

Lemma 4.7.4 (Composing Reproducible Algorithms with Rejection Sampling)**.** *Let* $\mathscr{A}(\mathbf{s}, r)$ *be a* ρ*-reproducible algorithm with sample complexity* m *Let* $\mu : \mathscr{X} \to [0,1]$*. Consider* $\mathscr{B}$*, the algorithm defined by composing* `RejectionSampler`$(\mathbf{s}', m, \mu; r')$ *with* $\mathscr{A}(\mathbf{s}; r)$*. Let* q *be the probability that* `RejectionSampler` *returns* $\perp$*. Then* $\mathscr{B}$ *is a* $2q + 2\rho$*-reproducible algorithm.*

Proof. Since $\mathscr{A}$ is ρ-reproducible, $\mathbf{Pr}_{\mathbf{s}_1,\mathbf{s}_2,r}[\mathscr{A}(\mathbf{s}_1; r) = \mathscr{A}(\mathbf{s}_2; r)] \geq 1 - \rho$. However, the rejection sampling is done with correlated randomness, so $\mathbf{s}_1$ and $\mathbf{s}_2$ are not independent. Consider an imaginary third run of algorithm $\mathscr{A}(\mathbf{s}_3; r)$, where $\mathbf{s}_3$ is drawn from D_μ using separate randomness. We will use a triangle-inequality-style argument (and a union bound) to derive the conclusion. Conditioned on `RejectionSampler` not returning $\perp$, algorithm $\mathscr{B}(\mathbf{s}'_1; r'||r)$ returns the same result as $\mathscr{A}(\mathbf{s}_3, r)$ (when both algorithms use randomness r for the execution of $\mathscr{A}$) with probability at least $1 - \rho$. The same statement holds for the second run $\mathscr{B}(\mathbf{s}'_1; r'||r)$. Thus,

$$\mathbf{Pr}_{\mathbf{s}'_1,\mathbf{s}'_2,r'||r}\left[\mathscr{B}(\mathbf{s}'_1; r'||r) = \mathscr{B}(\mathbf{s}'_2; r'||r) | \text{ neither run outputs } \perp\right] \geq 1 - 2\rho.$$

Finally, $\mathscr{B}$ may fail to be reproducible if either `RejectionSampler` call returns $\perp$, so we union bound over this additional $2q$ probability. $\square$

4.7.4 Reproducible Strong Halfspace Learner

We give two reproducible strong learners for halfspaces by combining boosting algorithm `rBoost` with reproducible weak halfspace learners `rHalfspaceWkL` and $\texttt{rHalfspaceWkL}^{\text{box}}$.

Corollary 4.7.5. *Let D be a distribution over $\mathbb{R}^d$, and let $f : \mathbb{R}^d \to \{\pm 1\}$ be a halfspace with margin τ in D. Let $\varepsilon > 0$. Then*

- *Algorithm* `rBoost` *run with weak learner* `rHalfspaceWkL` *ρ-reproducibly returns a hypothesis $\mathbf{h}$ such that, with probability at least $1-\rho$, $\mathbf{Pr}_{\mathbf{x}\sim D}[\mathbf{h}(\mathbf{x}) = f(\mathbf{x})] \geq 1-\varepsilon$, using a sample of size $\widetilde{O}\left(\frac{d^{10/9}}{\tau^{76/9}\rho^{20/9}\varepsilon^{28/9}}\right)$.*
- *Algorithm* `rBoost` *run with weak learner* $\texttt{rHalfspaceWkL}^{box}$ *ρ-reproducibly returns a hypothesis $\mathbf{h}$ such that, with probability at least $1-\rho$, $\mathbf{Pr}_{\mathbf{x}\sim D}[\mathbf{h}(\mathbf{x}) = f(\mathbf{x})] \geq 1-\varepsilon$, using a sample of size $\widetilde{O}\left(\frac{d^{15/4}}{\tau^{10}\rho^{5/2}\varepsilon^{9/2}}\right)$.*

Proof. For the first strong learner, we compose Theorem 4.7.1 with Theorem 4.6.2. `rHalfspaceWkL` has advantage $\gamma = \tau/4$, so `rBoost` has round complexity $T = O(1/(\varepsilon\gamma^2)) = O(1/(\varepsilon\tau^2))$. `rBoost` runs `rHalfspaceWkL` with parameter $\rho_{\texttt{rWkL}} = \rho/6T$, so the sample complexity $m_{\texttt{rHalfspaceWkL}}$ is $O\left(\frac{d^{10/9}}{\tau^{58/9}\rho^{20/9}\varepsilon^{10/9}}\right)$. Thus, the sample complexity for the boosting algorithm is

$$\widetilde{O}\left(\frac{d^{10/9}}{\tau^{76/9}\rho^{20/9}\varepsilon^{28/9}}\right)$$

For the second strong learner, we compose Theorem 4.7.1 with Theorem 4.6.13. As before, `rBoost` has round complexity $T = O(1/(\varepsilon\tau^2))$ and runs $\texttt{rHalfspaceWkL}^{\text{box}}$ with parameter $\rho_{\texttt{rWkL}} = \rho/6T$. The sample complexity $m_{\texttt{rHalfspaceWkL}^{\text{box}}}$ is $O\left(\left(\frac{d^{3/2}}{\tau^2\rho_{\texttt{rWkL}}}\right)^{5/2}\right) = \left(\left(\frac{d^{3/2}}{\tau^4\rho\varepsilon}\right)^{5/2}\right)$. Thus, the sample complexity for the boosting algorithm is $\widetilde{O}\left(\frac{m_{\texttt{rWkL}(\rho/(6T))}}{\varepsilon^2\gamma^2} + \frac{1}{\rho^2\varepsilon^3\gamma^2}\right) = \widetilde{O}\left(\frac{1}{\varepsilon^2\gamma^2}\left(\frac{d^{3/2}}{\tau^2\rho_{\texttt{rWkL}}}\right)^{5/2}\right) = \widetilde{O}\left(\frac{d^{15/4}}{\tau^{10}\rho^{5/2}\varepsilon^{9/2}}\right)$. □

Remark 4.7.6. $\texttt{rHalfspaceWkL}^{box}$ *can be run in time polynomial in the input parameters, so the strong learner obtained by running* `rBoost` *with weak learner* $\texttt{rHalfspaceWkL}^{box}$ *is a* $\mathrm{poly}(1/\varepsilon, 1/\rho, 1/\tau, d)$-

time algorithm. However, the other weak learner `rHalfspaceWkL` *uses a foams construction subroutine from Kindler et al. [2012] that takes expected exponential in d runtime. The corresponding strong learner runs in time polynomial in $1/\varepsilon, 1/\rho$, and $1/\tau$, but exponential in d.*

4.7.5 Discussion

Algorithm 21 follows the smooth boosting framework of Servedio Servedio [2003a], which also shows how to boost a weak halfspace learner under a margin assumption on the data. They show that their boosted halfspace learner obtains a hypothesis with good margin on the training data, and then apply a fat-shattering dimension argument to show generalization to the underlying distribution with sample complexity $\tilde{O}(1/(\tau\varepsilon)^2)$. Notably, this gives sample complexity independent of d. Moreover, their smooth boosting algorithm is tolerant to malicious noise perturbing an $\eta \in O(\tau\varepsilon)$ fraction of its sample.

A generic framework for differentially private boosting was given in Bun et al. [2020a], with an application to boosting halfspaces. Their boosting algorithm also follows the smooth boosting framework, but uses a variant of the round-optimal boosting algorithm given in Barak et al. [2009b]. Their halfspace learner similarly requires a margin assumption on the data and tolerates random classification noise at a rate $\eta \in O(\tau\varepsilon)$. They give two generalization arguments for their halfspace learner, both of which are dimension-independent. The first follows from prior work showing that differential privacy implies generalization Bassily et al. [2016] and gives sample complexity $\tilde{O}(\frac{1}{\varepsilon\alpha\tau^2} + \frac{1}{\varepsilon^2\tau^2} + \alpha^{-2} + \varepsilon^{-2})$ for approximate differential privacy parameters (α, β). The second follows from a fat-shattering dimension argument and gives a tighter bound of $\widetilde{O}\left(\frac{1}{\varepsilon\alpha\tau^2}\right)$.

Boosting algorithms have been thoroughly studied over the past few decades, and there are many types of boosting algorithms (e.g. distribution-reweighting, branching-program, gradient boosting) with different properties (e.g. noise-tolerance, parallelizability, smoothness, batch vs. filtering). It would be interesting to see which of these techniques can be made reproducible, and at what cost.

4.8 Reproducibility: Alternative Definitions and Properties

In this section, we consider a few alternative criteria for reproducibility and show how they relate to our definition of reproducibility. We also demonstrate other robustness properties of reproducibility such as amplifying the parameters, as well as data/randomness reuse.

Alternative Definitions and Amplification. Throughout most of this chapter, we choose to define $\mathscr{A}$ as have two sources of random inputs: samples $\mathbf{s}$ drawn from distribution D and internal randomness r. $\mathscr{A}$ has no additional inputs. However, we could more generally define $\mathscr{A}$ to have additional, nonrandom inputs. In this more general definition, we define $\mathscr{A}(x;\mathbf{s};r)$ where $\mathbf{s}$ and r are as defined previously, and x is an auxiliary input (or tuple of inputs). $\mathscr{A}(x;\mathbf{s};r)$ is ρ-reproducible with respect to distribution D if for every input x, $\mathscr{A}(x;\mathbf{s};r)$ is ρ-reproducible. This definition generalizes both pseudodeterministic algorithms (in which there is no underlying distribution, so $\mathbf{s}$ is empty) as well as our definition of reproducible learning algorithms (in which there are no additional inputs, so x is empty).

Rather than parameterize reproducibility by a single parameter ρ, one could use two variables (η,ν).

Definition 4.8.1 ((η,ν)-reproducibility)**.** Let $\mathscr{A}(x;\mathbf{s};r)$ be an algorithm, where $\mathbf{s}$ are samples from D, and r is the internal randomness. We say that a particular random string r is η-good for $\mathscr{A}$ on x with respect to D if there is a single "canonical" output Z_r such that $\mathbf{Pr}[\mathscr{A}(x;\mathbf{s};r)=Z_r] \geq 1-\eta$. Then $\mathscr{A}$ is (η,ν)-reproducible with respect to D if, for each x, the probability that a random r is η-good for $\mathscr{A}$ (on x and D) is at least $1-\nu$.

(η,ν)-reproducibility is qualitatively the same as ρ-reproducibility, but might differ by polynomial factors. If $\mathscr{A}$ is (η,ν)-reproducible, then $\mathscr{A}$ is ρ-reproducible on D, where $\rho \leq 2\eta+\nu$. The probability that two runs of $\mathscr{A}$, using the same internal randomness r, output different results is at most $\mathbf{Pr}[r \text{ not } \eta\text{-good}]$ plus the probability that at least one run is not the special output Z_r (conditioned on r being η-good). In the other direction, if $\mathscr{A}$ is ρ-reproducible, then $\mathscr{A}$ is $(\rho/\nu,\nu)$-

reproducible for any $\rho \le \nu < 1$. Say there is a ν probability that r is not η-good. Conditioned on picking a not η-good r, there is a conditional (at least) η probability of the second run of $\mathscr{A}$ returning something different than the first run.[5] Thus, $\rho \ge \eta\nu$.

A similar definition called "pseudo-global stability", developed independently to our work, appears in Ghazi et al. [2021]. That definintion parametrizes by the sample complexity m and does not explicitly parametrize by auxiliary input x. Additionally, their definition includes an (α, β)-accuracy guarantee on Z_r, the very likely output. To keep both definitions consistent with their original conventions, we write $\eta' \overset{\text{def}}{=} 1 - \eta$ and $\nu' \overset{\text{def}}{=} 1 - \nu$.

Definition 4.8.2 (Pseudo-global stability, Definition 15 in Ghazi et al. [2021])**.** A learning algorithm $\mathscr{A}$ with sample complexity m is said to be (α, β)-accurate, (η', ν')-pseudo-globally stable if there exists a hypothesis h_r for every $r \in \operatorname{supp}(R)$ (depending on D) such that $\mathbf{Pr}_{r\sim R}[\mathrm{err}_D(h_r) \le \alpha] \ge 1 - \beta$ and

$$\mathbf{Pr}_{r\sim R}\left[\mathbf{Pr}_{\mathbf{s}\sim D^m}[\mathscr{A}(\mathbf{s}; r) = h_r] \ge \eta'\right] \ge \nu'$$

where $\mathbf{s}$ is a sample of m (labeled) examples (x_i, y_i) drawn from distribution D.

The final condition of Definition 4.8.2 is equivalent to saying that i) a randomly chosen r is η'-good with probability at least ν', and ii) for every r, h_r is the output that witnesses the η-goodness. Carrying the accuracy guarantee through the previous argument, an (α, β)-accurate (η', ν')-pseudo-globally-stable algorithm $\mathscr{A}$ implies a $(2(1-\eta') + (1-\nu')) = (2\eta + \nu)$-reproducible algorithm $\mathscr{A}$ also with (α, β)-accuracy.

If we are willing to increase the sample complexity of $\mathscr{A}$, we can make the connection stronger:

Theorem 4.8.3 (Amplification of Reproducibility)**.** *Let* $0 < \eta, \nu, \beta < 1/2$ *and* $m > 0$. *Let* $\mathscr{A}$ *be an* (η, ν)*-reproducible algorithm for distribution D with sample complexity m and failure rate* β. *If* $\rho > 0$ *and* $\nu + \rho < 3/4$, *then there is a* ρ*-reproducible algorithm* $\mathscr{A}'$ *for D with sample complexity*

[5]If $1 - \eta \ge 1/2$, then the probability that two runs of $\mathscr{A}$ using the same randomness returns the same result is at most $(1-\eta)^2 + \eta^2$, i.e., when $\mathscr{A}$ has only two possible outputs. This is less than $1 - \eta$, assuming $1 - \eta \ge 1/2$. If $1 - \eta < 1/2$, then there must be more than two outputs, and the probability of nonreproducibility is again larger than η.

$m' = \widetilde{O}(m(\log 1/\beta)^3/(\rho^2(1/2-\eta)^2)$ *and failure rate at most* $O(\beta+\rho)$. *The construction of* $\mathscr{A}'$ *does not depend on D.*

Proof. Set $k = 3\log 1/\beta$. For each random string r, let D_r be the distribution on outputs of $\mathscr{A}(x;\mathbf{s};r)$ (over random $\mathbf{s}$). Algorithm $\mathscr{A}'$ randomly picks k-many strings $r_1,\dots,r_k$, runs the reproducible heavy-hitters algorithm (Algorithm 15) on the distributions $D_{r_1},\dots,D_{r_k}$, and outputs the first returned heavy-hitter (or $\perp$ if each subroutine returns the empty list). We say there are k *rounds* of $\mathscr{A}'$, one per random string r.

The reproducibility of `rHeavyHitters` implies the reproducibility of $\mathscr{A}'$. We show that a heavy-hitter in D_r for randomly chosen r is often a correct answer, except with probability comparable to β.

By definition, r is η-good iff D_r has a $1-\eta$ heavy-hitter. Since $\eta < 1/2$, this heavy-hitter will be unique, and there will be no other $1-\eta > 1/2$ heavy-hitters. Given any r, we can draw from distribution D_r by running algorithm $\mathscr{A}$ with fresh samples $\mathbf{s}$. Consider running the reproducible heavy-hitters algorithm with parameters $v = (3/2-\eta)/2$, $\varepsilon = (1/2-\eta)/2$, and reproducibility $\rho' = \rho/k$. These are chosen so that $v+\varepsilon = 1-\eta$ and $v-\varepsilon = 1/2$. If r is η-good, then `rHeavyHitters` will return the (unique) majority element for D_r with probability at least $1-\rho/k$. If r is not 1/2-good[6], the reproducible heavy-hitters algorithm with the same parameters will return the empty list with probability at least $1-\rho/k$.

Next, we compute the conditional probability that the first element returned by `rHeavyHitters` is correct. The probability that `rHeavyHitters` produces an empty list in one round is at most v (when the randomly chosen r is not η-good) plus ρ/k (when r is η-good but the heavy-hitters algorithm fails). At most a (2β)-fraction of random strings r satisfy both of the following two conditions: i) D_r has a majority element Z_r and ii) Z_r is an incorrect output. Thus, the conditional probability of outputting an incorrect answer, given `rHeavyHitters` produces a non-empty output, is at most $(2\beta+\rho/k)/(1-v-\rho/k)$. By assumption, $v+\rho/k < 3/4$, so this is $O(\beta+\rho)$.

So far, we have bounded the probability that $\mathscr{A}'$ returns an incorrect answer. $\mathscr{A}'$ could also fail

[6]Since $\eta < 1/2$ by assumption, r being not 1/2-good implies r is not η-good.

if `rHeavyHitters` returns the empty list in each of k rounds. Since $v + \rho/k < 3/4$, this happens with probability at most $(3/4)^k \leq \beta$. So, the overall probability of error is at most $O(\beta + \rho)$.

If two runs of $\mathscr{A}'$ use the same r_i's and same randomness for each heavy-hitters call, they only produce different answers if a pair of `rHeavyHitters` calls produces different answers in the same round. By the reproducibility of `rHeavyHitters`, this only happens with probability ρ/k each round, for a total non-reproducibility probability at most ρ.

$\mathscr{A}'$ calls `rHeavyHitters` $k = O(\log 1/\beta)$ times. Each example used by `rHeavyHitters` is created by running $\mathscr{A}$, which has sample complexity m. By Lemma 4.4.3, $\texttt{rHeavyHitters}_{\rho',v,\varepsilon}$ has sample complexity $\widetilde{O}\left(\frac{1}{\rho'^2\varepsilon^2(v-\varepsilon)^2}\right)$. Substituting in $\rho' = \rho/k, \varepsilon = (1/2-\eta)/2$, and $v - \varepsilon = 1/2$, $\mathscr{A}$ has sample complexity $km \cdot \widetilde{O}\left(\frac{k^2}{\rho^2(1/2-\eta)^2}\right) = \widetilde{O}\left(\frac{m\log^3(1/\beta)}{\rho^2(1/2-\eta)^2}\right)$. □

Corollary 4.8.4. *Let $\alpha > 0$ and $\rho < 1/4 - \alpha$. Let $\mathscr{A}$ be a ρ-reproducible algorithm using m samples is correct except with error at most β. Then for arbitrary ρ' satisfying $\rho > \rho' > 0$, there is a ρ'-reproducible algorithm $\mathscr{A}'$ with sample complexity $m' = \widetilde{O}\left(\frac{m\log^3(1/\beta)}{\rho'^2\alpha^2}\right)$ that is correct except with error at most $O(\beta + \rho')$.*

Proof. By the arguments immediately after Definition 4.8.1, a ρ-reproducible algorithm implies a $(\rho/x, x)$-reproducible algorithm. Choosing $x = 1/2 - \alpha$ allows us to apply Theorem 4.8.3 for any $\rho < 1/4$. The $(1/2 - \eta)$ term in Theorem 4.8.3 simplifies to $\alpha/(1-2\alpha)$ in this context. When α can be chosen as a constant, the sample complexity simplifies to $m' = \widetilde{O}(m\log^3(1/\beta)/\rho'^2)$. □

Public versus Private Randomness. We define reproducibility as the probability that when run twice using the same (public) randomness, but with independently chosen data samples, the algorithm returns the same answer. In Grossman and Liu [2019], the authors define a related concept, but divide up the randomness into two parts, where only the first randomness part gets reused in the second run of the algorithm. In their applications, there are no data samples, so re-running the algorithm using identical randomness would always give identical results; rather, they were trying to minimize the amount of information about the random choices that would guarantee reproducibility, i.e., minimize the length of the first part.

Similarly, we could define a model of reproducibility that involved two kinds of random choices. Define $\mathscr{A}(x;\mathbf{s};r_{\text{p}ub},r_{\text{p}riv})$, $\mathbf{s}=(s_1,\ldots,s_m)$ to be ρ-reproducible with respect to r_{pub} and D if for every x, random $\mathbf{s_1}$ and $\mathbf{s_2}$ drawn from D^m, and random $r_{\text{p}ub},r_{\text{p}riv},r'_{\text{p}riv}$,

$$\mathbf{Pr}[\mathscr{A}(x;\mathbf{s_1};r_{\text{pub}},r_{\text{priv}})=\mathscr{A}(x;\mathbf{s_2};r_{\text{pub}},r'_{\text{priv}})]\geq 1-\rho.$$

If we want to minimize the amount of information we need to store to guarantee reproducibility, keeping $r_{\text{p}riv}$ and $r_{\text{p}ub}$ distinct may be important. However, if all we want is to have a maximally reproducible algorithm, the following observation shows that it is always better to make the entire randomness public.

Lemma 4.8.5. *If $\mathscr{A}(x;\mathbf{s};r_{pub},r_{priv})$ is ρ-reproducible with respect to r_{pub} over D, then $\mathscr{A}(x;\mathbf{s};r_{pub},r_{priv})$ is ρ-reproducible with respect to (r_{pub},r_{priv}) over D.*

Proof. We show for each value of x and r_{pub},

$$\mathbf{Pr}[\mathscr{A}(x;\mathbf{s_1};r_{\text{p}ub},r_{\text{priv}})=\mathscr{A}(x;\mathbf{s_2};r_{\text{pub}},r'_{\text{priv}})]\leq \mathbf{Pr}[\mathscr{A}(x;\mathbf{s_1};r_{\text{pub}},r_{\text{priv}})=\mathscr{A}(x;\mathbf{s_2};r_{\text{p}ub},r_{\text{p}riv})].$$

Fix x and r_{pub}. For each possible value R of $r_{\text{p}riv}$ and each possible output Z, let $q_{R,Z}=\mathbf{Pr}[\mathscr{A}[(x;\mathbf{s_1};r_{\text{pub}},R)]=Z$, and let $\mathbf{q}_R$ be the vector indexed by Z whose Z^{th} coordinate is $q_{R,Z}$. Then

$$\mathbf{Pr}[\mathscr{A}[(x;\mathbf{s_1};r_{\text{pub}},R)=\mathscr{A}(x;\mathbf{s_2};r_{\text{pub}},R)]=\sum_Z(q_{R,Z})^2=||\mathbf{q}_R||_2^2,$$

and

$$\mathbf{Pr}[\mathscr{A}(x;\mathbf{s_1};r_{\text{p}ub},R)=\mathscr{A}(x;\mathbf{s_2};r_{\text{pub}},R')]=\sum_Z(q_{R,Z}q_{R',Z})=\langle\mathbf{q}_R,\mathbf{q}_{R'}\rangle.$$

Thus,

$$\begin{aligned}\mathbf{Pr}[\mathscr{A}(x;\mathbf{s_1};r_{\text{pub}},r_{\text{priv}}) = \mathscr{A}(x;\mathbf{s_2};r_{\text{pub}},r'_{\text{priv}})] &= \mathbb{E}_{R,R'}[\langle \mathbf{q}_R,\mathbf{q}_{R'}\rangle] \\ &\le \mathbb{E}_{R,R'}[||\mathbf{q}_R||_2||\mathbf{q}_{R'}||_2] \\ &= (\mathbb{E}_R[||\mathbf{q}_R||_2])^2 \\ &\le \mathbb{E}_R[||\mathbf{q}_R||_2^2] \\ &= \mathbf{Pr}[\mathscr{A}(x;\mathbf{s_1};r_{\text{pub}},r_{\text{priv}}) = \mathscr{A}(x;\mathbf{s_2};r_{\text{pub}},r_{\text{priv}})].\end{aligned}$$

□

We will implicitly use this observation in the boosting algorithm section, since it will be convenient to think of the two runs of the boosting algorithm as picking samples each step independently, when using the same random string would create some correlation.

Reproducibility Implies Generalization. We show that a hypothesis generated by a reproducible algorithm has a high probability of having generalization error close to the empirical error. Let h be a hypothesis, c be a target concept, and D be a distribution. The *risk* (generalization error) of $R(h) \stackrel{\text{def}}{=} \mathbf{Pr}_{x\sim D}[h(x) \neq c(x)]$. If $\mathbf{s}$ is a sample drawn i.i.d. from D, then the *empirical risk* $\widehat{R}_{\mathbf{s}}(h) \stackrel{\text{def}}{=} \mathbf{Pr}_{x\in\mathbf{s}}[h(x) \neq c(x)]$.

Lemma 4.8.6 (Reproducibility Implies Generalization)**.** *Let sample* $\mathbf{s} \sim D^n$*, and let* $\delta > 0$*. Let* h *be a hypothesis output by* ρ*-reproducible learning algorithm* $\mathscr{A}(\mathbf{s};r)$*, where* r *is a random string. Then, with probability at least* $1-\rho-\delta$ *over the choice of* $\mathbf{s}$ *and* r*,* $R(h) \le \widehat{R}_{\mathbf{s}}(h) + \sqrt{\ln(1/\delta)/(2n)}$*.*

Proof. Consider running $\mathscr{A}(\mathbf{s_2};r)$, where $\mathbf{s_2}$ is an independent sample of size m drawn from D, but r is the same as before. Let h_2 denote the returned hypothesis. Since h_2 is independent of $\mathbf{s}$, $\mathbf{Pr}_{\mathbf{s}\sim D^n}[\widehat{R}_{\mathbf{s}}(h_2) - R(h_2) \ge \varepsilon] \le \exp(-2n\varepsilon^2)$ for $\varepsilon > 0$ by Hoeffding's inequality. By the reproducibility of $\mathscr{A}$, $h_2 = h$ with probability at least $1-\rho$. By a union bound, $R(h) \ge \widehat{R}_{\mathbf{s}}(h) + \sqrt{\ln(1/\delta)/2n}$ with probability at least $1-\rho-\delta$. □

In the above argument, we use the definition of reproducibility to create independence between $\mathbf{s}$ and h, allowing us to use Hoeffding's inequality.

Connections to Data Reuse. We consider the adaptive data analysis model discussed in Dwork et al. [2015b] and Dwork et al. [2015a], and we prove that reproducible algorithms are resiliant against adaptive queries (Lemma 4.8.7). The proof is via a hybrid argument.

Lemma 4.8.7 (Reproducibility $\implies$ Data Reusability)**.** *Let D be a distribution over domain $\mathscr{X}$. Let $\mathscr{M}$ be a mechanism that answers queries of the form $q : \mathscr{X} \to \{0,1\}$ by drawing a sample S of n i.i.d. examples from D and returning answer a. Let $\mathscr{A}$ denote an algorithm making m* adaptive *queries, chosen from a set of queries Q, so that the choice of q_i may depend on q_j, a_j for all $j < i$. Denote by $[\mathscr{A}, \mathscr{M}]$ the distribution over transcripts $\{q_1, a_1, \ldots q_m, a_m\}$ of queries and answers induced by $\mathscr{A}$ making queries of $\mathscr{M}$. Let $\mathscr{M}'$ be a mechanism that behaves identically to $\mathscr{M}$, except it draws a single sample S' of n i.i.d. examples from D and answers all queries with S'.*

If $\mathscr{M}$ answers all queries $q \in Q$ with ρ-reproducible procedures, then $SD_\Delta([\mathscr{A}, \mathscr{M}], [\mathscr{A}, \mathscr{M}']) \leq (m-1)\rho$, where $SD_\Delta(D_1, D_2)$ denotes the statistical distance between distribtuions D_1 and D_2.

Proof. For $i \in [m]$, let $[\mathscr{A}, \mathscr{M}_i]$ denote the distribution on transcripts output by algorithm $\mathscr{A}$'s interaction with $\mathscr{M}_i$, where $\mathscr{M}_i$ is the analogous mechanism that draws new samples $S_1, \ldots, S_i$ for the first i queries, and reuses sample S_i for the remaining $m - i$ queries. Note that $\mathscr{M}' = \mathscr{M}_1$ and $\mathscr{M} = \mathscr{M}_m$.

For $i \in [m-1]$, consider distributions $[\mathscr{A}, \mathscr{M}_i]$ and $[\mathscr{A}, \mathscr{M}_{i+1}]$. We will bound the statistical distance by a coupling argument. Let $S_1, \ldots, S_{i+1}$ denote random variables describing the samples used, and let r denote the randomness used over the entire procedure. $[\mathscr{A}, \mathscr{M}_i]$ can be described as running the entire procedure (with randomness R) on $S_1, \ldots, S_{i-1}, S_{i+1}, S_{i+1}, \ldots, S_{i+1}$, and $[\mathscr{A}, \mathscr{M}_{i+1}]$ can be described as running the entire procedure (with randomness R) on $S_1, \ldots, S_{i-1}, S_i, S_{i+1}, S_{i+1}, \ldots, S_{i+1}$.

These distributions are identical for the first $i-1$ queries and answers, so the i'th query q_i is identical, conditioned on using the same randomness. Both S_i and S_{i+1} are chosen by i.i.d. sampling from D, so $\mathbf{Pr}_{S_i, S_{i+1}, r}[\mathscr{A}(q_i, S_{i+1}; r) = A(q_i, S_i; r)] \geq 1 - \rho$ by reproducibility. Conditioned on both transcripts including the same $(i+1)$'th answer a_{i+1} (and continuing to couple S_{i+1} and r for both runs), the remaining queries and answers $q_{i+1}, a_{i+1}, \ldots, q_m, a_m$ is identical. Therefore, $SD_\Delta([\mathscr{A}, \mathscr{M}_i], [\mathscr{A}, \mathscr{M}_{i+1}]) \leq \rho$ for all $i \in [m-1]$. Unraveling, $SD_\Delta([\mathscr{A}, \mathscr{M}], [\mathscr{A}, \mathscr{M}']) \leq (m-1)\rho$. □

Remark 4.8.8. *This connection may be helpful for showing that reproducibility cannot be achieved efficiently in contexts where data reuse is not efficiently achievable.*

4.9 SQ–Reproducibility Lower Bound

How much does it cost to make a nonreproducible algorithm into a reproducible one? In this section, we show a lower bound for reproducible statistical queries via a reduction from the coin problem.

Theorem 4.9.1 (SQ–Reproducibility Lower Bound)**.** *Let $\tau > 0$ and let $\delta \leq 1/16$. Let query $\phi : \mathscr{X} \to [0,1]$ be a statistical query. Let $\mathscr{A}$ be a ρ-reproducible SQ algorithm for ϕ with tolerance less than τ and success probability at least $1-\delta$. Then $\mathscr{A}$ has sample complexity at least $m \in \Omega(1/(\tau^2\rho^2))$.*

Note that this nearly matches the reproducible statistical query upper bound in Theorem 4.3.3, in the case that $\delta \in \Theta(\rho)$.

Recall the coin problem: promised that a 0-1 coin has bias either $1/2 - \tau$ or $1/2 + \tau$ for some fixed $\tau > 0$, how many flips are required to identify the coin's bias with high probability?

Proof of Theorem 4.9.1. A τ-tolerant ρ-reproducible SQ algorithm $\mathscr{A}$ for ϕ naturally induces a ρ-reproducible algorithm $\mathscr{B}$ for the τ-coin problem — $\mathscr{B}$ runs $\mathscr{A}$ (the results of the coin flips are the $\phi(x)$'s), and $\mathscr{B}$ accepts (outputs 1) if $\mathscr{A}$'s output is $\geq 1/2$, otherwise rejects. The success probability of $\mathscr{B}$ is at least that of $\mathscr{A}$. As $\mathscr{A}$ is reproducible for all distributions, $\mathscr{B}$ also satisfies the assumption in Lemma 4.9.2 that $\mathscr{B}$ is ρ-reproducible for coins with bias in $(1/2-\tau, 1/2+\tau)$. By Lemma 4.9.2, any reproducible algorithm solving the coin problem with these parameter has sample complexity $m \in \Omega(1/(\tau^2\rho^2))$, implying the lower bound. □

Lemma 4.9.2 (Sample Lower Bound for the Coin Problem)**.** *Let $\tau < 1/4$ and $\rho < 1/16$. Let $\mathscr{B}$ be a ρ-reproducible algorithm that decides the coin problem with success probability at least $1-\delta$ for $\delta = 1/16$. Furthermore, assume $\mathscr{B}$ is ρ-reproducible, even if its samples are drawn from a coin $\mathbf{C}$ with bias in $(1/2-\tau, 1/2+\tau)$. Then $\mathscr{B}$ requires sample complexity $m \in \Omega(1/(\tau^2\rho^2))$, i.e. $\rho \in \Omega(1/\tau\sqrt{m})$.*

Proof. Assume we have an algorithm $\mathscr{B}(b_1..b_m;r)$ of sample complexity m so that (i) if the b_i's are chosen i.i.d. in $\{0,1\}$ with bias $1/2-\tau$, $\mathscr{B}$ accepts with at most δ probability (over both random r and the b_i's), and (ii) if the b_i's are drawn i.i.d. with bias $1/2+\tau$, $\mathscr{B}$ accepts with at least $1-\delta$ probability.

Let $p \in [0,1]$ denote the bias of a coin. Since $\mathscr{B}$ is ρ-reproducible, $\mathscr{B}$ is ρ-reproducible for any distribution on p. In particular, pick $p \in_U [1/2-\tau, 1/2+\tau]$. Let $\mathbf{C}_{-\tau}$ denote a coin with bias $1/2-\tau$, and let $\mathbf{C}_{+\tau}$ denote a coin with bias $1/2+\tau$. By Markov's inequality, each of the following is true with probability at least $1-1/4$ over choice of r:

- $\mathbf{Pr}_{b_1..b_m \sim_{\text{i.i.d.}} \mathbf{C}_{-\tau}}[\mathscr{B}(b_1..b_m;r) \text{ accepts}] \leq 4\delta$
- $\mathbf{Pr}_{b_1..b_m \sim_{\text{i.i.d.}} \mathbf{C}_{+\tau}}[\mathscr{B}(b_1..b_m;r) \text{ accepts}] \geq 1-4\delta$
- When p is chosen between $1/2-\tau$ and $1/2+\tau$ uniformly, and then $b_1..b_m, b'_1..b'_m$ are sampled i.i.d. with expectation p, $\mathbf{Pr}[\mathscr{B}(b_1..b_m;r) = A(b'_1..b'_m;r)] \geq 1-4\rho$.

By a union bound, there exists an r^* so that every above statement is true. Note that for any p, given $\sum b_i = j$, the samples $b_1..b_m$ are uniformly distributed among all Boolean vectors of Hamming weight j. Let $a_j \overset{\text{def}}{=} \mathbf{Pr}[\mathscr{B}(b_1..b_m;r^*) \text{ accepts } | \sum b_i = j]$. Then the probability $\mathscr{B}$ accepts using r^* on bits with bias p is $\texttt{Acc}(p) = \sum_j a_j \binom{m}{j} p^j (1-p)^{m-j}$. In particular, this is a continuous and differentiable function.

Since $\texttt{Acc}(1/2-\tau) < 4\delta < 1/4$ and $\texttt{Acc}(1/2+\tau) > 1-4\delta > 3/4$, there is a $q \in (1/2-\tau, 1/2+\tau)$ with $\texttt{Acc}(q) = 1/2$. We show that $\texttt{Acc}(p)$ is close to $1/2$ for all p close to q by bounding the derivative $\texttt{Acc}'(p)$ within the interval $[1/4, 3/4]$, which contains $[1/2-\tau, 1/2+\tau]$.

By the standard calculus formulas for derivatives,

$$\begin{aligned}
\texttt{Acc}'(p) &= \sum_j a_j \binom{m}{j} (jp^{j-1}(1-p)^{m-j} - (m-j)p^j(1-p)^{m-j-1}) \\
&= \sum_j a_j \binom{m}{j} p^j(1-p)^{m-j}(j/p - (m-j)/(1-p)) \\
&= \sum_j a_j \binom{m}{j} p^j(1-p)^{m-j}(j-mp)/(p(1-p)).
\end{aligned}$$

Since $1/4 < p < 3/4$, $p(1-p) > 3/16 > 1/6$, and $0 \leq a_j \leq 1$. So this sum is at most

$$\sum_j \binom{m}{j} p^j(1-p)^{m-j} 6|j-mp| = 6\mathbb{E}_j[|j-mp|]$$

where the last expectation is over j chosen as the sum of m random Boolean variables of expectation p. This expectation is $O(m^{1/2})$ because the expectation of the absolute value of the difference between any variable and its expectation is at most the standard deviation for the variable.

Since the derivative is at most $O(\sqrt{m})$, there is an interval I of length $\Omega(1/\sqrt{m})$ around q so that $1/3 < \texttt{Acc}(p) < 2/3$ for all p in this interval. Since $\texttt{Acc}(p) \notin (1/3, 2/3)$ at $p = 1/2 - \tau$ and $p = 1/2 + \tau$, interval I is entirely contained in $(1/2-\tau, 1/2+\tau)$. So, there is an $\Omega(1/\tau\sqrt{m})$ chance that a random $p \in_U [1/2-\tau, 1/2+\tau]$ falls in interval I. For $p \in I$, there is a $2\texttt{Acc}(p)(1-\texttt{Acc}(p)) > 4/9$ conditional probability of non-reproducibility for $\mathscr{B}$. Therefore, $\rho \geq \Omega(1/\tau\sqrt{m})$ and $m \in \Omega(1/\tau^2\rho^2)$.

□

This chapter, in full, is based on the material as it appears in Symposium on Theory of Computing. Impagliazzo, Russell; Lei, Rex; Pitassi, Toniann; Sorrell, Jessica. “Reproducibility in Learning”. The dissertation author was the primary investigator and author of this material.

Bibliography

Aws Albarghouthi and Justin Hsu. Synthesizing coupling proofs of differential privacy. *Proc. ACM Program. Lang.*, 2(POPL):58:1–58:30, 2018.

Dana Angluin and Philip D. Laird. Learning from noisy examples. *Machine Learning*, 2(4): 343–370, 1987. doi: 10.1007/BF00116829. URL https://doi.org/10.1007/BF00116829.

Shi Bai, Tancrède Lepoint, Adeline Roux-Langlois, Amin Sakzad, Damien Stehlé, and Ron Steinfeld. Improved security proofs in lattice-based cryptography: Using the Rényi divergence rather than the statistical distance. *J. Cryptology*, 31(2):610–640, 2018.

Monya Baker. 1,500 scientists lift the lid on reproducibility. *Nature News*, 533 (7604):452, May 2016. doi: 10.1038/533452a. URL https://www.nature.com/news/1-500-scientists-lift-the-lid-on-reproducibility-1.19970.

B. Barak, M. Hardt, and S. Kale. The uniform hardcore lemma via approximate bregman projections. In Claire Mathieu, editor, *Proceedings of the Twentieth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2009*, pages 1193–1200. SIAM, 2009a.

Boaz Barak, Moritz Hardt, and Satyen Kale. The uniform hardcore lemma via approximate bregman projections. In Claire Mathieu, editor, *Proceedings of the Twentieth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2009, New York, NY, USA, January 4-6, 2009*, pages 1193–1200. SIAM, 2009b. URL http://dl.acm.org/citation.cfm?id=1496770.1496899.

Gilles Barthe, Marco Gaboardi, Emilio Jesús Gallego Arias, Justin Hsu, Aaron Roth, and Pierre-Yves Strub. Higher-order approximate relational refinement types for mechanism design and differential privacy. In Sriram K. Rajamani and David Walker, editors, *Proceedings of the 42nd Annual ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages, POPL 2015, Mumbai, India, January 15-17, 2015*, pages 55–68. ACM, 2015.

Gilles Barthe, Rohit Chadha, Paul Krogmeier, A. Prasad Sistla, and Mahesh Viswanathan. Deciding accuracy of differential privacy schemes. *Proc. ACM Program. Lang.*, 5(POPL):1–30, 2021. doi: 10.1145/3434289. URL https://doi.org/10.1145/3434289.

Peter Bartlett and John Shawe-Taylor. Generalization performance of support vector machines and other pattern classifiers, 1998.

Raef Bassily, Kobbi Nissim, Adam Smith, Thomas Steinke, Uri Stemmer, and Jonathan Ullman. Algorithmic stability for adaptive data analysis. In *Proceedings of the Forty-Eighth Annual ACM Symposium on Theory of Computing*, STOC 2016. Association for Computing Machinery, 2016.

C. Glenn Begley and Lee M Ellis. Raise standards for preclinical cancer research. *Nature (London)*, 483(7391):531–533, 2012. ISSN 0028-0836.

Amos Beimel, Kobbi Nissim, and Uri Stemmer. Characterizing the sample complexity of private learners. In Robert D. Kleinberg, editor, *Innovations in Theoretical Computer Science, ITCS '13, Berkeley, CA, USA, January 9-12, 2013*, pages 97–110. ACM, 2013. doi: 10.1145/2422436.2422450. URL https://doi.org/10.1145/2422436.2422450.

Olivier Bousquet and André Elisseeff. Stability and generalization. *J. Mach. Learn. Res.*, 2:499–526, 2002. URL http://jmlr.org/papers/v2/bousquet02a.html.

J. Bradley and R. Schapire. Filterboost: Regression and classification on large datasets. In *Proceedings of the Twenty-First Annual Conference on Neural Information Processing Systems (NIPS)*, 2007.

Leo Breiman. Prediction games and arcing algorithms. *Neural Computation*, 11(7):1493–1517, 1999. doi: 10.1162/089976699300016106. URL https://doi.org/10.1162/089976699300016106.

Mark Bun and Thomas Steinke. Concentrated differential privacy: Simplifications, extensions, and lower bounds. In Martin Hirt and Adam D. Smith, editors, *Theory of Cryptography - 14th International Conference, TCC 2016-B, Beijing, China, October 31 - November 3, 2016, Proceedings, Part I*, volume 9985 of *Lecture Notes in Computer Science*, pages 635–658, 2016. doi: 10.1007/978-3-662-53641-4_24. URL https://doi.org/10.1007/978-3-662-53641-4_24.

Mark Bun, Kobbi Nissim, Uri Stemmer, and Salil P. Vadhan. Differentially private release and learning of threshold functions. *2015 IEEE 56th Annual Symposium on Foundations of Computer Science*, pages 634–649, 2015.

Mark Bun, Marco Leandro Carmosino, and Jessica Sorrell. Efficient, noise-tolerant, and private learning via boosting. In Jacob D. Abernethy and Shivani Agarwal, editors, *Conference on Learning Theory, COLT 2020, 9-12 July 2020, Virtual Event [Graz, Austria]*, Proceedings of Machine Learning Research. PMLR, 2020a. URL http://proceedings.mlr.press/v125/bun20a.html.

Mark Bun, Roi Livni, and Shay Moran. An equivalence between private classification and online prediction. *2020 IEEE 61st Annual Symposium on Foundations of Computer Science (FOCS)*, pages 389–402, 2020b. doi: 10.1109/FOCS46700.2020.00044. URL https://doi.ieeecomputersociety.org/10.1109/FOCS46700.2020.00044.

Clément L Canonne. A survey on distribution testing: Your data is big. but is it blue? *Theory of Computing*, pages 1–100, 2020.

Clément L Canonne, Gautam Kamath, and Thomas Steinke. The discrete gaussian for differential privacy. In H. Larochelle, M. Ranzato, R. Hadsell, M. F. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 15676–15688. Curran Associates, Inc., 2020. URL https://proceedings.neurips.cc/paper/2020/file/b53b3a3d6ab90ce0268229151c9bde11-Paper.pdf.

Jung Hee Cheon, Andrey Kim, Miran Kim, and Yong Soo Song. Homomorphic encryption for arithmetic of approximate numbers. In *ASIACRYPT 2017, Part I*, volume 10624 of *LNCS*, pages 409–437. Springer, 2017.

Jung Hee Cheon, Kyoohyung Han, Seong-Min Hong, Hyoun Jin Kim, Junsoo Kim, Suseong Kim, Hosung Seo, Hyungbo Shim, and Yongsoo Song. Toward a secure drone system: Flying with real-time homomorphic authenticated encryption. *IEEE Access*, 6:24325–24339, 2018a.

Jung Hee Cheon, Kyoohyung Han, Andrey Kim, Miran Kim, and Yongsoo Song. Bootstrapping for approximate homomorphic encryption. In *EUROCRYPT 2018, Part I*, volume 10820 of *LNCS*, pages 360–384. Springer, 2018b.

Jung Hee Cheon, Kyoohyung Han, Andrey Kim, Miran Kim, and Yongsoo Song. A full RNS variant of approximate homomorphic encryption. In *SAC 2018*, volume 11349 of *LNCS*, pages 347–368. Springer, 2018c.

Jung Hee Cheon, Andrey Kim, and Donggeon Yhee. Multi-dimensional packing for HEAAN for approximate matrix arithmetics. *IACR Cryptology ePrint Archive*, 2018:1245, 2018d. URL https://eprint.iacr.org/2018/1245.

Jung Hee Cheon, Duhyeong Kim, Yongdai Kim, and Yongsoo Song. Ensemble method for privacy-preserving logistic regression based on homomorphic encryption. *IEEE Access*, 6:46938–46948, 2018e.

L. Devroye and T. J. Wagner. Distribution-free inequalities for the deleted and holdout error estimates. *IEEE Trans. Inform. Theory*, 25, 1979a.

L. Devroye and T. J. Wagner. Distribution-free performance bounds for potential function fules. *IEEE Trans. Inform. Theory*, 25, 1979b.

Luc Devroye, Abbas Mehrabian, and Tommy Reddad. The total variation distance between high-dimensional gaussians with the same mean, 2018. URL https://arxiv.org/abs/1810.08693.

Carlos Domingo and Osamu Watanabe. Madaboost: A modification of adaboost. In *Proceedings of the Thirteenth Annual Conference on Computational Learning Theory*, COLT '00, pages 180–189. Morgan Kaufmann, 2000.

Cynthia Dwork, Frank McSherry, Kobbi Nissim, and Adam Smith. Calibrating noise to sensitivity in private data analysis. In *Proceedings of the 3rd Conference on Theory of Cryptography*, TCC

’06, pages 265–284. Springer, 2006.

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

Cynthia Dwork, Vitaly Feldman, Moritz Hardt, Toniann Pitassi, Omer Reingold, and Aaron Roth. Generalization in adaptive data analysis and holdout reuse. In *Proceedings of the 28th International Conference on Neural Information Processing Systems - Volume 2*, NIPS’15, page 23502358, Cambridge, MA, USA, 2015a. MIT Press.

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*, STOC ’15, page 117126, New York, NY, USA, 2015b. Association for Computing Machinery. ISBN 9781450335362. doi: 10.1145/2746539.2746580. URL https://doi.org/10.1145/2746539.2746580.

V. Feldman. A general characterization of the statistical query complexity. *CoRR*, abs/1608.02198, 2016.

Vitaly Feldman and David Xiao. Sample complexity bounds on differentially private learning via communication complexity. In Maria-Florina Balcan, Vitaly Feldman, and Csaba Szepesvári, editors, *Proceedings of The 27th Conference on Learning Theory, COLT 2014, Barcelona, Spain, June 13-15, 2014*, volume 35 of *JMLR Workshop and Conference Proceedings*, pages 1000–1019. JMLR.org, 2014. URL http://proceedings.mlr.press/v35/feldman14b.html.

Matthew Fredrikson and Somesh Jha. Satisfiability modulo counting: a new approach for analyzing privacy properties. In Thomas A. Henzinger and Dale Miller, editors, *Joint Meeting of the Twenty-Third EACSL Annual Conference on Computer Science Logic (CSL) and the Twenty-Ninth Annual ACM/IEEE Symposium on Logic in Computer Science (LICS), CSL-LICS ’14, Vienna, Austria, July 14 - 18, 2014*, pages 42:1–42:10. ACM, 2014. doi: 10.1145/2603088.2603097. URL https://doi.org/10.1145/2603088.2603097.

Yoav Freund and Robert E. Schapire. Game theory, on-line prediction and boosting. In Avrim Blum and Michael J. Kearns, editors, *Proceedings of the Ninth Annual Conference on Computational Learning Theory, COLT 1996, Desenzano del Garda, Italy, June 28-July 1, 1996*, pages 325–332. ACM, 1996. doi: 10.1145/238061.238163. URL https://doi.org/10.1145/238061.238163.

Yoav Freund and Robert E. Schapire. A decision-theoretic generalization of on-line learning and an application to boosting. 55(1):119–139, 1997.

Marco Gaboardi, Andreas Haeberlen, Justin Hsu, Arjun Narayan, and Benjamin C. Pierce. Linear dependent types for differential privacy. In *The 40th Annual ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages, POPL ’13, Rome, Italy - January 23 - 25, 2013*, pages 357–370. ACM, 2013. doi: 10.1145/2429069.2429113. URL https://doi.org/10.1145/2429069.

2429113.

Marco Gaboardi, Kobbi Nissim, and David Purser. The complexity of verifying loop-free programs as differentially private. In Artur Czumaj, Anuj Dawar, and Emanuela Merelli, editors, *47th International Colloquium on Automata, Languages, and Programming, ICALP 2020, July 8-11, 2020, Saarbrücken, Germany (Virtual Conference)*, volume 168 of *LIPIcs*, pages 129:1–129:17. Schloss Dagstuhl - Leibniz-Zentrum für Informatik, 2020.

Wei Gao and Zhi-Hua Zhou. Approximation stability and boosting. In Marcus Hutter, Frank Stephan, Vladimir Vovk, and Thomas Zeugmann, editors, *Algorithmic Learning Theory, 21st International Conference, ALT 2010, Canberra, Australia, October 6-8, 2010. Proceedings*, volume 6331 of *Lecture Notes in Computer Science*, pages 59–73. Springer, 2010. doi: 10.1007/978-3-642-16108-7_9. URL https://doi.org/10.1007/978-3-642-16108-7_9.

Erann Gat and Shafi Goldwasser. Probabilistic search algorithms with unique answers and their cryptographic applications. *Electron. Colloquium Comput. Complex.*, 18:136, 2011.

Craig Gentry, Shai Halevi, and Nigel P. Smart. Homomorphic evaluation of the AES circuit. In *CRYPTO 2012*, volume 7417 of *LNCS*, pages 850–867. Springer, 2012.

Badih Ghazi, Ravi Kumar, and Pasin Manurangsi. User-Level Differentially Private Learning via Correlated Sampling . In *Proc. 35th Annual Conference on Neural Information Processing Systems (NeurIPS)*, 2021.

Anna C. Gilbert and Audra McMillan. Property testing for differential privacy. *CoRR*, abs/1806.06427, 2018. URL http://arxiv.org/abs/1806.06427.

Oded Goldreich. Multi-pseudodeterministic algorithms. *Electron. Colloquium Comput. Complex.*, 26:12, 2019. URL https://eccc.weizmann.ac.il/report/2019/012.

Oded Goldreich, Shafi Goldwasser, and Dana Ron. On the possibilities and limitations of pseudodeterministic algorithms. In *Proceedings of the 4th Conference on Innovations in Theoretical Computer Science*, ITCS '13, page 127138, New York, NY, USA, 2013. Association for Computing Machinery. ISBN 9781450318594. doi: 10.1145/2422436.2422453. URL https://doi.org/10.1145/2422436.2422453.

Shafi Goldwasser and Ofer Grossman. Bipartite perfect matching in pseudo-deterministic nc. In *ICALP*, 2017.

Shafi Goldwasser and Silvio Micali. Probabilistic encryption. *J. Comput. Syst. Sci.*, 28(2):270–299, 1984. doi: 10.1016/0022-0000(84)90070-9. URL https://doi.org/10.1016/0022-0000(84)90070-9.

Shafi Goldwasser, Ofer Grossman, and Dhiraj Holden. Pseudo-Deterministic Proofs. In Anna R. Karlin, editor, *9th Innovations in Theoretical Computer Science Conference (ITCS 2018)*, volume 94 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 17:1–17:18, Dagstuhl, Germany, 2018. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik. ISBN 978-3-95977-060-6. doi: 10.4230/LIPIcs.ITCS.2018.17. URL http://drops.dagstuhl.de/opus/volltexte/2018/8366.

Shafi Goldwasser, Ofer Grossman, Sidhanth Mohanty, and David P. Woodruff. Pseudo-deterministic streaming. *Electron. Colloquium Comput. Complex.*, 26:177, 2019.

Ofer Grossman and Yang P. Liu. Reproducibility and pseudo-determinism in log-space. In Timothy M. Chan, editor, *Proceedings of the Thirtieth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2019, San Diego, California, USA, January 6-9, 2019*, pages 606–620. SIAM, 2019. doi: 10.1137/1.9781611975482.38. URL https://doi.org/10.1137/1.9781611975482.38.

Kyoohyung Han, Seungwan Hong, Jung Hee Cheon, and Daejun Park. Logistic regression on homomorphic encrypted data at scale. In *AAAI 2019*, pages 9466–9471. AAAI Press, 2019.

HElib. HElib (release 2.2.0). https://github.com/homenc/HElib, 2021. IBM.

Peter Henderson, Riashat Islam, Philip Bachman, Joelle Pineau, Doina Precup, and David Meger. Deep reinforcement learning that matters, 2017. URL http://arxiv.org/abs/1709.06560. cite arxiv:1709.06560Comment: Accepted to the Thirthy-Second AAAI Conference On Artificial Intelligence (AAAI), 2018.

Justin Hsu, Aaron Roth, and Jonathan Ullman. Differential privacy for the analyst via private equilibrium computation. In Dan Boneh, Tim Roughgarden, and Joan Feigenbaum, editors, *Symposium on Theory of Computing Conference, STOC'13, Palo Alto, CA, USA, June 1-4, 2013*, pages 341–350. ACM, 2013. doi: 10.1145/2488608.2488651. URL https://doi.org/10.1145/2488608.2488651.

Justin Hsu, Aaron Roth, Tim Roughgarden, and Jonathan Ullman. Privately solving linear programs. In Javier Esparza, Pierre Fraigniaud, Thore Husfeldt, and Elias Koutsoupias, editors, *Automata, Languages, and Programming - 41st International Colloquium, ICALP 2014, Copenhagen, Denmark, July 8-11, 2014, Proceedings, Part I*, volume 8572 of *Lecture Notes in Computer Science*, pages 612–624. Springer, 2014. doi: 10.1007/978-3-662-43948-7_51. URL https://doi.org/10.1007/978-3-662-43948-7_51.

Russell Impagliazzo. Hard-core distributions for somewhat hard problems. In *Proceedings of the 36th Annual Symposium on Foundations of Computer Science*, FOCS '95, pages 538–545. IEEE Computer Society, 1995.

Russell Impagliazzo, Rex Lei, and Jessica Sorrell. Reproducibility in Learning. In *Theory and Practice of Differential Privacy*, 2021.

Russell Impagliazzo, Rex Lei, Toniann Pitassi, and Jessica Sorrell. Reproducibility in learning. In *Proceedings of the 54th Annual ACM SIGACT Symposium on Theory of Computing*, STOC 2022, pages 818–831, New York, NY, USA, 2022. Association for Computing Machinery.

ISBN 9781450392648. doi: 10.1145/3519935.3519973. URL https://doi.org/10.1145/3519935.3519973.

John P. A. Ioannidis. Why most published research findings are false. *PLoS Med 2(8): e124*, 2005. URL https://doi.org/10.1371/journal.pmed.0020124.

Riashat Islam, Peter Henderson, Maziar Gomrokchi, and Doina Precup. Reproducibility of benchmarked deep reinforcement learning tasks for continuous control. *CoRR*, abs/1708.04133, 2017. URL http://arxiv.org/abs/1708.04133.

Christopher Jung, Katrina Ligett, Seth Neel, Aaron Roth, Saeed Sharifi-Malvajerdi, and Moshe Shenfeld. A new analysis of differential privacy's generalization guarantees. *CoRR*, abs/1909.03577, 2019. URL http://arxiv.org/abs/1909.03577.

Haim Kaplan, Katrina Ligett, Yishay Mansour, Moni Naor, and Uri Stemmer. Privately learning thresholds: Closing the exponential gap. In Jacob D. Abernethy and Shivani Agarwal, editors, *Conference on Learning Theory, COLT 2020, 9-12 July 2020, Virtual Event [Graz, Austria]*, volume 125 of *Proceedings of Machine Learning Research*, pages 2263–2285. PMLR, 2020. URL http://proceedings.mlr.press/v125/kaplan20a.html.

Shiva Prasad Kasiviswanathan, Homin K. Lee, Kobbi Nissim, Sofya Raskhodnikova, and Adam Smith. What can we learn privately? 40(3):793–826, 2011.

M. J. Kearns. Efficient noise-tolerant learning from statistical queries. *Journal of the ACM*, 45(6): 983–1006, 1998.

M. J. Kearns and D. Ron. Algorithmic stability and sanity-check bounds for leave-one-out cross-validation. *Neural Comput.*, 11, 1999.

Guy Kindler, Ryan O'Donnell, Anup Rao, and Avi Wigderson. Spherical cubes: Optimal foams from computational hardness amplification. *Communications of the ACM*, 55, 10 2012. doi: 10.1145/2347736.2347757.

Jyrki Kivinen and Manfred K. Warmuth. Boosting as entropy projection. In Shai Ben-David and Philip M. Long, editors, *Proceedings of the Twelfth Annual Conference on Computational Learning Theory, COLT 1999, Santa Cruz, CA, USA, July 7-9, 1999*, pages 134–144. ACM, 1999. doi: 10.1145/307400.307424. URL https://doi.org/10.1145/307400.307424.

Samuel Kutin and Partha Niyogi. The interaction of stability and weakness in adaboost. 11 2001.

Samuel Kutin and Partha Niyogi. Almost-everywhere algorithmic stability and generalization error. In Adnan Darwiche and Nir Friedman, editors, *UAI '02, Proceedings of the 18th Conference in Uncertainty in Artificial Intelligence, University of Alberta, Edmonton, Alberta, Canada, August 1-4, 2002*, pages 275–282. Morgan Kaufmann, 2002. URL https://dslpitt.org/uai/displayArticleDetails.jsp?mmnu=1&smnu=2&article_id=870&proceeding_id=18.

Lattigo. Lattigo 2.2.0. Online: http://github.com/ldsec/lattigo, July 2021. EPFL-LDS.

Baiyu Li and Daniele Micciancio. On the security of homomorphic encryption on approximate numbers. In Anne Canteaut and François-Xavier Standaert, editors, *Advances in Cryptology – EUROCRYPT 2021*, pages 648–677, Cham, 2021. Springer International Publishing. ISBN 978-3-030-77870-5.

Baiyu Li, Daniele Micciancio, Mark Schultz, and Jessica Sorrell. Securing approximate homomorphic encryption using differential privacy. Cryptology ePrint Archive, Paper 2022/816, 2022. URL https://eprint.iacr.org/2022/816. https://eprint.iacr.org/2022/816.

Mario Lucic, Karol Kurach, Marcin Michalski, Sylvain Gelly, and Olivier Bousquet. Are gans created equal? a large-scale study. In S. Bengio, H. Wallach, H. Larochelle, K. Grauman, N. Cesa-Bianchi, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc., 2018. URL https://proceedings.neurips.cc/paper/2018/file/e46de7e1bcaaced9a54f1e9d0d2f800d-Paper.pdf.

Vadim Lyubashevsky, Chris Peikert, and Oded Regev. On ideal lattices and learning with errors over rings. *J. ACM*, 60(6):43:1–43:35, 2013. doi: 10.1145/2535925. URL https://doi.org/10.1145/2535925.

Daniele Micciancio and Michael Walter. Gaussian sampling over the integers: Efficient, generic, constant-time. In *CRYPTO (2)*, volume 10402 of *Lecture Notes in Computer Science*, pages 455–485. Springer, 2017.

Daniele Micciancio and Michael Walter. On the bit security of cryptographic primitives. In *EUROCRYPT (1)*, volume 10820 of *Lecture Notes in Computer Science*, pages 3–28. Springer, 2018.

Ilya Mironov. RÃ©nyi differential privacy. In *2017 IEEE 30th Computer Security Foundations Symposium (CSF)*, pages 263–275, 2017. doi: 10.1109/CSF.2017.11.

Nima Mousavi. How tight is chernoff bound? Unpublished manuscript.

Arjun Narayan, Ariel Feldman, Antonis Papadimitriou, and Andreas Haeberlen. Verifiable differential privacy. In Laurent Réveillère, Tim Harris, and Maurice Herlihy, editors, *Proceedings of the Tenth European Conference on Computer Systems, EuroSys 2015, Bordeaux, France, April 21-24, 2015*, pages 28:1–28:14. ACM, 2015. doi: 10.1145/2741948.2741978. URL https://doi.org/10.1145/2741948.2741978.

Huy L. Nguyễn, Jonathan Ullman, and Lydia Zakynthinou. Efficient private algorithms for learning halfspaces. 2019.

PALISADE. PALISADE lattice cryptography library (release 1.11.6). https://gitlab.com/palisade/, 2022. PALISADE Project.

Saerom Park, Jaewook Lee, Jung Hee Cheon, Juhee Lee, Jaeyun Kim, and Junyoung Byun. Security-preserving support vector machine with fully homomorphic encryption. In *SafeAI@AAAI 2019*, volume 2301 of *CEUR Workshop Proceedings*. CEUR-WS.org, 2019.

Chris Peikert. Public-key cryptosystems from the worst-case shortest vector problem: extended abstract. In *STOC*, pages 333–342. ACM, 2009.

Chris Peikert, Oded Regev, and Noah Stephens-Davidowitz. Pseudorandomness of ring-lwe for any ring and modulus. In *STOC*, pages 461–473. ACM, 2017.

Joelle Pineau, Philippe Vincent-Lamarre, Koustuv Sinha, Vincent Larivière, Alina Beygelzimer, Florence d'Alché Buc, Emily Fox, and Hugo Larochelle. Improving reproducibility in machine learning research (a report from the neurips 2019 reproducibility program). *arXiv preprint arXiv:2003.12206*, 2020.

Yury Polyanskiy and Yihong Wu. Lecture notes on information theory. *Lecture Notes for ECE563 (UIUC) and*, 6(2012-2016):7, 2014.

Thomas Pöppelmann, Léo Ducas, and Tim Güneysu. Enhanced lattice-based signatures on reconfigurable hardware. In Lejla Batina and Matthew Robshaw, editors, *Cryptographic Hardware and Embedded Systems – CHES 2014*, pages 353–370, Berlin, Heidelberg, 2014. Springer Berlin Heidelberg. ISBN 978-3-662-44709-3.

Alexander Rakhlin. Lecture notes on online learning (draft), 2009. URL http://www.mit.edu/~rakhlin/papers/online_learning.pdf.

Jason Reed and Benjamin C. Pierce. Distance makes the types grow stronger: a calculus for differential privacy. In Paul Hudak and Stephanie Weirich, editors, *Proceeding of the 15th ACM SIGPLAN ICFP 2010, Baltimore, Maryland, USA, September 27-29, 2010*, pages 157–168. ACM, 2010.

Oded Regev. On lattices, learning with errors, random linear codes, and cryptography. *J. ACM*, 56 (6):34:1–34:40, 2009.

Robert E. Schapire, Yoav Freund, Peter Barlett, and Wee Sun Lee. Boosting the margin: A new explanation for the effectiveness of voting methods. In Douglas H. Fisher, editor, *Proceedings of the Fourteenth International Conference on Machine Learning (ICML 1997), Nashville, Tennessee, USA, July 8-12, 1997*, pages 322–330. Morgan Kaufmann, 1997.

SEAL. Microsoft SEAL (release 3.6). https://github.com/Microsoft/SEAL, November 2020. Microsoft Research, Redmond, WA.

R. Servedio. Smooth boosting and learning with malicious noise. *Journal of Machine Learning Research*, 4:633–648, 2003a.

Rocco A. Servedio. PAC analogues of perceptron and winnow via boosting the margin. In *Proceedings of the Thirteenth Annual Conference on Computational Learning Theory (COLT 2000), June 28 - July 1, 2000, Palo Alto, California, USA*, pages 148–157, 2000.

Rocco A. Servedio. Pac analogues of perceptron and winnow via boosting the margin. *Machine Learning*, 47:133–151, 2002.

Rocco A. Servedio. Smooth boosting and learning with malicious noise. *J. Mach. Learn. Res.*, 4: 633–648, 2003b. URL http://jmlr.org/papers/v4/servedio03a.html.

S. Shalev-Schwartz, O. Shamir, N. Srebro, and K. Sridharan. Learnability, stability and uniform convergence. *J. Mach. Learn. Res.*, 11, 2010.

Shai Shalev-Shwartz. Online learning and online convex optimization. *Foundations and Trends in Machine Learning*, 4(2):107–194, 2012. doi: 10.1561/2200000018. URL https://doi.org/10.1561/2200000018.

L. G. Valiant. A theory of the learnable. In *Proc. 16th Annual ACM Symposium on Theory of Computing (STOC)*, pages 436–445. ACM Press, 1984a.

Leslie G. Valiant. A theory of the learnable (cacm). *Commun. ACM*, 27(11):1134–1142, 1984b. doi: 10.1145/1968.1972. URL http://doi.acm.org/10.1145/1968.1972.

Leslie G. Valiant. Learning disjunction of conjunctions. In *Proceedings of the 9th International Joint Conference on Artificial Intelligence*, IJCAI '85, pages 560–566. Morgan Kaufmann, 1985.

V. N. Vapnik and A. Ya. Chervonenkis. On the uniform convergence of relative frequencies of events to their probabilities. *Theory of Probability and its Applications*, 16, 1971.

Ulrike von Luxburg. Clustering stability: An overview. *Foundations and TrendsÂ® in Machine Learning*, 2(3):235–274, 2010. ISSN 1935-8237. doi: 10.1561/2200000008. URL http://dx.doi.org/10.1561/2200000008.

Danfeng Zhang and Daniel Kifer. Lightdp: towards automating differential privacy proofs. In Giuseppe Castagna and Andrew D. Gordon, editors, *Proceedings of the 44th ACM SIGPLAN Symposium on Principles of Programming Languages, POPL 2017, Paris, France, January 18-20, 2017*, pages 888–901. ACM, 2017.