STATISTICAL LEARNING VIA STOCHASTIC OPTIMIZATION UNDER DATA PRIVACY CONSIDERATIONS

by
Md Enayat Ullah

A dissertation submitted to The Johns Hopkins University in conformity with the requirements for the degree of Doctor of Philosophy

Baltimore, Maryland
January, 2024

Abstract

In this dissertation, we study statistical learning, formulated as stochastic optimization problems, under modern constraints motivated by data privacy considerations. The goal is to understand the statistical and computational complexity in algorithm design for fundamental classes of problems.

The first part concerns *differential privacy*, which, in recent years, has emerged as the de-facto standard for privacy-preserving data analysis. We study design of differentially private algorithms for *(a).* supervised learning of linear predictors with convex losses, also known as convex *generalized linear models*, and *(b).* non-convex optimization, where the goal is to approximate stationary points of the *risk function*. Our derived guarantees for the proposed algorithms are, as of yet, the best-known, and in most cases, are shown to be *nearly optimal*, in the worst case.

The second part concerns the problem of *machine unlearning*. The goal here is to efficiently update a trained model under requests to unlearn a data point in the training dataset. We delve into the problem for widely-studied classes of convex losses: smooth/non-smooth settings and generalized linear models. We propose learning and corresponding unlearning algorithms, which are (non-trivially) accurate and efficient. Further, we extend our techniques, to unlearn general *structured iterative procedures*, and a *streaming* setting, where the unlearning requests arrive sequentially.

Thesis Readers

Dr. Raman Arora (Primary Advisor)

Department of Computer Science

The Johns Hopkins University

Dr. Amitabh Basu

Department of Applied Mathematics and Statistics

The Johns Hopkins University

Dr. Michael Dinitz

Department of Computer Science

The Johns Hopkins University

Dedicated to my family.

Acknowledgements

I am indebted to many who have generously provided support and wisdom, aiding both my professional growth and personal well-being, during my doctoral journey. I am afraid my words may fall short in expressing this, but I will try my best.

I owe immense gratitude to my advisor, Prof. Raman Arora, for his continued guidance and support. His mentorship was instrumental in shaping my research interests and honing my critical-thinking skills. He always remained patient, kind and encouraging, even when I faced (extended) periods of little progress, or even a clear direction. I am particularly grateful for facilitating opportunities to visit the Institute for Advanced Study, Princeton, and Simons Institute for the Theory of Computing, as well as for fostering numerous collaborations.

I thank Prof. Amitabh Basu and Prof. Michael Dinitz, for agreeing to be a part of my dissertation committee, and for being excellent teachers and mentors.

I am fortunate to have regularly collaborated with Prof. Raef Bassily, Prof. Cristóbal Guzmán and Michael Menart, which has been richly rewarding and productive. Every meeting was a learning experience, markedly enhancing my understanding of the field. I deeply appreciate Raef and Cristóbal's fervent enthusiasm for research and progress, which continually fueled my motivation. I also thank Tomás González, who joined us on a project, and made invaluable contributions.

I undertook two summer internships, at Adobe and Google Research. My heartfelt thanks to my mentors, Anup Rao, Tung Mai and Ryan Rossi at Adobe, and Peter

Kairouz, Sewoong Oh and Christopher Choquette-Choo, at Google. The internships provided me with valuable experiences, and broadened my perspective, especially regarding research in industry. I'm also sincerely grateful to my mentors for the career counsel they generously offered whenever I sought it.

I thank my (former and current) labmates: Jalaj, Poorya, Teodor, Yunjuan, Austin, Anh, Thanh and Kaibo. I am lucky to have collaborated with many of you. I appreciate you all for serving as my sounding board, helping clarify my doubts and sharing in my excitement over various (often unrelated) topics. I am particularly thankful to Poorya and Teodor for guiding me in the initial years.

I am grateful to Prof. Vladimir (Vova) Braverman, which whom I worked during my initial Ph.D. years. Vova was thoughtful and generous in introducing me to many wonderful colleagues and mentors. Several of them became my co-authors, including Nikita Ivkin, Daniel Rothchild, Ashwinee Panda, Harry Lang, Samson Zhou, Prof. Ion Stoica and Prof. Joseph Gonzalez. I enjoyed and learned a lot working with them.

I thank Jalaj Upadhyay and Venkata Gandikota (GV) for their ample and earnest guidance, particularly during the onset of my Ph.D. Jalaj sparked my interest in *differential privacy*, which forms a core component of this thesis. He has looked after me throughout these years, promptly answering my queries, technical and otherwise.

I convey my sincere gratitude to my mentors at IIT Kanpur, Prof. Purushottam Kar, Prof. Debasis Kundu and Dr. Prateek Jain, for nurturing my early enthusiasm for (theoretical) research and encouraging me towards pursuing graduate studies.

I am indebted to the exceptional teachers at JHU, IIT Kanpur, and those before them, who have been essential in shaping my path to this moment.

I thank the CS staff, especially Kim Franklin, for streamlining the administrative tasks. Kudos to the coffee czars for ensuring the espresso machine in Malone works!

I thank my housemates, Samvit and Saurabh, at *Carolina*. The Ph.D. journey

would likely have been much more difficult if it were not for their company (especially during the COVID-19 times). I will dearly miss hanging out with you guys! I am fortunate to have made lifelong friends, including Ravi, Sandeep, Samik, Sayan, Yasamin, Niharika, Nikita, Aditya, Anirbit, Cristina, Piyush, Harsh, Razieh, Rohit, Eli, Jaron, Trung, Arka, Aarushi, Jingfeng, Akshay, Pushpendre (and others whose names I might be forgetting).

I fondly remember Ashish Arora, former Ph.D. student at JHU and a close friend who introduced me to the (then existent) *Corona Cricket* group. Sadly, grappling with mental health challenges, Ashish chose to end his life last year. I mourn his untimely loss and extend my deepest sympathies to his family.

I thank my friends, from IIT Kanpur, high school and others, who, to my great fortune, are too many to list. I thank you all for staying in touch even when miles and time zones separate us.

I thank my late father, mother, siblings and extended family, who, despite having little clue of what (or why) I am doing, supported me unwaveringly. I treasure the loving memories of my father, who tragically passed away with COVID-19. Each memory is a testament to his enduring love, which continues to guide me. Words fall short in conveying the immense gratitude I owe to my family.

Lastly, my enduring thanks to the small but vibrant town of *Madhupur, India*, where I was born and raised, for the spirit and warmth of the place and its people.

Contents

List of Figures

Chapter 1

Introduction

Over the past decade, machine learning has witnessed remarkable advancements, sparking a revolution across diverse domains like computer vision [RPG+21], language and speech processing [Ope23, RKX+23], game-play [SHM+16] and biochemistry [JEP+21]. This success stems from both conceptual and algorithmic breakthroughs, as well as the unprecedented access to substantial computational resources and vast datasets. However, the progress and widespread adoption has not come without its share of contemporary challenges. An important desideratum in modern applications, which is the focus of this dissertation, is *data privacy*.

With data-driven analysis and services becoming ubiquitous, there is a growing push for broader awareness of data privacy and ownership. Often, the data contains sensitive personal information, and its use poses considerable risks. A pertinent example is the prevalent use of *large language models*, and how raw personal data can be retrieved from these [CTW+21, NCH+23]. These concerns have driven the development of rigorous approaches to safeguard data privacy. Further, these efforts, in part, have led to regulatory bodies enacting laws, such as the *European Union's General Data Protection Regulation (GDPR)*, to uphold these standards.

In this dissertation, we focus on the following two problems, broadly-outlined, motivated by data privacy considerations. We study these problems under the unified

and ubiquitous framework of stochastic (convex) optimization.

Differentially Private Machine Learning. Differential privacy (DP), introduced in [DMNS06], has emerged as the gold standard for privacy protection in data analysis. It ensures the confidentiality of an individual's sensitive information in released models or statistics derived from data analysis, while concurrently allowing for the extraction of statistical insights about the broader population. Differential privacy has been widely adopted, notably by the U.S. Census Bureau [Abo18], and companies like Google and Apple in their products [XZA+23, Inc17].

The constraint of differential privacy often leads to a degradation in *utility*, both in theory and practice. The goal, thus, is to understand the complexities and devise algorithms which provide the best possible utility. We study two problems in differentially private machine learning: *(a).* learning convex generalized linear models, and *(b).* approximating stationary points in non-convex optimization. These foundational problems have been the subject of numerous prior works, yet significant gaps in our knowledge still remain. We bridge these gaps by proposing algorithms with improved guarantees and establishing fundamental limits on what is achievable.

Machine Unlearning. The problem of machine unlearning concerns with updating trained machine learning models to comply with requests to *unlearn/forget/delete* an individual's data from the training dataset. This problem has recently gained attention owing to various data privacy laws such as *European Union's General Data Protection Regulation (GDPR)* and *California Consumer Act (CCA)*, which empower users to make such requests to the entity possessing user data (see *Right to be forgotten*, [Wik21]). The entity is then required to update the state of the system such that (ideally) all *information* pertaining to the user data has been removed.

The goal here is to formalize what it means to *unlearn*, and develop meaningful

and efficient approaches towards achieving it. In our work, we consider the most strict unlearning criterion, called *exact unlearning*. We show that in various settings of stochastic convex optimization (and even beyond), we can design learning and a corresponding exact unlearning algorithm which are *non-trivially accurate* and *efficient*, in comparison to the standard benchmark of *recomputation*.

1.1 Organization

We give a brief outline and summary of the contributions presented in this dissertation below. In the subsequent Section 1.2, we setup the problem formally and introduce preliminaries, which allows us to expand on our contributions in Section 1.3.

1. In Chapter 2, we consider the problem of learning *Generalized Linear Models (GLM)* under differential privacy. GLMs form a large and fundamental class of supervised learning problems which includes linear and logistic regression and support vector machines. We study two classes of convex GLMS, *(a). Lipschitz GLM* (such as hinge loss in support vector machines) and *(b). non-negative and smooth GLM* (such as squared loss in linear regression), and propose efficient differentially private algorithms which we achieve *near-optimal* rates on the *excess population risk*. This chapter is based on [ABG+22], published in the proceedings of *Neural Information Processing Systems (NeurIPS), 2022*.

2. In Chapter 3, we consider the problem of *differentially private non-convex optimization*, where the goal is to find an approximate stationary point of the *empirical risk* or *population risk*. For both these criteria, we propose DP algorithms which achieve improved rates over existing works. Further, we give a lower bound demonstrating that in some regimes of problem parameters, our upper bounds are tight. Finally, motivated by the (in-general) gap between our upper and lower bound, we study the problem under the additional assumption

of convexity, and establish the optimal rate. This chapter is based on [ABG+23], published in the proceedings of *International Conference on Machine Learning (ICML), 2023*.

3. In Chapter 4, we study the problem of machine unlearning in Stochastic Convex Optimization (SCO). We formalize the notion of *exact unlearning*, which, roughly speaking, requires the updated model to be perfectly indistinguishable from retraining. We identify a notion of *algorithmic stability* called *Total Variation (TV) Stability*, and show how it can be useful for designing exact unlearning algorithms. For smooth convex losses, we propose a TV stable learning and a corresponding (exact) unlearning algorithm, which are accurate and more efficient than retraining. This chapter is based on [UMR+21], published in the proceedings of *Conference on Learning Theory (COLT), 2021*.

4. In Chapter 5, we further explore the problem of machine unlearning building on the afore-mentioned framework of Total Variation stability. We propose efficient unlearning algorithms for general *structured* learning algorithms. Specifically, we consider algorithms which perform, what we call, *adaptive query release*, from *structured query classes* on the dataset. This includes iterative procedures such as those in optimization and beyond. This yields unlearning algorithms for (smooth) SCO and GLMs. As an example, we obtain the first *sub-linear time* unlearning algorithm for logistic regression while achieving the optimal rate. This chapter is based on [UA23], published in the proceedings of *International Conference on Machine Learning (ICML), 2023*.

1.2 Problem Setup and Preliminaries

In this section, we formally setup the problem and introduce the preliminaries and notation, used in the rest of the dissertation. We present basic primitives in optimization

and differential privacy, and survey the most related works.

Notation and terminology. We use the standard Big-O notations, $O(\cdot), \Omega(\cdot), o(\cdot), \omega(\cdot)$ and $\Theta(\cdot)$. Further, we use $\widetilde{O}(\cdot)$ (and similarly others), to suppress poly-logarithmic factors of parameters. Claims of *optimality* of any quantity, are made in the sense of *asymptotic order optimality*, with the asymptotic parameter being the number of samples $n \to \infty$, unless otherwise specified. Further, we routinely deal with *distance* and *divergence* between probability distributions. We sometimes abuse notation and write distance and divergence between random variables - these should be interpreted as using the *laws* of the corresponding random variables.

In Table 1-I, we provide the notation and their description, used in this dissertation.

1.2.1 Machine Learning as Stochastic Optimization

Statistical machine learning tasks can often be posed as stochastic optimization problems. Let $\mathcal{Z}$ denote a *data space* and $\mathcal{W} \subseteq \mathbb{R}^d$ denote the set of parameters. We will often consider a **constrained** setting wherein we assume that the set $\mathcal{W}$ is closed, convex and bounded, with diameter $D = \sup_{w,w' \in \mathcal{W}} \|w - w'\| < \infty$. Here and everywhere else, the norm, $\|\cdot\|$, is the Euclidean norm, unless otherwise stated. Let $\ell : \mathcal{W} \times \mathcal{Z} \to \mathbb{R}$ denote a **loss function**. Consider a probability distribution $\mathcal{D}$ over $\mathcal{Z}$. The **population risk** of w with respect to $\mathcal{D}$, is defined as,

$$L(w; \mathcal{D}) = \mathbb{E}_{z \sim \mathcal{D}}[\ell(w; z)].$$

The goal is to design a procedure $\mathcal{A} : \mathcal{Z}^n \to \mathcal{W}$ which given n independently and identically distributed (i.i.d.) samples from $\mathcal{D}$ as input, called the *training set*, $S = \{z_1, z_2, \ldots, z_n\}$, the (expected) population risk of its output is small. Formally, we want to control the following, expected **excess population risk** quantity,

$$\mathbb{E}_{S \sim \mathcal{D}^n, \mathcal{A}} \left[L(\mathcal{A}(S); \mathcal{D}) - \min_{w \in \mathcal{W}} L(w; \mathcal{D}) \right].$$

The expectation above is with respect to the randomness in training data S as well as in the algorithm $\mathcal{A}$. Often, it is desirable to control the above quantity in *high-probability*, but in this dissertation, we will mostly limit to a control in expectation. We expect the excess population risk to go to zero as number of samples $n \to \infty$, which is why we will often refer to bounds on the quantity as *rate*. Finally, we want such a guarantee without imposing any structural constraints on the distribution $\mathcal{D}$, which is often referred to as the **distribution-free** setting.

1.2.1.1 Empirical Risk Minimization (ERM)

A canonical approach to the stochastic optimization problem is Empirical Risk Minimization. Given a dataset $S = \{z_1, z_2, \ldots, z_n\}$, the **empirical risk** is defined as,

$$\widehat{L}(w; S) = \frac{1}{n} \sum_{i=1}^{n} \ell(w; z_i).$$

The *empirical risk minimization rule* is defined as,

$$\widehat{w} \in \arg\min_{w \in \mathcal{W}} \widehat{L}(w; S).$$

Further, any minimizer above is called an *empirical risk minimizer*. We will overload the abbreviation ERM to refer to both the problem and the minimizer, however it should be clear from context what we refer to.

Even though ERM is motivated as an approach to solve the stochastic optimization problem, we do not impose any distributional constraints on the training set S. This serves two purposes: *(a).* it makes the ERM problem reducible from *distribution-free* stochastic optimization, which is useful in deducing *lower bounds*, and *(b).* it allows the results to apply to general *finite-sum optimization* problems, which need not arise from a machine learning context.

As in the previous section, we are interested in designing a procedure $\mathcal{A}$ with small (expected) **excess empirical risk**, defined as,

$$\mathbb{E}_{\mathcal{A}} \left[\widehat{L}(\mathcal{A}(S); S) - \min_{w \in \mathcal{W}} \widehat{L}(w; S) \right].$$

We note that since the above is a deterministic problem, it is non-trivial only under computational constraints and/or additional restrictions on the procedure $\mathcal{A}$, which prevents exact minimization.

1.2.1.2 Information-based Complexity

We are interested in understanding the statistical and computational aspects in solving the afore-mentioned problem. These are studied in an information-based complexity framework, which limits knowledge of the underlying population risk function to certain class of queries [NY83]. These complexities are framed in a *minimax* sense, which correspond to guarantees of the best procedure (min), for the worst instance (max). For a rigorous treatment, we refer the reader to standard optimization textbooks, such as [NY83] and [N+18].

Statistical (or sample) complexity refers to the number of samples n, needed to get an expected excess population risk of at most α. That is, for every query, we are given the full description of the map $w \mapsto \ell(w; z)$ for an independent z sampled from $\mathcal{D}$. Note that sample complexity for a target α, and a *rate* on excess population risk, in terms of a given number of samples n, provide the same information, and we will use them interchangeably.

Computational complexity stems from access to descriptions of only *local approximations* to the underlying function. Formally, in the **p-th order model**, for a query (w, z) for $z \in S$, we are provided $(\ell(w; z), \nabla\ell(w; z), \ldots \nabla^p\ell(w; z))$, where the gradient and higher-order derivatives are with respect to the argument w. We sometimes operate in the constrained setting over an abstract convex set $\mathcal{W}$. To describe the computation in interacting with the set $\mathcal{W}$, we consider a **projection oracle**, which given a $u \in \mathbb{R}^d$, outputs the point closest to it in $\mathcal{W}$, i.e. $v = \arg\min_{v \in \mathcal{W}} \|u - v\|^2$.

A procedure is an *adaptive querying protocol*, which, with small number of queries, seeks to find a w with small expected excess population (or empirical) risk. In this

dissertation, we limit to the **first-order model**, where for a query (w, z), we are provided $(\ell(w; z), \nabla\ell(w; z))$. Further, we treat the projection and gradient costs as identical, conflating both into one. We refer to this as **gradient complexity**, or simply **runtime**.

1.2.1.3 Regularity Assumptions of Lipschitzness and Smoothness

In order to make the problem non-trivial, we need to impose some regularity constraints on the loss function. We define two such properties.

Definition 1 (G-Lipschitzness)**.** *A function $f : \mathcal{U} \to \mathbb{R}$ is G-Lipschitz, if for all $u, u' \in \mathcal{U}$, we have,*

$$|f(u) - f(u')| \leq G \|u - u'\| .$$

If the function f is differentiable at all points in its domain, the above is equivalent to assuming that $\|\nabla f(u)\| \leq G$ for all $u \in \mathcal{U}$. We now define smoothness.

Definition 2 (H-Smoothness)**.** *A differentiable function $f : \mathcal{U} \to \mathbb{R}$ is H-smooth, if for all $u, u' \in \mathcal{U}$, we have,*

$$\|\nabla f(u) - \nabla f(u')\| \leq H \|u - u'\| .$$

We say that a loss function ℓ is Lipschitz and/or smooth, if the map $w \mapsto \ell(w; z)$ is Lipschitz and/or smooth for all $z \in \mathcal{Z}$.

Under the above assumptions, the optimal rate on the expected excess population risk is $\widetilde{\Omega}\left(\sqrt{\frac{d}{n}}\right)$ [LGOT23]. Further this is achievable, upto poly-log factors, by ERM. However, the two downsides to this result are *(a).* an explicit dimension dependence in the rate, and *(b).* the required gradient complexity is necessarily exponential in the dimension.

1.2.1.4 Stochastic Convex Optimization (SCO)

The setting of stochastic *convex* optimization addresses both the above limitations. We first define convex functions.

Definition 3 (Convexity)**.** *A function $f : \mathcal{U} \to \mathbb{R}$ is convex, if for all $u, u' \in \mathcal{U}$ and all $\lambda \in [0, 1]$,*

$$f(\lambda u + (1 - \lambda)u') \leq \lambda f(u) + (1 - \lambda) f(u').$$

In SCO, we assume that the loss function $w \mapsto \ell(w; z)$ is convex for all $z \in \mathcal{Z}$. Under the additional assumption of G-Lipschitzness, the seminal work of [NY83] showed that the optimal expected excess population risk is $\Theta\left(\frac{GD}{\sqrt{n}}\right)$. This is achieved by the *one-pass Stochastic Gradient Descent* (SGD) procedure with optimal gradient complexity of n. This remarkable result shows that *(a).* it is possible to get a non-trivial rate in all dimensions d, and *(b).* there is no separation of sample and gradient complexity, in this setting. Further, the additional assumption of (any level of) smoothness does not change the optimal rate or runtime.

Strong convexity. A stricter assumption is that of strong convexity, defined below.

Definition 4 (μ-strongly convexity)**.** *A function $f : \mathcal{U} \to \mathbb{R}$ is μ-strongly convex if for all u, u' and $\lambda \in [0, 1]$,*

$$f(\lambda u + (1 - \lambda)u') \leq \lambda f(u) + (1 - \lambda) f(u') - \frac{\lambda(1 - \lambda)\mu \left\|u - u'\right\|^2}{2}.$$

We note that a function can be both μ-strongly convex and G-Lipschitz, only over a set of diameter at most $\frac{G}{\mu}$. Under the μ-strong convexity and G-Lipschitzness assumptions on the loss function $w \mapsto \ell(w; z)$, [AWBR09] showed that the optimal expected excess population risk is $\Theta\left(\frac{G^2}{\mu n}\right)$. For constant G and μ, this improves over the prior rate quadratically. Further, as before, this is achievable by the one-pass SGD procedure with a gradient complexity of n.

The final setting we discuss is that of convex, smooth, and *bounded range* loss functions. Formally, L_0-*bounded range* is defined by $\sup_z \ell(0;z) - \min_w \ell(w;z) \leq L_0$. Via a simple translation argument, it is similar to assuming non-negativity and that $\sup_z \ell(0;z) \leq L_0$ – we will hence call this the *convex, non-negative and smooth* setting. In this case, the optimal expected excess population risk is $\Theta\left(\frac{\sqrt{HL_0}D}{\sqrt{n}}\right)$, which is yet again achieved by the one-pass SGD procedure [SST10, Sha15].

Non-Euclidean Settings. The theory of convex optimization can be generalized to (a large class of) abstract Banach spaces [NY83], which we discuss briefly. Herein, we assume that the the loss function is Lipschitz with respect to a *primal norm* $\|\cdot\|$, which is not necessarily the Euclidean norm, and diameter of the constraint set is bounded in its *dual norm*, $\|\cdot\|_*$[1]. A particularly important case is the ℓ_p/ℓ_q setup, wherein loss function is G_q-Lipschitz with respect to ℓ_p-norm, and diameter of the constraint set is bounded in ℓ_p norm by D_p, with $\frac{1}{p} + \frac{1}{q} = 1$ being *Hölder conjugates*, and $1 < p < \infty$. In this case, the optimal rates are $\Theta\left(\frac{G_p D_q}{\sqrt{(p-1)n}}\right)$ and $\Theta\left(G_p D_q \min\left(\frac{d^{1/2-1/p}}{\sqrt{n}}, \frac{1}{n^{1/p}}\right)\right)$ for $p \in (1,2)$ and $p \geq 2$ respectively [NY83, AWBR09]. This is achievable by the best of one-pass SGD and its non-Euclidean generalization, called the *(Stochastic) Mirror Descent* method.

1.2.1.5 Generalized Linear Model (GLM)

Generalized Linear Models are loss functions popularly encountered in supervised learning settings. They correspond to learning *linear predictors* (described by w) with real-valued losses. Formally, the data space $\mathcal{Z} = \mathcal{X} \times \mathcal{Y}$, where $\mathcal{X} \subseteq \mathbb{R}^d$ is the set of *features* and $\mathcal{Y} \subseteq \mathbb{R}$ is the set of *labels* or *responses*. A GLM is a loss function of the

[1]The **Dual space**, $\mathcal{V}^*$, of a vector space $\mathcal{V}$ over $\mathbb{R}$ is the vector space of all continuous linear functionals from $\mathcal{V}$ to $\mathbb{R}$. For a normed vector space, $(\mathcal{V}, \|\cdot\|)$, the **Dual norm** on $\mathcal{V}^*$ is defined as, $\|f\|_* = \sup\{|f(v)|, v \in \mathcal{V}, \|v\| \leq 1\}$.

following form,

$$\ell(w;(x,y)) = \phi_y(\langle w, x\rangle),$$

where $\phi_y : \mathbb{R} \to \mathbb{R}$ is some *link function*.

Examples of GLMs are abound in machine learning. We give a few below.

Example 1. ***A:*** *Squared loss in Linear Regression,*

$$\ell(w;(x,y)) = (y - \langle w, x\rangle)^2 .$$

B: *Hinge loss in Support Vector Machines (SVM),*

$$\ell(w;(x,y)) = \max(0, 1 - y\langle w, x\rangle) .$$

C: *Logistic loss in Logistic Regression,*

$$\ell(w;(x,y)) = \log(1 + \exp(-y\langle w, x\rangle)) .$$

D: *A single hidden neuron neural network with squared loss,*

$$\ell(w;(x,y)) = (y - tanh(\langle w, x\rangle))^2 .$$

Convexity, Lispchitzness and Smoothness. We say a GLM loss function ℓ is G-Lipschitz, if the map $\phi_y : \mathbb{R} \to \mathbb{R}$ is G-Lipschitz for all $y \in \mathcal{Y}$. Similarly, we say it is H-smooth, if the map $\phi_y : \mathbb{R} \to \mathbb{R}$ is H-smooth for all $y \in \mathcal{Y}$. In the above, Example 1.B and 1.C, are Lipschitz, convex GLMs, Example 1.A and 1.C are smooth, convex GLMs, and Example 1.D is a smooth (non-convex) GLM.

We use $\|\mathcal{X}\|$ and $\|\mathcal{Y}\|$ to denote the boundedness parameters on x and $\phi_y(0)$ i.e. $\|x\| \leq \|\mathcal{X}\|$ and $|\phi_y(0)| \leq \|\mathcal{Y}\|^2$. The later, in popular instances (such as linear regression), corresponds to a bound on responses y. Further, under the non-negativity assumption on the loss function, $\|\mathcal{Y}\|^2$ is simply the bounded range parameter L_0.

For Lipschitz, convex GLMs, the optimal expected excess population risk is $\Theta\left(\frac{1}{\sqrt{n}}\right)$, same as that for the more general class of Lipschitz, convex losses. This follows since the loss function in the hard instance, is a GLM [NY83]. As before, this is achieved by one-pass SGD with a gradient complexity of n. However, interestingly, even without convexity, Lipschitz GLMs (still) exhibit a dimension-independent rate of $\Theta\left(\frac{1}{\sqrt{n}}\right)$, in contrast to the rate of $\widetilde{\Theta}\left(\sqrt{\frac{d}{n}}\right)$ for general Lipschitz functions.

1.2.1.6 Non-convex Optimization

While the structure of convexity enables existence of efficient procedures, it arguably limits modelling. This is best exemplified by the success of deep learning in applications such as computer vision and language processing [RPG+21, Ope23] where the loss functions used are *highly non-convex* with respect to the neural network parameters.

The theory of (efficient) non-convex optimization can be broadly categorized into two: those involving *(a).* relaxed, yet useful, sub-optimality criterion, such as approximating stationary points, and *(b).* structured non-convexity, such as the *Polyak-Łojasiewicz (PL) condition* [Pol63], which despite non-convexity, admits efficient procedures guaranteeing global optimality. We limit to the former in this dissertation.

For a distribution $\mathcal{D}$, an α-**population stationary point** is a w such that $\|\nabla L(w; \mathcal{D})\| \leq \alpha$. Similarly, for a dataset S, an α-**empirical stationary point** is a w such that $\left\|\nabla \widehat{L}(w; S)\right\| \leq \alpha$. In the following, we will refer to the above criteria as **population stationarity** and **empirical stationarity** respectively.

We operate in the unconstrained setting and limit to smooth and Lipschitz loss functions which additionally satisfy *bounded range*, which we recall, means that $\ell(0; z) - \min_w \ell(w; z) \leq L_0$ for all $z \in \mathcal{Z}$.

Unlike the convex case, the (current) landscape of non-convex optimization is more complex. The sample complexity of approximating stationary points is a major

open problem, with the state, as of yet, as follows. The work of [ACD+19] showed that, in *high-enough dimensions*, with n gradient queries, the optimal population (and empirical) stationarity is $\Theta\left(n^{-1/3}\right)$. This is achieved by the *Stochastic Path-Integrated Differential Estimator (SPIDER)* [FLLZ18] procedure. Besides this, via a standard argument based on uniform convergence, it is possible to derive a population stationarity of $\widetilde{O}\left(\sqrt{\frac{d}{n}}\right)$, which is better than the rate of SPIDER in low dimensions. With regard to lower bounds, the work of [FSS18] show a lower bound of $\Omega\left(\frac{G}{\sqrt{n}}\right)$ for smooth (but non-Lipschitz) functions with $d > n$. Our work [ABG+22], presented in Chapter 3, extends this lower bound to smooth and Lipschitz functions for all $d \in \mathbb{N}$.

Finding stationary points in convex settings. While the stationary point criterion is often motivated as a tractable proxy for non-convex optimization, it has also been studied under convexity. The reasons are two-fold: *(a).* this often serves as an *easier* test-bed to settle the gaps in non-convex setting, and *(b).* there exists convex problems where the goal can be framed as finding stationary points, for instance, dual formulations of linearly constrained convex programs [Nes12].

The work of [FSS+19] studied this problem and established the optimal rate on population stationary and its gradient complexity. The optimal rate is shown to be $\widetilde{\Theta}\left(\frac{G}{\sqrt{n}}\right)$ and achieving it necessarily requires $\widetilde{\Omega}\left(\max\left(n, \frac{\sqrt{HL_0 n}}{G}\right)\right)$ gradient queries (in high enough dimensions). This result, interestingly, establishes a separation of sample and gradient complexity, unlike what happens in stochastic convex optimization (under the excess risk criterion).

1.2.2 Uniform Convergence, Stability and Generalization

The analysis of a machine learning procedure often involves investigating its **generalization gap**, which is the difference between its population and empirical risk,

$$\mathbb{E}_{\mathcal{A},S}\left[L(\mathcal{A}(S);\mathcal{D}) - \widehat{L}(\mathcal{A}(S);S)\right].$$

A bound on the generalization gap is useful since it readily gives excess population risk bounds provided that the procedure (approximately) minimizes the empirical risk.

Uniform Convergence. A canonical approach towards it is that of uniform convergence. This amounts to controlling the above deviation between the population and empirical risk, uniformly over an a-priori fixed set in which the algorithm is constrained, or likely, to output. In the following, we define **Rademacher Complexity** and discuss how it relates to uniform convergence.

Definition 5 (Rademacher Complexity)**.** *Let $n \in \mathbb{N}$. Given a class of functions $\mathcal{F} \subseteq \mathbb{R}^{\mathcal{Z}}$, the (worst-case) Rademacher Complexity of $\mathcal{F}$, with respect to n samples, is defined as,*

$$\mathfrak{R}_n(\mathcal{F}) = \sup_{S \in \mathcal{Z}^n} \mathbb{E}_{\sigma_i} \left[\sup_{f \in \mathcal{F}} \frac{1}{n} \sum_{i=1}^{n} \sigma_i f(z_i) \right],$$

where $\sigma_i \in \{-1, 1\}$ are i.i.d. Rademacher random variables i.e. $\mathbb{P}[\sigma_i = 1] = 0.5$.

A key result in learning theory is that Rademacher Complexity characterizes uniform convergence.

Theorem 1 (Theorem 2.1 in [BMR21])**.** *Let $b \geq 0$. Given sets $\mathcal{Z}$ and $\mathcal{W}$, consider the function class $\mathcal{L}_{\mathcal{W},b} = \left\{ w \mapsto \ell(w; z), w \in \mathcal{W} : \sup_{w \in \mathcal{W}, z \in \mathcal{Z}} \ell(w; z) \leq b \right\}$, then*

$$\frac{\mathfrak{R}_n(\mathcal{L}_{\mathcal{W},b})}{2} - \sqrt{\frac{b \log(2)}{2n}} \leq \mathbb{E}_{S \sim \mathcal{D}^n} \left[\sup_{w \in \mathcal{W}} \left(L(w; \mathcal{D}) - \widehat{L}(w; S) \right) \right] \leq 2\mathfrak{R}_n(\mathcal{L}_{\mathcal{W},b}).$$

For the class of G-Lipschitz, constrained (potentially non-convex) GLMs, the Rademacher complexity is $\Theta\left(\frac{G\|\mathcal{X}\|D}{\sqrt{n}}\right)$ [KST08]. This leads to obtaining the optimal rate via ERM. Unfortunately, for general stochastic convex optimization, the uniform convergence rate is necessarily dimension-dependent, $\widetilde{\Omega}\left(GD\left(\frac{d}{n} + \frac{1}{\sqrt{n}}\right)\right)$ [Fel16]. This is in sharp contrast to the dimension-independent upper bound of $O\left(\frac{GD}{\sqrt{n}}\right)$, achieved by the *one-pass SGD* procedure [NY83]. This discrepancy, in part, has motivated alternative approaches to study generalization.

Stability. A prominent and more *algorithm-dependent* approach to generalization is that of algorithmic stability. We define a few important notions below.

Given a dataset S of n items and a data point $z' \in \mathcal{Z}$, let $S^{(i)}$ denote that dataset where the i-th item is replaced by z'.

Definition 6 (Average Stability)**.** *A procedure $\mathcal{A} : \mathcal{Z}^n \to \mathcal{W}$ is α-on-average stable if for all datasets S and S', differing in at most one data point, we have that*

$$\mathop{\mathbb{E}}_{S\sim\mathcal{D}^n, z'\sim\mathcal{D}, z\sim\mathcal{D}, i\sim Unif([n]), \mathcal{A}} \left[\ell(\mathcal{A}(S); z) - \ell(\mathcal{A}(S^{(i)}); z)\right] \leq \alpha$$

Definition 7 (Uniform Stability)**.** *A procedure $\mathcal{A} : \mathcal{Z}^n \to \mathcal{W}$ is α-uniformly stable if for all datasets S and S', differing in at most one data point, for all $z \in \mathcal{Z}$, we have that $\mathbb{E}_{\mathcal{A}} |\ell(\mathcal{A}(S); z) - \ell(\mathcal{A}(S'); z)| \leq \alpha$.*

Definition 8 (Uniform Argument Stability)**.** *A procedure $\mathcal{A} : \mathcal{Z}^n \to \mathcal{W}$ is α-uniformly argument stable if for all datasets S and S', differing in at most one data point, for all $z \in \mathcal{Z}$, we have that $\mathbb{E}_{\mathcal{A}} \|\mathcal{A}(S) - \mathcal{A}(S')\| \leq \alpha$.*

Note that for Lipschitz losses, uniform argument stability implies uniform stability, which further, in general, implies average stability. A seminal result of [BE02] shows that generalization gap is equal to average stability.

Theorem 2 ([BE02])**.** *For any procedure $\mathcal{A} : \mathcal{Z}^n \to \mathcal{W}$, we have that*

$$\begin{aligned}
&\mathbb{E}_{S\sim\mathcal{D}^n, \mathcal{A}} \left[L(\mathcal{A}(S); \mathcal{D}) - \widehat{L}(\mathcal{A}(S); S)\right] \\
&= \mathop{\mathbb{E}}_{S\sim\mathcal{D}^n, z'\sim\mathcal{D}, z\sim\mathcal{D}, i\sim Unif([n]), \mathcal{A}} \left[\ell(\mathcal{A}(S); z) - \ell(\mathcal{A}(S^{(i)}); z)\right].
\end{aligned}$$

The expectations above include the potential randomness in $\mathcal{A}$. Further, the random variables $\mathcal{A}(S)$ and $\mathcal{A}(S')$ need not involve independent sample of the randomness in $\mathcal{A}$, but can be arbitrarily coupled.

Stability in Convex Optimization. The seminal work of [BE02] showed that ERM for μ-strongly convex and G-Lipschitz losses satisfies $O\left(\frac{G}{\mu n}\right)$-uniform argument stability. This implies that ERM achieves the optimal rate of $\Theta\left(\frac{G^2}{\mu n}\right)$. Further, the argument can be extended to (general) convex, Lipschitz losses by a standard *regularization* based reduction. Specifically, the *regularized ERM* procedure adds a small regularization penalty function $\frac{\lambda}{2}\|w\|^2$ to the empirical risk, and applies ERM. For an appropriately chosen λ, this achieves the optimal rate of $\Theta\left(\frac{GD}{\sqrt{n}}\right)$.

The work of [HRS16] extended the stability theory to gradient descent based procedures. They showed that (various variants of) gradient descent, run on *smooth* Lipschitz, (strongly) convex losses, satisfies uniform argument stability, and consequently achieves the optimal rates. This was subsequently extended to *non-smooth*, Lipschitz, (strongly) convex losses by [BFGT20].

An interesting case is that of convex, smooth and non-negative losses, where uniform argument stability (for regularized ERM and gradient descent) necessarily achieve a worse rate of $\Omega\left(\frac{H\max(L_0,D^2)}{\sqrt{n}}\right)$. However, for regularized ERM, [SST10] showed that an *average stability* based analysis suffices to show that it achieves the optimal rate of $\Theta\left(\frac{\sqrt{HL_0}D}{\sqrt{n}}\right)$.

We give a similar average stability result for gradient descent in Chapter 2, which we detail in Section 1.3.4.

1.2.3 Differential Privacy (DP)

We start with the definition of differential privacy.

Definition 9 ((ϵ,δ)-Differential Privacy)**.** *A procedure $\mathcal{A}$ satisfies (ϵ,δ) differential privacy if for any pair of datasets S and S' differing in one point and any measurable event $\mathcal{E}$ in the range of $\mathcal{A}$ it holds that*

$$\mathbb{P}[\mathcal{A}(S)\in\mathcal{E}] \le e^{\epsilon}\mathbb{P}[\mathcal{A}(S')\in\mathcal{E}] + \delta.$$

The above definition provides privacy in the sense that it ensures that the likelihood of any event remains approximately the same, regardless of whether a user decides to include or exclude their data.

The above definition is sometimes called *approximate* differential privacy, with the $\delta = 0$ case called *pure* differential privacy. We will sometimes write $(\epsilon, 0)$-DP as ϵ-DP. Here, ϵ and δ are called *privacy parameters*, with ϵ being a small constant, and δ, a negligible function in the number of samples (users) n, being the preferred settings for *reasonable* privacy protection.

The condition that datasets S and S' differ in one point is known as *neighbouring relation*, with the datasets S and S' called *neighbours*. Formally, given two equally sized multi-sets S and S', we define the neighbouring relation $S \sim_n S'$ if the symmetric difference between them, $S \Delta S' = 2$.

1.2.3.1 Basic Differentially Private Mechanisms and Properties

We review some basic results on differential privacy which serve as building blocks in differentially private algorithm design.

Differentially Private Query Release. A query is a function which takes as input the dataset, and outputs some *statistic*. The problem of differentially private query release is to find a DP procedure for evaluating the query on the dataset, the result of which is accurate, with respect to some error criterion.

We introduce two basic mechanisms for DP query release. Towards it, we first define the **ℓ_p-sensitivity** of a query, which simply is the difference of query evaluations, in ℓ_p norm, uniformly over all neighbouring datasets.

Definition 10. *For $p \in [1, \infty)$, the ℓ_p-sensitivity of a function $q : \mathcal{Z}^n \to \mathbb{R}^d$, is defined*

as,

$$\Delta_p(q) = \sup_{S \sim_n S'} \|q(S) - q(S')\|_p$$

We now state the **Laplace** and **Gaussian mechanism** and their guarantees.

Theorem 3 (Laplace mechanism [DMNS06, DR+14])**.** *The Laplace Mechanism $\mathcal{A}$, is defined as, $\mathcal{A}(S) = q(S) + [\xi_1, \xi_2, \ldots, \xi_d]^\top$, where $\xi_i \sim$ Laplace $\left(0, \frac{\Delta_1(q)}{\epsilon}\right)$ i.i.d.* [2] *he Laplace mechanism $\mathcal{A}$ satisfies ϵ-DP. Further, its error $\|\mathcal{A}(S) - q(S)\|_1 \leq \frac{2\Delta_1(q)(d+\log(1/\beta))}{\epsilon}$, with probability at least $1-\beta$.*

Theorem 4 (Gaussian mechanism [DMNS06, DR+14])**.** *Let $\epsilon \leq \log(1/\delta), \delta \in (0,1)$. The Gaussian Mechanism $\mathcal{A}$, is $\mathcal{A}(S) = q(S) + [\xi_1, \xi_2, \ldots, \xi_d]^\top$, where $\xi_i \sim \mathcal{N}\left(0, \frac{16\Delta_2^2(q)\log(1/\delta)}{\epsilon^2}\right)$ i.i.d.*[3] *Gaussian mechanism $\mathcal{A}$ satisfies ϵ-DP. Further, its error $\|\mathcal{A}(S) - q(S)\|_2 \leq \frac{8\Delta_2(q)\sqrt{\log(1/\delta)}\left(\sqrt{d}+\sqrt{\log(1/\beta)}\right)}{\epsilon}$, with probability at least $1-\beta$.*

Differentially Private Selection. The selection problem consists of a list of *candidates* $\mathcal{C} = \{c_1, c_2, \ldots, c_k\}$, a dataset $S \in \mathcal{Z}^n$, and a *score function* $s : \mathcal{Z}^n \times \mathcal{C} \to \mathbb{R}$, which evaluates the score of a candidate on the dataset. The problem is design a DP procedure to select the candidate with a large score.

A generic procedure to this problem is the **Exponential mechanism** [MT07]. It involves constructing a probability distribution by exponential tilting the uniform distribution over the candidates based on their scores. To describe it, we first define the sensitivity of the score function as follows,

$$\Delta_\infty = \sup_{c \in \mathcal{C}} \sup_{S \sim_n S'} |s(c; S) - s(c; S')| .$$

The description and guarantee of exponential mechanism is given below.

[2] Laplace (μ, σ) is defined by the probability density function $f_{\text{Laplace}(\mu,\sigma)}(x) \propto \exp\left(-\frac{|x-\mu|}{\sigma}\right)$.

[3] $\mathcal{N}(\mu, \sigma)$ is defined by the probability density function $f_{\mathcal{N}(\mu,\sigma)}(x) \propto \exp\left(-\frac{(x-\mu)^2}{2\sigma^2}\right)$.

Theorem 5 (Exponential Mechanism [MT07, DR+14]). *The Exponential Mechanism samples and outputs $\widehat{c}$ with probability $p(c) \propto \exp\left(-\frac{2\epsilon s(c;S)}{\Delta_\infty}\right)$. It satisfies ϵ-DP. Further, with probability at least $1-\beta$,*

$$s(\widehat{c}; S) \geq \max_c s(c; S) - \frac{2\Delta_\infty \ln\left(|C|/\beta\right)}{\epsilon}.$$

Exponential mechanism can be generalized beyond finite sets, by similarly defining an exponentially titled density with respect to an appropriate base measure. We omit this generalization for brevity.

Composition. Composition guarantees allows us to track the evolution of *overall privacy parameters* upon multiple queries made to a dataset. This is a crucial tool in differential private algorithm design, as it facilitates development of procedures built upon basic DP primitives, such as query release and selection. We present the composition theorem below.

Theorem 6. *[DNPR10, KOV15] Let $\epsilon, \delta > 0$, and let $\mathcal{A} = (\mathcal{A}_1, \mathcal{A}_2, \ldots, \mathcal{A}_k)$ be a sequence of (ϵ, δ)-DP procedures, potentially chosen sequentially and adaptively. Then, for any $\delta' > 0$, $\mathcal{A}$ satisfies $(\widetilde{\epsilon}, \widetilde{\delta})$-DP, where*

$$\widetilde{\epsilon} = \min\left\{k\epsilon, 2\epsilon\sqrt{k\log\left(1/\delta'\right)} + k\epsilon\frac{e^\epsilon - 1}{e^\epsilon + 1}\right\}, \text{ and}$$
$$\widetilde{\delta} = k\delta + \delta'.$$

Results with only the first and second term in the minimum for $\widetilde{\epsilon}$ above are often called *basic composition*, and *advanced composition* respectively.

Secrecy of the Sample/ Privacy Amplification by Sub-sampling. If a DP procedure is run on a random sub-sample of the dataset, and the identity of the sub-sample is kept secret, then the overall privacy guarantee is amplified. The formal result is presented below.

Theorem 7. *[KLN+08] Let $m, n \in \mathbb{N}$. Let $\mathcal{A} : \mathcal{Z}^m \to \mathcal{R}$ be an (ϵ, δ)-DP procedure. Then, the procedure $\mathcal{A} : \mathcal{Z}^m \to \mathcal{R}$, which first sub-samples m out of n items from its input dataset, and then applies $\mathcal{A}'$ on it, satisfies $\left(\frac{(e^\epsilon - 1)m}{n}, \frac{m\delta}{n}\right)$-DP.*

1.2.3.2 Differentially Private Optimization

Differentially private optimization has been extensively studied in the last decade. In the following, we limit our discussion to results under approximate DP, which is the notion we consider in this dissertation.

Convex setting. For the constrained, Lipschitz, convex ERM setting, the work of [BST14, ACG+16a] established the optimal expected excess empirical risk of $\Theta\left(\frac{GD\sqrt{d\log(1/\delta)}}{n\epsilon}\right)$ under (ϵ, δ)-DP. Moreover, under μ-strong convexity, [BST14, ACG+16a] showed that the optimal rate is $\Theta\left(\frac{G^2 d\log(1/\delta)}{\mu n^2\epsilon^2}\right)$. These were extended to the (harder) SCO setting in [BFTGT19], yielding optimal rates of $\Theta\left(GD\left(\frac{1}{\sqrt{n}} + \frac{\sqrt{d\log(1/\delta)}}{n\epsilon}\right)\right)$ and $\Theta\left(\frac{G^2}{\mu}\left(\frac{1}{n} + \frac{d\log(1/\delta)}{(n\epsilon)^2}\right)\right)$ for Lipschitz, convex and strongly convex settings respectively.

The setting of constrained, convex, smooth and non-negative losses under DP is considered in Chapter 2 of this dissertation – we detail the results in Section 1.3.4

Convex GLMs. The above rates show that, unlike the non-private setting, an explicit dimension dependence is unavoidable under DP. One can hope that perhaps under additional structure, for instance that of GLMs, it can potentially be alleviated. Unfortunately, as in the non-private setting, the loss function in the *hard instance*, $\ell(w; z) = G\langle w, z\rangle$, is a GLM [BST14]. However, interestingly enough, it turns out that the hardness stems from the *constrained* nature of the setting which requires the algorithm to output in the constrained set $\mathcal{W}$.

The work of [JT14] proposed a DP procedure for *unconstrained* Lipschitz, convex

GLM which achieved a *dimension-independent* rate of $O\left(G\|\mathcal{X}\|\,\|w^*\|\left(\frac{1}{\sqrt{n}}+\frac{1}{\sqrt{n\epsilon}}\right)\right)$ on excess population risk. Herein, w^* is *a* population risk minimizer, and $\|w^*\|$ is assumed to be known. Further, a more *fine-grained rate* of $O\left(G\|\mathcal{X}\|\,\|w^*\|\left(\frac{1}{\sqrt{n}}+\frac{\mathbb{E}_X[\sqrt{\text{rank}(X)}]}{n\epsilon}\right)\right)$ was established by [SSTT20], where $\text{rank}(X)$ is rank of the feature vectors in the training set. Since $\text{rank}(X) \leq \min(n, d)$, this rate is, in the worst-case, the same as that of [JT14].

Several questions remained open here, which our work, presented in Chapter 5, addresses. We give a detailed description of contributions in Section 1.3.1.

Non-Euclidean DP-SCO. Akin to the non-private setting, the non-Euclidean SCO problem has been studied under DP in the works of [AFKT21, BGN21]. In the ℓ_p/ℓ_q setting, these works proposed (ϵ, δ)-DP algorithms with expected excess population risk of $\widetilde{O}\left(G_p D_q\left(\frac{1}{\sqrt{n}}+\frac{\sqrt{d}}{n\epsilon}\right)\right)$ and $\widetilde{O}\left(G_p D_q\left(\frac{d^{1/2-1/p}}{\sqrt{n}}+\frac{d^{1-1/p}}{n\epsilon}\right)\right)$ for $p \in (1, 2]$ and $p > 2$ respectively. Further, the above rate for the $p \in (1, 2]$ regime is shown to be near-optimal by [BGN21].

The question about optimality of the rate for the $p > 2$ regime is tackled in our work, presented in Chapter 2 – we discuss our results in Section 1.3.4.

Non-convex Optimization. We review the most related works in DP non-convex optimization problem where the goal is approximate stationary points – we treat regularity parameters such Lipschitzness and smoothness as constants. The work of [WYX17], under the smoothness and Lipschitzness assumption, proposed an (ϵ, δ)-DP procedure with an excess empirical stationarity of $\widetilde{O}\left(\left(\frac{\sqrt{d}}{n\epsilon}\right)^{1/2}\right)$. This rate was subsequently reproduced by a number of works, albeit with different techniques [ZZMW17, WX19, WCX19a]. For population stationary, the work of [WX19] proposed an (ϵ, δ)-DP procedure with a rate of $O\left(\sqrt{\frac{d}{n}}+\left(\frac{\sqrt{d}}{n\epsilon}\right)^{1/2}\right)$ via a uniform convergence based analysis. Besides this, the work of [ZCH+20] obtained a rate of

$O\left(\sqrt{d}\epsilon + \left(\frac{\sqrt{d}}{n\epsilon}\right)^{1/2}\right)$, which, for any ϵ, is $\Omega\left(\left(\frac{d}{n}\right)^{1/3}\right)$.

Our work, presented in Chapter 2, makes several contributions to this problem – we elaborate on them in Section 1.3.2.

1.2.4 Machine Unlearning

In this section, we setup the notation and formally define what it means to unlearn. Recall that $\mathcal{Z}$ and $\mathcal{W}$ are the data and model space respectively. Further, let $\mathcal{M}$ denote the *meta-data* space, which, as we will see, will correspond to *additional information* a learning algorithm saves to aid unlearning. We consider a learning algorithm as a map $\mathbf{A} : \mathcal{Z}^* \to \mathcal{W} \times \mathcal{M}$ and an unlearning algorithm as a map $\mathbf{U} : \mathcal{W} \times \mathcal{M} \times \mathcal{Z} \to \mathcal{W} \times \mathcal{M}$. We use $\mathcal{A}$ and $\mathcal{U}$ to denote the first output (which belongs to $\mathcal{W}$) of $\mathbf{A}$ and $\mathbf{U}$ respectively.

The definition of exact unlearning requires that the entire state after unlearning be indistinguishable from the state obtained if the learning algorithm were applied to the dataset without the deleted point.

Definition 11 (Exact unlearning)**.** *A procedure* $(\mathbf{A}, \mathbf{U})$ *satisfies exact unlearning if for all datasets* S*, all* $z \in \mathcal{Z}$*, and for all measurable events* $\mathcal{E} \subseteq \mathcal{W} \times \mathcal{M}$*, we have,*

$$\mathbb{P}\left(\mathbf{A}\left(S \backslash \{z\}\right) \in \mathcal{E}\right) = \mathbb{P}\left(\mathbf{U}\left(\mathbf{A}(S), z\right) \in \mathcal{E}\right)$$

The problem of machine unlearning concerns with the design of learning and unlearning algorithms $\mathbf{A}$ and $\mathbf{U}$ respectively, with the following properties.

1. It satisfies the (exact) unlearning criterion.
2. The runtime of unlearning (and learning) algorithm is *non-trivially small*.
3. The outputs of learning and unlearning algorithm are *non-trivially accurate*.

We note that all the above three criteria are required to eliminate trivialities. For instance, a straightforward way to comply with the requirement of exact unlearning

is to recompute (or *retrain*, in machine learning jargon). This, however, violates the second requirement of *efficient* unlearning algorithms - herein and afterwards, we will use the word "efficient", in the context of unlearning, to mean that the unlearning runtime is smaller than recompute time.

Most of the prior work falls into two categories: *(a).* unlearning in *structured* problems, such as linear regression and clustering, where the proposed unlearning algorithms carefully leverage this structure for efficiency (such as [GGVZ19]), and *(b). approximate unlearning criterion*, a popular one inspired from differential privacy, which facilitates transfer to algorithmic techniques (such as [NRSM21a, SAKS21]).

In this dissertation, we consider exact unlearning in stochastic convex optimization and convex empirical risk minimization. Despite the minimal structure in the setup and the strictness of exact unlearning, we show that it is still possible to obtain non-trivial results in this regard.

1.2.4.1 Preliminaries on Optimal Transport and Couplings

We present a brief primer on optimal transport and couplings, which form the core of our techniques. We first define a **coupling**, also known as **transport plan**, between two probability distributions.

Definition 12 (Coupling)**.** *A coupling between two probability distributions P and Q over a common measurable space $\mathcal{U}$, is a joint distribution π over $\mathcal{U} \times \mathcal{U}$ such that the marginal distributions along the projections $(p, q) \mapsto p$ and $(p, q) \mapsto q$ are P and Q respectively.*

We use the notation $\Pi(P, Q)$ to denote the set of couplings between P and Q. Further, for a measurable set $\mathcal{E} \subseteq \mathcal{U}$, we use the notation $P(E)$ to denote its measure under P i.e. $P(\mathcal{E}) = \mathbb{P}_{p \sim P} \{p \in \mathcal{E}\}$.

The optimal transport problem is formulated as follows (see [Vil09] for details).

Given probability distributions P and Q over a common measurable space $\mathcal{U}$, and a *cost function* $c : \mathcal{U} \times \mathcal{U} \to \mathbb{R}$, the *Kantorovich's formulation* of optimal transport, seeks a transport plan π which minimizes the expected cost, i.e. $\inf_{\pi \in \Pi(P,Q)} \mathbb{E}_{(p,q)\sim\pi} c(p,q)$.

In our applications, we limit to cost function being the **discrete (or disagreement) metric**, $c(p,q) = \mathbb{1}\{p \neq q\}$. In the case, the optimal transport cost has a simple explicit form. To describe it, we define **Total Variation distance** below.

Definition 13 (Total Variation (TV) distance)**.** *The Total Variation distance between two probability distributions P and Q defined over a common measurable space $\mathcal{U}$ is defined as,*

$$TV(P,Q) = \sup_{measurable\ \mathcal{E}\subseteq\mathcal{U}} |P(\mathcal{E}) - Q(\mathcal{E})| = \frac{1}{2} \|\phi_P - \phi_Q\|_1 .$$

where the second equality holds if both distributions have probability density functions, with respect to a base measure which are denoted by ϕ_P an ϕ_Q respectively.

The connection between Total Variation distance and optimal transport is that, under the discrete metric, is that optimal transport cost between two distributions is **equal** to the Total Variation distance between them (Theorem 2.12 in [DH12]).

$$\mathrm{TV}(P,Q) = \inf_{\pi\in\Pi(P,Q)} \mathbb{E}_{(p,q)\sim\pi} \mathbb{1}(p \neq q) = \inf_{\pi\in\Pi(P,Q)} \mathbb{P}_{(p,q)\sim\pi}(p \neq q)$$

Further, a coupling, witnessing the "inf" above, always exists (but is not necessarily unique), and is referred to as a *maximal coupling*. This result is known as the **maximal coupling characterization** of Total Variation distance, or the **coupling lemma**.

In our application to machine unlearning, we will be interested in explicit constructions of (near) maximal couplings. Towards it, a key primitive that we will utilize, is the technique of **Reflection Coupling** [LR+86]. This involves sampling $p \sim P$, and performing a *rejection sampling* step, with respect to Q. If it results in *accept*, we output p as the sample for Q (as well). Otherwise, we obtain a sample for Q by *reflecting* p about the mid-point of the means of P and Q.

Formally, consider two probability distributions P and Q with means μ_P and μ_Q, and probability density functions ϕ_P and ϕ_Q, respectively. The reflection coupling construction is describe below.

1. Sample $p \sim P$.
2. Sample $u \sim \text{Unif}([0, \phi_P(p)])$.
3. If $u \leq \phi_Q(p)$, **Accept**: $q = p$.
4. Else, **Reflect**: $q = \mathsf{Reflect}(\mu_P, \mu_Q, p)$, defined as,

$$\mathsf{Reflect}(\mu_P, \mu_Q, p) = \mu_P - \mu_Q + p.$$

5. Output (p, q).

Under suitable assumptions on the probability distributions P and Q (see Lemma 26 for details), the above output (p, q) is a sample from a maximal coupling of P and Q.

Our contributions for this setting is presented in Section 1.3.3.

1.3 Contributions

In this section, we detail our contributions, using notation and context provided in the preceding Section 1.2. We organize our contributions as follows: *(a).* Differentially Private Convex GLMs, *(b).* Differentially Private Non-convex Optimization, *(c).* Machine Unlearning, and *(d).* Auxiliary Results.

1.3.1 Differentially Private Convex GLMs

Our work, presented in Chapter 2, makes progress towards understanding the complexity of learning convex GLMs under differential privacy – see Section 1.2.3.2 for a

Notation	Description
w	Model parameter vector
d	Dimensionality of w
$\mathcal{W} \subseteq \mathbb{R}^d$	(Constrained) set where w lies in
D	Upper bound on $\|w\|$ for $w \in \mathcal{W}$, if constrained
z	Data point
$\mathcal{Z}$	Set where z lies in
x	Feature vector (in supervised learning)
y	Label or response (in supervised learning)
$\mathcal{X} \subseteq \mathbb{R}^d$	Set where x lies in
$\mathcal{Y} \subseteq \mathbb{R}$	Set where y lies in
$\|\mathcal{X}\|$	Upper Bound on $\|x\|$ for $x \in \mathcal{X}$
$\|\mathcal{Y}\|$	Upper Bound on $\|y\|$ for $y \in \mathcal{Y}$
$\ell : \mathcal{W} \times \mathcal{Z} \to \mathbb{R}$	Loss function
$\phi_y : \mathbb{R} \to \mathbb{R}$	GLM link function, $\ell(w; (x, y) = \phi_y(\langle w, x \rangle)$
$\mathcal{D}$	Probability distribution over $\mathcal{Z}$
n	Number of data points
$S \in \mathcal{Z}^n$	Given (training) data set
$L(w; \mathcal{D}) = \mathbb{E}_{z \sim \mathcal{D}}[\ell(w; z)]$	Population risk of w w.r.t. $\mathcal{D}$
$\widehat{L}(w; S) = n^{-1} \sum_{i=1}^n \ell(w; z_i)$	Empirical risk of w w.r.t. S
G	Lipschitz parameter of $w \mapsto \ell(w; z)$ for all $z \in \mathcal{Z}$
H	Smoothness parameter of $w \mapsto \ell(w; z)$ for all $z \in \mathcal{Z}$
L_0	Bounded range parameter, $\sup_{z \in \mathcal{Z}}(\ell(0; z) - \min_{w \in \mathcal{W}} \ell(w; z)) \leq L_0$
ϵ, δ	Differential privacy parameters
ρ	Total variation stability parameter

Table 1-I. Notation, their definition and meaning, used in this dissertation.

background on existing results. In the following, for ease of notation, we treat regularity parameters such as Liphschitzness, smoothness, and $\|\mathcal{X}\|$ and $\|\mathcal{Y}\|$ as constants.

Optimality. The first question is whether the rate obtained in the prior works of [JT14, SSTT20] is optimal, or can be improved. We show that the optimal rate is $\widetilde{\Theta}\left(\|w^*\|\left(\frac{1}{\sqrt{n}}+\min\left(\frac{\mathbb{E}_X[\sqrt{\mathrm{rank}(X)}]}{n\epsilon},\frac{1}{\sqrt{n\epsilon}}\right)\right)\right)$, which improves over the prior works in the small ϵ (i.e. the *high privacy*) regime.

Non-negative, Smooth, Convex GLMs. In the unconstrained setting, several popular GLM loss functions, for instance the squared loss in linear regression, are non-Lipschitz. A natural question is where dimension-independent rates can be obtained here. We answer it positively showing that for non-negative, smooth, convex GLMs, we can achieve a rate of $\widetilde{O}\left(\frac{\|w^*\|}{\sqrt{n}}+\min\left\{\frac{\|w^*\|^2}{(n\epsilon)^{2/3}},\frac{\sqrt{d}\|w^*\|^2}{n\epsilon}\right\}\right)$. We also give a corresponding lower bound of $\widetilde{\Omega}\left(\frac{1}{\sqrt{n}}+\min\left\{\frac{\|w^*\|^{4/3}}{(n\epsilon)^{2/3}},\frac{\sqrt{d}\|w^*\|}{n\epsilon}\right\}\right)$, which shows that our rate is tight up-to dependence on $\|w^*\|$.

Linear Regression. The prototypical example of non-negative, smooth loss, is the squared loss in linear regression, which has been studied on its own, under DP, in prior works [Wan18, CWZ21]. Interestingly, our aforementioned upper and lower bounds apply to this special case, thus allowing us to establish the near-optimal rate for differentially private (distribution-free) linear regression.

Adaptivity to $\|w^*\|$. The prior works of [JT14, SSTT20] assume knowledge of the $\|w^*\|$ for their guarantees, which is, arguably, unreasonable. Our procedures, on the other hand, are adaptive to the knowledge $\|w^*\|$, yet achieving the above rate, upto poly-logarithmic factors in problem parameters.

Relation to the constrained setting. Our work also sheds light on the seeming discrepancies between the dimension-dependent and dimension-independent rates in the constrained and unconstrained settings respectively. We show that in fact these settings are intimately connected. Specifically, we show that high-dimensional, unconstrained, convex GLMs can be reduced to *low-dimensional* constrained SCO problems, in an efficient *black-box* manner. The resulting *low-dimension*, determined by problem parameters (such as smoothness H, number of samples n and privacy parameters ϵ and δ), essentially controls the complexity of the problem, in the same way as the *ambient dimension* d does, in the constrained setting.

1.3.2 Differentially Private Non-convex Optimization

Our work, presented in Chapter 2, makes several contributions, improving over existing works – see Section 1.2.3.2 for prior results. We provide details below.

Non-convex Empirical Stationarity. We propose an (ϵ, δ)-DP procedure with achieves an improved rate of $O\left(\left(\frac{\sqrt{L_0 H G}\sqrt{d\log(1/\delta)}}{n\epsilon}\right)^{2/3} + \frac{G\sqrt{d\log(1/\delta)}}{n\epsilon}\right)$ on expected empirical stationarity. Ignoring regularity parameters, our rate of $O\left(\left(\frac{\sqrt{d\log(1/\delta)}}{n\epsilon}\right)^{2/3}\right)$ improves upon the $\widetilde{O}\left(\left(\frac{\sqrt{d}}{n\epsilon}\right)^{1/2}\right)$ rate obtained in multiple prior works.

Non-convex Population Stationarity. In the statistical setting, our proposed procedure achieves a rate of $\widetilde{O}\left(G\left(1 + HL_0\right)\left(\frac{1}{n^{1/3}} + \left(\frac{\sqrt{d}}{n\epsilon}\right)^{1/2}\right)\right)$ on expected population stationarity , in linear time. This improves upon the aforementioned rates. Further, the *non-private rate* of $O\left(\frac{1}{n^{1/3}}\right)$ matches the optimal rate that can be obtained in linear runtime (in high dimensions) [ACD+19].

Lower Bounds. We establish a lower bound of $\Omega\left(\frac{G\sqrt{d\log(1/\delta)}}{n\epsilon}\right)$ on expected empirical stationarity of (ϵ, δ)-DP procedures. This lower bound holds for all values of

smoothness $H > 0$, demonstrating that our upper bounds are tight for small H.

Convex Setting. Given the apparent gap in our upper and lower bounds, a natural question is whether it can be resolved under the additional assumption of convexity. This would help identify whether the additional complexity is a manifestation of non-convexity or the criterion of stationarity (as opposed to that of excess risk). We resolve this question by constructing a DP procedure which attains a rate of $O\left(\frac{G\sqrt{d\log(1/\delta)}}{n\epsilon}\right)$ on expected empirical stationarity and $O\left(G\left(\frac{1}{\sqrt{n}} + \frac{\sqrt{d\log(1/\delta)}}{n\epsilon}\right)\right)$ on expected population stationarity, both matching their corresponding lower bounds.

1.3.3 Machine Unlearning

We list our contributions on machine unlearning, presented in full in Chapters 4 and 5 below – see Section 1.2.4 for context. As before, in the following, we will treat regularity parameters, such as smoothness and Lipschitzness, as constants.

Convex ERM. For smooth, Lipschitz convex losses, our proposed algorithm has expected excess empirical risk of $O\left(\min\left(\frac{1}{\sqrt{\rho n}}, \left(\frac{\sqrt{d}}{n\rho}\right)^{4/5}\right)\right)$, with expected unlearning runtime $= O(\rho \cdot \text{Training time})$, where training time refers to the optimal gradient complexity of the learning algorithm for the stated accuracy.

Stochastic Convex Optimization. For smooth, Lipschitz SCO, our proposed algorithm achieve an expected excess population risk of $\widetilde{O}\left(\frac{1}{\sqrt{n}} + \frac{\sqrt{d}}{n\rho}\right)$, with expected unlearning runtime $= \widetilde{O}(\rho n)$, for linear-time learning algorithm. Similarly, in the non-smooth setting, we achieve an expected excess population risk of $\widetilde{O}\left(\frac{1}{\sqrt{n}} + \left(\frac{\sqrt{d}}{n\rho}\right)^{1/2}\right)$, with expected unlearning runtime $= \widetilde{O}(\rho n)$.

Convex GLMs. For GLMs, we get dimension-independent rates. Specifically, we get $\widetilde{O}\left(\frac{1}{\sqrt{n}} + \frac{1}{(n\rho)^{2/3}}\right)$ and $\widetilde{O}\left(\frac{1}{\sqrt{n}} + \frac{1}{(n\rho)^{1/2}}\right)$ rates with expected unlearning runtime $=$

$\widetilde{O}(\rho n)$ for smooth and non-smooth settings respectively. The first case is particularly interesting, since it shows that in such settings (which includes problems like logistic regression), it is possible to achieve the near-optimal rate of $\widetilde{O}\left(\frac{1}{\sqrt{n}}\right)$ with *sub-linear* unlearning time of $\widetilde{O}(n^{2/3})$ with a linear time learning algorithm.

Algorithmic Stability. We identify a generic strategy for the unlearning problem, based on a notion of algorithmic stability, called *Total Variation (TV) stability*. This comprises of designing an accurate TV stable learning algorithm, and a corresponding efficient unlearning algorithm which constructs a (near) *maximal coupling* of the the outputs on original and updated datasets. Our results show that this strategy can be realized in settings such as SCO.

Adaptive Query Release and Streaming setting. Our techniques are general enough to enable design of unlearning algorithms corresponding to *iterative learning algorithms*, which perform, what we call, *adaptive query release*, from structured query classes. This allows us to apply our results beyond (known) optimization algorithms – for instance, *Lloyd's algorithm* for *k-means clustering*. We further generalize our results beyond a single unlearning request to that of *dynamic streams*, consisting of a sequence to *edits* (insertions and deletions).

1.3.4 Auxiliary Results

Besides the core results, our works also produce several novel findings, either as intermediate steps, or via extension of proposed techniques. We believe these could be of independent interest, and list them below.

Stability of Gradient Descent. In our work, presented in Chapter 2, we show, via an *average stability* based analysis, that the gradient descent method (and not just regularized ERM, considered in [SST10]) achieves the optimal rate of $\Theta\left(\frac{\sqrt{HL_0}D}{\sqrt{n}}\right)$.

Non-Euclidean DP-SCO Lower Bounds. In our work, presented in Chapter 2, we consider the Non-Euclidean ℓ_p/ℓ_q setting DP-SCO setting. We give a lower bound of $\Omega\left(G_q D_p \min\left(\frac{1}{(n\epsilon)^{1/p}}, \frac{d^{(p-1)/p}}{n\epsilon}\right)\right)$ on excess empirical (and population) risk of any (ϵ, δ)-DP procedure. For $p = \infty$ and $p \geq 2, d \leq n\epsilon$, this matches the best known upper bounds from [BGN21].

Population Stationarity Lower Bound. We give a lower bound of $\Omega\left(\frac{G}{\sqrt{n}}\right)$ on expected population stationary for smooth G-Lipschitz functions in *all dimensions*. This extends the prior lower bound of [FSS+19], in which the loss function in the hard instance is non-Lipschitz and requires $d \geq n$.

Constrained Non-negative, Smooth, Convex setting, under DP. This setting was studied in our work, the results of which are presented in Chapter 2. For the ERM setting, we proposed an (ϵ, δ)-DP procedure with expected excess empirical risk of $O\left(\frac{\sqrt{H}D\max\left(\sqrt{H}D, \sqrt{L_0}\right)\sqrt{d\log(1/\delta)}}{n\epsilon}\right)$. We also gave a lower bound of $\Omega\left(\min\left(\frac{\sqrt{HL_0}D\sqrt{d}}{n\epsilon}, \frac{(\sqrt{H}D)^{3/2}L_0^{1/3}}{(n\epsilon)^{2/3}}\right)\right)$, which establishes tightness of our rate, in the regime $L_0 \geq \sqrt{H}D$ and small $d \leq (n\epsilon)^{2/3}$. In the population risk setting, our upper bound is $O\left(\frac{\sqrt{HL_0}D}{\sqrt{n}} + \frac{\sqrt{H}D\max\left(\sqrt{H}D, \sqrt{L_0}\right)\sqrt{d\log(1/\delta)}}{n\epsilon}\right)$ – here, the additional *non-private term* $\frac{\sqrt{HL_0}D}{\sqrt{n}}$ matches a known lower bound [Sha15].

Chapter 2

Differentially Private Generalized Linear Models

Summary

In this chapter, we study the problem of (ϵ, δ)-differentially private learning of linear predictors with convex losses. We provide results for two subclasses of loss functions. The first case is when the loss is smooth and non-negative but not necessarily Lipschitz (such as the squared loss in linear regression). For this case, we establish an upper bound on the excess population risk of $\widetilde{O}\left(\frac{\|w^*\|}{\sqrt{n}} + \min\left\{\frac{\|w^*\|^2}{(n\epsilon)^{2/3}}, \frac{\sqrt{d}\|w^*\|^2}{n\epsilon}\right\}\right)$, where n is the number of samples, d is the dimension of the problem, and w^* is the minimizer of the population risk. Apart from the dependence on $\|w^*\|$, our bound is essentially tight in all parameters. In particular, we show a lower bound of $\widetilde{\Omega}\left(\frac{1}{\sqrt{n}} + \min\left\{\frac{\|w^*\|^{4/3}}{(n\epsilon)^{2/3}}, \frac{\sqrt{d}\|w^*\|}{n\epsilon}\right\}\right)$. We also revisit the previously studied case of Lipschitz losses [SSTT20]. For this case, we close the gap in the existing work and show that the optimal rate (up to polylog factors) is $\Theta\left(\frac{\|w^*\|}{\sqrt{n}} + \min\left\{\frac{\|w^*\|}{\sqrt{n\epsilon}}, \frac{\sqrt{\text{rank}}\|w^*\|}{n\epsilon}\right\}\right)$, where rank is the expected rank of the design matrix. This improves over existing work in the high privacy regime. Finally, our algorithms involve a private model selection approach that we develop to enable attaining the stated rates without a-priori knowledge of $\|w^*\|$.

		H-Smooth, Non-negative				G-Lipschitz	
		$\sqrt{H}\,\|w^*\|\,\|\mathcal{X}\| \le \|\mathcal{Y}\|$		$\sqrt{H}\,\|w^*\|\,\|\mathcal{X}\| > \|\mathcal{Y}\|$			
		$d \le \left(\frac{\|w^*\|\sqrt{H}\|\mathcal{X}\|n\epsilon}{\|\mathcal{Y}\|}\right)^{2/3}$	$d > \left(\frac{\|w^*\|\sqrt{H}\|\mathcal{X}\|n\epsilon}{\|\mathcal{Y}\|}\right)^{2/3}$	$d \le (n\epsilon)^{2/3}$	$d > (n\epsilon)^{2/3}$	rank $\le n\epsilon$	rank $> n\epsilon$
DP	UB	$\frac{\sqrt{H}\|w^*\|\|\mathcal{X}\|\|\mathcal{Y}\|\sqrt{d}}{n\epsilon}$ Theorem 8	$\frac{\left(\sqrt{H}\|w^*\|\|\mathcal{X}\|\right)^{4/3}\|\mathcal{Y}\|^{2/3}}{(n\epsilon)^{2/3}}$ Theorem 9	$\frac{H\|w^*\|^2\|\mathcal{X}\|^2\sqrt{d}}{n\epsilon}$ Theorem 8	$\frac{H\|w^*\|^2\|\mathcal{X}\|^2}{(n\epsilon)^{2/3}}$ Theorem 9, 10	$\frac{G\|w^*\|\|\mathcal{X}\|\sqrt{\text{rank}}}{n\epsilon}$ [SSTT20]	$\frac{G\|w^*\|\|\mathcal{X}\|}{\sqrt{n\epsilon}}$ Theorem 32, 12
	LB	Tight Theorem 11	Tight Theorem 11	$\frac{\sqrt{H}\|w^*\|\|\mathcal{X}\|\|\mathcal{Y}\|\sqrt{d}}{n\epsilon}$ Theorem 11	$\frac{\left(\sqrt{H}\|w^*\|\|\mathcal{X}\|\right)^{4/3}\|\mathcal{Y}\|^{2/3}}{(n\epsilon)^{2/3}}$ Theorem 11	Tight Theorem 13	Tight Theorem 13
Non-private	UB	$\frac{\sqrt{H}\|\mathcal{X}\|\|\mathcal{Y}\|\|w^*\|}{\sqrt{n}}$ [SST10]				$\frac{G\|w^*\|\|\mathcal{X}\|}{\sqrt{n}}$ [NY83]	
	LB	$\frac{\min\{\|\mathcal{Y}\|,\|w^*\|\|\mathcal{X}\|\}}{\sqrt{n}} + \min\left\{\|\mathcal{Y}\|^2, \frac{H\|w^*\|^2\|\mathcal{X}\|^2+d\|\mathcal{Y}\|^2}{n}, \frac{\sqrt{H}\|w^*\|\|\mathcal{Y}\|\|\mathcal{X}\|}{\sqrt{n}}\right\}$ [SST10, Sha15]				Tight [NY83]	

Table 2-I. Summary of Rates. Parameters: d: dimension, n: sample size, H: smoothness parameter, w^*: minimum norm population risk minimizer, $\|\mathcal{X}\|$: bound on feature vectors, $\|\mathcal{Y}\|$: bound on loss at zero, G: Lipschitzness parameter, rank: expected rank of the design matrix, ϵ: privacy parameter (δ factors omitted). The actual private excess risk bounds are the sum of the expressions shown in the DP rows and their non-private counterparts. Details on non-private lower bounds in Appendix A.5.

2.1 Introduction

In this chapter, we study one of the most basic machine learning problems under differential privacy: learning convex generalized linear models (GLM).

2.1.1 Contributions

Smooth, Non-negative GLMs. Our primary contribution is a new and nearly optimal rate for the problem of differentially private learning of smooth GLMs. In this setting, we focus on characterizing the excess risk in terms of n, d, ϵ and $\|w^*\|$. Specifically, we show that it is possible to achieve a rate of $\widetilde{O}\left(\frac{\|w^*\|}{\sqrt{n}} + \min\left\{\frac{\|w^*\|^2}{(n\epsilon)^{2/3}}, \frac{\sqrt{d}\|w^*\|^2}{n\epsilon}\right\}\right)$ on the excess population risk. Our new rates exhibit an interesting low/high dimensional transition at $d \approx (\|w^*\|\, n\epsilon)^{2/3}$. First, in the low dimensional regime, we develop a novel analysis of noisy gradient decent (GD) inspired by techniques from [SST10]. In particular, we show that Noisy GD gives an improved rate for non-negative smooth functions (not necessarily GLMs). This is based on an average stability analysis of Noisy GD. As we elaborate in Section 2.3.1, a straightforward application of uniform stability leads to sub-optimal bounds and hence a new analysis

is required. We note in passing that this upper bound works for (unconstrained) DP-SCO with smooth (non-Lipschitz) losses, which is of independent interest. For the high dimensional regime, we perform random projections of the data (specifically, the Johnson-Lindenstrauss transform) for dimensionality reduction, roughly reducing the problem to its low dimensional counterpart. We also develop a lower bound for the excess risk under DP of $\widetilde{\Omega}\left(\min\left\{\frac{\|w^*\|^{4/3}}{(n\epsilon)^{2/3}}, \frac{\sqrt{d}\|w^*\|}{n\epsilon}\right\}\right)$. We note that non-privately a lower bound of $\widetilde{\Omega}\left(\frac{1}{\sqrt{n}} + \min\left\{\frac{d+\|w^*\|^2}{n}, \frac{\|w^*\|}{\sqrt{n}}\right\}\right)$ is known on the excess population risk [SST10, Sha15]. We note that these private and non-private lower bounds imply that our bound is optimal up to factors of $\|w^*\|$ (see Table 2-I).

Lipschitz GLMs. For the Lipschitz case, we close a subtle but important gap in existing rates. In this setting, it has been shown that one can characterize the excess risk in terms of the expected rank of the design matrix, rank, instead of d [SSTT20]. In this setting, the best known rate was $\widetilde{O}\left(\frac{\|w^*\|}{\sqrt{n}} + \frac{\sqrt{\text{rank}}\|w^*\|}{n\epsilon}\right)$. We show an improved rate of $\widetilde{O}\left(\frac{\|w^*\|}{\sqrt{n}} + \min\left\{\frac{\|w^*\|}{\sqrt{n\epsilon}}, \frac{\sqrt{\text{rank}}\|w^*\|}{n\epsilon}\right\}\right)$. This improves in the high privacy regime where $\epsilon \leq \frac{\text{rank}}{n}$. In fact, the upper bound $O\left(\frac{\|w^*\|}{\sqrt{n\epsilon}}\right)$ for this rate can be obtained with only minor adjustments to the regularization method of [JT14]. Our second contribution in this setting is extending the lower bound of [SSTT20] to hold for all values of $\|w^*\| > 0$ and rank $\in [n]$. This is in contrast to the original lower bound which only holds for problem instances where $\|w^*\|^2 = \text{rank}$ and rank $\in [n\epsilon]$.

Model Selection. As part of our methods, we develop a differentially private model selection approach which eliminates the need for a-priori knowledge of $\|w^*\|$. Although such methods are well established in the non-private case, (see e.g. [SSBD14]), in the private case no such methods have been established. Our method, as in the non-private case, performs a grid search over estimates of $\|w^*\|$ and picks the best model based on the loss. However, in the private setting we must account for the fact the the loss

evaluation must be privatized. This is non-trivial in the non-Lipschitz smooth case as the loss at a point w may grow quadratically with $\|w\|$.

Lower Bounds for Non-Euclidean DP-SCO. Our lower bound construction generalizes to Non-Euclidean ℓ_p/ℓ_q variants of DP-SCO with Lipschitz convex losses [BGN21]. Herein, we assume that the loss function is G_q-Lipschitz with respect to ℓ_q norm, and radius of the constraint set is bounded in ℓ_p norm by D_p. For this setting, we give a lower bound of $\Omega\left(G_q D_p \min\left(\frac{1}{(n\epsilon)^{1/p}}, \frac{d^{(p-1)/p}}{n\epsilon}\right)\right)$ on excess empirical/population risk of any (potentially unconstrained) (ϵ, δ)-DP algorithm; see Corollary 5 in Appendix A.2.4 for a formal statement and proof. For $p = \infty$ and $p \geq 2, d \leq n\epsilon$, this matches the best known upper bounds in [BGN21].

Non-private settings. As by-products, we give the following new results for the non-private setting. For details on the parameters used below we refer to Table 2-I.

1. We show that gradient descent, when run on convex non-negative $\widetilde{H}$ smooth functions (not necessarily GLMs), it achieves the optimal rate of $O\left(\frac{\sqrt{\widetilde{H}}\|w^*\|\|\mathcal{Y}\|}{\sqrt{n}}\right)$ (see Corollary 2). This is done via an average-stability analysis of gradient descent. This result is interesting as it also shows GD only needs n iterations, which is known not to work for non-smooth SCO [BFGT20, ACKL21, AKL21].

2. In Section A.4, we give a procedure to boost the confidence of algorithms for risk minimization with convex non-negative $\widetilde{H}$ smooth functions (not necessarily GLMs). The standard boosting analysis based on Hoeffding's inequality does not give a bound with a linear dependence on the parameters $(\|w^*\|, \|\mathcal{X}\|, \|\mathcal{Y}\|)$, and hence a tighter analysis is required.

2.1.2 Techniques

Upper bounds. We give two algorithms for both the smooth and Lispschitz cases. The first method is simple and has two main steps. First, optimize the regularized empirical risk over the constraint set $\left\{w \in \mathbb{R}^d : \|w\| \leq D\right\}$ for some $D \geq \|w^*\|$. Then output a perturbation of the regularized minimizer with Gaussian noise (which is not requried to be in the constraint set). This method is akin to that of [JT14] with the modification that the regularized minimizer is constrained to a ball. We elaborate on this key difference shortly.

The second method is based on dimensionality reduction. We use smaller dimensional data-oblivious embeddings of the feature vectors. A linear JL transform suffices to give embeddings with the required properties. We then run a constrained DP-SCO method (Noisy GD) in the embedded space, and use the transpose of the JL transform to get a d dimensional model. In this method, the embedding dimension required is roughly the threshold on dimension at which the rates switch from dimension dependent to independent bounds. We also remark that [NêUZ20] applied a similar technique to provide dimension independent classification error guarantees for privately learning support vector machines under hard margin conditions.

We note that a crucial part in all of these methods is the use of constrained optimization as a subroutine, where the constraint set is a ball of radius $\|w^*\|$. This is in stark contrast to the Lipschitz case where existing methods such as those presented by [JT14, SSTT20] rely on the fact that projection is not required. In the smooth case however, constrained optimization helps ensure that the norm of the gradient is roughly bounded by the diameter of the constraint set. We note that in the high dimensional regime, the property that the *final* output of the algorithm can have large norm is still crucial to the success of our algorithms.

Lower bounds. For our lower bounds in the smooth case we rely on the connection between stability and privacy. Specifically, we will utilize a lemma from [CH12] which bounds the accuracy of one-dimensional differentially private algorithms. We then combine this with packing arguments to obtain stronger lower bounds for high dimensional problems. For the Lipschitz case, we adapt the method of [SSTT20].

2.1.3 Related Work

We briefly discuss the most related works. In the Lipschitz-convex setting, tight rates are known for both the empirical and population risk [BST14, BFTT19]. Specifically, it was shown that in the constrained setting, dependence on the dimension in the form of $\Omega\left(\frac{\sqrt{d}}{n\epsilon}\right)$ is unavoidable even for generalized linear models . In contrast, in the unconstrained setting, it has been shown that dimension independent rates are possible for GLMs [JT14]. In this setting, assuming prior knowledge of $\|w^*\|$, the best known rate is $O\left(\frac{\|w^*\|}{\sqrt{n}} + \frac{\|w^*\|\sqrt{\text{rank}}}{n\epsilon}\right)$ [SSTT20], where rank is the expected rank of the design matrix. However, without prior knowledge of $\|w^*\|$, these methods exhibit quadratic dependence on $\|w^*\|$. Furthermore, these results crucially rely on the assumption that the loss is Lipschitz to bound the sensitivity. Although gradient clipping has been proposed to remedy this problem [SSTT20, CWH20], it is known that the solution obtained by clipping may not coincide with the one of the original model.

Without Lipschitzness, work on differentially private GLMs has largely been limited to linear regression [Wan18, CWZ21]. Here, dimension independent rates have only been obtained under certain sparsity assumptions.

More generally, smooth non-negative losses have been studied in the non-private setting by [SST10], where it was shown such functions can obtain risk guarantees with linear dependence on the minimizer norm (as in the Lipschitz case). This work also established a lower bound of $\Omega\left(\frac{1}{\sqrt{n}}\right)$ on the excess population risk for this class of loss functions. [Sha15] additionally establishes a lower bound of $\Omega\left\{\min\left\{\frac{\|w^*\|^2+d}{n}, \frac{\|w^*\|}{\sqrt{n}}\right\}\right\}$

on the excess population risk by way of linear regression[1].

2.2 Preliminaries

In the following we detail some additional concepts needed for the presentation of this chapter. For our lower bounds, we will make use of the following Lemma from [CH12].

Lemma 1. *Let $\mathcal{Z}$ be a data domain and let S and S' be two datasets each in $\mathcal{Z}^n$ that differ in at most Δ entries, and let $\mathcal{A} : \mathcal{Z}^n \to \mathbb{R}$ be any (ϵ, δ)-DP algorithm. For all $\tau \in \mathbb{R}$, if $\Delta \leq \frac{\log(1/2\gamma)}{\epsilon}$ and $\delta \leq \frac{1}{16}(1 - e^{-\epsilon})$, then $\mathbb{E}\left[|\mathcal{A}(S) - \tau| + |\mathcal{A}(S') - \tau'|\right] \geq \frac{1}{4}|\tau - \tau'|$.*

Furhter, we introduce the Johnson-Lindenstrauss (JL) transform to perform random projections.

Definition 14 ((α, β)-JL property)**.** *A distribution over matrices $\mathbb{R}^{k \times d}$ satisfies (α, β)-JL property if for any $u, v \in \mathbb{R}^d$, $\mathbb{P}\left[|\langle \Phi u, \Phi v \rangle - \langle u, v \rangle| > \alpha \|u\| \|v\|\right] \leq \beta$.*

It is well known that several such "data-oblivious" (i.e. independent of u,v) distributions exist with $k = O\left(\frac{\log(1/\beta)}{\alpha^2}\right)$ [Nel11]. We note that the JL property is typically described as approximation of norms (or distances), but it is easy to deduce the above dot product preservation property from it; for completeness we give this as Lemma 13. Finally, we use $\Phi\mathcal{D}$ to denote the push-forward measure of the distribution $\mathcal{D}$ under the map $\Phi : (x, y) \mapsto (\Phi x, y)$. Similarly, given a data set $S = \{(x_i, y_i)\}_i$, we define $\Phi S := \{(\Phi x_i, y_i)_i\}$

2.3 Smooth Non-negative GLMs

For smooth non-negative GLMs, we present new upper and lower bounds on the excess risk. For our upper bounds, we here assume that the algorithm is given access to

[1]The [SST10] bound assumes $\|\mathcal{Y}\|, H, \|\mathcal{X}\| = \Omega(1)$. The bounds of [Sha15] were originally stated for the constrained setting, but can easily be converted. More details in Appendix A.5.

some upper bound on $\|w^*\|$, that we denote by D. We later show in Section 2.5 how to obtain such a rate without prior knowledge of $\|w^*\|$. We also emphasize that the privacy of these algorithms holds regardless of whether or not $D \geq \|w^*\|$.

2.3.1 Upper Bounds

Before presenting our algorithms, we highlight some key ideas underlying all our methods. A crucial property of non-negative smooth loss functions which allows one to bound sensitivity, and thus ensure privacy, is the self-bounding property (e.g. Sec. 12.1.3 in [SSBD14]), which states that for an $\widetilde{H}$-smooth non-negative function f and $u \in \text{dom(f)}$, $\|\nabla f(u)\| \leq \sqrt{4\widetilde{H} f(u)}$.

This property implies that the gradient grows at most linearly with $\|w\|$. More precisely, we have the following.

Lemma 2. *Let ℓ be an H-smooth non-negative GLM. Then for any $w \in \mathcal{B}(D)$ and $(x, y) \in (\mathcal{X} \times \mathcal{Y})$ we have $\|\nabla \ell(w, (x, y))\| \leq 2\|\mathcal{Y}\|\sqrt{H}\|\mathcal{X}\| + 2HD\|\mathcal{X}\|^2$.*

In order to leverage this property, all our algorithms in this setting utilize constrained optimization as a subroutine, where the constraint set is $\mathcal{B}(D)$ and the Lipschitz constant is $G = 2\|\mathcal{Y}\|\sqrt{H}\|\mathcal{X}\| + 2HD\|\mathcal{X}\|^2$. This, in conjunction with the self-bounding property ensures reasonable bounds on sensitivity. In turn, this allows us to ensure privacy without excessive levels of noise.

Finally, we note that our upper bounds for smooth GLMs distinguish low and high dimensional regimes, transitioning at $d = \min\left(\left(\frac{D\sqrt{H}\|\mathcal{X}\|}{\|\mathcal{Y}\|}\right)^{\frac{2}{3}}, 1\right)(n\epsilon)^{\frac{2}{3}}$.

2.3.1.1 Low dimensional regime

We start with the low dimensional setting where we use techniques developed for constrained DP-SCO for Lipschitz losses (not necessarily GLMs). Existing private algorithms for DP-SCO (e.g., [BFTT19, FKT20a]) lead to excess risk bounds that

scale with $\frac{HD^2}{\sqrt{n}}$. On the other hand, the optimal non-private rate [SST10] scales with $\frac{\sqrt{H}D}{\sqrt{n}}$, which may indicate that the private rate implied by the known methods is sub-optimal. We show that this gap can be closed by a novel analysis of private GD.

A standard proof of excess risk of (noisy) gradient descent for smooth convex functions is based on uniform stability [HRS16, BFTT19]. However, this still leads to sub-optimal rates. Hence we turn to an average stability based analysis of GD, yielding the following result.

Theorem 8. *Let ℓ be a non-negative, convex, $\widetilde{H}$-smooth loss function, bounded at zero by $\|\mathcal{Y}\|^2$. Let $D, n > 0$, $n_0 = \frac{\widetilde{H}D^2}{\|\mathcal{Y}\|^2}$, $G = 2\|\mathcal{Y}\|\sqrt{\widetilde{H}} + 2\widetilde{H}D$. Then, for any $\epsilon, \delta > 0$, Algorithm 1 invoked with $\mathcal{W} = \mathcal{B}(D)$, $T = n$, $\sigma^2 = \frac{8G^2T\log(1/\delta)}{n^2\epsilon^2}$, $\eta = \min\left(\frac{D}{\sqrt{T}\max\left(\sqrt{\widetilde{H}}\|\mathcal{Y}\|, \sigma\sqrt{d}\right)}, \frac{1}{4\widetilde{H}}\right)$ is (ϵ, δ)-differentially private. Further, given a dataset S of $n \geq n_0$ i.i.d samples from an unknown distribution $\mathcal{D}$, the excess risk of output of Algorithm 1 is bounded as,*

$$\mathbb{E}[L(\widehat{w}; \mathcal{D}) - L(w^*; \mathcal{D})] = O\left(\frac{\sqrt{\widetilde{H}}D\|\mathcal{Y}\|}{\sqrt{n}} + \frac{GD\sqrt{d\log(1/\delta)}}{n\epsilon}\right).$$

We note that since H-smooth GLMs satisfy the Theorem condition with parameter $\widetilde{H} \leq H\|\mathcal{X}\|^2$, we obtain results for GLMs as a direct corollary.

Non-private Risk Bound. As a corollary (see Corollary 2 in Appendix A.1.3.1), with no privacy constraint, the above result (setting $\epsilon \to \infty$ and $\delta = 1$) shows that gradient descent achieves the optimal excess risk bound, previously shown to be achievable by regularized ERM and one-pass SGD [SST10]. The lower bound, n_0, simply means that the trivial solution "zero" has larger excess risk.

Proof sketch of Theorem 8. The privacy proof simply follows from [BST14] since the loss function is G-Lipschitz in the constraint set. For utility, we first introduce two concepts used in the proof. Let S be a dataset of n i.i.d samples $\{(x_i, y_i)\}_{i=1}^n$, $S^{(i)}$

be the dataset where the i-th data point is replaced by an i.i.d. point (x', y'). Let $\mathcal{A}$ be an algorithm which takes a dataset as input and outputs $\mathcal{A}(S)$. The **average argument stability** of $\mathcal{A}$, denoted as $\mathrm{AAS}(\mathcal{A})$ is defined as

$$\mathrm{AAS}(\mathcal{A})^2 = \mathbb{E}_{S,i,(x',y')}\|\mathcal{A}(S) - \mathcal{A}(S^{(i)})\|^2.$$

The **average regret** of gradient descent (Algorithm 1) with iterates $\{w_t\}_{t=1}^T$ is

$$\varepsilon_{\mathsf{reg}}(\mathcal{A}; w^*) = \frac{1}{T}\sum_{j=1}^{T}\mathbb{E}[\widehat{L}(w_j; S) - \widehat{L}(w^*; S)].$$

The key arguments are as follows: we first bound the generalization gap, or on-average stability, in terms of average argument stability and excess empirical risk (Lemma 5). We then bound average argument stability in terms of average regret (Lemma 6). Finally, in Lemma 7, we provide bounds on excess empirical risk and average regret of noisy gradient descent. Substituting these in the excess risk decomposition gives the claimed bound. The full proof with the above lemmas is deferred to Appendix A.1.3.1.

Algorithm 1 Noisy GD

Input: Dataset S, loss function ℓ, constraint set $\mathcal{W}$, steps T, learning rate η, noise scale σ^2

1: $w_0 = 0$
2: **for** $t = 1$ to $T - 1$ **do**
3: $\quad \xi \sim \mathcal{N}(0, \sigma^2\mathbb{I})$
4: $\quad w_{t+1} = \Pi_{\mathcal{W}}\left(w_t - \eta\left(\widehat{L}(w_t; S) + \xi\right)\right)$
where $\Pi_{\mathcal{W}}$ is the Euclidean projection on to $\mathcal{W}$
5: **end for**

Output: $\widehat{w} = \frac{1}{T}\sum_{t=1}^{T} w_j$

2.3.1.2 High dimensional regime

In the high dimensional setting, we present two techniques.

JL method. In the JL method, Algorithm 2, we use a data-oblivious JL map Φ to embed all feature vectors in dataset S to $k < d$ dimensions. Let dataset

Algorithm 2 JL Method

Input: Dataset $S = \{(x_1, y_1), ..., (x_n, y_n)\}$, loss function ℓ, k, D, η, T, σ^2

1: Sample JL matrix $\Phi \in \mathbb{R}^{k \times d}$

2: $\widetilde{S} = \{(\Phi x_1, y_1), (\Phi x_2, y_2), \ldots, (\Phi x_n, y_n)\}$

3: $\widetilde{w} = \text{NoisyGD}(\widetilde{S}, \ell, \mathcal{B}(2D), T, \eta, \sigma^2)$

Output: $\widehat{w} = \Phi^\top \widetilde{w}$

$\widetilde{S} = \{(\Phi x_i, y_i)\}_{i=1}^n$. We then run projected Noisy GD method (Algorithm 1) on the loss with dataset $\widetilde{S}$ and the diameter of the constraint set as $2D$. Finally, we map the returned output $\widetilde{w}$ back to d dimensions using $\Phi^\top$ to get $\widehat{w} = \Phi^\top \widetilde{w}$. We note that no projection is performed on $\widehat{w}$ and thus the output may have large norm due to re-scaling induced by $\Phi^\top$.

Theorem 9. *Let* $k = O\left(\frac{D\sqrt{H}\|\mathcal{X}\| \log(2n/\delta)n\epsilon}{\|\mathcal{Y}\|\|\mathcal{X}\| + \sqrt{H}D\|\mathcal{X}\|^2}\right)^{2/3}$, $\mathcal{W} = \mathcal{B}(D)$, $T = n$, $\sigma^2 = \frac{8G^2 T \log(1/\delta)}{n^2\epsilon^2}$, $\eta = \min\left(\frac{D}{\sqrt{T}\max\left(\sqrt{H}\|\mathcal{X}\|\|\mathcal{Y}\|, \sigma\sqrt{d}\right)}, \frac{1}{4H\|\mathcal{X}\|^2}\right)$ *and* $n_0 = \frac{HD^2\|\mathcal{X}\|^2}{\|\mathcal{Y}\|^2}$. *Algorithm 2 satisfies* (ϵ, δ)*-differential privacy. Given a dataset* S *of* $n \geq n_0$ *i.i.d samples, of the output* $\widehat{w}$ *is bounded by*

$$\mathbb{E}[L(\widehat{w}; \mathcal{D}) - L(w^*; \mathcal{D})] \leq \widetilde{O}\Bigg(\frac{\sqrt{H}D\,\|\mathcal{X}\|\,\|\mathcal{Y}\|}{\sqrt{n}} + \frac{\left(\sqrt{H}D\,\|\mathcal{X}\|\sqrt{\|\mathcal{Y}\|}\right)^{\frac{4}{3}} + \left(\sqrt{H}D\,\|\mathcal{X}\|\right)^2}{(n\epsilon)^{\frac{2}{3}}}\Bigg).$$

Proof sketch of Theorem 9. From the JL property, with $k = O(\log(n/\delta)/\alpha^2)$, w.h.p. norms of all feature vectors and w^*, as well as inner products between are preserved upto an α tolerance (see Definition 14). The preservation of norms of feature vectors implies the gradient norms are preserved, and thus privacy guarantee of sub-routine, Algorithm 1, suffices to establish DP. For the utility proof, from our analysis of Noisy GD (Theorem 8) and using the JL property, the excess risk of $\widehat{w}$

Algorithm 3 Regularized Output Perturbation

Input: Dataset $S = \{(x_1, y_1), ..., (x_n, y_n)\}$, loss function ℓ, λ, D, σ^2

1: $\widetilde{w} = \underset{w \in \mathcal{B}(D)}{\arg\min} \left\{ L(w; S) + \frac{\lambda}{2} \|w\|^2 \right\}$

2: $\xi \sim \mathcal{N}(0, \sigma^2 \mathrm{I}_d)$

Output: $\widehat{w} = \widetilde{w} + \xi$

under $\mathcal{D}$ w.r.t. the risk of Φw^* under $\Phi\mathcal{D}$ is bounded as,

$$\mathbb{E}[L(\widehat{w}; \mathcal{D}) - L(\Phi w^*; \Phi\mathcal{D})] \\ \leq O\left(\frac{\sqrt{H}\|\mathcal{X}\| D \|\mathcal{Y}\|}{\sqrt{n}} + \frac{\left(\sqrt{H}\|\mathcal{X}\|\|\mathcal{Y}\| + H\|\mathcal{X}\|^2 D^2\right) D\sqrt{k \log(1/\delta)}}{n\epsilon} \right).$$

From smoothness and JL property, the, "JL error" is:

$$\mathbb{E}[L(\Phi w^*; \Phi\mathcal{D})] - L(w^*; \mathcal{D})] \leq \widetilde{O}\left(\frac{HD^2\|\mathcal{X}\|^2}{k} \right).$$

The above is optimized for the value of k prescribed in Theorem 9, substituting which gives the claimed bound.

Constrained regularized ERM + output perturbation. Our second technique is *constrained* regularized ERM with output perturbation (Algorithm 3). A similar technique for the Lipschitz case was seen in [JT14], however we note that the addition of the constraint set $\mathcal{B}(D)$ is crucial in bounding the sensitivity in the smooth case.

Theorem 10. *Let* $n_0 = \frac{HD^2\|\mathcal{X}\|^2}{\|\mathcal{Y}\|^2}$. *Then Algorithm 3 run with* $\sigma^2 = O\left(\frac{\left(\|\mathcal{Y}\|^2 + H^2D^2\|\mathcal{X}\|^4\right)\log(1/\delta)}{\lambda^2 n^2 \epsilon^2} \right)$ *and* $\lambda = \left(\frac{\left(\|\mathcal{Y}\| + HD\|\mathcal{X}\|^2\right)\sqrt{H}\|\mathcal{X}\|}{Dn\epsilon} \right)^{2/3} (\log(1/\delta))^{1/3}$ *satisfies* (ϵ, δ)*-differential privacy. Given a dataset* S *of* $n \geq n_0$ *i.i.d samples, the excess risk of its output* $\widehat{w}$ *is bounded as*

$$\mathbb{E}[L(\widehat{w}; \mathcal{D}) - L(w^*; \mathcal{D})] \leq \widetilde{O}\left(\frac{\sqrt{H} D \|\mathcal{X}\| \|\mathcal{Y}\| \|\mathcal{Y}\|^2}{\sqrt{n}} \right. \\ \left. + \frac{\left(\sqrt{H} D \|\mathcal{X}\|\right)^{4/3} \|\mathcal{Y}\|^{2/3} + \left(\sqrt{H} D \|\mathcal{X}\|\right)^2}{(n\epsilon)^{2/3}} \right).$$

We note that we can use the same technique in the low dimensional setting too, yielding a rate of $\frac{\sqrt{H}D\|\mathcal{X}\|\|\mathcal{Y}\|+\|\mathcal{Y}\|^2}{\sqrt{n}} + \frac{GD\sqrt{d}}{n\epsilon}$. However, in contrast to Theorem 8 and 9, these results have an additional $\frac{\|\mathcal{Y}\|^2}{\sqrt{n}}$ term (in both regimes). Thus, in the regime when $\|\mathcal{Y}\| \leq \sqrt{H}D\|\mathcal{X}\|$, the two upper bounds are of the same order.

Proof sketch of Theorem 10. The privacy proof follows the Gaussian mechanism guarantee together with the fact that the ℓ_2-sensitivity of constrained regularized ERM is $O\left(\frac{G}{\lambda n}\right)$ [BE02]. For utility, we use the Rademacher complexity based result of [SST10] to bound the generalization error of $\widetilde{w}$. The other term, error from noise, $\mathbb{E}[L(\widehat{w};\mathcal{D}) - L(\widetilde{w};\mathcal{D})] \leq O(H\sigma^2\|\mathcal{X}\|^2)$ from smoothness of GLM. Combining these two and optimizing for λ gives the claimed result.

2.3.2 Lower Bounds

The proof technique for our lower bound relies on the connection between differential privacy and sensitivity shown in [CH12]. The underlying mechanisms behind the proof is also similar in nature to the method seen in [SU17]. We note that although we present the following theorem for empirical risk, a reduction found in [BFTT19] implies the following result holds for population risk as well (up to log factors).

Theorem 11. *Let $\epsilon \in [1/n, 1]$, $\delta \leq \frac{1}{16}(1 - e^{-\epsilon})$. For any (ϵ, δ)-DP algorithm $\mathcal{A}$, there exists a dataset S with empirical minimizer of norm at most D such that the excess empirical risk of $\mathcal{A}$ on S is lower bounded by $\Omega\left\{\min\left\{\|\mathcal{Y}\|^2, \frac{(D\|\mathcal{X}\|)^{4/3}(H\|\mathcal{Y}\|)^{2/3}}{(n\epsilon)^{2/3}}, \frac{\sqrt{d}D\|\mathcal{X}\|\|\mathcal{Y}\|\sqrt{H}}{n\epsilon}\right\}\right\}$.*

The problem instance used in the proof is the squared loss function, and thus this result holds additionally for the more specific case of linear regression. We also note this lower bound implies our upper bound is optimal whenever $D\|\mathcal{X}\| \leq \|\mathcal{Y}\|$, which is a commonly studied regime in linear regression [SST10, KL15]. We here provide a proof sketch and defer the full proof to Appendix A.1.7.

Proof sketch of Theorem 11 Define $\widehat{L}(w;S) = \frac{1}{n}\sum_{(x,y)\in S}(\langle w,x\rangle - y)^2$. Let $d' < \min\{n,d\}$ and $b,p \in [0,1]$ be parameters to be chosen later. For any $\boldsymbol{\sigma} \in \{\pm 1\}^{d'}$, define the dataset $S_{\boldsymbol{\sigma}}$ which consists of the union of d' subdatasets, $S_1,...,S_{d'}$ given as follows. Set $\frac{pn}{d'}$ of the feature vectors in S_j as $\|\mathcal{X}\|e_j$ (the rescaled j'th standard basis vector) and the rest as the zero vector. Set $\frac{pn}{2d}(1+b)$ of the labels as $\boldsymbol{\sigma}_j\|\mathcal{Y}\|$ and $\frac{pn}{2d}(1-b)$ labels as $-\boldsymbol{\sigma}_j\|\mathcal{Y}\|$. Let $w^{\boldsymbol{\sigma}} = \arg\min_{w\in\mathbb{R}^d}\left\{\widehat{L}(w;S_{\boldsymbol{\sigma}})\right\}$ be the ERM minimizer of $\widehat{L}(\cdot;S_{\boldsymbol{\sigma}})$. Following from Lemma 2 of [Sha15] we have that for any $\bar{w}\in\mathbb{R}^d$ that

$$\widehat{L}(\bar{w};S_{\boldsymbol{\sigma}}) - \widehat{L}(w^{\boldsymbol{\sigma}};S_{\boldsymbol{\sigma}}) \geq \frac{p\|\mathcal{X}\|^2}{2d'}\sum_{j=1}^{d'}(\bar{w}_j - w_j^{\boldsymbol{\sigma}})^2.$$

We will now show lower bounds on the per-coordinate error. Consider any $\boldsymbol{\sigma}$ and $\boldsymbol{\sigma}'$ which differ only at index j for some $j\in[d']$. Note that the datasets $S_{\boldsymbol{\sigma}}$ and $S_{\boldsymbol{\sigma}'}$ differ in $\Delta = \frac{pn}{2d'}[(1+b)-(1-b)] = \frac{pbn}{d'}$ points. Let $\tau = w_j^{\boldsymbol{\sigma}} = \frac{\|\mathcal{Y}\|b}{\|\mathcal{X}\|}$ and $\tau' = w_j^{\boldsymbol{\sigma}'} = -\frac{\|\mathcal{Y}\|b}{\|\mathcal{X}\|}$ (i.e. the j components of the empirical minimizers for S and S'_j respectively). Note that $|w_j^{\boldsymbol{\sigma}} - w_j^{\boldsymbol{\sigma}'}| = \frac{2\|\mathcal{Y}\|b}{\|\mathcal{X}\|}$. Setting $d' = \left(\frac{p\|\mathcal{X}\|Dn\epsilon}{\|\mathcal{Y}\|}\right)^{2/3}$ and $b = \left(\frac{\|\mathcal{X}\|D}{\|\mathcal{Y}\|\sqrt{pn\epsilon}}\right)^{2/3}$ ensures $\Delta \leq \frac{1}{\epsilon}$, and thus Lemma 1 can be used to obtain

$$\mathbb{E}\left[|\mathcal{A}(S_{\boldsymbol{\sigma}})_j - w_j^{\boldsymbol{\sigma}}|^2 + |\mathcal{A}(S_{\boldsymbol{\sigma}'})_j - w_j^{\boldsymbol{\sigma}'}|^2\right] \geq \frac{1}{32}\frac{\|\mathcal{Y}\|^2b^2}{\|\mathcal{X}\|^2} = \frac{D^{4/3}\|\mathcal{Y}\|^{2/3}}{32(\|\mathcal{X}\|pn\epsilon)^{2/3}}.$$

One can now show via a packing argument that

$$\sup_{\boldsymbol{\sigma}\in\{\pm1\}^{d'}}\left\{\mathbb{E}\left[\widehat{L}(\mathcal{A}(S_{\boldsymbol{\sigma}});S_{\boldsymbol{\sigma}}) - \widehat{L}(w^{\boldsymbol{\sigma}};S_{\boldsymbol{\sigma}})\right]\right\} \geq \frac{(\|\mathcal{X}\|D)^{4/3}\|\mathcal{Y}\|^{2/3}p^{1/3}}{128(n\epsilon)^{2/3}},$$

The result then follows from setting $p = \min\left\{1, \frac{d^{3/2}\|\mathcal{Y}\|}{D\|\mathcal{X}\|n\epsilon}\right\}$.

2.4 Lipschitz GLMs

In the Lipschitz case, we close a subtle gap in existing rates. We recall that in this setting a more precise characterization in terms of the expected rank of the design matrix is possible (as opposed to using d). The best known upper bound is $\widetilde{O}\left(\frac{\|w^*\|\sqrt{\text{rank}}}{n\epsilon}\right)$ assuming knowledge of $\|w^*\|$. This bound was shown to be optimal when $\epsilon \geq \text{rank}/n$ and $\|w^*\| = \sqrt{\text{rank}}$ [SSTT20].

We first show that in the high privacy regime where $\epsilon \leq \text{rank}/n$, an improved rate is possible. Specifically, we show that in this regime constrained regularized ERM with output perturbation and achieves the optimal rate. In fact, we note that the method of [JT14] (i.e. *unconstrained* regularized ERM with output perturbation), can obtain this rate when $\epsilon = O(1)$ if the regularization parameter is set differently. We present the constrained version in order to leverage Rademacher complexity arguments and provide a slightly cleaner bound that holds for all $\epsilon > 0$.

Theorem 12. *Algorithm 3 run with parameters* $\sigma^2 = \frac{4G^2\|\mathcal{X}\|^2 \log(1/\delta)}{\lambda^2 n^2 \epsilon^2}$ *and* $\lambda = \frac{G\|\mathcal{X}\|(\log(1/\delta))^{1/4}}{D\sqrt{n\epsilon}}$ *satisfies* (ϵ, δ)*-DP. Given a dataset of* n *i.i.d. samples from* $\mathcal{D}$*, its output has excess risk*

$$\mathbb{E}\left[L(\widehat{w}; \mathcal{D}) - L(w^*; \mathcal{D})\right] = \widetilde{O}\left(\frac{GD\|\mathcal{X}\|}{\sqrt{n}} + \frac{GD\|\mathcal{X}\|}{\sqrt{n\epsilon}}\right).$$

We also state and prove a similar bound using the JL technique in Appendix A.2.

Next, we generalize the lower bound of [SSTT20] to show this new bound is optimal for all settings of D, rank, and ϵ. We now show that a modification of the lower bound present in [SSTT20] shows our upper bound is tight. We note that their lower bound only held for problem instances where $\text{rank} = \|w^*\|^2$ and $\epsilon \leq \text{rank}/n$. By contrast, the upper bound $O(\frac{\sqrt{\text{rank}}\|w^*\|}{n\epsilon})$ holds for any values of rank and $\|w^*\|$.

Theorem 13. *Let* $G, \|\mathcal{Y}\|, \|\mathcal{X}\|, D > 0$*,* $\epsilon \leq 1.2$ *and* $\delta \leq \epsilon$*. For any* (ϵ, δ)*-DP algorithm* $\mathcal{A}$*, there exists a* G*-Lipschitz GLM loss bounded at zero by* $\|\mathcal{Y}\|$ *and a distribution* $\mathcal{D}$ *with* $\|w^*\| \leq D$ *such that the output of* $\mathcal{A}$ *on* $S \sim \mathcal{D}^n$ *satisfies*

$$\mathbb{E}\left[L(\mathcal{A}(S); \mathcal{D}) - L(w^*; \mathcal{D})\right] = \Omega\left(GD\|\mathcal{X}\| \min\left(1, \frac{1}{\sqrt{n\epsilon}}, \frac{\sqrt{rank}}{n\epsilon}\right)\right).$$

All proofs for this section are deferred to Appendix A.2.

2.5 Adapting to $\|w^*\|$

Our method for privately adapting to $\|w^*\|$ is given in Algorithm 4. We start by giving a high level overview and defining some necessary preliminaries. The algorithm works in the following manner. First we define a number of "guesses" K for $\|w^*\|$, $D_1, ..., D_K$ where $D_j = 2^j : \forall j \in [K]$. Then given black box access to a DP optimization algorithm, $\mathcal{A}$, Algorithm 4 generates K candidate vectors $w_1, ..., w_K$ using $\mathcal{A}$, training set $S_1 \in (\mathcal{X} \times \mathcal{Y})^{n/2}$, and the guesses $D_1, ..., D_K$. We assume $\mathcal{A}$ satisfies the following accuracy assumption for some confidence parameter $\beta > 0$.

Assumption 1. *There exists a function* $\mathsf{ERR} : \mathbb{R}^+ \mapsto \mathbb{R}^+$ *such that for any* $D \in \mathbb{R}^+$, *whenever* $D \geq \|w^*\|$, *w.p. at least* $1 - \frac{\beta}{4K}$ *under the randomness of* $S_1 \sim \mathcal{D}^{\frac{n}{2}}$ *and* $\mathcal{A}$ *it holds that* $\mathbf{E}\left[L(\mathcal{A}(S_1, D); \mathcal{D}) - L(w^*; \mathcal{D})\right] \leq \mathsf{ERR}(D)$.

After generating the candidate vectors, the goal is to pick guess with the smallest excess population risk in a differentially private manner using a validation set S_2. The following assumption on $\mathcal{A}$ allows us both to ensure the privacy of the model selection algorithm and verify that $\widehat{L}(w_j; S_2)$ provides a tight estimate of $L(w_j; \mathcal{D})$.

Assumption 2. *There exist a function* $\Delta : \mathbb{R}^+ \mapsto \mathbb{R}^+$ *such that for any dataset* $S_2 \in (\mathcal{X} \times \mathcal{Y})^{n/2}$ *and* $D > 0$

$$\mathop{\mathbb{P}}_{\mathcal{A}}[\exists (x, y) \in S_2 : |\ell(\mathcal{A}(S_1, D); (x, y))| \geq \Delta(D)] \leq \frac{\min\{\delta, \beta\}}{4K}$$

Specifically, our strategy will be to use the Generalized Exponential Mechanism, GenExpMech, of [RS16] in conjunction with a penalized score function. Roughly, this score function penalty ensures the looser guarantees on the population loss estimate when D is large do not interfere with the loss estimates at smaller values of D. We provide the relevant details for GenExpMech in Appendix A.3.1. We now state our result.

Algorithm 4 Private Grid Search

Input: Dataset $\mathcal{S} \in (\mathcal{X} \times \mathcal{Y})^n$, grid parameter $K \in \mathbb{R}$, optimization algorithm: $\mathcal{A} : (\mathcal{X} \times \mathcal{Y})^n \times \mathbb{R} \mapsto \mathbb{R}^d$, privacy parameters (ϵ, δ)

1: Partition S into two disjoint sets, S_1 and S_2, of size $\frac{n}{2}$
2: $w_0 = 0$
3: **for** $j \in [K]$ **do**
4: $\quad D_j = 2^j$
5: $\quad w_j = \mathcal{A}(S_1, D_j)$
6: $\quad \widetilde{L}_j = \widehat{L}(w_j; S_2) + \frac{\Delta(D_j)\log(K/\beta)}{n} + \sqrt{\frac{4\|\mathcal{Y}\|^2 \log(K/\beta)}{n}}$
7: **end for**
8: Set j^* as the output of GenExpMech run with privacy parameter $\frac{\epsilon}{2}$, confidence parameter $\frac{\beta}{4}$, and sensitivity/score pairs $(0, \|\mathcal{Y}\|^2), (\Delta(D_1), \widetilde{L}_1)..., (\Delta(D_K), \widetilde{L}_K)$,
9: Output w_{j^*}

Theorem 14. *Let $\ell : \mathbb{R}^d \times (\mathcal{X} \times \mathcal{Y})$ be a smooth non-negative loss function such that $\ell(0, (x, y)) \leq \|\mathcal{Y}\|^2$ for any $x, y \in (\mathcal{X} \times \mathcal{Y})$. Let $\epsilon, \delta, \beta \in [0, 1]$. Let $K > 0$ satisfy $\mathsf{ERR}(2^K) \geq \|\mathcal{Y}\|^2$. Let $\mathcal{A}$ be an $(\frac{\epsilon}{2K}, \frac{\delta}{2K})$-DP algorithm satisfying Assumption 2. Then Algorithm 4 is (ϵ, δ)-DP. Further, if $\mathcal{A}$ satisfies Assumption 1 and $S_1 \sim \mathcal{D}^{n/2}$ then Algorithm 4 outputs $\bar{w}$ s.t. with probability at least $1 - \beta$,*

$$\mathbb{E}\left[L(\bar{w}; \mathcal{D}) - L(w^*; \mathcal{D})\right] \leq \min\Bigg\{\|\mathcal{Y}\|^2, \mathsf{ERR}(2\max\{\|w^*\|, 1\}) + \sqrt{\frac{4\|\mathcal{Y}\|^2 \log(4K/\beta)}{n}} + \frac{5\Delta(2\max\{\|w^*\|, 1\})}{n\epsilon}\Bigg\}.$$

We note that we develop a generic confidence boosting approach to obtain high probability guarantees from our previously described algorithms in Section 2.5, and thus obtaining algorithms which satisfy 1 is straightforward. We provide more details on how our algorithms satisfy Assumption 2 in Appendix A.3.4. The following Theorem details the guarantees implied by this method for output perturbation with boosting (see Theorems 35,37). Full details are in Appendix A.3.3.

Theorem 15. *Let $K, \epsilon, \delta, \beta > 0$ and $\mathcal{A}$ be the algorithm formed by running 3 with boosting and privacy parameters $\epsilon' = \frac{\epsilon}{K}$, $\delta' = \frac{\delta}{K}$. Then there exists a setting of K such that $K = \Theta\left(\log\left(\max\left\{\frac{\|\mathcal{Y}\|\sqrt{n}}{\|\mathcal{X}\|\sqrt{H}}, \frac{\|\mathcal{Y}\|^2(n\epsilon)^{2/3}}{\sqrt{H}\|\mathcal{X}\|^2}\right\}\right)\right)$ and Algorithm 4 run with $\mathcal{A}$ and K is*

(ϵ, δ)-DP and when given $S \sim \mathcal{D}^n$, satisfies the following w.p. at least $1 - \beta$ (letting $D^* = 2 \max\{\|w^*\|, 1\}$)

$$\mathbb{E}\left[L(\widehat{w}; \mathcal{D}) - L(w^*; \mathcal{D})\right] = \widetilde{O}\Bigg(\min\Bigg\{ \|\mathcal{Y}\|^2, \frac{\left(\sqrt{H} D^* \|\mathcal{X}\|\right)^{4/3} \|\mathcal{Y}\|^{2/3} + \left(\sqrt{H} D^* \|\mathcal{X}\|\right)^2}{(n\epsilon)^{2/3}} + \frac{\sqrt{H} D^* \|\mathcal{X}\| \max\{\|\mathcal{Y}\|, 1\} + \|\mathcal{Y}\|^2}{\sqrt{n}} + \frac{\|\mathcal{Y}\|^2 + H(D^* \|\mathcal{X}\|)^2}{n\epsilon} \Bigg\} \Bigg).$$

Confidence Boosting. We give an algorithm to boost the confidence of unconstrained, smooth DP-SCO (with possibly non-Lipschitz losses). We split the dataset S into $m+1$ chunks and run an (ϵ, δ)-DP algorithm over the m chunks to get m models, and then use Report Noisy Max mechanism to select a model with approximately the least empirical risk. We show that this achieves the optimal rate of $\widetilde{O}\left(\frac{\sqrt{H} D \|\mathcal{X}\| \|\mathcal{Y}\|}{\sqrt{n}}\right)$ whereas the previous high probability result of [SST10] had an additional $\widetilde{O}\left(\frac{\|\mathcal{Y}\|^2}{\sqrt{n}}\right)$ term, which was also limited to only GLMs. The key idea is that non-negativity, convexity, smoothness and loss bounded at zero, all together enable strong bounds on the variance of the loss, and consequently give stronger concentration bounds. The details are deferred to Appendix A.4.

2.6 Conclusion

In this chapter, we studied the problem of learning convex generalized linear models (GLM), under differential privacy, in the unconstrained setting. We designed DP procedures with near-optimal dimension-independent rates for two fundamental classes: Lipschitz GLM, and non-negative, smooth GLMs. Further, we proposed an adaptive procedure which attains near-optimal rates without knowledge of the norm of the optimal predictor.

Chapter 3

Differentially Private Non-convex Optimization

Summary

In this chapter, we study the problem of approximating stationary points of Lipschitz and smooth functions under (ε, δ)-differential privacy (DP) in both the finite-sum and stochastic settings. We provide a new efficient algorithm that finds an $\widetilde{O}\Big(\Big[\frac{\sqrt{d}}{n\varepsilon}\Big]^{2/3}\Big)$-stationary point in the finite-sum setting, where n is the number of samples. This improves on the previous best rate of $\widetilde{O}\Big(\Big[\frac{\sqrt{d}}{n\varepsilon}\Big]^{1/2}\Big)$. We also give a new construction that improves over the existing rates in the stochastic optimization setting, where the goal is to find approximate stationary points of the population risk. Our construction finds a $\widetilde{O}\Big(\frac{1}{n^{1/3}} + \Big[\frac{\sqrt{d}}{n\varepsilon}\Big]^{1/2}\Big)$-stationary point of the population risk in time linear in n. Furthermore, under the additional assumption of convexity, we completely characterize the sample complexity of finding stationary points of the population risk (up to polylog factors) and show that the optimal rate on population stationarity is $\widetilde{\Theta}\Big(\frac{1}{\sqrt{n}} + \frac{\sqrt{d}}{n\varepsilon}\Big)$. Finally, we show that our methods can be used to provide dimension-independent rates of $O\Big(\frac{1}{\sqrt{n}} + \min\Big(\Big[\frac{\sqrt{\text{rank}}}{n\varepsilon}\Big]^{2/3}, \frac{1}{(n\varepsilon)^{2/5}}\Big)\Big)$ on population stationarity for Generalized Linear Models (GLM), where rank is the rank of the design matrix, which improves upon the previous best known rate.

3.1 Introduction

In this chapter, we study differentially private non-convex optimization; namely, the task of approximating stationary points. This has been heavily studied in recent years in the non-private setting [FLLZ18, MWCC18, CDHS17, NP06, GL13, ACD+19, FSS+19]. This problem is motivated by the intractability of nonconvex (global) optimization, as well as by a number of settings where stationary points have been shown to be global minima [GLM16, SQW16].

3.1.1 Contributions

In this work, we make progress towards resolving the complexity of approximating stationary points in optimization under the constraint of differential privacy, for both empirical and population risks. A summary of our new results is available in Table 3-I. In what follows, d is the problem dimension, n is the dataset size, and ε, δ are the approximate DP parameters. Our first set of results pertains to the approximation of stationary points in empirical nonconvex optimization (a.k.a. finite-sum optimization setting). In this context, we provide algorithms with rate $O\Big(\Big[\frac{\sqrt{d}}{n\varepsilon}\Big]^{2/3}\Big)$, and oracle complexity[1] $\widetilde{O}\Big(\max\Big\{\Big(\frac{n^5\varepsilon^2}{d}\Big)^{1/3}, \Big(\frac{n\varepsilon}{\sqrt{d}}\Big)^2\Big\}\Big)$. This rate is sharper than the best known for this problem [WYX17].

Next, we focus on the task of approximating stationary points of the population risk. Results for this problem are scarce. We provide the fastest rate up to date for this problem under DP, of $\widetilde{O}\Big(\frac{1}{n^{1/3}} + \Big[\frac{\sqrt{d}}{n\varepsilon}\Big]^{1/2}\Big)$, with an algorithm that moreover has oracle complexity n (i.e., is single-pass). This algorithm is a noisy version of the SPIDER algorithm [FLLZ18], whose gradient estimators are built using a tree-aggregation data structure for prefix-sums [AFKT21].

We continue by investigating stationary points for convex losses and give an

[1] We consider for complexity the first-order oracle model, standard for continuous optimization [NY83].

Setting	Convergence	Our Rate	Previous best-known rate
Non-convex	Empirical	$\left(\frac{\sqrt{d}}{n\epsilon}\right)^{2/3}$ (Thm. 16)	$\left(\frac{\sqrt{d}}{n\epsilon}\right)^{1/2}$ [WYX17]
	Population	$\frac{1}{n^{1/3}} + \left(\frac{\sqrt{d}}{n\epsilon}\right)^{1/2}$ (Thm. 19)	$\sqrt{d}\epsilon + \left(\frac{\sqrt{d}}{n\epsilon}\right)^{1/2}$ [ZCH+20]
Convex	Population	$\frac{1}{\sqrt{n}} + \frac{\sqrt{d}}{n\epsilon}$ (Thm. 20)	None
Non-convex GLM	Empirical	$\left[\frac{\sqrt{\text{rank}}}{n\epsilon}\right]^{2/3} \wedge \frac{1}{(n\epsilon)^{2/5}}$ (Cor. 1)	$\left(\frac{\sqrt{\text{rank}}}{n\epsilon}\right)^{1/2}$ [SSTT21]
	Population	$\frac{1}{\sqrt{n}} + \left[\frac{\sqrt{\text{rank}}}{n\epsilon}\right]^{2/3} \wedge \frac{1}{(n\epsilon)^{2/5}}$ (Cor. 1)	None
Convex GLM	Population	$\frac{1}{\sqrt{n}} + \frac{\sqrt{\text{rank}}}{n\epsilon} \wedge \frac{1}{\sqrt{n\epsilon}}$ (Cor. 1)	None

Table 3-I. Results summary: We omit log factors and function-class parameters. The symbol $\wedge$ stands for minimum of the quantities.

algorithm based on the recursive regularization technique of [AZ18] which achieves the optimal rate of $\widetilde{\Theta}\left(\frac{1}{\sqrt{n}} + \frac{\sqrt{d}}{n\varepsilon}\right)$ on population stationarity. To establish optimality, we give a lower bound of $\Omega\left(\frac{\sqrt{d}}{n\varepsilon}\right)$ on empirical stationarity under DP (Theorem 17) and a non-private lower bound of $\Omega(\frac{1}{\sqrt{n}})$ on population stationarity (Theorem 40). We also give a linear-time method, which achieves the optimal rate when the smoothness parameter is not so large. We conclude the paper showing a black-box reduction that converts any DP method for finding stationary points of smooth and Lipschitz losses into a DP method with *dimension-independent rates* for the case of generalized linear models (GLM). Using our proposed method with Private Spiderboost as the base algorithm yields a rate of $\widetilde{O}\left(\frac{1}{\sqrt{n}} + \min\left(\left[\frac{\sqrt{\text{rank}}}{n\varepsilon}\right]^{2/3}, \frac{1}{(n\varepsilon)^{2/5}}\right)\right)$ on population stationarity. This improves upon the result of [SSTT21] which proposed a method with $\widetilde{O}\left(\left[\frac{\sqrt{\text{rank}}}{n\varepsilon}\right]^{1/2}\right)$ empirical stationarity[2].

3.1.2 Techniques

Our methods combine multiple techniques from optimization and differential privacy in novel ways. The lower bound for the empirical norm of the gradient uses fingerprinting codes to a loss similar to that used for Differentially Private-Empirical Risk Minimization (DP-ERM) [BST14], crafted to work in the unconstrained case. This lower bound

[2]This is the rate obtained after fixing a mistake in the proof of Theorem 4.1 in [SSTT21].

can be extended to the population gradient norm by a known re-sampling argument [BFTGT19]. We also give a non-private lower bound of $\Omega\left(1/\sqrt{n}\right)$ on population stationarity with n samples which holds even in dimension 1, as opposed to previous results [FSS+19].

Efficient algorithms for (both empirical and population) norm of the gradient are derived using noisy versions of variance-reduced stochastic first order methods, which have proved remarkably useful in DP stochastic optimization [AFKT21, BGN21, BGM21]. However, in contrast to previous work which scales noise proportionally to the Lipschitz constant [ZCH+20, ZMLX21] or (in the case of constrained optimization) the diameter of the constraint set [BGM21, BGN21], we observe that the gradient variations between iterates w, w' can be privatized more effectively by scaling the noise proportional to $H \left\|w - w'\right\|$. In the case of the empirical risk, we use a noisy version of SpiderBoost [WJZ+19]. We remark that our methods can achieve comparable rates when applied to similar algorithms such as Spider [FLLZ18] and Storm [CO19], but SpiderBoost allows for a larger learning rate which is considered better in practice. For the population risk, it is worth noting that the empirical norm of the gradient does not translate directly into population gradient guarantees, even if the algorithm in use is uniformly stable [BE02], since this type of guarantee does not enjoy a *stability-implies-generalization* property. Therefore, we opt for single pass methods that combine variance-reduction with tree-aggregation; these techniques are particularly suitable for the classical Spider algorithm [FLLZ18], which is the one we base our method on. For the convex setting, we use recursive regularization [AZ18] which was used to achieve the optimal non-private rate by [FSS+19].

Finally, our method for (non-convex) GLMs uses the Johnson-Lindenstrauss based dimensionality reduction technique similar to [ABG+22], which focused on the convex setting. Moreover, for population stationarity of GLMs, we give a new uniform convergence result of gradients of Lipschitz functions. This guarantee, unlike the prior

work of [FSS18], has only poly-logarithmic dependence on the radius of the constraint set, which is crucial for our analysis.

3.1.3 Related Work

The current work fits within the literature of differentially private optimization, which has primarily focused on the convex case [CMS11, JKT12, KST12, BST14, TTZ14, JT14, TTZ15, BFTGT19, FKT20b, AFKT21, BGN21]. The culmination of this line of work for the convex smooth case showed that optimal rates are achievable in linear time [FKT20b, AFKT21, BGN21]. Our work shows that in the convex case similar rates are achievable for the norm of the gradient: this result is useful, e.g., for dual formulations of linearly constrained convex programs [Nes12], and moreover it has become a problem of independent interest [AZ18, FSS+19]. [3]

Regarding stationary points for nonconvex losses, work in DP is far more recent, and primarily focused on the empirical stationarity [WYX17, ZZMW17, WX19, WCX19b]. Under similar assumptions to ours these works approximate stationary points with rate $\widetilde{O}\Big(\Big[\frac{\sqrt{d}}{n\varepsilon}\Big]^{1/2}\Big)$, which is slower than ours.

Works addressing population guarantees for the norm of the gradient under DP are scarce. [ZCH+20] proposed a noisy gradient method, whose population guarantee is obtained by generalization properties of DP. However, the best guarantee obtainable with their analysis is $O\Big(\Big[\frac{\sqrt{d}}{n\varepsilon}\Big]^{1/2} + \sqrt{d}\varepsilon\Big)$[4]. Note that for any ε this rate is $\Omega\Big([d/n]^{1/3}\Big)$. Under additional assumptions (on the Hessian), [WX19] obtains a rate of $\widetilde{O}(\sqrt{d/(n\varepsilon)})$ by uniform convergence of gradients, which is sharper when ε is constant. By contrast, our rate is much faster than both for $\varepsilon = \Theta(1)$. In particular, in this range, our rates

[3] To provide a specific example, consider the dual of the regularized discrete optimal transport problem, as discussed in [DG23], Section 5.6. If the marginals μ, ν in that model are accessed through i.i.d. samples, then this becomes an SCO problem. Moreover, it is argued in that reference that approximate stationary points provide approximately feasible and optimal transports through duality arguments. Hence, the result is an SCO problem where we require *approximate stationary points.*

[4] [ZCH+20] omits the term $\sqrt{d}\varepsilon$, but this omission is only valid when $\varepsilon < 1/[n\sqrt{d}]^{1/3}$.

are faster than those obtained by uniform convergence, $O(\sqrt{d/n})$ [FSS18]. Moreover, our method runs in time linear in n. On the other hand, in the much more restrictive setting where the loss satisfies the Polyak-Łojasiewicz (PL) inequality, [ZMLX21] provide *population risk* bounds of $\widetilde{O}(d/[n\varepsilon]^2)$ under DP.

The work of [BGM21] studies population guarantees for stationarity in constrained settings, obtaining rates $O\Big(\frac{1}{n^{1/3}} + \Big[\frac{\sqrt{d}}{n\varepsilon}\Big]^{2/5}\Big)$ in linear time. Notice first that these guarantees are based on the Frank-Wolfe gap, making those results incomparable to ours. Despite this fact, their rates are slower than ours.[5] On the other hand, they provide results for (close to nearly) stationary points in constrained/unconstrained settings, for a broader class of *weakly convex losses* (possibly nonsmooth). This result is then more general, but the rate of $O\Big(\frac{1}{n^{1/4}} + \Big[\frac{\sqrt{d}}{n\varepsilon}\Big]^{1/3}\Big)$ is substantially slower than ours, and their algorithm has oracle complexity which is superlinear in n.

The problem of stationary points in (nonprivate) stochastic optimization has drawn major attention recently [GL13, GL16, FLLZ18, AZ18, FSS18, FSS+19, ACD+19]. To the best of our knowledge, no lower bounds for the sample complexity[6] of this problem are known (beyond those known for the convex case [FSS+19]). On the other hand, oracle complexity is by now understood: in high dimensions, for (on average) smooth losses the optimal stochastic oracle complexity rate is $O(1/n^{1/3})$ [ACD+19]. Although this provides some evidence of the sharpness of our results (see Appendix B.2.2), note that these lower bounds require very high dimensional constructions (namely, $d = \Omega(1/\alpha^4)$, where α is the rate), which limits their applicability in the private setting.

[5] We believe our methods can be extended to constrained settings using gradient mapping, a guarantee for which is stronger than for Frank-Wolfe gap [Lan20, Section 7.5.1]. We defer this extension to future work.

[6] Sample complexity is the fundamental limit on the sample size needed, as a function of α, to achieve α stationarity. This is different from the oracle complexity as one is not limited to first-order methods.

Algorithm 5 Private SpiderBoost

Input: Dataset: $S \in \mathcal{Z}^n$, Function: $\ell : \mathbb{R}^d \times \mathcal{Z} \mapsto \mathbb{R}$, Learning Rate: η, Phase Size: q, Batch Sizes b_1, b_2, Privacy Parameters: (ϵ, δ), Iterations: T

$w_0 = 0$

$\sigma_1 = \frac{cG\sqrt{\log(1/\delta)}}{\epsilon} \max\left\{\frac{1}{b_1}, \frac{\sqrt{T}}{\sqrt{q}n}\right\}$, where c is a universal constant.

$\sigma_2 = \frac{cH\sqrt{\log(1/\delta)}}{\epsilon} \max\left\{\frac{1}{b_2}, \frac{\sqrt{T}}{n}\right\}$

$\widehat{\sigma}_2 = \frac{2cG\sqrt{\log(1/\delta)}}{\epsilon} \max\left\{\frac{1}{b_2}, \frac{\sqrt{T}}{n}\right\}$

for $t = 0, \ldots, T$ **do**

  if $\mod(t, q) = 0$ **then**

    Sample batch S_t of size b_1

    Sample $g_t \sim \mathcal{N}(0, \mathbb{I}_d \sigma_1^2)$

    $\nabla_t = \frac{1}{b_1} \sum_{z \in S_t} \nabla \ell(w_t; z) + g_t$

  else

    Sample batch S_t of size b_2

    Sample $g_t \sim \mathcal{N}\left(0, \mathbb{I}_d \min\left\{\sigma_2^2 \left\|w_t - w_{t-1}\right\|^2, \widehat{\sigma}_2^2\right\}\right)$

    $\Delta_t = \frac{1}{b_2} \sum_{x \in S_t} \left[\nabla \ell(w_t; z) - \nabla \ell(w_{t-1}; z)\right] + g_t$

    $\nabla_t = \nabla_{t-1} + \Delta_t$

  end if

  $w_{t+1} = w_t - \eta \nabla_t$

end for

return $\widehat{w}$ uniformly at random from $w_1, \ldots, w_T$

3.2 Stationary Points of Empirical Risk

3.2.1 Efficient Algorithm with Faster Rate

The algorithm for our upper bound is a noisy version of the SpiderBoost algorithm [WJZ+19][7]. The algorithm works by running a series of phases of length q. Each phase starts with a minibatch estimate of the gradient, and subsequent gradient estimates within the phase are then computed by adding an estimate of the gradient variation. The key to the analysis is to bound the error in the gradient estimate at each iteration. Towards this end, we have the following generalization of the [WJZ+19, Lemma 1], which follows directly from [FLLZ18, Proposition 1].

[7] SpiderBoost itself is essentially the Spider algorithm [FLLZ18] with a different learning rate and analysis.

Lemma 3. *Consider Algorithm 5, and for any $t \in \{0, .., T\}$ let $s_t = \left\lfloor \frac{t}{q} \right\rfloor q$. If each ∇_t computed in line 9 is an unbiased estimate of $\nabla \widehat{L}(w_t; S)$ satisfying $\mathbb{E}\left[\left\|\nabla_{s_t} - \nabla \widehat{L}(w_{s_t}; S)\right\|^2\right] \leq \tau_1^2$ and each Δ_t computed in line 13 is an unbiased estimate of the gradient variation satisfying $\mathbb{E}\left[\left\|\Delta_t - [\nabla \widehat{L}(w_t; S) - \nabla \widehat{L}(w_{t-1}; S)]\right\|^2\right] \leq \tau_2^2 \|w_t - w_{t-1}\|^2$. Then for any $t \geq s_t + 1$, the iterates of Algorithm 5 satisfy*

$$\mathbb{E}\left[\left\|\nabla_t - \nabla \widehat{L}(w_t; S)\right\|^2\right] \leq \tau_2^2 \sum_{k=s_t+1}^{t} \mathbb{E}\left[\|w_k - w_{k-1}\|^2\right] + \tau_1^2.$$

For privacy, using smoothness we observe the sensitivity of the gradient variation estimate at iteration t is proportional to $\beta \|w_t - w_{t-1}\|$. Thus we can apply the above lemma with $\tau_1^2 = \frac{G^2}{b_1} + G^2\sigma_1^2$ and $\tau_2^2 = \frac{H^2}{b_2} + H^2\sigma_2^2$ (note the Gaussian noise in line 13 is drawn with variance scale at most $\sigma_2^2 \|w_t - w_{t-1}\|^2$). By carefully balancing the algorithm parameters, we are then able to obtain the following result. The full proof is deferred to Appendix B.2.1.

Theorem 16 (Private Spiderboost ERM)**.** *Let $\epsilon, \delta \in [0, 1]$ and $n \geq \max\left\{\frac{(G\epsilon)^2}{L_0 H d \log(1/\delta)}, \frac{\sqrt{d}\max\{1, \sqrt{HL_0}/G\}}{\epsilon}\right\}$. Algorithm 5 is (ϵ, δ)-DP. Further, there exist settings of T, η, q, b_1, b_2 such that Algorithm 5 satisfies*

$$\mathbb{E}\left[\left\|\nabla \widehat{L}(\widehat{w}; S)\right\|\right] = O\left(\left(\frac{\sqrt{L_0 H G}\sqrt{d \log(1/\delta)}}{n\epsilon}\right)^{2/3} + \frac{G\sqrt{d \log(1/\delta)}}{n\epsilon}\right)$$

and has oracle complexity $\widetilde{O}\left(\max\left\{\left(\frac{n^{5/3}\epsilon^{2/3}}{d^{1/3}}\right), \left(\frac{n\epsilon}{\sqrt{d}}\right)^2\right\}\right)$.

In the case where the dominant error term is $\alpha = \widetilde{O}\left(\left[\frac{\sqrt{d}}{n\epsilon}\right]^{2/3}\right)$, then we approximately have oracle complexity $\widetilde{O}\left(\max\left\{\frac{1}{\alpha^3}, \frac{n}{\alpha}\right\}\right)$.

3.2.2 Lower Bound

We now show a lower bound for the sample complexity of finding a stationary point under differential privacy in the unconstrained setting, which shows that the $O\left(\frac{G\sqrt{d \log(1/\delta)}}{n\epsilon}\right)$ term in the rate given in Theorem 16 is necessary. Furthermore, as our

lower bound holds for all levels of smoothness, it also shows that our rate in Theorem 16 is optimal in the (admittedly uncommon) regime where $H \leq \frac{\sqrt{d}G^2}{L_0 n\epsilon}$. Our lower bound in fact holds even for convex functions. Furthermore, this result implies the same lower bound (up to log factors) for the population gradient using the technique in [BFTGT19, Appendix C].

Theorem 17. *Given $G, H, n, \epsilon = O(1), 2^{-\Omega(n)} \leq \delta \leq 1/n^{1+\Omega(1)}$, there exists an G-Lispchitz, H-smooth (convex) loss $\ell : \mathbb{R}^d \times \mathcal{Z} \to \mathbb{R}$ and a dataset S of n points such that any (ϵ, δ)-DP algorithm run on S with output $\widehat{w}$ satisfies,*

$$\left\|\nabla \widehat{L}(\widehat{w}; S)\right\| = \Omega\left(G \min\left(1, \frac{\sqrt{d \log(1/\delta)}}{n\epsilon}\right)\right).$$

The proof is based on a reduction to DP mean estimation. Specifically, we consider a instance of the Huber loss function for which the minimizer is the empirical mean of the dataset. We then argue that close to the minimizer, the empirical stationarity is lower bounded by DP mean estimation bound [SU15], and far away, by construction, the empirical stationarity is G. We defer the details to Appendix B.1.

Challenges for Further Rate Improvements. Given the above lower bound, the question arises as to whether the $\widetilde{O}\left(\left[\frac{\sqrt{d}}{n\epsilon}\right]^{2/3}\right)$ term can be improved. An informal argument using the oracle complexity lower bound of [ACD+19] suggests several major challenges in obtaining further rate improvements. A more detailed version of the following discussion can be found in Appendix B.2.2.

Consider methods which ensure privacy by directly privatizing the gradient/gradient variation queries. The aim of such methods is to design some private stochastic first order oracle, $\mathcal{O}_{\epsilon',\delta'}$, such that a set of Q queries to $\mathcal{O}_{\epsilon',\delta'}$ satisfies (ϵ, δ)-DP, and use this oracle in some optimization algorithm $\mathcal{A}(\mathcal{O}_{\epsilon',\delta'})$. Such a setup encapsulates numerous results in the convex setting [BFTGT19, KLL21], and is even more dominant in non-convex settings [WYX17, ZCH+20, ACG+16b]. Under advanced composition

based arguments, to make Q calls to such a private oracle one needs $\epsilon' \leq \epsilon/\sqrt{Q}$. Now, standard fingerprinting code arguments suggest lower bounds on the level of accuracy of any such private oracle [SU15]. Specifically, without leveraging further problem structure beyond Lipschitzness, one needs the gradient estimation error to be at least $\tau_1 = \Omega\left(\frac{G\sqrt{Qd\log(1/\delta)}}{n\epsilon}\right)$. A similar argument suggests the error in the gradient variation between iterates w, w' must at least $\tau_2 \|w - w'\| = \Omega\left(\frac{H\|w-w'\|\sqrt{Qd\log(1/\delta)}}{n\epsilon}\right)$. Now consider some optimization algorithm, $\mathcal{A}$, which takes as input a stochastic oracle $\mathcal{O}$ for some smooth function $\mathcal{L}$. The lower bound of [ACD+19] suggests that if $\mathcal{A}$ makes at most Q queries to $\mathcal{O}$, the algorithm satisfies $\mathbb{E}\left[\|\nabla\mathcal{L}(\mathcal{A}(\mathcal{O}))\|\right] = \Omega\left(\left(\frac{L_0\tau_2\tau_1}{Q}\right)^{1/3} + \frac{\tau_1}{\sqrt{Q}}\right)$. If $\mathcal{O}$ is a private oracle satisfying the previously mentioned conditions, we would then have under the setting of τ_1 and τ_2 suggested by privacy that

$$\mathbb{E}\left[\|\nabla\mathcal{L}(\mathcal{A}(\mathcal{O}))\|\right] = \Omega\left(\left(\frac{\sqrt{L_0HG}\sqrt{d\log(1/\delta)}}{n\epsilon}\right)^{2/3} + \frac{G\sqrt{d\log(1/\delta)}}{n\epsilon}\right).$$

This indicates a substantial challenge for future rate improvements, as alternative methods which avoid private gradients (see e.g. [FKT20b]) rely crucially on stability guarantees arising from convexity.

3.3 Stationary Points of Population Risk

Algorithm 6 Tree-based Private Spider

Input: $S = (z_1, \ldots, z_n) \in \mathcal{Z}^n$: private dataset, (ϵ, δ): privacy parameters, T: number of rounds, b: batch size at beginning of each round, D: depth of trees at each round, β: step-size parameter, $\widetilde{\alpha}$: accuracy parameter.

$w_{0,\ell(2^D-1)} = 0$
for $t = 1$ to T **do**
  Set $w_{t,\varnothing} = w_{t-1,\ell(2^D-1)}$
  Draw a batch $S_{t,\varnothing}$ of b data points, set $S \leftarrow S \setminus S_{t,\varnothing}$.
  Set $\sigma^2_{t,\varnothing} := \frac{8L_0^2 \log(1.25/\delta)}{b^2\epsilon^2}$.
  $\nabla_{t,\varnothing} = \frac{1}{b}\sum_{z\in S_{t,\varnothing}} \nabla \ell\left(w_{t,\varnothing}; z\right) + g_{t,\varnothing}$, where $g_{t,\varnothing} \sim \mathcal{N}\left(0, \mathbb{I}_d \sigma^2_{t,\varnothing}\right)$.
  for $u_{t,s} \in \mathrm{DFS}\,[D]$ **do**
    Let $s = \widehat{s}c$, where $c \in \{0, 1\}$.
    if $c = 0$ **then**
      $\nabla_{t,s} = \nabla_{t,\widehat{s}}$
      $w_{t,s} = w_{t,\widehat{s}}$
    else
      Draw a batch $S_{t,s}$ of $\frac{b}{2^{|s|}}$ data points, set $S \leftarrow S \setminus S_{t,s}$.
      Set noise variance $\sigma^2_{t,s} := \frac{8\cdot 2^D \beta^2 \log(1.25/\delta)}{b^2\epsilon^2}$.
      $\Delta_{t,s} = \frac{2^{|s|}}{b} \sum_{z\in S_{t,s}} \left(\nabla \ell\left(w_{t,s}; z\right) - \nabla \ell\left(w_{t,\widehat{s}}; z\right)\right) + g_{t,s}$, where $g_{t,s} \sim \mathcal{N}\left(0, \mathbb{I}_d \sigma^2_{t,s}\right)$.
      $\nabla_{t,s} = \nabla_{t,\widehat{s}} + \Delta_{t,s}$.
    end if
    if $|s| = D$ (i.e, $u_{t,s}$ is a leaf) **then**
      if $\|\nabla_{t,s}\| \le 2\widetilde{\alpha}$ **then**
        Return $w_{t,s}$
      end if
      Let u_{t,s^+} be the next vertex in DFS$[D]$.
      Set $\eta_{t,s} := \frac{\beta}{2^{D/2} H \|\nabla_{t,s}\|}$
      $w_{t,s^+} = w_{t,s} - \eta_{t,s}\nabla_{t,s}$.
    end if
  end for
end for
Return $\overline{w}$, chosen uniformly at random from $\{w_{t,s} : t \in [T], u_{t,s} \text{ is a leaf}\}$.

For the population gradient, we provide a linear time algorithm; see Algorithm 6 for pseudocode. It is a noisy variant of SPIDER [FLLZ18], and utilizes a variance reduction technique tailored to an underlying binary tree structure. Namely, we run

T rounds, where at the beginning of round t we build a binary tree of depth D, whose nodes are denoted by $u_{t,s}$, where $s \in \{0,1\}^D$. Every node $u_{t,s}$ is associated with a parameter vector $w_{t,s}$ and a gradient estimate $\nabla_{t,s}$. Next, we perform a Depth-First-Search traversal of the tree. We denote by DFS[D] the set of nodes in the visiting order excluding the root, for example: DFS[2] $= \{u_0, u_{00}, u_{01}, u_1, u_{10}, u_{11}\}$. When a left child node is visited, it receives the same parameter vector and gradient estimator of the parent node.

On the other hand, when a right child node is visited, it receives a fresh set of samples and uses it to update the gradient estimator coming from the parent node. Every time a leaf node is reached, a gradient step is performed using the gradient estimator associated to the leaf. Finally, the parameter vector of a right child node comes from the gradient step performed at the right-most leaf in the left sub-tree of it. The use of the binary tree structure is benefitial because every gradient estimator is updated at most D times within a round of 2^D optimization steps, as opposed to the original SPIDER algorithm where the gradient estimators are updated at every optimization step. This way, we are able to perform the same number of optimization steps but adding substantially smaller amounts of noise, leading to a faster rate than the one we would get without using the tree. In the following, we denote by $\ell(k)$ the binary representation of any number $k \in [0, 2^D - 1]$ and by $|s|$ the depth of $u_{t,s}$ for any $t \in [T]$.

The proposed algorithm is similar to the one in Section 5 of [BGN21] for constrained Differentially Private-Stochastic Convex Optimization (DP-SCO), with the key difference that Algorithm 6 executes each round with fixed depth trees, which is key for our convergence analysis, whereas the prior work leverages convexity to construct trees that increase depth by one at each round. In addition, to choose the step-size in [BGN21] the authors leverage the bounded diameter of the domain, while our step-size is chosen as that of [FLLZ18], i.e. normalized by the norm of the gradient

estimator and proportional to the target accuracy. This choice is crucial for controlling the sensitivity of the gradient variation estimator in the unconstrained setting, and consequently for the privacy analysis as well. Our results are presented below and the proofs are deferred to Appendix B.3.

Theorem 18 (Privacy guarantee)**.** *For any $\epsilon, \delta \in [0,1]$, Algorithm 6 is (ϵ, δ)-DP.*

Theorem 19 (Accuracy guarantee)**.** *Let $p \in (0,1)$, $\epsilon, \delta > 0$, $b = \max\left\{n^{2/3}, \frac{\sqrt{n}d^{1/4}}{\sqrt{\epsilon}}\right\}$, D be such that $D2^{D+1} = b$, $T = \frac{n}{b(D/2+1)}$, $\alpha = \sqrt{2}G\max\left\{\frac{1}{n^{1/3}}, \left(\frac{\sqrt{d}}{n\epsilon}\right)^{1/2}\right\}$, $\beta = \alpha\min\{1, \frac{\sqrt{b}\epsilon}{\sqrt{d}}\}$, and $\widetilde{\alpha} = \widetilde{C}\alpha$, where $\widetilde{C} = 256\log\left(\frac{1.25}{\delta}\right)\log\left(\frac{2T2^{D+1}}{p}\right) + \frac{8HL_0\sqrt{2D}(D/2+1)}{2G^2}$. Then, for any $n \geq \max\{\sqrt{d}(\frac{D}{2}+1)^2/\epsilon, (\frac{D}{2}+1)^3\}$, with probability $1-p$, Algorithm 6 ends in line 20, returning an iterate $w_{t,s}$ with*

$$\|\nabla L(w_{t,s}; \mathcal{D})\| \leq 3\sqrt{2}G\widetilde{C}\max\left\{\frac{1}{n^{1/3}}, \left(\frac{\sqrt{d}}{n\epsilon}\right)^{1/2}\right\}.$$

Furthermore, Algorithm 6 has oracle complexity of n.

3.4 Stationary Points in the Convex Setting

Algorithm 7 Recursive Regularization

Input: Dataset S, loss function ℓ, steps T, $\{\lambda_t\}_t$, $\{D_t\}_t$, PrivateSubRoutine, number of steps of sub-routine $\{K_t\}$, selector functions $\{\mathcal{S}_t(\cdot)\}_t$, step size $\{\eta_t\}_t$, noise variances $\{\sigma_t\}_t$

1: $w_0 = 0$, $n_0 = 1$
2: Define function $(w, z) \mapsto \ell^{(0)}(w; z) = \ell(w; z) + \frac{\lambda_0}{2}\|w - w_0\|^2$
3: **for** $t = 1$ to $T-1$ **do**
4: $\quad n_t = n_{t-1} + \left\lfloor \frac{|S|}{T} \right\rfloor$
5: $\quad \widehat{w}_t = \text{PrivateSubRoutine}\left(S_{n_{t-1}:n_t}, \ell^{(t-1)}, D_t, K_t, \eta_t, \mathcal{S}_t(\cdot), \sigma_t\right)$
6: $\quad$ Define function $(w, x) \mapsto \ell^{(t)}(w; z) = \ell^{(t-1)}(w; z) + \frac{\lambda_t}{2}\|w - \widehat{w}_t\|^2$
7: **end for**

Output: $\widehat{w} = \widehat{w}_T$

In this section, we additionally assume that the loss function is convex. The motivation for this is two-fold: firstly, this setting has recently gained attention in a

non-private setting [Nes12, AZ18, FSS+19]. Secondly, in this setting we are able to establish tightly the sample complexity of approximate stationary points.

Our method is based on the recursive regularization technique proposed in [AZ18], and further improved by [FSS+19]. The main idea, as the name suggests, is to recursively regularize the objective and optimize it via some solver. For the DP setting, the key idea is to use a private sub-routine as the inner solver. Furthermore, while a solver for the unconstrained problem suffices non-privately, we need to carefully increase the radius of the constrained set over which the solver operates.

Theorem 20. *Let $G, H, \epsilon, \delta > 0$, $d, n \in \mathbb{N}$. Let $w \mapsto \ell(w; z)$ be an G-Lipschitz H-smooth convex function for all x. Let $D_t = \left(\sqrt{2}\right)^t \|w^*\|$, $\lambda_t = 2^t\lambda$, $\eta_t = \frac{\log(K_t)}{\lambda_t K_t}$, $T = \left\lfloor \log_2\left(\frac{H}{\lambda}\right)\right\rfloor$, $\sigma_t^2 = \frac{64G^2K_t^2\log(1/\delta)}{n^2\epsilon^2}$, and $\mathcal{S}_t(\{w_k\}_k) = \frac{1}{\sum_{k=1}^{K_t}(1-\eta_t\lambda_t)^{-k}}\sum_{k=1}^{K_t}(1-\eta_t\lambda_t)^{-k}w_k$.*

1. *(Optimal rate) Algorithm 7 run with* NoisyGD *(Algorithm 18 in Appendix B.4) as the PrivateSubRoutine with above parameter settings and $\lambda = \frac{G^2}{H\|w^*\|}\min\left(\frac{1}{n}, \frac{d}{n^2\epsilon^2}\right)$ and $K_t = \max\left(\frac{H+\lambda_t}{\lambda_t}\log\left(\frac{H+\lambda_t}{\lambda_t}\right), \frac{n^2\epsilon^2\left(G^2\lambda+H^{3/2}\right)}{T^2\lambda dG^2\log(1/\delta)}\right)$ satisfies (ϵ, δ)-DP, and given a dataset S of n i.i.d. samples from $\mathcal{D}$, outputs $\widehat{w}$ such that*

$$\mathbb{E}\,\|\nabla L(\widehat{w}; \mathcal{D})\| = \widetilde{O}\left(\frac{G}{\sqrt{n}} + \frac{G\sqrt{d}}{n\epsilon}\right).$$

   Furthermore, the above rate is tight up to poly-logarithmic factors.

2. *(Linear time rate) Algorithm 7 run with* PhasedSGD *(Algorithm 16) as the PrivateSubRoutine with with above parameter settings and $\lambda = \max\left(\frac{G^2}{H\|w^*\|^2}\min\left(\frac{1}{n}, \frac{d}{n^2\epsilon^2}\right), \frac{H\log(n)}{n}\right)$ and $K_t = \lfloor\frac{n}{T}\rfloor$ satisfies (ϵ, δ)-DP and given a dataset S of n i.i.d. samples from $\mathcal{D}$, in linear time, outputs $\widehat{w}$ with*

$$\mathbb{E}\,\|\nabla L(\widehat{w}; \mathcal{D})\| = \widetilde{O}\left(\frac{G}{\sqrt{n}} + \frac{G\sqrt{d}}{n\epsilon} + \frac{H\,\|w^*\|}{\sqrt{n}}\right).$$

The proof of the above result is deferred to Appendix B.4. For the tightness of the rate, the necessity of the second term $\frac{G\sqrt{d}}{n\epsilon}$ is due to our DP empirical stationarity

lower bound, Theorem 17. For the first "non-private" term $\frac{L_0}{\sqrt{n}}$, even though [FSS+19] proved a sample complexity lower bound, their instance is not Lipschitz and has $d = \Omega(n \log(n))$, hence not applicable. To remedy this, we give a new lower bound construction with a Lispchitz function in $d = 1$, Theorem 40 in Appendix B.1. The polylog dependence on H and $\|w^*\|$ in the upper bounds, is consistent with the non-private sample complexity in [FSS+19].

The second result is a linear time method which has an additional $H\|w^*\|/\sqrt{n}$ term. Firstly, if the smoothness parameter is *small enough*, then there is no overhead; this small-enough smoothness is precisely the regime in which we have linear time methods with optimal rates for smooth DP-SCO [FKT20b]. More importantly, [FSS+19] showed that even in the non-private setting, a polynomial dependence on $H\|w^*\|$ is necessary in the stochastic oracle model. However, the optimal non-private term, shown in [FSS+19], is $H\|w^*\|/n^2$, achieved by accelerated methods. Improving this dependency, if possible, is an interesting direction for future work.

3.5 Generalized Linear Models

In this section, we assume that the loss function is a generalized linear model (GLM), $\ell(w;(x,y)) = \phi_y(\langle w, x\rangle)$. Also, assume the norm of data points x are bounded by $\|\mathcal{X}\|$ and the function $\phi_y : \mathbb{R} \to \mathbb{R}$ is G-Lipschitz and H-smooth for all y. Furthermore, let rank denote the rank of design matrix $X \in \mathbb{R}^{n\times d}$.

Algorithm 8 JL method

Input: Dataset S, function $(z,y) \mapsto \phi_y(z)$, Algorithm $\mathcal{A}$, JL matrix $\Phi \in \mathbb{R}^{k\times d}$, G, H, $\|\mathcal{X}\|$

1: $\widetilde{w} = \mathcal{A}((z,y) \mapsto \phi_y(z), \{(\Phi x_i, y_i)\}_{i=1}^n, 2G\|\mathcal{X}\|, 2H\|\mathcal{X}\|^2, \epsilon, \delta/2)$

Output: $\widehat{w} = \Phi^\top \widetilde{w}$

Algorithm 8 is a generic method which converts *any* for smooth Lipschitz losses with an empirical stationarity guarantee to get dimension-independent rates on population

stationarity for smooth Lipschitz GLMs. This algorithm is the JL method from [ABG+22] used therein to give excess risk bounds for convex GLM. We note that while the JL method there is limited to the Noisy GD method, ours is a black-box reduction. Furthermore, unlike [ABG+22], we show that the JL method gives finer rank based guarantees by leveraging the fact it acts as an oblivious approximate subspace embedding (see Definition 20 in Appendix B.5).

Theorem 21. *Let $\mathcal{A}$ be an (ϵ, δ)-DP algorithm which when run on a H-smooth G-Lipschitz function on a dataset $S = \{(x_i, y_i)\}_{i=1}^n$ where $x_i \in \mathcal{X} \subseteq \mathbb{R}^d$, guarantees $\mathbb{E}\left[\left\|\nabla \widehat{L}(\mathcal{A}(S); S)\right\|\right] \leq g(d, n, H, G, \epsilon, \delta)$ and $\|\mathcal{A}(S)\| \leq poly(n, d, G, H)$ with probability at least $1 - \frac{1}{\sqrt{n}}$. Then, Algorithm 8 run with*

$$k = \Bigg\lceil \min\Bigg(\arg\min_{j \in \mathbb{N}} \left(g(j, n, 2G\|\mathcal{X}\|, 2H\|\mathcal{X}\|^2, \epsilon, \delta/2) + \frac{G\|\mathcal{X}\| \log(n)}{\sqrt{j}} \right), \operatorname{rank} \log\left(\frac{2n}{\delta}\right)\Bigg)\Bigg\rceil,$$

on a G-Lipschitz, H-smooth GLM loss, is (ϵ, δ)-DP. Furthermore, given a dataset of n i.i.d samples from $\mathcal{D}$, its output $\widehat{w}$ satisfies,

$$\mathbb{E}\left[\left\|\nabla \widehat{L}(\widehat{w}; \mathcal{D})\right\|\right] \leq \widetilde{O}\left(\frac{G\|\mathcal{X}\|}{\sqrt{n}} + g(k, n, 2G\|\mathcal{X}\|, 2H\|\mathcal{X}\|^2, \epsilon, \delta/2) \right).$$

The expression for k above comes from the subspace embedding property of JL, and from balancing the dimension of the embedding with respect to the error of $\mathcal{A}$ and the approximation error of the JL embedding. The proof is based on the properties of JL matrices: oblivious subspace embedding and preservation of norms, together with a new uniform convergence result for gradients of Lipschitz GLMs. The full proof is deferred to Appendix B.5.

Below, we instantiate the above with our proposed algorithms.

Corollary 1. *Under the assumptions of Theorem 21, Algorithm 8 run with $\mathcal{A}$ as*

1. *Private Spiderboost (Alg. 5) yields* $\left\|\nabla \widehat{L}(\widehat{w}; \mathcal{D})\right\| = \widetilde{O}\left(\frac{1}{\sqrt{n}} + \min\left(\left(\frac{\sqrt{rank}}{n\epsilon} \right)^{2/3}, \frac{1}{(n\epsilon)^{2/5}} \right) \right)$.

2. *Algorithm 7 with NoisyGD as PrivateSubRoutine, under the additional assumption that* $w \mapsto \ell(w;(x,y))$ *is convex for all* x, y*, yields* $\left\|\nabla \widehat{L}(\widehat{w};\mathcal{D})\right\| = \widetilde{O}\left(\frac{1}{\sqrt{n}} + \min\left(\frac{\sqrt{rank}}{n\epsilon}, \frac{1}{\sqrt{n\epsilon}}\right)\right)$.

We remark that the above technique also gives bounds on empirical stationarity. In particular, the first term $\frac{1}{\sqrt{n}}$, in the above guarantees, is the uniform convergence bound and the second term is the bound on empirical stationarity.

3.6 Conclusion

In this chapter, we study differentially private optimization, where the goal is to approximate stationary points of a smooth, Lipschitz and potentially non-convex function. We consider two settings, where the objective is the empirical and population risk, and propose differentially private procedures with improved rates. We further give a lower bound which demonstrates tightness of our results in some regime of problem parameters. Finally, we close these gaps, in general, under the additional assumption of convexity.

Chapter 4

Machine Unlearning via Algorithmic Stability

Summary

In this chapter, we study the problem of machine unlearning and identify a notion of algorithmic stability, Total Variation (TV) stability, which we argue, is suitable for the goal of *exact* unlearning. For stochastic convex optimization and empirical risk minimization problems, we design TV-stable algorithms based on noisy Stochastic Gradient Descent (SGD). Our key contribution is the design of corresponding *efficient* unlearning algorithms, which are based on constructing a near-maximal coupling of Markov chains for the noisy SGD procedure. To understand the trade-offs between accuracy and unlearning efficiency, we give upper and lower bounds on excess empirical and populations risk of TV stable algorithms for convex risk minimization. Our techniques generalize to arbitrary non-convex functions, and our algorithms are differentially private as well.

4.1 Introduction

In this chapter, we study the problem of machine unlearning under the proposed criterion of *exact unlearning* – see Definition 11 for a formal definition. The primary

objective is design of efficient unlearning algorithms for convex learning settings. In these settings, and indeed in much of machine learning and even beyond, *gradient-based optimization* stands out as the principal algorithmic approach. Unlike prior works where the (learning) algorithms for specific problems (like linear regression) are tailored enough to the problem to be amenable to efficient unlearning, an optimization method hardly has any such useful structure. It is then natural to ask whether we can design unlearning algorithms with non-trivial guarantees for a class of convex problems.

We operate in a a streaming setting, where we start with a given initial dataset, and observe a stream of edits (insertion or deletion) to the dataset. The goal is to design a learning algorithm that outputs an initial model and a (corresponding) unlearning algorithm that updates the model after an edit request. We require the following properties to hold: *(a).* exact unlearning – at every time point in the stream, the output model is indistinguishable from what we would have obtained if trained on the *updated* dataset (i.e., without the deleted sample or with the inserted sample), *(b).* the unlearning runtime is *non-trivially small*, and *(c).* the output models are *non-trivially accurate*.

4.1.1 Contributions

Total Variation Stability. We develop new algorithmic principles which *enable* exact unlearning in general settings. In particular, we identify a notion of algorithmic stability, called total variation (TV) stability - an algorithmic property, which for any problem, yields an *in-principle* exact unlearning algorithm. Such an algorithm might not be efficiently implementable computationally or due to the data access restriction (sequential nature of edits). To demonstrate the generality of our framework, we discuss, in Section C.8.2 how the previous work of [GGVZ19] for unlearning in k-means clustering using randomized quantization can be interpreted as a special case of our

framework - a TV stable method, followed by efficient coupling based unlearning. Finally, we note that the notion of TV-stability has appeared before in [BNS+16], although in the seemingly unrelated context of adaptive data analysis.

Stochastic Convex Optimization and ERM. We make the above ideas of TV stability constructive in the special case of smooth convex ERM problems. To elaborate, we give a TV stable learning algorithm, and a corresponding *efficient* exact unlearning algorithm for smooth convex ERM. Informally, for n data points, and d dimensional model and a given $0 < \rho \leq 1$, our method retrains only on ρ fraction of edit requests, while satisfying exact unlearning and maintaining that the accuracy (excess empirical risk) is at most $\min\left\{\frac{1}{\sqrt{\rho n}}, \left(\frac{\sqrt{d}}{\rho n}\right)^{4/5}\right\}$ (see Theorem 22 for precise statement). This implies that for the (useful) regime of accuracy greater than $\min\left\{\frac{1}{\sqrt{n}}, \left(\frac{\sqrt{d}}{n}\right)^{4/5}\right\}$, our algorithms provide a *strict* improvement over the only known method of re-computation - see remarks after Theorem 22 for details. Furthermore, we also give excess population risk bounds by leveraging known connections between generalization and algorithmic stability (see Section C.10). Finally, to assess how far are our proposed (TV stable) algorithms from the most accurate, we give preliminary lower bounds on excess empirical and population risk for TV stable algorithms for convex risk minimization.

Extensions. Our results yield a number of interesting properties and extensions.

- *Privacy:* Even though privacy is not the goal of this work, as a consequence of our techniques, some of our ρ-TV stable algorithms, those based on noisy SGD (Algorithm 9, 21) are (ϵ, δ)-differentially private (see Definition 9) with $\epsilon = \rho\sqrt{\log(1/\delta)}$, for any $\delta > 0$. It is easy to see that these parameters can lie in the regime reasonable for *good* privacy properties i.e. $\epsilon = O(1), \delta < \frac{1}{n^2}$. However, not all TV-stable algorithms, for instance Algorithm 19, may have good privacy properties. Our work

therefore demonstrates interesting connections between techniques developed for differential privacy and the problem of unlearning.

- *Beyond Convexity and practical heuristics*: Interestingly, our unlearning techniques only require finite sum structure in the optimization problem (for exact unlearning) and Lipschitzness (for runtime bounds). Therefore our unlearning algorithms yield provable unlearning for gradient-descent based methods for *any* ERM problem. In particular, we can apply the unlearning algorithm to non-convex (and potentially non-Lipschitz) problems, like deep neural networks, and everytime the algorithm does not recompute, it still guarantees exact unlearning. In contrast, differential private training of non-convex models require Lipschitzness or use *clipping* of gradients - a popular heuristic in deep learning. We can similarly use clipping to give unlearning runtime bounds in non-Lipschitz scenarios. As is typical, the accuracy in these cases is estimated empirically.

4.1.2 Related Work

The problem of exact unlearning in convex ERM has not been studied before, and therefore the only baseline is re-computation (using some variant of gradient descent). The most related are the works of [GGVZ19] and [NRSM21a], which we discuss as follows. [GGVZ19] studied the problem of k-means clustering with exact unlearning in a streaming setting of deletion requests - we borrow the setting (while also allowing insertions) and the notion of exact unlearning from therein. We note that in [GGVZ19], the notion of efficiency is based on the amortized (over edits) unlearning time being at most the training time since that is a natural lower bound on the overall computational cost. We, on the other hand, do not place such a restriction and so our methods can have unlearning runtime smaller than the training time. Most importantly, the general framework here (of TV-stable methods and coupling based unlearning) captures the quantized-k-means algorithm of [GGVZ19] as a special case (see Section C.8.2 for

details).

The work of [NRSM21a] focuses on unlearning in convex ERM problems, with a stream of edit requests, the same as here. However there are two key differences. First, the notion of unlearning in [NRSM21a] is approximate, based on (ϵ, δ)-differential privacy, whereas we focus on exact unlearning. Second, the unlearning runtime in [NRSM21a] is deterministic, whereas that of ours is random. These are akin to *Monte-Carlo* vs. *Las Vegas* style of guarantee discrepancy. We refer the reader to an extended literature survey along with a detailed comparison to [NRSM21a] in Section C.1. We show therein that with the same unlearning time, the accuracy guarantees of [NRSM21a] are better than us only in regimes where their approximate unlearning parameters and hence the notion, is weak.

4.2 Additional Preliminaries

We review the problem setup and present additional preliminaries below.

We operate under the criterion of exact unlearning – see Definition 11. However, since we also consider insertions here, the unlearning requests can also include insertions to the dataset.

Let $S = S^0 = \{z_1, z_2, \ldots, z_n\}, z_i \in \mathcal{Z}$ be the initial dataset of n points. We observe k edit requests, each being either an insertion or deletion request. We use S^i to denote the set of data points available at time i in the stream. For simplicity, we assume that at any point in the stream, the number of available data points is at least $n/2$ and at most $2n$.

Total Variation Stability. We define Total Variation stability (TV-stability), which is the notion of algorithmic stability we propose to use.

Definition 15 (ρ-TV-stability)**.** *An algorithm $\mathcal{A}$ is said to be ρ-TV-stable if*

$$\sup_{S,S':S\sim_n S'} TV(\mathcal{A}(S),\mathcal{A}(S')) \leq \rho$$

See Section 1.2.4.1 for preliminaries on Total Variation distance (including its definition in Definition 13) as well as its connection to *maximal coupling*. We note that the notion of TV stability has appeared in prior works, such as [BNS+16], albeit in different contexts than machine unlearning.

A few remarks are in order.

Remark 1.

1. *In the above definition, we consider the marginals of output, and do not include the metadata.*

2. *Suppose S is a dataset of n points, and S' is a dataset of $n+k_2$ points such that $|S\backslash S'| = k_1$. Then, if algorithm $\mathcal{A}$ is ρ-TV stable, then by triangle inequality of TV distance, we have that $TV(\mathcal{A}(S),\mathcal{A}(S')) \leq (2k_1+k_2)\rho$*

4.3 Main Results

We state our main result on designing learning and unlearning algorithms in a stream of edit requests.

Theorem 22 (Main Theorem)**.** *For any $\frac{1}{n} \leq \rho < \infty$, there exists a learning and a corresponding unlearning algorithm such that for any $w \mapsto \ell(w;z)$, which is H-smooth and G-Lipschitz convex function $\forall\ z$, and a stream of edit requests,*

1. *Satisfies exact unlearning at every time point in the stream of edit requests.*

2. *At time i in the stream, outputs $\widehat{w}_{S^i}$ with excess empirical risk bounded as,*

$$\mathbb{E}\left[\widehat{L}(\widehat{w}_{S^i};S^i) - \widehat{L}(w^*_{S^i};S^i)\right] = O\left(\min\left\{\frac{GD}{\sqrt{\rho n}}, \left(\frac{H^{1/4}GD^{3/2}\sqrt{d}}{(\rho n)}\right)^{4/5}\right\}\right).$$

3. *For k edit requests, the expected unlearning runtime is* $O(\max\{\min\{\rho, 1\}\, k \cdot \textit{Training time}, k\})$.

We make some remarks about the result.

Training time. Informally, what the above theorem says is that the algorithms satisfy exact unlearning and are accurate while only recomputing a ρ fraction of times - this is indeed the nature of our proposed algorithms. "Training time" in the above theorem is the runtime of the learning algorithm (for the aforementioned accuracy). If we measure training time in terms of number of gradient (oracle) computations, as is typical in convex optimization, then for the above accuracy, our algorithm has optimal oracle complexity in *most* regimes (see details in Section C.7.1).

Role of ρ. The external parameter ρ controls the trade-off between accuracy and unlearning efficiency. In the extreme case where we don't care about unlearning efficiency and are fine with paying retraining computation for every edit request, then we can set $\rho > 1$ as large as we want to get, as expected, arbitrary small excess empirical risk. However, the interesting case is when we set $\rho < 1$: herein, we get an *improved* (see below) unlearning time and yet a non-trivial accuracy, upto $\rho = \Omega\left(\frac{1}{n}\right)$.

Improvements. The above result may seem like a trade-off, but, as we argue below, is a strict improvement over the baseline of retraining after every edit request (which is the only other known method for exact unlearning for this problem). Let the target excess empirical risk be $\alpha > \alpha_0 = \min\left\{\frac{GD}{\sqrt{n}}, \left(\frac{H^{1/4}GD^{3/2}\sqrt{d}}{n}\right)^{4/5}\right\}$. For any such α, from the above result, there exists a $\rho < 1$, such that our algorithms have $\rho k \cdot$ Training time(α) expected unlearning time, which is smaller than $k \cdot$ Training time(α) - the cost of retraining after every edit request. Furthermore, as remarked above, since our training time is optimal in number of gradient computations (for the said

accuracy), the aforementioned improvement holds for re-computation with *any* first-order optimization algorithm. A small caveat is that we are comparing our *expected* unlearning time with deterministic runtime of retraining. To summarize, with this caveat, we have a strict improvement in the *low* accuracy regime, whereas in the high accuracy regime: $\alpha < \alpha_0$, our unlearning algorithms are as good as trivial re-computation. However, this low accuracy regime is often the target in machine learning problems. To elaborate, the goal is to minimize the population risk rather than empirical risk, and it is well known that this statistical nature of the problem results in an information-theoretic lower bound of $\frac{1}{\sqrt{n}}$ on excess population risk. We show in Section C.10 that our algorithm guarantees an excess population risk of $\frac{1}{\sqrt{n}} + \alpha$, and so a very small α only becomes a lower order term in excess population risk.

Algorithms. The first upper bound on accuracy in Theorem 22 is obtained by standard SGD, which, in each iteration samples a fraction of datapoints, called mini-batch, to compute the gradient, and performs the descent step - we call this *sub-sample-GD*. The second upper bound is obtained using noisy accelerated mini-batch-SGD (*noisy-m-A-SGD*), which is also used for differentially private ERM. Our unlearning algorithm for *sub-sample-GD* is rather straightforward, and most of the work is design of unlearning algorithm for *noisy-m-A-SGD*, which is based of efficient coupling of Markov chains corresponding to the learning algorithm. We describe the algorithms in detail in Section 4.5.

Sub-optimality within the TV stability framework. If we consider $G, H, D = O(1)$, and a simple model of computation wherein we pay a unit computation when we recompute, otherwise not, then the unlearning problem is equivalent to design of TV-stable algorithms, and a corresponding (maximal) coupling (see Section C.2.1 for more details). Our coupling construction for unlearning in *noisy-m-A-SGD*, though efficient, is not maximal - this gap shows up in the accuracy bound (second term),

which is $\left(\frac{\sqrt{d}}{\rho n}\right)^{1/5}$ worse than what we would have obtained via a maximal coupling i.e $\frac{\sqrt{d}}{\rho n}$. We also note that in case we don't use acceleration, but rather vanilla noisy mini-batch SGD, and the "same" coupling construction for unlearning, then we obtain a worse accuracy bound of $\left(\frac{\sqrt{d}}{\rho n}\right)^{2/3}$ (see Section C.8.1 for details). Finally, apart from closing the gap with the maximal coupling, another potential improvement is by giving ρ-TV stable algorithms with *better* accuracy. We discuss such upper and lower bounds as follows.

As pointed out, intermediate to the result in Theorem 22 is the design and analysis of TV-stable algorithms for smooth convex ERM. Our main result on upper bounds on accuracy of such algorithms is the following.

Theorem 23 (Upper bound)**.** *For any $0 < \rho < \infty$, there exists an algorithm which is $\min\{\rho, 1\}$-TV stable, such that for any function $\ell(\cdot; z)$ which is H-smooth and G-Lipschitz convex $\forall\ z$, and any dataset S of n points, outputs $\widehat{w}_S$ which satisfies the following.*

$$\mathbb{E}\left[\widehat{L}(\widehat{w}_S; S) - \widehat{L}(w_S^*; S)\right] = O\left(GD \min\left\{\frac{1}{\sqrt{\rho n}}, \frac{\sqrt{d}}{\rho n}\right\}\right).$$

We show that the condition $\rho \geq \frac{1}{n}$ in Theorem 23 is fundamental for any non-trivial accuracy, as evidenced by our lower bounds, with a matching dependence on ρ. Furthermore, we omit the regime $\rho \geq 1$ in our lower bound since it puts no constraint on the algorithms to exhibit a non-trivial lower bound.

Theorem 24 (Lower bound)**.** *For any ρ-TV-stable algorithm $\mathcal{A}$, there exists a G-Lipschitz, 0-smooth convex function ℓ and a dataset S of n points such the expected excess empirical risk is lower bounded as:*

1. *For any $0 < \rho < 1$, and any dimension d, $\mathbb{E}\left[\widehat{L}(\mathcal{A}(S); S) - \widehat{L}(w_S^*; S)\right] = \Omega\left(GD \min\left\{1, \frac{1}{\rho n}\right\}\right)$.*

2. *Assuming that* $\mathcal{A}(S)$ *has a probability density function upper bounded by* $K \leq O(2^d)$*, then for* $n > 72, \frac{1}{n} \leq \rho \leq \frac{1}{4}$ *and large enough* d*,* $\mathbb{E}\left[\widehat{L}(\mathcal{A}(S); S) - \widehat{L}(w_S^*; S)\right] = \Omega\left(GD \min\left\{1, \frac{1}{\sqrt{\rho n}}\right\}\right)$.

In each of the lower bounds, the term GD is trivial as it is attained if an algorithm outputs a constant regardless of the problem instance. The first lower bound holds for *all* problem instances without any assumptions on the relationship between the problem parameters d, n and ρ. Note that if we assume that the upper bound given by Theorem 23 were tight, in that case we would expect to derive a lower bound of $\frac{\sqrt{d}}{\rho n}$ whenever $\frac{\sqrt{d}}{\rho n} \leq \frac{1}{\sqrt{\rho n}} \iff d \leq \frac{1}{\rho n}$ - we would therefore need to shrink the class of problem instances explicitly. Unfortunately, our techniques currently do not show improvement with this restriction. The second result is obtained by a *direct* analysis, where the key ingredient is the fact that normalized volume of spherical cap of a hypersphere goes to 0 as $d \to \infty$, for a fixed width of the cap. The condition that the probability distribution $\mathcal{A}(S)$ has bounded density prevents it to have discrete atoms - this is not desirable especially since our (upper bound) algorithm *sub-sample-SGD* outputs a mixture of discrete distributions, and therefore does not lie in this class. Please see Section C.9 for derivations of the lower bounds.

4.4 Main Ideas

In this section, we discuss the key ideas to our approach. The first is identifying a notion of stability. The key idea is maximal coupling characterization of total variation distance. Note that if the outputs on neighbouring datasets S (original) and S' (after an edit) are close in TV distance, then there exists a maximal coupling under which, the output on S can *likely* be reused while still satisfying exact unlearning. Our approach is to make this idea constructive for convex ERM problem. We also note that the notion of TV stability for unlearning can be motivated from and cast in

a more general framework based on optimal transport (see for Section C.2.1 more details). The second ingredient is the design of TV stable algorithms for convex ERM. One of the algorithms we propose is an existing differential private solution, which we show to be TV stable as well. Finally, the bulk of the work, is the design and analysis of efficient unlearning algorithms. We show that this problem can be reduced to efficiently constructing couplings between Markov chains, and we give such a construction using rejection sampling and reflection mappings.

4.4.1 TV-stable Learning Algorithms and Differential Privacy

In this section, we discuss the ideas underlying the design of TV-stable learning algorithms. We will use the notion of differential privacy (DP) – see Definition 9 – which will serve as a key tool.

A differentially private algorithm promises that the output distributions are close in a specific sense: the likelihood ratios for all events for two neighbouring datasets is uniformly close to $e^{\pm\epsilon}$, upto a failure probability δ. Note that we have identified that we want our outputs to be ρ-TV stable. A natural question is whether we can relate the (ϵ, δ)-DP notion to ρ-TV-stability. An easy to see direction is that any ρ-TV stable method is (at least) $(0, \rho)$-DP. Similarly, for the other direction, under additional assumptions, such relations can be derived. The important part is that certain widely used DP methods are TV stable as well. The primary example, which we will use in this work, is Gaussian mechanism. It is known that adding Gaussian noise of standard deviation $\frac{\sqrt{\log(1/\delta)}}{\epsilon}$ to a 1-sensitive function, provides (ϵ, δ)-DP [DR+14]. It can be shown that the same method also provides ρ-TV stability, with $\rho = \frac{\epsilon}{\sqrt{\log(1/\delta)}}$.

TV-stable Algorithm. For the problem of TV-stable convex empirical risk minimization, we propose two algorithms: *sub-sample-GD* and *noisy-m-A-SGD*. We show that the expected excess empirical risk of *noisy-m-A-SGD* is better than that of *sub-*

sample-GD, in regimes of small dimension. The algorithm *noisy-m-A-SGD* is essentially *noisy-SGD* algorithm of [BST14] for DP convex ERM, with an additional (Nesterov's) acceleration on top. In *noisy-m-A-SGD*, at iteration t, we sample a mini-batch b_t uniformly randomly, use it to compute the gradient at iterate $\mathring{w}_t = (1-\alpha_t)w_t + \alpha_t w_{t-1}$ denoted as $\nabla \widehat{L}(\mathring{w}_t; S_{b_t})$ and update as follows:

$$w_{t+1} = \mathring{w}_j - \eta \left(\nabla \widehat{L}(\mathring{w}_j; S_{b_j}) + \theta_t \right)$$

where $\theta_j \sim \mathcal{N}(0, \sigma^2 \mathbb{I}_d)$ and σ is set appropriately. This procedure can be viewed as sampling from a Markov chain depicted in Figure 4-1.

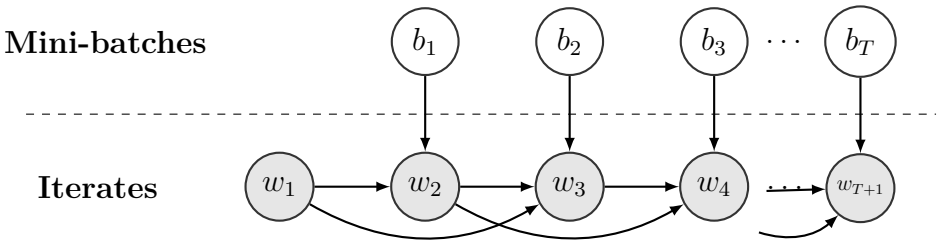

Figure 4-1. Markov chain for *noisy-m-A-SGD* Algorithm

A detailed discussion of challenges and certain subtleties in applying DP algorithms to our problem is provided in Section C.2.2

4.4.2 Unlearning via (un)couplings

The final, though the most important piece, is the design of unlearning algorithms. Let S be the initial dataset, S' is the dataset after one edit request, and we want to design a transport from $P = \mathcal{A}(S)$ to $Q = \mathcal{A}(S')$, which means that we want to construct a coupling of P and Q. Broadly, there are two challenges: the first is the data access restriction - when generating a sample from P, we don't know what Q would be, therefore, the coupling cannot be based on efficiently sampling from the joint distribution directly, but is limited to work with samples generated from P. The other is that construction of the coupling should be computationally more efficient

than drawing independent samples from P and Q, which essentially amounts to our baseline of re-computation.

Framework of verification and re-computation: We encapsulate some desirable design properties while constructing couplings as a framework which gives efficient unlearning algorithms. We first setup some terminology - the diagonal of a coupling π of two probability distributions, is the set $\{(p, q) : p = q\}$ and similarly, the non-diagonal is the set $\{(p, q) : p \neq q\}$. We have that the measure of the non-diagonal, under a maximal coupling π^*, is $\mathbb{P}_{(p,q)\sim\pi^*} \mathbb{1}\{(p, q) : p \neq q\} = \mathrm{TV}(P, Q)$. This implies that when using ρ-TV stable algorithms, the probability measure of the diagonal under a maximal coupling, is *large*: at least $1 - \rho$. At a high-level, our unlearning algorithms comprise of two stages: *verification* and *recomputation*. We first *verify* whether our output on dataset S (i.e. sample from P) *falls* on the diagonal of any maximal coupling of P and Q or not - if that is indeed the case, then the same sample for Q suffices. Importantly, for computational efficiency, we require that verification be computationally much cheaper then recomputing (smaller that $\rho \cdot$ recompute cost). If the verification fails, we sample from the non-diagonal of any maximal coupling P and Q, so that we have a valid transport. As the name suggests, the computational cost of *recomputation* that we will shoot for is to be of the same order as (full) recompute. If we are able to design such a method, then we will show that for k edit requests, the expected computational cost for unlearning is $k \cdot$ verification cost $+ k\rho \cdot$ recompute cost $\approx k\rho \cdot$ recompute cost.

Coupling of Markov chains. Our approach for unlearning is to construct a coupling of the optimization trajectories on neighbouring datasets. We have discussed that the iterates from *noisy-m-A-SGD* can be seen as generated from a Markov chain, depicted in Figure 4-1. Hence, for two neighbouring datasets, the iterates are sampled from two *different* Markov chains P and Q. Moreover, by design, we know that these Markov chains are ρ-TV close - we measure the total variation distance between the

marginals of outputs. The task is now to maximally couple these two Markov Chains. We remark that in the Markov chain literature, maximal coupling of Markov chains does not refer to the above but rather the setting wherein we have one Markov chain, but started at two different states, and the goal is to design a coupling such that their sampled states become and remain equal as soon as possible. In contrast, our notion of coupling of two Markov chains has also been recently studied by [Völ16] and [EKRR19], wherein they refer to this problem as design of *uncoupling* or *maximal agreement/exit couplings*. However, the ideas in the aforementioned works are analytical and arguably not computationally efficient in general.

4.5 Algorithms

In this section, we present the algorithms for learning and unlearning. In our algorithms, we use functions "save" and "load", which vaguely means saving and loading the variables to and from memory respectively. In Section C.7, we explain what data structures to use for computational efficiency. The main result Theorem 22 is obtained by combination of two algorithms: *sub-sample-GD* (superior in high dimensions) and *noisy-m-A-SGD* (superior in low dimensions). We note that *subsample-GD* is just standard mini-batch SGD and unlearning therein is simple. Hence, due to space constraints, we defer its description to Section C.3 and focus here on *noisy-m-A-SGD*. The proofs of results in this section are deferred to Section C.5.

4.5.1 TV-stable Learning Algorithm: *noisy-m-A-SGD*

The algorithm, superior in low dimensions, called *noisy-m-A-SGD*, is accelerated mini-batched SGD [Lan12] with appropriate Gaussian noise added at each iteration. In the literature, this algorithm (with or without acceleration) is used for DP training of (non) convex models. In each iteration, we save the mini-batch indices, the models, the gradients as well as the noise vectors to memory.

Algorithm 9 *noisy-m-A-SGD*(w_{t_0}, t_0)

Input: Initial model w_{t_0}, data points $\{z_1, \ldots, z_n\}$, T, η, m

$w_0 = 0$
for $t = t_0, t_0 + 1 \ldots, T$ **do**
  Sample mini-batch b_t of size m uniformly randomly
  Sample $\theta_t \sim \mathcal{N}(0, \sigma^2 \mathbb{I}_d)$
  $\mathring{w}_t = (1 - \alpha_t) w_t + \alpha_t w_{t-1}$
  $g_t = \frac{1}{m} \sum_{j \in b_t} \nabla \ell(\mathring{w}_t; z_j)$
  $w_{t+1} = \mathcal{P}\left(\mathring{w}_t - \eta \left(g_t + \theta_t\right)\right)$
  Save$(b_t, \theta_t, w_t, \mathring{w}_t, g_t)$
end for

Output: $\widehat{w}_S = w_{T+1}$

We now state our main result for *noisy-m-A-SGD*.

Proposition 1. *Let $\ell(.; z)$ be an H-smooth G-Lipschitz convex function $\forall\ z$. For any $0 < \rho < \infty$, Algorithm 9, run with $t_0 = 1, \eta = \min\left\{\frac{1}{2H}, \frac{D}{\left(\frac{G}{\sqrt{m}} + \sigma\right) T^{3/2}}\right\}$, $\alpha_0 = 0, \alpha_t = \frac{1-t}{t+2}$, $\sigma = \frac{8\sqrt{T}G}{n\rho}$, and $T geq \frac{(n\rho)^2}{16m^2}$ outputs $\widehat{w}_S$ which is $\min\{\rho, 1\}$-TV stable and satisfies*

$$\mathbb{E}\left[\widehat{L}(\widehat{w}_S; S) - \widehat{L}(w_S^*; S)\right] = O\left(\frac{HD^2}{T^2} + \frac{GD}{\sqrt{Tm}} + \frac{GD\sqrt{d}}{n\rho}\right).$$

Choosing T (number of iterations) and m (mini-batch size) appropriately gives an excess empirical risk of $O\left(\frac{GD\sqrt{d}}{\rho n}\right)$ – see Corollary 7.

4.5.2 Unlearning Algorithm for *noisy-m-A-SGD*

We now discuss the algorithm to handle edit requests which is based on efficiently constructing near-maximal couplings. As discussed before, an important component on constructing such couplings is, what we call *verification*, wherein we check if the current model suffices after the edit request or not. If the verification is successful, we don't do any additional computation, otherwise we do a partial or full recompute (i.e. retrain), which we call *recomputation*. The key idea is that verification can be done efficiently, and fails with small probability (depending on the TV-stability parameter).

Our unlearning algorithm for *noisy-m-A-SGD* is based on efficiently constructing a

Algorithm 10 Unlearning for *noisy-m-A-SGD*

Input: Data point index j to delete or data point z to insert (index $n+1$)

for $t = 1, 2 \ldots, T$ **do**
  Load$(\theta_t, w_t, \mathring{w}_t, b_t, g_t)$
  if *deletion* and $j \in b_t$ **then**
    Sample $i \sim \text{Uniform}([n] \backslash b_t)$
    $g'_t = g_t - \frac{1}{m}\left(\nabla \ell(\mathring{w}_t; z_j) - \nabla \ell(\mathring{w}_t; z_i)\right)$
    Save$(g'_t, b_t \backslash \{j\} \cup \{i\})$
  else if *insertion* and Bernoulli$\left(\frac{m}{n+1}\right)$ **then**
    Sample $i \sim \text{Uniform}(b_t)$
    $g'_t = g_t - \frac{1}{m}\left(\nabla \ell(\mathring{w}_t; z_i) - \nabla \ell(\mathring{w}_t; z)\right)$
    Save$(g'_t, b_t \backslash \{i\} \cup \{n+1\})$
  else
    continue
  end if
  $\xi_t = g_t + \theta_t$
  if $\text{Uniform}(0,1) \geq \frac{\phi_{\mathcal{N}(g'_t, \sigma^2 \mathbb{I})}(\xi_t)}{\phi_{\mathcal{N}(g_t, \sigma^2 \mathbb{I})}(\xi_t)}$ **then**
    $\xi'_t = \text{reflect}(\xi_t, g'_t, g_t)$
    $w_{t+1} = w_t - \eta \xi'_t$
    Save(ξ'_t)
    noisy-m-A-SGD$(w_{t+1}, t+1)$ // *Continue retraining on current dataset*
    break
  end if
end for

coupling of Markov chain describing *noisy-m-A-SGD*, with large mass on its diagonal. The proof of result in this section is deferred to Section C.6. We first describe how Algorithm 10 couples mini-batch indices while handling edit request.

Coupling Mini-batch Indices. After observing a deletion request, in Algorithm 10, we look at all iterations in which the deleted point was sampled. We then replace the deleted point with a uniformly random point not already sampled in that iteration. For insertion, at each step, we again replace a uniformly sampled point in the mini-batch of that step by the inserted point with probability $\frac{m}{n+1}$.

Reflection Coupling. We use the technique of *reflection coupling*, described in Section 1.2.4.1, for our coupling construction. The reflection map, given the means of

two distributions, denoted as μ_P and μ_Q and a sample from $p \sim P$, reflects p about the mid-point of μ_P and μ_Q. The context in which we will use it is p would be a sampled point from a Gaussian distribution under the old dataset S (on which the model was trained on), with μ_P and μ_Q being the means of the Gaussian distribution under the updated dataset S' (after edit request) and S respectively. The map essentially exploits the spherical symmetry of the Gaussian to generate a *good* sample for the distribution under S'. Please see Section C.6.2.2 for properties of the reflection map, which are used in the final proofs.

Iterative Rejection Sampling. Our unlearning algorithm is based on iteratively verifying each model w_{t+1} using rejection sampling. To elaborate, at each iteration, we check if the noisy iterate, defined as $\bar{w}_{t+1} = \mathring{w}_t - \eta(g_t + \theta_t)$ is a *good* sample for the dataset S', where g_t is the gradient computed on $\mathring{w}_t$ using a uniformly sub-sampled mini-batch from S. To do this, we need to compute a ratio of *estimated marginal densities* (we just use conditional density under the coupled mini-batch, more details below) of w_{t+1} (line 15 in Algorithm 10) for both datasets, evaluated at the noisy iterate, and compare it with Uniform$(0, 1)$. It it succeeds, we move to the next iteration and repeat. If any of the rejection sampling fails, we use the reflection map (line 16) to obtain a new w_{t+1}, and continue retraining on S'.

Verification and Recompuation stages. Herein, the rejection sampling steps comprise the verification stage, and if any of the rejection sampling fails, we move to re-computation. The reason why verification can be done efficiently is due to the finite sum structure of the ERM problem. To elaborate, at any iteration, to compute the estimated marginal density, we need to compute the gradient with the new dataset S' - this, using the gradient of the old dataset only requires subtracting the gradient at the deleted point, so $O_d(1)$ runtime as opposed to $O_d(m)$, if we were to compute from scratch, where m is the mini-batch size. Moreover, throughout verification,

this computation is done only for iterations which used the deleted point which are roughly $\frac{Tm}{n}$ iterations. Hence the total runtime of verification is $O_d\left(\frac{Tm}{n}\right)$ as opposed to $O_d(Tm)$ for re-computation. Similar reasoning applies to the insertion case.

Fast Algorithms and Maximal Coupling. The above procedure generates a coupling but not a maximal coupling - the measure of the diagonal under the coupling, and hence the probability to recompute, is $\sqrt{T}$ worse then the optimal, where T is the number of iterations run of *noisy-m-A-SGD*. This gives us that the faster the algorithm (in terms of iteration complexity) is, the smaller the probability to recompute, when using our coupling construction. This motivates why we use *accelerated* mini-batch SGD, since it has a quadratically faster iteration complexity than vanilla mini-batch SGD. In Section C.7.1, we also remark that using even faster variance-reduction based algorithms like Katyusha [AZ17] do not yield further improvements.

Estimation of Marginals. We explain what we mean by *estimated marginal densities* in the previous paragraph. Ideally, we want to create maximal coupling of marginals of the output, and therefore measure TV distance between marginals, rather than the entire algorithmic state. In particular, fix all iterates before iteration t, and consider noisy iterate $\bar{w}_{t+1} = \mathring{w}_t - \eta(g_t + \theta_t)$. If we also fix the sampled mini-batch b_t, then $\bar{w}_{t+1}$ is distributed as $\mathcal{N}(\mathring{w}_t - \eta g_t, \eta^2\sigma^2\mathbb{I})$. However, once we unfix b_t, then w_{t+1} is mixture of Gaussians, with the number of components being exponential in m. and therefore even evaluating the marginal density is infeasible. One solution is to just consider $m = n$ i.e. full gradient descent, and then there is no additional state. However, this makes the training runtime worse, which means that we would be using a *slower* learning algorithm than what we would have used if we were to simply recompute to unlearn. Hence, to tackle this, as alluded to in the previous paragraph, we evaluate the ratio of *conditional* probability densities, where the conditioning is

on the coupled mini-batch indices (say b_t^S and $b_t^{S'}$) i.e. $\frac{\phi_{S'}(\bar{w}_t|b_t^S)}{\phi_S(\bar{w}_t|b_t^S)}$ – this is done in line 15 of Algorithm 10, with a small change that we evaluate the ratio of conditional densities of noisy gradients rather than iterates, but it can be verified that the ratio is invariant to this shift and scaling. This use of conditional density corresponds to using unbiased estimates of the marginals densities. It is easy to verify, using convexity of the pointwise supremum for instance, that

$$\mathrm{TV}((\bar{w}_t^S, b_t^S), (\bar{w}_t^{S'}, b_t^{S'})) \geq \mathbb{E}_{(b_t^S, b_t^{S'})} \mathrm{TV}(\bar{w}_t^S | b_t^S, \bar{w}_t^{S'} | b_t^{S'}) \geq \mathrm{TV}(\bar{w}_t^S, \bar{w}_t^{S'})$$

However, in general, this might still not be ideal since we are estimating with just one sample from the mixture and hence the estimation error would be large. However, we will show that since we are anyway not able to construct maximal couplings, we don't pay extra with this coarse estimate.

Please see Section C.6.2.3 for a more formal treatment of the coupling procedure. We now state the main result for this section.

Proposition 2. *(Algorithm 9, Algorithm 10) satisfies exact unlearning. Moreover, for k edits, Algorithm 10 recomputes with probability at most $\frac{k\rho\sqrt{T}}{4}$.*

In the above proposition, we state unlearning efficiency in terms of probability to recompute. In Section C.7.2, we provide more details on runtime (analytical complexity bounds on total unlearning time, which is $O(\rho k\sqrt{T}$ Training time)) as well as space complexity with an efficient implementation.

4.6 Conclusion

In this chapter, we studied the problem of machine unlearning under the exact unlearning criterion. We defined a notion of algorithmic stability, Total Variation (TV) stability, which motivated a general strategy for unlearning. This involved designing a TV stable learning algorithm and a corresponding unlearning algorithm which

constructs a near maximal coupling of the outputs on the original and updated datasets. We applied this to the smooth convex ERM and stochastic convex optimization settings, yielding accurate and efficient unlearning algorithms. Our techniques generalize to the streaming setting involving deletions and insertions.

Chapter 5

From Adaptive Query Release to Machine Unlearning

Summary

In this chapter, we formalize the problem of machine unlearning as design of efficient unlearning algorithms corresponding to learning algorithms which perform a selection of adaptive queries from structured query classes. We give efficient unlearning algorithms for *linear* and *prefix-sum* query classes. As applications, we show that unlearning in many problems, in particular, stochastic convex optimization (SCO), can be reduced to the above, yielding improved guarantees for the problem. In particular, for smooth Lipschitz losses and any $\rho > 0$, our results yield an unlearning algorithm with excess population risk of $\widetilde{O}\Big(\frac{1}{\sqrt{n}} + \frac{\sqrt{d}}{n\rho}\Big)$ with unlearning query (gradient) complexity $\widetilde{O}(\rho \cdot \text{Retraining Complexity})$, where d is the model dimensionality and n is the initial number of samples. For non-smooth Lipschitz losses, we give an unlearning algorithm with excess population risk $\widetilde{O}\Big(\frac{1}{\sqrt{n}} + \Big(\frac{\sqrt{d}}{n\rho}\Big)^{1/2}\Big)$ with the same unlearning query (gradient) complexity. Furthermore, in the special case of Generalized Linear Models (GLMs), such as those in linear and logistic regression, we get dimension-independent rates of $\widetilde{O}\Big(\frac{1}{\sqrt{n}} + \frac{1}{(n\rho)^{2/3}}\Big)$ and $\widetilde{O}\Big(\frac{1}{\sqrt{n}} + \frac{1}{(n\rho)^{1/3}}\Big)$ for smooth Lipschitz and non-smooth Lipschitz losses respectively. Finally, we give generalizations of the above from one unlearning request to *dynamic* streams consisting of insertions and deletions.

5.1 Introduction

In this chapter, we continue our investigation into design of efficient unlearning algorithms. We recall that the machine unlearning is concerned with updating trained machine learning models upon request of deletions to the training dataset. We work under the notion of exact unlearning defined in the previous chapter – see Definition 11.

Motivating Example: The main objective of our work is to identify algorithmic design principles for unlearning such that it is more *efficient* than retraining, the naive baseline method. Towards this, we first discuss the example of unlearning for Gradient Descent (GD) method, which will highlight the key challenges as well as foreshadow the formal setup and techniques. GD and its variants are extremely popular optimization methods with numerous applications in machine learning and beyond. In a machine learning context, it is typically used to minimize the training loss, $\widehat{L}(w;S) = \frac{1}{n}\sum_{i=1}^{n}\ell(w;z_i)$ where $S = \{z_i\}_{i=1}^{n}$ is the training dataset and w, the model. Starting from an initial model w_1, in each iteration, the model is updated as:

$$w_{t+1} = w_t - \eta\nabla\widehat{L}(w_t;S) = w_t - \eta\left(\frac{1}{n}\sum_{i=1}^{n}\nabla\ell(w_t;z_i)\right),$$

where η is the learning rate. After training, a data-point, say z_n without loss of generality, is requested to be unlearnt and so the updated training set is $S' = \{z_i\}_{i=1}^{n-1}$. We now need to apply an *efficient* unlearning algorithm such that its output is equal to that of running GD on S'. Observe that the first iteration of GD is *simple* enough to be unlearnt efficiently by computing the *new* gradient $\nabla\widehat{L}(w_1;S') = \frac{1}{n-1}\Big(n\nabla\widehat{L}(w_1;S) - \nabla\ell(w_1;z_n)\Big)$ and updating as $w_2' = w_1 - \eta\nabla\widehat{L}(w_1;S')$. However, in the second iteration (and onwards), the gradient is computed on w_2' which can be different from w_2 and the above *adjustment* can no longer be applied and one may need to retrain from here onwards. This captures a key challenge for unlearning in problems solved by *simple iterative procedures* such as GD – **adaptivity** – that is, the gradients (or more

generally, the *queries*) computed in later iteration depend on the result of the previous iterations. We systematically formalize such procedures and design efficient unlearning algorithms for them.

5.1.1 Results and Techniques

Learning/Unlearning as Query Release. Iterative procedures are an integral constituent of the algorithmic toolkit for solving machine learning problems and beyond. As in the case of GD above, these often consist of a sequence of *simple* but *adaptive* computations. The simple computations are often efficiently undo-able (as in the first iteration of GD) but its *adaptive* nature – change of result of one iteration changing the trajectory of the algorithm – makes it difficult to undo computation, or unlearn, efficiently.

As opposed to designing unlearning (and learning) algorithms for specific (machine learning) problems, we study the design of unlearning algorithms corresponding to (a class of) learning algorithms. We formalize this by considering learning algorithms which perform *adaptive query release* on datasets. Specifically, this consists of a selection of adaptive queries from structured classes like linear and prefix-sum queries (see Section 5.3 for details). The above example of GD is an instance of linear query, since the query, which is the average gradient $\frac{1}{n}\sum_{i=1}^{n}\nabla\ell(w_t; z_i)$, is a sum of functions of data-points. With this view, we study how to design *efficient* unlearning algorithms for such methods.

We use efficiency in the sense of number of queries made (query complexity), ignoring the use of other resources, e.g., space, computation for selection of queries, etc. To elaborate on why this is interesting, firstly note that this does not make the problem trivial, in the sense that even with unlimited access to other resources, it is still challenging do design an unlearning algorithm with query complexity smaller than that of retraining (the naive baseline). Secondly, let us revisit the motivation from solving

optimization problems. The standard model to measure computation in optimization is the number of gradient queries a method makes for a target accuracy, often abstracted in an oracle-based setup [NY83]. Importantly, this setup imposes no constraints on other resources, yet it witnesses the optimality of well-known simple procedures like (variants of) GD. We follow this paradigm, and as applications of our results to Stochastic Convex Optimization (SCO), we make progress on the fundamental question of understanding the gradient complexity of unlearning in SCO. Interestingly, our proposed unlearning procedures are simple enough that the improvement over retraining in terms of query complexity also applies even with accounting for the (arithmetic) complexity of all other operations in the learning and unlearning methods.

Linear Queries. The simplest query class we consider is that of linear queries (details deferred to Appendix D.2). Herein, we show that the prior work of [UMR+21], which focused on unlearning in SCO and was limited to the stochastic gradient method, can be easily extended to general linear queries. This observation yields unlearning algorithms for algorithms for *Federated Optimization/Learning* and *k-means clustering*. Herein, we give a ρ-TV stable (see Definition 15) learning procedure with T adaptive queries and a corresponding unlearning procedure with a $O(\sqrt{T}\rho)$ relative unlearning complexity (the ratio of unlearning and retraining complexity; see Definition 17).

Prefix-sum Queries. Our main contribution is the case when we consider the class of prefix-sum queries. These are a sub-class of interval queries which have been extensively studied in differential privacy and are classically solved by the binary tree mechanism [DNPR10]. We note in passing that for differential privacy, the purpose of the tree is to enable a tight privacy accounting and no explicit tree may be maintained. In contrast, for unlearning, we show that maintaining the binary tree data structure aids for efficient unlearning. We give a binary-tree based ρ-TV stable learning procedure and a corresponding unlearning procedure with a $\widetilde{O}(\rho)$ relative

unlearning complexity.

Unlearning in Stochastic Convex Optimization (SCO). Our primary motivation for considering prefix-sum queries is its application to unlearning in SCO (see Section 5.2 for preliminaries).

1) Smooth SCO. The problem of unlearning in smooth SCO was studied in [UMR+21], presented in Chapter 4, which proposed algorithms with excess population risk of $\widetilde{O}\left(\frac{1}{\sqrt{n}} + \left(\frac{\sqrt{d}}{n\rho}\right)^{2/3}\right)$ where ρ is the relative unlearning complexity. We show that using a variant of variance-reduced Frank-Wolfe [ZSM+20], which uses prefix-sum queries, yields an improved excess population risk of $O\left(\frac{1}{\sqrt{n}} + \frac{\sqrt{d}}{n\rho}\right)$. This corresponds to $\widetilde{O}(\rho n)$ expected gradient computations upon unlearning.

2) Non-smooth SCO. In the non-smooth setting, which was not covered in the prior works, we give an algorithm based on Dual Averaging [Nes09], which again uses prefix-sum query access, and thus fits into the framework. This algorithm gives us an excess population risk of $O\left(\frac{1}{\sqrt{n}} + \frac{d^{1/4}}{\sqrt{n\rho}}\right)$ with $\widetilde{O}(\rho n)$ expected gradient complexity of unlearning.

3) Generalized Linear Models (GLM). GLMs are one of most basic machine learning problems which include the squared loss (in linear regression), logistic loss (in logistic regression), hinge loss (support vector machines), etc. We study unlearning in two classes of GLMs (see below), for which we combine recently our proposed techniques based on dimensionality reduction [ABG+22], presented in Chapter 3, with the above prefix-sum query algorithms to get the following *dimension-independent rates*.

3(a) Smooth GLM. For the smooth convex GLM setting, we combine Johnson-Lindenstrauss transform with variance reduced Frank-Wolfe to get $O\left(\frac{1}{\sqrt{n}} + \frac{1}{(n\rho)^{2/3}}\right)$ excess population risk. Note that we get no overhead in statistical rate even with very small relative unlearning complexity, $\rho \approx n^{-1/4}$. This class of smooth GLMs contains the well-studied problem of logistic regression. Hence, our result demonstrates that it is possible to unlearn logistic regression with *sub-linear*, specifically $O(n^{3/4})$, unlearning complexity with no sacrifice in the statistical rate.

3(b) Lipschitz GLM. Similarly, for the Lipschitz convex GLM setting, we combine Johnson-Lindenstrauss transform with Dual Averaging yielding a rate of $\widetilde{O}\left(\frac{1}{\sqrt{n}} + \frac{1}{(n\rho)^{1/3}}\right)$.

Please see Table 5-I for a summary of above results.

SCO in dynamic streams. Finally, we consider SCO in dynamic streams where we observe a sequence of insertions and deletions and are supposed to produce outputs after each time-point. In this case, we present two methods: one which satisfies the exact unlearning guarantee with worse update time, the other which satisfies *weak unlearning* – which only requires the model (and not metadata) to be indistinguishable (see Definition 16) – with improved update time. The exact unlearning method is inspired from our work [UMR+21], presented in Chapter 4, which dealt with insertions similar to deletions. The weak unlearning method is motivated from the observation that the above may be too pessimistic. To elaborate, inserting a new data item does not warrant a (unlearning) guarantee that the algorithm's state be indistinguishable to the case if the point was not inserted. Hence, insertions should require smaller update time which is indeed the case for our proposed methods.

Problem	Base algorithm	Rate
Smooth, Lipschitz-SCO	VR-FW	$\frac{1}{\sqrt{n}} + \frac{\sqrt{d}}{n\rho}$
Lipschitz SCO	DA	$\frac{1}{\sqrt{n}} + \frac{d^{1/4}}{\sqrt{n\rho}}$
Smooth, Lipschitz GLM	JL + VR-FW	$\frac{1}{\sqrt{n}} + \frac{1}{(n\rho)^{2/3}}$
Lipschitz GLM	JL + DA	$\frac{1}{\sqrt{n}} + \frac{1}{(n\rho)^{1/3}}$

Table 5-I. Excess population risk guarantees for various problems as well as the base algorithm; ρ: relative unlearning complexity (see Definition 17), VR-FW: Variance-reduced Frank Wolfe, DA: Dual averaging, JL: Johnson-Lindenstrauss transform.

5.1.2 Related Work

This chapter is a direct follow up of [UMR+21], presented in Chapter 4, wherein we proposed the framework of Total Variation (TV) stability and maximal coupling for the **exact** machine unlearning problem. We recap that this was applied for unlearning in smooth stochastic convex optimization (SCO) yielding a guarantee of $\frac{1}{\sqrt{n}} + \left(\frac{\sqrt{d}}{n\rho}\right)^{\frac{2}{3}}$ on excess population risk, where n is the number of data samples, d, model dimensionality and ρ is the relative unlearning complexity (see Definition 17). We improve upon the results in the previous work in multiple ways as described in the preceding section.

Besides this, the exact unlearning problem has been studied for k-means clustering [GGVZ19] and random forests [BL21]. The work of [BCCC+21] proposes a general methodology for exact unlearning for deep learning methods. Their focus is to devise *practical methods* and they do not provide theoretical guarantees on accuracy, even in simple settings. Finally, there are works which consider unlearning in SCO, however they use an *approximate* notion of unlearning inspired from differential privacy [GGHVDM20, NRSM21b, SAKS21, GJN+21], and therefore are incomparable to our work.

5.2 Additional Preliminaries

We review the problem setup and preliminaries briefly. We consider the setting where we start with a dataset of n samples and observe **one** unlearning request. We assume that the choice of unlearning request is oblivious to the learning process.

In Section 5.6, we generalize our result to a streaming setting of requests. Herein, besides exact unlearning (Definition 11), we also consider a weaker notion, called *weak unlearning*, wherein only the model output and not the entire state is required to be indistinguishable. We provide its definition below.

Definition 16 (Weak unlearning)**.** *A procedure* $(\mathbf{A}, \mathbf{U})$ *satisfies weak unlearning if for all all datasets* S*, all* $z \in \mathcal{Z}$*, and for all measurable events* $\mathcal{E} \subseteq \mathcal{W} \times \mathcal{M}$*, we have,*

$$\mathbb{P}\left(\mathcal{A}\left(S \backslash \{z\}\right) \in \mathcal{E}\right) = \mathbb{P}\left(\mathcal{U}\left(\mathbf{A}(S), z\right) \in \mathcal{E}\right)$$

We briefly recap how the framework of *Total Variation stability* (see Definition 15) and *maximal coupling*, is used in our construction of efficient unlearning algorithms. We recall that, given two distributions P and Q, a **maximal coupling** is a coupling (see Definition 12) π such that the disagreement probability $\mathbb{P}_{(p,q)\sim\pi}\{p \neq q\} = \mathsf{TV}(P, Q)$. In the unlearning context, $P = \mathcal{A}(S)$, the output on initial dataset, and $Q = \mathcal{A}(S')$, the output on the updated dataset. Hence, the unlearning problem simply becomes about transporting P to Q with small *computational cost*, akin to optimal transport [Vil09].

When sampled from a maximal coupling between P and Q, by definition, we get the **same sample** for both P and Q, expect with probability ρ, and yet satisfying the exact unlearning criterion. The main idea is that for certain learning algorithms of interest, during unlearning, we can **efficiently** construct a (near) maximal coupling of P and Q, and so the same model output from P suffices for Q, most of the times. In particular, the fraction of times that we need change the model is (roughly) the TV-stability parameter ρ of the learning algorithm. The goal, therefore, is to design

Algorithm 11 Template learning algorithm

Input: Dataset S, steps T, query functions $\{q_t(\cdot)\}_{t\le T}$ where $q_t \in \mathcal{Q}$, a query class, update functions $\{U_t(\cdot)\}_{t\le T}$, selector function $\mathcal{S}(\cdot)$

1: Initialize model $w_1 \in \mathcal{W}$
2: **for** $t = 1$ to $T-1$ **do**
3: Query dataset $u_t = q_t\left(\{w_i\}_{i\le t}, S\right)$
4: Update $w_{t+1} = U_t(\{w_i\}_{i\le t}, u_t)$
5: **end for**

Output: $\widehat{w} = \mathcal{S}\left(\{w_t\}_{t\le T}\right)$

an (accurate) TV-stable learning algorithm and a corresponding efficient coupling-based unlearning algorithm. In this work, we use the technique of *reflection coupling* described in Section 1.2.4.1 in Chapter 1.

5.3 Unlearning for Adaptive Query Release

We now set up the framework of adaptive query release, which is a lens to view (existing) iterative learning procedures; this view is useful in our design of corresponding unlearning algorithms. Iterative procedures run on datasets consist of a sequence of *interactions* with the dataset; each interaction computes a certain function, or query, on the dataset. The chosen query is typically adaptive, i.e., dependent on the prior query outputs. We consider iterative learning procedures which are composed of adaptive queries from a specified query class. Formally, consider a query class $\mathcal{Q} \subseteq \mathcal{W}^{\mathcal{W}^*\times\mathcal{Z}^*}$; herein, each query in $\mathcal{Q}$ is a function of a sequence of $\{w_i\}_{i<t}$ (typically, prior query outputs), and the dataset S, with output in $\mathcal{W}$. With this view, we give a general template of a learning procedure as Algorithm 11, where $\{U_t\}_t$ and $\mathcal{S}$ are the update and selector functions internal to the algorithm.

Query model: We describe the query model which we use to measure computational complexity. Under the model, a query function $q(\{w\}_i, S)$ takes $|S|$ unit computations (or queries, for brevity) for any q and $\{w_i\}_i$. In our applications to SCO, this will

correspond to the gradient oracle complexity.

Our algorithmic approach to unlearning is rooted in the relationship between TV stability and maximal couplings. With this view, for a specified query class, we have the following requirements.

1. **TV-stability:** We want a ρ-TV stable "modification" of the learning Algorithm 11, in the sense that it responds to the queries (line 3) while satisfying TV stability.
2. **Efficient unlearning algorithm**: We measure efficiency as the average number of queries the unlearning algorithm makes relative to the learning algorithm (retraining), defined as follows.

   Definition 17 (Relative Unlearning Complexity)**.** *The Relative Unlearning Complexity is defined as,*

   $$\frac{\mathbb{E}_{(\mathbf{A},\mathbf{U})}\left[\text{Query complexity of unlearning algorithm } \mathbf{U}\right]}{\mathbb{E}_{\mathbf{A}}\left[\text{Query complexity of learning algorithm } \mathbf{A}\right]}$$

   For a ρ-TV stable learning algorithm, we want that the relative unlearning complexity is (close to) ρ. This is motivated from the relationship between maximal coupling and TV distance. In the following, our proposed unlearning algorithm constructs a (near) maximal coupling of the learning algorithm's output under the original and updated dataset. This means that unlearning algorithm changes the original output (under the original dataset) with probability at most ρ – in this case, the unlearning algorithm makes a number of queries akin to retraining. In the other case when it does change the output, it makes a small (ideally, constant) number of queries. The above imply that relative unlearning complexity is (close to) ρ.

   We note that relative unlearning complexity, in itself, does not completely capture if the unlearning algorithm is *good*, since it may be the case that the corresponding learning algorithm is computationally more expensive than other existing methods. However, in our applications to SCO (Section 5.5), our learning algorithms are

linear time, so the denominator, in the definition above, is as small as it can be (asymptotically), i.e. $\Theta(n)$.

3. **Accuracy:** We will primarily be concerned with correctness of the unlearning algorithm and its efficiency. In the applications (Section 5.5), we will give accuracy guarantees for specific problems, where we will see our proposed TV stable modified algorithms are still accurate.

5.4 Prefix-sum Queries

We now consider prefix-sum queries, which is the main contribution of this chapter. The reason for this choice is that two powerful (family of) algorithms for SCO, Dual Averaging and Recursive Variance Reduction based methods, fit into this template (detailed in Section 5.5). We start by defining a prefix-sum query.

Definition 18. *A set of queries* $\{q_t\}_{t\geq 1}$ *where* $q_t : \mathcal{W}^t \times \mathcal{Z}^n \to \mathcal{W}$ *are called prefix-sum queries if* $q_1(w_1, S) = p_1(w_1, z_1)$ *and for all* $t > 1$, $q_t(\{w_i\}_{i\leq t}, S) = q_{t-1}(\{w_i\}_{i<t}, S) + p_t\Big(\{w_i\}_{i\leq t}, z_t)\Big)$ *for some functions* $\{p_t\}_{t\geq 1}$ *where* $p_t : \mathcal{W}^* \times \mathcal{Z} \to \mathcal{W}$.

Simply put, prefix-sum queries, sequentially query **new** data points and adds them to the previous accumulated query. A simple example is computing partial sums of data points $(z_1, z_1 + z_2, \ldots)$. Note that in the above definition, we can equivalently represent the prefix-sum queries using the sequence $\{p_t\}_t$. We also assume that the queries have bounded sensitivity, defined as follows.

Definition 19. *A query* $q : \mathcal{W}^* \times \mathcal{Z}^n \to \mathcal{W}$ *is* B*-sensitive if*

$$\sup_{\{w_i\}_i} \sup_{S \sim_n S'} \|q(\{w_i\}_i, S) - q(\{w_i\}_i, S')\| \leq B.$$

We note that the bounded sensitivity condition is satisfied in a variety of applications; see Section 5.5.

5.4.1 Learning with Binary Tree Data-Structure

The learning algorithm, given as Algorithm 12, is based on answering the adaptive prefix-sum queries with the binary tree mechanism [DNPR10]. For n samples (assume n is a power of two, otherwise we can append dummy "zero" samples without any change in asymptotic complexity), the binary tree mechanism constructs a complete binary tree $\mathcal{T}$ with the leaf nodes corresponding to the data samples. The key idea in the binary tree mechanism is that instead of adding *fresh* independent noise to each prefix-sum query, it is *better* to add correlated noise, where the correlation structure is described by a binary tree. For example, suppose we want to release the **seventh** prefix-sum query, $\sum_{i=1}^{7} p_i(\{w_j\}_{j\le i}, z_i)$, then consider the dyadic decomposition of 7 as $4, 2$ and 1, and release the sum,

$$\Big(\sum_{i=1}^{4} p_i(\{w_j\}_{j\le i}, z_i) + \xi_1\Big) + \Big(\sum_{i=5}^{6} p_i(\{w_j\}_{j\le i}, z_i)\xi_2\Big) \\ + \Big(p_7(\{w_j\}_{j\le i}, z_i) + \xi_3\Big),$$

where ξ_i's denote the added noise, which may have also been used in prior prefix-sum query responses. See Figure 5-1 (left) for a simplified description of the process.

We index the nodes of the tree using using binary strings $B = \{0,1\}^{\log(n)}$ which describes the path from the root. Let the tree $\mathcal{T} = \{v_b\}_{b\in B}$ which denotes the contents stored by the learning algorithm. Herein, each node contains the tuple (u, r, w, z) where $u \in \mathbb{R}^d$ is the query response, $r \in \mathbb{R}^d$ is the *noisy* response, $w \in \mathbb{R}^d$ a model and $z \in \mathcal{Z}$ a data point. In fact, only the leaf nodes store the model and data sample. The size of the tree is the space complexity of the learning procedure. Finally, define $\mathsf{leaf} : [n] \to \{0,1\}^{\log(n)}$ which gives the binary representation of the input leaf node.

This binary tree data structure supports the following operations:

1. $\mathsf{Append}(u, \sigma; \mathcal{T})$: Add a new leaf to $\mathcal{T}$, which consists of setting its query response and noisy query response to u, and $u + \mathcal{N}(0, \sigma^2\mathbb{I})$ respectively. Further, update

tree to add u to u_b, corresponding to nodes v_b in the path from this leaf to root, and add noise to their noisy response r_b for nodes which are left child in the path.

2. $\mathsf{GetPrefixSum}(t; \mathcal{T})$, where $t \in \mathbb{N}$: Get the t-th noisy response from $\mathcal{T}$, which consists of traversing the tree from t-th leaf to root, and adding the noisy responses of nodes which are left child.

3. $\mathsf{Get}(b; \mathcal{T})$ where $b \in \{0,1\}^{\log(n)}$: Get all items in the vertex of $\mathcal{T}$ indexed by b.

4. $\mathsf{Set}(b, v; \mathcal{T})$ where $b \in \{0,1\}^{\log(n)}$: Set the contents of vertex b in the $\mathcal{T}$ as v.

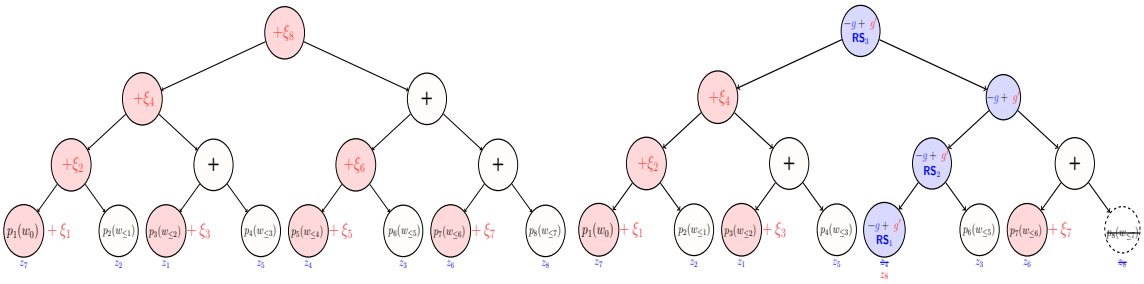

Figure 5-1. A simplified schematic of the learning (left) and unlearning (right) procedures for prefix-sum queries. In the left, the leaves contain (noisy, if $+\xi_i$) prefix-sum queries applied on the randomly permuted data-point (z_i's) below it. The intermediate nodes with $+$ adds the not-noised values of its children, where as others add noise to it. On the right, the deleted point z_4 is replaced with z_8 which amounts to adjusting the queries with $-g + g'$ (see Algorithm 13 for details) and performing Rejection Sampling (abbreviated $\mathbf{RS}_i$, where i's indicates the order of occurrence of sequence of rejection samplings) along the height of the tree.

Following [GTS13], we give pseudo-codes of the above operations in Section D.3, with minor modifications to aid the unlearning process.

5.4.2 Unlearning by Maximally Coupling Binary Trees

The unlearning Algorithm 13 is based on constructing a (near) maximal coupling of the binary trees under current and updated dataset. Let z_j be the element to be deleted and let v_s be the leaf node which contains z_j (we use z in place of z_j from here

Algorithm 12 $\mathsf{TreeLearn}(t_0; \mathcal{T})$

Input: Dataset S, steps T, B-sensitive prefix-sum queries $\{p_t\}_{t \le T}$, update functions $\{U_t\}_{t \le T}$, noise standard deviation σ

1: **if** $t_0 = 1$ **then** Permute dataset and initialize $\mathcal{T}$ **end if**
2: $(\cdot, \cdot, w_{t_0}, \cdot) = \mathsf{Get}(\mathsf{leaf}(t_0); \mathcal{T})$
3: **for** $t = t_0$ to $|S| - 1$ **do**
4: $\quad u_t = p_t(\{w_t\}_{i \le t}, z_t)$
5: $\quad \mathsf{Append}(u_t, \sigma; \mathcal{T})$
6: $\quad r_t = \mathsf{GetPrefixSum}(t; \mathcal{T})$
7: $\quad w_{t+1} = U_t\left(\{w_t\}_{\le t}, r_t\right)$
8: $\quad \mathsf{Set}(\mathsf{leaf}(t), (u_t, r_t, w_t, z_t); \mathcal{T})$
9: **end for**

Output: $\widehat{w} = \mathcal{S}\left(\{w_t\}_t\right)$

on, for simplicity). During unlearning, we simulate (roughly speaking) the dynamics of the learning algorithm if the deleted point was not present to begin with. In that case, in place of the deleted point, some other point would have been used. Now, since the dataset was randomly permuted, every point is equally likely to have been used, and thus we can use the point z' in the last leaf node, say v_l, in the tree – this choice of the last point is important for unlearning efficiency. Firstly, the computations associated with the last point z' needs to be undone – towards this, we update the contents of the nodes in the path from node v_l to root (line 5), finally removing node v_l from the tree (line 6). Then, we need to *replace* all the computations which used the deleted point z with the same computation under z'. Since the learning algorithm was based on the binary tree mechanism, the point z was only **explicitly** used in the nodes lying on the path from leaf v_s to the root (so, at most $\log(n)$ nodes). We say explicitly above because due to the adaptive nature of the process, in principle, all nodes after v_s depend on it, in the sense that their contents would change if the response in v_s were to change. However, importantly, the binary search structure of our learning algorithm and our coupling technique (details below) would enable us to (mostly) only care about explicit computations.

We first compute **two** new queries, under the data point z and z', with responses

Algorithm 13 TreeUnlearn

Input: z_j: data point to be deleted, $\mathcal{T}$: internal tree data-structure saved during learning

1: $s = \mathsf{leaf}(j)$ and $l = \mathsf{leaf}(|S|)$
2: $(\cdot, \cdot, w, z) = \mathsf{Get}(s; \mathcal{T})$ and $(\cdot, \cdot, \cdot, z') = \mathsf{Get}(l; \mathcal{T})$
3: $g = p_j(\{w_q\}_{q \le s}, z)$ and $g' = p_j(\{w_q\}_{q \le s}, z')$
4: Let $\mathsf{path} = \{l \to \cdots \to \text{root}\}$ be the path from l to root.
5: **for** $b \in \mathsf{path}$ **do** $u_b = u_b - g'$ **end for**
6: Remove node l from $\mathcal{T}$.
7: Let $b = s$ and $\text{ct} = 1$
8: **if** $j = |S|$ **then** let $b = \emptyset$ **end if**
9: **while** $b \neq \emptyset$ **do**
10: $(u, r, \cdot, \cdot) = \mathsf{Get}(b; \mathcal{T})$
11: $u' = u - g + g'$
12: **if** $\mathsf{Unif}\,(0, 1) \le \frac{\phi_{\mathcal{N}(u, \sigma^2\mathbb{I})}(r)}{\phi_{\mathcal{N}(u', \sigma^2\mathbb{I})}(r)}$ **then**
13: **if** $b = s$ **then** $\mathsf{Set}(b, (u', r, w, z')\,; \mathcal{T})$ **else then** $\mathsf{Set}(b, (u', r, \emptyset, \emptyset)\,; \mathcal{T})$ **end if**
14: **else**
15: $r' = \mathsf{Reflect}(u, u', r)$
16: **if** $b = s$ **then**
17: $\mathsf{Set}(b; (u', r', \emptyset, z')\,; \mathcal{T})$
18: $w' = U_j\left(\{w_q\}_{q \le b}, \mathsf{GetPrefixSum}(j; \mathcal{T})\right)$
19: $\mathsf{Set}(b, (u', r', w', z')\,; \mathcal{T})$
20: **else**
21: $\mathsf{Set}(b, (u', r', \emptyset, \emptyset)\,; \mathcal{T})$
22: **end if**
23: $\mathsf{TreeLearn}(j + \text{ct}; \mathcal{T})$ *// Continue Retraining*
24: **break**
25: **end if**
26: **if** b is left sibling **then** $\text{ct} = \text{ct} + 2^{|s|-|b|-1}$ **end if**
27: Set (new) b as binary representation of parent of b
28: **end while**
29: Update dataset $S = S \setminus \{z_j\}$

Output: $\widehat{w} = \mathcal{S}(\{w_b\}_b)$

$g = p_j(\{w_q\}_{q \le s}, z)$ and $g' = p_j(\{w_q\}_{q \le s}, z')$ respectively (line 3). Starting with leaf node v_s, we update the original unperturbed prefix-sum query response under z i.e. u to what it would have been under data-point z': $u' = u - g' + g$ (line 11). Further, since the training method adds noise $\mathcal{N}(0, \sigma^2\mathbb{I})$ to u to produce original noisy response r, we now need to produce a sample from $\mathcal{N}(u', \sigma^2\mathbb{I})$ to satisfy exact unlearning. Naively, we could simply get a *fresh* independent sample from $\mathcal{N}(u', \sigma^2\mathbb{I})$, however,

this would change the noisy response r, and hence require all subsequent computations to be redone (the adaptive nature). So, ideally, we want to reuse the same r and yet generate a sample from $\mathcal{N}(u', \sigma^2\mathbb{I})$. This is precisely the problem of constructing a maximal coupling, discussed in the Section 5.2, wherein we also discussed the method of reflection coupling to do it.

This amounts to doing a rejection sampling which (roughly) ascertains if response r is still sufficient under the new distribution $\mathcal{N}(u', \sigma^2\mathbb{I})$. Specifically we compute the ratio of the probability densities at r under the noise added to u and u', i.e. $\frac{\phi_{\mathcal{N}(u,\sigma^2\mathbb{I})}(r)}{\phi_{\mathcal{N}(u',\sigma^2\mathbb{I})}(r)}$ and compare it against a randomly sampled Unif(0,1); if it results in accept, we move to parent of the node v_s, and repeat. If any step fails, we reflect which generates a different noisy response r', and continue retraining from the next leaf w.r.t. the post order traversal of the tree (the variable ct in Algorithm 13 keeps track of this *next* node). See Figure 5-1 for a simplified description of the process.

The main result of this section is as follows.

Theorem 25. *The following are true for Algorithms 12 and 13.*

1. *The learning Algorithm 12 with* $\sigma^2 = \frac{64B^2 log^2(n)}{\rho}$ *satisfies* ρ*-TV stability.*
2. *The corresponding unlearning Algorithm 13 satisfies exact unlearning.*
3. *The relative unlearning complexity is* $\widetilde{O}(\rho)$

As discussed in the preceding section, in the Theorem above, we have all the properties we needed with the unlearning process. We now move on to applications and give accuracy guarantees.

5.5 Applications

In the following, we describe some problems and learning algorithms. The corresponding unlearning algorithms and its correctness simply follow as application of the

result of the preceding section, provided we show that it uses a bounded sensitivity prefix-sum query. The only other thing to show is the accuracy guarantee of the TV stable modification of the learning algorithm (Algorithm 12).

From here on, we use runtime to mean gradient complexity as is standard in convex optimization [NY83]. But, as pointed out before, our proposed unlearning algorithm yields similar improvements over retraining, even accounting for other operations in the method.

5.5.1 Smooth SCO with Variance Reduced Frank-Wolfe

We assume that the loss function $w \mapsto \ell(w; z)$ is H-smooth and G-Lipschitz for all z[1]. The algorithm we use is variance reduced Frank-Wolfe method where the variance reduced gradient estimate u_t is the Hybrid-SARAH estimate [TDPPN19] with $\gamma_t = \frac{1}{t+1}$ given as,

$$\begin{aligned} u_t &= (1-\gamma_t)\left(u_{t-1} + \nabla\ell(w_t; z_t) - \nabla\ell(w_{t-1}; z_t)\right) + \gamma_t \nabla\ell(w_t; z_t) \\ &= \frac{1}{t+1}\sum_{i=1}^{t}\left((i+1)\nabla\ell(w_i; z_i) - i\nabla\ell(w_{i-1}; z_i)\right) \end{aligned}$$

We show that the above is a prefix sum query with sensitivity $B = 2(HD + G)$, thus fits into our framework. The full pseudo-code is given as Algorithm 33 in Appendix D.5. We state the main result below where the accuracy guarantee follows from modifications to the analysis in [ZSM+20].

Theorem 26. *Let $\rho \leq 1$ and $\ell : \mathcal{W} \times \mathcal{Z} \to \mathbb{R}$ be an H-smooth, G-Lipschitz convex function over a closed convex set $\mathcal{W}$ of diameter D. Algorithm 33, as the learning algorithm, run with $\sigma^2 = \frac{64(HD+G)^2 log^2(n)}{\rho^2}$, $t_0 = 1$ and $\eta_t = \frac{1}{t+1}$ on a dataset S of n i.i.d. samples from $\mathcal{D}$ outputs $\widehat{w}$, with excess population risk bounded as,*

$$\mathbb{E}\left[L(\widehat{w}; \mathcal{D}) - L(w^*; \mathcal{D})\right] = \widetilde{O}\left((G + HD)\, D\left(\frac{1}{\sqrt{n}} + \frac{\sqrt{d}}{n\rho}\right)\right).$$

[1] A real valued function $x \mapsto f(x)$ is G-Lipschitz and H-smooth if $|f(x_1) - f(x_2)| \leq G\|x_1 - x_2\|$ an $\|\nabla f(x_1) - \nabla f(x_2)\| \leq H\|x_1 - x_2\|$ respectively.

Furthermore, the corresponding unlearning Algorithm 13 (with query and update functions as specified in the learning algorithm), satisfies exact unlearning with $\widetilde{O}(\rho n)$ expected runtime.

5.5.2 Non-smooth SCO with Dual Averaging

In this section, we only assume that loss function $w \mapsto \ell(w; z)$ is G-Lipschitz and convex $\forall\ z \in \mathcal{Z}$. Herein, we use dual averaging method [Nes09] where the model is updated as follows:

$$w_{t+1} = \Pi_{\mathcal{W}}\Big(w_0 - \eta \sum_{i=1}^{t} \nabla \ell(w_i; z_i)\Big),$$

where Π denotes the Euclidean projection on to the convex set $\mathcal{W}$. The above again is a prefix-sum query with sensitivity G, thus fits into our framework. The full pseudo-code is given as Algorithm 34 in Appendix D.5. The accuracy guarantee mainly follows from [KMS+21].

Theorem 27. *Let $\rho \leq 1$ and $\ell : \mathcal{W} \times \mathcal{Z} \to \mathbb{R}$ be a G-Lipschitz convex function over a closed convex set $\mathcal{W}$ of diameter D. Algorithm 34, as the learning algorithm, run with $\sigma^2 = \frac{64G^2 log^2(n)}{\rho^2}$, $t_0 = 1$ and $\eta = \frac{Dd^{1/4}\sqrt{\log(n)}}{G\sqrt{n\rho}}$ on a dataset S of n samples, drawn i.i.d. from $\mathcal{D}$, outputs $\widehat{w}$ with excess population risk bounded as,*

$$\mathbb{E}\left[L(\widehat{w}; \mathcal{D}) - L(w^*; \mathcal{D})\right] = \widetilde{O}\left(GD\left(\frac{1}{\sqrt{n}} + \sqrt{\frac{\sqrt{d}}{n\rho}}\right)\right).$$

Furthermore, the corresponding unlearning Algorithm 13 (with query and update functions as specified in the learning algorithm), satisfies exact unlearning with $\widetilde{O}(\rho n)$ expected runtime.

5.5.3 Convex GLM with JL Method

This JL method, proposed in our work [ABG+22], covered in Chapter 3, is a general technique to get dimension-independent rates for unconstrained convex GLMs

Algorithm 14 JL Method

Input: Dataset S, loss function ℓ, base algorithm $\mathcal{A}$, JL matrix $\Phi \in \mathbb{R}^{d\times k}$, noise variance σ^2

1: $\Phi S = \{\Phi x_i\}_{i=1}^n$
2: $\widetilde{w} = \mathcal{A}(\ell, \Phi S, 2G\|\mathcal{X}\|, 2H\|\mathcal{X}\|^2, \sigma)$

Output: $\widehat{w} = \Phi^\top \widetilde{w}$

from algorithms giving dimension-dependent rate for constrained (general) convex losses. The method, described in Algorithm 14, simply embeds the dataset into a low dimensional space, via a JL matrix Φ, and then runs a base algorithm on the low dimensional dataset.

Smooth, Lipschitz GLMs: We assume that $\phi_y : \mathbb{R} \to \mathbb{R}$ is convex, H-smooth and G-Lipschitz for all $y \in \mathcal{Y}$. We give the following result in this case using VR-Frank Wolfe as the base algorithm.

Theorem 28. *Let $\rho \le 1$ and $\ell : \mathcal{W}\times\mathcal{X}\times\mathcal{Y} \to \mathbb{R}$ be an H-smooth, G-Lipschitz convex GLM loss function. Algorithm 14 instantiated with Algorithm 33, as the learning algorithm, run with $\sigma^2 = \widetilde{O}\left(\frac{\left(H\|\mathcal{X}\|^2\|w^*\|+G\|\mathcal{X}\|\right)^2}{\rho^2}\right)$, $t_0 = 1$, $\eta_t = \frac{1}{t+1}$ and $k = \widetilde{O}\left(\left(\frac{H\|\mathcal{X}\|^2\|w^*\|}{\left(H\|\mathcal{X}\|^2\|w^*\|+G\|\mathcal{X}\|\right)}\right)^{2/3}(n\rho)^{2/3}\right)$ on a dataset S of n samples, drawn i.i.d. from $\mathcal{D}$, outputs $\widehat{w}$ with excess population risk bounded as,*

$$\mathbb{E}\left[L(\widehat{w};\mathcal{D}) - L(w^*;\mathcal{D})\right] = \widetilde{O}\left(\frac{\left(G\|\mathcal{X}\| + H\|\mathcal{X}\|^2\|w^*\|\right)\|w^*\|}{\sqrt{n}} + \frac{H^{1/3}G^{2/3}\|w^*\|^{4/3}\|\mathcal{X}\|^{4/3} + H\|\mathcal{X}\|^2\|w^*\|^2}{(n\rho)^{2/3}}\right).$$

Furthermore, the corresponding unlearning Algorithm 13 (with query and update functions as specified in the learning algorithm), satisfies exact unlearning with $\widetilde{O}(\rho n)$ expected runtime .

Lipschitz GLMs: We assume that $\phi_y : \mathbb{R} \to \mathbb{R}$ is convex and G-Lipschitz for all $y \in \mathcal{Y}$. We give the following result in this case using Dual Averaging as the base

algorithm.

Theorem 29. *Let $\rho \leq 1$ and $\ell : \mathcal{W} \times \mathcal{X} \times \mathcal{Y} \to \mathbb{R}$ be a G-Lipschitz convex GLM loss function. Algorithm 14 with Algorithm 34 as the sub-routine, as the learning algorithm, run with $\sigma^2 = O\left(\frac{G^2\|\mathcal{X}\|^2}{\rho^2}\right)$, $t_0 = 1$, $\eta = \frac{\|w^*\| d^{1/4}\sqrt{\log(n)}}{G\|\mathcal{X}\|\sqrt{n\rho}}$ and $k = \sqrt{n\rho}$ on a dataset S of n samples sampled i.i.d. from $\mathcal{D}$ outputs $\widehat{w}$, with excess population risk bounded as,*

$$\mathbb{E}\left[L(\widehat{w}; \mathcal{D}) - L(w^*; \mathcal{D})\right] = \widetilde{O}\left(G\,\|\mathcal{X}\|\,\|w^*\|\left(\frac{1}{\sqrt{n}} + \frac{1}{(n\rho)^{1/3}}\right)\right).$$

Furthermore, the corresponding unlearning Algorithm 13 (with query and update functions as specified in the learning algorithm), satisfies exact unlearning with $\widetilde{O}(\rho n)$ expected runtime.

5.6 SCO in Dynamic Streams

In this section, we extend our previous results to dynamic streams wherein we observe a sequence of insertions and deletions, starting with potentially zero data points. We assume that the number of available points throughout is positive and the data points are i.i.d. from an an unknown distribution as well as the requests are chosen independent of the algorithm.

To give a simple and unified presentation, let the accuracy, say expected excess population risk, of the ρ-TV stable Algorithm 12 with a dataset S be denoted as, $\alpha(\rho, |S|\,; \mathcal{P})$ where $\mathcal{P}$ denotes problem specific parameters such as Lipschitzness, diameter etc.

We present two techniques for dynamic streams; one of them satisfies exact unlearning but has a worse update time; this is similar to [UMR+21], covered in Chapter 4, and is deferred to Section D.6. The other, presented below, satisfies weak unlearning (see Definition 16) with better update time. A key component to both are *anytime* guarantees, which hold at every time-point in the stream, for any length of the stream.

Anytime binary tree mechanism. In the previous section, the depth of the initialized tree and the noise variance σ^2, both were chosen as a function of the dataset size n. However, the tree can be easily built in an online manner as in prior work of [GTS13]. For setting the noise variance: for target ρ-TV stability, we distribute the noise budget exponentially along the height of the tree; specifically, the leaf node contribute to $\rho/2$ TV stability, the nodes above them $\rho/4$ and so on. In this way, the final tree satisfies ρ-TV stability for any value of n.

Anytime accuracy. The other problem of changing data size is that the internal parameters of algorithm (step size, in our case) may be set as a function of n for desirable accuracy guarantees. Fortunately, the two algorithms that we consider, VR-Frank Wolfe and Dual Averaging, have known horizon-oblivious parameter settings [Ora19]. Their JL counterparts on the other hand, require setting the embedding dimension as a function of n, and thus not applicable unless we assume that the number of data points throughout the stream is $\Theta(n)$.

5.6.1 Weak Unlearning in Dynamic Streams

We first argue in what way insertions handled in [UMR+21] is deficient. The main reason is that they require insertions to also satisfy the unlearning criterion: the state of the system upon insertion is instinguishable to the state had the inserted point being present to begin with. However, this is an overkill; adding new points simply serve to yield improved statistical accuracy. Furthermore, methods which allow adding new points, are abound, particularly in the stochastic optimization setting, sometimes known as *incremental* methods. Importantly, in most cases, the insertion time of these methods is constant (in n). Hence, a natural question is whether, for dynamic streams, can we design unlearning methods in which we pay for update time only in proportion to the number of deletions? Our result shows that we can, albeit under

the weak unlearning (see Definition 16) guarantee.

Specifically, our procedure requires *hiding* the order in which data points are processed. Intuitively, an incremental method typically processes the newest data point the last. This ordering is problematic to our unlearning procedure, since if some point is to deleted, then we can no longer replace it with the last point, as we did before, since that would result in a different order. Our main result is as follows.

Theorem 30. *In the dynamic streaming setting with R requests, using anytime incremental learning and unlearning algorithms, Algorithm 12 and 13, without permuting the dataset, the following are true.*

1. *It satisfies weak unlearning at every time point in the stream.*
2. *The accuracy of the output $\widehat{w}_i$ at time point i, with corresponding dataset S_i, is*

$$\mathbb{E}[L(\widehat{w}_i; \mathcal{D})] - \min_w L(w; \mathcal{D}) = \alpha(\rho, |S_i|; \mathcal{P})$$

3. *The number of times retraining is triggered, for V unlearning requests, is at most $\widetilde{O}(\rho V)$*

Importantly, in the above guarantee, we only pay for the number of unlearning requests V rather than the number of requests R.

5.7 Conclusion

In this chapter, we proposed a general framework for designing unlearning algorithms for learning algorithms which can be viewed as performing adaptive query release on datasets. We applied this to yield improved guarantees for unlearning in various settings of stochastic convex optimization. Our results are obtained by instantiating it with the class of prefix-sum queries and linear queries.

Chapter 6

Conclusion

In this dissertation, we studied machine learning problems under constraints motivated by modern considerations of data privacy, through the lens of stochastic optimization. In Chapters 2 and 3, we studied differentially private convex generalized linear models and non-convex optimization respectively. In Chapters 4 and 5, we studied machine unlearning in multiple standard settings of stochastic convex optimization.

6.1 Ongoing and Future Work

This section provides a brief overview of some ongoing and future work, motivated from investigations in this dissertation.

(Private) Non-convex Optimization. In Chapter 3, we studied the design of differentially private algorithms for approximating stationary points of non-convex empirical and population risk. We presented new upper and lower bounds for the problem, which match in some, admittedly restrictive, regime of problem parameters. A natural direction is to close this gap in general. A related and important question is the *same problem* in the *non-private* setting. In particular, as of yet, the (optimal) sample complexity of approximating stationary points, in the population setting, is unknown. Moreover, this gap in our understanding essentially percolates to gaps in the private setting. Thus, progress in the non-private setting is crucial to resolve the

problem in the private setting.

Private Learning with Public Data. There is by now a rich literature (some of them discussed in this dissertation) of algorithms and complexity of statistical learning tasks under differential privacy. In many cases, there is an additional price, due to privacy, in sample complexity. An example, discussed in Chapters 1 and 2, is that of *constrained (convex) GLMs* in d dimensions: herein, the optimal rate, without privacy, with n samples, is $\Theta\left(\frac{1}{\sqrt{n}}\right)$, which is *dimension-independent*. However, under (ϵ, δ) DP, the optimal rate turns out to be *dimension-dependent*, $\Theta\left(\frac{1}{\sqrt{n}} + \frac{\sqrt{d}}{n\epsilon}\right)$ [BST14].

Such pessimistic results have motivated a hybrid model, where a *small* public dataset is leveraged to circumvent these limitations. Practically, this public data could originate from *opt-in users* who consent to minimal or no privacy assurances, or from a *related* dataset that is publicly available. This model has been explored for the *Probability Approximately Correct (PAC)* learning setting and statistical inference problems, where it has been shown to yield non-trivial improvements [ABM19, BCM+20, BKS22]. A natural question is investigating (various settings of) differentially private stochastic (convex) optimization with public data. This involves identifying settings and approaches where a (small) public dataset can be effectively used to get around the known barriers.

Lower Complexity Bounds for Machine Unlearning. The current thrust of most research in (exact) machine unlearning is towards proposing algorithms with (non-trivial) guarantees in comparison to (any) retraining. Indeed, this is the nature of results presented in Chapters 4 and 5 of this dissertation. However, in order to truly assess the quality of (existing) procedures, we need to establish corresponding *lower bounds* which tells us how far we are from what is achievable. A natural direction is to study the limitations of (exact) unlearning in the stochastic convex optimization,

borrowing the language and tools from information-based complexity of optimization. Concretely, the goal is to characterize the *Pareto frontier* of optimal unlearning runtime and accuracy, among all procedures which satisfy unlearning correctness and have optimal learning runtime (for the said accuracy).

6.2 Other Works

Besides the works presented in this dissertation, I have been involved in other projects, which I will briefly summarize.

In [MUA+23] (*to appear to ALT 2024*), we studied differentially non-convex optimization under the *Kurdyka-Łojasiewicz (KL)* condition. This is a widely studied class of non-convex functions which admit efficient (non-private) procedures with global optimality guarantees. We designed differentially private procedures for such settings and show that they achieve near-optimal rates.

In [UCKO23] (published in *ICML 2023*), we considered the *Federated Learning (FL)* setting, wherein we have a (large) number of *clients* (user devices such as mobile phones, typically), containing user data, and a typical goal is to train a *shared* machine learning model. Two important desiderata in an FL setting are communication-efficiency and privacy. These are motivated from the fact that the clients usually communicate over a slow medium, and may have sensitive data. We studied the design of procedures which are differentially private, communication-efficient, and importantly require minimal *tuning* of any *hyper-parameter* related to the size of messages communicated. This work was preceded by [IRU+19] (published in *NeurIPS 2019*) and [RPU+20] (published in *ICML 2020*) where we explored the use of *sketching* techniques to design communication-efficient algorithms in (non-private) distributed and federated settings, respectively.

In [WUMA22] (published in *NeurIPS 2022*), we studied the *adversarial robustness*

problem, where at prediction-time, the test data point can be slightly, but adversarially, perturbed. We showed that all *two-layer ReLU neural networks* in the so-called *lazy regime* (which is the dominant model in theoretical deep learning) are susceptible to such attacks. In [WUNTA23] (published in *NeurIPS 2023*), we considered the *multi-task representation learning* setting, where a complex *representation* is learnt from multiple (data-abundant) source tasks, which then is used to learn a simple predictor (on top of the representation) on a (data-scarce) target task. Under a standard assumption of *task diversity* , we derived *optimistic rates*, those which interpolate between the *slow* $n^{-1/2}$ and *fast* n^{-1} rates, depending on the *easiness* of the tasks.

In [UMMA18] (published in *NeurIPS 2018*), we studied the design and analysis of algorithms for *kernel Principal Component Analysis* with *random Fourier features*. This was followed by [UA22] (published in *TMLR 2022*) where we established improved generalization bounds for the regularized empirical risk minimization rule for the problem of *kernel Canonical Correlation Analysis*.

In [BLUZ19] (published in *APPROX 2019*), we delved into a streaming setting, wherein we designed *low-space* algorithms for clustering and *coreset* construction in *time-decay streams*. This was followed by [ULAB22] (published in *TMLR 2022*) where we considered the problem of clustering in data streams, and *coreset* construction, under access to an *approximate nearest neighbour* search oracle.

Bibliography

[ABG+22] Raman Arora, Raef Bassily, Cristóbal A Guzmán, Michael Menart, and Enayat Ullah. Differentially private generalized linear models revisited. In *Advances in Neural Information Processing Systems*, 2022.

[ABG+23] Raman Arora, Raef Bassily, Tomás González, Cristóbal A Guzmán, Michael Menart, and Enayat Ullah. Faster rates of convergence to stationary points in differentially private optimization. In *International Conference on Machine Learning*, pages 1060–1092. PMLR, 2023.

[ABM19] Noga Alon, Raef Bassily, and Shay Moran. Limits of private learning with access to public data. *Advances in neural information processing systems*, 32, 2019.

[Abo18] John M Abowd. The us census bureau adopts differential privacy. In *Proceedings of the 24th ACM SIGKDD international conference on knowledge discovery & data mining*, pages 2867–2867, 2018.

[ACD+19] Yossi Arjevani, Yair Carmon, John C. Duchi, Dylan J. Foster, Nathan Srebro, and Blake Woodworth. Lower bounds for non-convex stochastic optimization, 2019.

[ACG+16a] Martin Abadi, Andy Chu, Ian Goodfellow, H Brendan McMahan, Ilya Mironov, Kunal Talwar, and Li Zhang. Deep learning with differential

privacy. In *Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security*, pages 308–318, 2016.

[ACG+16b] Martin Abadi, Andy Chu, Ian Goodfellow, H Brendan McMahan, Ilya Mironov, Kunal Talwar, and Li Zhang. Deep learning with differential privacy. In *Proceedings of the 2016 ACM SIGSAC conference on computer and communications security*, pages 308–318, 2016.

[ACKL21] Idan Amir, Yair Carmon, Tomer Koren, and Roi Livni. Never go full batch (in stochastic convex optimization). *Advances in Neural Information Processing Systems*, 34, 2021.

[AFKT21] Hilal Asi, Vitaly Feldman, Tomer Koren, and Kunal Talwar. Private stochastic convex optimization: Optimal rates in l1 geometry. In *International Conference on Machine Learning*, pages 393–403. PMLR, 2021.

[AKL21] Idan Amir, Tomer Koren, and Roi Livni. Sgd generalizes better than gd (and regularization doesn't help). In *Proceedings of Thirty Fourth Conference on Learning Theory*, volume 134 of *Proceedings of Machine Learning Research*, pages 63–92. PMLR, 2021.

[AWBR09] Alekh Agarwal, Martin J Wainwright, Peter Bartlett, and Pradeep Ravikumar. Information-theoretic lower bounds on the oracle complexity of convex optimization. *Advances in Neural Information Processing Systems*, 22, 2009.

[AZ17] Zeyuan Allen-Zhu. Katyusha: The first direct acceleration of stochastic gradient methods. *The Journal of Machine Learning Research*, 18(1):8194–8244, 2017.

[AZ18] Zeyuan Allen-Zhu. How to make the gradients small stochastically: Even faster convex and nonconvex sgd. In *Advances in Neural Information Processing Systems*, pages 1157–1167, 2018.

[BBG18] Borja Balle, Gilles Barthe, and Marco Gaboardi. Privacy amplification by subsampling: Tight analyses via couplings and divergences. In *Advances in Neural Information Processing Systems*, pages 6277–6287, 2018.

[BBL03] Olivier Bousquet, Stéphane Boucheron, and Gábor Lugosi. Introduction to statistical learning theory. In *Summer school on machine learning*, pages 169–207. Springer, 2003.

[BCCC+21] Lucas Bourtoule, Varun Chandrasekaran, Christopher A Choquette-Choo, Hengrui Jia, Adelin Travers, Baiwu Zhang, David Lie, and Nicolas Papernot. Machine unlearning. In *2021 IEEE Symposium on Security and Privacy (SP)*, pages 141–159. IEEE, 2021.

[BCM+20] Raef Bassily, Albert Cheu, Shay Moran, Aleksandar Nikolov, Jonathan Ullman, and Steven Wu. Private query release assisted by public data. In *International Conference on Machine Learning*, pages 695–703. PMLR, 2020.

[BDRS18] Mark Bun, Cynthia Dwork, Guy N. Rothblum, and Thomas Steinke. Composable and versatile privacy via truncated cdp. In *Proceedings of the 50th Annual ACM SIGACT Symposium on Theory of Computing*, STOC 2018, page 74–86, New York, NY, USA, 2018. Association for Computing Machinery.

[BE02] Olivier Bousquet and André Elisseeff. Stability and generalization. *Journal of machine learning research*, 2(Mar):499–526, 2002.

[BFGT20] Raef Bassily, Vitaly Feldman, Cristóbal Guzmán, and Kunal Talwar. Stability of stochastic gradient descent on nonsmooth convex losses. *Advances in Neural Information Processing Systems*, 33, 2020.

[BFTGT19] Raef Bassily, Vitaly Feldman, Kunal Talwar, and Abhradeep Guha Thakurta. Private stochastic convex optimization with optimal rates. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019.

[BFTT19] Raef Bassily, Vitaly Feldman, Kunal Talwar, and Abhradeep Guha Thakurta. Private stochastic convex optimization with optimal rates. In *Advances in Neural Information Processing Systems*, pages 11282–11291, 2019.

[BGM21] Raef Bassily, Cristóbal Guzmán, and Michael Menart. Differentially private stochastic optimization: New results in convex and non-convex settings. *Advances in Neural Information Processing Systems*, 34, 2021.

[BGN21] Raef Bassily, Cristobal Guzman, and Anupama Nandi. Non-euclidean differentially private stochastic convex optimization. In Mikhail Belkin and Samory Kpotufe, editors, *Proceedings of Thirty Fourth Conference on Learning Theory*, volume 134 of *Proceedings of Machine Learning Research*, pages 474–499. PMLR, 15–19 Aug 2021.

[BH79] Jean Bretagnolle and Catherine Huber. Estimation des densités: risque minimax. *Zeitschrift für Wahrscheinlichkeitstheorie und verwandte Gebiete*, 47(2):119–137, 1979.

[BKS22] Alex Bie, Gautam Kamath, and Vikrant Singhal. Private estimation

with public data. *Advances in Neural Information Processing Systems*, 35:18653–18666, 2022.

[BL21] Jonathan Brophy and Daniel Lowd. Machine unlearning for random forests. In *International Conference on Machine Learning*, pages 1092–1104. PMLR, 2021.

[BLM13] Stéphane Boucheron, Gábor Lugosi, and Pascal Massart. *Concentration inequalities: A nonasymptotic theory of independence*. Oxford university press, 2013.

[BLUZ19] Vladimir Braverman, Harry Lang, Enayat Ullah, and Samson Zhou. Improved algorithms for time decay streams. *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques*, 2019.

[BMR21] Peter L Bartlett, Andrea Montanari, and Alexander Rakhlin. Deep learning: a statistical viewpoint. *Acta numerica*, 30:87–201, 2021.

[BNS+16] 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*, pages 1046–1059, 2016.

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

[Can22] Clément L Canonne. A short note on an inequality between kl and tv. *arXiv preprint arXiv:2202.07198*, 2022.

[CDHS17] Yair Carmon, John C. Duchi, Oliver Hinder, and Aaron Sidford. "convex until proven guilty": Dimension-free acceleration of gradient descent on non-convex functions. In *Proceedings of the 34th International Conference on Machine Learning - Volume 70*, ICML'17, page 654–663. JMLR.org, 2017.

[CH12] Kamalika Chaudhuri and Daniel J. Hsu. Convergence rates for differentially private statistical estimation. In *Proceedings of the 29th International Conference on Machine Learning, ICML 2012, Edinburgh, Scotland, UK, June 26 - July 1, 2012*. icml.cc / Omnipress, 2012.

[Chu91] Alexander Mikhailovich Chudnov. Game-theoretical problems of synthesis of signal generation and reception algorithms. *Problemy Peredachi Informatsii*, 27(3):57–65, 1991.

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

[CO19] Ashok Cutkosky and Francesco Orabona. Momentum-based variance reduction in non-convex sgd. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019.

[Coh16] Michael B Cohen. Nearly tight oblivious subspace embeddings by trace inequalities. In *Proceedings of the twenty-seventh annual ACM-SIAM symposium on Discrete algorithms*, pages 278–287. SIAM, 2016.

[CTW+21] Nicholas Carlini, Florian Tramer, Eric Wallace, Matthew Jagielski,

Ariel Herbert-Voss, Katherine Lee, Adam Roberts, Tom Brown, Dawn Song, Ulfar Erlingsson, et al. Extracting training data from large language models. In *30th USENIX Security Symposium (USENIX Security 21)*, pages 2633–2650, 2021.

[CWH20] Xiangyi Chen, Steven Z. Wu, and Mingyi Hong. Understanding gradient clipping in private sgd: A geometric perspective. In H. Larochelle, M. Ranzato, R. Hadsell, M. F. Balcan, and H. Lin, editors, *Advances in Neural Information Processing Systems*, volume 33, pages 13773–13782. Curran Associates, Inc., 2020.

[CWZ21] T. Tony Cai, Yichen Wang, and Linjun Zhang. The cost of privacy: Optimal rates of convergence for parameter estimation with differential privacy. *The Annals of Statistics*, 49(5):2825 – 2850, 2021.

[CY15] Yinzhi Cao and Junfeng Yang. Towards making systems forget with machine unlearning. In *2015 IEEE Symposium on Security and Privacy*, pages 463–480. IEEE, 2015.

[DG23] Jelena Diakonikolas and Cristóbal Guzmán. Complementary composite minimization, small gradients in general norms, and applications, 2023.

[DH12] Frank Den Hollander. Probability theory: The coupling method. *Lecture notes available online (http://websites. math. leidenuniv. nl/probability/lecturenotes/CouplingLectures. pdf)*, 2012.

[DMNS06] Cynthia Dwork, Frank McSherry, Kobbi Nissim, and Adam Smith. Calibrating noise to sensitivity in private data analysis. In *Theory of cryptography conference*, pages 265–284. Springer, 2006.

[DMR18] Luc Devroye, Abbas Mehrabian, and Tommy Reddad. The total

variation distance between high-dimensional gaussians. *arXiv preprint arXiv:1810.08693*, 2018.

[DNPR10] Cynthia Dwork, Moni Naor, Toniann Pitassi, and Guy N Rothblum. Differential privacy under continual observation. In *Proceedings of the forty-second ACM symposium on Theory of computing*, pages 715–724, 2010.

[DR+14] Cynthia Dwork, Aaron Roth, et al. The algorithmic foundations of differential privacy. *Foundations and Trends in Theoretical Computer Science*, 9(3-4):211–407, 2014.

[Duc16] John Duchi. Lecture notes for statistics 311/electrical engineering 377. *URL: https://stanford. edu/class/stats311/Lectures/full_notes. pdf. Last visited on*, 2:23, 2016.

[EKRR19] Philip A Ernst, Wilfrid S Kendall, Gareth O Roberts, and Jeffrey S Rosenthal. Mexit: Maximal un-coupling times for stochastic processes. *Stochastic Processes and their Applications*, 129(2):355–380, 2019.

[Fel16] Vitaly Feldman. Generalization of erm in stochastic convex optimization: The dimension strikes back. *Advances in Neural Information Processing Systems*, 29, 2016.

[FKT20a] Vitaly Feldman, Tomer Koren, and Kunal Talwar. Private stochastic convex optimization: optimal rates in linear time. In *Proceedings of the 52nd Annual ACM SIGACT Symposium on Theory of Computing*, pages 439–449, 2020.

[FKT20b] Vitaly Feldman, Tomer Koren, and Kunal Talwar. Private stochastic convex optimization: optimal rates in linear time. In *Proceedings of*

the 52nd Annual ACM SIGACT Symposium on Theory of Computing, pages 439–449, 2020.

[FLLZ18] Cong Fang, Chris Junchi Li, Zhouchen Lin, and Tong Zhang. Spider: Near-optimal non-convex optimization via stochastic path-integrated differential estimator. 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.

[FSS18] Dylan J Foster, Ayush Sekhari, and Karthik Sridharan. Uniform convergence of gradients for non-convex learning and optimization. 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.

[FSS+19] Dylan J Foster, Ayush Sekhari, Ohad Shamir, Nathan Srebro, Karthik Sridharan, and Blake Woodworth. The complexity of making the gradient small in stochastic convex optimization. In *Conference on Learning Theory*, pages 1319–1345. PMLR, 2019.

[GGHVDM20] Chuan Guo, Tom Goldstein, Awni Hannun, and Laurens Van Der Maaten. Certified data removal from machine learning models. In *International Conference on Machine Learning*, pages 3832–3842. PMLR, 2020.

[GGVZ19] Antonio Ginart, Melody Guan, Gregory Valiant, and James Y Zou. Making ai forget you: Data deletion in machine learning. In *Advances in Neural Information Processing Systems*, pages 3518–3531, 2019.

[GJN+21] Varun Gupta, Christopher Jung, Seth Neel, Aaron Roth, Saeed Sharifi-

Malvajerdi, and Chris Waites. Adaptive machine unlearning. *Advances in Neural Information Processing Systems*, 34, 2021.

[GL13] Saeed Ghadimi and Guanghui Lan. Stochastic first-and zeroth-order methods for nonconvex stochastic programming. *SIAM Journal on Optimization*, 23(4):2341–2368, 2013.

[GL16] Saeed Ghadimi and Guanghui Lan. Accelerated gradient methods for nonconvex nonlinear and stochastic programming. *Mathematical Programming*, 156(1):59–99, 2016.

[GLM16] Rong Ge, Jason D Lee, and Tengyu Ma. Matrix completion has no spurious local minimum. In D. Lee, M. Sugiyama, U. Luxburg, I. Guyon, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 29. Curran Associates, Inc., 2016.

[GTS13] Abhradeep Guha Thakurta and Adam Smith. (nearly) optimal algorithms for private online learning in full-information and bandit settings. *Advances in Neural Information Processing Systems*, 26, 2013.

[HRS16] Moritz Hardt, Ben Recht, and Yoram Singer. Train faster, generalize better: Stability of stochastic gradient descent. In *International conference on machine learning*, pages 1225–1234. PMLR, 2016.

[Inc17] Apple Inc. Learning with privacy at scale. *Apple Machine Learning Journal*, 2017.

[IRU+19] Nikita Ivkin, Daniel Rothchild, Enayat Ullah, Ion Stoica, Raman Arora, et al. Communication-efficient distributed sgd with sketching. *Advances in Neural Information Processing Systems*, 32, 2019.

[ISCZ21] Zachary Izzo, Mary Anne Smart, Kamalika Chaudhuri, and James Zou. Approximate data deletion from machine learning models. In *International Conference on Artificial Intelligence and Statistics*, pages 2008–2016. PMLR, 2021.

[JEP+21] John Jumper, Richard Evans, Alexander Pritzel, Tim Green, Michael Figurnov, Olaf Ronneberger, Kathryn Tunyasuvunakool, Russ Bates, Augustin Zidek, Anna Potapenko, et al. Highly accurate protein structure prediction with alphafold. *Nature*, 596(7873):583–589, 2021.

[JKT12] Prateek Jain, Pravesh Kothari, and Abhradeep Thakurta. Differentially private online learning. In *25th Annual Conference on Learning Theory (COLT)*, pages 24.1–24.34, 2012.

[JNG+19] Chi Jin, Praneeth Netrapalli, Rong Ge, Sham M Kakade, and Michael I Jordan. A short note on concentration inequalities for random vectors with subgaussian norm. *arXiv preprint arXiv:1902.03736*, 2019.

[JT14] Prateek Jain and Abhradeep Guha Thakurta. (near) dimension independent risk bounds for differentially private learning. In Eric P. Xing and Tony Jebara, editors, *Proceedings of the 31st International Conference on Machine Learning*, volume 32 of *Proceedings of Machine Learning Research*, pages 476–484, Bejing, China, 2014. PMLR.

[KL15] Tomer Koren and Kfir Y Levy. Fast rates for exp-concave empirical risk minimization. In *NIPS*, pages 1477–1485, 2015.

[KLL21] Janardhan Kulkarni, Yin Tat Lee, and Daogao Liu. Private non-smooth erm and sco in subquadratic steps. In M. Ranzato, A. Beygelzimer, Y. Dauphin, P.S. Liang, and J. Wortman Vaughan, editors, *Advances*

in Neural Information Processing Systems, volume 34, pages 4053–4064. Curran Associates, Inc., 2021.

[KLN+08] Shiva Prasad Kasiviswanathan, Homin K. Lee, Kobbi Nissim, Sofya Raskhodnikova, and Adam Smith. What can we learn privately? In *2008 49th Annual IEEE Symposium on Foundations of Computer Science*, pages 531–540, 2008.

[KMS+21] Peter Kairouz, Brendan McMahan, Shuang Song, Om Thakkar, Abhradeep Thakurta, and Zheng Xu. Practical and private (deep) learning without sampling or shuffling. In *International Conference on Machine Learning*, pages 5213–5225. PMLR, 2021.

[KMY+16] Jakub Konecny, H. Brendan McMahan, Felix X. Yu, Peter Richtarik, Ananda Theertha Suresh, and Dave Bacon. Federated learning: Strategies for improving communication efficiency. In *NIPS Workshop on Private Multi-Party Machine Learning*, 2016.

[KOV15] Peter Kairouz, Sewoong Oh, and Pramod Viswanath. The composition theorem for differential privacy. In *International conference on machine learning*, pages 1376–1385. PMLR, 2015.

[KST08] Sham M Kakade, Karthik Sridharan, and Ambuj Tewari. On the complexity of linear prediction: Risk bounds, margin bounds, and regularization. *Advances in neural information processing systems*, 21, 2008.

[KST12] Daniel Kifer, Adam Smith, and Abhradeep Thakurta. Private convex empirical risk minimization and high-dimensional regression. In *Conference on Learning Theory*, pages 25–1, 2012.

[KU20] Gautam Kamath and Jonathan Ullman. A primer on private statistics. *arXiv preprint arXiv:2005.00010*, 2020.

[Lan12] Guanghui Lan. An optimal method for stochastic composite optimization. *Mathematical Programming*, 133(1-2):365–397, 2012.

[Lan20] Guanghui Lan. *First-order and stochastic optimization methods for machine learning*. Springer, 2020.

[LeC98] Yann LeCun. The mnist database of handwritten digits. *http://yann. lecun. com/exdb/mnist/*, 1998.

[LGOT23] Daogao Liu, Arun Ganesh, Sewoong Oh, and Abhradeep Guha Thakurta. Private (stochastic) non-convex optimization revisited: Second-order stationary points and excess risks. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023.

[LR+86] Torgny Lindvall, L Cris G Rogers, et al. Coupling of multidimensional diffusions by reflection. *The Annals of Probability*, 14(3):860–872, 1986.

[Mir17] Ilya Mironov. Rényi differential privacy. In *2017 IEEE 30th Computer Security Foundations Symposium (CSF)*, pages 263–275. IEEE, 2017.

[MRTZ18] H Brendan McMahan, Daniel Ramage, Kunal Talwar, and Li Zhang. Learning differentially private recurrent language models. In *International Conference on Learning Representations*, 2018.

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

[MUA+23] Michael Menart, Enayat Ullah, Raman Arora, Raef Bassily, and Cristóbal Guzmán. Differentially private non-convex optimiza-

tion under the kl condition with optimal rates. *arXiv preprint arXiv:2311.13447*, 2023.

[MWCC18] Cong Ma, Kaizheng Wang, Yuejie Chi, and Yuxin Chen. Implicit regularization in nonconvex statistical estimation: Gradient descent converges linearly for phase retrieval and matrix completion. In Jennifer Dy and Andreas Krause, editors, *Proceedings of the 35th International Conference on Machine Learning*, volume 80 of *Proceedings of Machine Learning Research*, pages 3345–3354. PMLR, 10–15 Jul 2018.

[N+18] Yurii Nesterov et al. *Lectures on convex optimization*, volume 137. Springer, 2018.

[NCH+23] Milad Nasr, Nicholas Carlini, Jonathan Hayase, Matthew Jagielski, A Feder Cooper, Daphne Ippolito, Christopher A Choquette-Choo, Eric Wallace, Florian Tramèr, and Katherine Lee. Scalable extraction of training data from (production) language models. *arXiv preprint arXiv:2311.17035*, 2023.

[Nel11] Jelani Nelson. *Sketching and streaming high-dimensional vectors*. PhD thesis, Massachusetts Institute of Technology, 2011.

[Nes09] Yurii Nesterov. Primal-dual subgradient methods for convex problems. *Mathematical programming*, 120(1):221–259, 2009.

[Nes12] Yurii Nesterov. How to make the gradients small. *Optima. Mathematical Optimization Society Newsletter*, pages 10–11, 2012.

[NêUZ20] Huy Lê Nguyễn, Jonathan Ullman, and Lydia Zakynthinou. Efficient Private Algorithms for Learning Large-Margin Halfspaces. In Aryeh Kontorovich and Gergely Neu, editors, *Proceedings of the 31st International Conference on Algorithmic Learning Theory*, volume 117 of

Proceedings of Machine Learning Research, pages 704–724. PMLR, 08 Feb–11 Feb 2020.

[NP06] Yurii Nesterov and Boris Polyak. Cubic regularization of newton method and its global performance. *Mathematical Programming*, 108:177–205, 2006.

[NRSM21a] Seth Neel, Aaron Roth, and Saeed Sharifi-Malvajerdi. Descent-to-delete: Gradient-based methods for machine unlearning. In *Algorithmic Learning Theory*, pages 931–962. PMLR, 2021.

[NRSM21b] Seth Neel, Aaron Roth, and Saeed Sharifi-Malvajerdi. Descent-to-delete: Gradient-based methods for machine unlearning. In *Algorithmic Learning Theory*, 2021.

[NY83] A.S. Nemirovsky and E.R. Yudin. *Problem Complexity and Method Efficiency in Optimization*. A Wiley-Interscience publication. Wiley, 1983.

[Ope23] OpenAI. Gpt-4 technical report. *ArXiv*, abs/2303.08774, 2023.

[Ora19] Francesco Orabona. A modern introduction to online learning. *arXiv preprint arXiv:1912.13213*, 2019.

[Pol63] Boris Polyak. Gradient methods for the minimisation of functionals. *Ussr Computational Mathematics and Mathematical Physics*, 3:864–878, 12 1963.

[R+61] Alfréd Rényi et al. On measures of entropy and information. In *Proceedings of the Fourth Berkeley Symposium on Mathematical Statistics and Probability, Volume 1: Contributions to the Theory of Statistics*. The Regents of the University of California, 1961.

[RKX+23] Alec Radford, Jong Wook Kim, Tao Xu, Greg Brockman, Christine McLeavey, and Ilya Sutskever. Robust speech recognition via large-scale weak supervision. In *International Conference on Machine Learning*, pages 28492–28518. PMLR, 2023.

[RPG+21] Aditya Ramesh, Mikhail Pavlov, Gabriel Goh, Scott Gray, Chelsea Voss, Alec Radford, Mark Chen, and Ilya Sutskever. Zero-shot text-to-image generation. In *International Conference on Machine Learning*, pages 8821–8831. PMLR, 2021.

[RPU+20] Daniel Rothchild, Ashwinee Panda, Enayat Ullah, Nikita Ivkin, Ion Stoica, Vladimir Braverman, Joseph Gonzalez, and Raman Arora. Fetchsgd: Communication-efficient federated learning with sketching. In *International Conference on Machine Learning*, pages 8253–8265. PMLR, 2020.

[RS16] Sofya Raskhodnikova and Adam Smith. Lipschitz extensions for node-private graph statistics and the generalized exponential mechanism. In *2016 IEEE 57th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 495–504, 2016.

[RV10] Mark Rudelson and Roman Vershynin. Non-asymptotic theory of random matrices: extreme singular values. In *Proceedings of the International Congress of Mathematicians 2010 (ICM 2010) (In 4 Volumes) Vol. I: Plenary Lectures and Ceremonies Vols. II–IV: Invited Lectures*, pages 1576–1602. World Scientific, 2010.

[SAKS21] Ayush Sekhari, Jayadev Acharya, Gautam Kamath, and Ananda Theertha Suresh. Remember what you want to for-

get: Algorithms for machine unlearning. *Advances in Neural Information Processing Systems*, 34, 2021.

[Sha15] Ohad Shamir. The sample complexity of learning linear predictors with the squared loss. *J. Mach. Learn. Res.*, 16:3475–3486, 2015.

[SHM+16] David Silver, Aja Huang, Chris J Maddison, Arthur Guez, Laurent Sifre, George Van Den Driessche, Julian Schrittwieser, Ioannis Antonoglou, Veda Panneershelvam, Marc Lanctot, et al. Mastering the game of go with deep neural networks and tree search. *nature*, 529(7587):484–489, 2016.

[SQW16] Ju Sun, Qing Qu, and John Wright. A geometric analysis of phase retrieval. In *2016 IEEE International Symposium on Information Theory (ISIT)*, pages 2379–2383, 2016.

[SSBD14] Shai Shalev-Shwartz and Shai Ben-David. *Understanding machine learning: From theory to algorithms*. Cambridge university press, 2014.

[SST10] Nathan Srebro, Karthik Sridharan, and Ambuj Tewari. Optimistic rates for learning with a smooth loss. *arXiv preprint arXiv:1009.3896*, 2010.

[SSTT20] Shuang Song, Thomas Steinke, Om Thakkar, and Abhradeep Thakurta. Characterizing private clipped gradient descent on convex generalized linear problems. *CoRR*, abs/2006.06783, 2020.

[SSTT21] Shuang Song, Thomas Steinke, Om Thakkar, and Abhradeep Thakurta. Evading the curse of dimensionality in unconstrained private glms. In *International Conference on Artificial Intelligence and Statistics*, pages 2638–2646. PMLR, 2021.

[SU15] Thomas Steinke and Jonathan Ullman. Between pure and approximate differential privacy. *Journal of Privacy and Confidentiality*, 7, 01 2015.

[SU17] Thomas Steinke and Jonathan Ullman. Tight lower bounds for differentially private selection. In *2017 IEEE 58th Annual Symposium on Foundations of Computer Science (FOCS)*, pages 552–563. IEEE, 2017.

[TDPPN19] Quoc Tran-Dinh, Nhan H Pham, Dzung T Phan, and Lam M Nguyen. Hybrid stochastic gradient descent algorithms for stochastic nonconvex optimization. *arXiv preprint arXiv:1905.05920*, 2019.

[TTZ14] Kunal Talwar, Abhradeep Thakurta, and Li Zhang. Private empirical risk minimization beyond the worst case: The effect of the constraint set geometry. *arXiv preprint arXiv:1411.5417*, 2014.

[TTZ15] Kunal Talwar, Abhradeep Thakurta, and Li Zhang. Nearly optimal private lasso. In *NIPS*, 2015.

[UA22] Enayat Ullah and Raman Arora. Generalization bounds for kernel canonical correlation analysis. *Transactions on Machine Learning Research*, 2022.

[UA23] Enayat Ullah and Raman Arora. From adaptive query release to machine unlearning. In *International Conference on Machine Learning*, pages 34642–34667. PMLR, 2023.

[UCKO23] Enayat Ullah, Christopher A. Choquette-Choo, Peter Kairouz, and Sewoong Oh. Private federated learning with autotuned compression. In Andreas Krause, Emma Brunskill, Kyunghyun Cho, Barbara Engelhardt, Sivan Sabato, and Jonathan Scarlett, editors, *International*

Conference on Machine Learning, ICML 2023, 23-29 July 2023, Honolulu, Hawaii, USA, volume 202 of *Proceedings of Machine Learning Research*, pages 34668–34708. PMLR, 2023.

[ULAB22] Enayat Ullah, Harry Lang, Raman Arora, and Vladimir Braverman. Clustering using approximate nearest neighbour oracles. *Transactions on Machine Learning Research*, 2022.

[UMMA18] Enayat Ullah, Poorya Mianjy, Teodor V Marinov, and Raman Arora. Streaming kernel pca with $\widetilde{o}(\sqrt{n})$ random features. In *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, pages 7322–7332, 2018.

[UMR+21] Enayat Ullah, Tung Mai, Anup Rao, Ryan A Rossi, and Raman Arora. Machine unlearning via algorithmic stability. In *Conference on Learning Theory*, 2021.

[VEH14] Tim Van Erven and Peter Harremos. Rényi divergence and kullback-leibler divergence. *IEEE Transactions on Information Theory*, 60(7):3797–3820, 2014.

[Ver18] Roman Vershynin. *High-dimensional probability: An introduction with applications in data science*, volume 47. Cambridge university press, 2018.

[VGNA20] Mariia Vladimirova, Stéphane Girard, Hien Nguyen, and Julyan Arbel. Sub-weibull distributions: Generalizing sub-gaussian and sub-exponential properties to heavier tailed distributions. *Stat*, 9(1):e318, 2020.

[Vil09] Cédric Villani. *Optimal transport: old and new*, volume 338. Springer, 2009.

[Völ16] Florian Völlering. On maximal agreement couplings. *arXiv preprint arXiv:1608.01511*, 2016.

[Wal77] Alastair J Walker. An efficient method for generating discrete random variables with general distributions. *ACM Transactions on Mathematical Software (TOMS)*, 3(3):253–256, 1977.

[Wan18] Yu-Xiang Wang. Revisiting differentially private linear regression: optimal and adaptive prediction & estimation in unbounded domain. In Amir Globerson and Ricardo Silva, editors, *Proceedings of the Thirty-Fourth Conference on Uncertainty in Artificial Intelligence, UAI 2018, Monterey, California, USA, August 6-10, 2018*, pages 93–103. AUAI Press, 2018.

[WBK19] Yu-Xiang Wang, Borja Balle, and Shiva Prasad Kasiviswanathan. Subsampled rényi differential privacy and analytical moments accountant. In *International Conference on Artificial Intelligence and Statistics*, 2019.

[WCX19a] Di Wang, Changyou Chen, and Jinhui Xu. Differentially private empirical risk minimization with non-convex loss functions. In *International Conference on Machine Learning*, pages 6526–6535. PMLR, 2019.

[WCX19b] Di Wang, Changyou Chen, and Jinhui Xu. Differentially private empirical risk minimization with non-convex loss functions. In *Proceedings of the 36th International Conference on Machine Learning*, volume 97 of *Proceedings of Machine Learning Research*, pages 6526–6535. PMLR, 09–15 Jun 2019.

[Wik21] Wikipedia. Right to be forgotten — Wikipedia, the free encyclopedia. http://en.wikipedia.org/w/index.php?title=Right%20to%

20be%20forgotten&oldid=1007605238, 2021. [Online; accessed 23-February-2021].

[WJZ+19] Zhe Wang, Kaiyi Ji, Yi Zhou, Yingbin Liang, and Vahid Tarokh. Spiderboost and momentum: Faster variance reduction algorithms. In H. Wallach, H. Larochelle, A. Beygelzimer, F. d'Alché-Buc, E. Fox, and R. Garnett, editors, *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019.

[WS16] Blake E Woodworth and Nati Srebro. Tight complexity bounds for optimizing composite objectives. *Advances in neural information processing systems*, 29:3639–3647, 2016.

[WUMA22] Yunjuan Wang, Enayat Ullah, Poorya Mianjy, and Raman Arora. Adversarial robustness is at odds with lazy training. *Advances in Neural Information Processing Systems*, 35:6505–6516, 2022.

[WUNTA23] Austin Watkins, Enayat Ullah, Thanh Nguyen-Tang, and Raman Arora. Optimistic rates for multi-task representation learning. In *Thirty-seventh Conference on Neural Information Processing Systems*, 2023.

[WX19] Di Wang and Jinhui Xu. Differentially private empirical risk minimization with smooth non-convex loss functions: A non-stationary view. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 33, pages 1182–1189, 2019.

[WYX17] Di Wang, Minwei Ye, and Jinhui Xu. Differentially private empirical risk minimization revisited: Faster and more general. In *Advances in Neural Information Processing Systems*, pages 2722–2731, 2017.

[XZA+23] Zheng Xu, Yanxiang Zhang, Galen Andrew, Christopher A Choquette-Choo, Peter Kairouz, H Brendan McMahan, Jesse Rosenstock, and Yuanbo Zhang. Federated learning of gboard language models with differential privacy. *arXiv preprint arXiv:2305.18465*, 2023.

[ZCH+20] Yingxue Zhou, Xiangyi Chen, Mingyi Hong, Zhiwei Steven Wu, and Arindam Banerjee. Private stochastic non-convex optimization: Adaptive algorithms and tighter generalization bounds. *CoRR*, abs/2006.13501, 2020.

[ZMLX21] Qiuchen Zhang, Jing Ma, Jian Lou, and Li Xiong. Private stochastic non-convex optimization with improved utility rates. In *Proceedings of the Thirtieth International Joint Conference on Artificial Intelligence, IJCAI-21*, pages 3370–3376, 2021.

[ZSM+20] Mingrui Zhang, Zebang Shen, Aryan Mokhtari, Hamed Hassani, and Amin Karbasi. One sample stochastic frank-wolfe. In *International Conference on Artificial Intelligence and Statistics*, pages 4012–4023. PMLR, 2020.

[ZZMW17] Jiaqi Zhang, Kai Zheng, Wenlong Mou, and Liwei Wang. Efficient private erm for smooth objectives. In *Proceedings of the 26th International Joint Conference on Artificial Intelligence*, IJCAI'17, page 3922–3928. AAAI Press, 2017.

Appendix A

Appendix for Chapter 2

A.1 Missing Proofs from Section 2.3.1 (Smooth GLMs)

A.1.1 Utility Lemmas

Fact 1. *[SSBD14] For a $\widetilde{H}$-smooth non-negative function f, and for any $u \in \mathrm{dom}(f)$, we have $\|\nabla f(u)\| \leq \sqrt{4\widetilde{H} f(u)}$.*

A.1.2 Proof of Lemma 2

From the self-bounding property (Fact 1), the $\|\mathcal{Y}\|^2$ bound on loss at zero, and smoothness, we have the following bound on the gradient:

$$\begin{aligned}\|\nabla \ell(w;(x,y)\| &\leq \|\nabla \ell(0;(x,y))\| + \|\nabla \ell(w;(x,y)) - \nabla \ell(0;(x,y))\| \\ &\leq 2\sqrt{H}\,\|x\|\,\ell(0;(x,y)) + H\,\|x\|^2\,\|w\| \\ &\leq 2\left(\sqrt{H}\,\|\mathcal{Y}\|\,\|x\| + H\,\|x\|^2\,\|w\|\right). \end{aligned} \tag{A.1}$$

Lemma 4. *For any $w \in \mathcal{B}(D)$ and any $(x,y) \in (\mathcal{X} \times \mathcal{Y})$ it holds that $\ell(w;(x,y)) \leq 3(\|\mathcal{Y}\|^2 + HD^2\|\mathcal{X}\|^2)$.*

Proof. Using the fact that the loss function is $G = \|\mathcal{Y}\|\sqrt{H}\|\mathcal{X}\| + DH\|\mathcal{X}\|^2$-Lipschitz

in the constraint set (Lemma 2) we have

$$\begin{aligned}\ell(w;(x,y)) &\leq \ell(0;(x,y)) + |\ell(w;(x,y)) - \ell(0;(x,y))| \\ &\leq \|\mathcal{Y}\|^2 + G\|w\| \\ &\leq \|\mathcal{Y}\|^2 + \|\mathcal{Y}\|\sqrt{H}D\|\mathcal{X}\| + HD^2\|\mathcal{X}\|^2 \\ &\leq 3\left(\|\mathcal{Y}\|^2 + HD^2\|\mathcal{X}\|^2\right)\end{aligned}$$

where the last step follow from AM-GM inequality. □

A.1.3 Low Dimension

Before presenting the proof of Theorem 8, we provide formal statements of its Corollaries.

Corollary 2, stated below, gives an upper bound on excess risk of gradient descent in the non-private setting.

Corollary 2. *Let ℓ be a non-negative convex $\widetilde{H}$ smooth loss function, bounded at zero by $\|\mathcal{Y}\|^2$. Let $n_0 = \frac{\widetilde{H}D^2}{\|\mathcal{Y}\|^2}$, $\mathcal{W} = \mathcal{B}(D)$, $T = n$, and $\eta = \min\left(\frac{D}{\sqrt{T}\sqrt{\widetilde{H}}\|\mathcal{Y}\|}, \frac{1}{4\widetilde{H}}\right)$. Given a dataset S of $n \geq n_0$ i.i.d samples from an unknown distribution $\mathcal{D}$, the excess risk of output of Algorithm 1 with $\sigma = 0$ is bounded as,*

$$\mathbb{E}[L(\widehat{w};\mathcal{D}) - L(w^*;\mathcal{D})] \leq O\left(\frac{\sqrt{\widetilde{H}}D\|\mathcal{Y}\|}{\sqrt{n}}\right).$$

Corollary 3 below gives an upper bound on excess risk of noisy gradient descent for non-negative smooth GLMs.

Corollary 3. *Let $n_0 = \frac{H\|\mathcal{X}\|^2D^2}{\|\mathcal{Y}\|^2}$, $\eta = \min\left(\frac{D}{\sqrt{T}\max\left(\sqrt{H}\|\mathcal{X}\|\|\mathcal{Y}\|,\sigma\sqrt{d}\right)}, \frac{1}{4H\|\mathcal{X}\|}\right)$, $\mathcal{W} = \mathcal{B}(D)$, $\sigma^2 = \frac{8G^2T\log(1/\delta)}{n^2\epsilon^2}$ and $T = n$. Let ℓ be a non-negative convex H smooth GLM, bounded at zero by $\|\mathcal{Y}\|^2$. Algorithm 1 satisfies (ϵ,δ)-differential privacy. Given a dataset*

$S \sim \mathcal{D}^n$, $n \geq n_0$, then the excess risk of output of Algorithm 1 is bounded as,

$$\mathbb{E}[L(\widehat{w}; \mathcal{D}) - L(w^*; \mathcal{D})] = O\Bigg(\frac{\sqrt{H}\|\mathcal{X}\| D \|\mathcal{Y}\|}{\sqrt{n}} + \frac{\left(\sqrt{H}\|\mathcal{X}\|\|\mathcal{Y}\| + H\|\mathcal{X}\|^2 D^2\right) D\sqrt{d \log(1/\delta)}}{n\epsilon} \Bigg).$$

A.1.3.1 Proof of Theorem 8

Since Algorithm 1 uses a projection step, the iterates always lie in the constraint set $\{w : \|w\| \leq D\}$. Hence, the function over this constraint set is G-Lipschitz. From the analysis of Noisy (S)GD in [BST14, BFTT19], we have that the setting of noise variance σ^2 ensures that the algorithm satisfies (ϵ, δ)-DP

We now move to the utility part. We start with the decomposition of excess risk as

$$\mathbb{E}[L(\widehat{w}; \mathcal{D}) - L(w^*; \mathcal{D})] = \mathbb{E}[L(\widehat{w}; \mathcal{D}) - \widehat{L}(\widehat{w}; S)] + \mathbb{E}[\widehat{L}(\widehat{w}; S) - \widehat{L}(w^*; S)] \tag{A.2}$$

The key arguments are as follows: we first bound generalization gap, or on-average stability (first term in the right hand side above), in terms of average argument stability and excess empirical risk (Lemma 5). We then bound average argument stability in terms of average regret (Lemma 6). Finally, in Lemma 7, we provide bounds on excess empirical risk and average regret of gradient descent. Substituting these in the above equation gives the claimed bound. We now fill in the details.

We start with Lemma 5 which gives the following bound on the generalization gap:

$$\mathbb{E}[L(\mathcal{A}(S); \mathcal{D}) - \widehat{L}(\widehat{w}; S)] \leq \frac{\sqrt{\widetilde{H}} D}{\sqrt{n}\,\|\mathcal{Y}\|} \left(\mathbb{E}[\widehat{L}(\widehat{w}; S) - \widehat{L}(w^*; S)]\right) \tag{A.3}$$

$$+ \frac{2\sqrt{\widetilde{H} n}\|\mathcal{Y}\|}{D} \mathrm{AAS}(\mathcal{A})^2 + \frac{\sqrt{\widetilde{H}} D \|\mathcal{Y}\|}{\sqrt{n}} \tag{A.4}$$

Substituting the bound on $\mathrm{AAS}(\mathcal{A})$ from Lemma 6, the second term becomes,

$$\begin{aligned}
&\frac{2\sqrt{\widetilde{H}n}\|\mathcal{Y}\|}{D}\,\mathrm{AAS}(\mathcal{A})^2 \\
&\leq \frac{16\widetilde{H}^{3/2}\eta^2 T\|\mathcal{Y}\|}{\sqrt{n}D}\frac{1}{n}\sum_{j=1}^{T}\mathbb{E}[\widehat{L}(w_j;S)-\widehat{L}(w^*;S)]+\frac{16\widetilde{H}^{3/2}\eta^2 T\|\mathcal{Y}\|}{\sqrt{n}D} \\
&\leq \frac{16\sqrt{\widetilde{H}}D}{\sqrt{n}\|\mathcal{Y}\|}\frac{1}{n}\sum_{j=1}^{T}\mathbb{E}[\widehat{L}(w_j;S)-\widehat{L}(w^*;S)]+\frac{16\sqrt{\widetilde{H}}D}{\sqrt{n}\|\mathcal{Y}\|} \\
&\leq \frac{16}{n}\sum_{j=1}^{T}\mathbb{E}[\widehat{L}(w_j;S)-\widehat{L}(w^*;S)]+\frac{16\sqrt{\widetilde{H}}D}{\sqrt{n}\|\mathcal{Y}\|}.
\end{aligned}$$

Substituting the above in Eqn. (A.3) and using the fact that $\frac{\sqrt{\widetilde{H}}D}{\sqrt{n}\|\mathcal{Y}\|}\leq 1$ from the lower bound on n, we get:

$$\begin{aligned}
&\mathbb{E}[L(\mathcal{A}(S);\mathcal{D})-\widehat{L}(\widehat{w};S)] \\
&\leq \left(\mathbb{E}[\widehat{L}(\widehat{w};S)-\widehat{L}(w^*;S)]\right)+\frac{16}{n}\left(\sum_{j=1}^{T}\mathbb{E}[\widehat{L}(w_j;S)-\widehat{L}(w^*;S)]\right) \\
&+\frac{17\sqrt{\widetilde{H}}D\|\mathcal{Y}\|}{\sqrt{n}}.
\end{aligned}$$

From the excess empirical risk guarantee (Lemma 7), the terms excess empirical risk $\mathbb{E}[\widehat{L}(\widehat{w};S)-\widehat{L}(w^*;S)]$ and average regret $\frac{1}{n}\left(\sum_{j=1}^{T}\mathbb{E}[\widehat{L}(w_j;S)-\widehat{L}(w^*;S)]\right)$ are both bounded by the same quantity. Thus, substituting the above in Eqn. (A.2) and substituting the bound from 7, we have,

$$\begin{aligned}
\mathbb{E}[L(\widehat{w};\mathcal{D})-L(w^*;\mathcal{D})] &\leq \frac{18}{n}\sum_{j=1}^{n}\mathbb{E}[\widehat{L}(w_j;S)-\widehat{L}(w^*;S)]+\frac{17\sqrt{\widetilde{H}}D\|\mathcal{Y}\|}{\sqrt{n}} \\
&\leq O\left(\frac{\sqrt{H}D\|\mathcal{Y}\|}{\sqrt{n}}+\frac{\sqrt{d}GD\log(1/\delta)}{n\epsilon}\right).
\end{aligned}$$

Substituting the value of G completes the proof.

Lemma 5. *Let $n\geq\frac{\widetilde{H}D^2}{\|\mathcal{Y}\|^2}$. Let ℓ be a non-negative $\widetilde{H}$ smooth convex loss function. Let S be a dataset of n i.i.d. samples from an unknown distribution $\mathcal{D}$, and w^* denote the optimal population risk minimizer. The generalization gap of algorithm $\mathcal{A}$ is bounded as,*

$$\mathbb{E}[L(\mathcal{A}(S);\mathcal{D}) - \widehat{L}(\mathcal{A}(S);S)] \leq \frac{\sqrt{\widetilde{H}}D}{\sqrt{n}\,\|\mathcal{Y}\|}\left(\mathbb{E}[\widehat{L}(\mathcal{A}(S);S) - \widehat{L}(w^*;S)]\right) + \frac{2\sqrt{\widetilde{H}n}\|\mathcal{Y}\|}{D}\,\mathrm{AAS}(\mathcal{A})^2 + \frac{\sqrt{\widetilde{H}}D\|\mathcal{Y}\|}{\sqrt{n}}.$$

Proof. Let $\widehat{w} := \mathcal{A}(S)$, $S^{(i)}$ be the dataset where the i-th data point is replaced by an i.i.d. point (x', y') and let $\widehat{w}^{(i)}$ be the corresponding output of $\mathcal{A}$.

A standard fact (see [SSBD14]) is that generalization gap is equal to on-average stability:

$$\mathbb{E}[L(\widehat{w};\mathcal{D}) - \widehat{L}(\widehat{w};S)] = \mathbb{E}[\ell(\widehat{w}^{(i)};(x_i,y_i)) - \ell(\widehat{w};(x_i,y_i)].$$

From smoothness and self-bounding property, we have,

$$\begin{aligned}\ell(\widehat{w}^{(i)};(x_i,y_i)) - \ell(\widehat{w};(x_i,y_i) &\leq \|\nabla\ell(\widehat{w};(x_i,y_i))\|\left\|\widehat{w} - \widehat{w}^{(i)}\right\| + \frac{\widetilde{H}}{2}\left\|\widehat{w} - \widehat{w}^{(i)}\right\|^2 \\ &\leq 2\sqrt{\widetilde{H}\ell((\widehat{w};(x_i,y_i))}\left\|\widehat{w} - \widehat{w}^{(i)}\right\| + \frac{\widetilde{H}}{2}\left\|\widehat{w} - \widehat{w}^{(i)}\right\|^2.\end{aligned}$$

Taking expectation, using Cauchy-Schwarz inequality and substituting average argu-

ment stability, we get,

$$
\begin{aligned}
&\mathbb{E}\left[\ell(\widehat{w}^{(i)};(x_i,y_i)) - \ell(\widehat{w};(x_i,y_i)\right] \\
&\leq 2\sqrt{\widetilde{H}\mathbb{E}[\ell((\widehat{w};(x_i,y_i))]}\sqrt{\mathbb{E}[\|\widehat{w}-\widehat{w}^{(i)}\|^2]} + \frac{\widetilde{H}}{2}\mathbb{E}\left[\left\|\widehat{w}-\widehat{w}^{(i)}\right\|^2\right] \qquad \text{(A.5)} \\
&\leq 2\sqrt{\widetilde{H}\mathbb{E}[\widehat{L}(\widehat{w};S)]}\,\mathrm{AAS}(\mathcal{A}) + \frac{\widetilde{H}\,\mathrm{AAS}(\mathcal{A})^2}{2} \\
&\leq \frac{\sqrt{\widetilde{H}}D\mathbb{E}[\widehat{L}(\widehat{w};S)]}{\sqrt{n}\,\|\mathcal{Y}\|} + \frac{\sqrt{\widetilde{H}n}\,\|\mathcal{Y}\|\,\mathrm{AAS}(\mathcal{A})^2}{D} + \frac{\widetilde{H}\,\mathrm{AAS}(\mathcal{A})^2}{2} \\
&\leq \frac{\sqrt{\widetilde{H}}D}{\sqrt{n}\,\|\mathcal{Y}\|}\left(\mathbb{E}[\widehat{L}(\widehat{w};S) - \widehat{L}(w^*;S)]\right) + \frac{\sqrt{\widetilde{H}n}\,\|\mathcal{Y}\|\,\mathrm{AAS}(\mathcal{A})^2}{D} + \frac{\widetilde{H}\,\mathrm{AAS}(\mathcal{A})^2}{2} \\
&+ \frac{\sqrt{\widetilde{H}}D}{\sqrt{n}\,\|\mathcal{Y}\|}\mathbb{E}[\widehat{L}(w^*;S)] \\
&\leq \frac{\sqrt{\widetilde{H}}D}{\sqrt{n}\,\|\mathcal{Y}\|}\left(\mathbb{E}[\widehat{L}(\widehat{w};S) - \widehat{L}(w^*;S)]\right) + \left(\frac{\sqrt{\widetilde{H}n}\|\mathcal{Y}\|}{D} + \frac{\widetilde{H}}{2}\right)\mathrm{AAS}(\mathcal{A})^2 \\
&+ \frac{\sqrt{\widetilde{H}}D\|\mathcal{Y}\|}{\sqrt{n}} \qquad \text{(A.6)}
\end{aligned}
$$

where the third inequality follows from AM-GM inequality, and last follows since w^* is the optimal solution: $\mathbb{E}[\widehat{L}(w^*;S)] = L(w^*;\mathcal{D}) \leq L(0;\mathcal{D}) \leq \|\mathcal{Y}\|^2$. Finally, using the lower bound on n, we have $\frac{\widetilde{H}}{2} \leq \frac{\sqrt{\widetilde{H}n}\|\mathcal{Y}\|}{D}$, substituting which gives the claimed bound. □

Lemma 6. *The average argument stability for noisy GD (Algorithm 1) run for T iterations with step size $\eta \leq \frac{4}{\widetilde{H}}$ is bounded as*

$$
\mathrm{AAS}(\mathcal{A})^2 \leq \frac{8\widetilde{H}\eta^2 T}{n}\frac{1}{n}\sum_{j=1}^{T}\mathbb{E}[\widehat{L}(w_j;S) - \widehat{L}(w^*;S)] + \frac{8\widetilde{H}\eta^2 T\|\mathcal{Y}\|^2}{n}.
$$

Proof. The uniform argument stability analysis for (Noisy) (S)GD is limited to the Lipschitz setting [HRS16, BFTT19, BFGT20] and therefore not directly applicable. We therefore need to modify the arguments to give an average stability analysis in smooth (non-Lipschitz) case.

Let $\widehat{w} := \mathcal{A}(S)$, $S^{(i)}$ be the dataset where the i-th data point is replaced by an

i.i.d. point (x', y') and let $\widehat{w}^{(i)}$ be the corresponding output of $\mathcal{A}$. Moroever, let w_i denote the iterate of noisy SGD on dataset S and similarly $\widetilde{w}_i^{(i)}$ for dataset $S^{(i)}$.

We simply couple the Gaussian noise sampled at each iteration to be equal on both datasets. Using the fact the the updates are non-expansive, we have

$$\begin{aligned}\left\|w_{t+1} - w'_{t+1}\right\| &\leq \|w_t - w'_t\| + \frac{\eta\left(\|\nabla\ell(w_t;(x_i,y_i))\| + \|\nabla\ell(w'_t;(x';y'))\|\right)}{n} \\ &\leq \frac{\eta\sum_{j=1}^{t}\left(\|\nabla\ell(w_j;(x_i,y_i))\| + \left\|\nabla\ell(w'_j;(x';y'))\right\|\right)}{n}.\end{aligned}$$

From the self-bounding property in Fact 1, we get that $(\|\nabla\ell(w_j;(x_i,y_i))\|) \leq 2\sqrt{\widetilde{H}\ell(w_j;(x_i,y_i))})$. Therefore, we get,

$$\begin{aligned}\mathbb{E}[\|w_t - w'_t\|^2] &\leq \frac{4\widetilde{H}\eta^2 T}{n^2}\sum_{j=1}^{t}\left(\mathbb{E}[\ell(w_j;(x_i,y_i)) + \ell(w'_j;(x',y'))]\right) \\ &= \frac{8\widetilde{H}\eta^2 T}{n^2}\sum_{j=1}^{t}\mathbb{E}[\widehat{L}(w_j;S)].\end{aligned}$$

For the average iterate,

$$\begin{aligned}\mathbb{E}\left[\left\|\widehat{w} - \widehat{w}^{(i)}\right\|^2\right] &\leq \frac{1}{T^2}T\sum_{t=1}^{T}\mathbb{E}[\|w_t - w'_t\|^2] \leq \frac{8\widetilde{H}\eta^2 T}{n^2}\sum_{j=1}^{T}\mathbb{E}[\widehat{L}(w_j;S)] \\ &\leq \frac{8\widetilde{H}\eta^2 T}{n}\frac{1}{n}\sum_{j=1}^{T}\mathbb{E}[\widehat{L}(w_j;S) - \widehat{L}(w^*;S)] + \frac{8\widetilde{H}\eta^2 T}{n}\mathbb{E}[\widehat{L}(w^*;S)] \\ &\leq \frac{8\widetilde{H}\eta^2 T}{n}\frac{1}{n}\sum_{j=1}^{T}\mathbb{E}[\widehat{L}(w_j;S) - \widehat{L}(w^*;S)] + \frac{8\widetilde{H}\eta^2 T\|\mathcal{Y}\|^2}{n}\end{aligned}$$

where the last inequality follows since w^* is the population risk minimizer: $\mathbb{E}[\widehat{L}(w^*;S)] = L(w^*;\mathcal{D})] \leq L(0;\mathcal{D}) \leq \|\mathcal{Y}\|^2$. □

Lemma 7. *Let* $n \geq \frac{\widetilde{H}D^2}{\|\mathcal{Y}\|^2}$, $\eta = \min\left(\frac{D}{\sqrt{T}\max\left(\sqrt{\widetilde{H}}\|\mathcal{Y}\|,\sigma\sqrt{d}\right)}, \frac{1}{4\widetilde{H}}\right)$, $\sigma^2 = \frac{8G^2T\log(1/\delta)}{n^2\epsilon^2}$ *and* $T = n$. *We have,*

$$\begin{aligned}\mathbb{E}\left[\widehat{L}(\widehat{w};S) - \widehat{L}(w^*;S)\right] &\leq \frac{1}{T}\sum_{j=1}^{T}\mathbb{E}\left[\widehat{L}(w_j;S) - \widehat{L}(w^*;S)\right] \\ &= O\left(\frac{\sqrt{H}D\|\mathcal{Y}\|}{\sqrt{n}} + \frac{\sqrt{d}G\log(1/\delta)}{n\epsilon}\right).\end{aligned}$$

where w^* *is the (minimum norm) population risk minimizer.*

Proof. From standard analysis of (S)GD,

$$\begin{aligned}
\mathbb{E}\left[\|w_{t+1} - w^*\|^2\right] &\leq \mathbb{E}\left[\left\|w_t - \eta\left(\nabla \widehat{L}(w_t; S) + \xi_t\right) - w^*\right\|^2\right] \\
&\leq \mathbb{E}\left[\|w_t - w^*\|^2\right] + \eta^2 \mathbb{E}\left[\left\|\nabla \widehat{L}(w_t; S)\right\|^2\right] \\
&\quad + \eta^2\sigma^2 d - 2\eta\mathbb{E}\left[\widehat{L}(w_t; S) - \widehat{L}(w^*; S)\right] \\
&\leq \mathbb{E}\left[\|w_t - w^*\|^2\right] + 4\eta^2 \widetilde{H}\mathbb{E}[\widehat{L}(w_t; S)] \\
&\quad + \eta^2\sigma^2 d - 2\eta\mathbb{E}\left[\widehat{L}(w_t; S) - \widehat{L}(w^*; S)\right]
\end{aligned}$$

where the last inequality follows from self-bounding property (Fact 1). Rearranging, and using the fact that $\mathbb{E}[\widehat{L}(w^*; S) = L(w^*; \mathcal{D})] \leq L(0; \mathcal{D}) \leq \|\mathcal{Y}\|^2$ we get,

$$\begin{aligned}
\left(1 - 2\eta\widetilde{H}\right)\mathbb{E}\left[\widehat{L}(w_t; S) - \widehat{L}(w^*; S)\right] \leq \frac{\mathbb{E}\left[\|w_t - w^*\|^2 - \|w_{t+1} - w^*\|^2\right]}{2\eta} \\
+ 2\eta\left(\widetilde{H}\|\mathcal{Y}\|^2 + \sigma^2 d\right)
\end{aligned}$$

From the choice of η, we have $\left(1 - 2\eta\widetilde{H}\right) \geq \frac{1}{2}$. Averaging over T iterations, we get that the average regret is,

$$\frac{1}{T}\sum_{j=1}^{T}\mathbb{E}\left[\widehat{L}(w_j; S) - \widehat{L}(w^*; S)\right] \leq \frac{D^2}{\eta T} + 4\eta\left(\widetilde{H}\|\mathcal{Y}\|^2 + \sigma^2 d\right).$$

Setting $\eta = \min\left(\frac{D}{\sqrt{T}\max\left(\sqrt{\widetilde{H}}\|\mathcal{Y}\|, \sigma\sqrt{d}\right)}, \frac{1}{4\widetilde{H}}\right)$, we get

$$\frac{1}{T}\sum_{j=1}^{T}\mathbb{E}\left[\widehat{L}(w_j; S) - \widehat{L}(w^*; S)\right] = O\left(\frac{\sqrt{\widetilde{H}}D\|\mathcal{Y}\|}{\sqrt{T}} + \frac{D\sqrt{d}\sigma}{\sqrt{T}} + \frac{\widetilde{H}D^2}{T}\right).$$

Finally, substituting $\sigma^2 = \frac{8G^2T\log(1/\delta)}{n^2\epsilon^2}$, $T = n$ and using the lower bound on n gives the claimed bound on average regret. Applying convexity to lower bound average regret by excess empirical risk of $\widehat{w}$ gives the same bound on excess empirical risk. □

A.1.4 High Dimension

Proof of Theorem 9. Let $\alpha \leq 1$ be a parameter to be set later. From the JL property with $k = O\left(\frac{\log(2n/\delta)}{\alpha^2}\right)$, with probability at least $1 - \delta/2$, for all data points x_i,

$\|\Phi x_i\| \leq (1+\alpha)\|x_i\| \leq 2\|\mathcal{X}\|$, and $\|\Phi w^*\|^2 \leq 2\|w^*\|^2 \leq 2D^2$.

Further, by Lemma 2, for any w in the embedding space with $\|w\|^2 \leq D$, with probability at least $1-\delta/2$, the loss is G Lipschitz where $G = 2\|\mathcal{Y}\|\sqrt{H}\|\mathcal{X}\| + 2HD\|\mathcal{X}\|^2$. The privacy guarantee now follows from the privacy of Noisy SGD and post-processing.

For the utility guarantee, let $L(w;\Phi\mathcal{D})$ and $\widehat{L}(w;\Phi S)$ denote population and empirical risk (resp) where test and training feature vectors (resp) are mapped using Φ. The excess risk can be decomposed as:

$$\begin{aligned}\mathbb{E}\left[L(\Phi^\top\widetilde{w};\mathcal{D}) - L(w^*;\mathcal{D})\right] &= \mathbb{E}\left[L(\widetilde{w};\Phi\mathcal{D}) - L(\Phi w^*;\Phi\mathcal{D})\right] \\ &\quad + \mathbb{E}\left[L(\Phi w^*;\Phi\mathcal{D}) - L(w^*;\mathcal{D})\right]. \qquad \text{(A.7)}\end{aligned}$$

The first term in Eqn. (A.7) is bounded by the utility guarantee of the DP-SCO method. In particular, from Lemma 8 (below), we have

$$\begin{aligned}\mathbb{E}[L(\widetilde{w};\Phi\mathcal{D}) - L(\Phi w^*;\Phi\mathcal{D})] = O\Bigg(&\frac{\sqrt{H}\|\mathcal{X}\|D\|\mathcal{Y}\|}{\sqrt{n}} \\ &+ \frac{\left(\sqrt{H}\|\mathcal{X}\|\|\mathcal{Y}\| + H\|\mathcal{X}\|^2D^2\right)D\sqrt{k\log(1/\delta)}}{n\epsilon}\Bigg).\end{aligned}$$

The second term in Eqn. (A.7) is bounded by the JL property together with smoothness and the fact that w^* is the optimal solution thus $\nabla L(w^*;\mathcal{D}) = 0$. This gives us

$$\begin{aligned}\mathbb{E}[L(\Phi w^*;\Phi\mathcal{D})] - L(w^*;\mathcal{D})] &\leq \frac{H}{2}\mathbb{E}\left[|\langle\Phi w^*,\Phi x\rangle - \langle w^*,x\rangle|^2\right] \\ &\leq \frac{\alpha^2 H\|w^*\|^2\|\mathcal{X}\|^2}{2} \\ &\leq \widetilde{O}\left(\frac{HD^2\|\mathcal{X}\|^2}{k}\right).\end{aligned}$$

Combining, we get,

$$\begin{aligned}&\mathbb{E}\left[L(\Phi^\top\widetilde{w};\mathcal{D}) - L(w^*;\mathcal{D})\right] \\ &= \widetilde{O}\left(\frac{\sqrt{H}\|\mathcal{X}\|D\|\mathcal{Y}\|}{\sqrt{n}} + \frac{\left(\sqrt{H}\|\mathcal{X}\|\|\mathcal{Y}\| + H\|\mathcal{X}\|^2D\right)D\sqrt{k\log(1/\delta)}}{n\epsilon} + \frac{HD^2\|\mathcal{X}\|^2}{k}\right).\end{aligned}$$

Setting $k = O\left(\frac{DH\|\mathcal{X}\|\log(2n/\delta)n\epsilon}{G}\right)^{2/3}$ completes the proof.

□

Lemma 8. *Let $\Phi \in \mathbb{R}^{d\times k}$ be a data-oblivious JL matrix. Let $S = \{(x_i, y_i)\}_{i=1}^n$ of n i.i.d data points and let $\Phi S := \{(\Phi x_i, y_i)\}_{i=1}^n$. Let $\widetilde{w} \in \mathbb{R}^k$ be the average iterate returned by Algorithm 1 with $\sigma^2 = O\left(\frac{G^2\|\mathcal{X}\|^2\log(1/\delta)}{n^2\epsilon^2}\right)$ on dataset ΦS. For $k = \Omega(\log(n))$, the excess risk of $\widetilde{w}$ on $\Phi\mathcal{D}$ is bounded as,*

$$\mathbb{E}[L(\widetilde{w}; \Phi\mathcal{D}) - L(\Phi w^*; \Phi\mathcal{D})]$$
$$\leq O\left(\frac{\sqrt{H}D\|\mathcal{X}\|\|\mathcal{Y}\|}{\sqrt{n}} + \frac{\left(\sqrt{H}\|\mathcal{X}\|\|\mathcal{Y}\| + H\|\mathcal{X}\|^2 D\right) D\sqrt{k\log(1/\delta)}}{n\epsilon}\right).$$

Proof. Let $S^{(i)}$ be the dataset where the i-th data point is replaced by an i.i.d. point (x', y') and let $\widetilde{w}^{(i)}$ be the corresponding output of Noisy-SGD on $\Phi S^{(i)}$. Define $\overline{S} := \{S, (x', y')\}$ and let $H(\Phi, \overline{S})$ denote an upper bound on the smoothness parameter of the family of loss function $\{\ell(w; (\Phi x; y))\}_{(x,y)\in\overline{S}}$.

We want to apply Theorem 8, but the theorem requires that $n \geq \frac{2H\|\mathcal{X}\|^2D^2}{\|\mathcal{Y}\|^2}$. In the proof below, we will use it to bound $H(\Phi, \overline{S}) \leq \frac{n\|\mathcal{Y}\|^2}{D^2}$. We use the JL property to get this. Let $\alpha \leq 1$ be a parameter to be set later. Note that $H(\Phi, \overline{S}) \leq H \sup_{(x,y)\in\overline{S}} \|\Phi x\|^2 \leq H(1+\alpha)\|\mathcal{X}\|^2 \leq 2H\|\mathcal{X}\|^2$ with probability at least $1-\delta$ for $k = O\left(\frac{\log(2n/\delta)}{\alpha^2}\right)$. Thus, if we assume $n \geq \frac{2H\|\mathcal{X}\|^2D^2}{\|\mathcal{Y}\|^2}$, then w.h.p. $H(\Phi, \overline{S}) \leq \frac{n\|\mathcal{Y}\|^2}{D^2}$. Also from the JL property, $\|\Phi w^*\| \leq 2D$.

Decomposing excess risk and writing generalization gap as on-average stability, we have

$$\mathbb{E}_{\Phi,S}[L(\widetilde{w}; \Phi\mathcal{D}) - L(\Phi w^*; \Phi\mathcal{D})]$$
$$= \mathbb{E}_{\Phi,S}[L(\Phi^\top\widetilde{w}; \mathcal{D}) - \widehat{L}(\Phi^\top\widetilde{w}; S)] + \mathbb{E}_{\Phi,S}[\widehat{L}(\Phi^\top\widetilde{w}; S)] - \widehat{L}(\Phi w^*; \Phi S)]$$
$$= \mathbb{E}_{\overline{S},\Phi}[\ell(\widetilde{w}^{(i)}; (\Phi x_i, y_i)) - \ell(\widetilde{w}; (\Phi x_i, y_i)) + \widehat{L}(\Phi^\top\widetilde{w}; S) - \widehat{L}(\Phi w^*; \Phi S)].$$

We now fix the randomness of $\overline{S}$ and bound the terms in high probability w.r.t. the random Φ. Let $\mathrm{IAS}(\overline{S}, i) = \left\|\widehat{w} - \widehat{w}^{(i)}\right\|$ denote the instance argument stability. Re-

peating the analysis in Theorem 8, from Eqn. (A.5), the first term is bounded as,

$$\begin{aligned}
&\ell(\widetilde{w}^{(i)};(\Phi x_i,y_i))-\ell(\widetilde{w};(\Phi x_i,y_i))\\
&\leq \frac{\sqrt{H(\Phi,\overline{S})D}}{\sqrt{n}Y}\left(\ell(\widehat{w};(\Phi x_i,y_i))-\widehat{L}(\Phi w^*;\Phi S)\right)\\
&+\left(\frac{\sqrt{H(\Phi,\overline{S})n}\|\mathcal{Y}\|}{D}+\frac{H(\Phi,\overline{S})}{2}\right)\mathrm{IAS}(\overline{S},i)^2+\frac{\sqrt{H(\Phi,\overline{S})D\|\mathcal{Y}\|}}{\sqrt{n}}\\
&\leq \frac{2\sqrt{H}\|\mathcal{X}\|D}{\sqrt{n}Y}\left(\ell(\widehat{w};(\Phi x_i,y_i))-\widehat{L}(\Phi w^*;\Phi S)\right)\\
&+\frac{4\sqrt{Hn}\|\mathcal{X}\|\|\mathcal{Y}\|}{D}\mathrm{IAS}(\overline{S},i)^2+\frac{2\sqrt{HD\|\mathcal{Y}\|}\|\mathcal{X}\|}{\sqrt{n}}
\end{aligned}\tag{A.8}$$

where the last inequality holds from application of JL property: with probability at least $1-\delta$, $H(\Phi,\overline{S})\leq\frac{n\|\mathcal{Y}\|^2}{D^2}$ (from the lower bound on n) and $H(\Phi,\overline{S})\leq 2H\|\mathcal{X}\|^2$.

As in the proof of Lemma 6, $\mathrm{IAS}(\overline{S},i)$ is bounded as,

$$\begin{aligned}
\mathrm{IAS}(\overline{S},i)^2&\leq\frac{8H(\Phi,\overline{S})\eta^2T}{n}\frac{1}{n}\sum_{j=1}^{T}\mathbb{E}[\ell(w_j;(\Phi x_i,y_i))+\ell(w'_j;(\Phi x',y'))-2\widehat{L}(\Phi w^*;\Phi S)]\\
&+\frac{8H(\Phi,\overline{S})\eta^2T\|\mathcal{Y}\|^2}{n}\\
&\leq\frac{16H\|\mathcal{X}\|^2\eta^2T}{n}\frac{1}{n}\sum_{j=1}^{T}\mathbb{E}[\ell(w_j;(\Phi x_i,y_i))+\ell(w'_j;(\Phi x',y'))-2\widehat{L}(\Phi w^*;\Phi S)]\\
&+\frac{16H\|\mathcal{X}\|^2\eta^2T\|\mathcal{Y}\|^2}{n}.
\end{aligned}$$

Substituting the above in equation A.8, taking expectation with respect to $\overline{S}$ and from manipulations as in the proof of Theorem 8, we get that with probability at least $1-\delta$, we have

$$\begin{aligned}
\mathbb{E}_S[L(\Phi^\top\widetilde{w};\mathcal{D})-\widehat{L}(\Phi^\top\widetilde{w};S)]&\leq 2\left(\mathbb{E}[\widehat{L}(\widehat{w};\Phi S)-\widehat{L}(\Phi w^*;\Phi S)]\right)\\
&+\frac{32}{n}\left(\sum_{j=1}^{T}\mathbb{E}[\widehat{L}(w_j;\Phi S)-\widehat{L}(w^*;\Phi S)]\right)+\frac{34\sqrt{\widetilde{H}}D\|\mathcal{Y}\|}{\sqrt{n}}.
\end{aligned}$$

Let $G(\Phi,S)=G=2\|\mathcal{Y}\|\sqrt{H(\Phi,\overline{S})}+2H(\Phi,\overline{S})\|\Phi w^*\|$ denote the Lispchitzness parameter of the family of loss functions $\{\ell(w;(\Phi x,y)\}_{(x,y)\in\overline{S}}$. From the analysis in

Lemma 7, the average regret and excess empirical risk terms are both bounded by the following quantity with high probability.

$$O\left(\frac{\sqrt{H(\Phi,\overline{S})}D\|\mathcal{Y}\|}{\sqrt{n}}+\frac{\sqrt{k}G(\Phi,S)\log(1/\delta)}{n\epsilon}\right)\leq O\left(\frac{\sqrt{H}D\|\mathcal{Y}\|}{\sqrt{n}}+\frac{\sqrt{k}G\log(1/\delta)}{n\epsilon}\right).$$

This gives the following high probability bound,

$$\begin{aligned}&\mathbb{E}[L(\widetilde{w};\Phi\mathcal{D})-L(\Phi w^*;\Phi\mathcal{D})]\\&\leq O\left(\frac{\sqrt{H}D\|\mathcal{X}\|\|\mathcal{Y}\|}{\sqrt{n}}+\frac{\left(\sqrt{H}\|\mathcal{X}\|\|\mathcal{Y}\|+H\|\mathcal{X}\|^2D\right)D\sqrt{k\log(1/\delta)}}{n\epsilon}\right).\end{aligned}$$

For the in-expectation bound, note that in the above proof the JL property was used for: with probability at least $1-\delta$, $\sup_{(x,y)\in\overline{S}}\|\Phi x\|\leq\left(1+\frac{\sqrt{\log(n/\delta)}}{k}\right)\|\mathcal{X}\|$ and $\|\Phi w^*\|\leq\left(1+\frac{\sqrt{\log(n/\delta)}}{k}\right)D$. All these quantities appear in the numerator in the above bound. Therefore the excess risk random variable w.r.t Φ has a tail with a poly $\left(\frac{\sqrt{\log(n/\delta)}}{k}\right)$ term. This is a (non-centered) sub-Weibull random variable and from equivalence of tail and moments bounds (e.g. Theorem 3.1 in [VGNA20]) and $\frac{\log(n)}{k}\leq O(1)$, we get the claimed expectation bound.

□

A.1.5 Constrained Regularized ERM with Output Perturbation

We here state a key result from [SST10]. Let $\mathfrak{R}_n(D,\|\mathcal{X}\|)$ denote the Rademacher complexity of linear predictors with norm bound by D, with n datapoints and norm of each point bounded by $\|\mathcal{X}\|$.

Theorem 31. *[[SST10]] Let ℓ be an H-smooth GLM and let R be a bound on the loss function. For a dataset S of n i.i.d samples, with probability at least $1-\beta$, for all w*

such that $\|w\| \leq D$*, we have*

$$L(w;\mathcal{D}) \leq \widehat{L}(w;S) + O\left(\sqrt{\widehat{L}(w;S)}\left(\sqrt{H}\log^{1.5}(n)\mathfrak{R}_n(D,\|\mathcal{X}\|) + \sqrt{\frac{R\log(1/\beta)}{n}}\right)\right.$$
$$\left. + H\log^3(n)\mathfrak{R}_n^2(D,\|\mathcal{X}\|) + \frac{R\log(1/\beta)}{n}\right). \tag{A.9}$$

Corollary 4. *Let* $\widetilde{w} = \underset{w\in\mathcal{B}(D)}{\arg\min}\left\{\widehat{L}(w;S)\right\}$ *and* $n \geq \frac{HD^2\|\mathcal{X}\|^2}{\|\mathcal{Y}\|^2}$*. Then with probability at least* $1-\beta$ *under the randomness of* S *we have*

$$L(\widetilde{w};\mathcal{D}) - \widehat{L}(\widetilde{w};S) = \widetilde{O}\left(\frac{\sqrt{H}D\|\mathcal{X}\|\|\mathcal{Y}\|\sqrt{\log(1/\beta)}}{\sqrt{n}} + \frac{\|\mathcal{Y}\|^2\log(1/\beta)}{\sqrt{n}}\right).$$

Further, in expectation under the randomness of S *it holds that*

$$\mathbb{E}[L(\widetilde{w};\mathcal{D}) - \widehat{L}(\widetilde{w};S)] = \widetilde{O}\left(\frac{\sqrt{H}\|\mathcal{Y}\|D\,\|\mathcal{X}\|}{\sqrt{n}} + \frac{\|\mathcal{Y}\|^2}{\sqrt{n}}\right).$$

Proof. In our application, w will be the output of regularized ERM $\widetilde{w}$, thus $\widehat{L}(\widetilde{w};S) \leq \widehat{L}_\lambda(0;S) \leq \|\mathcal{Y}\|^2$.

From Lemma 4 we have that $R \leq 3(\|\mathcal{Y}\|^2 + HD^2\|\mathcal{X}\|^2)$. Also, $\mathfrak{R}_n(D,\|\mathcal{X}\|) \leq O\left(\frac{D\|\mathcal{X}\|}{\sqrt{n}}\right)$ [SSBD14]. We now plug in the quantities into the above theorem to get that with probability at least $1-\beta$,

$$\begin{aligned}
&L(\widetilde{w};\mathcal{D}) - \widehat{L}(\widetilde{w};S)\\
&= \widetilde{O}\left(\frac{\sqrt{H}D\|\mathcal{X}\|}{\sqrt{n}}\left(\|\mathcal{Y}\| + \frac{\sqrt{H}D\|\mathcal{X}\|}{\sqrt{n}}\right)\right.\\
&\left. + \frac{\left(\|\mathcal{Y}\| + \sqrt{H}D\|\mathcal{X}\|\right)\sqrt{\log(1/\beta)}}{\sqrt{n}}\left(\|\mathcal{Y}\| + \frac{\left(\|\mathcal{Y}\| + \sqrt{H}D\|\mathcal{X}\|\right)\sqrt{\log(1/\beta)}}{\sqrt{n}}\right)\right)\\
&= \widetilde{O}\left(\frac{\sqrt{H}D\|\mathcal{X}\|\|\mathcal{Y}\|\sqrt{\log(1/\beta)}}{\sqrt{n}} + \frac{\|\mathcal{Y}\|^2\log(1/\beta)}{\sqrt{n}}\right)
\end{aligned}$$

where the last follows by simplifications using the lower bound on n.

For the in expectation result, observe that the above Eqn (A.9) is a tail bound for a sub-Gamma random variable with variance parameter $O\left(\frac{\sqrt{H}D\|\mathcal{X}\|\|\mathcal{Y}\|}{\sqrt{n}}\right)$ and scale parameter $O\left(\frac{\|\mathcal{Y}\|^2}{\sqrt{n}}\right)$. From the equivalence of tail and moment bounds of sub-Gamma random variables, [BLM13] we get the claimed in-expectation bound. □

A.1.6 Proof of Theorem 10

Proof. Recall from Lemma 2 that the (un-regularized) loss is G-Lipschitz on the constraint set with $\|w\| \leq D$, where $\widetilde{G} := \left(\sqrt{H}\,\|\mathcal{Y}\| + HD\,\|\mathcal{X}\|\right)\|\mathcal{X}\|$. Note that from a standard analysis [BE02], we get that ℓ_2 sensitivity (or uniform argument stability) is $O\left(\frac{G}{n\lambda}\right)$, hence $\sigma^2 = O\left(\frac{G^2\log(1/\delta)}{\lambda^2 n^2\epsilon^2}\right)$ ensures (ϵ,δ)-DP.

For the utility analysis, the excess risk can be decomposed as follows,

$$\begin{aligned}&\mathbb{E}[L(\widehat{w};\mathcal{D}) - L(w^*;\mathcal{D})]\\ &= \mathbb{E}[L(\widetilde{w};\mathcal{D}) - \widehat{L}(\widetilde{w};S)] + \mathbb{E}[\widehat{L}(\widetilde{w};S) - \widehat{L}(w^*;S)] + \mathbb{E}[L(\widetilde{w}+\xi;\mathcal{D}) - L(\widetilde{w};\mathcal{D})].\end{aligned}$$

The first term is bounded by the Rademacher complexity result (Corollary 4).

The second term is simply bounded by $\frac{\lambda}{2}\|\widetilde{w}\|^2 \leq \frac{\lambda}{2}D^2$ since $\widetilde{w}$ lies in the constraint set. The third term is bounded using smoothness as follows:

$$\begin{aligned}\mathbb{E}[L(\widetilde{w}+\xi;\mathcal{D}) - L(\widetilde{w};\mathcal{D})] &= \mathbb{E}\left[\phi_y\left(\langle\widetilde{w}+\xi, x\rangle\right) - \phi_y\left(\langle\widetilde{w}, x\rangle\right)\right]\\ &\leq \mathbb{E}\left[\phi_y'(\langle\widetilde{w}, x\rangle)\,\langle\xi, x\rangle + \frac{H}{2}\,|\langle\xi, x\rangle|^2\right]\\ &\leq \frac{H}{2}\sigma^2\,\|\mathcal{X}\|^2 = O\left(\frac{HG^2\,\|\mathcal{X}\|^2\log(1/\delta)}{\lambda^2 n^2\epsilon^2}\right)\end{aligned}$$

where the last inequality follows since $\mathbb{E}\xi = 0$ and $\langle\xi, x\rangle \sim \mathcal{N}(0, \sigma^2\|x\|^2)$. We now plug in $\lambda = \left(\frac{G\sqrt{H}\|\mathcal{X}\|}{D}\right)^{2/3}\frac{(\log(1/\delta))^{1/3}}{(n\epsilon)^{2/3}}$. and $G = \left(\sqrt{H}\,\|\mathcal{Y}\| + HD\,\|\mathcal{X}\|\right)\|\mathcal{X}\|$ to get,

$$\begin{aligned}&\mathbb{E}[L(\widehat{w};\mathcal{D}) - L(w^*;\mathcal{D})]\\ &\leq \widetilde{O}\left(\frac{\sqrt{H}D\,\|\mathcal{X}\|\,\|\mathcal{Y}\| + \|\mathcal{Y}\|^2}{\sqrt{n}} + \frac{\left(\sqrt{H}D\,\|\mathcal{X}\|\right)^{4/3}\|\mathcal{Y}\|^{2/3} + \left(\sqrt{H}D\,\|\mathcal{X}\|\right)^2}{(n\epsilon)^{2/3}}\right).\end{aligned}$$

□

A.1.7 Proof of Theorem 11

Proof. Define $\widehat{L}(w;S) = \frac{1}{n}\sum_{(x,y)\in S}(\langle w, x\rangle - y)^2$. Let $d' < \min\{n, d\}$ and $b, p \in [0,1]$ be parameters to be chosen later. For any $\boldsymbol{\sigma} \in \{\pm 1\}^{d'}$, define the dataset $S_{\boldsymbol{\sigma}}$ which

consists of the union of d' subdatasets, $S_1, ..., S_{d'}$ given as follows. Set $\frac{pn}{d'}$ of the feature vectors in S_j as $\|\mathcal{X}\|e_j$ (the rescaled j'th standard basis vector) and the rest as the zero vector. Set $\frac{pn}{2d}(1+b)$ of the labels as $\boldsymbol{\sigma}_j\|\mathcal{Y}\|$ and $\frac{pn}{2d}(1-b)$ labels as $-\boldsymbol{\sigma}_j\|\mathcal{Y}\|$. Let $w^{\boldsymbol{\sigma}} = \arg\min_{w\in\mathbb{R}^d}\left\{\widehat{L}(w; S_{\boldsymbol{\sigma}})\right\}$ be the ERM minimizer of $\widehat{L}(\cdot; S_{\boldsymbol{\sigma}})$. Following from Lemma 2 of [Sha15] we have that for any $\bar{w}\in\mathbb{R}^d$ that

$$\widehat{L}(\bar{w}; S_{\boldsymbol{\sigma}}) - \widehat{L}(w^{\boldsymbol{\sigma}}; S_{\boldsymbol{\sigma}}) \geq \frac{p\|\mathcal{X}\|^2}{2d'}\sum_{j=1}^{d'}(\bar{w}_j - w_j^{\boldsymbol{\sigma}})^2. \tag{A.10}$$

We will now show lower bounds on the per-coordinate error. Consider any $\boldsymbol{\sigma}$ and $\boldsymbol{\sigma}'$ which differ only at index j for some $j\in[d']$. Note that the datasets $S_{\boldsymbol{\sigma}}$ and $S_{\boldsymbol{\sigma}'}$ differ in $\Delta = \frac{pn}{2d'}[(1+b)-(1-b)] = \frac{pbn}{d'}$ points. Let $\tau = w_j^{\boldsymbol{\sigma}} = \frac{\|\mathcal{Y}\|b}{\|\mathcal{X}\|}$ and $\tau' = w_j^{\boldsymbol{\sigma}'} = -\frac{\|\mathcal{Y}\|b}{\|\mathcal{X}\|}$ (i.e. the j components of the empirical minimizers for S and S'_j respectively). Note that $|w_j^{\boldsymbol{\sigma}} - w_j^{\boldsymbol{\sigma}'}| = \frac{2\|\mathcal{Y}\|b}{\|\mathcal{X}\|}$. We thus have by Lemma 1 that for a certain $b = b(\epsilon, n, d, D, p, \|\mathcal{Y}\|)$, $\mathcal{A}$ must satisfy

$$\mathbb{E}\left[|\mathcal{A}(S_{\boldsymbol{\sigma}})_j - w_j^{\boldsymbol{\sigma}}| + |\mathcal{A}(S_{\boldsymbol{\sigma}'})_j - w_j^{\boldsymbol{\sigma}'}|\right] \geq \frac{1}{4}\frac{\|\mathcal{Y}\|b}{\|\mathcal{X}\|}. \tag{A.11}$$

Since we need $\Delta \leq \frac{1}{\epsilon}$, we must set $b \leq \frac{d'}{pn\epsilon}$. Furthermore, if we are interested in problems with minimizer norm at most D, we need $b \leq \frac{\|\mathcal{X}\|D}{\|\mathcal{Y}\|\sqrt{d'}}$ to ensure the norm of the minimizer is bounded by D. Balancing these two restrictions, we set $d' = \left(\frac{p\|\mathcal{X}\|Dn\epsilon}{\|\mathcal{Y}\|}\right)^{2/3}$ which yields $b = \left(\frac{\|\mathcal{X}\|D}{\|\mathcal{Y}\|\sqrt{pn\epsilon}}\right)^{2/3}$. Assuming such settings of b and d' are possible (e.g. $b\in[0,1]$) we can apply Jensen's inequality and the fact that $(a+b)^2 \leq 2(a^2+b^2)$ to Eqn. (A.11) to obtain

$$\mathbb{E}\left[|\mathcal{A}(S_{\boldsymbol{\sigma}})_j - w_j^{\boldsymbol{\sigma}}|^2 + |\mathcal{A}(S_{\boldsymbol{\sigma}'})_j - w_j^{\boldsymbol{\sigma}'}|^2\right] \geq \frac{D^{4/3}\|\mathcal{Y}\|^{2/3}}{32(\|\mathcal{X}\|pn\epsilon)^{2/3}}.$$

We will now show this implies there exists a $\boldsymbol{\sigma}\in\{\pm1\}^{d'}$ such that $\widehat{L}(\mathcal{A}(S_{\boldsymbol{\sigma}}); S_{\boldsymbol{\sigma}}) - \widehat{L}(w^{\boldsymbol{\sigma}}; S_{\boldsymbol{\sigma}}) = \Omega(\frac{(\|\mathcal{X}\|D)^{4/3}\|\mathcal{Y}\|^{2/3}p^{1/3}}{(n\epsilon)^{2/3}})$. To prove this, we have the following analysis. Let $U = \{\pm1\}^{d'}$ and let $\boldsymbol{\sigma}_{-j}$ denote the vector $\boldsymbol{\sigma}$ with its j'th component negated. We

have

$$\sup_{\boldsymbol{\sigma}\in U}\left\{\mathbb{E}\left[\widehat{L}(\mathcal{A}(S_{\boldsymbol{\sigma}});S_{\boldsymbol{\sigma}})-\widehat{L}(w^{\boldsymbol{\sigma}};S_{\boldsymbol{\sigma}})\right]\right\}\geq\frac{1}{|U|}\sum_{\boldsymbol{\sigma}\in U}\mathbb{E}\left[\widehat{L}(\mathcal{A}(S_{\boldsymbol{\sigma}});S_{\boldsymbol{\sigma}})-\widehat{L}(w^{\boldsymbol{\sigma}};S_{\boldsymbol{\sigma}})\right]$$
$$=\frac{p\|\mathcal{X}\|^2}{d'|U|}\sum_{j\in[d']}\sum_{\boldsymbol{\sigma}\in U}\mathbb{E}\left[|\mathcal{A}(S_{\boldsymbol{\sigma}})_j-w_j^{\boldsymbol{\sigma}}|^2\right]$$
$$=\frac{p\|\mathcal{X}\|^2}{d'|U|}\sum_{j\in[d']}\sum_{\boldsymbol{\sigma}\in U:\boldsymbol{\sigma}_j=1}\mathbb{E}\Big[|\mathcal{A}(S_{\boldsymbol{\sigma}})_j-w_j^{\boldsymbol{\sigma}}|^2+|\mathcal{A}(S_{\boldsymbol{\sigma}_{-j}})_j-w_j^{\boldsymbol{\sigma}_{-j}}|^2\Big]$$
$$\geq\frac{(\|\mathcal{X}\|D)^{4/3}\|\mathcal{Y}\|^{2/3}p^{1/3}}{128(n\epsilon)^{2/3}}.$$

We recall this bound holds providing the settings of d' and b fall into the range $[1,\min\{n,d\}]$ and $[0,1]$ respectively. First note $b>0$ always. Furthermore, $d'<\min\{n,d\}$ and $b<1$ whenever

$$\frac{\|\mathcal{X}\|^2D^2}{\|\mathcal{Y}\|^2n\epsilon}\leq p\leq\min\left\{1,\frac{d^{3/2}\|\mathcal{Y}\|}{\|\mathcal{X}\|Dn\epsilon}\right\}. \tag{A.12}$$

In the following, assume $D\leq\frac{\|\mathcal{Y}\|\min\left\{\sqrt{n\epsilon},\sqrt{d}\right\}}{\|\mathcal{X}\|}$. Note this is no loss of generality as we can obtain problem instances with arbitrarily large D by adding a dummy point with $x=c'e_{d'+1}$ and $y=\|\mathcal{Y}\|$ where c' is arbitrarily small so that the minimizer norm is D. Under this assumption on D, using the restrictions on p, it can be verified that $d'>1$ whenever $\epsilon>\frac{1}{n}$. Thus we have $b\in[0,1]$ and $d'\in[1,\min\{n,d\}]$ as required. Furthermore this assumption on D implies $\frac{\|\mathcal{X}\|^2D^2}{\|\mathcal{Y}\|^2n\epsilon}\leq\min\left\{1,\frac{d^{3/2}\|\mathcal{Y}\|}{\|\mathcal{X}\|Dn\epsilon}\right\}$ and thus a valid setting of p is possible. We now turn to setting p in a way which satisfies (A.12). We consider two cases, the high and low dimensional regimes.

Case 1: $d\geq\left(\frac{D\|\mathcal{X}\|n\epsilon}{\|\mathcal{Y}\|}\right)^{2/3}$. Setting $p=1$ gives a lower bound of $\Omega\left(\min\left\{\|\mathcal{Y}\|^2,\frac{D^{4/3}\|\mathcal{X}\|^{4/3}\|\mathcal{Y}\|^{2/3}}{32(n\epsilon)^{2/3}}\right\}\right)$, where the min with $\|\mathcal{Y}\|^2$ from the upper bound on D.

Case 2: $d\leq\left(\frac{D\|\mathcal{X}\|n\epsilon}{\|\mathcal{Y}\|}\right)^{2/3}$. Setting $p=\frac{d^{3/2}\|\mathcal{Y}\|}{D\|\mathcal{X}\|n\epsilon}$ we obtain a bound of $\Omega\left(\min\left\{\|\mathcal{Y}\|^2,\frac{\sqrt{d}D\|\mathcal{X}\|\|\mathcal{Y}\|}{n\epsilon}\right\}\right)$ which we note is no larger than the bound from Case 1 in the low dimensional regime and no smaller than the bound from Case 1 in the high dimensional regime. Thus we can write the total bound as the minimum of these two

bounds. The $\|\mathcal{Y}\|^2$ term again comes from the restriction on D.

To obtain results for arbitrary H, we can set $\widehat{L}(w;S) = \frac{H}{2n}\sum_{(x,y)\in S}(\langle w,x\rangle - \frac{2}{\sqrt{H}}y)^2$. This satisfies H-smoothness and loss bounded at zero by $\|\mathcal{Y}\|^2$. Then substituting $\|\mathcal{Y}\|$ in the previous expressions for $2\|\mathcal{Y}\|/\sqrt{H}$ and multiplying through by $H/2$ one obtains the claimed bound. □

A.2 Missing Proofs from Section 2.4 (Lipschitz GLMs)

A.2.1 Proof of Theorem 12

Proof. Note that from a standard analysis [BE02], we get that ℓ_2 sensitivity of the regularized minimizer $\widetilde{w}$ is $\left(\frac{2G\|\mathcal{X}\|}{n\lambda}\right)$, hence $\sigma^2 = \frac{4G^2\|\mathcal{X}\|^2\log(1/\delta)}{\lambda^2 n^2\epsilon^2}$ ensures (ϵ,δ)-DP.

For the utility analysis, the excess risk can be decomposed as follows,

$$\begin{aligned}&\mathbb{E}[L(\widehat{w};\mathcal{D}) - L(w^*;\mathcal{D})]\\ &= \mathbb{E}[L(\widetilde{w};\mathcal{D}) - \widehat{L}(\widetilde{w};S)] + \mathbb{E}[\widehat{L}(\widetilde{w};S) - \widehat{L}(w^*;S)] + \mathbb{E}[L(\widetilde{w}+\xi;\mathcal{D}) - L(\widetilde{w};\mathcal{D})]\end{aligned}$$

The first term is bounded as $O\left(\frac{DG\|\mathcal{X}\|}{\sqrt{n}}\right)$ from Rademacher complexity results on bounded linear predictors [SSBD14].

The second term is simply bounded by $\frac{\lambda}{2}\|\widetilde{w}\|^2 \le \frac{\lambda}{2}D^2$ since $\widetilde{w}$ is the regularized ERM. The third term is bounded via the following

$$\begin{aligned}\mathbb{E}[L(\widetilde{w}+\xi;\mathcal{D}) - L(\widetilde{w};\mathcal{D})] &= \mathbb{E}\left[\phi_y\left(\langle\widetilde{w}+\xi,x\rangle\right) - \phi_y\left(\langle\widetilde{w},x\rangle\right)\right]\\ &\le \mathbb{E}\left[G\left|\langle\xi,x\rangle\right|\right]\\ &\le G\sigma\|\mathcal{X}\| \le \frac{4G^2\|\mathcal{X}\|^2\sqrt{\log(1/\delta)}}{\lambda n\epsilon}\end{aligned}$$

where the last inequality follows since $\mathbb{E}\xi = 0$ and $\langle\xi,x\rangle \sim \mathcal{N}(0,\sigma^2\|x\|^2)$. Now plugging in $\lambda = \frac{G\|\mathcal{X}\|(\log(1/\delta))^{1/4}}{D\sqrt{n\epsilon}}$ obtains the following result

$$\mathbb{E}[L(\widehat{w};\mathcal{D}) - L(w^*;\mathcal{D})] = O\left(\frac{DG\|\mathcal{X}\|}{\sqrt{n}} + \frac{DG\|\mathcal{X}\|\log(1/\delta)^{1/4}}{\sqrt{n\epsilon}}\right).$$

□

A.2.2 Upper Bound using JL Method

Theorem 32. *Let* $k = O\left(\log\left(2n/\delta\right) n\epsilon\right), \sigma^2 = \frac{8TG^2\|\mathcal{X}\|^2\log(2/\delta)}{n^2\epsilon^2}, \eta = \frac{D}{G\|\mathcal{X}\|\left(1+\frac{\sqrt{k\log(2/\delta)}}{n\epsilon}\right)T^{3/4}}$ *and* $T = n^2$. *Algorithm 2 satisfies* (ϵ, δ)*-differential privacy. Given a dataset* S *of* n *i.i.d samples, the excess risk of its output* $\widetilde{w}$ *is bounded as*

$$\mathbb{E}[\varepsilon_{\mathsf{risk}}(\widehat{w})] = \widetilde{O}\left(\frac{GD\|\mathcal{X}\|}{\sqrt{n}} + \frac{GD\|\mathcal{X}\|}{\sqrt{n\epsilon}}\right).$$

Proof. Let $\alpha \leq 1$ be a parameter to be set later. From the JL property, with $k = O\left(\frac{\log(2n/\delta)}{\alpha^2}\right)$, with probability at least $1 - \frac{\delta}{2}$, for feature vectors have $\|\Phi x_i\| \leq (1+\alpha)\|x_i\| \leq 2\|\mathcal{X}\|$, and $\|\Phi w^*\|^2 \leq 2\|w^*\|^2 \leq 2D^2$. Thus, gradient of loss for data point (x, y) in S at any w is bounded as $\|\ell(w; (\Phi x, y)\| = \phi'_y(\langle w, \Phi x\rangle)\|\Phi x\| \leq 2G\|\mathcal{X}\|$. The privacy guarantee thus follows from the privacy of DP-SCO and post-processing.

For the utility guarantee, we decompose excess risk as:

$$\begin{aligned}
&\mathbb{E}\left[L(\widehat{w}; \mathcal{D}) - L(w^*; \mathcal{D})\right] \\
&= \mathbb{E}\left[L(\widetilde{w}; \Phi\mathcal{D}) - L(\Phi w^*; \Phi S)\right] + \mathbb{E}\left[L(\Phi w^*; \Phi S) - L(w^*; \mathcal{D})\right]. \qquad \text{(A.13)}
\end{aligned}$$

The first term in Eqn. (A.13) is bounded by the utility guarantee of the DP-SCO method. In particular, from Lemma 9 (below), we have

$$\mathbb{E}\left[L(\widetilde{w}; \Phi\mathcal{D}) - L(\Phi w^*; \Phi S)\right] \leq \frac{GD\|\mathcal{X}\|}{\sqrt{n}} + O\left(\frac{G\|\mathcal{X}\| D\sqrt{k}}{n\epsilon}\right).$$

For the second term in Eqn (A.13), we use JL-property and Lipschitzness of GLM:

$$\begin{aligned}
\mathbb{E}\left[L(\Phi w^*; \Phi S) - L(w^*; \mathcal{D})\right] &\leq G\mathbb{E}\left[|\langle \Phi x, \Phi w^*\rangle - \langle x, w^*\rangle|\right] \\
&\leq \alpha G\|\mathcal{X}\|\|w^*\| \\
&= O\left(\frac{G\|\mathcal{X}\| D\sqrt{\log\left(2n/\delta\right)}}{\sqrt{k}}\right).
\end{aligned}$$

Combining, we get,

$$\mathbb{E}\left[L(\widehat{w};\mathcal{D}) - L(w^*;\mathcal{D})\right] \leq \widetilde{O}\left(\frac{G\left\|\mathcal{X}\right\| D}{\sqrt{n}} + \frac{G\left\|\mathcal{X}\right\| D\sqrt{k}}{n\epsilon} + \frac{G\left\|\mathcal{X}\right\| D}{\sqrt{k}}\right).$$

Balancing parameters by setting $k = \widetilde{O}(n\epsilon)$ gives the claimed bound. □

Lemma 9. *Let $\Phi \in \mathbb{R}^{d\times k}$ be a data-oblivious JL matrix. Let $S = \{(x_i, y_i)\}_{i=1}^n$ of n i.i.d data points and let $\Phi S := \{(\Phi x_i, y_i)\}_{i=1}^n$. Let $\widetilde{w} \in \mathbb{R}^k$ be the average iterate returned noisy SGD procedure with Gaussian noise variance $\sigma^2 = O\left(\frac{G^2\|\mathcal{X}\|^2\log(1/\delta)}{n^2\epsilon^2}\right)$ on dataset ΦS. For $k = \Omega\left(\log\left(n\right)\right)$, the excess risk of $\widetilde{w}$ on $\Phi\mathcal{D}$ is bounded as,*

$$\mathbb{E}_{\Phi,S}[L(\widetilde{w};\Phi\mathcal{D}) - L(\Phi w^*;\Phi\mathcal{D})] \leq O\left(\frac{GD\|\mathcal{X}\|}{\sqrt{n}} + \frac{GD\|\mathcal{X}\|\sqrt{k\log\left(1/\delta\right)}}{n\epsilon}\right).$$

Proof. The proof uses the analysis for excess risk bound for Noisy SGD in [BFGT20] with the JL transform. Let $S^{(i)}$ be the dataset where the i-th data point is replaced by an i.i.d. point (x', y') and let $\widetilde{w}^{(i)}$ be the corresponding output of Noisy-SGD on $\Phi S^{(i)}$. Define $\overline{S} := \{S, (x', y')\}$ and let $G(\Phi, \overline{S})$ denote an upper bound Lipschitzness parameter of the family of loss functions $\{\ell(w; (\Phi x; y))\}_{(x,y)\in\overline{S}}$.

We decompose the excess risk as:

$$\begin{aligned}
&\mathbb{E}_{\Phi,S}[L(\widetilde{w};\Phi\mathcal{D}) - L(\Phi w^*;\Phi\mathcal{D})] \\
&= \mathbb{E}_{\Phi,S}[L(\Phi^\top\widetilde{w};\mathcal{D}) - \widehat{L}(\Phi^\top\widetilde{w};S)] + \mathbb{E}_{\Phi,S}[\widehat{L}(\Phi^\top\widetilde{w};S)] - \widehat{L}(\Phi w^*;\Phi S)].
\end{aligned}$$

A well-known fact (see [SSBD14]) is that generalization gap is equal to on-average-stability:

$$\begin{aligned}
\mathbb{E}_{\Phi,S}[L(\Phi^\top\widetilde{w};\mathcal{D}) - \widehat{L}(\Phi^\top\widetilde{w};S)] &= \mathbb{E}_{\overline{S},\Phi}[\ell(\Phi^\top\widetilde{w}^{(i)};(x_i,y_i)) - \ell(\Phi^\top\widetilde{w};(x_i,y_i))] \\
&= \mathbb{E}_{\overline{S},\Phi}[\ell(\widetilde{w}^{(i)};(\Phi x_i,y_i)) - \ell(\widetilde{w};(\Phi x_i,y_i))].
\end{aligned}$$

From the analysis in Theorem 3.3 from [BFGT20], it follows that

$$\begin{aligned}
\ell(\widetilde{w}^{(i)};(\Phi x_i,y_i)) - \ell(\widetilde{w};(\Phi x_i,y_i)) &\leq G(\Phi;\overline{S})\left\|\widetilde{w}^{(i)} - \widetilde{w}\right\| \\
&\leq O\left(G(\Phi;\overline{S})^2\left(\eta\sqrt{T} + \frac{\eta T}{n}\right)\right). \qquad \text{(A.14)}
\end{aligned}$$

where the last equality follows from the GLM structure of the loss function.

From analysis of Noisy-SGD [BST14], the other term (excess empirical risk) can be bounded as

$$\begin{aligned}&\widehat{L}(\widetilde{w};\Phi S)] - \widehat{L}(\Phi w^*;\Phi S)\\&\leq O\left(\eta\left(G(\Phi;\overline{S})^2 + \frac{kG(\Phi;\overline{S})^2\log(1/\delta)}{n^2\epsilon^2}\right) + \frac{\|\Phi w^*\|^2}{T\eta}\right)\end{aligned} \tag{A.15}$$

We now take expectation with respect to Φ. Note that $G(\Phi,\overline{S}) = \sup_{(x,y)\in\overline{S}}\sup_w |g_x(\langle w,\Phi x\rangle)|\,\|\Phi x\|$, where g_x is the an element of the sub-differential of ϕ_x at $\langle w,x\rangle$. By the Lipschitzness assumption $|g_x(\langle w,\Phi x\rangle)| \leq G$ and from the JL property, with probability at least $1-\delta$, $\sup_{(x,y)\in\overline{S}}\|\Phi x\| \leq \left(1+\frac{\sqrt{\log(n/\delta)}}{k}\right)\|\mathcal{X}\|$. Similarly, $\left\|\Phi^\top w^*\right\| \leq \left(1+\frac{\sqrt{\log(n/\delta)}}{k}\right)D$. Observe that the tail is that of a (non-centered) sub-Gaussian random variable with variance parameter $O\left(\frac{1}{k}\right)$. Using the equivalence of tail and moment bounds of sub-Gaussian random variables [BLM13] and the fact that $\frac{\log(n)}{k} \leq O(1)$, we get the following bounds for Eqn. (A.14) and (A.15):

$$\mathbb{E}_\Phi\left[\ell(\widetilde{w}^{(i)};(\Phi x_i,y_i)) - \ell(\widetilde{w};(\Phi x_i,y_i)\right] \leq O\left(G^2\|\mathcal{X}\|^2\left(\eta\sqrt{T}+\frac{\eta T}{n}\right)\right)$$

$$\mathbb{E}_\Phi\left[\widehat{L}(\widetilde{w};\Phi S)] - \widehat{L}(\Phi w^*;\Phi S)\right] \leq \widetilde{O}\left(\eta\left(G^2\|\mathcal{X}\|^2 + \frac{kG^2\|\mathcal{X}\|^2\log(1/\delta)}{n^2\epsilon^2}\right) + \frac{D^2}{T\eta}\right).$$

Finally, as in [BFGT20], taking expectation with respect $\overline{S}$ in the two inequalities above, setting $\eta = \frac{D}{G\|\mathcal{X}\|\left(1+\frac{\sqrt{k\log(1/\delta)}}{n\epsilon}\right)T^{3/4}}$ and $T = n^2$ and combining gives the claimed bound. □

A.2.3 Proof of Theorem 13

The proof follows from the more general Theorem 33 stated below. Instantiating Theorem 33 with $p = q = 2$ satisfies all the requirements of our Theorem 13. Finally, let w^* be the population risk minimizer of the hard instance in Theorem 33. We then

have,

$$\begin{aligned}\mathbb{E}[L(\mathcal{A}(S);\mathcal{D}) - \min_w L(w;\mathcal{D})] &= \mathbb{E}[L(\mathcal{A}(S);\mathcal{D}) - L(w^*;\mathcal{D})] \\ &\geq \mathbb{E}[L(\mathcal{A}(S);\mathcal{D}) - \min_{w:\|w\|_2\leq D} L(w;\mathcal{D})] \\ &\geq \mathbb{E}[L(\widehat{w};\mathcal{D}) - L(\widetilde{\mathrm{w}};\mathcal{D})] \\ &= \Omega\left(GD\|\mathcal{X}\|\min\left(1, \frac{1}{\sqrt{n\epsilon}}, \frac{\sqrt{\mathrm{rank}}}{n\epsilon}\right)\right).\end{aligned}$$

This completes the proof.

A.2.4 Lower bound for Non-Euclidean DP-GLM

Theorem 33. *Let $G, \|\mathcal{Y}\|, \|\mathcal{X}\|, D > 0$, $\epsilon \leq 1.2, \delta \leq \epsilon$ and $p, q \geq 1$. For any (ϵ, δ)-DP algorithm $\mathcal{A}$, there exists sets $\mathcal{X}$ and $\mathcal{Y}$ such that for any $x \in \mathcal{X}$, $\|x\|_q \leq 1$, a distribution $\mathcal{D}$ over $\mathcal{X} \times \mathcal{Y}$, a G-Lipschitz GLM loss bounded at zero by $\|\mathcal{Y}\|$ and a $\widetilde{w}$ with $\|\widetilde{w}\|_p \leq D$ such that the output of $\mathcal{A}$ on $S \sim \mathcal{D}^n$ satisfies*

$$\mathbb{E}[L(\mathcal{A}(S);\mathcal{D}) - L(\widetilde{w};\mathcal{D})] = \Omega\left(GD\|\mathcal{X}\|\min\left(1, \frac{1}{(n\epsilon)^{1/p}}, \frac{(rank)^{(p-1)/p}}{n\epsilon}\right)\right).$$

Proof. The construction below is from [SSTT20] with some changes. We provide the complete proof below. Consider the following 1-Lipschitz loss function:

$$\ell(w;(x,y)) = |y - \langle w, x\rangle|.$$

Firstly, we argue that we can assume $\|\mathcal{Y}\| = \infty$. This is because for any arbitrarily large value of y we can translate the above function below so that the the loss is bounded by $\|\mathcal{Y}\|$. However, this translation doesn't change the excess risk. Also, as in [BST14], it suffices to consider $G = 1$, since we can simply scale the 1-Lipschitz loss function and get a factor of the G in the lower bound. Let $d' \leq \min(\mathrm{rank}, n\epsilon)$, $0 \leq \alpha \leq 1$ and $\beta > 0$ be parameters to be set later. Since $d' \leq \mathrm{rank}$, without loss of generality, we will represent the features x as d' dimensional vectors.

Consider a distribution $\mathcal{D}$, where $x = e_0 := \vec{0}$ with probability $1 - \alpha$, and with probability α, $x \sim \mathrm{Unif}(\{\|\mathcal{X}\| e_i\}_{i=1}^{d'})$. Note that $\|x\|_q \leq \|\mathcal{X}\|$ and for any $q \geq 1$ for

$x \in \text{supp}(\mathcal{D}_x)$ where $\mathcal{D}_x$ denotes the marginal of $\mathcal{D}$ w.r.t. the x variable. This implies the loss $w \mapsto \ell(w;(x,y))$ is $\|\mathcal{X}\|$-Lipschitz in ℓ_q-norm, when $x \in \text{supp}(\mathcal{D}_x)$ and for all y. Let the fingerprinting code $z \in \{0,1\}^{d'}$ be drawn from a product distribution with mean $\mu \in [0,1]^{d'}$ where each co-ordinate $\mu_i \sim \text{Beta}(\beta,\beta)$. Finally we have $y = \frac{D}{(d')^{1/p}} \langle x, z \rangle$. Let $S = \{(x_i, y_i)\}_{i=1}^n$ be n i.i.d. samples drawn from $\mathcal{D}$. Define $\widetilde{w} = \frac{D}{(d')^{1/p}}\mu$; note that $\|\widetilde{w}\|_p \le D$.

Let $\mathcal{A}$ be any (ϵ,δ)-DP algorithm, which given S outputs $\widehat{w}$. Its excess risk with respect to $\widetilde{w}$ can be lower bounded as,

$$\begin{aligned}\mathbb{E}[L(\widehat{w};\mathcal{D}) - L(\widetilde{w};\mathcal{D})] &= \mathbb{E}[|y - \langle \widehat{w}, x\rangle| - |y - \langle \widetilde{w}, x\rangle|] \\ &\ge \mathbb{E}[|\langle \widehat{w} - \widetilde{w}, x\rangle| - 2\,|y - \langle \widetilde{w}, x\rangle|]. \end{aligned} \tag{A.16}$$

The last term above is upper bounded as:

$$\mathbb{E}[|y - \langle \widetilde{w}, x\rangle|] = \frac{D}{(d')^{1/p}}\mathbb{E}[|\langle z, x\rangle - \langle \mu, x\rangle|] = \frac{D\,\|\mathcal{X}\|}{(d')^{1/p}}\frac{\alpha}{d'}\sum_{i=1}^{d'}\mathbb{E}\,|z_i - \mu_i|\,.$$

By direct computation,

$$\mathbb{E}\,|z_i - \mu_i| = \mathbb{E}\,[|1-\mu_i|\,\mu_i + |-\mu_i|\,(1-\mu_i)] = 2\mathbb{E}\mu_i\,(1-\mu_i) = \frac{2\beta}{1+2\beta}.$$

This gives us that

$$\mathbb{E}[|y - \langle \widetilde{w}, x\rangle|] \le \frac{2\alpha\beta D\,\|\mathcal{X}\|}{(1+2\beta)\,(d')^{(p+1)/p}}. \tag{A.17}$$

The first term in the right hand side Inequality (A.16) is,

$$\mathbb{E}[|\langle \widehat{w} - \widetilde{\mu}, x\rangle| = \frac{D}{(d')^{1/p}}\mathbb{E}\left|\left\langle \frac{(d')^{1/p}}{D}\widehat{w} - \mu, x\right\rangle\right| = \frac{D\,\|\mathcal{X}\|\,\alpha}{(d')^{(p+1)/p}}\mathbb{E}\sum_{j=1}^{d'}\left|\frac{(d')^{1/p}}{D}\widehat{w}_j - \mu_j\right|. \tag{A.18}$$

Define $v \in \mathbb{R}^{d'}$ with $v_i = \frac{(d')^{1/p}\widehat{w}_i}{D}$. Note that,

$$\sum_{j=1}^{d'}\left|\frac{(d')^{1/p}}{D}\widehat{w}_j - \mu_j\right| = \|v-\mu\|_1 \ge \|v-\mu\|_2 \ge \|\widehat{v}-\mu\|_2 \ge \|\widehat{v}-\mu\|_2^2$$

where the first inequality above follows from relationship between ℓ_1 and ℓ_2 norm, the second follows from projection property and $\widehat{v}$ denotes the projection of v onto $[0,1]^d$, and the last inequality follows from boundedness of coordinates of $\widehat{v} - \mu$.

The key step now is application of the fingerprinting lemma (simplified below, see Lemma B.1 in [SSTT20] for complete statement) from [SU17] which roughly speaking, "relates the error to correlation"; for any $\widehat{v} \in [0,1]^{d'}$, we have

$$\mathbb{E}\left\|\widehat{v} - \mu\right\|_2^2 \geq \frac{d'}{4\left(1+2\beta\right)} - \frac{1}{\beta}\sum_{i=1}^{n}\mathbb{E}\left\langle \widehat{v}, z_i - \mu\right\rangle. \tag{A.19}$$

As in [SSTT20], the correlation simplifies as

$$\begin{aligned}\mathbb{E}\left\langle \widehat{v}, z_i - \mu\right\rangle &= \sum_{j=0}^{d'}\mathbb{E}[\left\langle \widehat{v}, z_i - \mu\right\rangle | x_i = e_j]\mathbb{P}[x_i = \|\mathcal{X}\|e_j] \\ &= \frac{\alpha}{d'}\mathbb{E}[\widehat{v}_j\left(z_i - \mu\right)_j | x_i = \|\mathcal{X}\|e_j]\end{aligned}$$

where the last equality follows since $\widehat{v}$ only depends on the coordinate of z_i for which $x_i = 1$. Now, let $\widehat{v}^{\sim i}$ denotes the solution obtained if z_i were replaced by an independent sample. From boundedness, $\left\|\widehat{v}_j\left(z_i - \mu\right)_j\right\|_\infty \leq 1$. Using differential privacy, we have,

$$\begin{aligned}\mathbb{E}[\widehat{v}_j\left(z_i - \mu\right)_j | x_i = \|\mathcal{X}\|e_j] &\leq \left(e^\epsilon - 1\right)\mathbb{E}\left[\left|\widehat{v}_j^{\sim i}\left(z_i - \mu\right)_j\right| | x_i = \|\mathcal{X}\|e_j\right] + \delta \\ &\leq \left(e^\epsilon - 1\right) + \delta \leq 3\epsilon\end{aligned}$$

where the last inequality uses the assumption that $\epsilon \leq 1.2$ and $\delta \leq \epsilon$. Plugging this in Eqn. (A.19), we get

$$\mathbb{E}\left\|\widehat{v} - \mu\right\|_2^2 \geq \frac{d'}{4\left(1+2\beta\right)} - 3\alpha n\epsilon.$$

Plugging the above in Eqn. (A.18), we get

$$\mathbb{E}[|\left\langle \widehat{w} - \widetilde{w}, x\right\rangle|] \geq \frac{D\left\|\mathcal{X}\right\|\alpha}{{d'}^{(p+1)/p}}\left(\frac{d'}{4\left(1+2\beta\right)} - 3\alpha n\epsilon\right).$$

Using the above, and the bound in Eqn. (A.16), we get that,

$$\begin{aligned}\mathbb{E}[L(\widehat{w};\mathcal{D}) - L(\widetilde{w};\mathcal{D})] &\geq \frac{D\left\|\mathcal{X}\right\|\alpha}{{d'}^{(p+1)/p}}\left(\frac{d'}{4\left(1+2\beta\right)} - 3\alpha n\epsilon\right) - \frac{2\alpha\beta D\left\|\mathcal{X}\right\|}{\left(1+2\beta\right)\left(d'\right)^{1/p}} \\ &= \frac{D\left\|\mathcal{X}\right\|\alpha}{\left(1+2\beta\right)\left(d'\right)^{1/p}}\left(\frac{1}{4} - \frac{3\alpha\left(1+2\beta\right)n\epsilon}{d'} - 2\beta\right).\end{aligned}$$

Now, we set $\beta = \frac{1}{16}$. When rank $\leq 48n\epsilon$ we set $d' = \text{rank}$ and $\alpha = \min\left(\frac{d'}{48(1+2\beta)n\epsilon}, 1\right)$. The minimum term becomes $\frac{d'}{48(1+2\beta)n\epsilon}$ and obtain an excess risk lower bound of,

$$\mathbb{E}[L(\widehat{w}; \mathcal{D}) - L(\widetilde{w}; \mathcal{D})] \geq \frac{D \left\|\mathcal{X}\right\| (d')^{(p-1)/p}}{1024n\epsilon}.$$

On the other hand, when rank $> 48n\epsilon$, set $\alpha = 1$ and $d' = \lfloor 48n\epsilon \rfloor$ (so that it is at least 1) to get an excess risk lower bound of,

$$\mathbb{E}[L(\widehat{w}; \mathcal{D}) - L(\widetilde{w}; \mathcal{D})] \geq \min\left(\frac{D \left\|\mathcal{X}\right\|}{16 (n\epsilon)^{1/p}}, D \left\|\mathcal{X}\right\|\right).$$

Finally, note that we can arbitrarily increase the rank of the construction beyond $48n\epsilon$ by adding datapoints with orthogonal feature vectors of small enough magnitude and arbitrary labels. Combining the two lower bounds obtains the claimed bound. □

Corollary 5. *Let $G, D > 0$, $\epsilon \leq 1.2, \delta \leq \epsilon$ and $p, q \geq 1$. Let $\mathcal{W} \subset \mathbb{R}^d$ such that for any $w \in \mathcal{W}$, $\left\|w\right\|_p \leq D$. For any (ϵ, δ)-DP algorithm $\mathcal{A}$, there exists a set $\mathcal{Z}$, a distribution $\mathcal{D}$ over $\mathcal{Z}$ and a loss function $w \mapsto \ell(w; z)$, which is convex, G-Lipschitz w.r.t. ℓ_q norm for $w \in \mathcal{W}$, for all $z \in \mathcal{Z}$ such that the output of $\mathcal{A}$ on $S \sim \mathcal{D}^n$ (which may not lie in $\mathcal{W}$), satisfies*

$$\mathbb{E}_{\mathcal{A},S}\left[\mathbb{E}_{z\sim\mathcal{D}}\ell(\mathcal{A}(S); z) - \min_{w\in\mathcal{W}} \mathbb{E}_{z\sim\mathcal{D}}\ell(w; z)\right] = \Omega\left(GD \min\left(1, \frac{1}{(n\epsilon)^{1/p}}, \frac{d^{(p-1)/p}}{n\epsilon}\right)\right).$$

Using generalization properties of differential privacy, we get the same bound as above for excess empirical risk; see Corollary B.4 in [SSTT20] for details.

A.3 Missing Details for Section 2.5 (Adapting to $\left\|w^*\right\|$)

A.3.1 Generalized Exponential Mechanism

Theorem 34. *[RS16] Let $K > 0$ and $S \in \mathcal{Z}^n$. Let $q_1, ..., q_K$ be functions s.t. for any adjacent datasets S, S' it holds that $|q_j(S) - q_j(S')| \leq \gamma_j : \forall j \in [K]$.*

There exists an Algorithm, GenExpMech*, such that when given sensitivity-score pairs* $(\gamma_1, q_1(S)), ..., (\gamma_N, q_N(S))$*, privacy parameter* $\epsilon > 0$ *and confidence parameter* $\beta > 0$*, outputs* $j \in [N]$ *such that with probability at least* $1 - \beta$ *satisfies* $q_j(S) \leq \min_{j \in [N]} \left\{ q_j(S) + \frac{4\gamma_j \log(N/\beta)}{\epsilon} \right\}$.

A.3.2 Proof of Theorem 14

Note that by assumptions on $\mathcal{A}$, the process of generating $w_1, ..., w_K$ is $(\epsilon/2, \delta/2) - DP$. Furthermore, by Assumption 2 with probability at least $\delta/2$ the sensitivity values passed to GenExpMech bound sensitivity. Thus by the privacy guarantees of GenExpMech and composition we have that the entire algorithm is (ϵ, δ)-DP.

We now prove accuracy. In order to do so, we first prove that with high probability every $\widetilde{L}_j$ is an upper bound on the true population loss of w_j. Specifically, define $\tau_j = \frac{\Delta(D_j)\log(4K/\beta)}{n} + \sqrt{\frac{4\|\mathcal{Y}\|^2 \log(4K/\beta)}{n}}$ (i.e. the term added to each $L(w_j; S_2)$ in Algorithm 4). Note it suffices to prove

$$\mathbb{P}\left[\exists j \in [K] : |L(w_j; S_2) - L(w_j; \mathcal{D})| \geq \tau_j\right] \leq \frac{\beta}{2}. \tag{A.20}$$

Fix some $j \in [K]$. Note that the non-negativity of the loss implies that $\ell(w_D^*; (x, y)^2) \geq 0$. The excess risk assumption then implies that $\mathop{\mathbb{E}}_{(x,y)\sim\mathcal{D}}[\ell(w_j; (x, y))^2] \leq 4\|\mathcal{Y}\|^2$, which in turn bounds the variance. Further, with probability at least $1 - \frac{\beta}{4K}$ it holds that for all $(x, y) \in S_2$ that $\ell(w, (x, y)) \leq \Delta_0 + \Delta(D)$. Thus by Bernstein's inequality we have

$$\mathbb{P}\left[|L(w; S_2) - L(w; \mathcal{D})| \geq t\right] \leq \exp\left(-\frac{t^2 n^2}{\Delta(D_j) t n + 4n\|\mathcal{Y}\|^2}\right) + \frac{\beta}{4K}$$

Thus it suffices to set $t = \frac{\Delta(D_j)\log(4K/\beta)}{n} + \sqrt{\frac{4\|\mathcal{Y}\|^2 \log(4K/\beta)}{n}}$ to ensure $\mathbb{P}\left[|L(w; S_2) - L(w; \mathcal{D})| \geq t\right] \leq \frac{\beta}{2K}$. Taking a union bound over all $j \in K$ establishes (A.20). We now condition on this event for the rest of the proof.

Now consider the case where $j^* \neq 0$ and $\|w^*\| \leq 2^K$. Note that the unconstrained minimizer w^* is the constrained minimizer with respect to any $\mathcal{B}(r)$ for $r \geq \|w^*\|$.

With this in mind, let $j' = \min_{j \in [K]} \{j : w^* \in \mathcal{B}(2^j)\}$ (i.e. the index of the smallest ball containing w^*). In the following we condition on the event that $\forall j \in [K], j \geq j'$, the parameter vector w_j satisfies excess population risk at most $\mathsf{ERR}(2^j)$. We note by Assumption 2 that this (in addition to the event given in (A.20)) happens with probability at least $1 - \frac{3\beta}{4}$. By the guarantees of GenExpMech, with probability at least $1 - \beta$ we (additionally) have

$$\begin{aligned} L(w_{j^*}; \mathcal{D}) \leq L(w_{j^*}; S_2) + \tau_{j^*} &\leq \min_{j \in [K]} \left\{ L(w_j; S_2) + \tau_j + \frac{4\Delta(D_j) \log(4K/\beta)}{n\epsilon} \right\} \\ &\leq L(w_{j'}; S_2) + \tau_{j'} + \frac{4\Delta(D_{j'}) \log(4K/\beta)}{n\epsilon} \\ &\leq L(w_{j'}; \mathcal{D}) + 2\tau_{j'} + \frac{4\Delta(D_{j'}) \log(4K/\beta)}{n\epsilon}. \end{aligned}$$

Since $2^{j'} \leq \max\{2\|w^*\|, 1\}$ we have

$$\begin{aligned} &L(w_{j^*}; \mathcal{D}) - L(w^*; \mathcal{D}) \\ &\leq L(w_{j'}; \mathcal{D}) - L(w^*; \mathcal{D}) + 2\tau_{j'} + \frac{4\Delta(D_{j'}) \log(4K/\beta)}{n\epsilon} \\ &\leq \mathsf{ERR}(2\max\{\|w^*\|, 1\}) + 2\tau_{j'} + \frac{4\Delta(\max\{2\|w^*\|, 1\}) \log(4K/\beta)}{n\epsilon} \\ &\leq \mathsf{ERR}(2\max\{\|w^*\|, 1\}) + \sqrt{\frac{4\|\mathcal{Y}\|^2 \log(4K/\beta)}{n}} + \frac{5\Delta(\max\{2\|w^*\|, 1\}) \log(4K/\beta)}{n\epsilon} \end{aligned}$$

where the second inequality comes from the fact the assumption that $\|w^*\| \leq \|\mathcal{Y}\|^2$. Now note that by the assumption that $\mathsf{ERR}(2^K) \geq \|\mathcal{Y}\|^2$, whenever $\|w^*\| \geq 2^K$ it holds that $\|\mathcal{Y}\|^2 \leq \mathsf{ERR}(\|w^*\|)$. However since the sensitivity-score pair $(0, \|\mathcal{Y}\|^2)$ is passed to GenExpMech, the excess risk of the output is bounded by at most $\|\mathcal{Y}\|^2$ by the guarantees of GenExpMech).

A.3.3 Proof of Theorem 15

Let $\widehat{w}$ denote the output of the regularized output perturbation method with boosting and noise and privacy parameters $\epsilon' = \frac{\epsilon}{K}$ and $\delta' = \frac{\delta}{K}$. We have by Theorem 37 that

with probability at least $1 - \frac{\beta}{4K}$ that

$$L(\widehat{w}; \mathcal{D}) - L(w^*; \mathcal{D})$$
$$= \widetilde{O}\left(\frac{\sqrt{H}D \|\mathcal{X}\| \|\mathcal{Y}\| + \|\mathcal{Y}\|^2}{\sqrt{n}} + \frac{\left(\left(\sqrt{H}D \|\mathcal{X}\|\right)^{4/3} \|\mathcal{Y}\|^{2/3} + \left(\sqrt{H}D \|\mathcal{X}\|\right)^2\right)}{(n\epsilon)^{2/3}} \right.$$
$$\left. + \frac{(\|\mathcal{Y}\|^2 + HD^2\|\mathcal{X}\|^2)}{n\epsilon} + \frac{\left(\|\mathcal{Y}\| + \sqrt{H}D\|\mathcal{X}\|\right)}{\sqrt{n}} \right).$$

Note that this is no smaller than $\|\mathcal{Y}\|^2$ when $D = \Omega\left(\max\left\{\frac{\|\mathcal{Y}\|\sqrt{n\epsilon}}{\|\mathcal{X}\|\sqrt{H}}, \frac{\|\mathcal{Y}\|^2(n\epsilon)^{2/3}}{\sqrt{H}\|\mathcal{X}\|^2}\right\}\right)$, and thus it suffices to set $K = \Theta\left(\log\left(\max\left\{\frac{\|\mathcal{Y}\|\sqrt{n}}{\|\mathcal{X}\|\sqrt{H}}, \frac{\|\mathcal{Y}\|^2(n\epsilon)^{2/3}}{\sqrt{H}\|\mathcal{X}\|^2}\right\}\right)\right)$ to satisfy the condition of the Theorem statement.

Let σ_j denote the level noise used for when the guess for $\|w^*\|$ is D_j. To establish Assumption 2, by Lemma 11 we have that this assumption is satisfies with $\Delta(D) = \|\mathcal{Y}\|^2 + H\|\mathcal{X}\|^2\sigma_j^2 \log(K/\min\{\beta, \delta\})) + HD^2\|\mathcal{X}\|^2$. In particular, we note for the setting of σ_j implied by Theorem 37 and the setting of K we have for all $j \in [K]$ that $H\|\mathcal{X}\|^2\sigma_j^2 = \widetilde{O}(\|\mathcal{Y}\|^2)$. Thus $\Delta(D) = \widetilde{O}\left(\|\mathcal{Y}\|^2 + HD^2\|\mathcal{X}\|^2\right)$. The result then follows from Theorem 14.

A.3.4 Stability Results for Assumption 2

Lemma 10. *Algorithm 1 run with constraint set $\mathcal{B}(D)$ satisfies Assumption 2 with $\Delta(D) = \|\mathcal{Y}\|^2 + HD^2\|\mathcal{X}\|^2$.*

The proof is straightforward using Lemma 4 (provided in the Appendix). For the output perturbation method, we can obtain similar guarantees. Here however, we must account for the fact that the output may not lie in the constraint set. We also remark that the JL-based method (Algorithm 2) can also enjoy this same bound. However, in this case one must apply the norm adaptation method to the intermediate vector $\widetilde{w}$, as $\Phi^\top \widetilde{w}$ may have large norm.

Lemma 11. *Algorithm 3 run with parameter D and σ satisfies Assumption 2 with $\Delta(D) = \|\mathcal{Y}\|^2 + H\|\mathcal{X}\|^2\sigma^2 \log(K/\delta)) + HD^2\|\mathcal{X}\|^2$*

Proof. Note that since S and S' differ in only one point, it suffices to show that for any $(x,y),(x',y')$ that $\ell(\widehat{w};(x,y)) \leq \|\mathcal{Y}\|^2 + HD^2\|\mathcal{X}\|^2 + H\|\mathcal{X}\|^2\sigma^2\log(K/\delta)$ and similarly for $\ell(\widehat{w},(x',y'))$. Let $w \in \mathcal{B}(D)$ and let $\widehat{w} = w + b$ where $b \sim \mathcal{N}(0, \mathbb{I}_d\sigma^2)$. We have by previous analysis $\ell(\widehat{w};(x,y)) \leq \|\mathcal{Y}\|^2 + HD^2\|\mathcal{X}\|^2 + H\langle b,x\rangle^2$. Since $\langle b,x\rangle$ is distributed as a zero mean Gaussian with variance at most $\|\mathcal{X}\|^2\sigma^2$, we have $\mathbb{P}[|\langle b,x\rangle| \geq t] \leq \exp\left(\frac{-t^2}{\|\mathcal{X}\|^2\sigma^2}\right)$. Setting $t = \|\mathcal{X}\|\sigma\log(K/\delta)$ we obtain $\mathbb{P}[|\langle b,x\rangle|^2 \geq \|\mathcal{X}\|^2\sigma^2\log(K/\delta)] \leq \delta/K$. Thus with probability at least $1-\delta/K$ it holds that $\ell(\widehat{w};(x,y)) \leq \|\mathcal{Y}\|^2 + HD^2\|\mathcal{X}\|^2 + H\|\mathcal{X}\|^2\sigma^2\log(K/\delta)$. □

A.4 Missing Details for Confidence Boosting

Algorithm 15 Confidence Boosting

Input: Dataset S, loss function ℓ, Algorithm $\mathcal{A}$, $\widetilde{\sigma}$ privacy parameters ϵ, δ

1: Split the dataset S into equally sized chunks $\{S_i\}_{i=1}^{m+1}$

2: For each $i \in [m+1]$, $\widehat{w}_i = \mathcal{A}\left(S_i, \frac{\epsilon}{2}, \delta\right)$

3: $i^* = \arg\max_{i\in[m]}\left(-\widehat{L}(\widehat{w}_i; S_{m+1}) + \mathrm{Lap}(0,\widetilde{\sigma})\right)$

Output: $\widehat{w}_{i^*}$

We state the result of the boosting procedure in a general enough setup so as apply to our proposed algorithms. This leads to additional conditions on the base algorithm since our proposed methods may not produce the output in the constrained set.

Theorem 35. *Let ℓ be a non-negative, $\widetilde{H}$ smooth, convex loss function. Let $\epsilon, \delta > 0$. Let $\mathcal{A} : (S,\epsilon,\delta) \mapsto \mathcal{A}(S,\epsilon,\delta)$ be an algorithm such that*

1. *$\mathcal{A}$ satisfies (ϵ,δ)-DP*
2. *For any fixed S, $\mathcal{A}(S)$ is γ^2 sub-Gaussian [Ver18]:*

$$\sup_{\|u\|=1} \mathbb{E}\left[\exp\left(\langle \mathcal{A}(S), u\rangle^2/\gamma^2\right)\right] \leq 2$$

3. *For any fixed S, $\mathbb{P}_{(x,y)}[\ell(\mathcal{A}(S);(x,y)) > \Delta(\gamma,\beta)] < \beta$*

4. *Given a dataset S of n i.i.d. points,* $\mathbb{E}[L(\mathcal{A}(S);\mathcal{D}) - \min_{w\in\mathcal{B}_D} L(w;\mathcal{D})] \leq \mathsf{ERR}\,(n,\epsilon,\gamma)$

Let $\widetilde{\sigma}^2 = \frac{4(\|\mathcal{Y}\|^2+\widetilde{H}\widetilde{\gamma}^2\|\mathcal{X}\|^2)}{n\epsilon}$ *and* $n_0 = \frac{16\gamma^2\log^8(4/\beta)\widetilde{H}}{\|\mathcal{Y}\|^2}$. *Algorithm 15 with Algorithm* $\mathcal{A}$ *as input satisfies* (ϵ,δ)*-DP. Given a dataset* S *of* $n \geq n_0$ *samples, with probability at least* $1-\beta$*, the excess risk of its output* $\widehat{w}_{i^*}$ *is bounded as,*

$$L(\widehat{w};\mathcal{D}) - L(w^*;\mathcal{D}) \leq \widetilde{O}\Bigg(\mathsf{ERR}\left(\frac{n}{4\log(4/\beta)},\frac{\epsilon}{2},\gamma\right) + \frac{2\Delta(\gamma,\beta/2)}{n\epsilon} + \frac{2\Delta\left(\gamma,\frac{\beta}{2n}\right)}{n} + \frac{32\gamma\sqrt{\widetilde{H}}\|\mathcal{Y}\|}{\sqrt{n}} + \frac{16\|\mathcal{Y}\|}{\sqrt{n}} + \frac{128\widetilde{H}\gamma^2}{n}\Bigg).$$

We first establish the following concentration bound for convex $\widetilde{H}$ smooth non-negative functions.

Lemma 12. *Let* ℓ *be a convex* $\widetilde{H}$ *smooth non-negative function. Let* S *be a dataset of* n *i.i.d. samples. Let* w *be a random variable which is* γ^2 *sub-Gaussian and independent of* S *and let* $\Delta(\gamma,\beta)$ *be such that* $\mathbb{P}_{(x,y)}[\ell(w;(x,y)) > \Delta(\gamma,\beta)] \leq \beta$. *Then, with probability at least* $1-\beta$*,*

$$\widehat{L}(w;S) \leq (1+T(n,\beta))\,L(w;\mathcal{D}) + U(n,\beta)$$
$$L(w;\mathcal{D}) \leq (1+T(n,\beta))\,\widehat{L}(w;S) + V(n,\beta)$$

where $T(n,\beta) := \frac{4\gamma\log(4/\beta)\sqrt{\widetilde{H}}}{\|\mathcal{Y}\|\sqrt{n}}$, $U(n,\beta) := \frac{4\gamma\log(4/\beta)\|\mathcal{Y}\|\sqrt{\widetilde{H}}}{\sqrt{n}} + \frac{\|\mathcal{Y}\|\sqrt{\log(2/\beta)}}{\sqrt{n}}$ *and* $V(n,\beta) := \frac{4\gamma\log(4/\beta)\sqrt{\widetilde{H}}\|\mathcal{Y}\|}{\sqrt{n}} + \frac{2\Delta\left(\gamma,\frac{\beta}{4n}\right)\log(2/\beta)}{n} + \frac{\|\mathcal{Y}\|\sqrt{\log(2/\beta)}}{\sqrt{n}} + \frac{48\widetilde{H}\gamma^2\log^2(4/\beta)}{n}$.

Proof. With probability at least $1-\frac{\beta}{4}$, for each $(x,y)\in S, \ell(w;(x,y)) \leq \Delta\left(\gamma,\frac{\beta}{4n}\right)$. We condition on this event and apply Bernstein inequality to the random variable $L(w;\mathcal{D}) - \widehat{L}(w;S)$:

$$\mathbb{P}\left[\left|L(w;\mathcal{D}) - \widehat{L}(w;S)\right| > t\right] \leq \exp\left(-\frac{3nt^2}{6n\mathbb{E}[\left(L(w;\mathcal{D}) - \widehat{L}(w;S)\right)^2] + 2\Delta\left(\gamma,\frac{\beta}{4n}\right)t}\right)$$

This gives us that

$$\left|L(w;\mathcal{D}) - \widehat{L}(w;S)\right| \leq \frac{\Delta\left(\gamma, \frac{\beta}{4n}\right)\log\left(2/\beta\right)}{n} + \sqrt{\mathbb{E}\left(L(w;\mathcal{D}) - \widehat{L}(w;S)\right)^2 \log\left(2/\beta\right)} \tag{A.21}$$

The term $\mathbb{E}[\left(L(w;\mathcal{D}) - \widehat{L}(w;S)\right)^2 = \frac{1}{n}\mathbb{E}[(\ell(w;(x,y)) - \mathbb{E}[\ell(w;(x,y)])^2] \leq \frac{1}{n}\mathbb{E}[(\ell(w;(x,y)))^2]$.

Now,

$$\begin{aligned}\mathbb{E}[(\ell(w;(x,y)))^2] &\leq 2\mathbb{E}[(\ell(w;(x,y)) - \ell(0;(x,y))^2] + 2\mathbb{E}[(\ell(0;(x,y)))^2] \\ &\leq 2\mathbb{E}[(\langle\nabla\ell(w;(x,y)), w\rangle)^2] + 2\|\mathcal{Y}\|^2\end{aligned}$$

where the last step follows from convexity. We now use the fact that w is γ^2-sub-Gaussian, therefore $\langle\nabla\ell(w;(x,y)), w\rangle \leq \gamma\sqrt{\log\left(4/\beta\right)}\,\|\nabla\ell(w;(x,y))\|$ with probability at least $1-\beta/4$. We now use self-bounding property of non-negative smooth functions to get,

$$\begin{aligned}\mathbb{E}[(\ell(w;(x,y)))^2] &\leq 2\mathbb{E}[\|\nabla\ell(w;(x,y))\|^2\gamma^2\log\left(4/\beta\right) + 2\|\mathcal{Y}\|^2 \\ &\leq 8\widetilde{H}\mathbb{E}[\ell(w;(x,y))]\gamma^2\log\left(4/\beta\right) + 2\|\mathcal{Y}\|^2 \\ &= 8\widetilde{H}L(w;\mathcal{D})\gamma^2\log\left(4/\beta\right) + 2\|\mathcal{Y}\|^2\end{aligned}$$

Plugging the above in Eqn (A.21) gives us,

$$\begin{aligned}&\left|L(w;\mathcal{D}) - \widehat{L}(w;S)\right| \\ &\leq \frac{\Delta\left(\gamma, \frac{\beta}{4n}\right)\log\left(2/\beta\right)}{n} + 4\sqrt{\frac{\left(\widetilde{H}L(w;\mathcal{D})\gamma^2\log\left(4/\beta\right) + \|\mathcal{Y}\|^2\right)\log\left(1/\beta\right)}{n}} \\ &\leq \frac{\Delta\left(\gamma, \frac{\beta}{4n}\right)\log\left(2/\beta\right)}{n} + 4\sqrt{\frac{\widetilde{H}L(w;\mathcal{D})}{n}}\gamma\log\left(4/\beta\right) + \frac{\|\mathcal{Y}\|\sqrt{\log\left(2/\beta\right)}}{\sqrt{n}}.\end{aligned} \tag{A.22}$$

A simple application of AM-GM inequality gives,

$$\widehat{L}(w;S) \leq \left(1 + \frac{4\gamma \log(4/\beta)\sqrt{\widetilde{H}}}{\|\mathcal{Y}\|\sqrt{n}}\right) L(w;\mathcal{D}) + \frac{4\gamma \log(4/\beta)\|\mathcal{Y}\|\sqrt{\widetilde{H}}}{\sqrt{n}} + \frac{\|\mathcal{Y}\|\sqrt{\log(2/\beta)}}{\sqrt{n}}$$

This proves the first part of the lemma. For the second part, we use the following fact about non-negative numbers A, B, C [BBL03] (see after proof of Theorem 7)

$$A \leq B + C\sqrt{A} \implies A \leq B + C^2 + \sqrt{B}C$$

Thus, from Eqn. (A.22),

$$\begin{aligned}
L(w;\mathcal{D}) &\leq \widehat{L}(w;S) + \frac{\Delta\left(\gamma, \frac{\beta}{4n}\right)\log(2/\beta)}{n} + \frac{\|\mathcal{Y}\|\sqrt{\log(2/\beta)}}{\sqrt{n}} + \frac{16\widetilde{H}\gamma^2\log^2(4/\beta)}{n} \\
&+ \frac{4\gamma\log(4/\beta)\sqrt{\widetilde{H}}}{\sqrt{n}}\left(\sqrt{\widehat{L}(w;S)} + \sqrt{\frac{\Delta\left(\gamma, \frac{\beta}{4n}\right)\log(2/\beta)}{n}} + \sqrt{\frac{\|\mathcal{Y}\|\sqrt{\log(2/\beta)}}{\sqrt{n}}}\right) \\
&\leq \widehat{L}(w;S) + \frac{4\gamma\log(4/\beta)\sqrt{\widetilde{H}\widehat{L}(w;S)}}{\sqrt{n}} + \frac{\Delta\left(\gamma, \frac{\beta}{4n}\right)\log(2/\beta)}{n} + \frac{\|\mathcal{Y}\|\sqrt{\log(2/\beta)}}{\sqrt{n}} \\
&+ \frac{16\widetilde{H}\gamma^2\log^2(4/\beta)}{n} + \frac{4\gamma\sqrt{\widetilde{H}\Delta\left(\gamma, \frac{\beta}{4n}\right)}\log^{3/2}(4/\beta)}{n} + \frac{4\gamma\sqrt{\widetilde{H}\|\mathcal{Y}\|}\log^{5/4}(2/\beta)}{n^{3/4}} \\
&\leq \left(1 + \frac{4\gamma\log(4/\beta)\sqrt{\widetilde{H}}}{\|\mathcal{Y}\|\sqrt{n}}\right)\widehat{L}(w;S) + \frac{4\gamma\log(4/\beta)\|\mathcal{Y}\|\sqrt{\widetilde{H}}}{\sqrt{n}} + \frac{\Delta\left(\gamma, \frac{\beta}{4n}\right)\log(2/\beta)}{n} \\
&+ \frac{\|\mathcal{Y}\|\sqrt{\log(2/\beta)}}{\sqrt{n}} + \frac{16\widetilde{H}\gamma^2\log^2(4/\beta)}{n} + \frac{4\gamma\sqrt{\widetilde{H}\Delta\left(\gamma, \frac{\beta}{4n}\right)}\log^{3/2}(4/\beta)}{n} \\
&+ \frac{4\gamma\sqrt{\widetilde{H}\|\mathcal{Y}\|}\log^{5/4}(2/\beta)}{n^{3/4}}
\end{aligned}$$

where the last inequality follows from AM-GM inequality. Simplifying the expressions yields the claimed bound. □

Proof of Theorem 35. Since the models $\{\widehat{w}_i\}_{i=1}^m$ are trained on disjoint datasets, by parallel composition $\{\widehat{w}_i\}_{i=1}^m$ satisfies $\left(\frac{\epsilon}{2}, \frac{\delta}{2}\right)$-DP. We know that probability at least

$1-\frac{\delta}{2}$, $\ell(w;(x,y)) \leq \Delta\left(\gamma, \frac{\delta}{2}\right)$. Thus conditioning on this event, from the guarantee of the report noisy max procedure, we have that it satisfies $\left(\frac{\epsilon}{2}\right)$-DP. The privacy proof thus follows from absorbing the failure probability into δ part and adaptive composition.

We proceed to the utility part. Let $\widetilde{w}$ be the model among $\{\widehat{w}_i\}_{i=1}^m$ with minimum empirical error on the set S_{m+1}. The excess risk of each $\widehat{w}_i$ is bounded as,

$$\mathbb{E}[L(\widehat{w}_i;\mathcal{D})] - L(w^*;\mathcal{D}) \leq \mathsf{ERR}\left(\frac{n}{m+1}, \frac{\epsilon}{2}, \gamma\right)$$

From Markov's inequality, with probability at least $3/4$, $L(\widehat{w}_i;\mathcal{D}) \leq L(w^*;\mathcal{D}) + 4\mathsf{ERR}\left(\frac{n}{m+1}, \frac{\epsilon}{2}\right)$. From independence of $\{w_i\}_{i=1}^m$, with probability at least $1-1/4^m = 1-\frac{\beta}{4}$, there exists one model, say $\widehat{w}_{i^*}$, such $L(\widehat{w}_{i^*};\mathcal{D}) \leq L(w^*;\mathcal{D}) + 4\mathsf{ERR}\left(\frac{n}{m+1}, \frac{\epsilon}{2}\right)$.

Also, from the guarantee of Report-Noisy-Max, we have that with probability at least $1-\beta/4$

$$L(\widehat{w};S_{m+1}) \leq L(\widetilde{w};S_{m+1}) + \frac{\Delta(\gamma,\beta/4)(m+1)\log^2(4m/\delta)}{n\epsilon}$$

Now, we apply Lemma 12. From a union bound, with probability at least $1-\frac{\beta}{2}$, all $\{w_i\}_{i=1}^m$ satisfy the inequalities in Lemma 12 with β substituted as $\frac{\beta}{2m}$.

Thus, for the output $\widehat{w}$, probability at least $1-\frac{\beta}{2}$,

$$\begin{aligned}
&L(\widehat{w};\mathcal{D}) \\
&\leq \left(1+T\left(\frac{n}{m+1}, \frac{\beta}{2m}\right)\right) L(\widehat{w};S_{m+1}) + V\left(\frac{n}{(m+1)}, \frac{\beta}{2m}\right) \\
&\leq \left(1+T\left(\frac{n}{m+1}, \frac{\beta}{2m}\right)\right) L(\widetilde{w};S_{m+1}) \\
&+ \left(1+T\left(\frac{n}{m+1}, \frac{\beta}{2m}\right)\right) \frac{\Delta(\gamma,\beta/4)(m+1)\log^2(4m/\delta)}{n\epsilon} + V\left(\frac{n}{(m+1)}, \frac{\beta}{2m}\right) \\
&\leq \left(1+T\left(\frac{n}{m+1}, \frac{\beta}{2m}\right)\right) L(w_{i^*};S_{m+1}) + \frac{2\Delta(\gamma,\beta/4)(m+1)\log^2(4m/\delta)}{n\epsilon} \\
&+ V\left(\frac{n}{(m+1)}, \frac{\beta}{2m}\right)
\end{aligned}$$

where in the above we use that $T\left(\frac{n}{m+1}\right) \leq 1$ given the lower bound on n and the setting of m. Furthermore, the last inequality follows because $\widetilde{w}$ has lowest empirical risk on S_{m+1}. Let $W(n, m, \beta) = \frac{2\Delta(\gamma,\beta/4)(m+1)\log^2(4m/\delta)}{n\epsilon} + V\left(\frac{n}{(m+1)}, \frac{\beta}{2m}\right)$. We now apply the other guarantee in Lemma 12 and the fact that w_{i^*} has small excess risk. With probability at least $1 - \delta/2$,

$$
\begin{aligned}
&L(\widehat{w}; \mathcal{D}) \\
&\leq \left(1 + T\left(\frac{n}{m+1}, \frac{\beta}{2m}\right)\right)^2 L(w_{i^*}; \mathcal{D}) \\
&+ \left(1 + T\left(\frac{n}{m+1}, \frac{\beta}{2m}\right)\right) U\left(\frac{n}{m+1}, \frac{\beta}{2m}\right) + W(n, m, \beta) \\
&\leq L(w^*; \mathcal{D}) + 2T\left(\frac{n}{m+1}, \frac{\beta}{2m}\right) L(w^*; \mathcal{D}) + 4\mathsf{ERR}\left(\frac{n}{m+1}, \frac{\epsilon}{2}\right) \\
&+ 8T\left(\frac{n}{m+1}, \frac{\beta}{2m}\right) 4\mathsf{ERR}\left(\frac{n}{m+1}, \frac{\epsilon}{2}\right) + 2U\left(\frac{n}{m+1}, \frac{\beta}{2m}\right) + W(n, m, \beta)
\end{aligned}
$$

Let $X(n, m, \beta) = 4\mathsf{ERR}\left(\frac{n}{m+1}, \frac{\epsilon}{2}\right) + 8T\left(\frac{n}{m+1}, \frac{\beta}{2m}\right) 4\mathsf{ERR}\left(\frac{n}{m+1}, \frac{\epsilon}{2}\right) + 2U\left(\frac{n}{m+1}, \frac{\beta}{2m}\right) + W(n, m, \beta)$. Note that $m = 4\log(4/\beta)$ and $T\left(\frac{n}{m+1}, \frac{\beta}{2m}\right) \leq \frac{16\gamma\log^4(2/\beta)\sqrt{H}}{\|\mathcal{Y}\|\sqrt{n}}$. Substituting this and using the fact that, $L(w^*; \mathcal{D}) \leq L(0; \mathcal{D}) \leq \|\mathcal{Y}\|^2$, we get that with probability at least $1 - \delta$,

$$
L(\widehat{w}; \mathcal{D}) \leq L(w^*; \mathcal{D}) + \frac{16\gamma\log^4(2/\beta)\sqrt{H}\|\mathcal{Y}\|}{\sqrt{n}} + X(n, 4\log(2/\beta), \beta)
$$

Substituting and simplifying the $X(n, 4\log(4/\beta), \beta)$ we have that

$$
\begin{aligned}
&X(n, 4\log(2/\beta), \beta) \\
&\leq 12\mathsf{ERR}\left(\frac{n}{4\log(4/\beta)}, \frac{\epsilon}{2}, \gamma\right) + \frac{2\Delta(\gamma, \beta/4)\log^3(4/\beta)\log(2/\delta)}{n\epsilon} \\
&+ \frac{2\Delta\left(\gamma, \frac{\beta}{2n}\right)\log^4(8/\beta)}{n} + \frac{16\gamma\log^4(8/\beta)\sqrt{\widetilde{H}}\|\mathcal{Y}\|}{\sqrt{n}} \\
&+ \frac{16\|\mathcal{Y}\|\log^4(8/\beta)}{\sqrt{n}} + \frac{128\widetilde{H}\gamma^2\log^4(8/\beta)}{n}
\end{aligned}
$$

Hence, with probability at least $1-\beta$,

$$L(\widehat{w};\mathcal{D}) \leq L(w^*;\mathcal{D}) + 12\mathsf{ERR}\left(\frac{n}{4\log(4/\beta)}, \frac{\epsilon}{2}, \gamma\right) + \frac{2\Delta(\gamma,\beta/4)\log^3(4/\beta)\log(2/\delta)}{n\epsilon} + \frac{2\Delta\left(\gamma,\frac{\beta}{2n}\right)\log^4(8/\beta)}{n} + \frac{32\gamma\log^4(8/\beta)\sqrt{\widetilde{H}}\|\mathcal{Y}\|}{\sqrt{n}} + \frac{16\|\mathcal{Y}\|\log^4(8/\beta)}{\sqrt{n}} + \frac{128\widetilde{H}\gamma^2\log^4(8/\beta)}{n}$$

□

A.4.1 Boosting the JL Method

Theorem 36. *The boosting procedure (Algorithm 15) using the JL method (Algorithm 2) as Algorithm $\mathcal{A}$ satisfies (ϵ,δ)-DP, and with probability at least $1-\beta$, its output $\widehat{w}_{i^*}$ has excess risk,*

$$L(\widehat{w}_{i^*};\mathcal{D}) - L(w^*;\mathcal{D}) \leq \widetilde{O}\Bigg(\frac{\sqrt{H}D\,\|\mathcal{X}\|\,\|\mathcal{Y}\|}{\sqrt{n}} + \frac{\left(\left(\sqrt{H}D\,\|\mathcal{X}\|\right)^{4/3}\|\mathcal{Y}\|^{2/3} + \left(\sqrt{H}D\,\|\mathcal{X}\|\right)^2\right)}{(n\epsilon)^{2/3}} + \frac{(\|\mathcal{Y}\|^2 + HD^2\|\mathcal{X}\|^2)}{n\epsilon} + \frac{\left(\|\mathcal{Y}\| + \sqrt{H}D\|\mathcal{X}\|\right)}{\sqrt{n}}\Bigg)$$

Proof. For the JL method, we consider the boosting procedure in the k dimension space – this is only for the sake of analysis and the algorithm remains the same. In particular, consider the distribution to $\Phi\mathcal{D}$ which, when sampled from, gives the data point $(\Phi x, y)$ where $(x,y)\sim\mathcal{D}$.

Suppose the boosting procedure gives the following k dimensional models: $\widetilde{w}_1,\cdots\widetilde{w}_m$; note that the norm of all these are bounded by $2D$. Let $\widetilde{w}^* \in \arg\min_{\|w\|\leq 2D} L(w;\Phi\mathcal{D})$. Since the outputs satisfy $\|\widetilde{w}_i\| \leq 2D$, the sub-Gaussian parameter $\gamma = O(D)$. We now compute the other parameter $\Delta(\gamma,\beta)$, which is the high probability bound on loss. Note that for a fixed data point $(\Phi x, y)$, an H smooth, non-negative, bounded at zero loss, at any point w s.t. $\|w\| \leq 2D$

is upper bounded by $2\left(\|\mathcal{Y}\|^2 + 4D^2H\|\Phi x\|^2\right)$. From the JL guarantee, with the given k, the term $\|\Phi x\|^2 \leq 2\|\mathcal{X}\|$ with probability at least $1-\beta/4$. This gives us $\Delta(2D,\beta) = 2\left(\|\mathcal{Y}\|^2 + 16\|\mathcal{X}\|^2 D^2\right)$.

We now invoke 35 substituting Δ, γ and $\widetilde{H} = H\|\mathcal{X}\|^2$ to get that with probability at least $1-\frac{\beta}{2}$ output satisfies,

$$\begin{aligned}
L(\widetilde{w}_{i^*};\Phi\mathcal{D}) \leq L(\widetilde{w}^*;\Phi\mathcal{D}) &+ \frac{128D\sqrt{H}\|\mathcal{X}\|\|\mathcal{Y}\|\log^4(8/\beta)}{\sqrt{n}} \\
&+ \alpha\left(\frac{n}{4\log\left((8/\beta)\right)}, \frac{\epsilon}{2}, 2D\right) + \frac{128\left(\|\mathcal{Y}\|^2 + HD^2\|\mathcal{X}\|^2\right)\log^4(8/\beta)}{n\epsilon} \\
&+ \frac{128\left(\|\mathcal{Y}\| + \sqrt{H}D\|\mathcal{X}\|\right)\log^4(8/\beta)}{\sqrt{n}}
\end{aligned}$$

Define $W := \frac{128D\sqrt{H}\|\mathcal{X}\|\|\mathcal{Y}\|\log^4(8/\beta)}{\sqrt{n}} + \frac{128\left(\|\mathcal{Y}\|^2+HD^2\|\mathcal{X}\|^2\right)\log^4(8/\beta)}{n\epsilon} + \frac{128\left(\|\mathcal{Y}\|+\sqrt{H}D\|\mathcal{X}\|\right)\log^4(8/\beta)}{\sqrt{n}}$. The excess risk of the final output $\widehat{w}_{i^*} = \Phi^\top \widetilde{w}_{i^*}$ is bounded as,

$$\begin{aligned}
L(\widehat{w}_{i^*};\mathcal{D}) - L(w^*;\mathcal{D}) &= L(\widetilde{w}_{i^*};\Phi\mathcal{D}) - L(w^*;\mathcal{D}) \\
&\leq L(\widetilde{w}^*;\Phi\mathcal{D}) + \alpha\left(\frac{n}{4\log(8/\beta)}, \frac{\epsilon}{2}, 2D\right) + W - L(w^*;\mathcal{D}) \\
&\leq L(\Phi w^*;\Phi\mathcal{D}) + \alpha\left(\frac{n}{4\log(8/\beta)}, \frac{\epsilon}{2}, 2D\right) + W - L(w^*;\mathcal{D}) \\
&\leq \alpha\left(\frac{n}{4\log(8/\beta)}, \frac{\epsilon}{2}, 2D\right) + W + \frac{H}{2}\left|\langle\Phi x, \Phi w^*\rangle - \langle x, w^*\rangle\right|^2
\end{aligned}$$

where the last inequality follows from smoothness and that $\nabla L(w^*;\mathcal{D}) = 0$. Finally, from the JL property, with probability at least $1-\frac{\beta}{2}$, $\left|\langle\Phi x, \Phi w^*\rangle - \langle x, w^*\rangle\right|^2 \leq \alpha^2\|w^*\|^2\|\mathcal{X}\|^2$. Combining and substituting the values of α and $\alpha\left(\frac{n}{4\log(8/\beta)}, \frac{\epsilon}{2}\right)$ from Theorem 9 gives the claimed result. □

A.4.2 Boosting Output Perturbation Method

Theorem 37. *The boosting procedure (Algorithm 15) using the output perturbation method (Algorithm 3) as input Algorithm $\mathcal{A}$ satisfies (ϵ,δ)-DP, and with probability at*

least $1-\beta$*, its output* $\widehat{w}_{i^*}$ *has excess risk,*

$$L(\widehat{w}_{i^*};\mathcal{D}) - L(w^*;\mathcal{D}) \leq \widetilde{O}\Bigg(\frac{\sqrt{H}D\left\|\mathcal{X}\right\|\left\|\mathcal{Y}\right\| + \left\|\mathcal{Y}\right\|^2}{\sqrt{n}} + \frac{\left(\left(\sqrt{H}D\left\|\mathcal{X}\right\|\right)^{4/3}\left\|\mathcal{Y}\right\|^{2/3} + \left(\sqrt{H}D\left\|\mathcal{X}\right\|\right)^2\right)}{(n\epsilon)^{2/3}} + \frac{(\left\|\mathcal{Y}\right\|^2 + HD^2\|\mathcal{X}\|^2)}{n\epsilon} + \frac{\left(\left\|\mathcal{Y}\right\| + \sqrt{H}D\|\mathcal{X}\|\right)}{\sqrt{n}}\Bigg)$$

Proof. Firstly, note that $\widetilde{H} = H\|\mathcal{X}\|^2$. We now compute γ and Δ to invoke Theorem 35. Since the algorithm simply adds Gaussian noise of variance $\sigma^2\mathbb{I}_d$ to a vector $\widetilde{w}$ where $\|\widetilde{w}\| \leq D$, we have that $\gamma^2 \leq D^2 + \sigma^2$. For the bound on loss parameter Δ, by direct computation, $\Delta(\gamma,\beta) \leq 2\left(\left\|\mathcal{Y}\right\|^2 + H\left\langle\widetilde{w}+\xi, x\right\rangle^2\right) \leq 2\left\|\mathcal{Y}\right\|^2 + 2HD^2\left\|\mathcal{X}\right\|^2 + 2H\left\langle\xi,x\right\rangle^2 \leq 2\left\|\mathcal{Y}\right\|^2 + 2HD^2\left\|\mathcal{X}\right\|^2 + 2H\sigma^2\log\left(1/\beta\right)$ where the last inequality follows since $\langle\xi,x\rangle \sim \mathcal{N}(0,\|x\|^2)$. Plugging these and the value of σ^2 from Theorem 10 into Theorem 35 gives the claimed bound. □

A.5 Non-private Lower Bounds

We first note a simple one-dimensional lower bound.

Theorem 38. *(Implicit in [SST10]) Let* $\|\mathcal{X}\| \geq 1$ *and* $H \geq 2$*. For any Algorithm* $\mathcal{A}$*, there exists a 1-dimensional* H*-smooth non-negative GLM,* $\ell : \mathbb{R}\times(\mathcal{X}\times\mathcal{Y}) \mapsto \mathbb{R}$*, and a distribution* $\mathcal{D}$ *over* $(\mathcal{X}\times\mathcal{Y})$ *with* $\|w^*\| = \Theta(\min\{\|\mathcal{Y}\|, D\}$ *such that the excess population risk of the output of* $\mathcal{A}$ *when run on* $S\sim\mathcal{D}^n$ *is lower bounded as* $\Omega\left(\frac{\min\{\|\mathcal{Y}\|, D\|\mathcal{X}\|\}}{\sqrt{n}}\right)$.

We remark that this requires a slight modification of the example used in [SST10]. Specifically, therein the loss function is defined as

$$\ell(w,(x,y)) = \begin{cases}(w-y)^2 & |w-y| \leq \frac{1}{2}\\ |w-y| - 1/4 & |w-y| \geq \frac{1}{2}\end{cases}$$

with $y \in \{\pm 1\}$ and $x = 1$. Our statement is obtained by setting the domain of labels as $\{\pm \min\{D, \|\mathcal{Y}\|\}\}$. A reduction in [Sha15] enables lower bounds from problem instances with general $\|\mathcal{X}\|$ and $\|w^*\| = R$ to instances with $\|\mathcal{X}\| = 1$ to $\|w^*\| = R\|\mathcal{X}\|$.

We now show that the lower bounds presented in [Sha15] to the unconstrained setting. We start by stating the original bound from [Sha15] which holds for squared loss.

Theorem 39. *Let $\ell(w, (x, y)) = \frac{1}{2}(\langle w, x\rangle - y)^2$ be the squared loss function and $D > 0$. Then for any algorithm, $\mathcal{A}$, there exists a distribution $\mathcal{D}$ over $(\mathcal{X} \times \mathcal{Y})$ and a constant C such that for $S \sim \mathcal{D}^n$ it holds that $\mathbb{E}\left[L(\mathcal{A}(S); \mathcal{D}) - L(w^*_D; \mathcal{D})\right] \geq C \min\left\{\|\mathcal{Y}\|^2, \frac{D^2\|\mathcal{X}\|^2 + d\|\mathcal{Y}\|^2}{n}, \frac{D\|\mathcal{Y}\|\|\mathcal{X}\|}{\sqrt{n}}\right\}$, where $w^*_D = \underset{w \in \mathcal{B}(D)}{\arg\min}\{L(w; \mathcal{D})\}$.*

We now make three observations. First we note that this Theorem holds even for $\mathcal{A}(S) \notin \mathcal{B}(D)$. Second we note that the construction of the distribution in the above Theorem is such that $\min_{w^* \in \mathcal{B}(D)}\{L(w; \mathcal{D})\} = \min_{w^* \in \mathbb{R}^d}\{L(w; \mathcal{D})\}$. Finally, note that $\frac{H}{2}(\langle w, x\rangle - \frac{y}{\sqrt{H}})^2 = \frac{1}{2}(\sqrt{H}\langle w, x\rangle - y)^2$. This gives the following corollary.

Corollary 6. *Let $D > 0$. Then for any algorithm, $\mathcal{A}$, there exists a distribution $\mathcal{D}$ over $(\mathcal{X} \times \mathcal{Y})$ and an H-smooth non-negative GLM, $\ell : \mathbb{R}^d \times (\mathcal{X} \times \mathcal{Y}) \mapsto \mathbb{R}$, with minimizer $w^* = \underset{w \in \mathbb{R}^d}{\arg\min}\{L(w; \mathcal{D})\}$ such that $\|w^*\| = D$ and for $S \sim \mathcal{D}^n$ it holds that*

$$\begin{aligned}&\mathbb{E}\left[L(\mathcal{A}(S); \mathcal{D}) - L(w^*; \mathcal{D})\right] \\ &= \Omega\left\{\min\left\{\|\mathcal{Y}\|^2, \frac{HD^2\|\mathcal{X}\|^2 + d\|\mathcal{Y}\|^2}{n}, \frac{\sqrt{H}D\|\mathcal{Y}\|\|\mathcal{X}\|}{\sqrt{n}}\right\}\right\}.\end{aligned}$$

A.6 Additional Results

Lemma 13. *Let $\Phi \in \mathbb{R}^{d \times k}$ be a random matrix such that for any $u \in \mathbb{R}^d$, with probability at least $1 - \beta$, $(1 - \alpha)\|u\|^2 \leq \|\Phi u\|^2 \leq (1 + \alpha)\|u\|^2$. Then for any u, v, with probability at least $1 - 2\beta$, $|\langle \Phi u, \Phi v\rangle - \langle u, v\rangle| \leq \alpha \|u\| \|v\|$.*

Proof. Firstly, note that it suffices to prove the result for unit norm vectors u and v. From the norm preservation result, with probability at least $1 - 2\beta$, we have that,

$$(1-\alpha)\left\|u+v\right\|^2 \leq \left\|\Phi(u+v)\right\| \leq (1+\alpha)\left\|u+v\right\|^2$$
$$(1-\alpha)\left\|u-v\right\|^2 \leq \left\|\Phi(u-v)\right\| \leq (1+\alpha)\left\|u-v\right\|^2$$

Therefore, we have

$$\begin{aligned}\langle \Phi u, \Phi v\rangle &= \frac{1}{4}\left(\left\|\Phi\left(u+v\right)\right\|^2 - \left\|\Phi\left(u-v\right)\right\|^2\right) \\ &\leq \frac{1}{4}\left((1+\alpha)\left\|u+v\right\|^2 - (1-\alpha)\left\|u-v\right\|^2\right) \\ &\leq \langle u, v\rangle + \alpha\end{aligned}$$

This gives us that $\langle \Phi u, \Phi v\rangle \leq \langle u, v\rangle + \alpha$. The other inequality follows in the same way. □

Appendix B

Appendix for Chapter 3

B.1 Lower Bounds

B.1.1 Missing Details from DP Empirical Stationarity Lower Bound

Proof of Theorem 17. For any $r > 0$, let $\mathcal{W}_r$ denote the ball of radius r centered at the origin. Let $D = \frac{G}{H}$. Consider the loss function:

$$\ell(w; z) = \begin{cases} \frac{H}{2} \|w - z\|^2 & \text{if } \|w - z\| \leq D \\ G \|w - z\| - \frac{G^2}{2H} & \text{otherwise} \end{cases}$$

The function $\ell(w; z)$ is convex, H-smooth and G-Lispchitz in $\mathbb{R}^d$. We restrict to datasets $S = \{z_i\}_{i=1}^n$ where $z_i \in \mathcal{W}_{D/4}$ for all i, and let $\widehat{L}(w; S) = \frac{1}{n} \sum_{i=1}^n \ell(w; z_i)$ be the empirical risk on S. The unconstrained minimizer of $\widehat{L}(w; S)$ is $w^* = \frac{1}{n} \sum_{i=1}^n z_i$ which lies in $\mathcal{W}_{D/4}$.

For any $w \in \mathcal{W}_{3D/4}$, w lies in the quadratic region around all data points. Hence, from H-strong convexity of $w \mapsto \widehat{L}(w; S)$ on $\mathcal{W}_{3D/4}$, we have that whenever $\widehat{w} \in \mathcal{W}_{3D/4}$,

$$\begin{aligned} \left\|\nabla \widehat{L}(\widehat{w}; S)\right\| \|\widehat{w} - w^*\| &\geq \left\langle \nabla \widehat{L}(\widehat{w}; S), w^* - \widehat{w} \right\rangle \\ &\geq \widehat{L}(\widehat{w}; S) - \widehat{L}(w^*; S) \\ &\geq \frac{H}{2} \|\widehat{w} - w^*\|^2 . \end{aligned}$$

Let E be the event that $\bar{w} \in \mathcal{W}_{3D/4}$ and let $\mathbb{E}_E$ denote the conditional expectation

(conditioned on event E) operator. Then,

$$\mathbb{E}_E \left\|\nabla \widehat{L}(\widehat{w}; S)\right\| \geq \frac{H}{2}\mathbb{E}\left\|\widehat{w} - w^*\right\| \geq \frac{H}{2}\Omega\left(\left(\frac{G}{4H}\right)\min\left(1, \frac{\sqrt{d\log(1/\delta)}}{n\epsilon}\right)\right).$$

where the last inequality follows from known lower bounds for DP mean estimation [SU15, KU20]. We remark that the lower bound in the referenced work is for algorithms which produce outputs in the ball of the same radius as the dataset, i.e. $\mathcal{W}_{D/4}$. However, a simple post-processing argument shows that the same lower bound applies to algorithms which produce output in $\mathcal{W}_{3D/4}$. Specifically, assuming the contrary, we simply project the output in $\mathcal{W}_{3D/4}$ to $\mathcal{W}_{D/4}$: privacy is preserved by post-processing and the distance to the mean cannot increase by the non-expansiveness property of projection to convex sets, hence a contradiction. This gives us,

$$\mathbb{E}_E\left[\left\|\nabla \widehat{L}(\widehat{w}; S)\right\|\right] \geq \Omega\left(G\min\left(1, \frac{\sqrt{d\log(1/\delta)}}{n\epsilon}\right)\right)$$

Let $\widetilde{\mathcal{W}} = \{w : \|w - w^*\| \leq D/2\}$. Since $\widetilde{\mathcal{W}} \subseteq \mathcal{W}_{3D/4}$, we have that the above conditional lower bound applies for $\widehat{w} \in \widetilde{\mathcal{W}}$ as well. We now consider $\widehat{w} \notin \widetilde{\mathcal{W}}$. Let w' be *any* point on the boundary of $\widetilde{\mathcal{W}}$, denoted as $\partial\mathcal{W}$. Note that w' lies in the region where, for any data point, the corresponding loss is a quadratic function. Hence, by direct computation, $\nabla \widehat{L}(w'; S) = H(w' - w^*)$. Therefore,

$$\left\langle \nabla \widehat{L}(w'; S), w' - w^* \right\rangle = H\left\|w' - w^*\right\|^2 = \frac{HD^2}{4}.$$

We now apply Lemma 14 which gives us,

$$\mathbb{E}_{E^c}\left\|\nabla \widehat{L}(\widehat{w}; S)\right\| \geq \frac{HD^2}{4}\cdot\frac{2}{D} = \frac{G}{2},$$

where E^c denotes the complement set of E. We combine the above bounds using the law of total expectation as follows,

$$\begin{aligned}
\mathbb{E}[\|\nabla \widehat{L}(\bar{w}; S)\|] &= \mathbb{E}_E[\|\nabla \widehat{L}(\bar{w}; S)\|]\mathbb{P}(\bar{w} \in E) + \mathbb{E}_{E^c}[\|\nabla \widehat{L}(\bar{w}; S)\|]\mathbb{P}(\bar{w} \in E^c) \\
&= \Omega\left(G\min\left\{1, \frac{\sqrt{d\log(1/\delta)}}{n\epsilon}\right\}\right)\mathbb{P}(\bar{w} \in E) + \Omega(G)\mathbb{P}(\bar{w} \in E^c) \\
&= \Omega\left(G\min\left\{1, \frac{\sqrt{d\log(1/\delta)}}{n\epsilon}\right\}\right).
\end{aligned}$$

This completes the proof.

□

Lemma 14. *Let $G, R \geq 0, d \in \mathbb{N}$. Let $\mathcal{W}_R(w_0)$ denote the Euclidean ball around w_0 of radius R and let $\partial\mathcal{W}_R(w_0)$ denote its boundary. Let $f : \mathbb{R}^d \to \mathbb{R}$ be a differentiable convex function. Suppose $w_0 \in \mathbb{R}^d$ is such that for every $v \in \partial\mathcal{W}_R(w_0)$, $\langle \nabla f(v), v - w_0 \rangle \geq G$, then for any $w \notin \mathcal{W}_R(w_0)$, we have $\|\nabla f(w)\| \geq \frac{G}{R}$.*

Proof. For a unit vector $u \in \mathbb{R}^d$, define directional directive $f'_u(w) = \langle \nabla f(w), u \rangle$. We first show that for any $u \in \mathbb{R}^d : \|u\| = 1$ and any $w' \in \mathbb{R}^d$, the function $f'_u(w' + ru)$ is non-decreasing in $r \in \mathbb{R}_+$. This simply follows from monotonicity of gradients since f is convex. In particular, for any $r' > r > 0$, we have

$$\begin{aligned} f'_u(w' + r'u) - f'_u(w' + ru) &= \langle \nabla f(w' + r'u) - \nabla f(w' + ru), u \rangle \\ &= \frac{1}{r' - r} \langle \nabla f(w' + r'u) - \nabla f(w' + ru), w' + ru - (w' + ru) \rangle \\ &> 0 \end{aligned}$$

We now prove the claim in the lemma statement. Let $w \notin \partial W_R$ and define $u = \frac{w - w_0}{\|w - w_0\|}$. Then from Cauchy-Schwarz inequality and the above monotonicity property, we have,

$$\begin{aligned} \|\nabla f(w)\| \geq \langle \nabla f(w), u \rangle = f'_u(w) \geq f'_u(w_0 + Ru) = \langle \nabla f(w_0 + Ru), u \rangle \\ = \frac{1}{R} \langle \nabla f(v), v - w_0 \rangle \geq \frac{G}{R} \end{aligned}$$

which finishes the proof. □

B.1.2 Non-private Sample Complexity Lower Bound

Theorem 40. *For any G, H, $n, d \in \mathbb{N}$, there exists a distribution $\mathcal{D}$ over some set $\mathcal{Z}$ and a G-Lipschitz, H-smooth (convex) loss function $w \mapsto \ell(w; z)$ such that given n i.i.d samples from $\mathcal{D}$, the output $\widehat{w}$ of any algorithm satisfies,*

$$\mathbb{E} \|\nabla L(\widehat{w}; \mathcal{D})\| = \Omega\left(\frac{G}{\sqrt{n}}\right)$$

Proof. We construct a hard instance in $d = 1$ dimension. Let $p \in [0, 1]$ be a parameter to be set later and let $v \in \{-1, 1\}$ be chosen by an adversary. Let the data domain $\mathcal{Z} = \{-1, 1\}$ and consider the distribution $\mathcal{D}$ on $\mathcal{Z}$ as follows:

$$z = \begin{cases} 1 & \text{with probability } \frac{1+vp}{2} \\ -1 & \text{with probability } \frac{1-vp}{2} \end{cases}$$

Note that $\mathbb{E}[z] = vp$. Consider the loss function $\ell(w; z)$ as

$$\ell(w; z) = \frac{G}{2} wz + \frac{H}{2} \Delta(w)$$

where Δ is the Huber regularization function, defined as,

$$\Delta(w) = \begin{cases} |w|^2 & \text{if } |w| \leq \frac{G}{2H} \\ \frac{G|w|}{H} - \frac{G^2}{4H^2} & \text{otherwise} \end{cases}$$

Note that the loss function $w \mapsto \ell(w; z)$ is convex, G-Lipschitz and H-smooth in $\mathbb{R}^d$, for all z. The population risk function is,

$$L(w; \mathcal{D}) = \frac{G}{2} wpv + \frac{H}{2} \Delta(w)$$

Let $\widehat{w}$ be output some algorithm given n i.i.d. samples from $\mathcal{D}$. Consider two cases:

Case 1: $|\widehat{w}| > \frac{G}{2H}$**:** The gradient norm in this case is

$$\begin{aligned} |\nabla L(\widehat{w}; \mathcal{D})|^2 &= \left| \frac{G}{2} vp + \frac{G\widehat{w}}{2|\widehat{w}|} \right|^2 \\ &= \frac{G^2 p^2}{4} + \frac{G^2}{4} + \frac{G^2}{2|\widehat{w}|} vp\widehat{w} \\ &\geq \frac{G^2}{4} - \frac{G^2}{2} p \\ &= \frac{G^2}{4} - \frac{G^2}{8\sqrt{n}} \\ &\geq \frac{G^2}{8} \end{aligned}$$

where the first inequality follows since $v\frac{\widehat{w}}{|\widehat{w}|} \geq -1$, the third equality follows by setting $p = \frac{1}{\sqrt{16n}}$ and the second inequality follows since $n \geq 1$. We therefore have that $\mathbb{E}\,|\nabla L(\widehat{w}; \mathcal{D})| \geq \frac{G}{2\sqrt{2}}$.

Case 2: $|\widehat{w}| \leq \frac{G}{2H}$**:** In this case, the gradient norm is,

$$|\nabla L(\widehat{w}; \mathcal{D})|^2 = \left|\frac{G}{2} vp + H\widehat{w}\right|^2$$

Suppose there exists an algorithm with output $\widehat{w}$, which, with n samples guarantees that $\mathbb{E}\,|\nabla L(\widehat{w}; \mathcal{D})| < o\left(\frac{G}{\sqrt{n}}\right)$. Then from Markov's inequality, with probability at least 0.9, we have that $|\nabla L(\widehat{w}; \mathcal{D})|^2 < o\left(\frac{G^2}{n}\right)$. Let $\widetilde{w} = -\frac{2H\widehat{w}}{G}$, then we have that with probability at least 0.9,

$$|\nabla L(\widehat{w}; \mathcal{D})|^2 \leq o\left(\frac{G^2}{n}\right) \iff |vp - \widetilde{w}|^2 < o\left(\frac{1}{n}\right)$$

This contradicts the well-known bias estimation lower bounds, with $p = \frac{1}{\sqrt{16n}}$, using Le Cam's method ([Duc16], Example 7.7), hence $\mathbb{E}\,|\nabla L(\widehat{w}; \mathcal{D})| \geq \Omega\left(\frac{G}{\sqrt{n}}\right)$. Combining the two cases finishes the proof. □

B.2 Missing Results for Empirical Stationary Points

B.2.1 Private Spiderboost

The following lemma largely follows from the analysis in [WJZ+19]. We present a full proof below for completeness.

Lemma 15. *Let the conditions of Lemma 3 be satisfied. Let $\eta \leq \frac{1}{2H}$ and $q \leq O\left(\frac{1}{\tau_2^2 \eta^2}\right)$. Then the output of Private SpiderBoost, $\widehat{w}$ satisfies*

$$\mathbb{E}\left[\left\|\nabla \widehat{L}(\widehat{w}; S)\right\|\right] = O\left(\sqrt{\frac{L_0}{\eta T}} + \tau_1\right). \tag{B.1}$$

Proof. In the following, for any $t \in [T]$, let $s_t = \left\lfloor \frac{t}{q} \right\rfloor q$ (i.e. the index corresponding to the start of the phase containing iteration t).

By a standard analysis for smooth functions we have (recalling that ∇_t is an unbiased estimate of $\nabla \widehat{L}(w_t; S)$ for any $t \in [T]$)

$$\widehat{L}(w_{t+1}; S) \leq \widehat{L}(w_t; S) + \frac{\eta}{2}\left\|\nabla \widehat{L}(w_t; S) - \nabla_t\right\|^2 - \left(\frac{\eta}{2} - \frac{H\eta^2}{2}\right)\|\nabla_t\|^2.$$

Taking expectation we have the following manipulation using the update rule of Algorithm 5, we get

$$\begin{aligned}
&\mathbb{E}\left[\widehat{L}(w_{t+1};S)-\widehat{L}(w_t;S)\right]\\
&\leq \frac{\eta}{2}\mathbb{E}\left[\left\|\nabla\widehat{L}(w_t;S)-\nabla_t\right\|^2\right]-\left(\frac{\eta}{2}-\frac{H\eta^2}{2}\right)\mathbb{E}\left[\left\|\nabla_t\right\|^2\right]\\
&\leq \frac{\eta\tau_2^2}{2}\sum_{k=s_t+1}^{t}\mathbb{E}\left[\left\|w_{k+1}-w_k\right\|^2\right]+\frac{\eta}{2}\mathbb{E}\left[\left\|\nabla_{s_t}-\widehat{L}(w_{s_t};S)\right\|^2\right]\\
&\quad-\left(\frac{\eta}{2}-\frac{H\eta^2}{2}\right)\mathbb{E}\left[\left\|\nabla_t\right\|^2\right]\\
&\leq \frac{\eta^3\tau_2^2}{2}\sum_{k=s_t+1}^{t}\mathbb{E}\left[\left\|\nabla_k\right\|^2\right]+\frac{\eta\tau_1^2}{2}-\left(\frac{\eta}{2}-\frac{H\eta^2}{2}\right)\mathbb{E}\left[\left\|\nabla_t\right\|^2\right],
\end{aligned}$$

where the second inequality follows from Lemma 3 and the last inequality follows from the update rule. Note that if $t=s_t$ the sum is empty. Summing over a given phase we have

$$\begin{aligned}
&\mathbb{E}\left[\widehat{L}(w_{t+1};S)-\widehat{L}(w_{s_t};S)\right]\\
&\leq \frac{\eta^3\tau_2^2}{2}\sum_{k=s_t}^{t}\sum_{j=s_t+1}^{k}\mathbb{E}\left[\left\|\nabla_j\right\|^2\right]+\sum_{k=s_t}^{t}\left[\tfrac{\eta\tau_1^2}{2}-\left(\tfrac{\eta}{2}-\tfrac{H\eta^2}{2}\right)\mathbb{E}\left[\left\|\nabla_k\right\|^2\right]\right]\\
&\leq \frac{\eta^3\tau_2^2 q}{2}\sum_{k=s_t}^{t}\mathbb{E}\left[\left\|\nabla_k\right\|^2\right]+\sum_{k=s_t}^{t}\left[\tfrac{\eta\tau_1^2}{2}-\left(\tfrac{\eta}{2}-\tfrac{H\eta^2}{2}\right)\mathbb{E}\left[\left\|\nabla_k\right\|^2\right]\right]\\
&=-\sum_{k=s_t}^{t}\left[\underbrace{\left(\frac{\eta}{2}-\frac{H\eta^2}{2}-\frac{\eta^3\tau_2^2 q}{2}\right)}_{A}\mathbb{E}\left[\left\|\nabla_k\right\|^2\right]-\frac{\eta\tau_1^2}{2}\right],
\end{aligned}\tag{B.2}$$

where the second inequality comes from the fact that each gradient appears at most q times in the sum. We now sum over all phases. Let $P=\{p_0,p_1,...,\}=\left\{0,q,2q,...,\left\lfloor\frac{T-1}{q}\right\rfloor q,T\right\}$. We have

$$\begin{aligned}
\mathbb{E}\left[\widehat{L}(w_T;S)-\widehat{L}(w_0;S)\right]&\leq\sum_{i=1}^{|P|}\mathbb{E}\left[\widehat{L}(w_{p_i};S)-\widehat{L}(w_{p_{i-1}};S)\right]\\
&\leq-\sum_{t=0}^{T}A\,\mathbb{E}\left[\left\|\nabla_k\right\|^2\right]+\frac{T\eta\tau_1^2}{2}.
\end{aligned}$$

Rearranging the above yields

$$\frac{1}{T}\sum_{t=0}^{T}\mathbb{E}\left[\left\|\nabla_k\right\|^2\right] \leq \frac{L_0}{TA} + \frac{\eta\tau_1^2}{2A}. \tag{B.3}$$

Now let i^* denote the index of $\widehat{w}$ selected by the algorithm. Note that

$$\mathbb{E}\left[\left\|\nabla\widehat{L}(w_{i^*};S)\right\|^2\right] \leq 2\mathbb{E}\left[\left\|\nabla\widehat{L}(w_{i^*};S) - \nabla_{i^*}\right\|^2\right] + 2\mathbb{E}\left[\left\|\nabla_{i^*}\right\|^2\right]. \tag{B.4}$$

The second term above can be bounded via inequality (B.3). To bound the first term we have by Lemma 3 that

$$\begin{aligned}
\mathbb{E}\left[\left\|\nabla_{i^*} - \nabla\widehat{L}(w_{i^*};S)\right\|^2\right] &\leq \tau_2^2 \sum_{k=s_{t^*}+1}^{t^*}\mathbb{E}\left[\left\|w_k - w_{k-1}\right\|^2\right] + \tau_1^2 \\
&= \eta^2\tau_2^2 \sum_{k=s_{t^*}+1}^{t^*}\mathbb{E}\left[\left\|\nabla_k\right\|^2\right] + \tau_1^2 \\
&\leq \frac{q\eta^2\tau_2^2}{T}\sum_{k=0}^{T}\mathbb{E}\left[\left\|\nabla_k\right\|^2\right] + \tau_1^2 \\
&\leq \frac{\tau_2^2\eta^2 q L_0}{TA} + \frac{\eta^3 q\tau_2^2}{2A}\tau_1^2 + \tau_1^2,
\end{aligned}$$

where the last inequality comes from inequality (B.3) and the expectation over i^*. Plugging into inequality (B.4) one can obtain

$$\mathbb{E}\left[\left\|\nabla\widehat{L}(w_{i^*};S)\right\|^2\right] \leq \frac{2L_0}{TA}(1+\tau_2^2\eta^2 q) + \left(\frac{\eta}{A} + 2 + \frac{\tau_2^2\eta^3 q}{A}\right)\tau_1^2. \tag{B.5}$$

Now recall $A = \frac{\eta}{2} - \frac{H\eta^2}{2} - \frac{\eta^3\tau_2^2 q}{2}$. Since $q \leq O\left(\frac{1}{\tau_2^2\eta^2}\right)$ and $\eta \leq \frac{1}{2H}$ we have $A = \Theta(\eta)$. Thus plugging into inequality (B.5) and again using the fact that $q \leq O\left(\frac{1}{\tau_2^2\eta^2}\right)$ we have

$$\mathbb{E}\left[\left\|\nabla\widehat{L}(w_{i^*};S)\right\|^2\right] = O\left(\frac{L_0}{T\eta}(1+\tau_2^2\eta^2 q) + \left(3 + \frac{\tau_2^2\eta^3 q}{A}\right)\tau_1^2\right) = O\left(\frac{L_0}{T\eta} + \tau_1^2\right).$$

The claim then follows from the Jensen inequality. □

For privacy, we will rely on the moments accountant analysis of [ACG+16b]. This roughly gives the same analysis as using privacy amplification via subsampling and the advanced composition theorem, but allows for improvements in log factors. We provide

the following theorem implicit in [ACG+16b] Theorem 1 below. The same result can be obtained using the analysis for [KLL21] Theorem 3.1 which uses the truncated central differential privacy guarantees of the Gaussian mechanism [BDRS18].

Theorem 41 ([ACG+16b, KLL21]). *Let $\epsilon, \delta \in (0, 1]$ and c be a universal constant. Let $D \in \mathcal{Y}^n$ be a dataset over some domain $\mathcal{Y}$, and let $h_1, ..., h_T : \mathcal{Y} \mapsto \mathbb{R}^d$ be a series of (possibly adaptive) queries such that for any $y \in \mathcal{Y}$, $t \in [T]$, $\|h_t(y)\|_2 \leq \lambda_t$. Let $\sigma_t = \frac{c\lambda_t\sqrt{\log(1/\delta)}}{\epsilon} \max\left\{\frac{1}{b}, \frac{\sqrt{T}}{n}\right\}$. Then the algorithm which samples batches of size $B_1, .., B_t$ of size b uniformly at random and outputs $\frac{1}{n}\sum_{y\in B_t} h_t(y) + g_t$ for all $t \in [T]$ where $g_t \sim \mathcal{N}(0, \mathbb{I}\sigma_t^2)$, is (ϵ, δ)-DP.*

We note that the original statement of the Theorem in [ACG+16b] requires $\sigma_t \geq \frac{c\lambda_t\sqrt{T\log(1/\delta)}}{n\epsilon}$ and $T \geq \frac{n^2\epsilon}{b^2}$ (or $T \geq \frac{n^2}{b^2}$ so long as $\epsilon \leq 1$). However, in the case where $T \leq \frac{n^2}{b^2}$, one can simply consider the meta algorithm that does run $T' = \frac{n^2}{b^2}$ steps and only outputs the first T results. This algorithm is at least as private as the algorithm which outputs every result, and under the setting T' the scale of noise is $\frac{8\lambda_t\sqrt{\log(1/\delta)}}{b\epsilon}$.

We can now prove the main result for Private Spiderboost, restated below. We note that the setting of b_2 given below will always be less than n under required conditions. More details are provided in the proof below.

Theorem 42 (Private Spiderboost). *Let $n \geq \max\left\{\frac{(G\epsilon)^2}{L_0 Hd\log(1/\delta)}, \frac{\sqrt{d}\max\left\{1, \sqrt{HL_0}/G\right\}}{\epsilon}\right\}$. Private Spiderboost run with parameter settings $\eta = \frac{1}{2H}$, $b_1 = n$, $b_2 = \left\lfloor \max\left\{\left(\frac{Gn\epsilon}{\sqrt{L_0 Hd\log(1/\delta)}}\right)^{2/3}, \frac{(Gnd\log(1/\delta))^{1/3}}{(HL_0)^{1/6}\epsilon^{2/3}}\right\}\right\rfloor$, $T = \left\lfloor \max\left\{\left(\frac{(L_0H)^{1/4}n\epsilon}{\sqrt{Gd\log(1/\delta)}}\right)^{4/3}, \frac{n\epsilon}{\sqrt{d\log(1/\delta)}}\right\}\right\rfloor$, and $q = \left\lfloor \frac{n^2\epsilon^2}{H^2Td\log(1/\delta)}\right\rfloor$ satisfies*

$$\mathbb{E}\left[\left\|\nabla\widehat{L}(\widetilde{w}; S)\right\|\right] = O\left(\left(\frac{\sqrt{L_0 HGd\log(1/\delta)}}{n\epsilon}\right)^{2/3} + \frac{\sqrt{d\log(1/\delta)}G}{n\epsilon}\right)$$

is (ϵ, δ)-DP and has oracle complexity $\widetilde{O}\left(\max\left\{\left(\frac{n^{5/3}\epsilon^{2/3}}{d^{1/3}}\right), \left(\frac{n\epsilon}{\sqrt{d}}\right)^2\right\}\right)$.

Proof. For privacy, we rely on the moment accountant analysis of the Gaussian mechanism as per Theorem 41. Note that each gradient estimate computed in line 9 has elements with ℓ_2-norm at most G, and this estimate is computed at most $\frac{T}{q}$ times. Similarly, for a gradient variation at step t in line 13 we have norm bound $H\left\|w_t - w_{t-1}\right\|$, and have that at most T such estimates are computed. As such, the scale of noise in both cases ensures the overall algorithm is (ϵ, δ)-DP by Theorem 41.

We now prove the convergence result. To simplify notation in the following, we define $\bar{\alpha} = \frac{\sqrt{d\log(1/\delta)}}{n\epsilon}$. If $b_1 = n$ (full batch gradient), the conditions of Lemma 3 are satisfied with $\tau_1^2 = O\left(\frac{G^2 T\bar{\alpha}^2}{q}\right)$ and $\tau_2^2 = O\left(\frac{H^2}{b_2} + H^2 T\bar{\alpha}^2\right)$ and some setting of q so long as $T \geq q\frac{n^2}{b_1^2} = q$ and $T \geq \frac{n^2}{b_2^2}$. Further, if $b_2 \geq \frac{1}{T\bar{\alpha}^2}$ then $\tau_2^2 = O\left(H^2 T\bar{\alpha}^2\right)$. Thus the condition on q in Lemma 15 is satisfied with $q = \frac{H^2}{\tau_2^2} = \frac{1}{T\bar{\alpha}^2}$ since $\eta = \frac{1}{2H}$

Plugging into Eqn. (B.1) we obtain

$$\begin{aligned}\mathbb{E}\left[\left\|\nabla\widehat{L}(\widetilde{w}; S)\right\|\right] &= O\left(\sqrt{\frac{L_0 H}{T}} + \frac{G\sqrt{T}\bar{\alpha}}{\sqrt{q}}\right) \\ &= O\left(\sqrt{\frac{L_0 H}{T}} + GT\bar{\alpha}^2\right). \end{aligned} \tag{B.6}$$

We now consider the setting of T. Since $q = \frac{1}{T\bar{\alpha}^2}$, it suffices to set $T \geq \frac{1}{\bar{\alpha}}$ to ensure $T \geq q$. We now set $T = \max\left\{\left(\frac{(HL_0)^{1/4}}{\sqrt{G\bar{\alpha}}}\right)^{4/3}, \frac{1}{\bar{\alpha}}\right\}$. Using Eqn. (B.6) above we have

$$\mathbb{E}\left[\left\|\nabla\widehat{L}(\widetilde{w}; S)\right\|\right] = O\left(\left(\sqrt{L_0 H G\bar{\alpha}}\right)^{2/3} + G\bar{\alpha}\right).$$

The claimed rate now follows if there exists a valid setting for b_2 satisfying the previously stated conditions. The restrictions on the batch size implied by T imply we need $b_2 \geq \frac{n}{\sqrt{T}}$ and thus it suffices to have $b_2 \geq \frac{G^{1/3} n\bar{\alpha}^{2/3}}{(HL_0)^{1/6}}$ to satisfy this condition since $T \geq \left(\frac{(HL_0)^{1/4}}{\sqrt{G\bar{\alpha}}}\right)^{4/3}$. We recall that for the setting of q to be valid we also require $b_2 \geq \frac{1}{T\bar{\alpha}^2}$ and because $T \geq \left(\frac{(HL_0)^{1/4}}{\sqrt{G\bar{\alpha}}}\right)^{4/3}$ it suffices that $b_2 \geq \left(\frac{G}{\sqrt{L_0 H\bar{\alpha}}}\right)^{2/3}$. Thus we need $b_2 = \max\left\{\left(\frac{G}{\sqrt{L_0 H\bar{\alpha}}}\right)^{2/3}, \frac{G^{1/3} n\bar{\alpha}^{2/3}}{(HL_0)^{1/6}}\right\}$. Finally, we need $b_2 \leq n$ whenever $q \geq 1$. Note that by the setting of q and T we have $q \leq \left(\frac{G}{\sqrt{L_0 H\bar{\alpha}}}\right)^{2/3}$ and thus $q \geq 1 \implies \left(\frac{\sqrt{HL_0}\bar{\alpha}}{G}\right) \leq 1$. Under this same condition we have $\frac{G^{1/3} n\bar{\alpha}^{2/3}}{(HL_0)^{1/6}} \leq n$. We further have $\left(\frac{G}{\sqrt{L_0 H\bar{\alpha}}}\right)^{2/3} \leq n$

under the assumption $n \geq \frac{(G\epsilon)^2}{L_0 H d \log(1/\delta)}$ given in the theorem statement. It can also be verified that under the condition on n given in the theorem statement that $q \geq 1$. Thus the parameter settings obtain the claimed rate.

Note the number of gradient computations is bounded by

$$O\left(Tb_2 + \frac{Tb_1}{q}\right) = \widetilde{O}\left(\left(\frac{n\epsilon}{\sqrt{d}}\right)^{4/3} \max\left\{\left(\frac{n\epsilon}{\sqrt{d}}\right)^{2/3}, \frac{(nd)^{1/3}}{\epsilon^{2/3}}\right\} + n\left(\frac{n\epsilon}{\sqrt{d}}\right)^{2/3}\right)$$
$$= \widetilde{O}\left(\max\left\{\left(\frac{n\epsilon}{\sqrt{d}}\right)^{2}, \frac{n^{5/3}\epsilon^{2/3}}{d^{1/3}}\right\}\right).$$

□

B.2.2 Additional Discussion of Rate Improvement Challenges

We here give a more detailed version of the informal discussion in Section 3.2.2. We want to emphasize that the goal of the following discussion is not to provide a universal lower bound, but rather to inform future research.

Let $\mathcal{L} : \mathbb{R}^d \mapsto \mathbb{R}$ be a loss function. We say the randomized mapping $\mathcal{O} : \mathbb{R}^d \times (\mathbb{R}^d \cup \perp) \mapsto \mathbb{R}^d$, is a (τ_1, τ_2)-accurate oracle for $\mathcal{L}$ if $\forall w, w' \in \mathbb{R}^d$

$$\mathop{\mathbb{E}}_{\mathcal{O}}\left[\mathcal{O}(w, \perp)\right] = \nabla\mathcal{L}(w), \qquad \mathop{\mathbb{E}}_{\mathcal{O}}\left[\mathcal{O}(w, w')\right] = \nabla\mathcal{L}(w) - \nabla\mathcal{L}(w')$$
$$\mathop{\mathbb{E}}_{\mathcal{O}}\left[\left\|\mathcal{O}(w, \perp) - \nabla\mathcal{L}(w)\right\|^2\right] \leq \tau_1^2, \qquad \mathop{\mathbb{E}}_{\mathcal{O}}\left[\left\|\mathcal{O}(w, w')\right\|^2\right] \leq \tau_2^2 \left\|w - w'\right\|^2.$$

In short, $\mathcal{O}$ is an unbiased and accurate gradient/gradient variation oracle for $\mathcal{L}$. Define

$$\mathfrak{m}(Q, H, \mathcal{L}_0, \tau_1, \tau_2) = \inf_{\mathcal{A}} \sup_{\mathcal{O}, \mathcal{L}} \inf\left\{\alpha : \mathbb{E}\left[\left\|\nabla\mathcal{L}(\mathcal{A}(\mathcal{O}, H, \mathcal{L}_0, \tau_1, \tau_2)\right\|\right] \leq \alpha\right\},$$

where the supremum is taken over H-smooth functions $\mathcal{L}$ satisfying $\mathcal{L}(0) - \arg\min_{w \in \mathbb{R}^d}\{\mathcal{L}(w)\} \leq \mathcal{L}_0$, and (τ_1, τ_2)-accurate oracles for $\mathcal{L}$. The infimum is taken over algorithms which make at most Q calls to $\mathcal{O}$.

We have the following lower bound on $\mathfrak{m}$ (i.e. a lower bound on the accuracy of optimization algorithms which make at most Q queries to the oracle) following from

[ACD+19, Theorem 3] and the fact that the oracle model described above is a special case of the multi-query oracles considered by [ACD+19].

Theorem 43 ([ACD+19]). *Let* $Q, \mathcal{L}_0, H, \tau_1, \tau_2 \geq 0$ *and define* $\alpha = \left(\frac{\mathcal{L}_0 \tau_2 \tau_1}{Q}\right)^{1/3} + \frac{\tau_1}{\sqrt{Q}}$. *If* $d = \widetilde{\Omega}\left(\left[\frac{\mathcal{L}_0 H}{\alpha^2}\right]^2\right)$, *then* $\mathfrak{m}(Q, H, \mathcal{L}_0, \tau_1, \tau_2) = \Omega(\alpha)$.

Now consider $\mathcal{L}$ such that $\mathcal{L}(w) = \frac{1}{n}\sum_{z \in S} \ell(w; z)$ for some G-Lipschitz and H-smooth loss $\ell : \mathbb{R}^d \times \mathcal{X} \mapsto \mathbb{R}$ and $S \in \mathcal{X}^n$. We are interested in designing some $(\widehat{\tau}_1, \widehat{\tau}_2)$-accurate and differentially private oracle, $\widehat{\mathcal{O}}$, which can then be used by an optimization algorithm, $\mathcal{A}$, to obtain an approximate stationary point $\widehat{w} = \mathcal{A}(\widehat{\mathcal{O}}, H, \mathcal{L}_0, \widehat{\tau}_1, \widehat{\tau}_2)$. Specifically, we want $\widehat{\mathcal{O}}$ to be capable of answering Q queries under (ϵ, δ)-DP. A common method for achieving this is to ensure each query to $\mathcal{O}$ is at least $(\frac{\epsilon}{\sqrt{Q}}, \delta)$-DP and use advanced composition (or the more refined moment accountant) analysis. Such a setup encapsulates numerous results in the convex setting [BFTGT19, KLL21], and is even more dominant in non-convex settings [WYX17, ZCH+20, ACG+16b].

Our key observation is that under such a setup, any increase in the number of oracle calls to Q must be met with a proportional increase in the accuracy parameters $(\widehat{\tau}_1, \widehat{\tau}_2)$. Thus, if such an oracle, $\widehat{\mathcal{O}}$ is applied in a black box fashion to a stochastic optimization algorithm $\mathcal{A}$, one can obtain a lower bound on the accuracy of the overall algorithm independent of Q.

Specifically, since estimating the gradient and gradient variation can be viewed as mean estimation problems on n vectors, we can use fingerprinting code arguments to lower bound $\widehat{\tau}_1$ and $\widehat{\tau}_2$ [SU15]. In Lemma 16 below, we prove that any $(\widehat{\tau}_1, \widehat{\tau}_2)$-accurate oracle which ensures that any query is $(\frac{\epsilon}{\sqrt{Q}}, \delta)$-DP must have $\widehat{\tau}_1 = \Omega\left(\frac{G\sqrt{Qd\log(1/\delta)}}{n\epsilon}\right)$ and $\widehat{\tau}_2 = \Omega\left(\frac{H\sqrt{Qd\log(1/\delta)}}{n\epsilon}\right)$. Now, observe that by Theorem 43, we have

$$\mathfrak{m}(Q, H, \mathcal{L}_0, \widehat{\tau}_1, \widehat{\tau}_2) = \Omega\left(\left(\frac{\sqrt{L_0 H G}\sqrt{d\log(1/\delta)}}{n\epsilon}\right)^{2/3} + \frac{G\sqrt{d\log(1/\delta)}}{n\epsilon}\right),$$

which matches our upper bound.

We now remark on several ways the above barrier could be circumvented. The first and most obvious possibility is to employ a different privatization method than private oracles. However, this is particularly difficult in the nonconvex setting as existing methods which avoid private gradients (see e.g. [FKT20b] for several such methods) rely crucially on stability guarantees arising from convexity. Other possible ways to beat the above rate is by designing a stochastic optimization algorithm which leverages the structure of the noise used in private implementations of the oracle or makes use of additional assumptions to beat the $\Omega\left(\left(\frac{\mathcal{L}_0\tau_2\tau_1}{Q}\right)^{1/3} + \frac{\tau_1}{\sqrt{Q}}\right)$ non-private lower bound.

Additional Details on Fingerprinting Bound. We conclude by giving a concrete construction for the fingerprinting argument mentioned above.

Lemma 16. *Let $G, H \geq 0$, $\epsilon = O(1)$, $2^{-\Omega(n)} \leq \delta \leq \frac{1}{n^{1+\Omega(1)}}$ and $\sqrt{d\log(1/\delta)}/(n\epsilon) = O(1)$. Let $\ell, \mathcal{L}, S$ satisfy the assumptions above. Then there exists ℓ, S such that for any oracle, $\mathcal{O}$, which is (τ_1, τ_2)-accurate for $\mathcal{L}$ it holds that*

$$\tau_1 = \Omega\left(\frac{G\sqrt{d\log(1/\delta)}}{n\epsilon}\right) \qquad \text{and} \qquad \tau_2 = \Omega\left(\frac{H\sqrt{d\log(1/\delta)}}{n\epsilon}\right).$$

Proof. In the following, we use u_j to denote the j'th component of some vector u. Let $B = \frac{G}{H\sqrt{d}}$ and define $h : \mathbb{R} \mapsto \mathbb{R}$ as

$$h(z) = \begin{cases} \frac{H}{2}w^2 & \text{if} |w| \leq B \\ \frac{G}{\sqrt{d}}|w| - \frac{G^2}{2dH} & \text{otherwise} \end{cases}$$

Define $d' = \frac{d}{2}$ (assume d is even for simplicity) and for any vector $u \in \mathbb{R}^d$ let $u^{(1)} = [u_1, ..., u_{d'}]^\top$ and $u^{(2)} = [u_{d'+1}, ..., u_d]^\top$. Define $\ell(w; z) = \ell_1(w; z) + \ell_2(w; z)$ where

$$\ell_1(w; z) = \frac{G}{\sqrt{d}}\left\langle w^{(1)}, z^{(1)}\right\rangle, \qquad \ell_2(w; z) = \frac{1}{2}\sum_{j=d'+1}^{d} h(w_j)z_j.$$

Let $\mathcal{W} = \{w : \|w\|_\infty \leq B\}$ and note for any $w \in \mathcal{W}$ we have

$$\nabla \ell(w; z) = [\frac{z_1}{\sqrt{d}}, ..., \frac{z_{d'}}{\sqrt{d}}, w_{d'+1} z_{d'+1}, ..., w_d z_d]^\top,$$
$$\nabla^2 \ell_2(w; z) = H \cdot \mathsf{Diag}(0, ..., 0, z_{d'+1}, ..., z_d)$$

That is, the Hessian of $\ell_2(w; z)$ is a diagonal matrix with entries from x. Thus one can observe that for any $x \in \{\pm 1\}^d$ we have that $\ell(\cdot; z)$ is G-Lipschitz and H-smooth over $\mathbb{R}^d$.

To prove a lower bound on τ_1 and τ_2, it suffices to show that for any (ϵ, δ)-DP implementation of $\mathcal{O}$ there exists $w \in \mathbb{R}^d$ such that $\mathop{\mathbb{E}}_{\mathcal{O}} \left[\|\mathcal{O}(w; \perp) - \nabla \mathcal{L}(w)\|^2 \right] \geq \tau_1^2$ and there exist $w, w' \in \mathbb{R}^d$ such that $\mathop{\mathbb{E}}_{\mathcal{O}} \left[\|\mathcal{O}(w, w')\|^2 \right] \geq \tau_2^2 \|w - w'\|^2$. For sake of generality, we will show that these properties hold for a set of w, w'.

Note that to lower bound the gradient error, it suffices to lower bound the error with respect to the first d' components. We thus argue using ℓ_1, and will in fact show a lower bound for any $w \in \mathbb{R}^d$. Let $w \in \mathbb{R}^d$. We have for any (ϵ, δ)-DP oracle $\mathcal{O}$ there exists a dataset $S \subseteq \{\pm 1\}^d$, where $|S| = n$, of fingerprinting codes such that

$$\mathop{\mathbb{E}}_{\mathcal{O}} \left[\|\mathcal{O}(w; \perp) - \nabla \mathcal{L}(w)\| \right] \geq \mathop{\mathbb{E}}_{\mathcal{O}} \left[\left\| \mathcal{O}(w; \perp)^{(1)} - \frac{1}{n} \sum_{x \in S} z^{(1)} \right\| \right] = \Omega \left(\frac{G\sqrt{d \log(1/\delta)}}{n\epsilon} \right).$$

The bound follows from standard fingerprinting code arguments. See [BST14, Lemma 5.1] for a lower bound and [SU15, Theorem 1.1] for a group privacy reduction that obtains the additional $\sqrt{\log(1/\delta)}$ factor. This fingerprinting result also induces the parameter constraints in the theorem statement. We thus have $\tau_1 = \Omega \left(\frac{G\sqrt{d \log(1/\delta)}}{n\epsilon} \right)$.

Similarly, we will argue a bound on the gradient variation using ℓ_2. Let $w, w' \in \mathcal{W}$ and $u = (w - w')^{(2)}$. In what follows, we only use the second half of the components for each vector, and thus omit the superscript $^{(2)}$ from all vectors for readability. We have $\nabla \ell_2(w; z) - \nabla \ell_2(w'; z) = H[u_1 z_1, ..., u_{d'} z_{d'}]^\top$. Then for any $c \in (0, \frac{2G}{H\sqrt{d}}]$ and

$u \in \{\pm c\}^2$ we have

$$\begin{aligned}
&\mathop{\mathbb{E}}_{\mathcal{O}}\left[\|\mathcal{O}(w,w') - (\nabla\mathcal{L}(w) - \nabla\mathcal{L}(w'))\|^2\right] \\
&= H^2 \cdot \mathop{\mathbb{E}}_{\mathcal{O}}\left[\sum_{j=1}^{d'}\left(\mathcal{O}(w,w')_j - \frac{u_j}{n}\sum_{x\in S} z_j\right)^2\right] \\
&= H^2 \cdot \mathop{\mathbb{E}}_{\mathcal{O}}\left[\sum_{j=1}^{d'}\left(u_j\left(\frac{\mathcal{O}(w,w')_j}{u_j} - \frac{1}{n}\sum_{x\in S} z_j\right)\right)^2\right] \\
&= H^2 \cdot \mathop{\mathbb{E}}_{\mathcal{O}}\left[c^2\sum_{j=1}^{d'}\left(\frac{\mathcal{O}(w,w')_j}{u_j} - \frac{1}{n}\sum_{x\in S} z_j\right)^2\right] \\
&= \Omega\left(H^2c^2\frac{d^2\log(1/\delta)}{n^2\epsilon^2}\right),
\end{aligned}$$

where the last step again comes from fingerprinting results. Note that the extra factor of d as compared to the previous bound comes from the fact that we are considering fingerprinting codes with norm larger by a factor of $\sqrt{d}$. We also use the fact that the vector $\mathcal{O}(w,w')$ transformed using u is (ϵ,δ)-DP by post processing. Now since $c = \frac{\|w-w'\|}{\sqrt{d}}$ we have

$$\mathop{\mathbb{E}}_{\mathcal{O}}\left[\|\mathcal{O}(w,w') - (\nabla\mathcal{L}(w) - \nabla\mathcal{L}(w'))\|\right] = \left(H\|w-w'\|\frac{\sqrt{d\log(1/\delta)}}{n\epsilon}\right).$$

Finally, noting that $\mathop{\mathbb{E}}_{\mathcal{O}}\left[\|\mathcal{O}(w,w') - (\nabla\mathcal{L}(w) - \nabla\mathcal{L}(w'))\|^2\right] \leq \mathop{\mathbb{E}}_{\mathcal{O}}\left[\|\mathcal{O}(w,w')\|^2\right]$ we obtain $\tau_2 = \Omega\left(\frac{H\sqrt{d\log(1/\delta)}}{n\epsilon}\right)$. This completes the proof. □

We remark that the accuracy lower bound for the gradient variation can hold for a much more general set of vectors than that given in the proof. Specifically, the same result can be obtained for any $u = w - w'$ such that u has $\Theta(d)$ components which are $\Omega\left(\frac{\|u\|}{\sqrt{d}}\right)$ (i.e. any sufficiently spread out vector). This uses the fact that it suffices to bound the number of components which disagree in sign with the fingerprinting mean and that fingerprinting codes are sampled using a product distribution, and thus the tracing attack used by fingerprinting constructions holds over any sufficiently large subset of dimensions.

B.3 Missing Results for Population Stationary Points

Here we present the proof of privacy and accuracy for Algorithm 6. We start by proving the privacy guarantee.

Proof of Theorem 18. By parallel composition of differential privacy, and since the used batches are disjoint, it suffices to prove that each step in lines 6 and 15 of the algorithm is (ϵ, δ)-DP. Note that the gradient estimator in step 6 has ℓ_2-sensitivity $2G/b$, so by the Gaussian mechanism this step is (ϵ, δ)-DP.

For step 15, suppose $S_{t,s}$ and $S'_{t,s}$ are neighboring datasets that differ in at most one element: $z_{i^*} \neq z'_{i^*}$, and let η_{t,s_i} and η'_{t,s_i} the respective step-sizes used in step 23. Then

$$\|\Delta_{t,s} - \Delta'_{t,s}\| = \frac{2^{|s|}}{b}\|\nabla \ell\left(w_{t,s}; z_{i^*}\right) - \nabla \ell\left(w_{t,\widehat{s}}; z_{i^*}\right) - \left(\nabla \ell\left(w_{t,s}; z'_{i*}\right) - \nabla \ell\left(w_{t,\widehat{s}}; z'_{i^*}\right)\right)\|,$$

and note between the parent node $u_{t,\widehat{s}}$ and $u_{t,s}$ there are $2^{D-|s|}$ iterates generated by the algorithm, which we denote as $w_{t,\widehat{s}} = w_{t,s_0}, w_{t,s_1}, ..., w_{t,s_{2^{|D|-s}}} = w_{t,s}$. Then, by smoothness of f and the triangle inequality

$$\begin{aligned}
&\|\Delta_{t,s} - \Delta'_{t,s}\| \\
&= \frac{2^{|s|}}{b}\|\nabla \ell\left(w_{t,s}; z_{i^*}\right) - \nabla \ell\left(w_{t,\widehat{s}}; z_{i^*}\right) - \left(\nabla \ell\left(w_{t,s}; z'_{i*}\right) - \nabla \ell\left(w_{t,\widehat{s}}; z'_{i^*}\right)\right)\| \\
&\leq \sum_{i=1}^{2^{D-|s|}} \frac{2^{|s|}}{b}\left[\|\nabla \ell\left(w_{t,s_i}; z_{i^*}\right) - \nabla \ell\left(w_{t,s_{i-1}}; z_{i^*}\right)\| + \|\left(\nabla \ell\left(w_{t,s_i}; z'_{i*}\right) - \nabla \ell\left(w_{t,s_{i-1}}; z'_{i^*}\right)\right)\|\right] \\
&\leq \sum_{i=1}^{2^{D-|s|}} \frac{2^{|s|}}{b} H\eta_{t,s_{i-1}}\|\nabla_{t,s_{i-1}}\| + \sum_{i=1}^{2^{D-|s|}} \frac{2^{|s|}}{b} H\eta'_{t,s_{i-1}}\|\nabla'_{t,s_{i-1}}\| \\
&= 2\sum_{i=1}^{2^{D-|s|}} \frac{2^{|s|}}{b}\frac{\beta}{2^{D/2}} = \frac{2\beta 2^{D/2}}{b}.
\end{aligned}$$

The Gaussian mechanism combined with our choice of $\sigma_{t,s}$ certifies privacy of this step. □

To prove Theorem 19 we will need some technical lemmas. Define $(\mathcal{T}, \mathcal{S})$ as a random stopping time that indicates when Algorithm 6 ends. Also, we say $(t_1, s_1) \preceq_2 (t_2, s_2)$ whenever w_{t_1,s_1} comes before w_{t_2,s_2} in the algorithm iterates.

Lemma 17 (Gradient estimation error, extension of Lemma 6 in [FLLZ18])**.** *Let $p \in (0,1)$. Then, with probability $1-p$ the event*

$$\mathcal{E} = \{\|\nabla_{t,s} - \nabla L(w_{t,s}; \mathcal{D})\|^2 \leq \alpha \cdot \widetilde{\alpha} \quad \forall (t,s) \preceq_2 (\mathcal{T}, \mathcal{S})\}$$

holds, under the parameter setting of $\sigma_{t,\varnothing}, \sigma_{t,s}$ and $\eta_{t,s}$ in Algorithm 6, for $\alpha^2 \geq \left(\frac{G^2}{b} + \frac{\beta^2 D 2^D}{b}\right) \max\left\{1, \frac{(d+1)}{b\epsilon^2}\right\}$ and $\widetilde{\alpha} \geq 256 \log\left(\frac{1.25}{\delta}\right) \log\left(\frac{2T2^{D+1}}{p}\right) \alpha$.

Proof. Recall the gradient estimate associated to a left child node is the same as that of the parent node. Hence, the gradient estimate of a non-leaf node is the same as that of the left-most leaf of its left sub-tree. In addition, we only need to control the gradient estimation error when we perform a gradient step, which occurs at the leaves. Then, to prove the claim, it suffices to prove that we can control the gradient estimation error at the leaves. Since, the number of iterations (and leaves) is at most $T2^{D-1}$, to prove event $\mathcal{E}$ happens with probability $1-p$, by the union bound it suffices to prove that $\mathbb{P}[\|\nabla_{t,s} - \nabla L(w_{t,s}; \mathcal{D})\|^2 > \alpha \cdot \widetilde{\alpha}] \leq \frac{p}{T2^{D-1}}$ for every $(t,s) \preceq_2 (\mathcal{T}, \mathcal{S})$ where $u_{t,s}$ is a leaf.

Denote by $\mathcal{F}_t$ the sigma algebra generated by randomness in the algorithm until the end of round t. Fix $(t,s) \preceq_2 (\mathcal{T}, \mathcal{S})$ such that $u_{t,s}$ is leaf, and let $u_{t,s_\varnothing} = u_{t,s_0}, u_{t,s_1}, ..., u_{t,s_k} = u_{t,s}$ be the path from the root to s. Next, extract a sub-sequence of it including only the root and the nodes that are right children, obtaining $u_{t,s_\varnothing} = u_{t,s_{a_0}}, u_{t,s_{a_1}}, ..., u_{t,s_{a_m}} = u_{t,s}$. Now we can write

$$\nabla_{t,s} - \nabla L(w_{t,s}; \mathcal{D}) = \sum_{i=0}^{m} g_{t,s_{a_i}} + \sum_{z \in S_{t,\varnothing}} \underbrace{\frac{1}{b}\left(\nabla \ell(w_{t,\varnothing}; z) - \nabla L(w_{t,\varnothing}; \mathcal{D})\right)}_{\gamma_{1,z}}$$

$$+ \sum_{i=1}^{m} \sum_{z \in S_{t,s_{a_i}}} \underbrace{\frac{2^{|s_{a_i}|}}{b}\left[\left(\nabla \ell(w_{t,s_{a_i}}; z) - \nabla \ell(w_{t,s_{a_{i-1}}}; z)\right) - \left(\nabla L(w_{t,s_{a_i}}; \mathcal{D}) - \nabla L(w_{t,s_{a_{i-1}}}; \mathcal{D})\right)\right]}_{\gamma_{2,z,i}}.$$

To bound the estimation error, we note that

$$\begin{aligned}&\mathbb{P}[\|\nabla_{t,s} - \nabla L(w_{t,s};\mathcal{D})\|^2 > \alpha\cdot\widetilde{\alpha}|\mathcal{F}_{t-1}]\\&\leq \mathbb{P}\Big[\Big\|\sum_{i=0}^{m} g_{t,s_{a_i}}\Big\|^2 > \frac{\alpha\cdot\widetilde{\alpha}}{4}\Big|\mathcal{F}_{t-1}\Big] + \mathbb{P}\Big[\Big\|\sum_{z\in S_{t,\varnothing}}\gamma_{1,z} + \sum_{i=1}^{m}\sum_{z\in S_{t,s_{a_i}}}\gamma_{2,z,i}\Big\|^2 > \frac{\alpha\cdot\widetilde{\alpha}}{4}\Big|\mathcal{F}_{t-1}\Big].\end{aligned}$$

and proceed to bound each term on the right hand side separately. By vector subgaussian concentration (see Lemma 1 in [JNG+19]) and noting that the Gaussians are independent of $\mathcal{F}_{t-1}$, we know that

$$\mathbb{P}\left[\left\|\sum_{i=0}^{m} g_{t,s_{a_i}}\right\|^2 > \frac{\alpha\cdot\widetilde{\alpha}}{4}\right] \leq 4^d \exp\left(-\frac{\alpha\cdot\widetilde{\alpha}}{32(\sigma_{t,\varnothing}^2 + \sum_{i=1}^{m}\sigma_{t,s_{a_i}}^2)}\right),$$

and in order to bound this probability by $\frac{p}{2T2^{D-1}}$, since $m \leq D$, it suffices that

$$\begin{aligned}\alpha\cdot\widetilde{\alpha} &> 32\log\left(\frac{4^d T 2^D}{p}\right)\left[\frac{8G^2\log(1.25/\delta)}{b^2\epsilon^2} + \frac{8D2^D\beta^2\log(1.25/\delta)}{b^2\epsilon^2}\right]\\&= 256\log\left(\frac{1.25}{\delta}\right)\left[d\log(4) + \log\left(\frac{T2^D}{p}\right)\right]\left[\frac{L_0^2}{b^2\epsilon^2} + \frac{D2^D\beta^2}{b^2\epsilon^2}\right].\end{aligned}$$

Now, noting that surely

$$\|\gamma_{1,x}\| \leq \frac{2G}{b} \quad \text{and} \quad \|\gamma_{2,x,i}\| \leq \frac{2\beta 2^{D/2}}{b},$$

where the second bound comes from following similar steps as in the privacy analysis in Theorem 18, we have that $\sum_{x\in S_{t,\varnothing}}\gamma_{1,x} + \sum_{i=1}^{m}\sum_{x\in S_{t,s_{a_i}}}\gamma_{2,x,i}$ is a sum of bounded martingale differences when conditioned on $\mathcal{F}_{t-1}$, thus by concentration of martingale-difference sequences in ℓ_2 (see Proposition 2 in [FLLZ18]), and using the fact that $|S_{t,\varnothing}| = b$ and $|S_{t,s_{a_i}}| = b/2^{|s_{a_i}|}$ it follows that

$$\mathbb{P}\left[\left\|\sum_{x\in S_{t,\varnothing}}\gamma_{1,x} + \sum_{i=1}^{m}\sum_{x\in S_{t,s_{a_i}}}\gamma_{2,x,i}\right\|^2 > \frac{\alpha\cdot\widetilde{\alpha}}{4} \mid \mathcal{F}_{t-1}\right] \leq 4\exp\left(-\frac{\alpha\cdot\widetilde{\alpha}}{16\left[\frac{4G^2}{b} + \sum_{i=1}^{m}\frac{4\beta^2 2^D}{2^{|s_{a_i}|}b}\right]}\right).$$

Repeating a similar argument as before, to bound this term by $\frac{p}{2T2^{D-1}}$, it suffices that

$$\alpha\cdot\widetilde{\alpha} \geq 64\log\left(\frac{2T2^{D+1}}{p}\right)\left[\frac{G^2}{b} + \frac{\beta^2 D2^D}{b}\right].$$

Finally, both conditions hold simultaneously for

$$\alpha^2 \geq \left(\frac{G^2}{b} + \frac{\beta^2 D 2^D}{b}\right) \max\left\{1, \frac{(d+1)}{b\epsilon^2}\right\}$$

and

$$\widetilde{\alpha} \geq 256 \log\left(\frac{1.25}{\delta}\right) \log\left(\frac{2T2^{D+1}}{p}\right) \alpha.$$

□

Lemma 18 (Descent lemma; Lemma 7 in [FLLZ18])**.** *Under the assumption that the event $\mathcal{E}$ from Lemma 17 occurs and $\beta \leq 2^{D/2}\widetilde{\alpha}$, we have that if Algorithm 6 reaches the last line, then*

$$L(w_{T,\ell(2^D)}; \mathcal{D}) - L(0; \mathcal{D}) \leq -(T2^{D-1}) \frac{\beta \cdot \widetilde{\alpha}}{4 \cdot 2^{D/2} H}.$$

where $w_{T,\ell(2^D)}$ is the last iterate in the T-th tree of Algorithm 6.

We provide the proof of Lemma 18 adapted to our case for completeness.

Proof. By standard analysis for smooth functions we have

$$L(w_{t,s^+}; \mathcal{D}) \leq L(w_{t,s}; \mathcal{D}) - \frac{\eta_{t,s}}{2}(1 - \eta_{t,s}H)\|\nabla_{t,s}\|^2 + \frac{\eta_{t,s}}{2}\|\nabla_{t,s} - \nabla L(w_{t,s}; \mathcal{D})\|^2,$$

where $\eta_{t,s} = \frac{\beta}{2^{D/2} H \|\nabla_{t,s}\|}$ and u_{t,s^+} is the node after $u_{t,s}$ in the tree. Since $\beta \leq 2^{D/2}\widetilde{\alpha}$ and $\|\nabla_{t,s}\| > 2\widetilde{\alpha}$, we have that $(1 - \eta_{t,s}H) \geq 1/2$. Using this inequality, the definition of $\eta_{t,s}$ and the fact that we are assuming $\mathcal{E}$ occurs, we obtain

$$\begin{aligned} L(w_{t,s^+}; \mathcal{D}) - L(w_{t,s}; \mathcal{D}) &\leq -\frac{\beta}{4 \cdot 2^{D/2} H \|\nabla_{t,s}\|}\|\nabla_{t,s}\|^2 + \frac{\beta}{2 \cdot 2^{D/2} H \|\nabla_{t,s}\|}\alpha \cdot \widetilde{\alpha} \\ &\leq -\frac{\beta}{4 \cdot 2^{D/2} H} \cdot \widetilde{\alpha}, \end{aligned}$$

where the second inequality comes from $\|\nabla_{t,s}\| > 2\widetilde{\alpha}$ and $\alpha \leq \widetilde{\alpha}$. Then telescoping over all $T2^{D-1}$ iterations provides the claimed bound. □

We are now ready to prove the convergence guarantee of Algorithm 6.

Proof of Theorem 19. From Lemma 17, we know that $\|\nabla_{t,s} - \nabla L(w_{t,s}; \mathcal{D})\|^2 \leq \alpha \cdot \widetilde{\alpha}$ with probability $1 - p$ when

$$\alpha = \sqrt{2}G \max\left\{\frac{1}{n^{1/3}}, \left(\frac{\sqrt{d}}{n\epsilon}\right)^{1/2}\right\},$$
$$\widetilde{\alpha} = \left(256 \log\left(\frac{1.25}{\delta}\right) \log\left(\frac{2T2^{D+1}}{p}\right) + \frac{8HL_0\sqrt{2D}(D/2+1)}{2G^2}\right)\alpha.$$

Indeed, using our parameter setting, and noting that $d > b\epsilon^2$ if and only if, $d > n^{2/3}\epsilon^2$, yields

$$\begin{aligned}
\alpha^2 &\geq \frac{G^2}{b}\max\left\{1, \frac{(d+1)}{b\epsilon^2}\right\} + \frac{\beta^2}{2}\max\left\{1, \frac{(d+1)}{b\epsilon^2}\right\} \\
&= G^2\left(\frac{1}{n^{2/3}}\mathbb{1}_{\{d+1\leq n^{2/3}\epsilon^2\}} + \frac{\sqrt{d}}{n\epsilon}\mathbb{1}_{\{d+1>n^{2/3}\epsilon^2\}}\right) \\
&+ \frac{\alpha^2}{2}\min\left\{1, \frac{b\epsilon^2}{d}\right\}\max\left\{1, \frac{(d+1)}{b\epsilon^2}\right\} \\
&\geq G^2\max\left\{\frac{1}{n^{2/3}}, \frac{\sqrt{d}}{n\epsilon}\right\} + \frac{\alpha^2}{2},
\end{aligned}$$

which shows our values of α and $\widetilde{\alpha}$ are valid for controlling the gradient estimation error with high probability, as claimed in Lemma 17.

Now, suppose for the sake of contradiction that Algorithm 6 does not end in line 20 under $\mathcal{E}$. This means it performs $T2^{D-1}$ gradient updates. We'll show this implies $(T2^{D-1})\frac{\beta\cdot\widetilde{\alpha}}{4\cdot 2^{D/2}H} > L_0$ and thus contradicts Lemma 18, which claims that $L_0 \geq -[L(w_{T,\ell(2^D)}; \mathcal{D}) - L(w_{0,\ell(2^D)}; \mathcal{D})] \geq (T2^{D-1})\frac{\beta\cdot\widetilde{\alpha}}{4\cdot 2^{D/2}H}$. Indeed, note that by our parameter setting:

$$\begin{aligned}
(T2^{D-1})\frac{\beta\cdot\widetilde{\alpha}}{4\cdot 2^{D/2}H} > L_0 &\iff \beta\cdot\widetilde{\alpha} > \frac{8HL_0}{T2^{D/2}} \\
&\iff \alpha\min\left\{1, \frac{\sqrt{b}\epsilon}{\sqrt{d}}\right\}\cdot\widetilde{\alpha} > \frac{8HL_0\sqrt{2D}}{T\sqrt{b}} \\
&\iff \alpha\cdot\widetilde{\alpha} > \frac{8HL_0\sqrt{2D}(D/2+1)\sqrt{b}}{n}\max\left\{1, \frac{\sqrt{d}}{\sqrt{b}\epsilon}\right\} \\
&\iff \alpha\cdot\widetilde{\alpha} > 8HL_0\sqrt{2D}(D/2+1)\max\left\{\frac{\sqrt{b}}{n}, \frac{\sqrt{d}}{n\epsilon}\right\},
\end{aligned}$$

and noting that by the setting of b we have $\max\left\{\frac{\sqrt{b}}{n}, \frac{\sqrt{d}}{n\epsilon}\right\} = \max\left\{\frac{1}{n^{2/3}}, \frac{\sqrt{d}}{n\epsilon}\right\}$, we conclude the following

$$(T2^{D-1})\frac{\beta\cdot\widetilde{\alpha}}{4\cdot 2^{D/2}H} > L_0 \iff \alpha\cdot\widetilde{\alpha} > 8HL_0\sqrt{2D}(D/2+1)\max\left\{\frac{1}{n^{2/3}}, \frac{\sqrt{d}}{n\epsilon}\right\}$$
$$\iff \alpha\cdot\widetilde{\alpha} > \frac{8HL_0\sqrt{2D}(D/2+1)}{2G^2}\alpha^2.$$

Finally, note $\alpha\cdot\widetilde{\alpha} = \left(256\log(1.25/\delta)\log\left(2T2^{D+1}/p\right) + \frac{8HL_0\sqrt{2D}(D/2+1)}{2G^2}\right)\alpha^2$ and thus the last inequality holds under our parameter setting. Since this is equivalent to $(T2^{D-1})\frac{\beta\cdot\widetilde{\alpha}}{4\cdot 2^{D/2}H} > L_0$, we are done with the contradiction. It follows that with high probability, Algorithm 6 ends in line 20 returning $w_{t,s}$ such that $\|\nabla_{t,s}\| \leq 2\widetilde{\alpha}$. Also, by Lemma 17 we have $\|\nabla F(w_{t,s};\mathcal{D}) - \nabla_{t,s}\| < \widetilde{\alpha}$, so the returned iterate satisfies by the triangle inequality

$$\|\nabla L(w_{t,s};\mathcal{D})\| < 3\widetilde{\alpha}.$$

In addition, the linear time oracle complexity follows from the fact that at each binary tree we use b samples at the root, and then $b/2$ in levels 1 to D. This gives a total of $b(D/2+1)$ samples used at every round. Since we run the algorithm for $T = \frac{n}{b(D/2+1)}$ rounds, we compute exactly n gradients. To conclude, note the condition $n \geq \max\{\sqrt{d}(D/2+1)^2/\epsilon, (D/2+1)^3\}$ implies the number of rounds T is at least 1. Besides, since the definition of D implies $2^D < b$, the size of the mini-batches are well-defined (meaning Algorithm 6 uses batches with at least 1 sample). This concludes the proof. □

B.4 Missing Results for Stationary Points in the Convex Setting

We first give pseudo-codes of algorithms used in the section.

Proof of Theorem 20. The privacy guarantee, in both cases, follows from the privacy guarantees of Algorithm 18 and Algorithm 16, in Lemmas 21 and 24 respectively,

Algorithm 16 Phased SGD$(S, (w,z) \mapsto f(w;z)), D, \eta, \mathcal{S}(\cdot), \sigma)$

Input: Dataset S, loss function $f(\cdot;z))$, radius D of the constraint set $\mathcal{W}$, steps T, η, Selection function $\mathcal{S}$, Noise variance σ

$w_1 = 0$

$K = \lceil \log(|S|) \rceil$ and $T_0 = 1$

for $k = 1$ to $K-1$ **do**

  $T_k = 2^{-k}|S|, \eta_k = 4^{-k}\eta, \sigma_k = \eta_k \sigma$

  $w_{k+1} = \mathsf{OutputPerturbedSGD}(w_k, S_{T_{k-1}+1:T_k}, D, \eta_k, \sigma_k, \mathcal{S}(\cdot))$

end for

Output: $\widehat{w} = w_K$

Algorithm 17 OutputPerturbedSGD$(w_1, S, (w,z) \mapsto \ell(w;z), \Delta(\cdot), D, \eta, \mathcal{S}(\cdot)$

Input: Dataset S, loss function $\ell(\cdot;z))$, regularizer $\Delta(\cdot)$, radius D of the constraint set $\mathcal{W}$, steps T, η, Selection function $\mathcal{S}$, Noise variance σ

for $t = 1$ to $|S|-1$ **do**

  $w_{t+1} = \Pi_{\mathcal{W}}\left(w_t - \eta\left(\nabla \ell(w_t; z_t)\right)\right)$

end for

$\xi \sim \mathcal{N}(0, \sigma^2 \mathbb{I})$

$\widetilde{w} = \mathcal{S}\left(\{w_t\}_{t=1}^{|S|}\right)$

Output: $\widehat{w} = \widetilde{w} + \xi$

together with parallel composition.

We now proceed to the utility part. For simplicity of notation, let $D := \|w^*\|$. Recall the definition of the regularized losses $\ell^{(t)}(w,x)$ in Algorithm 7. Let $\{\alpha_t\}_t$ be such that $\mathbb{E}[L^{(t-1)}(\widehat{w}_t; \mathcal{D})] - L^{(t-1)}(w^*_{t-1}; \mathcal{D}) \leq \alpha_t$ where $\widehat{w}_t$ are the iterates produced in the algorithm and $w^*_{t-1} = \arg\min_{w \in \mathbb{R}^d} L^{(t-1)}(w; \mathcal{D})$. Following [AZ18, FSS+19], we

Algorithm 18 Noisy GD$(S, (w,z) \mapsto \ell(w;z), D, T, \eta, \mathcal{S}(\cdot), \sigma)$

Input: Dataset S, loss function $(w,z) \mapsto \ell(w;z)$, radius D of the constraint set $\mathcal{W}$, steps T, η, Selection function $\mathcal{S}$, Noise variance σ

1: $w_1 = 0$
2: **for** $t = 1$ to $T-1$ **do**
3: $\quad \xi_t \sim \mathcal{N}(0, \sigma^2 \mathbb{I})$
4: $\quad w_{t+1} = \Pi_{\mathcal{W}}\left(w_t - \eta\left(\nabla \widehat{L}(w_t; S) + \xi_t\right)\right)$
5: **end for**

Output: $\widehat{w} = \mathcal{S}\left(\{w_t\}_{t=1}^{T}\right)$

first establish a general result which will be useful for both parts of the result.

$$
\begin{aligned}
&\mathbb{E}\left\|\nabla L(\widehat{w}_T; \mathcal{D})\right\| \\
&= \mathbb{E}\left\|\nabla L^{(T-1)}(\widehat{w}_T; \mathcal{D}) + \lambda \sum_{t=0}^{T} 2^t \left(\widehat{w}_t - \widehat{w}_T\right)\right\| \\
&\leq \mathbb{E}\left\|\nabla L^{(T-1)}(\widehat{w}_T; \mathcal{D})\right\| + \lambda \sum_{t=0}^{T-1} 2^t \mathbb{E}\left(\left\|\widehat{w}_t - w^*_{T-1}\right\| + \left\|\widehat{w}_T - w^*_{T-1}\right\|\right) \\
&\leq 2\mathbb{E}\left\|\nabla \widehat{L}^{(T-1)}(\widehat{w}_T; \mathcal{D})\right\| + \lambda \sum_{t=1}^{T-1} 2^t \mathbb{E}\left\|\widehat{w}_t - w^*_{T-1}\right\| + \lambda \mathbb{E}\left\|w_0 - w^*_{T-1}\right\| \\
&\leq 2\mathbb{E}\left\|\nabla L^{(T-1)}(\widehat{w}_T; \mathcal{D})\right\| + 4 \sum_{t=1}^{T-1} \sqrt{\lambda 2^t \alpha_t} + \lambda D_{T-1} \\
&\leq 4\sqrt{H\alpha_T} + 4 \sum_{t=1}^{T-1} \sqrt{\lambda 2^{t+1} \alpha_t} + \lambda 2^{T/2} D \\
&\leq 4 \sum_{t=1}^{T} \sqrt{\lambda 2^{t+1} \alpha_t} + \sqrt{\lambda H} D
\end{aligned}
$$

where the third and fourth inequality follows from strong convexity of $L^{(T-1)}(\cdot; \mathcal{D})$ and Lemma 20 respectively. The last inequality follows from the setting of T since we have that $L^{(T-1)}$ is $H + \sum_{t=1}^{T-1} 2^t \lambda \leq H + \lambda 2^T \leq 2H$ smooth. Note that the definition of D_t and Lemma 19, $\left\|w^*_{T-1}\right\| \leq D_{T-1}$, so the unconstrained minimizer lies in the constraint set. Therefore $\mathbb{E}\left\|\nabla L^{(T-1)}(\widehat{w}_T; \mathcal{D})\right\| = \mathbb{E}\left\|\nabla L^{(T-1)}(\widehat{w}_T; \mathcal{D}) - \nabla L^{(T-1)}(w^*_{T-1}; \mathcal{D})\right\| \leq 2\sqrt{H\alpha_T}$.

Observe that from the setting of T, $L^{(T)}$ is $4H$ smooth for all t. Furthermore, the radius of the constraint set in the t-th round is $D_t = 2^{T/2} D$. Hence, the Lipschitz constant $G_t \leq G + 8HD_t \leq O\left(G + H2^{T/2}\right)$. Now we instantiate α_t, which is the

excess population risk bound of the DP-SCO sub-routine.

Optimal rate. The excess population risk guarantee of Algorithm 18 is in Lemma 21, with (in context of the notation in the Lemma) Lipschitz parameter G being the same and $G_\Delta = O\left(H2^{T/2}\right)$. Therefore, we have $\alpha_t = \widetilde{O}\left(\frac{G^2}{\lambda_t n} + \frac{dG^2}{\lambda_t n^2\epsilon^2}\right)$. Plugging in the above estimate, we get,

$$\mathbb{E}\left\|\nabla L(\widehat{w};\mathcal{D})\right\| = \widetilde{O}\left(\frac{G}{\sqrt{n}} + \frac{\sqrt{d}G}{n\epsilon} + \sqrt{\frac{\lambda}{H}}D\right) = \widetilde{O}\left(\frac{G}{\sqrt{n}} + \frac{\sqrt{d}G}{n\epsilon}\right)$$

where the last step follows by setting of λ.

The optimality claim follows by combining the non-private lower bound in Theorem 20, and the DP empirical stationarity lower bound in Theorem 17 together with a reduction to population stationarity as in [BFTGT19, Appendix C].

Linear time rate. The excess population risk guarantee of Algorithm 16 is in Lemma 24, with Lipschitz parameter G being the same and $G_\Delta = O\left(H2^{T/2}\right)$. This gives us $\alpha_t = \widetilde{O}\left(\frac{G^2}{\lambda_t n} + \frac{dG^2}{\lambda_t n^2\epsilon^2}\right)$, and thus

$$\mathbb{E}\left\|\nabla L(\widehat{w};\mathcal{D})\right\| = \widetilde{O}\left(\frac{G}{\sqrt{n}} + \frac{\sqrt{d}G}{n\epsilon} + \sqrt{\lambda H}D\right) = \widetilde{O}\left(\frac{G}{\sqrt{n}} + \frac{\sqrt{d}G}{n\epsilon} + \frac{HD}{\sqrt{n}}\right)$$

where the last step follows by setting of λ. Finally, note that the Lemma 24 requires that $n = \widetilde{\Omega}\left(\frac{H+\lambda_t}{\lambda_t}\right)$ for all t. This can be checked to be satisfied by substituting the value of λ_t. □

B.4.1 Utility Lemmas

We first present some key results which will be useful in the proofs.

Lemma 19. *Let $f : \mathbb{R}^d \to \mathbb{R}$ be an H-smooth convex function and let $w^* = \arg\min_{w\in\mathbb{R}^d} f(w)$. Let $\|w^*\| = D$ and $w_0 \in \mathbb{R}^d$ such that $\|w_0\| \le D$. Define $\widetilde{f}(w) = f(w) + \frac{\lambda}{2}\|w - w_0\|^2$ and let $\widetilde{w} = \arg\min \widetilde{f}(w)$. Then for any $\lambda \ge 0$, $\|\widetilde{w}\| \le \sqrt{2}D$.*

Proof. From optimality criterion, $0 = \nabla \widetilde{f}(\widetilde{w}) = \nabla f(\widetilde{w}) + \lambda(\widetilde{w} - w_0)$. Therefore, $\nabla f(\widetilde{w}) = \lambda(w_0 - \widetilde{w})$ and thus $\langle \nabla f(\widetilde{w}), w_0 - \widetilde{w} \rangle > 0$. Furthermore, since f is convex, from monotonicity, $\langle \nabla f(\widetilde{w}), w^* - \widetilde{w} \rangle \leq 0$. Since both w_0 and w^* lie in the ball of radius D (say $\mathcal{W}_D$), the above two implies that the hyperplane $H = \{w : \langle \nabla f(\widetilde{w}), w - \widetilde{w} \rangle = 0\}$ intersects with $\mathcal{W}_D$. Furthermore, since $\nabla f(\widetilde{w}) = \lambda(w_0 - \widetilde{w})$, we have that $\widetilde{w}$ is the projection of w_0 on H i.e. $\Pi_H(w_0)$.

Let $w' = \Pi_H(0)$. We have that $w' \in \mathcal{W}_D$; this is because the hyperplane cuts the hypersphere $\mathcal{W}_D$ creating a spherical cap and w' is the center of the cap. From properties of convex projections $\|\Pi_H(w_0) - \Pi_H(0)\| \leq \|w_0 - 0\| \leq D$. Furthermore, $\Pi_H(0)$ and $\Pi_H(w_0) - \Pi_H(0)$ are orthogonal. Hence $\|\widetilde{w}\|^2 = \|\Pi_H(w_0)\|^2 = \|\Pi_H(0)\|^2 + \|\Pi_H(w_0) - \Pi_H(0)\|^2 \leq 2D^2$. □

We state the following result from [AZ18, FSS+19].

Lemma 20. *Suppose for every $t = 1, 2, \ldots T$, $\mathbb{E}[L^{(t-1)}(\widehat{w}_t; \mathcal{D})] - L^{(t-1)}(w^*_{t-1}; \mathcal{D}) \leq \alpha_t$ where $\widehat{w}_t$ are the iterates produced in the algorithm, $w^*_{t-1} = \arg\min_{w \in \mathbb{R}^d} L^{(t-1)}(w; \mathcal{D})$ and $\lambda_t = 2^t \lambda$, we have,*

1. *For every $t \geq 1$, $\mathbb{E}[\left\|\widehat{w}_t - w^*_{t-1}\right\|^2] \leq \frac{2\alpha_t}{\lambda_{t-1}}$*
2. *For every $t \geq 1$, $\mathbb{E}[\|\widehat{w}_t - w^*_t\|^2] \leq \frac{\alpha_t}{\lambda_t}$*
3. *$\mathbb{E}[\sum_{t=1}^T \lambda_t \|\widehat{w}_t - w^*_T\|] \leq 4 \sum_{t=1}^T \sqrt{\alpha_t \lambda_t}$*

B.4.2 Lemmas for NoisyGD (Algorithm 18)

Lemma 21. *Consider a function $f(w; z) = \ell(w; z) + \Delta(w)$, where $w \mapsto \ell(w; z)$ is convex and G Lipschitz for all x, and $\Delta(w)$ is λ strongly convex, G_Δ Lipschitz and H_Δ smooth over a bounded convex set $\mathcal{W}$. Algorithm 17 run with parameters $\eta = \frac{\log(T)}{\lambda T}$, $\sigma^2 = \frac{64 G^2 T \log(1/\delta)}{n^2 \epsilon^2}$, $T = \max\left(\frac{H + H_\Delta}{\lambda} \log\left(\frac{H + H_\Delta}{\lambda}\right), \frac{n^2 \epsilon^2 \left(G^2 + G_\Delta^2\right)}{d G^2 \log(1/\delta)}\right)$ and $\mathcal{S}(\{w_t\}_t) =$*

$\frac{1}{\sum_{t=1}^{T}(1-\eta\lambda)^{-t}} \sum_{t=1}^{T} (1-\eta\lambda)^{-t} w_t$ *satisfies* (ϵ, δ)*-DP and given a dataset* S *of* n *i.i.d. points from* $\mathcal{D}$*, the excess population risk of its output* $\widehat{w}$ *is bounded by,*

$$\mathbb{E}\left[F(\widehat{w}; \mathcal{D}) - \min_{w \in \mathcal{W}_D} F(w; \mathcal{D})\right] = O\left(\frac{G^2}{\lambda n} + \frac{dG^2 \log(1/\delta)}{\lambda n^2 \epsilon^2}\right).$$

Proof. For the privacy analysis, as in [BST14], for fixed w, the sensitivity of the gradient update is bounded by $\frac{2G}{n}$. Applying advanced composition, we have that $\sigma^2 = \frac{64G^2T\log(1/\delta)}{n^2\epsilon^2}$ suffices for (ϵ, δ)-DP.

For utility, we first compute a bound on uniform argument stability of the algorithm; let $\{w_t\}$ and $\{w'_t\}$ be sequence of iterates on neighbouring datasets. Note that the function $w \mapsto f(w; z)$ is $H + H_\Delta$-smooth and λ-strongly convex for all z. From the setting of T, we have that the step size $\eta \leq \frac{1}{H+H_\Delta}$, hence from the standard stability analysis,

$$\begin{aligned}
w_{t+1} - w'_{t+1} &= w_t - \eta \nabla \widehat{L}(w_t; S) - \eta \nabla \Delta(w_t) - w'_t + \eta \nabla \widehat{L}(w'_t; S') + \eta \nabla \Delta(w'_t) \\
&= w_t - w'_t - \eta \left(\nabla \widehat{L}(w_t; S) + \nabla \Delta(w_t) - \nabla \widehat{L}(w'_t; S) - \eta \nabla \Delta(w'_t)\right) \\
&\quad + \eta \left(\nabla \widehat{L}(w'_t; S') - \nabla \widehat{L}(w'_t; S)\right) \\
&= \left(\mathbb{I} - \eta \left(\nabla^2 \widehat{L}(\widetilde{w}_t; S) + \nabla^2 \Delta(\widetilde{w}_t)\right)\right)(w_t - w'_t) \\
&\quad + \eta \left(\nabla \widehat{L}(w'_t; S') - \nabla \widehat{L}(w'_t; S)\right)
\end{aligned}$$

where the last equality follows from Taylor remainder theorem where $\widetilde{w}_t$ is some intermediate point on the line joining w_t and w'_t. Using the fact that $\eta \leq \frac{1}{H+H_\Delta}$, we have

$$\left\|w_{t+1} - w'_{t+1}\right\| \leq (1 - \eta\lambda) \left\|w_t - w'_t\right\| + \frac{2\eta G}{n} \leq \frac{2G}{\lambda n}$$

The above gives the same bound for the iterate using the selector $\mathcal{S}$,

$$\|\mathcal{S}(\{w_t\}) - \mathcal{S}(\{w'_t\})\| \leq \frac{2G}{\lambda n}$$

Note that the overall Lipschitz constant for the empirical loss is $\widetilde{G} = G + G_\Delta$. For the

excess empirical risk guarantee, we use Lemma 5.2 in [FKT20b] to get,

$$\begin{aligned}\mathbb{E}\left[\widehat{L}\left(\widehat{w};S\right)+\Delta(\widehat{w})-\widehat{L}(w^*;S)-\Delta(w^*)\right] &= \mathbb{E}\left[\widehat{F}\left(\widehat{w};S\right)-\widehat{F}(w^*;S)\right] \\ &= \widetilde{O}\left(\frac{\widetilde{G}^2}{\lambda T}\right) \\ &= \widetilde{O}\left(\frac{\widetilde{G}^2+\sigma^2 d}{\lambda T}\right) \\ &= \widetilde{O}\left(\frac{\widetilde{G}^2}{\lambda T}+\frac{dG^2\log\left(1/\delta\right)}{\lambda n^2\epsilon^2}\right) \\ &= O\left(\frac{dG^2\log\left(1/\delta\right)}{\lambda n^2\epsilon^2}\right)\end{aligned}$$

where the last step follows from the setting of T. For the population risk guarantee, we have,

$$\begin{aligned}\mathbb{E}\left[F(\widehat{w};\mathcal{D})-F(w^*;\mathcal{D})\right] &= \mathbb{E}\left[F(\widehat{w};\mathcal{D})-\widehat{F}(\widehat{w};S)\right]+\mathbb{E}\left[F(\widehat{w};\mathcal{D})-F(w^*;\mathcal{D})\right] \\ &= \mathbb{E}[F(\widehat{w};\mathcal{D})-\widehat{F}(\widehat{w};S)]+O\left(\frac{dG^2\log\left(1/\delta\right)}{\lambda n^2\epsilon^2}\right) \\ &\leq G\mathbb{E}\left\|\widehat{w}-\widehat{w}'\right\|+O\left(\frac{dG^2\log\left(1/\delta\right)}{\lambda n^2\epsilon^2}\right) \\ &= \widetilde{O}\left(\frac{G^2}{\lambda n}+\frac{dG^2\log\left(1/\delta\right)}{\lambda n^2\epsilon^2}\right)\end{aligned}$$

where the inequality follows from Lipschitzness and standard generalization gap to stability argument. □

B.4.3 Lemmas for PhasedSGD (Algorithm 16)

The following lemma gives population risk guarantees for strongly convex functions under privacy, in terms of variance of stochastic gradients, as opposed to standard Lipschitzness bounds.

Lemma 22 (Variance based bound for constant step-size SGD for strongly-convex functions)**.** *Consider a function $\ell(w;z)$ such that $w\mapsto\ell(w;z)$ is λ strongly convex, H smooth over a convex set $\mathcal{W}$ for all x and let $\mathbb{E}_x\left\|\nabla\ell(w;z)-\mathbb{E}_z\nabla\ell(w;z)\right\|^2\leq\mathcal{V}^2$ for*

all $w \in \mathcal{W}$. *Let* $\gamma_t = (1-\eta\lambda)^{-t}$. *Given a dataset* $S = \{z_1, z_2, \ldots, z_n\}$ *sampled i.i.d from* $\mathcal{D}$ *and* $\eta \leq \frac{1}{2\beta}$ *as input, for any* $w \in \mathcal{W}$, *the iterates of Algorithm 17 satisfy*

$$\mathbb{E}\left[\frac{1}{\sum_{t=1}^{n}\gamma_t}\sum_{t=1}^{n}\gamma_t L(w_t;\mathcal{D})\right] - L(w;\mathcal{D}) \leq \frac{\lambda}{e^{\eta\lambda n}-1}\left\|w_0 - w\right\|^2 + \eta\mathcal{V}^2$$

Furthermore, for $n = \Omega\left(\frac{H}{\lambda}\log\left(\frac{H}{\lambda}\right)\right)$, *with* $\eta = \frac{\log(n)}{\lambda n}$ *and* $\mathcal{S}(\{w_t\}_t) = \frac{1}{\sum_{t=1}^{n}\gamma_t}\sum_{t=1}^{n}\gamma_t w_t$, *the excess population risk of* $\widetilde{w} = \mathcal{S}(\{w_t\}_t)$ *satisfies*

$$\mathbb{E}\left[L(\widetilde{w};\mathcal{D}) - \min_{w\in\mathcal{W}} L(w;\mathcal{D})\right] = O\left(\frac{\mathcal{V}^2\log(n)}{\lambda n}\right)$$

Proof. An equivalent way to write the update in Algorithm 17 is

$$w_{t+1} = \arg\min_{w\in\mathcal{W}}\left(\langle\nabla\ell(w_t;z_t), w\rangle + \frac{1}{\eta}\left\|w_t - w\right\|^2 + \psi(w)\right)$$

where $\psi(w) = 0$ if $w \in \mathcal{W}$, otherwise ∞.

Following standard arguments in convex optimization, for any $w \in \mathcal{W}$, we have

$$
\begin{aligned}
&L(w_{t+1};\mathcal{D}) - L(w;\mathcal{D}) \\
&= L(w_{t+1};\mathcal{D}) + \psi(w_{t+1}) - L(w;\mathcal{D}) - \psi(w) \\
&\leq L(w_t) + \langle \nabla L(w_t), w_{t+1} - w_t \rangle + \frac{H}{2}\left\|w_{t+1} - w_t\right\|^2 + \psi(w_{t+1}) \\
&+ L(w;\mathcal{D}) - \psi(w) \\
&\leq \langle \nabla L(w_t;\mathcal{D}), w_{t+1} - w_t \rangle + \langle \nabla L(w_t;\mathcal{D}), w_t - w \rangle - \frac{\lambda}{2}\left\|w_t - w\right\|^2 + \frac{H}{2}\left\|w_{t+1} - w_t\right\|^2 \\
&+ \psi(w_{t+1}) + L(w;\mathcal{D}) - \psi(w) \\
&= \mathbb{E}_{z_t}\left[\langle \nabla \ell(w_t;z_t) - \nabla L(w;\mathcal{D}), w_t - w_{t+1} \rangle + \frac{H}{2}\left\|w_{t+1} - w_t\right\|^2 + \langle \nabla \ell(w_t;z_t), w_t - w \rangle\right] \\
&- \frac{\lambda}{2}\left\|w_t - w\right\|^2 + \psi(w_{t+1}) + L(w;\mathcal{D}) - \psi(w) \\
&\leq \mathbb{E}_{z_t}\Bigg[\langle \nabla \ell(w_t;z_t) - \nabla L(w;\mathcal{D}), w_t - w_{t+1} \rangle - \left(\frac{1}{2\eta} - \frac{H}{2}\right)\left\|w_{t+1} - w_t\right\|^2 \\
&+ \left(\frac{1}{2\eta} - \frac{\lambda}{2}\right)\left\|w_t - w\right\|^2 - \frac{1}{2\eta}\left\|w_{t+1} - w\right\|^2\Bigg] \\
&\leq \mathbb{E}_{z_t}\left[\frac{\eta}{2\left(1 - \eta H\right)}\left\|\nabla \ell(w_t;z_t) - \nabla L(w;\mathcal{D})\right\|^2 + \left(\frac{1}{2\eta} - \frac{\lambda}{2}\right)\left\|w_t - w\right\|^2 - \frac{1}{2\eta}\left\|w_{t+1} - w\right\|^2\right] \\
&\leq \eta \mathcal{V}^2 + \mathbb{E}_{z_t}\left[\left(\frac{1}{2\eta} - \frac{\lambda}{2}\right)\left\|w_t - w\right\|^2 - \frac{1}{2\eta}\left\|w_{t+1} - w\right\|^2\right]
\end{aligned}
$$

where the first inequality follows from smoothness, the second from strong convexity, the third from Fact D.1 in [AZ18], fourth from AM-GM inequality and the last from the assumption about variance bound on the oracle.

Now, the above is exactly the bound obtained in the proof of Lemma 5.2 in [FKT20b] with the second moment on gradient norm replaced by variance. Repeating the rest of the arguments in that Lemma gives us the claimed result. □

Lemma 23 (Privacy of Algorithm 17)**.** *Consider a function $f(w;z) = \ell(w;z) + \Delta(w)$ such that $w \mapsto \ell(w;z)$ is convex, G Lipschitz, H-smooth for all z, and $\Delta(\cdot)$ is λ strongly convex, G_Δ Lipschitz and H_Δ smooth over a bounded set $\mathcal{W}$. For $n = \Omega\left(\frac{H+H_\Delta}{\lambda}\log\left(\frac{H+H_\Delta}{\lambda}\right)\right)$, Algorithm 17 with input as function $(w,x) \mapsto f(w;z)$, $\sigma^2 = \frac{64G^2(\log(n))^2\log(1/\delta)}{\lambda^2 n^2 \epsilon^2}$, $\eta = \frac{\log(n)}{\lambda n}$ and $\mathcal{S}\left(\{w_t\}_{t=1}^n\right) = \frac{1}{\sum_{t=1}^n \gamma_t}\sum_{t=1}^n \gamma_t w_t$ for any weights*

γ_t *satisfies* (ϵ, δ)*-DP.*

Proof. We start with computing the sensitivity of the algorithm's output: let $\{w_t\}$ and $\{w'_t\}$ be sequence of iterates produced by Algorithm 17 on neighbouring datasets. Note that the function $w \mapsto f(w; z)$ is $H' = H + H_\Delta$-smooth and λ-strongly convex for all z. From the assumption on n, we have that the step size $\eta \leq \frac{1}{H+H_\Delta}$. Suppose the differing sample between neighbouring datasets is z_j, then $w_t = w'_t$ for all $t \leq j$. Also,

$$\left\|w_{j+1} - w'_{j+1}\right\| = \eta \left\|\nabla \ell(w_j; z_j) - \nabla \ell(w_j; z'_j)\right\| \leq 2\eta G = \frac{2G \log(n)}{\lambda n}$$

Now, for any $t > j$, as in the standard stability analysis we have,

$$\begin{aligned} w_{t+1} - w'_{t+1} &= w_t - \eta \nabla \ell(w_t; z_t) - \eta \nabla \Delta(w_t) - w_t + \eta \nabla \ell(w'_t; z_t) + \eta \nabla \Delta(w'_t) \\ &= \left(\mathbb{I} - \eta \left(\nabla^2 \ell(\widetilde{w}_t; z_t) + \nabla^2 \Delta(\widetilde{w}_t)\right)\right)(w_t - w'_t) \end{aligned}$$

where the last equality follows from Taylor's remainder theorem where $\widetilde{w}_t$ is some intermediate point in the line joining w_t and w'_t. Using the fact that $\eta \leq \frac{1}{H+H_\Delta}$ and λ strong convexity, we have

$$\left\|w_{t+1} - w'_{t+1}\right\| \leq (1 - \eta\lambda) \left\|w_t - w'_t\right\| \leq \left\|w_{j+1} - w'_{j+1}\right\| \leq \frac{2G \log(n)}{\lambda n}$$

Applying convexity to the weights in the definition of the selector function $\mathcal{S}$, we get,

$$\left\|\mathcal{S}(\{w_t\}) - \mathcal{S}(\{w'_t\})\right\| \leq \frac{2G \log(n)}{\lambda n}$$

The privacy proof now follows from the Gaussian mechanism guarantee. □

Lemma 24 (Phased SGD composite guarantee)**.** *Consider a function* $f(w; z) = \ell(w; z) + \Delta(w)$ *where* $w \mapsto \ell(w; z)$ *is convex, G-Lipschitz, H-smooth for all* z*, and* $\Delta(w)$ *is* λ *strongly convex,* G_Δ *Lipschitz and* H_Δ *smooth over a bounded set* $\mathcal{W}$*. For* $n = \Omega\left(\frac{K(H+H_\Delta)}{\lambda} \log\left(\frac{H+H_\Delta}{\lambda}\right)\right)$*, Algorithm 17 with* $\sigma^2 = \frac{64G^2K^2(\log(n))^2 \log(1/\delta)}{\lambda^2 n^2 \epsilon^2}$*, satisfies* (ϵ, δ)*-DP. Furthermore, with input as function* $(w, x) \mapsto f(w; z)$*, a dataset* S *of* n *samples*

drawn i.i.d. from $\mathcal{D}$, $\eta = \frac{\log(n)}{\lambda n}$, $K = \ln\ln n$, $\gamma_t = (1-\eta\lambda)^{-t}$ *and* $\mathcal{S}\left(\{w_t\}_{t=1}^n\right) = \frac{1}{\sum_{t=1}^n \gamma_t}\sum_{t=1}^n \gamma_t w_t$, *the excess population risk of output* w_K *is bounded as*

$$\mathbb{E}\left[F(w_K;\mathcal{D})\right] - \min_{w\in\mathcal{W}} F(w;\mathcal{D}) = \widetilde{O}\left(\frac{G^2}{\lambda n} + \frac{dG^2}{\lambda n^2\epsilon^2}\right)$$

Proof. The privacy proof simply follows from parallel composition. For the utility proof, we repeat the arguments in Theorem 5.3 in [FKT20b] substituting the variance-based bound from Lemma 22. Note that the variance of the stochastic gradients used, $\mathcal{V}^2 \leq G^2$, this gives us,

$$\mathbb{E}\left[F(w_K;\mathcal{D}) - \min_{w\in\mathcal{W}} F(w;\mathcal{D})\right] = \widetilde{O}\left(\frac{G^2}{\lambda n} + \frac{dG^2}{\lambda n^2\epsilon^2}\right)$$

□

B.5 Missing Results for Generalized Linear Models

We first give the definition of an oblivious subspace embedding.

Definition 20 ((r,τ,β)-oblivious subspace embedding)**.** *A random matrix* $\Phi \in \mathbb{R}^{k\times d}$ *is an* (r,τ,β)*-oblivious subspace embedding if for any* r *dimensional linear subspace in* $\mathbb{R}^d$*, say* V*, we have that with probability at least* $1-\beta$*, for all* $x \in V$*,*

$$(1-\tau)\|x\|^2 \leq \|\Phi x\|^2 \leq (1+\tau)\|x\|^2$$

It is well-known that Johnson-Lindenstrauss (JL) matrices with embedding dimension $k = O\left(\frac{r\log(2/\beta)}{\tau^2}\right)$ are (r,τ,β)-oblivious subspace embeddings and can be constructed efficiently [Coh16]. A simple example is a scaled Gaussian random matrix, $\Phi = \frac{1}{\sqrt{k}}\mathbf{G}$ where entries of $\mathbf{G}$ are independent and distributed as $\mathcal{N}(0,1)$.

Proof of Theorem 21. We first prove privacy. Let $G(S)$ and $H(S)$ be the bounds on the Lipschitz and smoothness constants of the family of loss functions $\{w \mapsto f(w;\Phi x)\}_{x\in S}$. With $k = \Omega(\log(2n/\delta))$, from the JL-property, it follows that with probability at

least $1-\delta/2$, $G(S) \leq 2G \|\mathcal{X}\|$ and $H(S) \leq 2H \|\mathcal{X}\|^2$. Hence, using the fact that $\mathcal{A}$ is $(\epsilon, \delta/2)$-DP, we have that Algorithm 8 is (ϵ, δ)-DP.

We now proceed to the utility part. Let $\widetilde{w} \in \mathbb{R}^k$ be the output of the base algorithm in low dimensions. Note that the final output is $\widehat{w} = \Phi^\top \widetilde{w}$. The transpose of the JL matrix can only increase the norm by the polynomial factor of d and n, hence $\|\widehat{w}\| \leq \text{poly}(n,d) \|\widetilde{w}\|$. By assumption, $\mathbb{P}(\|\widetilde{w}\| > \text{poly}(n,d,G,H)) \leq \frac{1}{\sqrt{n}}$. Hence we also have that $\mathbb{P}(\|\widehat{w}\| > \text{poly}(n,d,G,H)) \leq \frac{1}{\sqrt{n}}$. Let $\mathcal{W} \subseteq \mathbb{R}^d$ denote the above set with radius $\text{poly}(n,d,G,H)$.

We now decompose the population stationarity as,

$$\begin{aligned}
\mathbb{E}\|\nabla L(\widehat{w};\mathcal{D})\| &\leq \mathbb{E}\left\|\nabla L(\widehat{w};\mathcal{D}) - \nabla \widehat{L}(\widehat{w};S)\right\| + \left\|\nabla \widehat{L}(\widehat{w};S)\right\| \\
&\leq \mathbb{E}\sup_{w\in\mathcal{W}}\left\|\nabla L(w;\mathcal{D}) - \nabla \widehat{L}(w;S)\right\| + \frac{G\|\mathcal{X}\|}{\sqrt{n}} + \mathbb{E}\left\|\nabla \widehat{L}(\widehat{w};S)\right\|, \qquad \text{(B.7)}
\end{aligned}$$

where the last inequality follows from the above reasoning that that $\mathbb{P}(\widehat{w} \in \mathcal{W}) \geq 1 - \frac{1}{\sqrt{n}}$. The first term is bounded from uniform convergence guarantee in Lemma 25 noting that the dependence on $\|\mathcal{W}\|$ in the Lemma is only poly-logarithmic.

$$\mathbb{E}\sup_{w\in\mathcal{W}}\left\|\nabla L(w;\mathcal{D}) - \nabla \widehat{L}(w;S)\right\| = \widetilde{O}\left(\frac{G\|\mathcal{X}\|}{\sqrt{n}}\right) \qquad \text{(B.8)}$$

We now prove a bound on the empirical stationarity. Note that it suffices to prove a high-probability (over the random JL matrix) bound because the norm of gradient is bounded in worst case by $G\|\mathcal{X}\|$. Thus the expected norm of gradient of the output is bounded by the high probability bound by considering a small enough failure probability.

From the assumption on $\mathcal{A}$, with probability at least $1-\delta/2$,

$$\left\|\nabla \widehat{L}(\widetilde{w};\Phi S)\right\| = \mathbb{E}\left\|\frac{1}{n}\sum_{i=1}^{n}\phi'_{y_i}(\langle \widetilde{w}, \Phi x_i\rangle)\Phi x_i\right\| \leq g(k,n,2G\|\mathcal{X}\|, 2G\|\mathcal{X}\|, \epsilon, \delta/2)$$

We now use the fact that if $k = O(\mathsf{rank}\log(2n/\delta))$, then the JL transform is an $(\text{rank}, 1/2, \delta/2)$ oblivious subspace embedding (see Definition 20). Thus, it approxi-

mates the norm of any vector in $\mathsf{span}(\{x_i\}_{i=1}^n)$, and hence any gradient. Therefore,

$$\begin{aligned}
\mathbb{E}\left\|\nabla \widehat{L}(\widetilde{w};\Phi S)\right\| &= \mathbb{E}\left\|\Phi\left(\frac{1}{n}\sum_{i=1}^{n}\phi'_{y_i}(\langle \widetilde{w},\Phi x_i\rangle)x_i\right)\right\| \\
&\geq \left(1-\sqrt{\frac{\mathsf{rank}}{k}}\right)\mathbb{E}\left\|\frac{1}{n}\sum_{i=1}^{n}\phi'_{y_i}(\langle \widetilde{w},\Phi x_i\rangle)x_i\right\| \\
&\geq \frac{1}{2}\mathbb{E}\left\|\frac{1}{n}\sum_{i=1}^{n}\phi'_{y_i}(\langle \widetilde{w},\Phi x_i\rangle)x_i\right\| \\
&= \frac{1}{2}\mathbb{E}\left\|\frac{1}{n}\sum_{i=1}^{n}\phi'_{y_i}(\left\langle \Phi^\top \widetilde{w}, x_i\right\rangle)x_i\right\| \\
&= \frac{1}{2}\mathbb{E}\left\|\nabla \widehat{L}(\widehat{w};S)\right\|
\end{aligned}$$

Thus with $k = O\left(\mathsf{rank}\log\left(2n/\delta\right)\right)$, we get

$$\mathbb{E}\left\|\nabla \widehat{L}(\widehat{w};S)\right\| \leq g(k,n,2G\left\|\mathcal{X}\right\|,2H\left\|\mathcal{X}\right\|^2,\epsilon,\delta) = g(\mathsf{rank},n,2G\left\|\mathcal{X}\right\|,2H\left\|\mathcal{X}\right\|^2,\epsilon,\delta)$$

For the other bound, let $I_{d-k} \in \mathbb{R}^{d\times k}$ denote the matrix with first k diagonal entries, $(I_{d-k})_{j,j}$ with $j \in [k]$, are 1 and the rest of the matrix is zero. We have,

$$\begin{aligned}
&\mathbb{E}\left\|\nabla \widehat{L}(\widehat{w};S)\right\| \\
&= \mathbb{E}\left\|\frac{1}{n}\sum_{i=1}^{n}\phi'_{y_i}(\left\langle \Phi^\top \widetilde{w}, x_i\right\rangle)x_i\right\| \\
&\leq \mathbb{E}\left\|\frac{1}{n}\sum_{i=1}^{n}\phi'_{y_i}(\langle \widetilde{w},\Phi x_i\rangle)I_{d-k}\Phi x_i\right\| \\
&+ \mathbb{E}\left[\left\|\frac{1}{n}\sum_{i=1}^{n}\phi'_{y_i}(\langle \widetilde{w},\Phi x_i\rangle)x_i - \frac{1}{n}\sum_{i=1}^{n}\phi'_{y_i}(\langle \widetilde{w},\Phi x_i\rangle)I_{d-k}\Phi x_i\right\|\right] \\
&\leq \mathbb{E}\left\|I_{d-k}\right\|\left\|\frac{1}{n}\sum_{i=1}^{n}\phi'_{y_i}(\langle \widetilde{w},\Phi x_i\rangle)\Phi x_i\right\| + \frac{1}{n}\mathbb{E}\sum_{i=1}^{n}\left|\phi'_{y_i}(\langle \widetilde{w},\Phi x_i\rangle)\right|\left|\left\|x_i - I_{d-k}\Phi x_i\right\|\right| \\
&\leq \mathbb{E}\left\|\nabla \widehat{L}(\widetilde{w};\Phi S)\right\| + \frac{1}{n}\mathbb{E}\sum_{i=1}^{n}G\left\|I - I_{d-k}\Phi\right\|\left\|x_i\right\| \\
&\leq g(k,n,2G\left\|\mathcal{X}\right\|,2H\left\|\mathcal{X}\right\|^2,\epsilon,\delta/2) + G\left\|\mathcal{X}\right\|\mathbb{E}\left\|I-\mathbf{H}\right\|
\end{aligned}$$

where the second inequality follows from triangle inequality, the third inequality follows from G-Lipschitzness of the GLM, the third inequality follows from the accuracy guarantee of the base algorithm and substituting $\mathbf{H} = I_{d-k}\Phi$. To bound $\mathbb{E}\left\|I-\mathbf{H}\right\|$, we use concentration properties of distribution used in the construction of JL matrices.

Specifically, using the scaled Gaussian matrix construction, from concentration of extreme eignevalues of square Gaussian matrices, we have that $\mathbb{E}\left\|I - \mathbf{H}\right\| = \widetilde{O}\left(\frac{1}{\sqrt{k}}\right)$ [RV10]. This gives us,

$$\mathbb{E}\left\|\nabla \widehat{L}(\widehat{w}; S)\right\| \leq g(k, n, 2G\left\|\mathcal{X}\right\|, 2H\left\|\mathcal{X}\right\|^2, \epsilon, \delta/2) + \widetilde{O}\left(\frac{G\left\|\mathcal{X}\right\|}{\sqrt{k}}\right)$$

Choosing k to minimize the above yields the bound of $\widetilde{O}\left(\frac{G\|\mathcal{X}\|}{\sqrt{k}}\right)$. Combining the two cases, yields the bound of $g(k, n, 2G\left\|\mathcal{X}\right\|, 2H\left\|\mathcal{X}\right\|^2, \epsilon, \delta/2)$ on gradient norm. Plugging this and the bound in Eqn. (B.8) in Inequality (B.7) gives the claimed bound. □

Lemma 25. *Let $\mathcal{D}$ be a probability distribution over $\mathcal{X}$ such that $\|x\| \leq \|\mathcal{X}\|$ for all $x \in supp(\mathcal{D})$. Let $\ell(w; (x, y)) = \phi_y(\langle w, x\rangle)$ be an H-smooth G-Lipschitz GLM. Then, with probability at least $1 - \beta$, over a draw of n i.i.d. samples S from $\mathcal{D}$, we have*

$$\sup_{w \in \mathcal{W}} \left\|\nabla L(w; \mathcal{D}) - \nabla \widehat{L}(w; S)\right\| \leq \frac{4G\left\|\mathcal{X}\right\| \log\left(2n^{3/2}\left\|\mathcal{W}\right\| H\left\|\mathcal{X}\right\|/G\right)}{\sqrt{n}} + \frac{4G\left\|\mathcal{X}\right\|\sqrt{\log(1/\beta)}}{\sqrt{n}}$$

Proof. We first give a bound on the expected uniform deviation term, which is $\mathbb{E}_{S\sim\mathcal{D}^n} \sup_{w\in\mathcal{W}} \left\|\nabla L(w; \mathcal{D}) - \nabla \widehat{L}(w; S)\right\|$. Note that the gradient of the loss function is $\nabla \ell(w; (x, y)) = \phi'_y(\langle w, x\rangle) x$. We start with the standard symmetrization trick,

$$\begin{aligned}
&\mathbb{E}_{S\sim\mathcal{D}^n} \sup_{w\in\mathcal{W}} \left\|\nabla L(w; \mathcal{D}) - \nabla \widehat{L}(w; S)\right\| \\
&= \mathbb{E}_{S\sim\mathcal{D}^n} \sup_{w\in\mathcal{W}} \left\|\mathbb{E}\phi'_y(\langle w, x\rangle) x - \frac{1}{n}\sum_{i=1}^n \phi'_{y_i}(\langle w, x_i\rangle) x_i\right\| \\
&= \mathbb{E}_{S\sim\mathcal{D}^n} \sup_{w\in\mathcal{W}} \left\|\mathbb{E}_{\{x'_i\}\sim\mathcal{D}^n} \frac{1}{n}\sum_{i=1}^n \phi'_{y'_i}(\langle w, x'_i\rangle) x'_i - \frac{1}{n}\sum_{i=1}^n \phi'_{y_i}(\langle w, x_i\rangle) x_i\right\| \\
&\leq \mathbb{E}_{S,S'\sim\mathcal{D}^n} \sup_{w\in\mathcal{W}} \left\|\frac{1}{n}\sum_{i=1}^n \phi'_{y'_i}(\langle w, x'_i\rangle) x'_i - \frac{1}{n}\sum_{i=1}^n \phi'_{y_i}(\langle w, x_i\rangle) x_i\right\| \\
&= \mathbb{E}_{S,S'\sim\mathcal{D}^n}\mathbb{E}_{\{\sigma_i\}} \sup_{w\in\mathcal{W}} \left\|\frac{1}{n}\sum_{i=1}^n \sigma_i\left(\phi'_{y'_i}(\langle w, x'_i\rangle) x'_i - \phi'_{y_i}(\langle w, x_i\rangle) x_i\right)\right\| \\
&\leq 2\mathbb{E}_{S\sim\mathcal{D}^n}\mathbb{E}_{\{\sigma_i\}} \sup_{w\in\mathcal{W}} \left\|\frac{1}{n}\sum_{i=1}^n \sigma_i \phi'_{y_i}(\langle w, x_i\rangle) x_i\right\| \qquad \text{(B.9)}
\end{aligned}$$

where σ_i are i.i.d. Rademacher random variables. For fixed $\{(x_i, y_i)\}_{i=1}^n$, consider a set $\mathcal{W}_0$ s.t. for all $w \in \mathcal{W}$ and $i \in [n]$, there exists $w_0 \in \mathcal{W}_0$ such that $|\langle w, x_i\rangle - \langle w_0, x_i\rangle| \leq \tau$. Since $\|w\| \leq \|\mathcal{W}\|$ and $\|x_i\| \leq \|\mathcal{X}\|$, we require only $\frac{2n\|\mathcal{W}\|\|\mathcal{X}\|}{\tau}$ points in $\mathcal{W}_0$ to satisfy the above covering condition. Therefore,

$$
\begin{aligned}
&\mathbb{E}_{S\sim\mathcal{D}^n}\mathbb{E}_{\{\sigma_i\}} \sup_{w\in\mathcal{W}} \left\| \frac{1}{n}\sum_{i=1}^n \sigma_i \phi'_{y_i}\left(\langle w, x_i\rangle\right) x_i \right\| \\
&= \mathbb{E}_{S\sim\mathcal{D}^n}\mathbb{E}_{\{\sigma_i\}} \sup_{w\in\mathcal{W}, w_0\in\mathcal{W}_0} \left\| \frac{1}{n}\sum_{i=1}^n \sigma_i \left(\phi'_{y_i}\left(\langle w, x_i\rangle\right) - \phi'_{y_i}\left(\langle w_0, x_i\rangle\right) + \phi'_{y_i}\left(\langle w_0, x_i\rangle\right)\right) x_i \right\| \\
&\leq \mathbb{E}_{S\sim\mathcal{D}^n}\mathbb{E}_{\{\sigma_i\}} \sup_{w\in\mathcal{W}, w_0\in\mathcal{W}_0} \left\| \frac{1}{n}\sum_{i=1}^n \sigma_i \left(\phi'_{y_i}\left(\langle w, x_i\rangle\right) - \phi'_{y_i}\left(\langle w_0, x_i\rangle\right)\right) x_i \right\| \\
&+ \left\| \frac{1}{n}\sum_{i=1}^n \sigma_i \phi'_{y_i}\left(\langle w_0, x_i\rangle\right) x_i \right\| \\
&\leq \mathbb{E}_{S\sim\mathcal{D}^n}\mathbb{E}_{\{\sigma_i\}} \sup_{w\in\mathcal{W}, w_0\in\mathcal{W}_0} H\left|\langle w, x_i\rangle - \langle w_0, x_i\rangle\right| \|\mathcal{X}\| \\
&+ \mathbb{E}_{S\sim\mathcal{D}^n}\mathbb{E}_{\{\sigma_i\}} \sup_{w_0\in\mathcal{W}_0} \left\| \frac{1}{n}\sum_{i=1}^n \sigma_i \phi'_{y_i}\left(\langle w_0, x_i\rangle\right) x_i \right\| \\
&\leq H\tau\|\mathcal{X}\| + \mathbb{E}_{S\sim\mathcal{D}^n}\mathbb{E}_{\{\sigma_i\}} \sup_{w_0\in\mathcal{W}_0} \left\| \frac{1}{n}\sum_{i=1}^n \sigma_i \phi'_{y_i}\left(\langle w_0, x_i\rangle\right) x_i \right\| \qquad \text{(B.10)}
\end{aligned}
$$

where the second last inequality follows from smoothness and the last from the definition of cover $\mathcal{W}_0$. For fixed w_0, from standard manipulations, we have,

$$
\begin{aligned}
\mathbb{E}_{\{\sigma_i\}} \left\| \frac{1}{n}\sum_{i=1}^n \sigma_i \phi'_{y_i}\left(\langle w_0, x_i\rangle\right) x_i \right\| &\leq \sqrt{\mathbb{E}_{\{\sigma_i\}} \left\| \frac{1}{n}\sum_{i=1}^n \sigma_i \phi'_{y_i}\left(\langle w_0, x_i\rangle\right) x_i \right\|^2} \\
&= \sqrt{\frac{1}{n^2}\mathbb{E}_{\{\sigma_i\}} \sum_{i=1}^n \left\| \sigma_i \phi'_{y_i}\left(\langle w_0, x_i\rangle\right) x_i \right\|^2} \\
&\leq \frac{G\|\mathcal{X}\|}{\sqrt{n}}
\end{aligned}
$$

Using Massart's finite class lemma to handle all $w_0 \in \mathcal{W}_0$, and substituting the above in Eqn. (B.10), we get,

$$
\mathbb{E}_{S\sim\mathcal{D}^n}\mathbb{E}_{\{\sigma_i\}} \sup_{w\in\mathcal{W}} \left\| \frac{1}{n}\sum_{i=1}^n \sigma_i \phi'_{y_i}\left(\langle w, x_i\rangle\right) x_i \right\| \leq H\tau\|\mathcal{X}\| + \frac{G\|\mathcal{X}\| \log\left(2n\|\mathcal{W}\|\|\mathcal{X}\|/\tau\right)}{\sqrt{n}}
$$

Choosing $\tau = \frac{G}{H\sqrt{n}}$, we get,

$$
\mathbb{E}_{S\sim\mathcal{D}^n}\mathbb{E}_{\{\sigma_i\}} \sup_{w\in\mathcal{W}} \left\| \frac{1}{n}\sum_{i=1}^n \sigma_i \phi'_{y_i}\left(\langle w, x_i\rangle\right) x_i \right\| \leq \frac{2G\|\mathcal{X}\| \log\left(2n^{3/2}\|\mathcal{W}\| H \|\mathcal{X}\|/G\right)}{\sqrt{n}}
$$

Finally, substituting the above in Eqn. (B.9) gives us the following in-expectation bound.

$$\mathbb{E}_{S\sim\mathcal{D}^n} \sup_{w\in\mathcal{W}} \left\|\nabla L(w;\mathcal{D}) - \nabla\widehat{L}(w;S)\right\| \leq \frac{4G\left\|\mathcal{X}\right\| \log\left(2n^{3/2}\left\|\mathcal{W}\right\| H\left\|\mathcal{X}\right\|/G\right)}{\sqrt{n}}$$

For the high-probability bound, let $\psi(S) = \sup_{w\in\mathcal{W}} \left\|\nabla L(w;\mathcal{D}) - \nabla\widehat{L}(w;S)\right\|$ and let $w^* \in \mathcal{W}$ achieves the supremum. We can bound the increment between neighbouring datasets S and S' as,

$$\begin{aligned}
|\psi(S) - \psi(S')| &\leq \left|\left\|\nabla L(w^*;\mathcal{D}) - \nabla\widehat{L}(w^*;S)\right\| - \left\|\nabla L(w^*;\mathcal{D}) - \nabla\widehat{L}(w^*;S')\right\|\right| \\
&\leq \left\|\nabla\widehat{L}(w^*;S) - \nabla\widehat{L}(w^*;S')\right\| \\
&\leq \frac{2G\left\|\mathcal{X}\right\|}{n}
\end{aligned}$$

Finally, applying McDiarmid's inequality gives the claimed bound. □

Proof of Corollary 1. The results follow from Theorem 21 provided we show that the conditions on the base algorithm in the Theorem statement are satisfied. The privacy and accuracy claims follow from Theorem 19 and 20 respectively. We note that even though we are given population stationarity guarantee for the convex case, the same bound for empirical stationarity guarantee simply follows from the re-sampling argument in [BFTGT19]. The only thing left to show is the high-probability bound on the trajectory of the algorithm.

Non-convex setting with Private Spiderboost. From the update in Algorithm 5, we have that for any t

$$\left\|\nabla_t\right\| \leq \sum_{i=1}^{t}\left\|\Delta_i\right\| + \left\|\sum_{i=1}^{t} g_t\right\| \leq 2tG + \left\|\sum_{i=1}^{t} g_t\right\|$$

where the last inequality follows from the Lipschitzness assumption. Note that $g_t \sim \mathcal{N}(0,\sigma_t^2\mathbb{I})$ where $\sigma_t \leq O\left(\max\left(\sigma_1,\widehat{\sigma}_2\right)\right) = O\left(\text{poly}(n,d,G,H)\right)$. Hence $\left\|\sum_{i=1}^{t} g_t\right\| \leq \sqrt{d\log\left(1/\beta'\right)}O\left(\text{poly}(n,d,G,H)\right)$ with probability at least $1-\beta'$. Taking a union bound

over all $t \in T$ gives us $\|w_t\| \leq \text{poly}(n, d, G, H, \log(\text{poly}(n, d)/\beta))$ with probability at least $1 - \beta$. Substituting $\beta = \frac{1}{\sqrt{n}}$ yields the guarantee of Theorem 21.

Convex setting with Recursive Regularization. Since the iterates are restricted to the constraint set, the final output, with probability one, lies in the set of radius

$$D_T = 2^{T/2} \|w^*\| = O\left(\sqrt{\frac{H}{\lambda}} \|w^*\|\right) = O\left(\frac{H \|w^*\|^{3/2} n}{G}\right)$$

which completes the proof. □

Appendix C

Appendix for Chapter 4

C.1 Additional Related Work

We survey the works on machine unlearning - [CY15] were one of the first papers to study the topic of machine unlearning. Their approach implements statistical query (SQ) algorithms by estimating the statistical queries using training data. Since the estimates are usually the mean of query evaluations computed on training data, unlearning is cheap, as we only need to subtract the evaluation on the deleted point. [BCCC+21] studies this problem, with the goal to design systems to efficiently handle deletion requests. Their approach, called SISA, is essentially a divide-and-conquer strategy, wherein the data is divided into disjoint sets, called *shards*, and a model on each shard is trained separately and aggregated. Furthermore, they do several check-pointing of states for each shard. In the average case, this provides a speedup of $\frac{(R+1)S}{2}$ for S shards and R checkpoints per shard, over retraining. They however give no guarantees on accuracy with this divide-and-conquer training method. [GGHVDM20] is another work which uses (ϵ, δ)-differential privacy based unlearning criterion. They study unlearning in generalized linear models, and propose a Newton-step based method, leveraging connections with influence functions. Their computational cost is $O(d^3)$ computations for one unlearning. They, however give no guarantees on excess empirical risk achieved by the training method. Finally, the work of [ISCZ21] studies

batch unlearning in linear regression, with the goal to improve the computational cost of batch k unlearning requests. Their method achieves a runtime of $O(k^2d)$ as opposed to $O(kd^2)$ for a naive approach. However, their notion of unlearning is again approximate, in the sense that model returned after unlearning is closest to the exact unlearning model among models in the d dimensional subspace spanned by the to-be-deleted k points. So it is easy to see that with larger k, the notion of approximation improves, which explains the k^2 term in the runtime as opposed to k.

Comparison with [NRSM21a]. Our algorithm guarantees provable exact unlearning with probabilistic runtime guarantees, whereas [NRSM21a] give algorithms with deterministic runtime and provide only an approximate (ϵ, δ)-DP based unlearning guarantee – the δ can be interpreted as probability of the failure event in Monte-Carlo guarantees. To handle these discrepancies when comparing, our stated runtime is the in-expecatation runtime. For a fixed runtime, we will look at regimes of ϵ and δ, when the accuracy guarantee of [NRSM21a] is smaller than ours. We remind that a large ϵ, δ means a weaker unlearning criterion. We have that with the same runtime, the accuracy of [NRSM21a] is smaller than ours in the regime when their unlearning parameters and hence the notion, is rather weak.

Considering the Lipschitz, smoothness parameters and diameter as constants, for smooth convex functions and k edit requests, [NRSM21a] (Theorem 3.4) achieve an excess empirical risk of $O\left(\frac{\sqrt{d}\sqrt{\log(1/\delta)}}{\epsilon nk}\right)^{2/5}$ with an unlearning runtime of k^2 full-gradient computations. On the other hand, our algorithms achieve an an excess empirical risk of $\min\left\{\frac{1}{\sqrt{\rho n}}, \left(\frac{\sqrt{d}}{\rho n}\right)^{4/5}\right\}$ with ρk expected re-computations. Each re-computation takes $m \cdot T$ gradient computations where m is the mini-batch size and T the number of iterations. Therefore, in order to have the same runtime, we need $\rho kmT = k^2 n \iff \rho = \frac{kn}{mT}$. Firstly, note that as as long as $d \leq (\rho n)^{3/4}$, *noisy-m-A-SGD* has smaller excess empirical risk than *sub-sample-GD* - this are the two

regimes of interest. We now set m and T for both the algorithms: for Algorithm 19, $m = \frac{\rho n}{T}$ and $T = \sqrt{\rho n}$. This gives us $\rho = \frac{kn}{\rho n} \iff \rho = \sqrt{k}$, however ρ is the total variation distance and is at most 1. Hence in regime $d \geq (\rho n)^{3/4}$, our runtime is always smaller than [NRSM21a]: kn as opposed to $k^2 n$ gradient computations. Even with $\rho = 1$, our excess empirical risk is $O\left(\frac{1}{\sqrt{n}}\right)$ and the excess empirical risk of [NRSM21a] is smaller than ours when $\left(\frac{\sqrt{d}\sqrt{\log(1/\delta)}}{\epsilon n k}\right)^{2/5} = O\left(\frac{1}{\sqrt{n}}\right) \iff \frac{\epsilon}{\sqrt{\log(1/\delta)}} = \Omega\left(\frac{\sqrt{d} n^{1/4}}{k}\right)$. In the second regime $d < (\rho n)^{3/4}$, we use Algorithm 9, wherein we have $mT = \frac{(\rho n)^2}{d}$. This gives us $\rho = \frac{knd}{(\rho n)^2} \iff \rho = \sqrt{\frac{kd}{n}}$, and our excess empirical risk is $O\left(\left(\frac{\sqrt{d}}{\rho n}\right)^{4/5}\right) = O\left(\frac{1}{(nk)^{2/5}}\right)$. Therefore, excess empirical risk of [NRSM21a] is smaller than ours when $\left(\frac{\sqrt{d}\sqrt{\log(1/\delta)}}{\epsilon n k}\right)^{2/5} = O\left(\frac{1}{(nk)^{2/5}}\right) \iff \frac{\epsilon}{\sqrt{\log(1/\delta)}} = \Omega(d)$. We therefore have that unless k is very large, the accuracy of [NRSM21a] is smaller than ours when ϵ and δ, take prohibitively large values which correspond to a weak notion of approximate unlearning. We can similarly compare against Theorem 3.5 in [NRSM21a], which will yield qualitatively similar conclusions.

We now compare our space complexity with that of [NRSM21a]. Firstly, note that we need not consider regime $d \geq (\rho n)^{3/4}$, since here we have better accuracy than [NRSM21a], and our algorithm (*sub-sample-GD*) need not save any iterate. For regime, $d < (\rho n)^{3/4}$, from Section C.7.3, the space complexity of *noisy-m-A-SGD* is $\max\left\{\frac{(\rho n)^2}{d}, d^{3/4}\sqrt{\rho n}\right\} = \max\left\{kn, (kn)^{1/4}\right\}$, where we plugged in $\rho = \sqrt{\frac{kd}{n}}$ for our runtime to be same as [NRSM21a]. Since $d < (\rho n)^{3/4}$, the maximum term is kn. Hence, we get same runtime as [NRSM21a], better accuracy (for reasonable ϵ, δ) with space complexity $= O(kn)$ – so for moderate values of k, this is smaller than space to store the dataset, which both our result and [NRSM21a] require.

C.2 Additional Discussion

C.2.1 Total Variation Stability from Optimal Transport

In this section, we give a didactic treatment of our approach to motivate the notion of total variation stability. Consider neighbouring datasets S and S' and let $P = \mathcal{A}(S)$ and $Q = \mathcal{A}(S')$ for some randomized algorithm $\mathcal{A}$. The algorithm first computes on S, and then observes edit requests which generate S' as the current dataset. To satisfy exact unlearning , we need a procedure which *moves* P to Q. This is akin to the well-studied optimal transport problem [Vil09], discussed in Section 1.2.4.1, which we briefly recall below.

Given probability distributions P and Q over measurable space $\mathcal{X}$, and a cost function $c : \mathcal{X} \times \mathcal{X} \to \mathbb{R}$, the goal is to find a transport plan π which minimizes the expected cost: $\min_{\pi \in \Pi(P,Q)} \mathbb{E}_{(x,y)\sim\pi} c(x, y)$.

A model of computation. Note that there is of course the trivial coupling in which we generate independent samples from P and Q - this corresponds to re-computation, which is not practical in general. Instead, we should correlate P and Q so that transporting from P to Q can reuse the randomness (computation) used for P. For this, we use the cost function in the optimal transport problem as a surrogate of modelling computation. In the optimal transport problem, the cost is typically a *distance* on the space, whereas we are concerned with computational cost. So is there a distance function which corresponds to computational cost? Note that the sequential nature of the problem already gives us samples generated from P, so a natural question is, can we use this to transport to Q? We can set the cost function as $c(x, y) = \begin{cases} 1 & \text{if } x \neq y \\ 0 & \text{otherwise} \end{cases}$. This corresponds to an oracle which charges a unit computation if we use y which is different from x, which can correspond to a recomputation. Under this simple model of computation, the optimal expected

computational cost becomes exactly equal to the total variation distance between P and Q: $\inf_{\pi \in \Pi(P,Q)} \mathbb{1}\{x \neq y\}$ - the maximal coupling characterization of total variation distance.

TV Stability. The above establishes that if we want to transport P to Q using minimum computation cost, the expected computation cost cannot be smaller than the total variation distance between P and Q. Intuitively, this means that is least $1 - \mathrm{TV}(P, Q)$ fraction of samples are *representative* for both P and Q. From the sequential nature of our problem, when we generate P - the output on dataset S, we don't know what Q would be, since we don't know the incoming edit request. Hence a reasonable property to have in the algorithm is that its output is close in total variation distance *uniformly* over all possible Q's. This motivates our definition of total variation stability.

Optimal Transport vs Unlearning: Unlike the optimal transport problem wherein we are given P and Q, and the task is to find a coupling, in our setup, we have to find an algorithm generating P and Q as well as the coupling. Moreover, for a fixed ρ, there may be many algorithms which are ρ-TV stable. The goal therefore, is to find among these algorithms, the one with the maximum accuracy for the (convex ERM) problem, and for which we can design a corresponding *efficient* unlearning algorithm.

C.2.2 DP Convex ERM Algorithms for Unlearning

We first discuss an important distinction in the differential privacy and our unlearning setup. In the DP setup, we have a curator which possesses the dataset, and an analyst/adversary, against which the curator want to provide privacy. The analyst queries the dataset, and the curator provides DP answers to the queries. The curator can also reveal additional information pertaining to the algorithmic details, however, it is beneficial to the curator to only release limited information. In particular, the

curator can chose to keep certain states of the algorithm secret. This could be done in the case when only the marginals of the output satisfy a strong DP-guarantee. So, if the curator were to release the secret state as well, the adversary can correlate information and then the privacy level, which is now measured using the joint distribution of output and state, degrades. In the *noisy-m-A-SGD* algorithm for example, the output typically is the average or final iterate whereas the rest of iterates and mini-batch indices b_j's are the secret state.

In the unlearning setup, there is no such adversary per se, or in the idealized application, the curator is the adversary and the dataset owners wants it to have as little control as possible. It is therefore natural to demand that the probability distribution of the entire state maintained by the algorithm, and not just the output be exactly identical after performing the unlearning operation. This, with slight differences, is referred to as perfect unlearning in [NRSM21a], and what our algorithms satisfy. We have argued that designing TV stable algorithms is a good start, and for a moment suppose that the TV stability is same as DP. Then should we measure TV stability between the joint distributions over the entire state? This would limit the application of DP techniques in which keeping additional state hidden has stronger privacy property. In that case, TV stability parameter, and hence the computational cost of unlearning would be large. Interestingly, even though the previous work in differentially private convex ERM, for example [BST14], show that the released iterate (average/final iterate) is differentially private, the analysis is typically carried out by first arguing, via a composition step, that *all* iterates together are differentially private. This means that all iterates can be released without any additional cost of privacy. This innocuous property arguably provides no benefit for privacy, but turns out to be extremely beneficial to us in unlearning. However, even though the all the iterates can be released, the mini-batches still need to kept secret. We handle this in the unlearning algorithm using an estimation step - see paragraph titled "Estimation

of marginals" in Section 4.5.2.

C.3 sub-sample-GD

The algorithm which is superior in high dimensions, called *sub-sample-GD*, is just vanilla mini-batch SGD. Herein, at each iteration, a mini-batch of size m is sub-sampled uniformly randomly to compute the gradient, and make the update. Finally, we save all the mini-batch indices, gradients and iterates to memory. We will see that the unlearning algorithm presented (Algorithm 20) uses all the saved iterates. However this is done only for ease of presentation - in Section C.7.3, we discuss a simple efficient implementation (of the unlearning algorithm), which doesn't need *any* iterate, yet has the same unlearning time complexity.

Algorithm 19 *sub-sample-GD*(w_{t_0}, t_0)

Input: Initial model w_{t_0}, data points $\{z_1, \ldots, z_n\}, T, m, \eta$

1: **for** $t = t_0, t_0 + 1 \ldots, T$ **do**
2: Sample mini-batch b_t of size m uniformly randomly
3: $g_t = \frac{1}{m} \sum_{j \in b_t} \nabla \ell(w_t; z_j)$
4: $w_{t+1} = \mathcal{P}\left(w_t - \eta g_t\right)$
5: Save(b_t, w_t, g_t)
6: **end for**

Output: $\widehat{w}_S = \frac{1}{T} \sum_{t=1}^{T+1} w_t$

We now give guarantees on excess empirical risk for *sub-sample-GD*.

Proposition 3. *Let $\ell(.; z)$ be an H-smooth G-Lipschitz convex function $\forall\ z$. Algorithm 19, run with $t_0 = 1, \eta = \min\left\{\frac{1}{2H}, \frac{D\sqrt{\rho n}}{GT}\right\}, T = \frac{DH\sqrt{\rho n}}{G}$, and $m = \max\left\{\frac{G\sqrt{\rho n}}{DH}, 1\right\}$, outputs $\widehat{w}_S$ which is $\min\{\rho, 1\}$-TV-stable and satisfies $\mathbb{E}\left[\widehat{L}(\widehat{w}_S; S) - \widehat{L}(w^*_S; S)\right] = O\left(\frac{GD}{\sqrt{\rho n}}\right)$.*

C.3.1 Unlearning for *sub-sample-GD*

At the start of the stream, at every iteration of *sub-sample-SGD*, we sample a mini-batch of size m out of n points uniformly randomly, and then compute a gradient

using these samples - note that this is the only source of randomness in the algorithm. As we progress along the stream observing edit requests, the number of available data points changes. Therefore, if the algorithm were executed on this dataset of, say $\tilde{n}$ points, at every iteration it would have sub-sampled m out of $\tilde{n}$ (and not n) points. The way to account for this discrepancy is to simply adjust the sub-sampling probability measure accordingly.

Coupling mini-batch indices. The main idea to unlearning in Algorithm 20 is to couple the sub-sample indices. For deletion, we just look at each mini-batch, and (literally) verify if the deleted point were used or not. If the deletion point was not used in any iterations, then we don't do anything, otherwise, we trigger a recompute. In the case of insertion, there is no such way of selecting iterations in which the point was sampled, because the inserted point was absent. However, we know that the new point would have been sampled with probability $m/(n+1)$. We can thus verify by selecting each iteration with the same probability. We then replace a uniformly sampled point in the mini-batch of that step by the inserted point. Algorithm 20 implements the above procedure.

Algorithm 20 Unlearning for *sub-sample-GD*

Input: Data point index j to delete or data point z to insert (index $n+1$)

for $t = 1, 2 \ldots, T$ **do**
  Load(b_t, g_t, w_t)
  if *deletion* and $j \in b_t$ **then**
    sub-sample-GD(w_t, t) // *Continue training on current dataset*
    break
  else if *insertion* and Bernoulli$\left(\frac{m}{n+1}\right)$ **then**
    Sample $i \sim$ Uniform(b_t)
    $g'_t = g_t - \frac{1}{m}\left(\nabla \ell(w_t; z_i) - \nabla \ell(w_t; z)\right)$
    $w_{t+1} = \mathcal{P}\left(w_t - \eta\left(g'_t + \theta_t\right)\right)$
    Save$(w_{t+1}, g'_t, b_t \backslash \{i\} \cup \{n+1\})$
    sub-sample-GD$(w_{t+1}, t+1)$ // *Continue training on current dataset*
    break
  end if
end for

We state our main result for unlearning with Algorithm 20 below.

Proposition 4. *(Algorithm 19, Algorithm 20) satisfies exact unlearning. Moreover, for k edits, Algorithm 20 recomputes with probability at most $2k\rho$.*

C.4 Proofs of Main Results

In this section, we give the proofs of main results, stated in Section 4.3, using the results in the preceding sections.

C.4.1 Proof of Theorem 22

The proof follows by combining the guarantees for the two algorithms we present: *sub-sample-GD* (Algorithm 19) and *noisy-m-A-SGD* (Algorithm 9), and their corresponding unlearning algorithms: Algorithm 20 and Algorithm 10. We discuss these one by one. From Proposition 3, we have that, given $0 < \rho \leq 1$, *sub-sample-GD* is ρ-TV stable and has excess empirical risk bounded by $O\left(\frac{GD}{\sqrt{\rho n}}\right)$. This holds at every point in the stream by assumption that the number of samples are between $\frac{n}{2}$ and $2n$. Furthermore, from Proposition 4, we have that the unlearning algorithm satisfies exact unlearning at every point in the stream, proving the first part of the claim for *sub-sample-GD*. Moreover, it states that recompute probability for k edit requests is $O(\rho k)$. Finally, from Claim 7, we have that there exist efficient implementations, such that the runtime of unlearning for *sub-sample-GD* is $O(\max\{k, \min\{\rho, 1\}\, k \cdot \text{Training time}\}$, where "Training time" is the runtime of the corresponding learning algorithm - this means that re-computations overwhelm the total unlearning time. This establishes all the guarantees for one algorithm and recovers one of the upper bounds in the second claim.

The situation for the other algorithm is a little more involved. From Proposition 1, for dataset S of n points, we have that, given $0 < \widetilde{\rho} \leq 1$, *noisy-m-A-SGD* is $\widetilde{\rho}$-TV

stable and its excess empirical risk is bounded as follows:

$$\mathbb{E}\left[\widehat{L}(\widehat{w}_S; S) - \widehat{L}(w_S^*; S)\right] = O\left(\frac{HD^2}{T^2} + \frac{GD}{\sqrt{Tm}} + \frac{GD\sqrt{d}}{n\widetilde{\rho}}\right),$$

where T is the number of iterations for *noisy-m-A-SGD* algorithm, and m the mini-batch size. From Proposition 2, we have that the unlearning algorithm satisfies exact unlearning (establishing the first claim) and recomputes, for k edit requests, with probability $O(\widetilde{\rho}k\sqrt{T})$. Finally, from Claim 8, we have that there exist efficient implementations, such that the runtime of unlearning for *noisy-m-A-SGD* is $O(\max\{k, k\min\left\{\widetilde{\rho}\sqrt{T}, 1\right\} \cdot \text{Training time}\})$. In the statement of Theorem 22, we want that the unlearning runtime be such that we recompute for a ρ fraction of edit requests (as opposed to something dependent on T). Therefore, we substitute $\widetilde{\rho} = \frac{\rho}{\sqrt{T}}$, and this changes the excess empirical risk bound for *noisy-m-A-SGD*, as follows:

$$\mathbb{E}\left[\widehat{L}(\widehat{w}_S; S) - \widehat{L}(w_S^*; S)\right] = O\left(\frac{HD^2}{T^2} + \frac{GD}{\sqrt{Tm}} + \frac{GD\sqrt{d}\sqrt{T}}{n\rho}\right).$$

We use the largest mini-batch size, which does not hurt runtime, which is $m = \left(\frac{G}{HD}\right)^2 T^3$. This simplifies the upper bound to $\frac{HD^2}{T^2} + \frac{GD\sqrt{d}\sqrt{T}}{n\rho}$. Optimizing the trade-off, we have $\frac{HD^2}{T^2} = \frac{GD\sqrt{d}\sqrt{T}}{n\rho} \iff T = \left(\frac{HD(n\rho)}{G\sqrt{d}}\right)^{2/5}$, and the excess empirical risk becomes $\mathbb{E}\left[\left[\widehat{L}(\widehat{w}_S; S) - \widehat{L}(w_S^*; S)\right]\right] = O\left(\frac{HD^2}{T^2}\right) = O\left(\left(\frac{H^{1/4}GD^{3/2}\sqrt{d}}{(\rho n)}\right)^{4/5}\right)$ – this recovers the other term in the upper bound in Theorem 22. However, note that Proposition 1 has an additional condition that $T \geq \frac{(n\widetilde{\rho})^2}{16m^2}$ - we show that in our setting of $\widetilde{\rho}$ and m, this condition is equivalent to the excess empirical risk of *noisy-m-A-SGD* being smaller than that of *sub-sample-GD*. Hence, the regime in which the aforementioned condition is violated is the same regime in which it is better to use the other *sub-sample-GD* algorithm, and therefore is benign. Setting $m = \left(\frac{G}{HD}\right)^2 T^3$ and $\widetilde{\rho} = \rho/\sqrt{T}$, the condition simplifies as $T \geq \frac{(n\rho)^2}{16T \cdot T^6}\left(\frac{HD}{G}\right)^2 \iff T^8 \geq \frac{(n\rho)^2}{16}\left(\frac{HD}{G}\right)^4 \iff \left(\frac{HD(n\rho)}{G\sqrt{d}}\right)^{16/5} \geq \frac{(n\rho)^2}{16}\left(\frac{HD}{G}\right)^4 \iff \left(\frac{\sqrt{d}}{(n\rho)}\right)^{4/5} \leq \frac{2}{\sqrt{n\rho}}\left(\frac{HD}{G}\right)^{1/5} \iff \left(\frac{H^{1/4}GD^{3/2}\sqrt{d}}{(\rho n)}\right)^{4/5} \leq \frac{2GD}{\sqrt{n\rho}}$, where the final inequality indicates that the expected excess empirical risk

of *noisy-m-A-SGD* is at most that of *sub-sample-GD*, up to constants. The above is established for dataset S but holds for any dataset S^i in the stream using the assumption that the number of samples are between $\frac{n}{2}$ and $2n$.

Combining the above arguments finishes the proof of Theorem 22. ■

C.4.2 Proof of Theorem 23

We give two algorithms, *sub-sample-GD* (Algorithm 19) and *noisy-m-A-SGD* (Algorithm 9), one for each of the upper bounds. From Proposition 3 and Corollary 7, we have that, given $0 < \rho < \infty$, these are $\min\{\rho, 1\}$-TV stable and their excess empirical risk is bounded is $O\left(\frac{GD}{\sqrt{\rho n}}\right)$ and $O\left(\frac{GD\sqrt{d}}{\widetilde{\rho n}}\right)$ respectively. Hence combining the above by taking a minimum, establishes the claimed result. ■

C.4.3 Proof of Theorem 24

In all the lower bounds, we have a GD term - this is a trivial lower bound, since if an algorithm is defined as $\mathcal{A}(S) = 0$ (or any constant), then this is perfectly TV stable ($\rho = 0$), and the expected excess empirical risk is upper bounded as $\widehat{L}(\mathcal{A}(S); S) - \widehat{L}(w_S^*; S) \leq G \left\|\mathcal{A}(S) - w_S^*\right\| \leq GD$, where the first inequality uses G-Lipschitzness of $\widehat{F}_S$ and the second the fact the both $\mathcal{A}(S)$ and w_S^* lie in a ball of diameter D. Hence, attaining an excess empirical risk of GD is trivial, and we now focus on deriving the other terms in the bounds.

Firstly, as discussed in [BST14], we consider $G = D = 1$, since a simple reduction gives a factor of GD for general G and D. Furthermore, similar to [BST14], we show that the problem of TV-stable convex ERM is at least as hard as that of TV stable mean computation of a dataset with bounded mean - we state this reduction in Proposition 10. We now focus on showing accuracy lower bounds for ρ-TV-stable mean computation of dataset S of size n, with mean $\frac{M}{2} \leq \|\mu(S)\| \leq 2M$. The accuracy, denoted by α, is defined as $\alpha^2 = \mathbb{E}\left\|\mathcal{A}(S) - \mu(S)\right\|^2$, $\mathcal{A}$ is a ρ-TV stable algorithm,

and the expectation is taken over the algorithm's randomness. The first part of Theorem 24 follows Theorem 46 which is based on a simple reduction argument. This gives us that $\alpha \geq \frac{1}{\rho n}$ with $M = \frac{1}{\rho n}$. Plugging it in Proposition 10, this gives us that excess empirical risk is lower bounded by $\Omega\left(\frac{1}{\rho n}\right)$. Similarly, the second part follows from Theorem 47 which gives us $\alpha \geq \frac{1}{\sqrt{\rho n}}$ with $M = \frac{1}{\sqrt{\rho n}}$ - the condition $\alpha \leq \frac{1}{4}$ in the statement of Theorem 47 can be absorbed in the trivial lower bound GD. ■

C.5 Proofs for Section 4.5.1

Proof of Proposition 3. We first show that Algorithm 19 is $\min\{1, \rho\}$-TV stable for the aforementioned choice of number of iterations T and mini-batch size m. Consider neighbouring dataset S and S' of n points which differs in one sample, without loss of generality, say the n^{th} sample. Let $\mathcal{A}(S) := \widehat{w}_S$ and $\mathcal{A}(S') := \widehat{w}_{S'}$ denote the outputs of Algorithm 19 on S and S' respectively. Since in Algorithm 19, the randomness is only on indices, rather than actual data points, say that $S = \{1, 2, \ldots, n\}$. Now we consider neighbouring dataset S', which contains $n+1$ or $n-1$ samples. We will now consider the case when S' contains $n-1$ elements and the case with $n+1$ elements will follow analogously. Let n be the index present in S but absent in S' i.e. $S' = \{1, 2, \ldots, n-1\}$. Let the sigma-algebra on these sets be the power sets of S and S' respectively, denoted by $\text{Pow}(S)$ and $\text{Pow}(S')$ respectively. Moreover, let $\mu_{n,m}$ denote the sub-sampling probability measure on n points in S i.e it sub-samples m out of n elements in S uniformly randomly. Let $\mu_{n,m}^{\otimes T}$ denote the product measure of T of $\mu_{n,m}$'s. We similarly define $\mu_{n-1,m}$ and $\mu_{n-1,m}^{\otimes T}$ for S'.

We first extend the sigma-algebra for the probability spaces so that the random variables $\mu_{n,m}$ and $\mu_{n-1,m}$, are defined on a common probability space. For this, we will just add an event where the index n can be sampled under $\mu_{n-1,m}$ with probability 0. We define $\mu'_{n,m}$ as follows: for any set $b \in \text{Pow}(S)$, $\mu'_{n,m}(b) = \begin{cases} \mu_{n-1,m}(b) & \text{if } n \notin b \\ 0 & \text{otherwise} \end{cases}$.

We similarly extend the sigma algebra for the product space with measure $\mu_{n-1,m}^{\otimes T}$ to get $\mu_{n,m}^{\prime\otimes T}$.

Observe that for fixed initialization w_0 and other parameters, Algorithm $\mathcal{A}(S)$ and $\mathcal{A}(S')$ is the *same* (deterministic) map from $\mathbf{b} = (b_1, b_2, \ldots, b_T)$ where $b_j \in [n]^m$ to $\mathcal{W}$. They only differ because of different measures on the input space. Hence total variation distance between $\mathcal{A}(S)$ and $\mathcal{A}(S')$ is just the total variation distance between the push-forward measures $\mathcal{A}(S)_{\#}\mu_{n,m}^{\otimes T}$ and $\mathcal{A}(S')_{\#}\mu_{n,m}^{\prime\otimes T}$ which by using the fact that $\mathcal{A}(S) \equiv \mathcal{A}(S')$ and data-processing inequality, is at most the total variation distance between $\mu_{n,m}^{\otimes T}$ and $\mu_{n,m}^{\prime\otimes T}$. Now the total variation distance can be bounded as,

$$\begin{aligned}
\mathrm{TV}(\mathcal{A}(S), \mathcal{A}(S')) \leq \mathrm{TV}(\mu_{n,m}^{\otimes T}, \mu_{n,m}^{\prime\otimes T}) &= \sup_{\mathbf{b}\in \mathrm{Pow}([n]^m)^T)} \left|\mu_{n,m}^{\otimes T}(\mathbf{b}) - \mu_{n,m}^{\prime\otimes T}(\mathbf{b})\right| \\
&= \mu_{n,m}^{\otimes T}\left(\mathbf{b} \text{ such that at least one } b_j \text{ contains } n\right) \\
&\leq T\mu_{n,m}(b_1 \text{ contains } n) = \frac{Tm}{n}
\end{aligned}$$

where the inequality follows using a union bound.

A similar argument works when S' is an neighbouring dataset of $n+1$ elements, yielding a total variation bound of $\frac{Tm}{n+1} \leq \frac{Tm}{n}$. Taking a uniform bound over all neighbouring datasets S', we get that $\sup_{\Delta(S,S')=1} \mathrm{TV}(\mathcal{A}(S), \mathcal{A}(S')) \leq \frac{Tm}{n}$. By definition of TV distance, we trivially have that $\sup_{\Delta(S,S')=1} \mathrm{TV}(\mathcal{A}(S), \mathcal{A}(S')) \leq 1$. Therefore, setting $m = \frac{\rho n}{T}$, we get the desired result that the output of Algorithm 19 is $\min\{\rho, 1\}$-TV stable.

We now proceed to the accuracy guarantee which follows directly by analysis of SGD. We first show that the sub-sampling procedure produces unbiased gradients and

bound its variance. For a fixed model w, we have that

$$\begin{aligned}\mathbb{E}_b\left[\frac{\sum_{j\in b}\nabla\ell(w;z_j)}{m}\right] &= \sum_{\binom{n}{m}\text{ choices for }b}\frac{\sum_{j\in b}\nabla\ell(w;z_j)}{m\binom{n}{m}}\\ &= \frac{\binom{n-1}{m-1}}{m\binom{n}{m}}\sum_{j=1}^{n}\nabla\ell(w;z_j)\\ &= \frac{\sum_{j=1}^{n}\nabla\ell(w;z_j)}{n},\end{aligned}$$

where in the second equality, we use the observation that every z_j appears in exactly $\binom{n-1}{m-1}$ terms over all choices for b. We now bound its variance, denoted by a $\mathcal{V}^2$ by direct computation.

$$\begin{aligned}\mathcal{V}^2 &= \mathbb{E}_b\left\|\frac{\sum_{j\in b}\nabla\ell(w;z_j)}{m} - \mathbb{E}_b\left[\frac{\sum_{j\in b}\nabla\ell(w;z_j)}{m}\right]\right\|^2\\ &= \mathbb{E}_b\left\|\frac{\sum_{j\in b}\nabla\ell(w;z_j)}{m}\right\|^2 - \left\|\mathbb{E}_b\left[\frac{\sum_{j\in b}\nabla\ell(w;z_j)}{m}\right]\right\|^2\\ &= \sum_{\binom{n}{m}\text{ choices for }b}\frac{1}{\binom{n}{m}}\frac{1}{m^2}\left\|\sum_{j\in b}\nabla\ell(w;z_j)\right\|^2 - \left\|\frac{\sum_{j=1}^{n}\nabla\ell(w;z_j)}{n}\right\|^2\end{aligned}$$

In the first term, expanding the square and summing over all choices of b, we get exactly $\binom{n-1}{m-1}$ terms of the form $\|\nabla\ell(w;z_j)\|^2$ for $j=1$ to n, and $\binom{n-2}{m-2}$ cross terms of the form $\langle\nabla\ell(w;z_i),\nabla\ell(w;z_j)\rangle$ for $i\neq j$, $i,j=1$ to n. Similarly, expanding the second term produces both these kind of terms. Accumulating the coefficients of all

the terms, we get

$$
\begin{aligned}
\mathcal{V}^2 &= \mathbb{E}_b \left\| \frac{\sum_{j\in b} \nabla \ell(w; z_j)}{m} - \mathbb{E}_b \left[\frac{\sum_{j\in b} \nabla \ell(w; z_j)}{m} \right] \right\|^2 \\
&= \left(\frac{\binom{n-1}{m-1}}{m^2 \binom{n}{m}} - \frac{1}{n^2} \right) \sum_{j=1}^{n} \|\nabla \ell(w; z_j)\|^2 + \left(\frac{\binom{n-2}{m-2}}{m^2 \binom{n}{m}} - \frac{1}{n^2} \right) \sum_{i,j=1, i\neq j}^{n} \langle \nabla \ell(w; z_i), \nabla \ell(w; z_j) \rangle \\
&\leq \left(\frac{1}{mn} - \frac{1}{n^2} \right) nG^2 + \left| \frac{m-1}{nm(n-1)} - \frac{1}{n^2} \right| \sum_{i,j=1, i\neq j}^{n} \|\nabla \ell(w; z_i)\| \, \|\nabla \ell(w; z_j)\| \\
&\leq \left(\frac{1}{m} - \frac{1}{n} \right) G^2 + \left| \frac{m-1}{nm(n-1)} - \frac{1}{n^2} \right| n(n-1)G^2 \\
&= \left(\frac{1}{m} - \frac{1}{n} \right) G^2 + \left| \frac{m-1}{m} - \frac{(n-1)}{n} \right| G^2 \\
&= \left(\frac{1}{m} - \frac{1}{n} \right) G^2 + \left| \frac{1}{n} - \frac{1}{m} \right| G^2 \\
&= 2 \left(\frac{1}{m} - \frac{1}{n} \right) G^2 \leq \frac{2G^2}{m}
\end{aligned}
$$

where in the first inequality we used Cauchy-Schwartz inequality, and the fact the G-Lipschitzness implies the gradient norms are bounded by G. Finally, in the second last equality and the last inequality we used the fact that $m \leq n$.

Since the sub-sampled gradients are unbiased, we can use the convergence guarantee of SGD on smooth convex function (see Theorem 4.1 in [AZ18]) which when using step size $\eta \leq \frac{1}{H}$ gives us

$$
\mathbb{E}\left[\widehat{L}(\widehat{w}_S; S) - \widehat{L}(w_S^*; S) \right] = O\left(\frac{\eta \mathcal{V}^2}{(1 - \eta H)} + \frac{D^2}{\eta T} \right)
$$

Using step size $\eta \leq \frac{1}{2H}$, the right hand side simplifies to $2\eta\mathcal{V}^2 + \frac{D^2}{\eta T} \leq \frac{4G^2\eta}{m} + \frac{D^2}{\eta T} = \frac{4G^2 T\eta}{\rho n} + \frac{D^2}{\eta T}$, where in the last equality, we substituted $m = \frac{\rho n}{T}$ to ensure ρ TV-stability. Balancing the trade off in η gives us $\eta = \frac{D\sqrt{\rho n}}{GT}$. Therefore setting $\eta = \min\left\{ \frac{1}{2H}, \frac{D\sqrt{\rho n}}{GT} \right\}$ gives us

$$
\mathbb{E}\left[\widehat{L}(\widehat{w}_S; S) - \widehat{L}(w_S^*; S) \right] = O\left(\frac{GD}{\sqrt{\rho n}} + \frac{D^2 H}{T} \right)
$$

Setting $T = \frac{DH\sqrt{\rho n}}{G}$ achieves the claimed result. □

Proof of Proposition 1. We first prove the stability guarantee. For this, we use the Rènyi-divergence based analysis used in differential privacy literature. Let P and Q be probability distributions such that P is absolutely continuous with respect to Q and have densities ϕ_P and ϕ_Q, respectively. For $\alpha \in (1, \infty)$, the α Rènyi-divergence between P and Q is defined as follows [R+61]:

$$D_\alpha(P\|Q) = \frac{1}{\alpha - 1} \ln \left(\int f_P(x)^\alpha f_Q(x)^{1-\alpha} dx \right)$$

Consider two neighbouring datasets $S = \{z_j\}_j$ and $S' = \left\{z'_j\right\}_j$ such that $\Delta(S, S') = 1$, and let $\{b'_t\}_{t=1}^T$ and $\{w'_t\}_{t=1}^T$ denote the mini-batch indices and iterates of Algorithm 9 on dataset S' respectively. We look at iteration t, and fix all the randomness before t i.e. fix w_t (and w'_t), as well as randomness in sub-sampling mini-batch indices i.e. fix b_t. The α-Rènyi Divergence between w_{t+1} and w'_{t+1} can be bounded as,

$$\begin{aligned}
&D_\alpha(w_{t+1}\|w'_{t+1}) \\
&= D_\alpha\left(\mathcal{P}\left(\mathring{w}_t - \eta\left(\frac{\sum_{j\in b_t}\nabla\ell(\mathring{w}_t; z_j)}{m} + \theta_t\right)\right)\middle\|\mathcal{P}\left(\mathring{w}_t - \eta\left(\frac{\sum_{j\in b_t}\nabla\ell(\mathring{w}_t; z'_j)}{m} + \theta_t\right)\right)\right) \\
&\leq D_\alpha\left(\mathring{w}_t - \eta\left(\frac{\sum_{j\in b_t}\nabla\ell(\mathring{w}_t; z_j)}{m} + \theta_t\right)\middle\|\mathring{w}_t - \eta\left(\frac{\sum_{j\in b_t}\nabla\ell(\mathring{w}_t; z'_j)}{m} + \theta_t\right)\right) \\
&\leq D_\alpha\left(\frac{\sum_{j\in b_t}\nabla\ell(\mathring{w}_t; z_j)}{m} + \theta_t\middle\|\frac{\sum_{j\in b_t}\nabla\ell(\mathring{w}_t; z'_j)}{m} + \theta_t\right) \\
&\leq \frac{2\alpha G^2}{m^2\sigma^2}
\end{aligned}$$

where in the first and second inequality, we used post-processing property of Rènyi divergence, and in the last inequality, we use the fact that datasets S and S' differ in at most one sample, therefore $\left\|\frac{\sum_{j\in b_t}\nabla\ell(\mathring{w}_t;z_j)}{m} - \frac{\sum_{j\in b_t}\nabla\ell(\mathring{w}_t;z'_j)}{m}\right\|^2 \leq \frac{4G^2}{m^2}$. Hence the divergence is between two multivariate Gaussians of same variance and with the square of the separation of their means at most $\frac{2\alpha G^2}{m^2\sigma^2}$. Therefore, the inequality follows by using the formula for Rènyi divergence between two such multivariate Gaussians.

We now unfix b_t, and use the fact the b_t is a uniform sample of m out of n

(or $n-1$ or $n+1$) indices. By privacy amplification by sub-sampling result in [BBG18], for $\alpha \leq 2$, we will argue that the Rènyi divergence upper bound amplifies to $\frac{32\alpha G^2}{n^2\sigma^2}$. There are certain subtleties about the application of this result, so we explain, as follows. The first is that Theorem 9 stated in [BBG18], when considering $\alpha \leq 2$, the right hand side simplifies as $\frac{1}{\alpha-1}\log\left(1+\frac{m^2}{n^2}\frac{\alpha(\alpha-1)}{2}4\left(\exp\left(\frac{8G^2}{m^2\sigma^2}\right)-1\right)\right) \leq \frac{2\alpha m^2}{n^2}\left(\exp\left(\frac{8G^2}{m^2\sigma^2}\right)-1\right) \leq \frac{32\alpha G^2}{n^2\sigma^2}$ where the last inequality use the numeric inequality $\exp(x) \leq 1+2x$ when $x \leq 1.256$; this means that we need the following condition $\frac{8G^2}{m^2\sigma^2} \leq 1.256$ - we will revisit this condition later. The second point is that Theorem 9 in [BBG18] holds integer $\alpha \geq 2$, which only leaves us with $\alpha = 2$. In the subsequent part of the proof, we will need to take $\alpha \to 1$. This discrepancy can be accounted for by using the fact the α-Rènyi Divergence is non-decreasing for $\alpha \in [0, \infty]$ (see Theorem 3 in [VEH14]). Therefore the result holds for all $\alpha \leq 2$, and we can replace the upper bound to be $\frac{64G^2}{n^2\sigma^2}$ The third and final point is that even though the amplification result in [BBG18] is established under the neighbouring relation that one point is replaced between datasets, it can be shown that the same result holds (perhaps upto constants) when the neighbouring relation is add/delete one data-point; see Lemma 3, [ACG+16a] for example. We now use adaptive sequential composition property of Rènyi divergence (Proposition 1 in [Mir17]) which linearly accumulates the divergence across iterations, yielding that the Rènyi divergence between the iterates $(w_1, w_2, \ldots, w_T)$ and $(w'_1, w'_2, \ldots, w'_T)$ is bounded as, $D_\alpha((w_1, w_2, \ldots, w_T)\|(w'_1, w'_2, \ldots, w'_T)) \leq \frac{64TG^2}{n^2\sigma^2}$. An application of data-processing inequality gives us the same upper bound on the Rènyi divergence between the final iterates $\widehat{w}_S$ and $\widehat{w}'_S$. We now use the result that $\lim_{\alpha\to 1} D_\alpha(\widehat{w}_S\|\widehat{w}'_S) = D_{\mathrm{KL}}(\widehat{w}_S\|\widehat{w}'_S)$ where D_{KL} denotes the KL-divergence (see Theorem 5 in [VEH14]). Hence we get that $D_{\mathrm{KL}}(\widehat{w}_S\|\widehat{w}'_S) \leq \frac{64TG^2}{n^2\sigma^2}$. Finally, we use Pinsker's inequality to further lower bound the left hand side by total variation distance, which yields $\mathrm{TV}(\widehat{w}_S\|\widehat{w}'_S) \leq \sqrt{\frac{D_{\mathrm{KL}}(\widehat{w}_S\|\widehat{w}'_S)}{2}} \leq \frac{8\sqrt{T}G}{n\sigma}$. As remarked before, this is a uniform bound over all neighbouring datasets. Finally, as before, we trivially have

that $\mathrm{TV}(\widehat{w}_S \| \widehat{w}'_S) \leq 1$; therefore setting $\sigma = \frac{8\sqrt{T}G}{n\rho}$ gives us that the algorithm's output is $\min\{\rho, 1\}$ TV-stable.

We now proceed to the accuracy guarantee. This follows simply by guarantee of Accelerated SGD on smooth convex functions. We have already shown in Proposition 3 that the gradients computed by sub-sampling are unbiased and its variance bounded by $\frac{2G^2}{m}$. The mean-zero Gaussian noise added preserves unbiasedness but the variance is bounded as,

$$\begin{aligned}
\mathcal{V}^2 &= \mathbb{E}\left[\left\| \frac{\sum_{j\in b_t} \nabla \ell(\mathring{w}_t; z_j)}{m} + \theta_t - \nabla \widehat{L}(\mathring{w}_t; S) \right\|\right]^2 \\
&= \mathbb{E}\left[\left\| \frac{\sum_{j\in b_t} \nabla \ell(\mathring{w}_t; z_j)}{m} - \nabla \widehat{L}(\mathring{w}_t; S) \right\|\right]^2 + \mathbb{E}\left[\|\theta_t\|\right]^2 \\
&\leq \frac{2G^2}{m} + \sigma^2 d
\end{aligned}$$

We now use Theorem 2 from [Lan12] - they use notation $\{\beta_t\}_t$ and $\{\gamma_t\}_t$ for the step size schedule of Accelerated SGD and set $\beta_t = \frac{t+1}{2}$ and $\gamma_t = \frac{t+1}{2}\gamma$. Even though the updates of their A-SGD seem different than us, it can be verified that they are the same with $\alpha_t = \beta_{t+1}(1 - \beta_t^{-1}) = \frac{1-t}{t+2}$ with $\alpha_0 = 0$ and $\eta = \gamma$. Finally, using step-size $\eta \leq \frac{1}{2H}$, and appealing to Theorem 2 in [Lan12], we get,

$$\mathbb{E}\left[\widehat{L}(\widehat{w}_S; S) - \widehat{L}(w^*; S)\right] \leq O\left(T\eta\mathcal{V}^2 + \frac{D^2}{\eta T^2}\right) = O\left(\eta T\left(\frac{2G^2}{m} + \sigma^2 d\right) + \frac{D^2}{\eta T^2}\right)$$

Let $\widetilde{G}^2 = \frac{2G^2}{m} + \sigma^2 d$, balancing the trade-off in η gives us $\eta = \frac{D}{\widetilde{G}T^{3/2}}$. Therefore, setting $\eta = \min\left\{\frac{1}{2H}, \frac{D}{\widetilde{G}T^{3/2}}\right\}$ gives us

$$\begin{aligned}
\mathbb{E}\left[\widehat{L}(\widehat{w}_S; S) - \widehat{L}(w^*; S)\right] &\leq O\left(\frac{HD^2}{T^2} + \frac{\widetilde{G}D}{\sqrt{T}}\right) \leq O\left(\frac{HD^2}{T^2} + \frac{GD}{\sqrt{Tm}} + \frac{\sigma\sqrt{d}D}{\sqrt{T}}\right) \\
&\leq O\left(\frac{HD^2}{T^2} + \frac{GD}{\sqrt{Tm}} + \frac{GD\sqrt{d}}{n\rho}\right)
\end{aligned}$$

Finally, note that when using the amplification lemma, we arrived at the condition $\frac{8G^2}{m^2\sigma^2} \leq 1.256$. Substituting $\sigma = \frac{8\sqrt{T}G}{n\rho}$, this reduces to $\frac{(n\rho)^2}{8m^2T} \leq 1.256 \iff T \geq \frac{(n\rho)^2}{16m^2}$. □

Proof of Corollary 7. We start with the result in Proposition 1, and balance the two trade-offs: the first between the terms $\frac{GD}{\sqrt{mT}}$ and $\frac{GD\sqrt{d}}{\rho n}$, and the second between $\frac{GD\sqrt{d}}{\rho n}$ and $\frac{HD^2}{T^2}$. Note that as long as $\frac{GD}{\sqrt{mT}} \geq \frac{HD^2}{T^2} \iff m \leq \frac{T^3G^2}{(HD)^2}$, the second term is larger than the first. Optimizing the trade-off between second and third term gives us $\frac{GD}{\sqrt{mT}} = \frac{GD\sqrt{d}}{\rho n} \iff T = \frac{(\rho n)^2}{md}$. Similarly, optimizing the trade-off between the first and third term gives us $\frac{GD\sqrt{d}}{\rho n} = \frac{HD^2}{T^2} \iff T = \sqrt{\frac{HD(\rho n)}{G\sqrt{d}}}$. Hence setting $T = \max\left(\frac{(\rho n)^2}{md}, \sqrt{\frac{HD(\rho n)}{G\sqrt{d}}}\right)$ yields an expected excess empirical risk of $O\left(\frac{GD\sqrt{d}}{n\rho}\right)$.

We now look at the given condition $T \geq \frac{(n\rho)^2}{16m^2}$ given in Proposition 1. We have set $T = \max\left(\frac{(\rho n)^2}{md}, \sqrt{\frac{HD(\rho n)}{G\sqrt{d}}}\right)$, there we need to ensure that $\frac{(\rho n)^2}{md} \geq \frac{(\rho n)^2}{16m^2} \iff m \geq \frac{d}{16}$, as well as $\sqrt{\frac{HD(\rho n)}{G\sqrt{d}}} \geq \frac{(\rho n)^2}{16m^2} \iff m \geq \frac{1}{4}\left(\frac{(\rho n)^3 G\sqrt{d}}{HD}\right)^{1/4}$ - this recovers the condition $m \geq \min\left\{\frac{d}{16}, \frac{1}{4}\left(\frac{(\rho n)^3 G\sqrt{d}}{HD}\right)^{1/4}\right\}$ in the Proposition statement. Combining all the above arguments, we get that for any $m \geq \min\left\{\frac{d}{16}, \frac{1}{4}\left(\frac{(\rho n)^3 G\sqrt{d}}{HD}\right)^{1/4}\right\}$, setting $T = \max\left\{\frac{(\rho n)^2}{md}, \sqrt{\frac{HD(\rho n)}{G\sqrt{d}}}\right\}$, yields an expected excess empirical risk of $O\left(\frac{GD\sqrt{d}}{n\rho}\right)$. □

Remark 2. *Note that in the above proof, if we use the stronger variance bound of* $2H^2\left(\frac{1}{m} - \frac{1}{n}\right)$ *from sub-sampling (derived in the proof of Proposition 3), we get that when doing full-gradient descent, the variance, as expected is zero, which yields a running time of* $T = \sqrt{\frac{HD\rho n}{G\sqrt{d}}}$.

Corollary 7. *Let* $\ell(.; z)$ *be an* H*-smooth* G*-Lipschitz convex function* $\forall\ z$*. For any* $0 < \rho < \infty$*, Algorithm 9, run with* $t_0 = 1, m \geq \min\left\{\frac{d}{16}, \frac{1}{4}\left(\frac{(\rho n)^3 G\sqrt{d}}{HD}\right)^{1/4}\right\}$, $\eta = \min\left\{\frac{1}{2H}, \frac{D}{\left(\frac{G}{\sqrt{m}} + \sigma\right)T^{3/2}}\right\}$, $\alpha_0 = 0, \alpha_t = \frac{1-t}{t+2}$, $\sigma = \frac{8\sqrt{T}G}{n\rho}$, *and* $T = \max\left\{\frac{(\rho n)^2}{md}, \sqrt{\frac{HD\rho n}{G\sqrt{d}}}\right\}$ *outputs* $\widehat{w}_S$ *which is* $\min\{\rho, 1\}$*-TV stable and satisfies* $\mathbb{E}\left[\widehat{F}_S(\widehat{w}) - \widehat{F}_S(w_S^*)\right] = O\left(\frac{GD\sqrt{d}}{\rho n}\right)$.

Remark 3. *The choice of* T *in Corollary 7 yields that the largest mini-batch size that can be set, without hurting runtime, is* $m = \left(\frac{(\rho n)^3 G}{\sqrt{d}^3 HD}\right)^{1/2} = \left(\frac{G}{HD}\right)^2 T^3$. *Furthermore, the condition* $m \geq \min\left\{\frac{d}{16}, \frac{1}{4}\left(\frac{(\rho n)^3 G\sqrt{d}}{HD}\right)^{1/4}\right\}$ *yields* $(\rho n) \geq \left(\frac{HD(\sqrt{d})^7}{256G}\right)^{1/3}$.

Next we show that the upper bound on total variation stability parameter of Algorithm 9 derived in Proposition 1 is tight in all problem parameters, upto constants.

Proposition 5. *There exists neighbouring datasets S and S' of n points, and smooth G-Lipshcitz convex functions ℓ and constraint set $\mathcal{W}$ such that the total variation distance between iterates produced by Algorithm 9 run on datasets S and S', denoted by $\{w_1, w_2, \ldots, w_T\}$ and $\{w'_1, w'_2, \ldots, w'_T\}$ respectively, is bounded as* $TV((w_1, w_2, \ldots, w_T), (w'_1, w'_2, \ldots, w'_T)) \geq \min\left\{\Omega\left(\frac{G\sqrt{T}}{n\sigma}\right), 1\right\}$.

Proof of Proposition 5. We first prove this without projection - let the constraint set $\mathcal{W} = \mathbb{R}^d$, and so the projection $\mathcal{P}$ is the identity map. Also, for simplicity, let the initial model be 0. Consider data sets S and S' such that all points are 0 but the n^{th} differing point. Let the n^{th} point of S be $-G\mathrm{e}_1$ and that of S' be $G\mathrm{e}_1$, where e_1 is the first canonical basis vector. Let the function $\ell(w; z) = \langle w, z\rangle$. The gradients are just data points z, therefore gradients are 0 on all but the differing points, wherein in the differing point in dataset S, the gradient is a constant $-G\mathrm{e}_1$ and for dataset S', it is $G\mathrm{e}_1$. Consider the map $\Psi : (z_1, z_2, \ldots, z_T) \to z_T$; using data processing inequality and this map, we have that

$$\begin{aligned}
&\mathrm{TV}((w_1, w_2, \ldots, w_T), (w'_1, w'_2, \ldots, w'_T)) \\
&\geq \mathrm{TV}(\Psi(w_1, w_2, \ldots, w_T), \Psi(w'_1, w'_2, \ldots, w'_T)) \\
&= \mathrm{TV}(w_T, w'_T)
\end{aligned}$$

We now focus on bounding the total variation distance between the last iterates. Furthermore, by data-processing inequality, we can get rid of the step size scaling, and therefore can consider the last iterates as just the sum of all gradients. By simple calculations, we get that w_T is a mixture of multivariate Gaussians, all with variance $T\sigma^2\mathbb{I}$ but with varying means: $G\mathrm{e}_1, 2G\mathrm{e}_1, \ldots, TG\mathrm{e}_1$, similarly for w'_T. We denote the mixtures probabilities by π_i where the i^{th} conditional distribution, denoted by w_T^i

and $w_T'^i$ respectively, has means $iG\mathrm{e}_1$ and $-iG\mathrm{e}_1$ respectively. Also, we denote the conditional probability densities of the i^{th} distribution by $\phi_S^i(w)$ and $\phi_{S'}^i(w)$ respectively. We will show that the total variation between these mixtures is expected total variation distance between the mixture components. This follows due the symmetry between these two mixtures, which implies that the set that achieves the total variation distance is $\{w : w_1 \geq 0\}$. We can therefore write the total variation distance as,

$$\begin{aligned}
&\mathrm{TV}(w_T \| w_T') = \frac{1}{2} \left\| \phi_S(w) - \phi_{S'}(w) \right\|_1 = \int_{w_1 \geq 0} \phi_S(w) - \phi_{S'}(w) dw \\
&= \int_{w_1 \geq 0} \sum_i \pi_i (\phi_S^i(w) - \phi_{S'}^j(w)) dw = \sum_i \pi_i \int_{w \geq 0} (\phi_S^i(w) - \phi_{S'}^j(w)) dw \\
&= \sum_i \pi_i \mathrm{TV}(w_T^i, w_T'^i) = \Omega\left(\sum_i \pi_i \frac{2Gi}{m\sqrt{T}\sigma} \right) = \Omega\left(\frac{2G}{m\sqrt{T}\sigma} \mathbb{E}\left[i\right] \right) = \Omega\left(\frac{2G\sqrt{T}}{n\sigma} \right)
\end{aligned}$$

where in the inequality, we use the fact that w_T^i and $w_T'^i$ are Gaussians with means separated by $2G$, and variance being $T\sigma^2\mathbb{I}$ and use the lower bound result on TV between high-dimensional Gaussians [DMR18]. Finally, in the last equality, we compute the Expected value of i under the mixture distribution - recall that i is a sum of T Bernoulli random variables with bias $\frac{m}{n}$, the expectation of which is $\frac{Tm}{n}$.

We now argue why projection doesn't change the above claim. Note the with the projection, all the Gaussians in the mixture are truncated forming a discrete distributions at the boundary of the constraint set. The probability mass on either sides of the (original) mean is unchanged. Hence $\{w : w_1 \geq 0\}$ is still the witness set of total variation distance between the mixtures, and the total variation distance in both constrained/unconstrained cases is the same. The same holds for the total variation between the corresponding mixture components. These observations suffices for application of proof of the unconstrained case. Finally, since TV distance, by definition is upper bounded by 1 - this gives a trivial lower bound of 1, and hence the TV distance is lower bound by $\min\left\{\Omega\left(\frac{G\sqrt{T}}{n\rho}\right), 1\right\}$. □

C.6 Proofs for Section 4.5.2

We introduce some notation and setup the roadmap. In the start of the stream, we have a model trained on the initial dataset of n samples. We then observe an insertion or deletion request. We enumerate the data points from 1 to n, and without loss of generality, assume that the n^{th} sample is to be deleted, and the inserted sample has index $n+1$. We want to show that the unlearning algorithm satisfies exact unlearning at every time point in the stream, and what suffices is to argue that this holds for one edit request, since by mathematical induction it then holds for the entire stream. For one edit request, we will show the following: 1. unlearning (deletion/insertion) algorithm is a valid transport, and 2. the probability of recompute is small, and we will see that together these will imply, that it is a coupling, with large enough measure of the diagonal.

Let $\mu_{n,m}$ denote the sub-sampling probability measure to sample m out of n elements uniformly randomly. In the deletion and insertion algorithms, we replace *some* mini-batch indices in *some* iterations - let these operations be denoted by DEL and INS respectively. To elaborate, DEL is a (deterministic) map from $([n]^m)^T$ to $([n-1]^m)^T$ and INS is a map from $([n]^m)^T$ to $([n+1]^m)^T$. For an input $b \in ([n]^m)^T$, we have that $\mathbf{b} \sim \mu_{n,m}^{\otimes T}$. Furthermore, define $\mu_{n,m}^{\text{del}\otimes T} := \text{DEL}_{\#}\mu_{n,m}^{\otimes T}$ and $\mu_{n,m}^{\text{ins}\otimes T} := \text{INS}_{\#}\mu_{n,m}^{\otimes T}$. An important observation is that in the unlearning Algorithm 20, the sub-sampled indices $\mathbf{b}$ are drawn from a product distribution $\mu_{n,m}^{\otimes T}$ and in each iteration of Algorithm 20 or Algorithm 10, the maps DEL and INS act *component-wise* and *symmetrically*. This implies that $\text{DEL}(\mathbf{b}) = [\text{del}(b_1), \text{del}(b_2), \ldots, \text{del}(b_T)]$ where $\text{del} : [n]^m \to [n-1]^m$ is the function which describes one iteration of the unlearning algorithm for handling mini-batch indices. We similarly have function $\text{ins} : [n]^m \to [n+1]^m$ for insertion. We finally define $\mu_{n,m}^{\text{del}} := \text{del}_{\#}\mu_{n,m}$ and $\mu_{n,m}^{\text{ins}} := \text{ins}_{\#}\mu_{n,m}$ - these are the probability measures induced on the sub-sampling indices by deletion and insertion operations,

respectively.

C.6.1 Unlearning for *sub-sample-GD*

We first show that $\mu^{\text{del}}_{n,m}$, the probability distribution, induced at a given iteration during deletion, over mini-batch indices $b \in [n]^m$ is a transport.

Claim 1 (Deletion)**.** *For any set $b \in [n]^m$, we have that $\mu^{del}_{n,m}(b) = \mu_{n-1,m}(b)$*

Proof of Claim 1. First note that if the verification is unsuccessful, then a recompute is triggered and therein at each iteration, we drawn $b \sim \mu_{n-1,m}$. Therefore, $\mu^{\text{del}}_{n,m}(b) = \mu_{n-1,m}(b)$ follows trivially. We now argue for the other case. The verification is successful if the deleted point was not present in any of iterations, i.e. at any iteration the sub-sample batch b_t doesn't contain the deleted point z. The measure $\mu^{\text{del}}_{n,m}$ is therefore just the probability under the original sub-sampling measure $\mu_{n,m}$ conditioned on the event that $z \notin b$. We therefore have,

$$\mu^{\text{del}}_{n,m}(b) = \mu_{n,m}(b|\,\{z \notin b\}) = \frac{\mu_{n,m}(b \cap \{z \notin b\})}{\mu_{n,m}(\{z \notin b\})}$$

By direct computation, $\mu_{n,m}(\{z \notin b\}) = 1 - \mu_{n,m}(\{z \in b\}) = 1 - \frac{\binom{n-1}{m-1}}{\binom{n}{m}} = 1 - \frac{m}{n}$. We now look at two choices for b. First suppose $z \in b$, then the numerator $\mu_{n,m}(b \cap \{z \notin b\}) = 0$, which gives us that $\mu^{\text{del}}_{n,m}(b) = 0 = \mu_{n-1,m}(b)$. We now look at a b such that $z \notin b$. We have,

$$\begin{aligned}\mu^{\text{del}}_{n,m}(b) &= \frac{\mu_{n,m}(b)}{\mu_{n,m}(\{z \notin b\})} = \frac{1/\binom{n}{m}}{1 - m/n} = \frac{n}{n-m}\frac{(n-m)!m!}{n!} \\ &= \frac{(n-m-1)!m!}{(n-1)!} = \frac{1}{\binom{n-1}{m}} = \mu_{n-1,m}(b)\end{aligned}$$

□

Similarly, for insertion, we show that μ^{ins}_{n}, the probability distribution, induced at a given iteration during insertion, over mini-batch indices $b \in [n]^m$, is a is valid transport.

Claim 2. *For any set $b \in [n+1]^m$, we have that $\mu^{ins}_{n,m}(b) = \mu_{n+1,m}(b)$*

Proof of Claim 2. Let ν denote the uniform probability measure over $n+1-m$ elements. Given b, we consider two cases based of whether last/inserted index $n+1$ lies in b or not. In the first case, we know that the outcome of Bernoulli$(m/(n+1))$ must have been 1 i.e. the iteration was selected. Furthermore, in that case, the inserted point would have replaced some other point not in b - the total number of possibilities are $n+1-m$. Let E_i be event that the inserted point replaced the i^{th} data point, whose index we denote by s_i. Note that the events $E_i's$ are disjoint and the event b is $\cup_{i=1}^{n+1-m} E_i$. Furthermore, $\mu^{\text{ins}}_{n,m}(E_i) = \mu^{\text{ins}}_{n,m}(\text{original subsample is } b\setminus\{n+1\}\cup\{s_i\} \,|\, \{s_i\} \text{ replaced})\mu^{\text{ins}}_{n,m}(\{s_i\}\text{ replaced})) = \mu_{n,m}(b\setminus\{n+1\}\cup\{s_i\}\,|\,\{s_i\})\nu(\{s_i\}) = \frac{1}{\binom{n}{m-1}}\frac{1}{(n+1-m)}$. We therefore have that

$$\begin{aligned}
\mu^{ins}_n(b) &= \frac{m}{n+1}\mu^{ins}_n(\cup_{i=1}^{n+1-m}E_i) = \frac{m}{n+1}\sum_{i=1}^{n+1-m}\mu^{ins}_n(E_i) \\
&= \frac{m}{n+1}\sum_{i=1}^{n+1-m}\frac{1}{\binom{n}{m-1}}\frac{1}{(n+1-m)} = \frac{m}{n+1}\frac{1}{\binom{n}{m-1}} = \frac{m(m-1)!(n-(m-1))!}{(n+1)n!} \\
&= \frac{1}{\binom{n+1}{m}} = \mu_{n+1,m}(b)
\end{aligned}$$

In the other case, we know that Bernoulli$(m/(n+1))$ resulted in 0, so there is no replacement. Therefore, we have

$$\mu^{ins}_n(b) = \left(1-\frac{m}{n+1}\right)\frac{1}{\binom{n}{m}} = \frac{(n+1-m)(n-m)!m!}{n!(n+1)} = \frac{1}{\binom{n+1}{m}} = \mu_{n+1,m}(b)$$

□

Coupling. We formally describe the coupling constructed by the unlearning Algorithm 20. We first the discuss deletion case - consider datasets S and S' of sizes n and $n-1$ respectively, and wlog assume that the last sample of S differs. We first sample $\mathbf{b} = [b_1, b_2, \ldots, b_T] \sim \mu^{\otimes T}_{n,m}$. We set $\mathbf{b}^{(1)} = \mathbf{b}$. For each $j \in T$, if $n \in b_j$, then

sample $b_j^{(2)} \sim \mu_{n-1,m}$, otherwise set $b_j^{(2)} = b_j$. This produces the coupled mini-batches $(\mathbf{b}^{(1)}, \mathbf{b}^{(2)})$ for deletion.

For insertion, we have datasets S and S' of sizes n and $n+1$ respectively, and again assume that the last of point of S' differs. Sample $\mathbf{b} = [b_1, b_2, \ldots, b_T] \sim \mu_{n,m}^{\otimes T}$. and set $\mathbf{b}^{(1)} = \mathbf{b}$. Now sample $\{c_j\}_{j=1}^T$, where $c_j \sim \text{Bernoulli}\left(\frac{m}{n}\right)$, if $c_j = 1$, then sample uniformly a point in $b_j^{(2)}$, and replace it with $n+1$. Otherwise set $b_j^{(2)} = b_j$, which gives us the coupled mini-batches $(\mathbf{b}^{(1)}, \mathbf{b}^{(2)})$.

It is easy to see that the above procedure is how Algorithm 20 handles insertions and deletions going from S to S'. We first show that this is a valid coupling.

Claim 3. *For the coupling described above, for any b,*

1. $\mathbb{P}\left[\mathbf{b}^{(1)} = b\right] = \mu_n^{\otimes T}(b)$
2. $\mathbb{P}\left[\mathbf{b}^{(2)} = b\right] = \mu_{n-1}^{\otimes T}(b)$ *(deletion),* $\mathbb{P}\left[\mathbf{b}^{(2)} = b\right] = \mu_{n+1}^{\otimes T}(b)$ *(insertion)*

Proof of Claim 3. Follows immediately from Claims 1 and 2 . □

We now show that the probability of disagreement under the above coupling is upper bounded by k times TV-stability parameter of Algorithm 19.

Claim 4. *For the ρ-TV stable Algorithm 19, under the coupling described above, the following holds*

$$\mathbb{P}_{(\mathbf{b}^{(1)}, \mathbf{b}^{(2)})}[\mathbf{b}^{(1)} \neq \mathbf{b}^{(2)}] \leq \rho$$

Proof. For deletion, we have,

$$\begin{aligned}
\mathbb{P}_{(\mathbf{b}^{(1)}, \mathbf{b}^{(2)})}[\mathbf{b}^{(1)} \neq \mathbf{b}^{(2)}] &= \mathbb{P}_{(\mathbf{b}^{(1)}, \mathbf{b}^{(2)})}[\exists j \in [T] : \mathbf{b}_j^{(1)} \neq \mathbf{b}_j^{(2)}] \\
&= \mathbb{P}_{\mathbf{b}}[\exists j \in [T] : n \in \mathbf{b}_j] \\
&\leq \frac{Tm}{n}
\end{aligned}$$

For insertion, we have

$$\mathbb{P}_{(\mathbf{b}^{(1)},\mathbf{b}^{(2)})}[\mathbf{b}^{(1)} \neq \mathbf{b}^{(2)}] = \mathbb{P}_{\mathbf{b},\mathbf{c}}[\exists j \in [T] : c_j = 1] \leq \frac{Tm}{n}$$

In Proposition 3, we showed that the total variation distance of the algorithm under change of one point is at most $\frac{Tm}{n} = \rho$, which completes the proof. □

We are now ready to prove Proposition 4.

Proof of Proposition 4. The following argument is for deletion, but the insertion case follows similarly. Consider dataset S and S' of n points and $n-1$ points respectively, differing in one sample. As in the proof of Proposition 3, we embed the randomness for Algorithm 19 executed on S and S' into a common probability space. Therefore, similar to the proof of Proposition 3 given the datasets (and other parameters), Algorithm 19, $\mathcal{A}(S)$ is a deterministic map from sub-sampled indices $\mathbf{b} = (b_1, b_2, \ldots, \mathrm{b}_T)$ to the model: $\mathcal{A}(S) : \mathbf{b} \to \mathcal{W}$, where $b_j \in [n]^m$, for both datasets. Hence, what suffices is to show that the input probability measure $\mu_{n,m}^{\otimes T}$ is transported to the one that would have been produced on the current dataset S' i.e $\mu_{n-1,m}^{\otimes T}$ - this follows from Claim 3. Hence it follows that the output generated by Algorithm 19 has the same measure as $\mathcal{A}(S)_{\#}\mu_{n-1,m}^{\otimes T}$, which proves first part of the claim. The probability of recompute, being at most ρ, for one edit, follows directly from 4. Finally, from Remark 1, for k edits, and the assumption the number of samples throughout the stream is between $n/2$ and $2n$, the recompute probability is at most $2k\rho$. □

C.6.2 Unlearning for *noisy-m-A-SGD*

C.6.2.1 Coupling mini-batches

In this section, we show that Algorithm 10 transports sub-sampling probability measures while handling edit requests. We remind that $\mu_{n,m}^{\mathrm{del}}$ denotes the probability measure induced on the sub-sampled indices by the deletion procedure, in any iteration.

We show that, for any mini-batch, the probability mass of the mini-batched indices under $\mu^{\text{del}}_{n,m}$ is same as that under the sub-sampling measure $\mu_{n-1,m}$.

Claim 5. *For any set $b \in [n-1]^m$, we have that $\mu^{del}_{n,m}(b) = \mu_{n-1,m}(b)$*

Proof of Claim 5. Firstly, note that deletion uses additional randomness which is used to uniformly sample one element from $n-(m-1)$ elements - let ν denote the uniform probability measure on $n-(m-1)$ elements. Let E be the event that the n^{th} was sub-sampled originally, and therefore replaced upon verification. By direct computation $\mu_{n,m}(E) = \frac{m}{n}$. We can therefore write $\mu^{\text{del}}_{n,m}(b)$ as follows

$$\mu^{\text{del}}_{n,m}(b) = \mu^{\text{del}}_{n,m}(b|E)\mu_{n,m}(E) + \mu^{\text{del}}_{n,m}(b|E^c)\mu_{n,m}(E^c)$$

Under event E, we have that the deleted index was replaced. But it can be any element of b that rose out of this replacement. Hence we decompose the event $b|E$ into events E_i's, where E_i corresponds to the event that b_i was replaced. We have that $b|E = \cup_{i=1}^m E_i$, and furthermore, due to the uniform measure, $\mu^{\text{del}}_{n,m}(E_i) = \mu^{\text{del}}_{n,m}(E_j) \forall i,j$. Note that in the event E_i, we require that the original sub-sampling measure on n points $\mu_{n,m}$ to have produced the set $b \backslash b_i \cup \{n\}$ and then a uniform b_i is drawn upon replacement. Therefore, $\mu^{\text{del}}_{n,m}(E_i) = \mu_{n,m}(b\backslash b_i \cup \{n\})\nu(b_i) = \frac{1}{\binom{n-1}{m-1}} \frac{1}{n-1-(m-1)}$. Similarly, when the event E^c occurs, probability of outputting b corresponds to the event when b was generated using the original sub-sampling measure μ_m (and no additional randomness used upon verification). Therefore, we get $\mu^{\text{del}}_{n,m}(b|E^c) = \mu_{n,m}(b|E^c) =$

$\frac{1}{\binom{n-1}{m}}$. Plugging these in, and with simple calculations, we have

$$\begin{aligned}
\mu^{\text{del}}_{n,m}(b) &= \sum_{i=1}^{m} \mu^{\text{del}}_{n,m}(E_i)\mu_{n,m}(E_i) + \mu^{\text{del}}_{n,m}(b|E^c)\mu_{n,m}(E) \\
&= \sum_{i=1}^{m} \mu_{n,m}(b\backslash b_i \cup \{n\})\nu(b_i)\frac{m}{n} + \frac{1}{\binom{n-1}{m}}\left(1 - \frac{m}{n}\right) \\
&= \frac{m}{\binom{n-1}{m-1}}\frac{1}{n-1-(m-1)}\frac{m}{n} + \frac{1}{\binom{n-1}{m}}\left(1 - \frac{m}{n}\right) \\
&= \frac{1}{\binom{n-1}{m}} + \frac{m}{n}\left(\frac{m}{\binom{n-1}{m-1}(n-1-(m-1))} - \frac{1}{\binom{n-1}{m}}\right) \\
&= \frac{1}{\binom{n-1}{m}} + \frac{m}{n}\left(\frac{m(m-1)!(n-1-(m-1))!}{(n-1)!(n-1-(m-1))} - \frac{1}{\binom{n-1}{m}}\right) \\
&= \frac{1}{\binom{n-1}{m}} + \frac{m}{n}\left(\frac{1}{\binom{n-1}{m}} - \frac{1}{\binom{n-1}{m}}\right) = \frac{1}{\binom{n-1}{m}} = \mu_{n-1,m}(b)
\end{aligned}$$

□

Similarly, for insertion, we now show that the probability mass of any mini-batch under $\mu^{\text{ins}}_{n,m}$, the probability measure induced by insertion on the $n+1$ data points, is same as that under $\mu_{n+1,m}$.

Claim 6. *For any set $b \in [n+1]^m$, we have that $\mu^{ins}_{n,m}(b) = \mu_{n+1,m}(b)$.*

Proof of Claim 6. Same as that of Claim 2. □

C.6.2.2 Lemmas for reflection coupling

We state and prove some results about reflection mapping and couplings.

Lemma 26. *Let P and Q be probability distributions over $\mathbb{R}^d$. Let $\psi : \mathbb{R}^d \to \mathbb{R}^d$ be a bijection such that $\phi_P(\psi(x)) = \phi_Q(x)$, $\phi_P(\psi^{-1}(x)) = \phi_Q(x)$ and $\left|\det\left(\frac{d\psi(x)}{dx}\right)\right| = 1$, where $\frac{d\psi(x)}{dx}$ is the Jacobian of the multivariate map ψ. Let $x \sim P$ be a sample from P. Let $y = x$ if Unif$(0,1) \leq \frac{\phi_Q(x)}{\phi_P(x)}$, otherwise $y = \psi(x)$. Then (x, y) is a maximal coupling of P and Q.*

Proof. We first show that y is a sample from Q. Let E be an event in the range of Q. Let accept be the event when $u \sim \text{Unif}(0,1)$, $u \leq \frac{\phi_Q(x)}{\phi_P(x)}$. We have,

$$\begin{aligned}
&\mathbb{P}\left[y \in E\right] = \mathbb{P}\left[y \in E, \text{accept}\right] + \mathbb{P}\left[y \in E, \text{reject}\right] \\
&= \mathbb{E}_{x,u}\left[\mathbb{1}\left\{x \in E\right\} \mathbb{1}\left\{u \leq \frac{\phi_Q(x)}{\phi_P(x)}\right\}\right] + \mathbb{E}_{x,u}\left[\mathbb{1}\left\{\psi(x) \in E\right\} \mathbb{1}\left\{u > \frac{\phi_Q(x)}{\phi_P(x)}\right\}\right] \\
&= \mathbb{E}_x\left[\mathbb{1}\left\{x \in E\right\} \mathbb{P}\left[\left\{u \leq \frac{\phi_Q(x)}{\phi_P(x)}\right\} \middle| x\right]\right] \\
&+ \mathbb{E}_x\left[\mathbb{1}\left\{\psi(x) \in E\right\} \mathbb{P}\left[\left\{u > \frac{\phi_Q(x)}{\phi_P(x)}\right\} \middle| x\right]\right] \\
&= \int_{\mathbb{R}^d} \mathbb{1}\left\{x \in E\right\} \min\left\{1, \frac{\phi_Q(x)}{\phi_P(x)}\right\} \phi_P(x) dx \\
&+ \int_{\mathbb{R}^d} \mathbb{1}\left\{\psi(x) \in E\right\} \left(1 - \min\left\{1, \frac{\phi_Q(x)}{\phi_P(x)}\right\}\right) \phi_P(x) dx \\
&= \int_{\mathbb{R}^d} \mathbb{1}\left\{x \in E\right\} \min\left\{\phi_P(x), \phi_Q(x)\right\} dx \\
&+ \int_{\mathbb{R}^d} \mathbb{1}\left\{\psi(x) \in E\right\} \max\left\{0, \phi_P(x) - \phi_Q(x)\right\} dx
\end{aligned}$$

For the second term, we now do change of variable - let $v = \phi(x)$ - using the given properties of ψ, we have $\phi_P(x) = \phi_P(\psi^{-1}(v)) = \phi_Q(v)$ and $\phi_Q(x) = \phi_P(v)$. Furthermore $dv = \left|\det\left(\frac{d\psi(x)}{dx}\right)\right| dx = dx$. Finally, we are integrating over $\mathbb{R}^d$, and since ϕ is a bijection, it can flip the limits of some of the coordinates, however, that is taken into account with using the absolute value of the determinant of the Jacobian. The second term therefore becomes $\int_{\mathbb{R}^d} \mathbb{1}\left\{v \in E\right\} \max\left\{0, \phi_Q(v) - \phi_P(v)\right\} dv$. We now combine the integrands of both the terms, and substitute $v = x$ as the variable in the second term. This gives us,

$$\mathbb{P}\left[y \in E\right] = \int_{\mathbb{R}^d} \mathbb{1}\left\{x \in E\right\} \left(\min\left\{\phi_P(x), \phi_Q(x)\right\} + \max\left\{0, \phi_Q(x) - \phi_P(x)\right\}\right) dx$$

Note that for a fixed x, if $\phi_P(x) \leq \phi_Q(x)$, the integrand becomes $\mathbb{1}\left\{x \in E\right\} \left(\phi_P(x) + \phi_Q(x) - \phi_P(x)\right) = \mathbb{1}\left\{x \in E\right\} Q(x)$. On the other hand, if $\phi_P(x) > \phi_Q(x)$, the integrand becomes $\mathbb{1}\left\{x \in E\right\} \phi_Q(x)$. Hence, for all cases, we get that,

$$\mathbb{P}\left[y \in E\right] = \int_{\mathbb{R}^d} \mathbb{1}\left\{x \in E\right\} \phi_Q(x) dx = Q(E)$$

We now show that it is a maximal coupling i.e. the probability of accept is $1 - \mathrm{TV}(P, Q)$. We have,

$$\begin{aligned}\mathbb{P}\left[\text{accept}\right] &= \mathbb{E}_{x,u}\left[\mathbb{1}\left\{u \leq \frac{\phi_Q(x)}{\phi_P(x)}\right\}\right] = \int_{\mathbb{R}^d} \min\left\{1, \frac{\phi_Q(x)}{\phi_P(x)}\right\}\phi_P(x)dx \\ &= \int_{\mathbb{R}^d} \min\left\{\phi_P(x), \phi_Q(x)\right\} dx = 1 - \mathrm{TV}(P, Q)\end{aligned}$$

□

Lemma 27. *Let P and Q be two isotropic probability distributions over $\mathbb{R}^d$ with means μ_P and μ_Q such that for any vectors x, y, $\phi_P(\mathrm{x}) = \phi_Q(\mathrm{y})$ if $\|\mathrm{x} - \mu_P\| = \|\mathrm{y} - \mu_Q\|$. Given vector u in $\mathbb{R}^d$, the reflection of u under (Q, P), $\mathrm{v} = \mathit{reflect}(\mathrm{u}, \mu_Q, \mu_P) = \mu_Q + (\mu_P - \mathrm{u})$, satisfies:*

1. *Invertibility:* $\mathrm{u} = \mathrm{u}_Q + (\mu_P - \mathrm{v})$
2. $\phi_Q(\mathrm{v}) = \phi_P(\mathrm{u})$ *and* $\phi_P(\mathrm{v}) = \phi_Q(\mathrm{u})$
3. $\left|det\left(\frac{d\ \mathit{reflect}(\mathrm{u},\mu_Q,\mu_P)}{d\mathrm{u}}\right)\right| = 1$

Proof of Lemma 27. The proofs follows immediately using the given assumptions. □

C.6.2.3 Coupling Markov chains

We setup some notation to describe the coupling that Algorithm 10 constructs. The following discussion is for deletion of index n, but it can be verified that the arguments naturally extend to the insertion case. We remind that $\mu_{n,m}$ denotes the distribution of sampling m elements uniformly randomly from $[n]$, and mini-batches $b_j \sim \mu_{n,m}$. Furthermore, we will use $\mathbf{b}_j = [b_1, b_2, \ldots, b_j]$ denote the set of indices upto j. For dataset S and mini-batch indices b, let the gradient $\nabla\widehat{L}(w, z_b; S) := \frac{1}{|b|}\sum_{j\in b}\nabla\ell(w; z_j)$. Define $\widetilde{w}_{j+1} := \mathring{w}_j - \eta\nabla\widehat{L}(\mathring{w}_j; z_{b_j})$, $\bar{w}_{j+1} := \widetilde{w}_{j+1} - \eta\theta_j$ and $w_{j+1} := \mathcal{P}(\bar{w}_{j+1})$. Note that $\widetilde{w}_j$ is also function of b_j but this dependency is not highlighted for notational simplicity.

The iterates and the mini-batches $[(\bar{w}_2, b_1), (\bar{w}_3, b_2), \ldots, (\bar{w}_{T+1}, b_T)]$ produced by Algorithm 9 is a sample from a T-step first order Markov Chain over an uncountable state space $\mathbb{R}^d \times [n]^*$. We remark that $\bar{w}_1$ is a constant initialization, and so isn't considered. Let P be the joint distribution over the T iterates $\times$ mini-batches. The joint density of P can be factored as,

$$\phi_P((\bar{w}_2, b_1), (\bar{w}_3, b_2), \ldots, (\bar{w}_{T+1}, b_T)) = \phi_P(\bar{w}_2, b_1)\phi_P(\bar{w}_3, b_2|w_2)\ldots \\ \cdot \phi_P(\bar{w}_{T+1}, b_T|w_T, w_{T-1})$$

where $\phi_P(\bar{w}_2, b_1) = \bar{\phi}_P(\bar{w}_2|b_1)\mu_{n,m}(b_1)$ and $\bar{\phi}_P(\bar{w}_2|b_1)$ is the density of $\mathcal{N}(\widetilde{w}_2, \eta^2\sigma^2\mathbb{I})$. Similarly, the conditionals $\phi_P(\bar{w}_j, b_{j-1}|w_{j-1}, w_{j-2}) = \bar{\phi}_P(\bar{w}_j|b_{j-1}, w_{j-1}, w_{j-2})\mu_{n,m}(b_{j-1})$. Furthermore, let $\widetilde{P}$ denote the marginal of $[\bar{w}_2, \bar{w}_2, \ldots, \bar{w}_{T+1}]$, the joint density of which can be factored as,

$$\widetilde{\phi}_{\widetilde{P}}(\bar{w}_2, \bar{w}_3, \ldots, \bar{w}_{T+1}) = \phi_{\widetilde{P}}(\bar{w}_2)\phi_{\widetilde{P}}(\bar{w}_3|w_2)\ldots\phi_{\widetilde{P}}(\bar{w}_{T+1}|w_T, w_{T-1})$$

where $\phi_{\widetilde{P}}(\bar{w}_2) = \mathbb{E}_{b_1}\phi_P(\bar{w}_2, b_1)$, and the conditional $\phi_{\widetilde{P}}(\bar{w}_j|w_{j-1}, w_{j-2}) = \mathbb{E}_{b_{j-1}}\bar{\phi}_P(\bar{w}_j, b_{j-1}|w_{j-1}, w_{j-2})$. Finally, given a fixed mini-batch sequence $\mathbf{b} = \{b_1, b_2, \ldots, b_T\}$, let $P_{\mathbf{b}}$ denote the joint conditional distribution of $\{\bar{w}_2, \bar{w}_3, \ldots, \bar{w}_{T+1}\}$ given $\mathbf{b}$. In this case, $P_{\mathbf{b}}$ factorizes as:

$$\phi_{P_{\mathbf{b}}}(\bar{w}_2, \bar{w}_3, \ldots, \bar{w}_{T+1}) = \phi_{P_{b_1}}(\bar{w}_2)\phi_{P_{b_2}}(\bar{w}_3|w_2)\ldots\phi_{P_{b_T}}(\bar{w}_{T+1}|w_T, w_{T-1})$$

where $\phi_{P_{b_1}}(\bar{w}_2) = \bar{\phi}_P(w_2|b_1)$ and $\phi_{P_{b_{j-1}}}(\bar{w}_j|w_{j-1}, w_{j-2}) = \bar{\phi}_P(\bar{w}_j|b_{j-1}, w_{j-1}, w_{j-2})$. We similarly have a Markov Chain to generate the iterates for dataset S' - call this joint distribution over iterates and mini-batches as Q, the marginals over iterates as $\widetilde{Q}$ and for a given $\mathbf{b} \sim \{\mu_{n-1,m}\}^{\otimes T}$, the conditionals over the iterates as $Q_{\mathbf{b}}$.

We now describe how the unlearning Algorithm 10 constructs a coupling between P and Q to generate $(\bar{\mathbf{w}}^{(1)}, \bar{\mathbf{w}}^{(2)})$. We first describe the coupling of mini-batch indices. Sample $\mathbf{b} \sim (\mu_n^m)^{\otimes T}$, let $\mathbf{b}^{(1)} = \mathbf{b}$. We now look at all $b_j^{(1)} \in \mathbf{b}^{(1)}$: if $n \notin b_j^{(1)}$, then let $b_j^{(2)} = b_j^{(1)}$, otherwise for each such $b_j^{(1)}$, we replace n by randomly

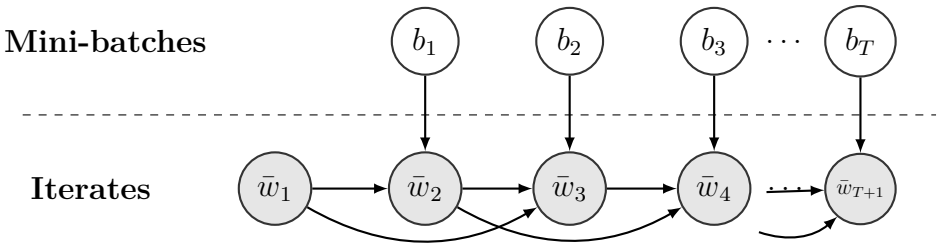

Figure C-1. Markov chain for *noisy-m-A-SGD* Algorithm

sampling an index from $[n]\backslash b_j^{(1)}$, and call this $b_j^{(2)}$. We then define the ordered set $\mathbf{b}^{(2)} = \left\{b_j^{(2)}\right\}_{j=1}^{T}$. From Claim 5, this is a valid coupling of mini-batch indices. Sample $\bar{\mathbf{w}} = [\bar{w}_2, \bar{w}_3, \ldots, \bar{w}_{T+1}] \sim P_{\mathbf{b}^{(1)}}$, which corresponds to training with Algorithm 9 on dataset S. Set $\bar{\mathbf{w}}^{(1)} := \bar{\mathbf{w}}$. To generate $\bar{\mathbf{w}}^{(2)}$, we do rejection sampling steps at each iteration. At the first step, we sample $u_1 \sim \text{Unif}(0,1)$, and check if $u_1 \leq \frac{\phi_{Q_{\mathbf{b}^{(2)}}}(\bar{w}_2)}{\phi_{P_{\mathbf{b}^{(1)}}}(\bar{w}_2)}$. If the step succeeds, then we proceed to the second iteration, wherein we again do a step of rejection sampling with ratio of conditional densities and so on. However, if anyone of the rejection sampling step fails, lets say the t^{th} step, then we do a *reflection* of iterate $\bar{w}_{t+1}$ about the mid-point of the means of $P_{\mathbf{b}^{(1)}}(\cdot|w_t, w_{t-1})$ and $Q_{\mathbf{b}^{(2)}}(\cdot|w_t, w_{t-1})$, which are $\widetilde{w}_{t+1}^{(1)} = w_t - \eta\nabla\widehat{L}(w_t; z_{b_t})$ and $\widetilde{w}_{t+1}^{(2)} = w_t - \eta\nabla\widehat{L}(w_t; z_{b'_t})$ respectively. Set $\bar{w}_{t+1}^{(2)} = \text{reflect}(\bar{w}_{t+1}, \widetilde{w}_{t+1}^{(2)}, \widetilde{w}_{t+1}^{(1)})$. After the reflection, we continue training on dataset S' which corresponds to continue sampling from the $(t+1)^{\text{th}}$ step of the Markov chain for $Q_{\mathbf{b}^{(2)}}$ conditioned on the t^{th} sample being $\bar{w}_t^{(2)}$. This generates the random variables $\bar{\mathbf{w}}^{(1)}$ and $\bar{\mathbf{w}}^{(2)}$.

We now show that this is indeed a coupling.

Lemma 28. *For any measurable set* $E \subseteq \mathbb{R}^{dT}$, $\mathbb{P}\left[\bar{\mathbf{w}}^{(2)} \in E\right] = \widetilde{Q}(E)$

Proof of Lemma 28. We will first show that $\mathbb{P}\left[\bar{\mathbf{w}}^{(2)} \in E|(\mathbf{b}^{(1)}, \mathbf{b}^{(2)})\right] = Q_{\mathbf{b}^{(2)}}(E)$. The proof is based on induction on the length of the Markov chain T. Define $\bar{\mathbf{w}}_T^{(2)} := \left\{\bar{w}_2^{(2)}, \cdots, \bar{w}_{T-1}^{(2)}\right\}$. The key to the proof is the observation that the marginals $P_{b_1^{(1)}}(\cdot)$ and $Q_{b_1^{(2)}}(\cdot)$ are Gaussian $\mathcal{N}(\widetilde{w}_2^{(1)}, \eta^2\sigma^2\mathbb{I})$ and $\mathcal{N}(\widetilde{w}_2^{(2)}, \eta^2\sigma^2\mathbb{I})$ respectively, and the conditionals $P_{b_j^{(1)}}(\cdot|w_j)$ and $Q_{b_j^{(2)}}(\cdot|w_j)$ are also Gaussian $\mathcal{N}(\widetilde{w}_{j+1}^{(1)}, \eta^2\sigma^2\mathbb{I})$ and

$\mathcal{N}(\widetilde{w}_{j+1}^{(2)}, \eta^2\sigma^2\mathbb{I})$.

For $T = 1$, we only care about the marginals $P_{b_1^{(1)}}(\cdot)$ and $Q_{b_1^{(2)}}(\cdot)$, which as argued before, are normally distributed. From Lemma 27, we have established that the reflection map satisfies the conditions in Lemma 26. Combining these, we have that the base case $T = 1$ follows from the reflection coupling result stated as Lemma 26.

We proceed to the induction step. There are two cases, depending on whether we do a rejection sampling in the T^{th} step or not: we call these "rej-sample" and "no-rej-sample" respectively. If we do a rejection sampling, we further have two cases (1a). accept: either all rejection samplings, including the one in the T^{th} step are accepts, (1b). reflect: all rejection samplings, except the one in the T^{th} step are accepts, and in the T^{th} step, we reflect. Finally, if we don't do a rejection sampling step, we have the third case (2). reject: some rejection sampling prior to T results in reject; in this case, the T^{th} sample $\bar{w}_{T+1}^{(2)} \sim Q_{b_T^{(2)}}(\cdot|w_T^{(2)}, w_{T-1}^{(2)})$. Cases (1) and (2) partition the whole event space for T draws, whereas cases (1a) and cases (1b) partitions the space of the T^{th} draw, conditioned on the first event. Also note that case (1) vs (2) distinction is measurable w.r.t. the natural filtration generated by the Markov chain upto $T-1$ draws.

Note that conditioned on the events "rej sample" as well as $\mathbf{w}_{T-1}^{(2)}$, the last step is just a one-step reflection coupling method. To elaborate, the conditionals $Q_{b_T^{(2)}}(\cdot|w_T^{(2)}, w_{T-1}^{(2)})$ and $P_{b_T^{(2)}}(\cdot|w_T^{(2)}, w_{T-1}^{(2)})$ used in the T^{th} rejection sampling are Gaussians, which along with the reflection map satisfies properties of Lemma 26, as in the base case. Let $E_{last} = \{y|\exists \mathbf{x} : (\mathbf{x}, y) \in E\}$ be the projection of E on the last co-ordinate and $E_y = \{\mathbf{x}|(\mathbf{x}, y) \in E\}$. According to Lemma 27, the conditional distribution of $w_{T+1}^{(2)}$ is

$Q_{b_T^{(2)}}(\cdot|w_T^{(2)}, w_{T-1}^{(2)})$:

$$\begin{aligned}
&\mathbb{P}\left[\bar{w}_{T+1}^{(2)} \in E_{last}|\text{rej-sample}, \bar{\mathbf{w}}_{T-1}^{(2)}, (\mathbf{b}^{(1)}, \mathbf{b}^{(2)})\right] \\
&= \mathbb{P}\left[\bar{w}_{T+1}^{(2)} \in E_{last}, \text{accept}|\text{rej-sample}, \bar{\mathbf{w}}_{T-1}^{(2)}, (\mathbf{b}^{(1)}, \mathbf{b}^{(2)})\right] \\
&\quad + \mathbb{P}\left[\bar{w}_{T+1}^{(2)} \in E_{last}, \text{reflect}|\text{rej-sample}, \bar{\mathbf{w}}_{T-1}^{(2)}, (\mathbf{b}^{(1)}, \mathbf{b}^{(2)})\right] \\
&= \int_{\mathbb{R}^d} \mathbb{1}\left(\bar{w}_{T+1}^{(2)} \in E_{last}\right) \phi_{Q_{b_T^{(2)}}}(\bar{w}_{T+1}^{(2)}|w_T^{(2)}, w_{T-1}^{(2)}) d\bar{w}_{T+1}^{(2)}
\end{aligned}$$

For the "no-rej-sample" case, we have:

$$\begin{aligned}
&\mathbb{P}\left[\bar{w}_{T+1}^{(2)} \in E_{last}|\text{no-rej-sample}, \bar{\mathbf{w}}_{T-1}^{(2)}, (\mathbf{b}^{(1)}, \mathbf{b}^{(2)})\right] = \mathop{\mathbb{E}}_{\bar{w}_{T+1}^{(2)}}\left[\mathbb{1}\left(\bar{w}_{T+1}^{(2)} \in E_{last}\right)\right] \\
&= \int_{\mathbb{R}^d} \mathbb{1}\left(\bar{w}_{T+1}^{(2)} \in E_{last}\right) \phi_{Q_{b_T^{(2)}}}(\bar{w}_{T+1}^{(2)}|w_T^{(2)}, w_{T-1}^{(2)}) d\bar{w}_{T+1}^{(2)}
\end{aligned}$$

We will now combine the two cases. Let $\phi^{(r)}(\cdot)$ and $\phi^{(nr)}(\cdot)$ denote the densities of $\bar{\mathbf{w}}_{T-1}^{(2)}$ under the "rej-sample" and "no-rej-sample" events respectively.

$$\begin{aligned}
&\mathbb{P}\left[\bar{\mathbf{w}}^{(2)} \in E|(\mathbf{b}^{(1)}, \mathbf{b}^{(2)})\right] \\
&= \mathbb{P}\left[\bar{w}_{T+1}^{(2)} \in E_{last} \,\middle|\, \text{rej-sample}, \bar{\mathbf{w}}_{T-1}^{(2)}, (\mathbf{b}^{(1)}, \mathbf{b}^{(2)})\right] \mathbb{P}\left[\bar{\mathbf{w}}_{T-1}^{(2)} \in E_{\bar{w}_{T+1}^{(2)}}, \text{rej-sample}\right] \\
&+ \mathbb{P}\left[\bar{w}_{T+1}^{(2)} \in E_{last} \,\middle|\, \text{no-rej-sample}, \bar{\mathbf{w}}_{T-1}^{(2)}, (\mathbf{b}^{(1)}, \mathbf{b}^{(2)})\right] \mathbb{P}\left[\bar{\mathbf{w}}_{T-1}^{(2)} \in E_{\bar{w}_{T+1}^{(2)}}, \text{no-rej-sample}\right] \\
&= \int_{\mathbb{R}^{dT}} \mathbb{1}\left\{\bar{w}_{T+1}^{(2)} \in E_{last}\right\} \mathbb{1}\left\{\bar{\mathbf{w}}_{T-1}^{(2)} \in E_{\bar{w}_{T+1}^{(2)}}\right\} \\
&\cdot \left\{\mathbb{1}_{\text{rej-sample}}\left\{\bar{\mathbf{w}}_{T-1}^{(2)}\right\} \phi^{(r)}(\bar{\mathbf{w}}_{T-1}^{(2)}) + \mathbb{1}_{\text{no-rej-sample}}\left\{\bar{\mathbf{w}}_{T-1}^{(2)}\right\} \phi^{(nr)}(\bar{\mathbf{w}}_{T-1}^{(2)})\right\} \\
&\cdot \phi_{Q_{b_T^{(2)}}}(\bar{w}_{T+1}^{(2)}|w_T^{(2)}, w_{T-1}^{(2)}) d\bar{\mathbf{w}}_T^{(2)} \\
&= \int_{\mathbb{R}^{dT}} \mathbb{1}\left\{\bar{\mathbf{w}}^{(2)} \in E\right\} \phi_{Q_{\mathbf{b}_{T-1}^{(2)}}}(\bar{\mathbf{w}}_{T-1}^{(2)}) \phi_{Q_{b_T^{(2)}}}(\bar{w}_{T+1}^{(2)}|w_T^{(2)}, w_{T-1}^{(2)}) d\bar{\mathbf{w}}_T^{(2)} \\
&= \int_{\mathbb{R}^{dT}} \mathbb{1}\left\{\bar{\mathbf{w}}^{(2)} \in E\right\} \phi_{Q_{\mathbf{b}^{(2)}}}(\mathbf{w}_T^{(2)}) d\bar{\mathbf{w}}_T^{(2)} = Q_{\mathbf{b}^{(2)}}(E)
\end{aligned}$$

where the third equality uses the induction hypothesis that $\bar{\mathbf{w}}_{T-1}^{(2)}$, conditioned on $\mathbf{b}^{(1)}$ and $\mathbf{b}^{(2)}$, is distributed as $Q_{\mathbf{b}_{T-1}^{(2)}}$. Finally, we integrate with respect to the coupling

generating $(\mathbf{b}^{(1)}, \mathbf{b}^{(2)})$; we get

$$\begin{aligned}\mathbb{P}\left[\bar{\mathbf{w}}^{(2)} \in E\right] &= \sum_{(\mathbf{b}^{(1)},\mathbf{b}^{(2)})} \mathbb{P}\left[\bar{\mathbf{w}}^{(2)} \in E | (\mathbf{b}^{(1)}, \mathbf{b}^{(2)})\right] \mathbb{P}\left[\mathbf{b}^{(1)}, \mathbf{b}^{(2)}\right] \\ &= \sum_{(\mathbf{b}^{(1)},\mathbf{b}^{(2)})} Q_{\mathbf{b}^{(2)}}(E) \mathbb{P}\left[\mathbf{b}^{(1)}, \mathbf{b}^{(2)}\right] = \sum_{\mathbf{b}^{(2)}} Q_{\mathbf{b}^{(2)}}(E) \mathbb{P}\left[\mathbf{b}^{(2)}\right] = \widetilde{Q}(E)\end{aligned}$$

This completes the proof. □

We now show that not only the marginals over the iterates, but the entire state maintained by the algorithm, which includes the mini-batching indices is transported.

Lemma 29. *For any measurable event in* $\left(\mathbb{R}^d \times [n]^m\right)^{\otimes T}$*, we have*

$$\mathbb{P}\left[(\bar{\mathbf{w}}^{(2)}, \mathbf{b}^{(2)}) \in E\right] = Q(E)$$

Proof of Lemma 29. We first decompose the event $E \subseteq \left(\mathbb{R}^d \times [n]^m\right)^{\otimes T}$ as two events, $E = E_1 \times E_2$ where $E_1 \subseteq \mathbb{R}^{dT}$ and $E_2 \subseteq ([n]^m)^T$. We have

$$\begin{aligned}\mathbb{P}\left[(\bar{\mathbf{w}}^{(2)}, \mathbf{b}^{(2)}) \in E\right] &= \mathbb{E}_{\mathbf{b}^{(1)}} \mathbb{P}\left[\bar{\mathbf{w}}^{(2)} \in E_1 | \mathbf{b}^{(1)}, \mathbf{b}^{(2)}\right] \mathbb{P}\left[\mathbf{b}^{(2)} \in E_2\right] \\ &= \mathbb{E}_{\mathbf{b}^{(1)}} \widetilde{Q}_{\mathbf{b}^{(2)}}(E_1) \mu_{n,m}^{\mathrm{del}\otimes T}(E_2) \\ &= \mathbb{E}_{\mathbf{b}^{(1)}} \widetilde{Q}_{\mathbf{b}^{(2)}}(E_1) \mu_{n-1,m}^{\otimes T}(E_2) \\ &= \widetilde{Q}_{\mathbf{b}^{(2)}}(E_1) \mu_{n-1,m}^{\otimes T}(E_2) = Q(E)\end{aligned}$$

where the second and third equality follows from Lemma 28 and Claim 5, and the final equality follows from the definition of event E and probability distribution Q. □

We now lower bound the probability of accepting at all rejection sampling steps.

Lemma 30. *Let "accept" be the event in which all rejection sampling result in accepts so there is no reflection or recompute. The probability of accept is lower bounded as,*

$$\mathbb{P}\left[accept\right] \geq 1 - \frac{\sqrt{T}\rho}{8}$$

Proof of Lemma 30. We evaluate the probability that all rejection sampling steps result in accepts. We first do it conditioned on $\mathbf{b}^{(1)}, \mathbf{b}^{(2)}$

$$\begin{aligned}
&\mathbb{P}\left[\text{accept}|\mathbf{b}^{(1)}, \mathbf{b}^{(2)}\right] \\
&= \mathop{\mathbb{E}}_{\bar{w}_T^{(1)}, \bar{w}_{T+1}^{(2)}, \{u_j\}_{j=1}^T} \left[\prod_{j=1}^T \mathbb{1}_{\text{accept}}(u_j)\right] \\
&= \mathop{\mathbb{E}}_{\bar{\mathbf{w}}_T^{(1)}, \bar{\mathbf{w}}_{T+1}^{(2)}} \left[\prod_{j=1}^T \mathbb{P}\left[u_j \leq \frac{\phi_{Q_{b_j^{(2)}}}(\bar{w}_{j+1}^{(1)}|w_j^{(1)}, w_{j-1}^{(1)})}{\phi_{P_{b_j^{(1)}}}(\bar{w}_{j+1}^{(1)}|w_j^{(1)}, w_{j-1}^{(1)})} \middle| \bar{\mathbf{w}}_T^{(1)}, \bar{\mathbf{w}}_T^{(2)}\right]\right] \\
&= \int_{\mathbb{R}^{dT}} \prod_{j=1}^T \min\left\{\phi_{P_{b_j^{(1)}}}(\bar{w}_{j+1}^{(1)}|w_j^{(1)}) w_{j-1}^{(1)}), \phi_{Q_{b_j^{(2)}}}(\bar{w}_{j+1}^{(1)}|w_j^{(1)}), w_{j-1}^{(1)})\right\} d\bar{\mathbf{w}}_T^{(1)} \\
&= \prod_{j=1}^T \left(\int_{\mathbb{R}^d} \min\left\{\phi_{P_{b_j^{(1)}}}(\bar{w}_{j+1}^{(1)}|w_j^{(1)}), w_{j-1}^{(1)}), \phi_{Q_{b_j^{(2)}}}(\bar{w}_{j+1}^{(1)}|w_j^{(1)}), w_{j-1}^{(1)})\right\} d\bar{w}_{j+1}^{(1)}\right)
\end{aligned}$$

The term

$$\begin{aligned}
&\int_{\mathbb{R}^d} \min\left\{\phi_{P_{b_j^{(1)}}}(\bar{w}_{j+1}^{(1)}|w_j^{(1)}), w_{j-1}^{(1)}), \phi_{Q_{b_j^{(2)}}}(\bar{w}_{j+1}^{(1)}|w_j^{(1)}), w_{j-1}^{(1)})\right\} d\bar{w}_{j+1}^{(1)} \\
&= 1 - \text{TV}_{\bar{\mathbf{w}}_j^{(1)}, \mathbf{b}^{(1)}, \mathbf{b}^{(2)}}(P_{b_j^{(1)}}, Q_{b_j^{(2)}})
\end{aligned}$$

where the notation $\text{TV}_x(\cdot, \cdot)$ denotes the conditional TV between the arguments, conditioned on the subscript. Let $\gamma_j = \frac{\Delta(b_j^{(1)}, b_j^{(2)})}{2}$ i.e. the number of elements differing in $b_j^{(1)}$ and $b_j^{(2)}$. Note that if $\gamma_j = 0$, then $b_j^{(1)} = b_j^{(2)}$ and $P_{b_j^{(1)}} = Q_{b_j^{(2)}}$, and hence $\text{TV}_{\bar{w}_{j-1}^{(1)}, \mathbf{b}^{(1)}, \mathbf{b}^{(2)}}(P_{b_j^{(1)}}, Q_{b_j^{(2)}}) = 0$. In the other case, $\gamma_j = 1$, which corresponds to the case when the deleted point was used in the j^{th} mini-batch. In this case, the means of $P_{b_j^{(1)}}$ are $Q_{b_j^{(2)}}$ at separated by at most $\frac{2G\eta}{m}$ - this follows as in the proof of Proposition 1. In particular, fixing previous iterates and $\bar{w}_{j-1}^{(1)}$ and mini-batch indices $\mathbf{b}^{(1)}, \mathbf{b}^{(2)}$, using the fact that gradients are in norm bounded by G, $P_{b_j^{(1)}}$ and $Q_{b_j^{(2)}}$ are Gaussians with variance $\eta^2\sigma^2\mathbb{I}$ and means separated by either $\frac{2G\eta\gamma_j}{m}$ or 0, depending on γ_j. Combining the two cases, and using TV between Gaussians formula [DMR18], we have $1 - \text{TV}_{\bar{w}_{j-1}^{(1)}, \mathbf{b}^{(1)}, \mathbf{b}^{(2)}}(P_{b_j^{(1)}}, Q_{b_j^{(2)}}) \geq \left(1 - \frac{G\eta}{\eta\sigma m}\right)^{\gamma_j} = \left(1 - \frac{G}{\sigma m}\right)^{\gamma_j}$.

We therefore get $\int_{\mathbb{R}^d} \min\left\{\phi_{P_{b_j^{(1)}}}(\bar{w}_{j+1}^{(1)}|w_j^{(1)}), w_{j-1}^{(1)}), \phi_{Q_{b_j^{(2)}}}(\bar{w}_{j+1}^{(1)}|w_j^{(1)}), w_{j-1}^{(1)})\right\} d\bar{w}_{j+1}^{(1)} \geq \left(1 - \frac{G}{\sigma m}\right)^{\gamma_j}$. Plugging this in the conditional probability of accept expression, we get

$$\mathbb{P}\left[\text{accept}|\mathbf{b}^{(1)}, \mathbf{b}^{(2)}\right] \geq \prod_{j=1}^{T}\left(1 - \frac{G}{\sigma m}\right)^{\gamma_j} = \left(1 - \frac{G}{\sigma m}\right)^{\sum_{j=1}^{T}\gamma_j} \geq 1 - \frac{G\sum_{j=1}^{T}\gamma_j}{\sigma m}$$

We now integrate with respect to $\mathbf{b}^{(1)}, \mathbf{b}^{(2)}$. Note that $\sum_{j=1}^{T}\gamma_j$ is the number of mini-batches which contain the deleted point. Since in each mini-batch, m points are selected uniformly randomly from n, $\mathbb{E}_{\mathbf{b}^{(1)},\mathbf{b}^{(2)}}\gamma_j = \frac{m}{n}$, which gives us $\mathbb{E}_{\mathbf{b}^{(1)},\mathbf{b}^{(2)}}\sum_{j=1}^{T}\gamma_j = \frac{Tm}{n}$. Hence,

$$\mathbb{P}\left[\text{accept}\right] \geq 1 - \frac{G\mathbb{E}_{\mathbf{b}^{(1)},\mathbf{b}^{(2)}}\sum_{j=1}^{T}\gamma_j}{\sigma m} = 1 - \frac{GT}{\sigma n} = 1 - \frac{\sqrt{T}\rho}{8}$$

where the last equality follows from plugging in $\sigma = \frac{8\sqrt{T}G}{n\rho}$ as in Proposition 1. □

We are now ready to prove Proposition 2.

Proof of Proposition 2. We need to show that upon deletion and insertion, the probability distribution of the entire state maintained by the algorithm, which is all iterates as well as mini-batches indices is transported - this, for one deletion, follows from Lemma 29 (which is for unprojected iterates), together with the fact that projection is a deterministic operation. Moreover, as before, the above argument also holds for insertion and generalizes arbitrary edit requests.

We now proceed to bound the probability to recompute. This follows directly combining Lemma 30, Proposition 1 and Remark 1. From Remark 1, upon k edits, the total variation distance is at most k times total variation distance upon 1 edit. Since the algorithm is ρ-TV stable (Proposition 1), and the assumption that the number of samples are between $n/2$ and $2n$, the total variation distance upon k edits is at most $2k\rho$. Hence, using Lemma 30, we have that the probability to recompute is probability of "reject" is at most $\frac{k\rho\sqrt{T}}{4}$. □

C.7 Runtime and Space Complexity

In this section, we discuss, in detail, the learning and unlearning runtime of the algorithms, as well as their space complexity.

C.7.1 Learning Runtime

In this work, we did not aim to carefully optimize the runtime for training/learning algorithm, as long as the algorithm achieves the rate in Theorem 23. However, we briefly discuss the runtime of each algorithm, and highlight easy improvements, where possible. Algorithm 19 requires $mT = \rho n$ stochastic gradient computations. On the other hand, for Algorithm 9, if $m \leq \sqrt{\frac{LD\rho n}{G\sqrt{d}}}$, it requires $mT = m\left(\frac{\rho n}{md}\right) = \frac{(\rho n)^2}{d}$ stochastic gradient computations - setting larger m only hurts the total runtime, without any advantage. Note that total stochastic gradient descent computations of *noisy-m-SGD* (i.e. without acceleration, see Section C.8.1) is also $\frac{(\rho n)^2}{d}$; however, the key advantage of acceleration is that it allows setting larger mini-batch sizes: T^3 as opposed to T, which leads to smaller number of iterations: $\sqrt{\frac{\rho n}{\sqrt{d}}}$ as opposed to $\frac{\rho n}{\sqrt{d}}$ and hence a smaller probability of recompute. From [WS16], we know that mini-batch SGD (with or without acceleration) is optimal for smooth convex composite/ERM optimization in the low accuracy regime: when accuracy $\alpha = \Omega\left(\frac{1}{\sqrt{n}}\right)$. In this regime, an algorithm makes at least $\Omega\left(\frac{1}{\alpha^2}\right)$ calls to a stochastic gradient oracle. It can be then verified that for our accuracy, our algorithms make the optimal number of oracle calls.

For Algorithm 9, as discussed, faster algorithms lead to better unlearning times. It is natural to ask what happens if we additionally introduce variance reduction techniques on top of acceleration to yield even faster runtimes. In particular, what if we use Katyusha [AZ17], which has optimal runtime in terms of stochastic gradient computations. We argue that even though it improves the runtime of the learning

algorithm, it does not yield improvement for unlearning beyond what we have with acceleration. From Corollary 5.8 in [AZ17], setting largest allowed $m = \sqrt{n}$, we get that $Tm = O\left(\frac{\sqrt{n}}{\sqrt{\epsilon}}\right)$ – in our case, $\epsilon = \frac{\sqrt{d}}{\rho n}$, which yields $Tm = O\left(\sqrt{n}\sqrt{\frac{\rho n}{\sqrt{d}}}\right)$ stochastic gradient computations. Note that this is smaller than that of *noisy-m-A-SGD* (unless ρ is very small), however $T = \sqrt{\frac{\rho n}{\sqrt{d}}}$ – same as that of *noisy-m-A-SGD*, and hence yields no improvement in unlearning time. However, note that using Katyusha would give us optimal oracle complexity even in the *high* accuracy regime.

C.7.2 Unlearning Runtime

We now look at how much compute it takes for Algorithm 20 and 10 to handle the edit requests. We first give a general result, which holds for any TV-stable algorithm with the unlearning algorithm being the one which constructs a coupling with acceptance probability at least $1 - \rho$. We give in-expectation bounds on the number of times verification fails or a full or partial recompute is triggered.

Proposition 6. *For a coupling based unlearning algorithm with acceptance probability at least $1 - \rho$, for k edit requests, the expected number of times recompute is triggered is at most $4k\rho$.*

Proof of Proposition 6. We first setup some notation. In the general setup, for k edit requests, let s be the number of times a recompute is triggered. Let $\{Z_1, Z_2, \ldots, Z_s\}$ be a set of random variables, where each Z_i denotes how many edit requests the i^{th} recompute can *handle*. To elaborate, Z_i takes value j, if upon j edit requests, a recompute is triggered. The Z_i's comprises to the randomness used in the algorithm like mini-batching indices or Gaussian noise, as well as the randomness used for rejection sampling. It is important to note that $Z_i's$ are not necessarily independent. In particular, in Algorithm 10, we reuse the Gaussian noise upto the iteration in which rejection sampling fails, and only use fresh/independent Gaussian noise in the later

steps. However, note that we have exact unlearning, and the output at each step is ρ-TV stable (w.r.t. all the randomness used). Hence, since the above description of the distribution of Z_i's depend only on the TV stability parameter, it follows that $Z_i's$ are (marginally) identical.

We now use the fact that the unlearning algorithm constructs a coupling with acceptance probability at least $1 - \rho$ to describe the probability distribution of Z_i. We have that upon one edit request, the probability that a recompute is triggered is at most ρ. This means that $Z_i < 2$ with probability $\leq \rho$. Using Remark 1, this generalizes as $Z_i < j$ with probability at most $(j-1)\rho$. Note that in our setup, we observe at most k requests, so Z_i taking values larger than k is not meaningful. However if $k\rho < 1$, it means that probability that Z_i takes values smaller than k is less than 1, and therefore there is a positive probability of Z_i being larger than k. To remedy this, we define another random variable X_i's which takes values in the set $\{1, 2, \ldots, k\}$. Furthermore, for any i, $\mathbb{P}[X_i = j] = \mathbb{P}[Z_i = j]$ for $j \leq k$, but $\mathbb{P}[Z_i = j] = \sum_{l=k}^{\infty} \mathbb{P}[X_i = l]$. By construction, this ensures that $1 \leq Z_i \leq k$, when we observe at most k requests.

We want upper bounds on s conditioned on the fact that k requests are addressed i.e. $X_1 + X_2 \ldots, X_s \geq k$. For this we write s as $s := \min_q \{2k \geq X_1 + X_2 \ldots, X_q \geq k\}$. The first inequality holds trivially since we ensured that $X_i \leq k$. It is easy to see that s is a stopping time with respect to the filtered probability space of the stochastic process $\{X_i\}_{i \in \mathbb{N}}$. Furthermore, since X_i's are identical, we can apply Wald's equation to get,

$$2k \geq \mathbb{E}[X_1 + X_2 + \ldots X_s] = \mathbb{E}[s]\mathbb{E}[X_1] = \mathbb{E}[s] \sum_{j=1}^{k} \mathbb{P}\{X_i \geq j\} = \mathbb{E}[s] \sum_{j=1}^{\infty} \mathbb{P}\{Z_i \geq j\}$$
$$\geq \mathbb{E}[s] \sum_{j=1}^{1+1/\rho} \mathbb{P}\{Z_i \geq j\} = \mathbb{E}[s] \sum_{j=1}^{1+1/\rho} (1 - \mathbb{P}\{Z_i < j\}) \geq \mathbb{E}[s] \left(\frac{1}{\rho} - \sum_{j=1}^{1+1/\rho} (j-1)\rho \right)$$
$$\geq \mathbb{E}[s] \left(\frac{1}{\rho} - \int_{j=0}^{1/\rho} j\rho dj \right) \geq \mathbb{E}[s] \left(\frac{1}{\rho} - \frac{1}{2\rho^2}\rho \right) \geq \frac{\mathbb{E}[s]}{2\rho}$$

This gives us that $\mathbb{E}[s] \leq 4k\rho$. □

Next, we look at the runtimes for Algorithm 20 and Algorithm 10 to handle one deletion or insertion request. For this, we look at the runtime of verification, i.e., deciding if recompute needs to be triggered or not. We show how in the standard algorithmic model of computation (say, word RAM model), using suitable data structures, this can be done efficiently. Furthermore, as standard in convex optimization, we can use oracle model of computation [NY83] which counts the number of accesses to the first-order (gradient) information of the function, and a projection oracle. Let $\mathfrak{G}$ denote the compute cost for one gradient access or projection in the standard model of computation – we assume that both oracles require the same compute. In the rest of the discussion, we provide runtime as a function of the problem parameters ignoring all constants. Furthermore, since we assumed that the number of samples at any point in the stream is between $\frac{n}{2}$ and $2n$, we will just work with n samples, and everything would still be the same, up to constants.

Verification runtime of Algorithm 20. For Algorithm 20, note that for deletion, for every iteration, we need to check if the used mini-batch b_t contained the requested point. A brute force search takes $O(m)$ time, whereas if we sort when we save the mini-batch indices b_t, we can do a binary search in $O(\log(m))$ time; we can even do constant time search by storing a dictionary/hash table, giving us an $O(T)$ total time. The most efficient way however is to store a dictionary of sample to mini-batch iterations

that the sample was used in. For this, it takes $O(1)$ time lookup for every edit request. For insertion, similarly, at every iteration, we first sample from a Bernoulli with bias m/n which takes constant time, giving us $O(T)$ total time. However, equivalently, we just sample one Bernoulli with bias Tm/n and recompute based on its outcome. This gives us an $O(1)$ time lookup for every edit request.

Verification runtime of Algorithm 10. For Algorithm 10, we can similarly search in constant time whether the deleted point was used in any iteration or not. For every iteration in which the deleted point is in the mini-batch, we need to compute a gradient at a new point, so as to replace the deleted point. Sampling a point uniformly from a discrete universe takes linear time (in the size) in the worst case, but with some pre-processing can be done in logarithmic/constant time. For example, when saving the mini-batch indices b_t, if we save a sorted list of the indices not sampled, using binary search, we can sample in $O(\log(n-(m-1)))$ time. The more efficient way is, if we save a probability table, then we can use Alias method to sample in $O(1)$ time [Wal77]. Hence for such iterations, we query *two* gradients, and it takes $O(d)$ compute to add/subtract this gradients. Since the total number of iterations in which a deleted point was sampled in, in expectation, is $\frac{Tm}{n}$, the expected total compute is $\frac{Tm(\mathfrak{G}+d)}{n}$.

We now consider the computational cost of rejection sampling. In Algorithm 10, at every iteration we check if $\mathrm{Unif}(0,1) \leq \frac{\phi_{\mathcal{N}(g_t', \sigma^2\mathbb{I})}(\xi_t)}{\phi_{\mathcal{N}(g_t, \sigma^2\mathbb{I})}(\xi_t)}$, where $\phi_{\mathcal{N}(g_t, \sigma^2\mathbb{I})}(\cdot)$ and $\phi_{\mathcal{N}(g_t', \sigma^2\mathbb{I})}(\cdot)$ are probability densities evaluated at the sampled point ξ_t. We thus need to compute this ratio of probability densities – since these are Gaussian densities, the ratio is just the following the expression:

$$\frac{\phi_{\mathcal{N}(g_t', \sigma^2\mathbb{I})}(\xi_t)}{\phi_{\mathcal{N}(g_t, \sigma^2\mathbb{I})}(\xi_t)} = \frac{\frac{1}{(\sqrt{2\pi\sigma^2})^d} \exp\left(-\frac{\left\|g_t' - \xi_t\right\|^2}{2\sigma^2}\right)}{\frac{1}{(\sqrt{2\pi\sigma^2})^d} \exp\left(-\frac{\|g_t - \xi_t\|^2}{2\sigma^2}\right)} = \exp\left(\frac{1}{2\sigma^2}\left(\|g_t - \xi_t\|^2 - \|g_t' - \xi_t\|^2\right)\right).$$

It takes $O(d)$ time to do the above computation. Moreover, we only need to

compute the ratio in iterations where the means differ – these correspond to the iterations where the deleted point was sampled or the inserted point would have been sampled. By a direct computation, the expected number of such iterations is $\frac{Tm}{n}$. This gives us a computational cost of $\frac{Tmd}{n}$ for rejection sampling, and hence the expected runtime of verification is $\frac{Tm(\mathfrak{G}+d)}{n}$.

We now state bounds on runtime for both unlearning algorithms.

Claim 7. *The expected total unlearning runtime of Algorithm 20 for k edit requests is $O\left(\max\left\{k, \min\left\{\rho, 1\right\} k \cdot \textit{Training time}\right\}\right)$.*

Proof of Claim 7. The total runtime of Algorithm 20 is the time for verification plus the runtime for recomputation, whenever a recompute is triggered. The recomputation time is just the training time, and in the model considered, excepted cost of one recomputation takes $O\left(Tm\left(\mathfrak{G}+d\right)\right)$ time, since at every iteration, m gradients are computed and vectors added. As discussed in Section C.7, the expected verification time for Algorithm 20 is $O(1)$. From Proposition 4, the unlearning Algorithm 20 recomputes with probability $O(\min\left\{\rho, 1\right\})$ for one edit request. Therefore, using Proposition 6 which bounds the number of recomputes, we have that the expected total runtime is bounded as $kO(1) + 4k \min\left\{\rho, 1\right\} \cdot O\left(Tm\left(\mathfrak{G}+d\right)\right) \leq O\left(\max\left(1, \min\left\{\rho, 1\right\} Tm\left(\mathfrak{G}+d\right)\right) k\right)$. For a sufficiently large ρ, the unlearning time of Algorithm 20 is clearly dominated by the training time. In particular, in the corresponding batch Algorithm 19, we set $m = \frac{\rho n}{T}$, giving a total runtime of $O\left(\max\left(1, \min\left\{\rho^2, 1\right\} n\left(\mathfrak{G}+d\right)\right) k\right)$. Hence for $\rho = \Omega\left(\frac{1}{\sqrt{n(\mathfrak{G}+d)}}\right)$, the total runtime in expectation is at most $O(\min\left\{\rho, 1\right\} \cdot k \cdot \text{Training time})$. In the other case, the expected total runtime is just $O(k)$. □

Claim 8. *The expected total unlearning runtime of Algorithm 10 for k edit requests is $O\left(\max\left\{k, \min\left\{\rho\sqrt{T}, 1\right\} \cdot k \cdot \textit{Training time}\right\}\right)$.*

Proof of Claim 8. As before, the total runtime of Algorithm 20 is the time for ver-

ification plus the runtime for recomputation, whenever a recompute is triggered. As discussed in Section C.7, the expected verification time for Algorithm 10 is $O\left(\frac{Tm(\mathfrak{G}+d)}{n}\right)$. The recomputation in this case may be partial but it also includes a reflection. The reflection operation with d dimensional vectors takes $O(d)$ compute. Furthermore, we upper bound the partial recomputation time by worst-case full recomputation time, giving a recomputation time $O\left(Tm\left(\mathfrak{G}+d\right)\right)$. From Lemma 30, we have that the unlearning coupling is not maximal but recomputes with probability $\min\left\{\rho\sqrt{T}, 1\right\}$. Finally, by Proposition 6 we have that the expected total runtime is bounded as $kO\left(Tm\left(\mathfrak{G}+d\right)\right) + 4k\min\left\{\rho\sqrt{T}, 1\right\} \cdot O\left(\frac{Tm(\mathfrak{G}+d)}{n}\right) \leq O\left(\max\left(\min\left\{\rho\sqrt{T}, 1\right\}, \frac{1}{n}\right) kmT(\mathfrak{G}+d)\right)$. In contrast, for Algorithm 10, the runtime is at most $O\left(\max\left(\min\left\{\rho\sqrt{T}, 1\right\}, \frac{1}{n}\right) kmT(\mathfrak{G}+d)\right)$. Our lower bounds will show that we need $\rho = \Omega\left(\frac{1}{n}\right)$ to get any non-trivial accuracy. Therefore the maximum is always obtained by $\min\left\{\rho\sqrt{T}, 1\right\}$. Moreover, k is a trivial lower bound on runtime, since we need to observe all k edit requests. Hence, we get that the total runtime in expectation, is at most $O\left(\max\left\{k, \min\left\{\rho\sqrt{T}, 1\right\} \cdot k \cdot \text{Training time}\right\}\right)$. □

C.7.3 Space Complexity

In this work, the objective was not to optimize the memory used, but rather, to study if the problem can be solved computationally efficiently, no matter how much (reasonable) memory the algorithm uses. However, we discuss, in this section, that the space complexities of the proposed algorithms, which we will see, is arguably, reasonably small. We ignore the space used to store the dataset. In both algorithms, we save a hash-table of iterations to samples - since we do T iterations with m samples each, this takes space of $O(Tm)$ words. We also store all the iterates, which are d-dimensional vectors, so this takes a space of $O(dT)$ words. Finally, we also store a dictionary of iterations to models, which takes $O(T)$ space. The space complexity therefore is $O(T(\max\{m, d\})$. Plugging in the values of T, we get the following.

Algorithm 19. Plugging $T \leq \frac{\rho n}{m}$ from Proposition 3, we get space complexity $= O\left(\rho n \max\left\{1, \frac{d}{m}\right\}\right)$. As remarked in Section C.3, we can improve the space complexity by not requiring to save all the iterates and yet have the same unlearning runtime. In the proof of Claim 7, we upper bound the recomputation time by a *full* re-computation time - this means that the upper bound on unlearning runtime holds even if the algorithm does full retraining everytime verification fails. The unlearning Algorithm 20 can thus be modified as follows: for deletion, instead of continue retraining from iteration t where the deleted point participates, we can just do *full* retraining, with fresh randomness for all mini-batches. For insertion, note that when if condition is met (line 6 in Algorithm 20), we use the iterate w_t to compute the gradient on the inserted point (line 8 in Algorithm 20); however, if we don't save w_t, we can just compute it on the fly by doing a full retraining with the same old mini-batches. After w_t is computed, we just continue as in Algorithm 20.

With the above modification, we only need to save a hash-table of used samples to binary values which correspond to whether they were used or not, which takes $O(Tm)$ words, and a d dimensional model. Hence, the space complexity of Algorithm 20 is $O(Tm + d)$ words.

Algorithm 9: From Proposition 1, note that if $m \leq O(T^3)$, $T = \frac{\rho n}{md}$, and therefore, $dT = \frac{\rho n}{m}$. If we use the largest mini-batch size $m = O(T^3)$, then $T = \sqrt{\frac{\rho n}{\sqrt{d}}}$, and hence $dT = d^{3/4}\sqrt{\rho n}$. Therefore, the space complexity is $O(T \max\{m, d\}) \leq O\left(\max\left\{\frac{(\rho n)^2}{d}, d^{3/4}\sqrt{\rho n}\right\}\right)$ words.

C.8 Other Algorithms and Batch Unlearning

To demonstrate the generality of our framework, we give two more algorithms. The first is *noisy-m-SGD* which is the same as Algorithm 9 but without acceleration, and the second is *quantized-m-SGD*, based on randomized quantization. We note that

both algorithms have worse theoretical guarantees than Algorithm 9, however the first establishes our claim that acceleration is beneficial, whereas the second shows how a previous work of [GGVZ19] for k-means clustering, can, not only be seen as a special case of our framework, but also extended to general convex risk minimization problems. Moreover, in the second case, we consider a more general setup of *batch* edit requests, and show that our techniques are flexible enough to easily generalize to the batch variant.

C.8.1 *noisy-m-SGD*

Algorithm 21 *noisy-m-SGD*(w_{t_0}, t_0)

Input: Initial model w_{t_0}, data points $\{z_1, \ldots, z_n\}$, T, η, m

$w_0 = 0$
for $t = t_0, t_0 + 1, \ldots, T$ **do**
  Sample mini-batch b_t of size m uniformly randomly
  $g_t = \frac{1}{m} \sum_{j \in b_t} \nabla \ell(w_t; z_j)$
  Sample $\theta_t \sim \mathcal{N}(0, \sigma^2 \mathbb{I}_d)$
  $w_{t+1} = \mathcal{P}\left(w_t - \eta\left(g_t + \theta_t\right)\right)$
  Save$(b_t, \theta_t, w_t, g_t)$
end for

Output: $\widehat{w}_S = \frac{1}{T} \sum_{t=1}^{T+1} w_t$

Proposition 7. *Let $\ell(.; z)$ be an H-smooth G-Lipschitz convex function $\forall\ z$. Algorithm 21, run with $t_0 = 1, \eta = \min\left\{\frac{1}{2H}, \frac{D}{\left(\frac{G}{\sqrt{m}} + \sigma\right)\sqrt{T}}\right\}$, $\sigma = \frac{8\sqrt{T}G}{n\rho}$, and $T \geq \frac{(\rho n)^2}{16m^2}$ outputs $\widehat{w}_S$ which is $\min\{1, \rho\}$-TV stable and satisfies $\mathbb{E}\left[\widehat{L}(\widehat{w}_S; S) - \widehat{L}(w_S^*; S)\right] = O\left(\frac{GD\sqrt{d}}{\rho n}\right)$.*

Proof of Proposition 7. The TV-stability guarantee of $\rho = \frac{8\sqrt{T}G}{n}$ follows exactly as in the proof of Proposition 1. We now proceed to the accuracy guarantee, which follows simply by guarantee of SGD on smooth convex functions. We have already shown in Proposition 1 that the gradients are unbiased and its variance bounded by $\frac{2G^2}{m} + \frac{2G^2}{m} + \sigma^2 d$.

Therefore, using Theorem 4.1 in [AZ18] with step-size $\eta \leq \frac{1}{2H}$, we have

$$\mathbb{E}\left[\widehat{L}(\widehat{w};S) - \widehat{L}(w^*;S)\right] = O\left(2\eta\mathcal{V}^2 + \frac{D^2}{\eta T}\right) = O\left(2\eta\left(\frac{2G^2}{m} + \sigma^2 d\right) + \frac{D^2}{\eta T}\right)$$

Let $\widetilde{G}^2 = \frac{2G^2}{m} + \sigma^2 d$, balancing the trade-off in η gives us $\eta = \frac{D}{\widetilde{G}\sqrt{T}}$. Therefore setting $\eta = \min\left\{\frac{1}{2H}, \frac{D}{\widetilde{G}\sqrt{T}}\right\}$ gives us

$$\begin{aligned}\mathbb{E}\left[\widehat{L}(\widehat{w};S) - \widehat{L}(w^*;S)\right] &= O\left(\frac{HD^2}{T} + \frac{\widetilde{G}D}{\sqrt{T}}\right) \leq O\left(\frac{HD^2}{T} + \frac{GD}{\sqrt{Tm}} + \frac{\sigma\sqrt{d}D}{\sqrt{T}}\right) \\ &\leq O\left(\frac{HD^2}{T} + \frac{GD}{\sqrt{Tm}} + \frac{GD\sqrt{d}}{n\rho}\right)\end{aligned}$$

Finally, the condition in the sub-sampling amplification $\frac{8G^2}{m^2\sigma^2} \leq 1.256$ again becomes $T \geq \frac{(n\rho)^2}{16m^2}$. □

We now show that Algorithm 21 achieves the same upper bound on excess empirical risk as Algorithm 9.

Corollary 8. *Let $\ell(.;z)$ be an H-smooth G-Lipschitz convex function $\forall\ z$. Algorithm 21, run with $m \geq \min\left\{\frac{d}{16}, \frac{1}{4}\left(\frac{(\rho n)G\sqrt{d}}{HD}\right)^{1/2}\right\}$, $\eta = \min\left\{\frac{1}{2H}, \frac{D}{\left(\frac{G}{\sqrt{m}}+\sigma\right)\sqrt{T}}\right\}$, $\sigma = \frac{8\sqrt{T}G}{n\rho}$, and $T = \max\left\{\frac{(\rho n)^2}{md}, \frac{HD\rho n}{G\sqrt{d}}\right\}$ outputs $\widehat{w}_S$ which is ρ-TV stable and satisfies $\mathbb{E}\left[\widehat{L}(\widehat{w}_S;S) - \widehat{L}(w^*_S;S)\right] = O\left(\frac{GD\sqrt{d}}{\rho n}\right)$.*

Proof of Corollary 8. We start with the result in Proposition 7. Note that as long as $\frac{GD}{\sqrt{mT}} = \Omega\left(\frac{HD^2}{T}\right) \iff m = O\left(\frac{TG^2}{HD}\right)$, the second term is larger than the first. We balance the two trade-offs in T. Optimizing the trade-off between second and third term gives us $\frac{GD}{\sqrt{mT}} = \frac{GD\sqrt{d}}{\rho n} \iff T = \frac{(\rho n)^2}{md}$; and optimizing the second trade-off gives us $\frac{\sqrt{d}}{\rho n} = \frac{HD^2}{T} \iff T = \frac{HD^2(\rho n)}{\sqrt{d}}$. Hence setting $T = \max\left(\frac{(\rho n)^2}{md}, \frac{HD^2(\rho n)}{\sqrt{d}}\right)$ yields an expected excess empirical risk of $O\left(\frac{GD\sqrt{d}}{n\rho}\right)$.

We now look at the condition $T \geq \frac{(n\rho)^2}{16m^2}$ given in Proposition 7, with T set as $T = \max\left(\frac{(\rho n)^2}{md}, \frac{HD^2(\rho n)}{\sqrt{d}}\right)$. We therefore require $\frac{(\rho n)^2}{md} \geq \frac{(\rho n)^2}{16m^2} \iff m \geq \frac{d}{16}$, as well as $\frac{HD(\rho n)}{G\sqrt{d}} \geq \frac{(\rho n)^2}{16m^2} \iff m \geq \frac{1}{4}\left(\frac{(\rho n)G\sqrt{d}}{HD}\right)^{1/2}$ - this recovers the condition

$m \geq \min\left\{\frac{d}{16}, \frac{1}{4}\left(\frac{(\rho n)G\sqrt{d}}{HD}\right)^{1/2}\right\}$ in the Proposition statement. Hence, combining all the above arguments, we get that for any $m \geq \min\left\{\frac{d}{16}, \frac{1}{4}\left(\frac{(\rho n)G\sqrt{d}}{HD}\right)^{1/2}\right\}$, setting $T = \max\left\{\frac{(\rho n)^2}{md}, \frac{HD(\rho n)}{G\sqrt{d}}\right\}$, yields an expected excess empirical risk of $O\left(\frac{GD\sqrt{d}}{n\rho}\right)$. □

Algorithm 22 Unlearning for *noisy-m-SGD*

Input: Delete point with index j or insert z (with index $n+1$) for *noisy-m-SGD*

for $t = 1, 2 \ldots, T$ **do**
  Load$(\theta_t, w_t, b_t, g_t)$)
  if *deletion* and $j \in b_t$ **then**
    Sample $i \sim$ Uniform$([n] \backslash b_t)$
    $g_t' = g_t - \frac{1}{m}\left(\nabla \ell(w_t; z_j) - \nabla \ell(w_t; z_i)\right)$
    Save$(g_t', b_t \backslash \{j\} \cup \{i\})$
  else if *insertion* and Bernoulli$\left(\frac{m}{n+1}\right)$ **then**
    Sample $i \sim$ Uniform(b_t)
    $g_t' = g_t - \frac{1}{m}\left(\nabla \ell(w_t; z_i) - \nabla \ell(w_t; z)\right)$
    Save$(g_t', b_t \backslash \{i\} \cup \{n+1\})$
  else
    continue
  end if
  $\xi_t = g_t + \theta_t$
  if Uniform$(0,1) \geq \frac{\phi_{\mathcal{N}(g_t', \sigma^2 \mathbb{I})}(\xi_t)}{\phi_{\mathcal{N}(g_t, \sigma^2 \mathbb{I})}(\xi_t)}$ **then**
    $\xi_t' =$ reflect(ξ_t, g_t', g_t)
    $w_{t+1} = w_t - \eta \xi_t'$
    Save(ξ_t')
    noisy-m-SGD$(w_{t+1}, t+1)$ // *Continue retraining, on current dataset*
    break
  end if
end for

Remark 4. *The choice of T in Proposition 1 yields that the largest mini-batch size that can be set, without hurting runtime, is $m = \frac{\rho n G}{\sqrt{d} HD}$. Furthermore, the condition $m \geq \min\left\{\frac{d}{16}, \frac{1}{4}\left(\frac{(\rho n)G\sqrt{d}}{HD}\right)^{1/2}\right\}$ becomes $T \geq \left(\frac{\sqrt{d} HD}{4G}\right)^2$.*

We now state and prove the main theorem for this section.

Theorem 44. *Let $\ell(\cdot; z)$ be an H-smooth G-Lipschitz convex function $\forall\ z$. For any $\frac{1}{n} \leq \rho \leq 1$, using Algorithm 21 as the learning algorithm and Algorithm 22 as its unlearning algorithms, then given a stream of edit requests,*

1. *Satisfies exact unlearning at every point in the stream.*

2. *At time i in the stream of edit requests, outputs $\widehat{w}_{S^i}$, such that if $\left(\frac{H^{1/2}D^2\sqrt{d}}{G(\rho n)}\right)^{2/3} \leq \frac{GD}{\sqrt{\rho n}}$, then its with excess empirical risk bounded as,*

$$\mathbb{E}\left[\widehat{L}(\widehat{w}_{S^i}; S^i) - \widehat{L}(w^*_{S^i}; S^i)\right] = O\left(\left(\frac{H^{1/2}D^2\sqrt{d}}{G(\rho n)}\right)^{2/3}\right).$$

3. *For k edit requests, the expected total unlearning runtime is $O(\max\{\rho k \cdot \textit{Training time}, k\})$*

Proof of Theorem 44. We proceed as in the proof of Theorem 22. For any $0 < \widetilde{\rho} \leq 1$, from Proposition 7, the output $\widehat{w}_S$ is $\widetilde{\rho}$-TV stable, and the excess empirical risk using Algorithm 21 on a dataset S on n points, is bounded as,

$$\mathbb{E}\left[\widehat{L}(\widehat{w}_S; S) - \widehat{L}(w^*_S; S)\right] = O\left(\frac{HD^2}{T} + \frac{GD}{\sqrt{Tm}} + \frac{GD\sqrt{d}}{n\widetilde{\rho}}\right)$$

It can be easily verified that Proposition 2 and Proposition 2 still holds for *noisy-m-SGD*, which gives us that the algorithm satisfies exact unlearning at every time in the stream, proving the first part of the claim, Moreover, its recompute probability bounded by $O(\widetilde{\rho}k\sqrt{T})$ and therefore the unlearning runtime bounded by $O(\max\left\{k, \widetilde{\rho}k\sqrt{T} \cdot \text{Training time}\right\}$. Substituting $\widetilde{\rho} = \frac{\rho}{\sqrt{T}}$, and using the largest mini-batch size $m = \left(\frac{G}{HD}\right)^2 T$, the upper bound on excess empirical risk becomes $\frac{HD^2}{T} + \frac{GD\sqrt{d}\sqrt{T}}{n\rho}$. Optimizing the trade-off, we have $\frac{HD^2}{T} = \frac{HD^2}{T} \iff T = \left(\frac{HD(\rho n)}{G\sqrt{d}}\right)^{2/3}$, and the excess empirical risk bound upper bound is $\frac{HD}{T} = \left(\frac{H^{1/2}D^2\sqrt{d}}{G(\rho n)}\right)^{2/3}$. Note that this also proves the third part of the claim. Furthermore, as in the proof of Theorem 22, it can be verified that the condition $T \geq \frac{(\widetilde{\rho} n)^2}{16m^2}$ is equivalent to $\left(\frac{H^{1/2}D^2\sqrt{d}}{G(\rho n)}\right)^{2/3} \leq \frac{1}{\sqrt{\rho n}}$, which just means that the excess empirical risk of *noisy-m-SGD* is at most that of *sub-sample-GD*. Finally, the upper bound holds for any point in the stream using the assumption that the number of samples are between $\frac{n}{2}$ and $2n$, thereby establishing the second claim. □

C.8.2 *quantized-m-SGD*

The work of [GGVZ19] considers unlearning in k-means clustering. The key algorithmic technique is randomized quantization of vectors to a τ-lattice. The intuition is that if the vector is an average of n data points which are bounded in norm, then upon changing one data point, the vectors $O\left(\frac{1}{n}\right)$ close. Therefore, if the lattice is sufficiently coarse, then it would ensure that both are mapped up the same point in the lattice. However, if we consider deterministic quantization, then there exists points such that for any $\epsilon > 0$, shifting the point by ϵ changes the quantized point. Therefore, we first shift the lattice by a uniformly random phase, which ensures that such a situation occurs with a small probability.

In their application of k-means clustering, this vector is a cluster centroid, which is an average of the data points in the cluster. We apply this idea to convex risk minimization problems, wherein we quantize average gradients, which by the Lipshcitzness assumption are bounded in norm.

We now introduce the quantization operation formally. Given a vector x, let $\theta \sim \mathrm{Unif}\left[-\frac{1}{2}, -\frac{1}{2}\right]^d$, consider the quantization given by:

$$Q_\theta(\mathrm{x}) = \tau\left(\theta + \arg\min_{j \in \mathbb{Z}^d} (\mathrm{x} - \tau(\theta + j))\right)$$

We now state a result about the quantization operation.

Lemma 31. *Let $B_\delta(\mathrm{u})$ denote the Euclidean call of radius δ centered at u. The following holds for the quantization operation,*

1. *For any x, $\mathbb{E}\left[Q_\theta(\mathrm{x})\right] = \mathrm{x}$ and $\mathbb{E}\left[\|\mathrm{x} - \mathbb{E}\left[\mathrm{x}\right]\|\right]^2 \leq \tau^2 d$*

2. *For any vector u, $\mathbb{P}\left[\exists \mathrm{v} \in B_\delta(\mathrm{u}) : Q_\theta(\mathrm{u}) \neq \mathrm{Q}_\theta(\mathrm{v})\right] \leq \frac{2d\delta}{\tau}$*

Proof of Lemma 31. Note that for a given x, $\mathrm{Q}_\theta(\mathrm{x}) \sim \mathrm{Unif}\left[\mathrm{x} - \frac{\tau}{2}, \mathrm{x} + \frac{\tau}{2}\right]^d$, hence $\mathbb{E}\left[Q_\theta(w)\right] = w$. Furthermore, since $w - \mathbb{E}\left[w\right] \sim \mathrm{Unif}\left[-\frac{\tau}{2}, \frac{\tau}{2}\right]^d$, we have

$\mathbb{E}\left[\|w - \mathbb{E}[w]\|\right]^2 = d\mathbb{E}\left(w_1 - \mathbb{E}[w_1]\right)^2 = \frac{d\tau^2}{12}$. The second part of the claim is Lemma C.2 in [GGVZ19]. □

To see why [GGVZ19] is a special case of our framework, note that the total variation distance between two random variables is at most the probability of disagreement under *any* coupling. [GGVZ19] uses the same quantization randomness (used for training) for verifying after the edit request - this corresponds to a trivial coupling between the quantization randomness, hence the total variation distance between the outputs is bounded by the upper bound on the probability that the quantized points change (see Lemma 31). This establishes that it is a TV stable method. Finally, as said before, using the same quantization randomness corresponds to a trivial coupling, but can be shown to be maximal since the probability distribution is uniform around the to-be-quantized point. Therefore, we have that [GGVZ19] uses a maximal coupling based unlearning method.

Batch unlearning. We consider a batch unlearning setup, wherein instead of observing an insertion or deletion request, we observe a batch edit request with insertions and deletions. We demonstrate that our general approach of coupling mini-batch indices is flexible enough to handle this variant naturally. The batch unlearning ideas and results extend to other algorithms: *noisy-m-A-SGD*, *noisy-m-SGD* and *subsample-GD*. We also note that the computational benefit of batch unlearning as opposed to handling edits one by one is only a constant factor, which at best is two.

We now discuss how we extend the randomized quantization idea to convex risk minimization. In our learning algorithm *quantized-m-SGD*, at each iteration, we draw a mini-batch of m samples, uniformly randomly from n samples, use it to compute the gradient on the previous iterate , quantize using a randomly sampled phase, and update. Algorithm 24 implements the above procedure.

We first prove a lemma which bounds the total variation distance between outputs

Algorithm 23 *quantized-m-SGD*

Input: Initial model w_1, data points $\{z_1, \ldots, z_n\}$,T, η

1: **for** $t = 1, 2 \ldots, T$ **do**
2: Sample mini-batch b_t of size m uniformly randomly
3: Sample $\theta_t \sim \text{Unif}\left[-\frac{1}{2}, \frac{1}{2}\right]^d$
4: $g_t = \frac{1}{m} \sum_{j \in b_t} \nabla \ell(w_t; z_j)$
5: $w_{t+1} = w_t - \eta Q_{\theta_t}(g_t)$
6: Save$(\theta_t, w_t, b_t, g_t)$
7: **end for**

Output: $\widehat{w}_S = \frac{1}{T+1} \sum_{j=1}^{T+1} w_j$

generated by *quantized-m-SGD* on arbitrarily differing datasets - these can be thought of as arising after a batch edit request.

Lemma 32. *Let S and S' be two datasets of n and $n + k_2$ points respectively, such that S has k_1 points which differ from S' i.e. $|S \backslash S'| = k_1$, therefore S and S' differ by $k_1 + k_2$ points. Let $\{w_j\}_{j=1}^T$ and $\left\{w'_j\right\}_{j=1}^T$ be iterates of* quantized-m-SGD *on datasets S and S' respectively. The total variation distance between distribution of average iterates $\widehat{w}_S$ and $\widehat{w}_{S'}$ is bounded as,*

$$TV(\widehat{w}_S, \widehat{w}_{S'}) \leq \frac{4GTd(k_1 + k_2)}{n\tau}$$

Proof of Lemma 32. Without loss of generality, we enumerate S and S' into subsets as follows: let S_1 and S'_1 be the first $n - k_1$ elements of S and S' which are the same. Let S_2 and S'_2 be the next k_1 differing elements in S and S' respectively. Finally, let S'_3 be the last k_2 elements of S'.

We look at iteration t of *quantized-m-SGD* and fix the previous model $w_t = w'_t = w$. We will now compute the conditional total variation distance between w_{t+1} and w'_{t+1}. Note that since the only randomness is in the sub-sampling and quantization, we can compute the total variation distance between sub-sampled quantized gradients on fixed w for both datasets, and this will lower bound total variation distance between the iterates w_{t+1} and w'_{t+1} by data processing inequality. Let $b^{(1)}$ and $b^{(2)}$ be a uniform sample of m points from datasets S and S' respectively. For a fixed w, let the gradient

on S indexed by $b^{(1)}$ be denoted as $g^S_{b^{(1)}}(w) = \frac{1}{m}\sum_{j\in b^{(1)}} \nabla\ell(w; z^S_j)$, and similarly for S'. Let P and Q denote the probability distribution of $g^S_{b^{(1)}}(w)$ and $g^{S'}_{b^{(2)}}(w)$ respectively. We have the following claim, which we will prove via mathematical induction on k_2: for any measurable set R, for any k_2, $|P(R) - Q(R)| \leq \frac{4Gdk_1}{n\tau} + \sum_{j=1}^{k_2} \frac{4Gd}{(n+j)\tau}$.

Base case 1: $\mathbf{k_2 = 0}$. Firstly note that both $b^{(1)} \sim \text{Unif}([n], m)$ and $b^{(2)} \sim \text{Unif}([n], m)$, and consider the trivial coupling $b^{(1)} = b^{(2)} = b$, where $b \sim \text{Unif}([n], m)$, be a uniform sample of m points from $[n]$. We now use the fact that total variation distance is at most the probability of disagreement for any coupling. This gives us that

$$\text{TV}(Q_\theta(g^S_{b^{(1)}}(w)), Q_\theta(g^{S'}_{b^{(2)}}(w)) \leq \mathbb{P}\left[Q_\theta(g^S_b(w)) \neq Q_\theta(g^{S'}_b(w))\right]$$

We will focus on upper bounding the right hand side. The proof follows by using the quantization guarantee (Claim 31) combined amplification from subsampling. Without loss of generality, assume that the first k_1 samples in S and S' are the ones that differ. Fix the random (uniform) sample b of indices - suppose for this fixed value of b, exactly j differing data points are sampled. From G Lipschitzness, and that we have exactly j differing data points, $\left\|g^S_b(w) - g^{S'}_b(w)\right\| \leq \frac{2Gj}{m}$. Hence, applying Claim 31, we have that

$$\mathbb{P}\left[Q_\theta(g^S_b(w)) \neq Q_\theta(g^{S'}_b(w))\middle| b \text{ producing } j \text{ differing samples}\right] \leq \frac{4Gdj}{\tau m}.$$

We will now integrate with respect to the randomness in b - for this, we need to calculate the probability that a sample of b (uniform m out of n) produces exactly j differing data points, call it $p(j)$. By direct computation, we have that $p(j) = \frac{\binom{k_1}{j}\binom{n-k_1}{m-j}}{\binom{n}{m}}$.

Hence we have,

$$\begin{aligned}
&\mathbb{P}\left[Q_\theta(g_b^S(w)) \neq Q_\theta(g_b^{S'}(w))\right] \\
&= \sum_{j=0}^{k_1} p(j)\mathbb{P}\left[Q_\theta(g_b^S(w)) \neq Q_\theta(g_b^{S'}(w))\middle|\ b \text{ produces } j \text{ differing samples}\right] \\
&\leq \sum_{j=0}^{k_1} \frac{\binom{k_1}{j}\binom{n-k_1}{m-j}}{\binom{n}{m}} \frac{4Gdj}{\tau m} = \frac{mk_1}{n}\frac{4Gd}{\tau m} = \frac{4Gdk_1}{\tau n}
\end{aligned}$$

where the second last equality is a consequence of Vandermonde's identity, as we show below. We need to show that $\sum_{j=0}^{k_1} \frac{\binom{k_1}{j}\binom{n-k_1}{m-j}j}{\binom{n}{m}} = \frac{mk_1}{n} \iff \sum_{j=0}^{k_1}\binom{k_1}{j}\binom{n-k_1}{m-j}j = \frac{mk_1}{n}\binom{n}{m} = k_1\binom{n-1}{m-1}$. This holds because,

$$\begin{aligned}
\sum_{j=0}^{k_1}\binom{k_1}{j}\binom{n-k_1}{m-j}j &= \sum_{j=0}^{k_1}\frac{k_1}{j}\binom{k_1-1}{j-1}\binom{n-k_1}{m-j}j = k_1\sum_{j=0}^{k_1}\binom{k_1-1}{j-1}\binom{n-k_1}{m-j} \\
&= k_1\sum_{j=0}^{k_1-1}\binom{k_1-1}{j}\binom{n-k_1}{(m-1)-j} = k_1\binom{n-1}{m-1}
\end{aligned}$$

where in the second last equality, we re-indexed the sum which removes the first element, but it was zero anyway, and the last equality follows from Vandermonde's identity.

Base case 2: $\mathbf{k_2 = 1}$:. In this case, S' has one more element that S' - let this point be denoted as q. In this case, the probability distribution using S' has the form $Q = \left(1 - \frac{m}{n+1}\right)Q_1 + \frac{m}{n+1}Q_2$, where Q_1 is the probability distribution conditioned on the event that q is sub-sampled, and Q_2 is the probability distribution conditioned on the complementary event. For any measurable set R, we have,

$$|P(R) - Q(R)| = \left|\left(1 - \frac{m}{n+1}\right)Q_1(R) + \frac{m}{n+1}Q_2(R) - P(R)\right|$$

Note that Q_1, Q_2 and P are all probability distributions over n elements. Furthermore, P and Q_1 are probability distributions over k_1 differing elements, therefore we can use base case $k_2 = 0$ to get that $|P(R) - Q_1(R)| \leq \epsilon_1 := \frac{4Gdk_1}{n\tau}$. We therefore get,

$$\begin{aligned}
|P(R) - Q(R)| &\leq \left|\left(1 - \frac{m}{n+1}\right)Q_1(R) + \frac{m}{n+1}Q_2(R) - Q_1(R) + \epsilon_1\right| \\
&= \left|\frac{m}{n+1}\left(Q_2(R) - Q_1(R)\right) + \epsilon_1\right|
\end{aligned}$$

Finally note that Q_1 and Q_2 are probability distributions over n such that upon sub-sampling m elements, there is exactly one differing element, therefore we get, $|Q_2(R) - Q_1(R)| \leq \frac{4Gd}{m\tau}$. We therefore have that

$$|P(R) - Q(R)| \leq \frac{m}{n+1}\frac{4Gd}{m\tau} + \frac{4Gdk_1}{n\tau} = \frac{4Gd}{(n+1)\tau} + \frac{4Gdk_1}{n\tau}$$

Induction Hypothesis. Suppose the following holds for $k_2 \leq \widetilde{k}$: for any measurable set R, $|P(R) - Q(R)| \leq \frac{4Gdk_1}{n\tau} + \sum_{j=1}^{\widetilde{k}} \frac{4Gd}{(n+j)\tau}$.

Induction Step: $k_2 = \widetilde{k} + 1$. Let the *last* element of S' be q. As in the base case, we decompose the distribution Q into a mixture of two components based on whether q is sampled or not. We have $Q = \left(1 - \frac{m}{n+\widetilde{k}+1}\right) Q_1 + \frac{m}{n+\widetilde{k}+1} Q_2$. Note that Q_1 is a probability distribution which does not use the last element of S'. Therefore we can use Induction hypothesis which gives us that $|Q_1(R) - P(R)| \leq \frac{4Gdk_1}{n\tau} + \sum_{j=1}^{\widetilde{k}} \frac{4Gd}{(n+j)\tau}$. We therefore get,

$$\begin{aligned}
|P(R) - Q(R)| &= \left| P(R) - \left(1 - \frac{m}{n+\widetilde{k}+1}\right) Q_1(R) - \frac{m}{n+\widetilde{k}+1} Q_2(R) \right| \\
&\leq \left| \frac{m}{n+\widetilde{k}+1} \left(Q_1(R) - Q_2(R)\right) + \frac{4Gdk_1}{n\tau} + \sum_{j=1}^{\widetilde{k}} \frac{4Gd}{(n+j)\tau} \right| \\
&\leq \left| \frac{m}{n+\widetilde{k}+1} \frac{4Gd}{m\tau} + \frac{4Gdk_1}{n\tau} + \sum_{j=1}^{\widetilde{k}} \frac{4Gd}{(n+j)\tau} \right| \\
&= \frac{4Gdk_1}{n\tau} + \sum_{j=1}^{\widetilde{k}+1} \frac{4Gd}{(n+j)\tau}
\end{aligned}$$

where in the last inequality, as in the base case, we used that fact that distributions Q_1 and Q_2 differ because in one we subsample the last element where as in the other we don't, so from Claim 31, for two data sets of size m differing in one element, the failure probability is $\frac{4Gd}{m\tau}$. This completes the induction argument. We bound the sum simply as $\sum_{j=1}^{k_2} \frac{4Gd}{(n+j)\tau} \leq \frac{4Gdk_2}{n\tau}$, which gives us that the whole term is bounded by $\frac{4Gd(k_1+k_2)}{n\tau}$.

The above, by an application of data processing inequality, shows that the conditional TV distance between w_{t+1} and w'_{t+1} is at most $\frac{4Gd(k_1+k_2)}{n\tau}$. Note that the upper bound holds uniformly over all conditioning events. Moreover, from the maximal coupling characterization of TV distance, we have that for any coupling of w_{t+1} and w'_{t+1}, the conditional probability of disagreement is at most $\frac{4Gd(k_1+k_2)}{n\tau}$. Consider the coupling which just concatenates all these couplings, then an application of union bound over the T iterates, the joint probability of disagreement under this coupling is at most $\frac{4Gd(k_1+k_2)T}{n\tau}$ which gives us our upper bound on TV distance between joint iterates. Finally, by data processing inequality, the same upper bound holds for the average iterates which finishes the proof. □

We now establish the guarantees on the learning Algorithm 23. To handle batch edit request, we extend the notion of exact unlearning with one edit request to batch request: we term it exact batch unlearning. We similarly also extend ρ-TV-stability to (k_1, k_2, ρ)-TV stability, which is ρ-TV stability under arbitrary k_1 deletions and $k_1 + k_2$ insertions, as well as k_1 insertions and $k_1 + k_2$ deletions.

Proposition 8. *Let $\ell(.; z)$ be an H-smooth G-Lipschitz convex function $\forall$ z. Algorithm 21, run with $\eta = \min\left\{\frac{1}{2H}, \frac{D}{\left(\frac{G}{\sqrt{m}}+\tau\sqrt{d}\right)\sqrt{T}}\right\}$, $\tau = \frac{4GdT}{\rho n}$, and $T = \max\left\{\frac{(\rho n)}{d^{3/2}\sqrt{m}}, \left(\frac{HD(\rho n)}{Gd^{3/2}}\right)^{2/3}\right\}$ outputs $\widehat{w}_S$ which is $(k_1, k_2, (k_1 + k_2)\rho)$-TV stable and satisfies $\mathbb{E}\left[\widehat{L}(\widehat{w}_S; S) - \widehat{L}(w^*_S; S)\right] = O\left(\left(\frac{\sqrt{H}GD^2d^{3/2}}{\rho n}\right)^{2/3}\right)$.*

Proof of Proposition 8. The $(k_1, k_2, (k_1 + k_2)\rho)$-TV stability guarantee follows from Lemma 32 by taking a supremeum over all datasets S and S' of sizes n and $n + k_2$ (or $n - k_2$) to get that that TV stability is uniformly upper bound by $\frac{4GTd(k_1+k_2)}{n\tau} = (k_1 + k_2)\rho$, where the equality follows upon setting $\tau = \frac{4GdT}{\rho n}$. For the excess empirical risk bound, we use the guarantee on excess empirical risk of SGD on smooth convex functions (for example, Theorem 4.1 from [AZ18]), combined with the fact in Lemma 31

that quantization produces unbiased estimates of the gradient with bounded variance $\mathcal{V}^2 \leq \frac{2G^2}{m} + \tau^2 d = \frac{2G^2}{m} + \frac{16G^2d^3T^2}{(\rho n)^2}$. Therefore, choosing step size $\eta \leq \frac{1}{2H}$, we get

$$\mathbb{E}\left[\widehat{L}(\widehat{w}_S; S) - \widehat{L}(w_S^*; S)\right] \leq O\left(2\eta\mathcal{V}^2 + \frac{D^2}{\eta T}\right) = O\left(2\eta\left(\frac{2G^2}{m} + \frac{16G^2d^3T^2}{(\rho n)^2}\right) + \frac{D^2}{\eta T}\right)$$

Define $\widetilde{G}^2 = 2\left(\frac{2G^2}{m} + \frac{16G^2d^3T^2}{(\rho n)^2}\right)$ and set $\eta = \min\left\{\frac{1}{2H}, \frac{D}{\widetilde{G}\sqrt{T}}\right\}$, which makes the upper bound

$$\begin{aligned}\mathbb{E}\left[\widehat{L}(\widehat{w}_S) - \widehat{L}(w_S^*; S)\right] &= O\left(2\eta\mathcal{V}^2 + \frac{D^2}{\eta T}\right) = O\left(\frac{HD^2}{T} + \frac{\widetilde{G}D}{\sqrt{T}}\right) \\ &\leq O\left(\frac{HD^2}{T} + \frac{GD}{\sqrt{Tm}} + \frac{GDd^{3/2}\sqrt{T}}{(\rho n)}\right)\end{aligned}$$

Balancing the trade-off between the las two terms gives us $T = \frac{(\rho n)}{d^{3/2}\sqrt{m}}$. Similarly, balancing the trade-off between the first and last term gives us $T = \left(\frac{HD(\rho n)}{Gd^{3/2}}\right)^{2/3}$. Hence setting $T = \max\left\{\frac{(\rho n)}{d^{3/2}\sqrt{m}}, \left(\frac{HD(\rho n)}{Gd^{3/2}}\right)^{2/3}\right\}$ gives us that the expected excess empirical risk is bounded by $\left(\frac{\sqrt{H}GD^2d^{3/2}}{\rho n}\right)^{2/3}$ and completes the proof. □

Remark 5. *We see that the TV stability parameter above is $(k_1 + k_2)\rho$ as opposed to $(k_1 + 2k_2)\rho$ which is what we would obtain with ρ-TV stability for one edit request and using the triangle inequality of TV distance (see Remark 1).*

Remark 6. *The largest mini-batch size, without hurting runtime, is $m = \left(\frac{G}{HD}\right)^2 T = \left(\frac{G^2(\rho n)}{(HD)^2 d^{3/2}}\right)^{2/3}$, which gives us $T = \left(\frac{HD(\rho n)}{Gd^{3/2}}\right)^{2/3}$.*

We now proceed to unlearning. The unlearning algorithm (Algorithm 24) upon observing an edit request comprising of both insertions and deletions, first couples the mini-batch indices (described formally in the next paragraph), and computes the gradient on the new mini-batch It then uses the *same* quantization randomness as in training, and checks if the quantized point changes. If it does, in any iteration, then it calls recompute. The use of the same quantization randomness corresponds to a trivial coupling between the quantization randomness. We explain the coupling procedure in detail below.

Batch coupling. We setup some notation. Consider the training dataset S and dataset realized after the batch edit request S'. Given a vector w, let $Q_{\theta_1}(g^S_{b_1}(w))$ denote the quantized gradient vector where θ_1 is the quantization randomness and b_1 is the mini-batching randomness on dataset S'. Similarly, $Q_{\theta_2}(g^{S'}_{b_1}(w))$ denotes the quantized vector with θ_2 as the quantization randomness and b_2 as the mini-batching randomness on dataset S'. We couple θ_1 and θ_2 by considering the trivial coupling $\theta_1 = \theta_2$ i.e. the joint probability measure is defined only on the diagonal of the product measure. To couple the mini-batch indices, we consider two cases: if the training dataset S has less more or more points than S'. For simplicity, Algorithm 24 is the pseudo-code corresponding only to the first case.

In the first case, suppose S has n points and S' has $n + k_2$ points, realized after k_1 deletions and $k_1 + k_2$ insertions. Without loss of generality, order the two datasets as follows: the first $n - k_1$ points in S and S' are the same, call these $S_1 = S'_1$, next we have the last k_1 points of S, and arbitrary k_1 points of S' - call these S_2 and S'_2, and moreover let the mapping of indices from $S_2 \to S'_2$ by denoted by ι. Finally we have the rest of k_2 points of S', call this S'_3. In the following discussion, and in Algorithm 24, when we consider elements of these sets, we mean their indices. As before, let $\mu_{n,m}$ and $\mu_{n+k_2,m}$ denote the probability measures correspondingly to sampling m elements uniformly from a discrete universe of size n (i.e. S) and $n + k_2$ (i.e. S') respectively. These sub-sampling measures are coupled in the following way in Algorithm 24. We first sample $b^{(1)} \sim \mu_{n,m}$ (during training). Let $b^{(1)}_2$ be the m_2 indices in S_2: replace these by the corresponding indices in S'_2 i.e. $\iota(b^{(1)} \cap S_2)$. Next, sample $b \sim \mu_{n+k_2,m}$: let b_3 be the m_3 indices which are in S'_3. We now resample $b^{(2)}_1 \sim \text{Unif}(b^{(1)}, m_3)$ - these are indices used in training, which are now to be replaced. Define $b^{(2)} = \left(b^{(1)} \setminus \left\{b^{(1)} \cap S_2\right\} \cup \left\{\iota(b^{(1)} \cap S_2)\right\}\right) \setminus b^{(2)}_1 \cup \{S'_3 \cap b\}$. Let the distribution of $b^{(2)}$ produced in the above way be denoted as $\mu^{\text{edit}}_{n,m}$. We now show that $(b^{(1)}, b^{(2)})$ is indeed a coupling of $\mu_{n,m}$ and $\mu_{n+k_2,m}$.

Claim 9. *With the construction described above, we have that* $b^{(1)} \sim \mu_{n,m}$ *and* $b^{(2)} \sim \mu^{edit}_{n,m} = \mu_{n+k_2,m}$.

Proof of Claim 9. $b^{(1)} \sim \mu_{n,m}$ follows trivially by construction. For the other part, for any set E of m indices arising from the coupling construction, let E_1 be the set of $m - m_3$ points from $S \cup S'_2$ and E_2 be the set of m_3 points from S'_3. Since these m_3 points of E_2 need to be selected when sampling b, the probability of sampling these points is $\frac{\binom{n}{m-m_3}}{\binom{n+k_2}{m}}$, where the numerator denotes the number of ways to sample from $S' \backslash S'_3$. For the points in E_1, these come from $b^{(1)}$ and replacement using S'_2 (which is a deterministic operation). Hence, probability of E_1 is $\frac{\binom{n-(m-m_3)}{m_3}}{\binom{n}{m}}$, where the numerator denotes the number of ways to sample rest of elements not in E_1 when sampling $b^{(1)}$. Finally, we need to consider the re-sampling step i.e sampling $b^{(2)}_1$ - note that the draw of E_1 and E_2 fixes the set produced by this re-sampling, and thus its probability is $\frac{1}{\binom{m}{m_3}}$. This gives us

$$\begin{aligned}
\mu^{\text{edit}}_{n,m}(E) &= \frac{\binom{n}{m-m_3}}{\binom{n+k_2}{m}} \cdot \frac{\binom{n-(m-m_3)}{m_3}}{\binom{n}{m}} \cdot \frac{1}{\binom{m}{m_3}} \\
&= \frac{1}{\binom{n+k_2}{m}} \cdot \frac{n!(n-(m-m_3))!m!(n-m)!m_3!(m-m_3)!}{(m-m_3)!(n-(m-m_3))!m_3!(n-m)!n!m!} \\
&= \frac{1}{\binom{n+k_2}{m}} \\
&= \mu_{n+k_2,m}(E).
\end{aligned}$$

This finishes the proof. □

In the second case, S has more samples than S' - let number of samples in S be n, and in S' be $n - k_2$ and there k_1 samples in S' not in S. As before we order the sets as: let $S_1 = S'_1$ be the $n - k_2 - k_1$ samples which are the same in both S and S'. Let S_2 be the next k_1 samples in S, which correspond to S'_2, the rest of k_1 samples in S' - the mapping from S_2 to S'_2 being ι. Finally let S_3 be the rest of k_2 samples in S. We first sample $b^{(1)} \sim \mu_{n,m}$ (during training). Let $b^{(1)}_2$ be the

m_2 indices in S_2: replace these by the corresponding indices in S_2' i.e. $\iota(b^{(1)} \cap S_2)$. Let $b_3^{(1)}$ denote the sub-sampled indices which are in the last k_2 indices of S, and let $m_3 = \left|b_3^{(1)}\right|$. We re-sample m_3 indices as $b = \text{Unif}((S' \backslash \iota(b^{(1)} \cap S_2) \backslash b^{(1)}), m_3)$. Finally, define $b^{(2)} = \left(b^{(1)} \backslash \left\{b^{(1)} \cap S_2\right\} \cup \left\{\iota(b^{(1)} \cap S_2)\right\}\right) \backslash b_3^{(1)} \cup b$. Let the distribution of $b^{(2)}$ produced in the above way be denoted as $\mu_{n,m}^{\text{edit}}$. We now show that $(b^{(1)}, b^{(2)})$ is indeed a coupling of $\mu_{n,m}$ and $\mu_{n-k_2,m}$.

Claim 10. *With the construction described above, we have that* $b^{(1)} \sim \mu_{n,m}$ *and* $b^{(2)} \sim \mu_{n,m}^{edit} = \mu_{n-k_2,m}$.

Proof of Claim 10. $b^{(1)} \sim \mu_{n,m}$ follows trivially by construction. For the other part, let E be a set of m indices from $[n-k_2]$. Note that *any* number of points in E can arise due to re-sampling (i.e. when sampling b'), hence we need to consider all such possibilities - let m_3 be the number of indices in E produced via re-sampling. Fixing one of $\binom{m}{m_3}$ combinations, the probability that it was re-sampled is $\frac{1}{\binom{n-k_2-(m-m_3)}{m_3}}$. From the rule of sum, the probability that *any* m_3 sized set was produced via re-sampling is $\frac{\binom{m}{m_3}}{\binom{n-k_2-(m-m_3)}{m_3}}$. For each such set, it could arise from any of m_3 points from k_2, which gives us $\binom{k_2}{m_3}$ possibilities. The probability of choosing any such set, when sampling $b^{(1)}$, is $\frac{\binom{k_2}{m_3}}{\binom{n}{m}}$. We now combine these and apply the rule of sum on different choices of m_3, from 0 to m. We get,

$$\begin{aligned}
\mu_{n,m}^{\text{edit}}(E) &= \sum_{m_3=0}^{m} \frac{\binom{m}{m_3}}{\binom{n-k_2-(m-m_3)}{m_3}} \cdot \frac{\binom{k_2}{m_3}}{\binom{n}{m}} \\
&= \frac{1}{\binom{n}{m}} \sum_{m_3=0}^{m} \frac{m!(n-k_2-m)!m_3!}{m_3!(m-m_3)!(n-k_2-m+m_3)!} \binom{k_2}{m_3} \\
&= \frac{1}{\binom{n}{m}} \sum_{m_3=0}^{m} \frac{m!(n-k_2-m)!}{(n-k_2)!} \cdot \frac{(n-k_2)!}{(m-m_3)!(n-K_2-(m-m_3))!} \binom{k_2}{m_3} \\
&= \frac{\binom{n-k_2}{m}}{\binom{n}{m}} \sum_{m_3=0}^{m} \binom{n-k_2}{m-m_3} \binom{k_2}{m_3} = \frac{\binom{n-k_2}{m}}{\binom{n}{m}} \cdot \binom{n}{m} = \binom{n-k_2}{m} = \mu_{n-k_2,m}
\end{aligned}$$

where the third last equality follows from Vandermonde's identity. □

Algorithm 24 Batch unlearning for *quantized-m-SGD*

Input: Edit request produces dataset S' of $n+k_2$ points, with k_1 deletions and k_1+k_2 insertions; let S_1, S_2 and S_1', S_2' and S_3' be partitions of S and S' respectively, as defined in "Batch coupling"

for $t = 1, 2 \ldots, T$ **do**
  Load$(\theta_t, w_t, b_t, g_t)$
  $b \sim \text{Unif}(S', m)$
  $m_3 = |\{x \in S_3 \cap b\}|$
  $b_1^{(2)} \sim \text{Unif}(b_t, m_3)$
  $$g_t' = g_t - \frac{1}{m}\left(\sum_{j \in b_t \cap S_2} \nabla \ell(w_t; z_j) + \sum_{j \in \iota(b_t \cap S_2)} \nabla \ell(w_t; z_j) - \sum_{j \in b_1^{(2)}} \nabla \ell(w_t; z_j) + \sum_{j \in S_3 \cap b} \nabla \ell(w_t; z_j) \right)$$
  $b_t^{(2)} = (b_t \setminus \{b_t \cap S_2\} \cup \{\iota(b_t \cap S_2)\}) \setminus b_1^{(2)} \cup \{S_3' \cap b\}$
  Save$(g_t', b_t^{(2)})$
  if $Q_{\theta_t}(g_t) \neq Q_{\theta_t}(g_t')$ **then**
    quantized-m-SGD // *Recompute on current dataset*
    break
  end if
end for

We now state the main result about unlearning.

Proposition 9. *(Algorithm 23, Algorithm 24) satisfies exact batch unlearning. Moreover, for k batch edit requests, where the i^{th} request comprises of k_1^i deletions and $k_1^i + k_2^i$ insertions, or k_1^i insertions and $k_1^i + k_2^i$ deletions, Algorithm 24 recomputes with probability at most $2\sum_{i=1}^{k}(k_1^i + k_2^i)\rho$.*

Proof of Proposition 9. We consider one batch edit request of k_1 deletions and $k_1 + k_2$ insertions (case 1) and k_1 insertions and $k_1 + k_2$ deletions (case 2). We have that applications of Claims 9 and 9 give us that mini-batches are transported, for cases 1 and 2 respectively. Moreover, since we consider a trivial coupling of quantization randomness, we can consider it part of the (randomized) algorithmic map. Therefore, as in the proof of Proposition 4, transportation of mini-batches suffices to give us that Algorithm 24 satisfies exact unlearning. Repeated application of the above generalizes

it to arbitrary k edits. We now proceed to bound the probability of recompute directly for a batch edit request. For a fixed model w, and a fixed iteration, we fix the mini-batches $(b^{(1)}, b^{(2)})$ such that $b^{(1)}$ and $b^{(2)}$ differ by j indices. From Lemma 31, we have

$$\begin{aligned}&\mathbb{P}_{\theta,(b^{(1)},b^{(2)})}\left[Q_\theta(g^S_{b^{(1)}}(w)) \neq Q_\theta(g^{S'}_{b^{(2)}}(w))|(b^{(1)}, b^{(2)}) : b^{(1)}, b^{(2)} \text{ differ in } j \text{ indices}\right]\\&\leq \frac{4Gdj}{m\tau}\end{aligned}$$

We now integrate over the conditioning event. To do this, we need to compute the probability of the event that sampling $(b^{(1)}, b^{(2)})$ generates j differing indices - denote this as $p(j)$.

Since we have two case for coupling constructions, we consider each one by one. We first look at the second case: from construction of the coupling, it is easy to verify that j differing indices can be produced when, for any i, $b^{(1)}$ samples i elements from the k_1 differing items and $j-i$ indices from the last k_2 indices, for any i from 0 to j. Hence, by direct computation, we have

$$p(j) = \sum_{i=0}^{j} \frac{\binom{k_1}{i}\binom{k_2}{j-i}\binom{n-(k_1+k_2)}{m-j}}{\binom{n}{m}} = \frac{\binom{n-(k_1+k_2)}{m-j}}{\binom{n}{m}} \sum_{i=0}^{j} \binom{k_1}{i}\binom{k_2}{j-i} = \frac{\binom{n-(k_1+k_2)}{m-j}\binom{k_1+k_2}{j}}{\binom{n}{m}}$$

where the last equality follows from Vandermonde's identity. Plugging this in the following, we have,

$$\begin{aligned}&\mathbb{P}_{\theta,(b^{(1)},b^{(2)})}\left[Q_\theta(g^S_{b^{(1)}}(w)) \neq Q_\theta(g^{S'}_{b^{(2)}}(w))\right]\\&= \sum_{j=0}^{k_1+k_2} p(j)\mathbb{P}_{\theta,(b^{(1)},b^{(2)})}\left[Q_\theta(g^S_{b^{(1)}}(w)) \neq Q_\theta(g^{S'}_{b_2}(w))|b^{(1)}, b^{(2)} \text{ differ in } j \text{ indices}\right]\\&\leq \sum_{j=0}^{k_1+k_2} \frac{\binom{n-(k_1+k_2)}{m-j}\binom{k_1+k_2}{j}}{\binom{n}{m}} \frac{4Gdj}{m\tau} = \frac{m(k_1+k_2)}{n}\frac{4Gd}{m\tau}\\&= \frac{4Gd(k_1+k_2)}{n\tau} \leq \frac{k\rho}{T}\end{aligned}$$

where the second equality is a consequence of Vandermonde's identity proved in Lemma 32 (Base case $k_2 = 0$) and the last inequality follows by plugging in $\tau = \frac{4GdT}{\rho n}$.

We now look at the first case (when S is smaller than S'), which is slightly more involved. Let i_1 denote the number of indices in $b^{(1)} \cap S_1$, and let i_2 be the number of indices in $b^{(2)} \cup S'_3$. Furthermore, since we resample i_2 indices from $b^{(1)}$, let i_3 be the number of indices from S_1 which are re-sampled. It can be verified that if $b^{(1)}$ and $b^{(2)}$ differ in j indices, then we need to have $i_1 + i_3 = j$. This is because it can happen that both $i^{(2)}$ is large, but upon re-sampling, it chooses elements from k_1, which does not increase the number of different indices between $b^{(1)}$ and $b^{(2)}$ Also, note that by construction $i_3 \leq i_2 \leq j$. Hence the probability $p(j)$, by direct computation is,

$$\begin{aligned} p(j) &= \sum_{i_1,i_2,i_3=0, i_1+i_3=j, i_3\leq i_2\leq j}^{j} \frac{\binom{k_1}{i_1}\binom{n-k_1}{m-i_1}}{\binom{n}{m}} \cdot \frac{\binom{k_2}{i_2}\binom{n}{m-i_2}}{\binom{n+k_2}{m}} \cdot \frac{\binom{i_1}{i_2-i_3}}{\binom{m}{i_2}} \\ &= \sum_{i_1=0}^{j} \sum_{i_2=j-i_1}^{j} \frac{\binom{k_1}{i_1}\binom{n-k_1}{m-i_1}}{\binom{n}{m}} \cdot \frac{\binom{k_2}{i_2}\binom{n}{m-i_2}}{\binom{n+k_2}{m}} \cdot \frac{\binom{i_1}{i_2-(j-i_1)}}{\binom{m}{i_2}} \end{aligned}$$

where in the second equality, we substituted $i_3 = j - i_1$. We now claim that $p(j) = \frac{\binom{n-k_1}{m-j}\binom{k_1+k_2}{j}}{\binom{n+k_2}{m}}$, which we will argue via a double counting argument. Note that it suffices to show that $\sum_{i_1=0}^{j}\sum_{i_2=j-i_1}^{j}\binom{k_1}{i_1}\binom{n-k_1}{m-i_1}\cdot\binom{k_2}{i_2}\binom{n}{m-i_2}\frac{\binom{i_1}{i_2-(j-i_1)}}{\binom{m}{i_2}} = \binom{n-k_1}{m-j}\binom{k_1+k_2}{j}\binom{n}{m}$. Consider set A of $n+k_2$ elements, composed of A_1 of $n-k_1$, A_2 of k_1 and A_3 of k_2 elements, and a B of n elements, composed of B_1 of $n-k_1$ and B_2 of k_1 elements. Note that the expression $\binom{n-k_1}{m-j}\binom{k_1+k_2}{j}\binom{n}{m}$ is the size of number of combinations of $2m$ elements, m each from A and B such that the number of elements from $A_2 \cup A_3$ is j. We will show that the other expression also counts this set, via basic combinatorial rules. For this, consider combinations of m elements from B A_3 and such that we have i_2 elements from A_3 and the rest $m - i_2$ from B. Also, consider combinations of m elements from $A_1 \cup A_2$ which consists of i_1 elements from A_2 the rest from A_1. We now modify these as follows, out of m elements from A, select i_2 elements and replace thse from elements from A_3 - not that if it turns out that out of i_2 selected, $j - i_1$ are from A_1, then the

number of elements from $A_2 \cup A_3$ after replacement becomes exactly j. However, also note that for each such combination arising, there are $\binom{m}{i_2}$ combinations of samples from A_1 and A_2, which give the same *final* combination after replacement. Hence, we need to apply the rule of division, so as not to repeatedly count the same combination. Finally, using the rule of sum to consider all possible values of i_1 and i_2 retrieves the expression $\sum_{i_1=0}^{j}\sum_{i_2=j-i_1}^{j}\binom{k_1}{i_1}\binom{n-k_1}{m-i_1}\cdot\binom{k_2}{i_2}\binom{n}{m-i_2}\frac{\binom{i_1}{i_2-(j-i_1)}}{\binom{m}{i_2}} = \binom{n-k_1}{m-j}\binom{k_1+k_2}{j}\binom{n}{m}$ and completes the argument.

We again plug in the above in the following expression to get,

$$
\begin{aligned}
&\mathbb{P}_{\theta,(b^{(1)},b^{(2)})}\left[Q_\theta(g^S_{b^{(1)}}(w)) \neq Q_\theta(g^{S'}_{b^{(2)}}(w))\right] \\
&= \sum_{j=0}^{k_1+k_2} p(j)\mathbb{P}_{\theta,(b^{(1)},b^{(2)})}\left[Q_\theta(g^S_{b^{(1)}}(w)) \neq Q_\theta(g^{S'}_{b_2}(w))|b^{(1)}, b^{(2)} \text{ differ in } j \text{ indices}\right] \\
&\leq \sum_{j=0}^{k_1+k_2} \frac{\binom{n-k_1}{m-j}\binom{k_1+k_2}{j}}{\binom{n+k_2}{m}}\frac{4Gdj}{m\tau} = \frac{m(k_1+k_2)}{n}\frac{4Gd}{m\tau} \\
&= \frac{4Gd(k_1+k_2)}{n\tau}
\end{aligned}
$$

where the second equality is again a consequence of Vandermonde's identity as in Lemma 32, and the last inequality follows by plugging in $\tau = \frac{4GdT}{\rho n}$. Finally, we condition on the iterates till iteration t, which gives us the conditional probability of the iterates differing at iteration t is at most $\frac{(k_1+k_2)\rho}{T}$. Taking a union bound over all T iterations gives us that probability is at most $(k_1+k_2)\rho$. Finally, we extend it to k edit request, by using the fact, by assumption than the number of data points at any point in the stream is between $\frac{n}{2}$ and $2n$. This, with the result for one edit request, directly give us the probability to recompute is at most $2\sum_{i=1}^{k}(k_1^i+k_2^i)\rho$. □

We now state and prove the main result.

Theorem 45. *Let $\ell(\cdot; z)$ be an H-smooth G-Lipschitz convex function $\forall$ z. For any $\frac{1}{n} \leq \rho < \infty$, using Algorithm 23 as the learning algorithm and Algorithm 24 as its unlearning algorithm, then given a stream of batch edit requests,*

1. *Satisfies exact batch unlearning at every point in the stream.*

2. *At time i in the stream of edit requests, outputs $\widehat{w}_{S^i}$, such that its excess empirical risk bounded as,*

$$\mathbb{E}\left[\widehat{L}(\widehat{w}_{S^i}; S^i) - \widehat{L}(w^*_{S^i}; S^i)\right] = O\left(\left(\frac{\sqrt{H}GD^2d^{3/2}}{\rho n}\right)^{2/3}\right).$$

3. *For k batch edit requests, where the i^{th} request comprises of k_1^i deletions and $k_1^i + k_2^i$ insertions, or k_1^i insertions and $k_1^i + k_2^i$ deletions, the expected total unlearning runtime is $O(\max\left\{\min\{\rho, 1\}\sum_{i=1}^k (k_1^1 + k_2^i) \cdot \textit{Training time}, k\right\})$*

Proof of Theorem 45. The first and the second claims follow from Proposition 9 and Proposition 8 respectively combined with the assumption that the number of samples at every point in the stream is between $\frac{n}{2}$ and $2n$. Finally, as in the the proof Claim 8 for runtime *noisy-m-A-SGD*, we can use the same data-structures together with the fact the quantization operation takes $O(d)$ time, to get that the claimed runtime. These together finish the proof of Theorem 45. □

C.9 Lower Bounds on Excess Empirical Risk

Give a convex function $f(\cdot, z)$, we consider empirical risk minimization on a dataset of n points. We assume $f(\cdot, z)$ is 1-Lipschitz for all z, and $\text{diam}(\mathcal{W}) \leq 1$. This is only for simplification as the bounds scale naturally with these constants, as discussed in [BST14]. We look at algorithms, which given two datasets S and S' of size n differing by one point, *disagree* only on a set of measure at most an ρ.

We have from the optimal transport connection that this requirement is equivalent to the total variation distance being at most ρ. We want to understand then what is the lower bound on excess empirical risk:

$$\sup_{\Delta(S,S')=1} \mathbb{P}\left[\mathcal{A}(S') \neq \mathcal{A}(S)\right] \leq \rho \iff \sup_{\Delta(S,S')=1} \text{TV}(\mathcal{A}(S), \mathcal{A}(S')) \leq \rho$$
$$\implies \mathbb{E}\left[\text{excess empirical risk}\right] \geq \alpha(\rho, n, d)$$

We focus on proving the implication. [BST14] gave lower bounds on accuracy for DP algorithms by providing a reduction to computing mean of the dataset. We present and give the proof of the reduction, adapted to our context, for completeness. The reduction is that if we have a TV-stable algorithm for empirical risk minimization for a particular f with some accuracy, then we have a TV-stable algorithm for mean computation problem with *certain* accuracy. We will look at mean computation problem over datasets with norm of the mean being $\Theta(M)$, for some given M. Let $\mu(S) = \frac{1}{n}\sum_{j=1}^{n} z_i$ denote the mean of dataset $S = \{z_1, z_2, \cdots z_n\}$.

Let the optimal accuracy of such a mean computation problem be denoted as follows:

$$\alpha^2_{\text{mean}}(n, \rho, d, M) := \min_{\mathcal{A}:\rho\text{-TV-stable}} \max_{\substack{S=\{z_i\}_{i\in[n]}:\|z_i\|\leq 1,\\ M/2\leq\|\mu(S)\|\leq 2M}} \mathbb{E}_{\mathcal{A}} \left\| \mathcal{A}(S) - \frac{1}{n}\sum_{i=1}^{n} z_i \right\|^2$$

Proposition 10. *For any ρ-TV stable algorithm $\mathcal{A}$, there exists a 1-Lipschitz convex function f, a constraint set $\mathcal{W}$ with diameter$(\mathcal{W}) \leq 1$ and a dataset S of n data point such that*

$$\mathbb{E}\left[\widehat{L}(\mathcal{A}(S); S) - \widehat{L}(w^*; S)\right] \geq \max_{M}\left\{\frac{\alpha^2_{mean}(n, \rho, d, M)}{2M}\right\}$$

Proof of Proposition 10. We follow the proof in [BST14]. Consider dataset $S = \{z_1, z_2, \ldots, z_n\}$, $z_i \in \left\{-\frac{1}{\sqrt{d}}, \frac{1}{\sqrt{d}}\right\}^d$ - the dataset is therefore constrained to lie in the unit Euclidean ball. Consider the following function $f(w, z) = -\langle w, z\rangle$ with the constraint set $\mathcal{W}$ being the unit Euclidean ball. It is easy to see that $f(\cdot, z)$ is 1-Lipschitz for all z. The empirical risk becomes $\widehat{F}_S(w) = -\left\langle w, \frac{1}{n}\sum_{i=1}^{n} z_i\right\rangle$, the minimum of which over the unit ball is $w^*_S = \frac{\frac{1}{n}\sum_{i=1}^{n} z_i}{\left\|\frac{1}{n}\sum_{i=1}^{n} z_i\right\|}$.

Given an algorithm $\mathcal{A}$ for empirical risk minimization, let the reduced mean

estimate be $\widehat{\mu}(S) = \left\|\frac{1}{n}\sum_{i=1}^{n} z_i\right\| \mathcal{A}(S)$. The accuracy (mean-squared error) of $\widehat{\mu}$ is,

$$\begin{aligned}
\|\widehat{\mu}(S) - \mu(S)\|^2 &= \left\|\left\|\frac{1}{n}\sum_{i=1}^{n} z_i\right\| \mathcal{A}(S) - \frac{1}{n}\sum_{i=1}^{n} z_i\right\|^2 \\
&= \left\|\left\|\frac{1}{n}\sum_{i=1}^{n} z_i\right\| \left(\mathcal{A}(S) - \frac{\frac{1}{n}\sum_{i=1}^{n} z_i}{\left\|\frac{1}{n}\sum_{i=1}^{n} z_i\right\|}\right)\right\|^2 \\
&= \left\|\frac{1}{n}\sum_{i=1}^{n} z_i\right\|^2 \|\mathcal{A}(S) - w_S^*\|^2 \\
&\leq \left\|\frac{1}{n}\sum_{i=1}^{n} z_i\right\| 2\left(\widehat{F}_S(\mathcal{A}(S)) - \widehat{F}_S(w^*)\right)
\end{aligned}$$

where the last inequality follows using the following computation, wherein we use the fact all data point are in the unit ball.

$$\begin{aligned}
\left\|\frac{1}{n}\sum_{i=1}^{n} z_i\right\|^2 \|\mathcal{A}(S) - w_S^*\|^2 &\leq \left\|\frac{1}{n}\sum_{i=1}^{n} z_i\right\| 2\left(1 - \langle \mathcal{A}(S), w_S^*\rangle\right) \\
&= 2\left(\left\|\frac{1}{n}\sum_{i=1}^{n} z_i\right\| - \left\|\frac{1}{n}\sum_{i=1}^{n} z_i\right\| \left\langle \mathcal{A}(S), \frac{\frac{1}{n}\sum_{i=1}^{n} z_i}{\left\|\frac{1}{n}\sum_{i=1}^{n} z_i\right\|}\right\rangle\right) \\
&= 2\left(\left\langle w_S^*, \frac{1}{n}\sum_{i=1}^{n} z_i\right\rangle - \left\langle \mathcal{A}(S), \frac{1}{n}\sum_{i=1}^{n} z_i\right\rangle\right) \\
&= 2(\widehat{L}(\mathcal{A}(S); S) - \widehat{L}(w^*; S)
\end{aligned}$$

We therefore get,

$$\widehat{L}(\mathcal{A}(S); S) - \widehat{L}(w^*; S) \geq \frac{1}{2\left\|\frac{1}{n}\sum_{i=1}^{n} z_i\right\|} \|\widehat{\mu}(S) - \mu(S)\|^2$$

There are two things left to show: a bound on $\frac{1}{2\left\|\frac{1}{n}\sum_{i=1}^{n} z_i\right\|} \|\widehat{\mu}(S) - \mu(S)\|^2$ and show that the reduced algorithm $\widehat{\mu}(S)$ is also ρ-TV stable. We proceed with the latter: note that $\widehat{\mu}(S) = \left\|\frac{1}{n}\sum_{i=1}^{n} z_i\right\| \mathcal{A}(S)$. However the term $\left\|\frac{1}{n}\sum_{i=1}^{n} z_i\right\|$ depends on the dataset, and even if, for a neighbouring dataset S', $\mathcal{A}(S)$ and $\mathcal{A}(S')$ are ρ-close in total variation, this data dependent scaling can potentially increase the distance. However, if instead we define $\widehat{\mu}(S) = M\mathcal{A}(S)$, where M is a constant, then it is indeed ρ TV stable. Moreover, for reasonable values of M, the there exists dataset for which $\left\|\frac{1}{n}\sum_{i=1}^{n} z_i\right\| = \Theta(M)$. Finally, note that by definition, $\|\widehat{\mu}(S) - \mu(S)\|^2 \geq \alpha^2_{\text{mean}}(n, \rho, d, M)$. Taking a max over all M gives us the desired statement. □

C.9.1 Lower Bound for Mean Computation

In this section, we look at the problem of mean computation with TV stability constraint. Note that to establish lower bounds on excess empirical risk, we need to look at mean computation over data sets with means between $M/2$ and $2M$, for a given M. However, we will see the mean computation even over the unit ball has same accuracy convex ERM. We will therefore establish lower bounds for the general mean computation problem, but the construction will use datasets with means $\Theta(M)$ for certain values of M. Given a dataset $S = \{x_1, x_2, \ldots, x_n\}$, where $x_i \in \left\{-\frac{1}{\sqrt{d}}, \frac{1}{\sqrt{d}}\right\}^d$ for all i, the task is to compute the mean $\mu(S) = \frac{1}{n}\sum_{j=1} x_j$, while ensuring that the procedure is ρ-TV-stable. This task is often considered in the differential privacy literature, however with the data points being $x_j \in \{0,1\}^d$. The mean computation task then corresponds to releasing all one-way marginals of the database. Since we want to consider data points which lie inside the Euclidean ball, we therefore scale it accordingly. Given an algorithm $\mathcal{A}(S)$, the accuracy is defined as mean-squared error: $\alpha^2 := \alpha^2_{\text{mean}}(n, \rho, d) = \mathbb{E}\left[\|\mathcal{A}(S) - \mu(S)\|^2\right]$ where the expectation is over the randomization of the algorithm.

We first describe two algorithms for this problem and give upper bounds.

Subsample-mean. Consider an algorithm which sub-samples a ρ-fraction of the dataset and outputs the mean on it.

Claim 11. *The* Subsample-mean *procedure satisfies ρ-TV-stability and has accuracy $\alpha^2 \leq O\left(\frac{1}{\rho n}\right)$.*

Proof of Claim 11. The ρ-TV stability claim follows since TV distance is witnessed by the event that a differing sample is sub-sampled, which happens with probability $\frac{\rho n}{n} = \rho$. The proof of accuracy follows from the proof of Proposition 3, wherein we computed the gradient on uniformly sub-sampled m out of n points - we showed

that the mean on sub-sampled points is an unbiased estimate of the average gradient. Furthermore, since the gradients were bounded as well, the expected accuracy of mean computation is the same as the variance of gradient computation, which we derived to be $O\left(\frac{1}{m}\right) = O\left(\frac{1}{\rho n}\right)$. □

Noisy-mean. The algorithm computes the mean and adds $N(0, \sigma^2 I)$ noise to it with $\sigma^2 = \frac{C}{n^2\rho^2}$, where C is an appropriate universal constant.

Claim 12. *The* Noisy-mean *procedure satisfies ρ-TV-stability and has accuracy $\alpha^2 \leq O\left(\frac{d}{(\rho n)^2}\right)$*

Proof of Claim 12. Since the difference in means of two datasets, in norm, is at most $\frac{2}{n}$, the outputs are two multivariate Gaussians with variance σ^2 and means separated by $\frac{2}{n}$. From [DMR18], the total variation distance between such Gaussian sis at most $O\left(\frac{1}{n\sigma}\right) = \rho$. For the accuracy, we have $\alpha^2 = \mathbb{E}\left[\|\mu(S) + xi - \mu(S)\|^2\right] = \mathbb{E}\left[\|xi\|^2\right] = d\sigma^2 = O\left(\frac{d}{n^2\rho^2}\right)$. □

If the above procedures are optimal, then we expect a lower bound of $\alpha^2 = \Omega\left(\min\left\{\frac{1}{\rho n}, \frac{d}{(\rho n)^2}\right\}\right)$. Equivalently, for a fixed accuracy α, we expect a sample complexity lower bound of $n = \Omega\left(\min\left\{\frac{1}{\rho\alpha^2}, \frac{\sqrt{d}}{\rho\alpha}\right\}\right)$.

C.9.1.1 Lower Bound I

In this section, we give a $\Omega\left(\frac{1}{\alpha\rho}\right)$ lower bound on sample complexity. The key ingredient is the following result, where the proof is based on a simple reduction argument.

Proposition 11. *Suppose there exists a ρ-TV-stable algorithm such that for any dataset of n points, it achieves an accuracy of α. Then there exists a 0.1-TV stable algorithm such the for any dataset of size $\lceil 100n\alpha\rho \rceil$, it achieves a 0.1-accuracy.*

Proof of Proposition 11. Let $n' = \lceil 100n\alpha\rho \rceil$. Consider a dataset S' of n' points. We construct a dataset S of n points by concatenating $K = \lceil 0.1/\rho \rceil$ copies of S' followed

by $\lceil \frac{n-Kn'}{2} \rceil$ copies of a constant sample, all ones $(\frac{1}{\sqrt{d}}, ..., \frac{1}{\sqrt{d}})$ and $\lceil \frac{n-Kn'}{2} \rceil$ copies of a constant sample, all ones $(-\frac{1}{\sqrt{d}}, ..., -\frac{1}{\sqrt{d}})$.

Consider the algorithm wherein we compute the stable-mean on D' by $\mathcal{A}'$, defined as computing the stable-mean on D using $\mathcal{A}$ and adjusting:

$$\mathcal{A}'(D') = \frac{n}{Kn'}\mathcal{A}(D)$$

Let $\widetilde{S}'$ be a neighbouring dataset of S'. By construction, note that S and $\widetilde{S}$ differ by K samples. Furthermore, since the algorithm $\mathcal{A}$ on D is ρ-TV stable, on K-neighbouring datasets, it is $K\rho = 0.1$-TV. This establish the stability part of claim. The accuracy, by direct computation is $\mathbb{E}\left[\|\mathcal{A}'(D') - \mu(D')\|^2\right] = \frac{n^2}{K^2n'^2}\mathbb{E}\left[\|\mathcal{A}(D) - \mu(D)\|^2\right] \leq (0.1)^2$. □

Theorem 46. *For the d-dimensional mean computation problem over the Euclidean ball, there exists a dataset S of n samples with mean $\|\mu(S)\| = \Theta\left(\frac{1}{\rho n}\right)$ such that the accuracy of any ρ TV stable algorithm is $\alpha \geq \Omega\left(\frac{1}{\rho n}\right)$.*

Proof of Theorem 46. Even for accuracy $\alpha_0 = 0.1$ accuracy and $\rho_0 = 0.1$ stability, we need at least one sample. Hence, using Proposition 10, we get that sample complexity is $n \geq \Omega\left(\frac{1}{\rho\alpha}\right)$, which equivalently gives the claimed accuracy lower bound. Note that for this one-sample dataset S', $\|\mu(S')\| = 1$. Finally, from the reduction in Proposition 10, the mean of dataset S becomes $\|\mu(S)\| = \lceil \frac{0.1}{\rho n} \rceil$, which finishes the proof. □

C.9.1.2 Lower Bound II

In this section, we will prove the $\Omega\left(\frac{1}{\alpha^2\rho}\right)$ lower bound. We first introduce a technical assumption.

Assumption 3. *For any dataset S, we assume that the probability distribution $\mathcal{A}(S)$ is defined over the unit Euclidean ball, is absolutely continuous with respect to the*

uniform measure (in the unit Euclidean ball) and its probability density function, with respect to the uniform measure, is bounded by K in absolute value.

As a remark, the above assumption can also be stated with respect to the Lebesgue measure, but then we would get a scaling of $\frac{\pi^{d/2}}{\Gamma\left(1+\frac{d}{2}\right)}$, which is the Lebesgue volume of the $B_d(0,1)$, to some of our terms. In order to simplify, we therefore use the uniform measure.

Theorem 47. *Let $n \geq 72, \alpha \leq \frac{1}{4}$ and $\frac{1}{n} \leq \rho \leq \frac{1}{4}$. Let $\mathcal{A}$ be any ρ-TV-stable algorithm satisfying Assumption 3 with $K \leq 2^d$. For large enough dimension d, there exists a dataset S of n points with $\|\mu(S)\| = \Theta\left(\frac{1}{\sqrt{\rho n}}\right)$ such that accuracy is lower bounded as $\alpha \geq \Omega\left(\frac{1}{\sqrt{\rho n}}\right)$.*

Proof of Theorem 47. We will prove the result by contradiction. Let "Vol" of a set refer to its volume with respect to the uniform measure on the unit ball. Consider the following high-dimensional setup. Consider a dataset S (or S^0) which mean $\mu(S)$ such that $\|\mu(S)\| = \Theta\left(\frac{1}{\sqrt{\rho n}}\right)$. It is easy to construct such datasets by considering points such that sum of $n - \lceil\sqrt{\frac{n}{\rho}}\rceil$ points is 0 and the rest of points is the same point repeated - this uses the assumption that $\rho \geq \frac{1}{n}$. Now consider neighbouring datasets S^i's, $i \in [n]$ such that the means of S^i's are all $\frac{1}{n}$ far from that of S, in norm. We also need that the means of any two datasets $\|\mu(S^i) - \mu(S^j)\| \geq \frac{1}{2n}$ for $i, j = 0$ to n and $i \neq j$. It is easy to see the existence of such datasets, by considering the means of S^i's in *near* orthogonal directions to that of S, which is possible when d is large enough.

Suppose the algorithm $\mathcal{A}$ has expected error α^2 i.e. $\mathbb{E}\left[\|\mathcal{A}(S) - \mu(S)\|\right]^2 \leq \alpha^2$ with $72 \leq n \leq \frac{1}{\alpha^2\rho}$. Consider $B_d\left(\mu(S), \frac{1}{K^{1/d}}\right)$, the d dimensional Euclidean ball centered at

$\mu(S)$ of radius $\frac{1}{K^{1/d}}$. From Markov's inequality, we have that

$$\begin{aligned}\mathbb{P}\left[\mathcal{A}(S) \notin B_d\left(\mu(S), \frac{1}{K^{1/d}}\right)\right] &\leq \mathbb{P}\left[\mathcal{A}(S) \notin B_d\left(\mu(S), \frac{1}{2}\right)\right] \\ &= \mathbb{P}\left[\|\mathcal{A}(S) - \mu(S)\|^2 \geq \frac{1}{4}\right] \\ &\leq \frac{\mathbb{E}\left[\|\mathcal{A}(S) - \mu(S)\|\right]^2}{(1/4)} \\ &\leq 4\alpha^2\end{aligned}$$

, where in the first inequality, we used the assumption $K \leq 2^d$. Therefore, we have $\mathbb{P}\left[\mathcal{A}(S) \in B_d\left(\mu(S), \frac{1}{K^{1/d}}\right)\right] \geq 1 - 4\alpha^2$.

We now setup some additional notation. Let A_i denote the set $B_d\left(\mu(S^i), \frac{1}{K^{1/d}}\right) \setminus \left(\cup_{j=0, j\neq i}^{n} B_d\left(\mu(S^j), \frac{1}{K^{1/d}}\right)\right)$ i.e. the region in the ball $B_d(\mu(S^i)$ which is not contained in any of the other balls. Let B_{ij} denote the region of intersection between $B_d\left(\mu(S^i), \frac{1}{K^{1/d}}\right)$ and $B_d\left(\mu(S^j), \frac{1}{K^{1/d}}\right)$ where $i \neq j$ and i and j go from 0 to n. Note that set B_{ij} is constituted of two spherical caps. By construction the centers of the intersecting spheres are at least $\frac{1}{2n}$ apart. To study the properties of such a set, we define **cap** as the region in a d dimensional sphere of radius $\frac{1}{K^{1/d}}$ which intersects with another sphere of the same radius but with centers being apart by $1/2n$. From known results [Chu91], the volume of **cap** is asymptotic to $\text{Vol}\left(B_d\left(0, \frac{1}{K^{1/d}}\right)\right)\left(1 - \Phi\left(\frac{\sqrt{d}}{2n}\right)\right)$ as $d \to \infty$ where Φ is the cumulative distribution function of a standard normal random variable. We therefore have that $\lim_{d\to\infty} \text{Vol}(\textbf{cap}) \sim \lim_{d\to\infty} \text{Vol}\left(B_d\left(0, \frac{1}{K^{1/d}}\right)\right)\left(1 - \Phi\left(\frac{\sqrt{d}}{2n}\right)\right) = 0$. Furthermore, using Assumption 3, we have $\mathbb{P}\left[\mathcal{A}(S) \in \text{cap}\right] \leq K\text{Vol}(\textbf{cap}) \sim K\text{Vol}\left(B_d\left(0, \frac{1}{K^{1/d}}\right)\right)\left(1 - \Phi\left(\frac{\sqrt{d}}{n}\right)\right) \leq K\left(\frac{1}{K^{1/d}}\right)^d\left(1 - \Phi\left(\frac{\sqrt{d}}{n}\right)\right) = \left(1 - \Phi\left(\frac{\sqrt{d}}{n}\right)\right)$ as $d \to \infty$. Since $\Phi(t) = \mathbb{P}_{g\sim\mathcal{N}(0,1)}[g \leq t]$, we have that $1 - \Phi(t) = \mathbb{P}_{g\sim\mathcal{N}(0,1)}[g > t] \leq \frac{e^{-t^2/2}}{\sqrt{2\pi}t}$ where the last inequality follows from standard bounds on tails of normal distribution (See Proposition 2.1.2 in [Ver18]). Therefore, we have $\mathbb{P}\left[\mathcal{A}(S) \in \textbf{cap}\right] = O\left(\frac{ne^{-d/4n^2}}{\sqrt{d}}\right)$. For constant $\epsilon > 0$, choosing $d = \Omega\left(4n^2 \ln\left(n/\epsilon\right)\right)$ ensures that $\mathbb{P}\left[\mathcal{A}(S) \in \textbf{cap}\right] \leq \frac{\epsilon}{2n}$ for large enough n (to be specified later). Since B_{ij} is made up of two conjoined caps,

this gives us that for any $j = 0$ to n and $i \neq j$, we have that $\mathbb{P}\left[\mathcal{A}(S^i) \in B_{ij}\right] \leq \frac{\epsilon}{n}$. Finally, we look at A_i's by removing the mass of all B_{ij}'s, and using a union bound, we get that

$$\begin{aligned}\mathbb{P}\left[\mathcal{A}(S^i) \in A_i\right] &= \mathbb{P}\left[\mathcal{A}(S^i) \in B_d\left(\mu(S^i), \frac{1}{K^{1/d}}\right)\right] - \mathbb{P}\left[\mathcal{A}(S^i) \in \cup_{j=0, j\neq i}^{n} B_{ij}\right] \\ &\geq 1 - 4\alpha^2 - \sum_{j=0, j\neq i}^{n} \mathbb{P}\left[\mathcal{A}(S^i) \in B_{ij}\right] \geq 1 - 4\alpha^2 - \epsilon \geq \frac{1}{2} - 4\alpha^2\end{aligned}$$

where the last inequality holds for $\epsilon \leq \frac{1}{2}$. We now evaluate how large n we need for this regime of ϵ: recall that we set $d = \Omega\left(4n^2 \ln\left(n/\epsilon\right)\right)$, this gives $\frac{ne^{-d/4n^2}}{\sqrt{d}} \leq \frac{\epsilon}{2n\sqrt{\ln\left(n/\epsilon\right)}}$. We want the right hand side to be at most $\frac{\epsilon}{2n}$ for $\epsilon \leq 1/2$. Plugging in this worst-case value of ϵ, we get the condition $\ln\left(2n\right) \geq 1$ which holds for any $n \geq 1.4$ and therefore is valid by our assumption of n.

We now use the fact that $A_i's$ are disjoint by construction. Therefore the total measure of $\mathcal{A}(S)$ on union of $A_i's$ is at most 1 i.e $\mathbb{P}\left[\mathcal{A}(S) \in \cup_{i=1}^{n} A_i\right] = \sum_{i=1}^{n} \mathbb{P}\left[\mathcal{A}(S) \in A_i\right] \leq 1$. Furthermore, since $\mathcal{A}(S)$ is ρ-TV stable, we have that $\mathbb{P}\left[\mathcal{A}(S) \in A_i\right] \geq \mathbb{P}\left[\mathcal{A}(S^i) \in A_i\right] - \rho$. Combining this with the previous analysis which gives a lower bound on $\mathbb{P}\left[\mathcal{A}(S^i) \in A_i\right]$ yields

$$n\left(\frac{1}{2} - 4\alpha^2 - \rho\right) \leq \sum_{i=1}^{n} \mathbb{P}\left[\mathcal{A}(S^i) \in A_i\right] - \rho \leq \sum_{i=1}^{n} \mathbb{P}\left[\mathcal{A}(S) \in A_i\right] \leq 1 \tag{C.1}$$

We now proceed in two cases:

Case 1. Suppose $72 \leq n \leq \frac{17}{4\alpha^2}$. The latter condition gives us that $4\alpha^2 \leq \frac{17}{n}$. Using Eq. (C.1) gives us $n(1/2 - 4\alpha^2 - \rho) \leq 1 \iff 4\alpha^2 \geq \frac{1}{2} - \frac{1}{n} - \rho$. Upper bounding $4\alpha^2$ by $\frac{17}{n}$ gives us that $\frac{18}{n} \geq \frac{1}{2} - \rho \iff n \leq \frac{18}{(1/2-\rho)} \leq 72$ where in the last inequality we used $\rho \leq 1/4$. This gives us a contradiction.

Case 2. Suppose $\frac{17}{4\alpha^2} \leq n$. We again start with Eq. (C.1) which gives us $n(1/2 - 4\alpha^2 - \rho) \leq 1 \iff \rho \geq \frac{1}{2} - \frac{1}{n} - 4\alpha^2$. We want to prove the right hand side is at least $\frac{1}{n\alpha^2}$, which would give us that $n \geq \frac{1}{\rho\alpha^2}$. Suppose this is not true i.e. $\frac{1}{2} - \frac{1}{n} - 4\alpha^2 \leq \frac{1}{n\alpha^2} \iff \frac{n-2-8\alpha^2 n}{2n} \leq \frac{1}{n\alpha^2} \iff \alpha^2 n(1 - 8\alpha^2) \leq 2(1 + \alpha^2) \iff n \leq$

$\frac{2(1+\alpha^2)}{\alpha^2(1-8\alpha^2)}$. Finally using the fact that $\alpha \leq \frac{1}{4}$ gives that $n \leq \frac{2(1+1/16)}{\alpha^2(1-8/16)} \leq \frac{17}{4\alpha^2}$ which yields a contradiction.

Hence, we see that with $n > 72$ samples and accuracy α^2, we have established that $n \geq \frac{1}{\rho\alpha^2}$ and so $\alpha \geq \frac{1}{\sqrt{\rho n}}$. □

C.10 Excess Population Risk Bounds

In this section, consider the error criterion as the population risk, denoted as $L(w; \mathcal{D})$. We assume that the edits requests are independent of the dataset, so that the resulting current dataset is still independently and identically distributed.

C.10.1 Upper Bounds

In this section, we will bound the expected excess population risk by appealing to connections between algorithmic stability and generalization [BE02].

Theorem 48 (Upper bound)**.** *There exists a ρ TV stable algorithm, such that for any function $f(\cdot, z)$ which is H-smooth G-Lipschitz convex $\forall$ z and any dataset S of n points, it outputs $\widehat{w}_S$ which satisfies the following.*

$$\mathbb{E}\left[L(\widehat{w}_S; \mathcal{D}) - L(w^*; \mathcal{D})\right] = O\left(\frac{GD}{\sqrt{n}} + GD \min\left\{\frac{1}{\sqrt{\rho n}}, \frac{\sqrt{d}}{\rho n}\right\}\right).$$

Proof of Theorem 48. We use *sub-sample-GD* (Algorithm 19) and *noisy-m-SGD* (Algorithm 21). From Lemma 3.2 in [BFTT19], we have that $\alpha_{\text{stable}}(n) \leq \frac{G^2\eta T}{n}$. From Proposition 3, we set $\eta = \min\left\{\frac{1}{2H}, \frac{D}{G\sqrt{T}}\right\}$, and $T = \frac{DH\sqrt{\rho n}}{G}$. We therefore have the uniform stability parameter $\alpha_{\text{stable}}(n) \leq \frac{G^2 T}{2Hn} = \frac{GD\sqrt{\rho}}{\sqrt{\rho n}}$. Using the excess empirical risk bound from Proposition 3, and the fact that $\rho \leq 1$, the excess population risk is bounded as,

$$\mathbb{E}\left[L(\widehat{w}_S; \mathcal{D}) - L(w^*; \mathcal{D})\right] \leq \frac{GD\sqrt{\rho}}{\sqrt{n}} + \frac{GD}{\sqrt{\rho n}} \leq \frac{GD}{\sqrt{n}} + \frac{GD}{\sqrt{\rho n}}$$

For *noisy-m-SGD*, we need to balance the trade-offs more directly. In Proposition 7, we arrived at that when using $\eta \leq \frac{1}{2L}$, the expected excess empirical risk is bounded by $\eta\mathcal{V}^2 + \frac{D^2}{\eta T}$. Using the uniform stability bound of $\frac{G^2T\eta}{n}$, the expected excess population risk is bounded as,

$$\mathbb{E}\left[L(\widehat{w}_S;\mathcal{D}) - L(w^*;\mathcal{D})\right] \leq \frac{G^2T\eta}{n} + \eta\mathcal{V}^2 + \frac{D^2}{\eta T} = \eta\left(\frac{G^2T}{n} + \mathcal{V}^2\right) + \frac{D^2}{\eta T}$$

Define $\widetilde{G}^2 = \left(\frac{G^2T}{n} + \mathcal{V}^2\right)$, where $\mathcal{V}^2 = O\left(G^2 + \sigma^2 d\right) = O\left(G^2 + \frac{G^2Td}{n^2\rho}\right)$. Setting $\eta = \min\left\{\frac{1}{2H}, \frac{D}{\widetilde{G}\sqrt{T}}\right\}$, we get,

$$\begin{aligned}
\mathbb{E}\left[L(\widehat{w}_S;\mathcal{D}) - L(w^*;\mathcal{D})\right] &\leq \frac{HD^2}{T} + \frac{\widetilde{G}D}{\sqrt{T}} \\
&= O\left(\frac{HD^2}{T} + \frac{G\sqrt{T}D}{\sqrt{T}\sqrt{n}} + \frac{GD}{\sqrt{T}} + \frac{GD\sqrt{T}\sqrt{d}}{\rho n\sqrt{T}}\right) \\
&= O\left(\frac{HD^2}{T} + \frac{GD}{\sqrt{n}} + \frac{GD}{\sqrt{T}} + \frac{GD\sqrt{d}}{\rho n}\right).
\end{aligned}$$

Setting $T = \max\left\{\min\left\{\sqrt{n}, \frac{\rho n}{\sqrt{d}}\right\}, \frac{HD}{G}\min\left\{\sqrt{n}, \frac{\rho n}{\sqrt{d}}\right\}\right\}$, and combining the two results finishes the proof. □

C.10.2 Lower Bounds

In this section, we will prove a lower bound on excess population risk for any ρ-TV stable algorithm. As before, we will consider the Lipschitz constant G and diameter D to be both 1, as the bounds scale naturally with these constants. We first define the following quantity, which denotes the lower bound on expected excess empirical risk of ρ-TV-stable algorithm with n points.

$$\widehat{\alpha}(n,\rho) := \inf_{\mathcal{A}:\rho\text{-TV-stable}} \sup_{S:|S|=n} \mathbb{E}_{\mathcal{A}}\widehat{L}(\mathcal{A}(S);S) - \widehat{L}(\widehat{w}_S;S)$$

Theorem 49. *For the problem of stochastic convex optimization, there exists a data distribution $\mathcal{D}$, such that any ρ-TV-stable algorithm $\mathcal{A}$ incurs expected excess population risk, bounded as follows*

$$\mathop{\mathbb{E}}_{S\sim\mathcal{D}^n,\mathcal{A}}\left[L(\mathcal{A}(S);\mathcal{D}) - L(w^*;\mathcal{D})\right] \geq \max\left\{\Omega\left(\frac{1}{\sqrt{n}}\right), \widehat{\alpha}(n,\rho)\right\}$$

Proof of Theorem 49. The $\frac{1}{\sqrt{n}}$ term follows directly since it is the lower bound for any algorithm, and so applies to ρ-TV stable algorithms as well. We now focus on the second term $\widehat{\alpha}(n,\rho)$ The proof is based on a standard reduction argument: if there is ρ-TV stable algorithm, which with n i.i.d. samples from any distribution, achieves an expected excess population risk less than $\widehat{\alpha}(n,\rho)$, then there is an ρ-TV stable algorithm which achieves an expected excess empirical risk less than $\widehat{\alpha}(n,\rho)$ on any dataset of n samples. Since the latter contradicts the definition of $\widehat{\alpha}(n,\rho)$, this gives us that the expected excess population risk is at least or equal to $\widehat{\alpha}(n,\rho)$. We now focus on the proof of the reduction. Consider a dataset S of n points. Consider $\mathcal{A}$ as the following algorithm: sample n i.i.d. samples from S, call this set $\widetilde{S}$, and run *some* ρ-TV algorithm $\widetilde{\mathcal{A}}$ on $\widetilde{S}$. For a fixed $\widetilde{S}$, from TV-stable property of $\widetilde{\mathcal{A}}$, for any neighbouring dataset $\widetilde{S}'$ with one point differing, we have that $\mathrm{TV}(\widetilde{\mathcal{A}}(\widetilde{S}),\widetilde{\mathcal{A}}(\widetilde{S}')) \leq \rho$. Furthermore, using the *group* property of TV-stability, for any dataset $\widetilde{S}'$, we have $\mathrm{TV}(\widetilde{\mathcal{A}}(\widetilde{S}),\widetilde{\mathcal{A}}(\widetilde{S}')) \leq \Delta(S,S')\rho$. Using the maximal coupling characterization of total variation distance, we have that there exists a coupling $\widetilde{\pi}$ of random variables $\widetilde{\mathcal{A}}(\widetilde{S})$ and $\widetilde{\mathcal{A}}(\widetilde{S}')$ such that $\mathrm{TV}(\widetilde{\mathcal{A}}(\widetilde{S}),\widetilde{\mathcal{A}}(\widetilde{S}')) = \mathbb{E}_{\widetilde{\pi}}\mathbb{1}\left\{\widetilde{\mathcal{A}}(\widetilde{S}) \neq \widetilde{\mathcal{A}}(\widetilde{S}')\right\}$. We now show that the algorithm $\mathcal{A}$ is also TV-stable for dataset S.

Consider dataset S' of n points which differs from S in the first sample. We now generate $\widetilde{S}'$ by drawing n i.i.d samples from S'. For this, consider the following coupling: we draw n i.i.d samples from S, call it $\widetilde{S}$. For every draw of the first sample, replace it by the first sample of S', call it $\widetilde{S}'$. It is easy to check the $\widetilde{S}$ and $\widetilde{S}'$ are i.i.d. samples from S and S' respectively. We now proceed to show the $\mathcal{A}$ is ρ-TV stable. We will use the fact the total variation distance is at most the probability of disagreement under any coupling. The coupling π we consider is that we first generate $\widetilde{S}$ and $\widetilde{S}'$ using the aforementioned coupling, and then use the coupling $\widetilde{\pi}$ which achieves total variation distance for worst-case fixed neighbouring datasets $\widetilde{S}$

and $\widetilde{S}'$. We have,

$$\begin{aligned}\mathrm{TV}(\mathcal{A}(S), \mathcal{A}(S')) &\leq \mathbb{E}_\pi \mathbb{1}\left\{\mathcal{A}(S) \neq \mathcal{A}(S')\right\} = \mathbb{E}_\pi \mathbb{1}\left\{\widetilde{\mathcal{A}}(\widetilde{S}) \neq \widetilde{\mathcal{A}}(\widetilde{S}')\right\} \\ &\leq \mathbb{E}_\pi \sup_{\widetilde{S},\widetilde{S}'} \mathbb{1}\left\{\widetilde{\mathcal{A}}(\widetilde{S}) \neq \widetilde{\mathcal{A}}(\widetilde{S}')\right\} \leq \mathbb{E}_\pi \Delta(\widetilde{S}, \widetilde{S}')\rho = \rho\end{aligned}$$

where the last equality follows from direct computation of $\Delta(\widetilde{S}, \widetilde{S}')$: number of differing samples, under coupling π.

We now proceed to the accuracy guarantee. From a straight-forward computation, the excess population risk, under the sampling of $\widetilde{S}$, is $\widehat{L}(\widetilde{\mathcal{A}}(\widetilde{S}); S) - \widehat{L}(w_S^*; S)$ - this is the excess empirical risk for dataset S. So if we have an upper bound on excess population risk using algorithm $\widetilde{\mathcal{A}}$, we have an upper bound on excess empirical risk for dataset S. This completes the reduction argument and hence the proof. □

C.11 Algorithms for Approximate Unlearning

We consider an approximate notion of unlearning, based on differential privacy, which has appeared in the literature [NRSM21a, GGHVDM20]. With such a notion, we show a simple black-box reduction to DP algorithm, to handle unlearning requests, and show how to use *group privacy* to trade-off accuracy and runtime. For convex ERM, this method performs competitively with existing works.

We first define the notion of approximate unlearning.

Definition 21 ((ϵ, δ)-approximate-unlearning)**.** *We say a procedure* $(\mathbf{A}, \mathbf{U})$ *satisfies* (ϵ, δ)*-approximate-unlearning unlearning if for any* $S, S' \subset \mathcal{X}^*$ *such that* $\Delta(S, S') = 1$ *and for any measurable event* $\mathcal{E} \in$ *Range*$(\mathbf{U}) \cap$ *Range*$(\mathbf{A})$*, with probability at least* $1 - \delta$*,*

$$\mathrm{e}^{-\epsilon}\mathbb{P}\left[\mathbf{U}(\mathbf{A}(S), S' \backslash S \cup S \backslash S') \in \mathcal{E}\right] \leq \mathbb{P}\left[\mathbf{A}(S') \in \mathcal{E}\right] \leq e^{\epsilon}\mathbb{P}\left[\mathbf{U}(\mathbf{A}(S), S' \backslash S \cup S \backslash S') \in \mathcal{E}\right]$$

We now define (g, ϵ, δ)-group differential privacy.

Definition 22 ((g, ϵ, δ)-group differential privacy)**.** *An algorithm $\mathcal{A}$ satisfies (ϵ, δ)-differential privacy if for any two datasets S and S' such that $\Delta(S, S') \leq g$, for any measurable event $\mathcal{E} \in Range(\mathcal{A})$, it satisfies*

$$\mathbb{P}(\mathcal{A}(S) \in \mathcal{E}) \leq e^{\epsilon}\mathbb{P}(\mathcal{A}(S') \in \mathcal{E}) + \delta$$

Remark 7. *[DR+ 14] If an algorithm satisfies (ϵ, δ)-DP, then for any $g \in \mathbb{N}$, it satisfies $(g, g\epsilon, ge^{(g-1)\epsilon}\delta)$-group differential privacy.*

We now define *privateCompute* oracle which, basically is a differentially private solver for the said task.

Definition 23 (*privateCompute*(S, ϵ, δ) oracle)**.** *For a problem instance, given a dataset S of n points, and privacy parameters ϵ and δ, a* privateCompute *oracle outputs a (ϵ, δ)-differentially private solution with accuracy $\alpha_{private}(n, \epsilon, \delta)$*

We now give a very simple algorithm (Algorithm 25) based on the observation above using *privateCompute* oracle calls.

Algorithm 25 Approximate unlearning

Input: ϵ, δ, ρ

$i \leftarrow 0$

$\widehat{w}_S \leftarrow PrivateCompute\left(S, \rho\alpha, \beta_{\text{group}}\left(\left\lfloor \frac{1}{\rho} \right\rfloor, \rho\alpha, \rho\beta\right)\right)$

// Observe k edit requests

while $t = 1, 2, \ldots, k$ **do**

  $S_t \leftarrow$ Update dataset(edit request)

  $i+ = 1$

  if $i = \left\lfloor \frac{1}{\rho} \right\rfloor$ **then**

    $\widehat{w}_{S_t} \leftarrow PrivateCompute\left(S_t, \rho\alpha, \beta_{\text{group}}\left(\left\lfloor \frac{1}{\rho} \right\rfloor, \rho\alpha, \rho\beta\right)\right)$

    $i \leftarrow 0$

  end if

end while

Theorem 50. *Given a set of n data points to start with, and observing a stream of k requests, at any time t in the stream, the following hold about Algorithm 25:*

1. *It satisfies (ϵ, δ)-approximate unlearning.*

2. *The unlearning runtime for k requests is at most $2\rho k$ privateCompute oracle calls.*

3. *The accuracy is at most $\alpha_{private}\left(\frac{n}{2}, \rho\epsilon, \delta_{group}\left(\left\lfloor\frac{1}{\rho}\right\rfloor, \rho\epsilon, \rho\delta\right)\right)$.*

Proof of Theorem 50. Consider a point t in the stream, and let j be such that $j\left\lfloor\frac{1}{\rho}\right\rfloor \leq t \leq (j+1)\left\lfloor\frac{1}{\rho}\right\rfloor$. Since the algorithm uses *privateCompute* with parameters $\rho\epsilon$ and $\delta_{\text{group}}\left(\left\lfloor\frac{1}{\rho}\right\rfloor, \rho\epsilon, \rho\delta\right)$, it satisfies $(\rho\epsilon, \delta_{\text{group}}\left(\left\lfloor\frac{1}{\rho}\right\rfloor, \rho\epsilon, \rho\delta\right)$ differential privacy and hence $\left(\left\lfloor\frac{1}{\rho}\right\rfloor, \alpha, \delta\right)$-group privacy. Therefore, for any such t, since the number of requests after time $j\left\lfloor\frac{1}{\rho}\right\rfloor$ is less that or equal to $\left\lfloor\frac{1}{\rho}\right\rfloor$, this implies it satisfies (α, δ)-approximate unlearning. For the second part of the claim, note that for k updates, the number of times the algorithm calls *privateCompute* is $\frac{k}{\left\lfloor\frac{1}{\rho}\right\rfloor}$. Note that $\frac{1}{\rho} \geq 1$, so if $1 \leq \frac{1}{\rho} < 2$, then $\left\lfloor\frac{1}{\rho}\right\rfloor = 1$, which gives that the update complexity is $k \leq 2k$. However, if $\frac{1}{\rho} \geq 2$, we have that $\frac{k}{\left\lfloor\frac{1}{\rho}\right\rfloor} \leq \frac{k}{\left(\frac{1}{\rho}-1\right)} \leq 2k\rho$, which gives the update complexity is at most $2\rho k$ in both cases. For the third part of the claim, at time $j\left\lfloor\frac{1}{\rho}\right\rfloor \leq t \leq (j+1)\left\lfloor\frac{1}{\rho}\right\rfloor$, the private estimator is computed with $n\left(j\left\lfloor\frac{1}{\rho}\right\rfloor\right)) \geq \frac{n}{2}$, by assumption. Moreover the privacy parameters of the algorithm are $\rho\epsilon$ and $\delta_{\text{group}}\left(\left\lfloor\frac{1}{\rho}\right\rfloor, \rho\epsilon, \rho\delta\right)$ which gives the claimed accuracy bound. □

As an example, consider $\rho = \frac{1}{\sqrt{k}}$, we first do *privatecompute* with parameters $(\epsilon/\sqrt{k}, , \delta_{\text{group}}(\epsilon/\sqrt{k}, \delta/\sqrt{k}, \lfloor\sqrt{k}\rfloor))$. Since after $\lfloor\sqrt{k}\rfloor$ edit requests, we would no longer satisfy the unlearning guarantee, so we now need to do *privateCompute* again. However note that we would only need to do *privateCompute* $\sqrt{k}$ times which gives the update computation cost.

Example: Convex ERM. For convex ERM, we can use [BST14] to instantiate the oracle. In this case, accuracy α_{private} is the excepted excess empirical risk, which

is $\alpha_{\text{private}}(n, \epsilon, \delta) = O\left(\frac{GD\sqrt{d}\sqrt{\log(1/\delta)}}{n\epsilon}\right)$. Using Algorithm 25, given $0 \leq \rho \leq 1$, at any point in the stream, we have,

$$\begin{aligned}
\mathbb{E}\left[\widehat{L}(\widehat{w}_S; S) - \widehat{L}(w^*_S; S)\right] &\leq \alpha_{\text{private}}\left(\frac{n}{2}, \rho\epsilon, \delta_{\text{group}}\left(\left\lfloor\frac{1}{\rho}\right\rfloor, \rho\epsilon, \rho\delta\right)\right) \\
&\leq O\left(\frac{GD\sqrt{d}\sqrt{\log\left((1/\rho\delta)\exp\left(\rho\epsilon\left(\left\lfloor\frac{1}{\rho}\right\rfloor - 1\right)\right)\right)}}{n\rho\epsilon}\right) \\
&\leq O\left(GD\left(\frac{\sqrt{d}\sqrt{\log(1/\rho\delta)}}{n\rho\epsilon} + \frac{\sqrt{d}}{n\rho\sqrt{\epsilon}}\right)\right) \\
&\leq O\left(\frac{GD\sqrt{d}\sqrt{\log(1/\rho\delta)}}{n\rho\epsilon}\right)
\end{aligned}$$

where the last inequality holds when $\frac{\epsilon}{\log(1/\rho\delta)} \leq O(1)$, which usually is the case in DP, and so is a reasonable regime. We now compare against [NRSM21a] - we ignore G, D and log factor in both the bounds. To have the same runtime, we need $\rho kTm = k^2 n \iff \rho = \frac{kn}{Tm} = \frac{kd}{\epsilon^2 n}$, where in the last equality we substituted $Tm = \frac{(\epsilon n)^2}{d}$, parameters for the DP convex ERM algorithm. Our accuracy bound is $O\left(\frac{\sqrt{d}}{n\rho\epsilon}\right) = O\left(\frac{\epsilon}{k\sqrt{d}}\right)$, which is smaller than that of [NRSM21a], when $\frac{\epsilon}{k\sqrt{d}} \leq \left(\frac{\sqrt{d}}{nk\epsilon}\right)^{2/5} \iff \epsilon^7 \leq \frac{\sqrt{d}^7 k^3}{n^2} \iff \epsilon \leq \frac{\sqrt{d}k^{3/7}}{n^{2/7}}$. Hence in regimes where the unlearning parameter ϵ is small enough, which corresponds to a stronger unlearning criterion, this algorithm is better than that of [NRSM21a].

C.12 Experiments

We run experiments on MNIST [LeC98], a standard digit classification computer vision dataset with 10 classes. We train a logistic regression model, which can be formulated as a smooth convex risk minimization problem. Starting with a training dataset of 60k points, we simulate a stream of 300 deletions of randomly chosen points and 300 insertions of new points, randomly permuted. We use Algorithm 21 as the learning algorithm, and the corresponding Algorithm 22 as the unlearning algorithm. We train

for $T = 200$ iterations, with mini-batch of size $m = 50$ with a constant learning rate $\eta = 0.05$. We run experiments on a range of values of standard deviation σ of Gaussian noise, from 0 to 1.1 separated by the intervals of size 0.005. For every value of σ, we run 10 instances of the whole unlearning procedure and report average performance: accuracy and number of unstable edits (i.e. number of times a recompute is triggered) , and their standard deviations. Note that $\sigma = 0$ corresponds to standard mini-batch SGD, and therefore the accuracy obtained is the accuracy for the standard training method with the aforementioned setting of the hyperparameters. Moreover, the $\sigma = 0$ setting also corresponds to Algorithm 19, and therefore the corresponding unlearning algorithm Algorithm 20 handles edits for this case.

In Fig. C-2, we report the test accuracy (fraction of mis-classified samples in the test set) and the number of unstable edits i.e the number of times a retrain is triggered, as a function of σ. As expected, as σ increases, we get less unstable edits. Interestingly, for small values of σ, for example 0.05, the degradation in accuracy: 88% (vanilla SGD) to 87% (our method) is not as much as compared to decrease in the the number of unstable edits: 520 to 210 i.e. a factor of two and a half. Apart from the above improvement, there is an additional improvement from the fact that our method triggers a partial retraining rather than full retraining i.e. only some of the iterations of SGD are to be rerun. This behaviour is shown in Fig. C-3 - the orange line shows the average number of iterations we need to do if we did full retraining on every unstable edit, and the blue line shows the number of iterations for our method. For $\sigma = 0.05$, we see an advantage of a factor of two. Hence, our method with $\sigma = 0.05$ has accuracy comparable to retraining (87% vs 88%) but the computational advantage, in terms of number of iterations rerun, over retraining, is about a factor of five. Similar conclusions can be drawn by looking at different values of σ in the plots which show a smooth trade off between accuracy and unlearning efficiency.

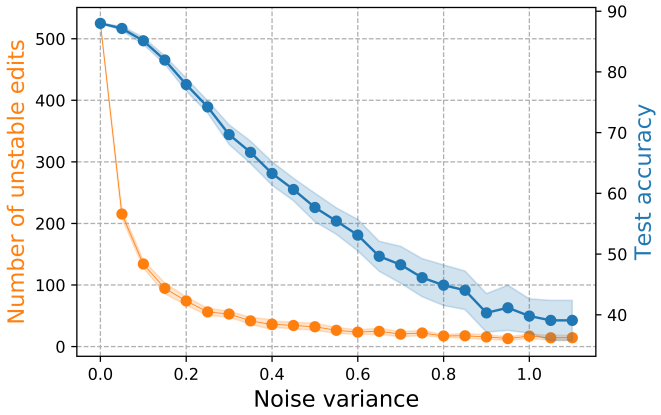

Figure C-2. Accuracy and number of unstable edits as a function of variance of noise used.

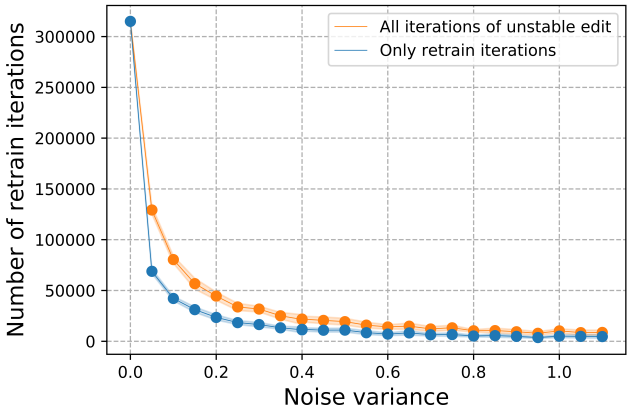

Figure C-3. Number of retraining iterations by unlearning algorithm compared to all full retraining (all iterations)

Appendix D

Appendix for Chapter 5

D.1 Auxiliary Results

We recall some concepts from differential privacy which will be useful in our algorithmic techniques.

Definition 24. *An algorithm $\mathcal{A}$ satisfies $(\alpha, \epsilon(\alpha))$-Rényi Differential Privacy (RDP), if for any two datasets S and S' which differ in one data point ($|S \Delta S'| = 1$), the α-Rényi Divergence between $\mathcal{A}(S)$ and $\mathcal{A}(S')$, with probability densities $\phi_{\mathcal{A}(S)}$ and $\phi_{\mathcal{A}(S')}$, defined as follows:*

$$D_\alpha\left(\mathcal{A}(S) \| \mathcal{A}(S')\right) = \frac{1}{\alpha - 1} \ln \left(\int_{Range(\mathcal{A})} \phi_{\mathcal{A}(S)}(x)^\alpha \phi_{\mathcal{A}(S')}(x)^{1-\alpha} dx \right)$$

is bounded as, $D_\alpha(\mathcal{A}(S) \| \mathcal{A}(S')) \leq \epsilon(\alpha)$.

RDP satisfies many desirable properties such as adaptive and parallel composition and amplification by sub-sampling [Mir17, WBK19]. Furthermore, we give the following lemma which relates TV stability to RDP.

Lemma 33 (RDP $\implies$ TV-stability)**.** *If an algorithm satisfies $(\alpha, \epsilon(\alpha))$-RDP, then it satisfies $\left(1 - \exp\left(-\lim_{\alpha \downarrow 1} \epsilon(\alpha)\right)\right)^{\frac{1}{2}}$-TV stability.*

Proof of Lemma 33. From Theorem 4 in [VEH14], we have that $\lim_{\alpha \downarrow 1} \mathrm{D}_\alpha(P \| Q) = \mathrm{KL}\left(P \| Q\right)$, where $KL(\cdot \| \cdot)$ denotes the Kullback-Leibler (KL) divergence between the

two distributions. Finally, we relate the TV distance with the KL divergence using Bretagnolle–Huber bound [BH79, Can22] which gives the claimed bound. □

D.2 Unlearning for Linear Queries

A basic form of a query we consider is a linear query, defined as follows.

Definition 25. *A query $q : \mathcal{W}^* \times \mathcal{Z}^n \to \mathcal{W}$ is a linear query if $q\left(\{w_i\}_i ; S\right) = \sum_{j \in S} p_j\left(\{w_i\}_i ; z_j\right)$ for some functions $p_j : \mathcal{W}^* \times \mathcal{Z} \to \mathcal{W}$.*

We consider the class of B-sensitive linear queries. We give the TV stable modified learning procedure in Algorithm 26 which basically releases the linear queries perturbed with Gaussian noise of appropriate variance.

Algorithm 26 LearnLinearQueries(w_{t_0}, t_0)

Input: Dataset S, initial iteration t_0, steps T, query functions $\{q_t(\cdot)\}_{t \le T}$, update functions $\{U_t(\cdot)\}_{t \le T}$, selector function $\mathcal{S}(\cdot)$, noise variance σ^2

1: Initialize model $w_1 \in \mathcal{W}$
2: **for** $t = t_0$ to $T - 1$ **do**
3: Query the dataset $u_t = q_t\left(\{w_i\}_{i \le t} ; S\right)$.
4: Perturb: $r_t = u_t + \xi_t$ where $\xi_t \sim \mathcal{N}(0, \sigma^2 \mathbb{I}_d)$.
5: Update $w_{t+1} = U_t(\{w_i\}_{i \le t}, r_t)$
6: Save (u_t, r_t, w_{t+1})
7: **end for**

Output: $\widehat{w} = \mathcal{S}\left(\{w_t\}_{t \le T}\right)$

Note that the underlying probability distribution that the above learning algorithm samples from is a Markov chain. The corresponding unlearning procedure, described in Algorithm 27, is based on constructing a coupling between the Markov chains for the current dataset and the dataset without the to-be-deleted point. In particular, we start from the first iteration, perform rejection sampling, if it results in acceptance, then we proceed to the second iteration and so on. If some iteration results in rejection, then we do the reflection step, and continue retraining from there on.

Algorithm 27 Unlearning algorithm for linear queries

Input: Deleted point z_j,

for $t = 1$ to $T-1$ **do**
  $(u_t, r_t, w_t) = \mathsf{Load}\,()$
  Compute $u'_t = u_t - p_t^j\left(\{w_i\}_{i\le t}\,; z_j\right)$
  if $\mathsf{Unif}\,(0,1) \le \frac{\phi_{\mathcal{N}(u_t,\sigma^2\mathbb{I})}(r_t)}{\phi_{\mathcal{N}(u'_t,\sigma^2\mathbb{I})}(r_t)}$ **then**
    $\mathsf{Save}\,(u'_t)$
  else
    $r'_t = \mathsf{reflect}(r_t, u_t, u'_t)$
    $w_{t+1} = U_t\left(\{w_i\}_{i\le t}, r'_t\right)$
    $\mathsf{LearnLinearQueries}(w_{t+1}, t+1)$
    break
  end if
end for

The above is basically the same unlearning algorithm as that of [UMR+21] but presented in the general context of linear queries. Therefore, it generalizes the framework of [UMR+21] which was limited to the Stochastic Gradient Descent algorithm. We also remark that linear queries can often be augmented with a sub-sampling operator yielding *amplified* guarantees, as done in [UMR+21] . However, we omit this extension for brevity. The main result of this section is as follows.

Theorem 51. *The following are true for Algorithms 26 and 27,*

1. *The learning algorithm, Algorithm 26 with* $\sigma^2 = \frac{64B^2}{n^2\rho^2}$ *satisfies* ρ*-TV stability.*
2. *The unlearning algorithm, Algorithm 27, corresponding to Algorithm 26, satisfies exact unlearning.*
3. *The relative unlearning complexity is* $O\left(\rho\sqrt{T}\right)$.

Proof. This proof simply follows from the observation that the analysis of [UMR+21] only uses the bounded sensitivity linear query structure of the stochastic gradient method for their TV stability bound as well as correctness and runtime of the unlearning procedure. □

D.2.1 Applications

This generalization yields the following applications.

D.2.2 Federated Unlearning for Federated Averaging

In the federated learning setting, we have C clients (which typically correspond to user devices) with their own datasets and a parameter server (aggregator). A typical, informal, goal is training a single globally shared model using all the dataset with small communication between the clients and the server, and without moving any private data (explicitly) to the server. Federated Averaging [KMY+16], described in Algorithm 28, is a widely used method in federated learning. Note that in the every round of the method, the client outputs, $\{w_t^c\}_{c=1}^C$, are aggregated using an averaging operation:

$$w_t = \frac{1}{C}\sum_{c=1}^{C} w_t^c.$$

In Algorithm 28, ClientUpdate is a function which runs on the client's data using the current model w_t and problem specific-parameter $\mathcal{P}$ (such as as number of steps, learning rate of some optimization routine). For brevity, we do not instantiate the ClientUpdate function, but usually some variant of stochastic gradient descent is used.

Algorithm 28 Federated Averaging (Server side)

Input: Number of clients C, number of rounds T, client-specific parameters $\mathcal{P}$

1: Initialize model $w_1 \in \mathcal{W}$
2: **for** $t = 1$ to $T - 1$ **do**
3: **for** $c = 1$ to C **do**
4: $w_{t+1}^c = \mathsf{ClientUpdate}\,(c, w_{t-1}, \mathcal{P})$
5: **end for**
6: $w_{t+1} = \frac{1}{C}\sum_{c=1}^{C} w_{t+1}^c$
7: **end for**

Output: $\widehat{w} = \mathcal{S}\left(\{w_t\}_{t\leq T}\right)$

Federated Unlearning: In the federated unlearning problem, after a model is trained, one of the clients requests to remove themselves from the process. The

parameter server then needs to update the model (and state) in such a way that it is indistinguishable to the state if the client were absent. Hence, this is analogous to the standard unlearning problem with the client playing the role of a data point. This analogy also occurs with private federated learning wherein the *widely-used* granularity of differential privacy is user-level differential privacy [MRTZ18]. In this case, a client (potentially containing multiple data items) plays the role of a data item, the presence/absence of which is used in the differential privacy definition.

TV-stable learning and unlearning: The model aggregation step (line 6 in Algorithm 28) of the federated averaging method is a linear query over the clients. Moreover, if the clients output models that are bounded in norm, then it is a bounded sensitivity linear query (typically enforced by clipping the updates). Hence, this fits into the template of linear query release method and thus can be modified, as in Algorithm 26 to be TV stable. The corresponding unlearning method is the one given in Algorithm 27.

D.2.3 Lloyd's Algorithm for k-means Clustering

In this section, we briefly discuss how an algorithm for k-means clustering fits into the linear query release framework. We remark that the prior work [GTS13]. gave an unlearning method for this problem based on randomized quantization, which can also be seen as a specific TV-stable algorithm followed by a coupling based unlearning method.

Lloyd's algorithm is a widely used method for k-means clustering. Herein, starting with an arbitrary choice of centers, we construct a partition of the dataset, which thereby gives a new set of centers. This process is repeated for a certain number of rounds. The method is described as Algorithm 29.

We notice again that the updates for every cluster, line 7 in Algorithm 29, is a linear query, hence it fits into the linear query release template and thus learning and

unlearning algorithms based on linear queries readily follow.

Algorithm 29 Lloyd's algorithm

Input: Number of clusters C, number of rounds T, dataset $S = \{z_i\}_{i=1}^n$.

Initialize centers $\{w_c\}_{c=1}^C$
for $t = 1$ to $T - 1$ **do**
  for $c = 1$ to C **do**
    Compute $S_c = \left\{z_1^c, z_2^c, \dots z_{|S_c|}^c\right\}$, the set of data-points closest to w_c.
  end for
  for $c = 1$ to C **do**
    Update $w_c = \frac{1}{|S_c|} \sum_{i=1}^{|S_c|} z_i^c$
  end for
end for

Output: $\{w_c\}_{c=1}^C$

D.3 Missing Details from Section 5.4

In this section, we provide pseudo-code of the operations supported by the binary tree data structure.

Algorithm 30 Append$(u, \sigma; \mathcal{T})$

Input: Query response u, noise variance σ,Tree $\mathcal{T}$

Let s be the (binary representation of) first empty leaf.
Let q be the index with the first 1 in s.
$\mathsf{path} = \{s \to \cdots \text{root}\}$ be the path from s to root consisting of at most $q+1$ nodes from leaf.
$\mathsf{UpdateTree}(u, \mathsf{path}, \sigma; \mathcal{T})$

Algorithm 31 UpdateTree$(u, \mathsf{path}, \sigma; \mathcal{T})$

Input: Query response u, Set of nodes path, noise variance σ,Tree $\mathcal{T}$

for $b \in \mathsf{path}$ **do**
  $u_b = u_b + u$
  if b is a left child or b is a leaf **then**
    $\xi \sim \mathcal{N}(0, \sigma^2 \mathbb{I})$
    $r_b = u_b + \xi$
    break
  end if
end for

Algorithm 32 GetPrefixSum$(t; \mathcal{T})$

Input: $t \in \mathbb{N}$, Tree $\mathcal{T}$,

Initialize $g \in \mathbb{R}^p$ to 0
$s \leftarrow \mathsf{leaf}(t)$
Let path be the path from s to root.
while $b \neq \emptyset$ **do**
  if b is a leaf child or b is a leaf **then**
    $g = g + r_b$
  end if
end while

Output: g

D.4 Missing Proofs from Section 5.4

Proof of Theorem 25. The first part of the Theorem follows from Lemma 34 followed by post-processing to argue that the same TV stability parameter holds for the final iterate.

The second part, exact unlearning, follows from Lemma 37 wherein Q denotes the distribution of the algorithm's output run on the dataset without the to-be-deleted point.

For the third part, note that the unlearning algorithm 13 makes two queries if no retraining is triggered. If a retraining is triggered, the number of queries it makes is at most the query complexity of learning algorithm, $T = n$. Finally, the probability of retraining, from Lemma 38 is at most $\log(n)\,\rho$. Combining, this gives the stated bound on relative unlearning complexity. □

D.4.1 Lemmas for Unlearning

Additional notation: We first present some additional notation used in the statement and proof of the following lemmas. Let S and S' be datasets before and after the unlearning request. Let P and Q denote the probability measures over the range of tree data-structure, which is $\mathfrak{T} = \left(\mathbb{R}^d \times \mathbb{R}^d \times \mathbb{R}^d \times [n]\right)^n$, induced by the output of

learning algorithm on S and S' respectively. We order the nodes of the binary tree w.r.t. the post-order traversal of tree. Hence, given two nodes v and v' or their binary representations s and s', we use $v \leq v$ or $s \leq s'$ w.r.t the above ordering. Given a node b, let $P_b\left(\cdot|\mathcal{T}_{\leq b}\right)$ denote the conditional distribution of the nodes given the prefix nodes of the tree.

Let $\mathbf{p}$ be a permutation over $[n]$ and p_b denote the index on the b-th node, when b is a leaf. Let μ denote the probability, and conditional probability, depending on context, of $\mathbf{p}$ and p_b, under the random permutation model. Specifically, we use $\mu(\mathbf{p})$ and $\mu(p_b|p_{\leq b})$ to denote the probability of the sequence $\mathbf{p}$ and conditional probability of p_b given the previous values.

Let $\mathcal{T}^{(1)}$ denote the initial binary tree i.e. the one constructed after the algorithm is run on dataset S, and $\mathcal{T}^{(2)}$ be the binary tree constructed after unlearning. Let $P^{\mathbf{p}}$ and $Q^{\mathbf{p}}$ denote the conditional distributions for P and Q respectively given permutation $\mathbf{p}$.

We factor the probability density of P as:

$$\phi_P\left(\mathcal{T}^{(1)}\right) = \prod_{b\in B} \phi_{P_b}\left(v_b^{(1)}|\mathcal{T}_{\leq b}^{(1)}\right) = \prod_{b\in B} \mu(p_b^{(1)}|p_{\leq b}^{(1)}) \phi_{P_b^{p_{\leq b}^{(1)}}}\left(u_b^{(1)}, r_b^{(1)}, w_b^{(1)}|\mathcal{T}_{\leq b}^{(1)}\right)$$

Fixing the permutation sequence $\mathbf{p}^{(1)}$, denote and factor the conditional distribution as,

$$\phi_P^{\mathbf{p}^{(1)}}(\mathcal{T}^{(1)}) = \prod_{b\in B} \phi_{P_b^{\mathbf{p}_{\leq b}^{(1)}}}\left(u_b^{(1)}, r_b^{(1)}, w^{(1)}|\mathcal{T}_{\leq b}^{(1)}\right)$$

Finally, define response trees $\widetilde{\mathcal{T}}^{(1)}$ and $\widetilde{\mathcal{T}}^{(2)}$ which only contain the response variables $(r_b)_b$. Moreover, define distributions $\widetilde{P}$, $\widetilde{P}_b$, $\widetilde{P}^{\mathbf{p}}$, $\widetilde{P}_b^{\mathbf{p}}$ and $\widetilde{Q}$, $\widetilde{Q}_b$, $\widetilde{Q}^{\mathbf{p}}$ $\widetilde{Q}_b^{\mathbf{p}}$ as before.

We first show the the tree $\widetilde{\mathcal{T}}$ produced by the learning algorithm is TV-stable.

Lemma 34. *Let $0 < \rho \leq 1, B \geq 0, n \in \mathbb{N}$. For B-sensitive prefix sum queries, setting $\sigma^2 = \frac{64B^2 log^2(n)}{\rho^2}$, the response tree data structure $\widetilde{\mathcal{T}}$ is ρ-TV stable.*

Proof. The proof of privacy of tree aggregation is classical in differential privacy, see [GTS13] for example. The proof has three ingredients: Gaussian mechanism guarantee, parallel composition (to argue that accounting along the height of the tree suffices) and adaptive composition (for accounting along the height of the tree). Since the noise is Gaussian and these composition properties also holds under RDP [Mir17], therefore we can give an RDP guarantee of $\epsilon(\alpha) \leq \log^2(n) \cdot \frac{64\alpha B^2}{\sigma^2}\alpha\rho^2$. Finally, using Lemma 33 and a numerical simplification since $\rho \leq 1$ gives the claimed result. □

Recall that j is the index of the data item (after permutation) which is deleted. Without loss of generality, assume that the original index of the deleted data-point is n. We first argue the following about the distribution of $\mathbf{p}^{(1)}$ and $\mathbf{p}^{(2)}$.

Lemma 35. *For any set $E \subseteq [n]^n$ and any set $E' \subseteq [n-1]^{n-1}$, we have*

$$\mathbb{P}_{\mathbf{p}^{(1)}}\left(\mathbf{p}^{(1)} \in E\right) = \mu_n(E)$$
$$\mathbb{P}_{\mathbf{p}^{(2)}}\left(\mathbf{p}^{(2)} \in E'\right) = \mu_{n-1}(E')$$

Proof. Since $\mathbf{p}^{(1)}$ and $\mathbf{p}^{(2)}$ are discrete distributions, it suffices to argue the above for the atoms. Firstly, by construction, $\mathbf{p}^{(1)} \sim \mu_n$ and hence the first part is done. For the second part for any sequence $h = (h_i)_{i=1}^{n-1}$ where $h_i \in [n-1]$. Let $[h, j]$ denote the concatenation of h and j (the deleted index). By symmetry, the probability

$$\mathbb{P}_{\mathbf{p}^{(2)}}(h) = \frac{1}{n+1}\mathbb{P}_{\mathbf{p}^{(1)}}([h,j]) = \mu_{n-1}(h)$$

This completes the proof. □

We now show transport of the conditional distribution by the unlearning operation.

Lemma 36. *For any measurable event $E \subseteq \mathrm{R}^{d|\mathcal{T}^{(2)}|}$,*

$$\mathbb{P}\left(\widetilde{\mathcal{T}}^{(2)} \in E | \mathbf{p}^{(1)}, \mathbf{p}^{(2)}\right) = \widetilde{Q}^{\mathbf{p}^{(2)}}(E).$$

Proof. The proof is based on induction on the nodes of $\widetilde{\mathcal{T}}^{(2)}$ in the post-order traversal. Let $\left(v_b^{(1)}\right)_b$ and $\left(v_b^{(2)}\right)_b$ be the nodes of the tree arranged in the post-order traversal order. Given j, index of the item deleted, let $s = \mathsf{leaf}(j)$. Define $\mathsf{prefix}(s)$ and $\mathsf{suffix}(s)$, as set of nodes before and after s respectively in the $\leq$ order.

Given an event $E \subseteq \mathbb{R}^{d|\widetilde{\mathcal{T}}^{(2)}|}$ and $r_{\leq b}$, define $E_b^{r_{\leq b}}$ as follows:

$$E_b^{r_{\leq b}} = \left\{ e \in \mathbb{R}^d : \exists \overline{\mathbf{e}} \in \left(\times_{>b} \mathbb{R}^d \right) : (r_{\leq b}, e, \overline{\mathbf{e}}) \in E \right\}$$

where $\times_{>b}\mathbb{R}^d$ denote the Cartesian product of $\mathbb{R}^d$'s of upto $> b$ but smaller than or equal to $\left|\mathcal{T}^{(1)}\right|$ elements. Similarly, define $E_{\geq b}^{r_{\leq b}}$ as,

$$E_{\geq b}^{r_{\leq b}} = \left\{ \mathbf{e} \in \left(\times_{\geq b} \mathbb{R}^d \right) : (r_{\leq b}, \mathbf{e}) \in E \right\}$$

Finally, define $E_{<b}$ as

$$E_{<b} = \left\{ \mathbf{e} \in \left(\times_{<b} \mathbb{R}^d \right) : \exists \overline{\mathbf{e}} \in \left(\times_{\geq b} \mathbb{R}^d \right) : (\mathbf{e}, \overline{\mathbf{e}}) \in E \right\}$$

We now factorize the probability below as,

$$\begin{aligned}
&\mathbb{P}\left(\widetilde{\mathcal{T}}^{(2)} \in E | \mathbf{p}^{(1)}, \mathbf{p}^{(2)}\right) \\
&= \prod_{b \in \mathsf{prefix}(s)} \mathbb{P}\left(r_b^{(2)} \in E_b^{r_{<b}^{(2)}} | p_b^{(2)}, r_{<b}^{(2)}\right) \mathbb{P}\left(\widetilde{\mathcal{T}}_{\geq s}^{(2)} \in E_{\geq s}^{r_{<s}^{(2)}} | \widetilde{\mathcal{T}}_{<s}^{(2)}, \mathbf{p}^{(1)}, \mathbf{p}^{(2)}\right) \\
&= \prod_{b \in \mathsf{prefix}(s)} \mathbb{P}\left(r_b^{(1)} \in E_b^{r_{<b}^{(1)}} | p_b^{(1)}, r_{<b}^{(1)}\right) \mathbb{P}\left(\widetilde{\mathcal{T}}_{\geq s}^{(2)} \in E_{\geq s}^{r_{<s}^{(2)}} | \widetilde{\mathcal{T}}_{<s}^{(2)}, \mathbf{p}^{(1)}, \mathbf{p}^{(2)}\right) \\
&= \prod_{b \in \mathsf{prefix}(s)} P_b\left(E_b^{r_{<b}^{(1)}} | p_b^{(1)}, r_{<b}^{(1)}\right) \mathbb{P}\left(\widetilde{\mathcal{T}}_{\geq s}^{(2)} \in E_{\geq s}^{r_{<s}^{(2)}} | \widetilde{\mathcal{T}}_{<s}^{(2)}, \mathbf{p}^{(1)}, \mathbf{p}^{(2)}\right) \\
&= \prod_{b \in \mathsf{prefix}(s)} Q_b\left(E_b^{r_{<b}^{(2)}} | p_b^{(2)}, r_{<b}^{(2)}\right) \mathbb{P}\left(\widetilde{\mathcal{T}}_{\geq s}^{(2)} \in E_{\geq s}^{r_{<s}^{(2)}} | \widetilde{\mathcal{T}}_{<s}^{(2)}, \mathbf{p}^{(1)}, \mathbf{p}^{(2)}\right) \\
&= Q_{<s}\left(E_{<s} | p_{<s}^{(2)}, r_{<s}^{(2)}\right) \mathbb{P}\left(\widetilde{\mathcal{T}}_{\geq s}^{(2)} \in E_{\geq s}^{r_{<s}^{(2)}} | \widetilde{\mathcal{T}}_{<s}^{(2)}, \mathbf{p}^{(1)}, \mathbf{p}^{(2)}\right)
\end{aligned}$$

where the second equality follows since $r_{\leq b}^{(1)} = r_{\leq b}^{(2)}$ and $p_b^{(1)} = p_b^{(2)}$ for all $b < s$ by construction. The third equality follows since $r_b^{(1)}$ is distributed as P_b conditionally and fourth and final follows since conditioned on the permutation being the same, the prefix is also distributed as $Q_{<s}$.

We now start the induction: let I(induction variable) be $I = s$ i.e the last item is deleted. In this case, the unlearning algorithm simply removes the s-th node of the tree and all we are left with is the tree with $\mathsf{prefix}(s)$ nodes, which as argued above is distributed as $Q_{<s} = Q$.

For the case $I = s + 1$: we simply focus on $\widetilde{\mathcal{T}}^{(2)}_{\geq s} = \widetilde{\mathcal{T}}^{(2)}_{s} = r^{(2)}_{s}$. Note that $r^{(1)}_{s}$ is distributed as $\mathcal{N}(u^{(1)}, \sigma^2\mathbb{I})$ and we want $r^{(2)}_{s}$ distributed as $\mathcal{N}(u^{(2)}, \sigma^2\mathbb{I})$. The operation in the algorithm is basically a one step reflection coupling which from Lemma 1 in [UMR+21] satisfies,

$$\mathbb{P}\left(r^{(2)}_{s} \in E^{r^{(2)}_{<s}}_{s} | \mathbf{p}^{(1)}, \mathbf{p}^{(2)}\right) = Q^{p^{(2)}_{s}}_{s}\left(E^{r^{(2)}_{<s}}_{s}\right)$$

Therefore,

$$\mathbb{P}\left(\widetilde{\mathcal{T}}^{(2)} \in E | \mathbf{p}^{(1)}, \mathbf{p}^{(2)}\right) = Q_{<s}\left(E_{<s} | p^{(2)}_{<s}, r^{(2)}_{<s}\right) \widetilde{Q}^{p^{(2)}_{s}}_{s}\left(E^{r^{(2)}_{<s}}_{s}\right) = \widetilde{Q}^{\mathbf{p}^{(2)}}(E)$$

This finishes the base cases.

We now proceed to the induction step: suppose the following claim holds for nodes upto $I = k$ – for any event E, the marginal distribution

$$\mathbb{P}\left(\mathcal{T}^{(2)}_{\leq k} \in E | \mathbf{p}^{(1)}, \mathbf{p}^{(2)}\right) = \widetilde{Q}_{\leq k}\left(E | \mathbf{p}^{(2)}\right)$$

For node $k + 1$, consider a few cases:

1. A: All rejection sampling steps prior to node k resulted in accepts:
   (a) AP: Node $k + 1$ lies in the path from the s to root.
      i. APA: The rejection sampling at this node succeeds.
      ii. APR: The rejection sampling at this node fails i.e. a reflection step is performed.
   (b) AN: Node $k + 1$ doesn't lie in the path from s root.

2. R: Some rejection sampling step resulted in rejection.

For case R, we have that $r_{k+1}^{(2)} \sim \widetilde{Q}_{k+1}(\cdot|\widetilde{\mathcal{T}}_{\leq k}^{(2)}, \mathbf{p}^{(2)})$. For the case AN, note that the random variable $r_{k+1}^{(2)} = r_{k+1}^{(1)}$, hence,

$$\begin{aligned}\mathbb{P}\left(r_{k+1}^{(2)} \in E_{k+1}^{r_{\leq k}^{(2)}}|\mathsf{AN}, \mathcal{T}_{\leq k}^{(2)}, \mathbf{p}^{(1)}, \mathbf{p}^{(2)}\right) &= \widetilde{P}_{k+1}\left(E_{k+1}^{r_{\leq k}^{(2)}}|\mathbf{p}^{(2)}, \widetilde{\mathcal{T}}_{\leq k}^{(2)}\right) \\ &= \widetilde{Q}_{k+1}\left(E_{k+1}^{r_{\leq k}^{(2)}}|\mathbf{p}^{(2)}, \widetilde{\mathcal{T}}_{\leq k}^{(2)}\right)\end{aligned}$$

where the last equality follows since the dependence of $r_{k+1}^{(2)}$ is only on data points which are leaves of the sub-tree rooted at node $k+1$. These, by assumption do not contain the data point s, hence is identically distributed as P_{k+1}.

For the event AP, we have,

$$\begin{aligned}\mathbb{P}\left(r_{k+1}^{(2)} \in E_{k+1}^{r_{\leq k}^{(2)}}|\mathsf{AP}, \mathbf{p}^{(1)}\mathbf{p}^{(2)}, \widetilde{\mathcal{T}}^{(2)}\right) &= \mathbb{P}\left(r_{k+1}^{(2)} \in E_{k+1}^{r_{\leq k}^{(2)}}, \mathsf{APA}|\mathsf{AP}, \mathbf{p}^{(1)}, \mathbf{p}^{(2)}, \widetilde{\mathcal{T}}_{\leq k}^{(2)}\right) \\ &\quad + \mathbb{P}\left(r_{k+1}^{(2)} \in E_{k+1}^{r_{\leq k}^{(2)}}, \mathsf{APR}|\mathsf{AP}, \mathbf{p}^{(1)}, \mathbf{p}^{(2)}, \widetilde{\mathcal{T}}_{\leq k}^{(2)}\right) \\ &= \widetilde{Q}_{k+1}\left(E_{k+1}^{r_{\leq k}^{(2)}}|\mathbf{p}^{(1)}, \mathbf{p}^{(2)}, \widetilde{\mathcal{T}}_{\leq k}^{(2)}\right)\end{aligned}$$

where the last step follows from Lemma 1 in [UMR+21].

Hence, combining AP and AN cases,

$$\mathbb{P}\left(r_{k+1}^{(2)} \in E_{k+1}^{r_{\leq k}^{(2)}}|\mathsf{AN}, \mathcal{T}_{\leq k}^{(2)}, \mathbf{p}^{(1)}, \mathbf{p}^{(2)}\right) = \widetilde{Q}_{k+1}\left(E_{k+1}^{r_{\leq k}^{(2)}}|\mathbf{p}^{(2)}, \widetilde{\mathcal{T}}_{\leq k}^{(2)}\right)$$

We now combine all the cases: let $\phi_{\leq k}^{(\mathsf{A})}, \phi_{\leq k}^{(\mathsf{R})}$ denote the conditional densities of $\widetilde{\mathcal{T}}_{\leq k}^{(2)}$ under events A and R respectively. Let $T_k = \left|\widetilde{\mathcal{T}}_{\leq k}^{(2)}\right|$. For any event E,

$$
\begin{aligned}
&\mathbb{P}\left(\widetilde{\mathcal{T}}^{(2)}_{\leq k+1} \in E | \mathbf{p}^{(1)}, \mathbf{p}^{(2)}\right) \\
&= \mathbb{P}\left(r^{(2)}_{k+1} \in E^{r^{(2)}_{\leq k}}_{k+1} | \mathsf{A}, \widetilde{\mathcal{T}}^{(2)}_{\leq k} \in E_{\leq k}, \mathbf{p}^{(1)}, \mathbf{p}^{(2)}\right) \mathbb{P}\left(\widetilde{\mathcal{T}}^{(2)}_{\leq k} \in E^{r^{(2)}_{\leq k}}_{k+1}, \mathsf{A} | \mathbf{p}^{(1)}, \mathbf{p}^{(2)}\right) \\
&+ \mathbb{P}\left(r^{(2)}_{k+1} \in E^{r^{(2)}_{\leq k}}_{k+1} | \mathsf{R}, \widetilde{\mathcal{T}}^{(2)}_{\leq k} \in E_{\leq k}, \mathbf{p}^{(1)}, \mathbf{p}^{(2)}\right) \mathbb{P}\left(\widetilde{\mathcal{T}}^{(2)}_{\leq k} \in E_{\leq k}, \mathsf{R} | \mathbf{p}^{(1)}, p^{(2)}\right) \\
&= \int_{\mathbb{R}^{dT_{k+1}}} \mathbb{1}\left(r^{(2)}_{k+1} \in E^{r^{(2)}_{\leq k}}_{k+1}\right) \mathbb{1}\left(\widetilde{\mathcal{T}}^{(2)}_{\leq k} \in E_{\leq k}\right) \Bigg(\mathbb{1}\left(\widetilde{\mathcal{T}}^{(2)}_{\leq k} \in \mathsf{A}\right) \phi^{(\mathsf{A})}_{\leq k}\left(\widetilde{\mathcal{T}}^{(2)}_{\leq k}\right) \\
&+ \mathbb{1}\left(\widetilde{\mathcal{T}}^{(2)}_{\leq k} \in \mathsf{R}\right) \phi^{(\mathsf{R})}_{\leq k}\left(\widetilde{\mathcal{T}}^{(2)}_{\leq k}\right) \Bigg) \phi_{\widetilde{Q}^{\mathbf{p}^{(2)}}_{k+1}}\left(r^{(2)}_{k+1} | \widetilde{\mathcal{T}}^{(2)}_{\leq k}\right) d\widetilde{\mathcal{T}}^{(2)}_{\leq k} dr^{(2)}_{k+1} \\
&= \int_{\mathbb{R}^{dT_{k+1}}} \mathbb{1}\left(\mathcal{T}^{(2)}_{\leq k+1} \in E\right) \phi_{Q^{p^{(2)}}_{\leq k}}\left(\widetilde{\mathcal{T}}^{(2)}_{\leq k}\right) \phi_{\widetilde{Q}^{p^{(2)}}_{k+1}}\left(r^{(2)}_{k+1} | \widetilde{\mathcal{T}}^{(2)}_{\leq k}\right) d\widetilde{\mathcal{T}}^{(2)}_{\leq k} dr^{(2)}_{k+1} \\
&= \widetilde{Q}^{p^{(2)}}_{\leq k+1}(E)
\end{aligned}
$$

where in the third equality, we use the induction hypothesis. This completes the proof of the lemma. □

Lemma 37. *For any measurable event $E \subseteq \mathfrak{T}$, $\mathbb{P}[\mathcal{T}^{(2)} \in E] = Q(E)$.*

Proof. This follows primarily from Lemma 36, and the fact that other elements in nodes of $\mathcal{T}$, namely u_b and w_b are deterministic functions of the prefix vertices in the tree $\widetilde{\mathcal{T}}$. Consider a decomposition of the event $E = E_u \times E_r \times E_w \times E_z$. Now,

$$
\begin{aligned}
\mathbb{P}[\mathcal{T}^{(2)} \in E] &= \mathbb{E}_{\mathbf{p}^{(1)}} \mathbb{P}\left(\mathcal{T}^{(2)} \in E_u \times E_r \times E_w \times E_z | \mathbf{p}^{(1)}, \mathbf{p}^{(2)} \in E_z\right) \mathbb{P}\left(\mathbf{p}^{(2)} \in E_z\right) \\
&= \mathbb{E}_{\mathbf{p}^{(1)}} \mathbb{P}\left(\widetilde{\mathcal{T}}^{(2)} \in E_r | \mathbf{p}^{(1)}, \mathbf{p}^{(2)}\right) \mu_{n-1}(E_z) \\
&= \mathbb{E}_{\mathbf{p}^{(1)}} \widetilde{Q}^{\mathbf{p}^{(2)}}(E_r)\, \mu_{n-1}(E_2) \\
&= \mathbb{E}_{\mathbf{p}^{(1)}} Q^{\mathbf{p}^{(2)}}(E_u \times E_w \times E_r)\, \mu_{n-1}(E_z) \\
&= Q(E)
\end{aligned}
$$

where the second and fourth equality follows since variables w_b and u_b are deterministic functions of the responses $r_{\leq b}$. The second and third equality also uses Lemma 35 and Lemma 36 respectively. □

Lemma 38. *The probability of retraining is at most* $\log(n)\,\rho$.

Proof. A retraining is triggered only when a rejection sampling step fails. Note that a rejection sampling step happens only when the node b belongs to the path from s to root, say path. Let Accept be the event when all rejection sampling steps succeed.

$$\begin{aligned}
\mathbb{P}(\mathsf{Accept}) &= \mathbb{E}_{\mathcal{T}^{(1)},\mathcal{T}^{(2)},\{u_b\}} \prod_{b\in\mathsf{path}} \mathbb{1}\left(u_b \le \frac{\phi_{\widetilde{Q}_b^{\mathbf{p}^{(2)}}}\left(r_b^{(1)}|\mathcal{T}_{<b}^{(1)}\right)}{\phi_{\widetilde{P}_b^{\mathbf{p}^{(2)}}}\left(r_b^{(1)}|\mathcal{T}_{<b}^{(1)}\right)}\right) \\
&= \mathbb{E}_{\widetilde{\mathcal{T}}^{(1)},\mathbf{p}^{(1)},\mathbf{p}^{(2)}} \prod_{b\in\mathsf{path}} \mathbb{P}\left(u_b \le \frac{\phi_{\widetilde{Q}_b^{\mathbf{p}^{(2)}}}\left(r_b^{(1)}|\widetilde{\mathcal{T}}_{<b}^{(1)}\right)}{\phi_{\widetilde{P}_b^{\mathbf{p}^{(1)}}}\left(r_b^{(1)}|\widetilde{\mathcal{T}}_{<b}^{(1)}\right)}\right) \\
&= \mathbb{E}_{\mathbf{p}^{(1)},\mathbf{p}^{(2)}} \prod_{b\in\mathsf{path}} \int_{\mathbb{R}^d} \min\left(\phi_{\widetilde{Q}_b^{\mathbf{p}^{(2)}}}\left(r_b^{(1)}|\widetilde{\mathcal{T}}_{<b}^{(1)}\right), \phi_{\widetilde{P}_b^{\mathbf{p}^{(1)}}}\left(r_b^{(1)}|\widetilde{\mathcal{T}}_{<b}^{(1)}\right)\right) dr_b^{(2)} \\
&= \mathbb{E}_{\mathbf{p}^{(1)},\mathbf{p}^{(2)}} \prod_{b\in\mathsf{path}} \left(1 - \mathsf{TV}\left(\widetilde{Q}_b^{\mathbf{p}^{(2)}}, \widetilde{P}_b^{\mathbf{p}^{(1)}}|\widetilde{\mathcal{T}}_{<b}^{(1)}\right)\right) \\
&= \prod_{b\in\mathsf{path}} (1-\rho_b) \\
&\ge 1 - \sum_{b\in\mathsf{path}} \rho_b \\
&\ge 1 - \log(n) \max_b \rho_b \\
&\ge 1 - \log(n)\,\rho
\end{aligned}$$

where the fourth equality follows from the definition of TV distance and in the last equality, ρ_b denotes the (conditional) TV distance of node b. The third to last inequality follows from Lemma 39 and the second to last inequality follows from Holder's inequality. For the last inequality, we simply upper bound $\rho_b \le \rho$ since the algorithm is ρ-TV stable (Lemma 34). This completes the proof. □

Lemma 39. *Let* $\{a_i\}_{i=1}^k$ *be real numbers such that* $a_i \in (0,1)$ *for all* i *and* $\sum_{i=1}^k a_i \le 1$. *Then,* $\prod_{i=1}^k (1-a_i) \ge 1 - \sum_{i=1}^k a_i$

Proof. We prove this via induction on k. The base case $k=1$ is immediate. For the

induction step k, we have

$$\begin{aligned}\prod_{i=1}^{k}(1-a_i) = \prod_{i=1}^{k-1}(1-a_i)(1-a_k) &\geq \left(1-\sum_{i=1}^{k-1}a_i\right)(1-a_k)\\ &= 1-\sum_{i=1}^{k}a_i + \left(\sum_{i=1}^{k-1}a_i\right)a_k\\ &\geq 1-\sum_{i=1}^{k}a_i\end{aligned}$$

This completes the proof. □

D.5 Missing Proofs from Section 5.5

D.5.1 Variance-reduced Frank Wolfe

Algorithm 33 Variance-reduced Frank Wolfe$(t_0; \mathcal{T})$

Input: Dataset S, loss function $(w,z) \mapsto \ell(w,z)$, steps T, σ,$\{\eta_t\}_t$

1: **if** $t_0 = 1$ **then** Permute dataset, initialize $\mathcal{T}$, set $w_{t_0} = 0$ **end if**
2: **for** $t = 1$ to $T-1$ **do**
3: $\quad u_t = \sum_{i=1}^{t}\left((i+1)\nabla\ell(w_i;z_i) - i\nabla\ell(w_{i-1};z_i)\right)$
4: $\quad \mathsf{Append}(u_t, \sigma; \mathcal{T})$
5: $\quad r_t = \mathsf{GetPrefixSum}(t; \mathcal{T})$
6: $\quad v_t = \arg\min_{w\in\mathcal{W}} \left\langle w, \frac{r_t}{t+1}\right\rangle$
7: $\quad w_{t+1} = (1-\eta_t)w_t + \eta_t v_t$
8: $\quad \mathsf{Set}(\mathsf{leaf}(t), (u_t, r_t, w_t, z_t); \mathcal{T})$
9: **end for**

Output: $\widehat{w} = w_T$

Proof of Theorem 26. For the accuracy guarantee, we follow the proof of Theorem 1 in [ZSM+20]. Let $d_t = \frac{r_t}{t+1}$. From smoothness, we have

$$
\begin{aligned}
&L(w_{t+1};\mathcal{D}) \leq L(w_t;\mathcal{D}) + \langle \nabla L(w_t;\mathcal{D}), w_{t+1} - w_t \rangle + \frac{H}{2} \|w_{t+1} - w_t\|^2 \\
&\leq L(w_t;\mathcal{D}) + \eta_t \langle \nabla L(w_t;\mathcal{D}) - d_t, v_t - w_t \rangle + \langle d_t, v_t - w_t \rangle + \frac{\eta_t^2 H D^2}{2} \\
&= L(w_t;\mathcal{D}) + \eta_t \langle \nabla L(w_t;\mathcal{D}) - d_t, v_t - w_t \rangle + \eta_t \langle d_t, w^* - w_t \rangle + \frac{\eta_t^2 H D^2}{2} \\
&\leq L(w_t;\mathcal{D}) + \eta_t \langle \nabla L(w_t;\mathcal{D}), w^* - w_t \rangle + \eta_t \langle d_t - \nabla L(w_t), w^* - v_t \rangle + \frac{\eta_t^2 H D^2}{2} \\
&\leq (1-\eta_t) L(w_t;\mathcal{D}) - \eta_t L(w^*;\mathcal{D}) + \frac{2D}{t+1} \|d_t - \nabla L(w_t;\mathcal{D})\| + \frac{\eta_t^2 H D^2}{2}
\end{aligned}
$$

where the second inequality follows from the update and the fact that iterates lie in the set of diameter D. The third inequality follows from the optimality of v_t in the update in Algorithm 33. Finally, the last inequality follows from convexity, Cauchy-Schwarz inequality and by substituting the step-size. We now take expectation, and use the bound on gradient estimation error in Lemma 40 to get,

$$
\begin{aligned}
&\mathbb{E}[L(w_{t+1};\mathcal{D}) - L(w^*;\mathcal{D})] \\
&\leq (1-\eta_t)\, \mathbb{E}[L(w_t;\mathcal{D}) - L(w^*;\mathcal{D})] + \widetilde{O}\left((HD+G)\, D \left(\frac{1}{(t+1)^{3/2}} + \frac{\sqrt{d}}{(t+1)^2 \rho} \right) \right) \\
&+ \frac{HD^2}{2\,(t+1)^2}
\end{aligned}
$$

The above recursion gives us,

$$
\begin{aligned}
&\mathbb{E}[L(w_T;\mathcal{D}) - L(w^*;\mathcal{D})] \\
&\leq (L(w_1;\mathcal{D}) - L(w^*)) \prod_{t=1}^{T-1} (1-\eta_t) \\
&+ \sum_{i=1}^{T-1} \widetilde{O}\left((HD+G)\, D \left(\frac{1}{(i+1)^{3/2}} + \frac{\sqrt{d}}{(i+1)^2 \rho} \right) + \frac{HD^2}{(i+1)^2} \right) \prod_{t=i+1}^{T-1} (1-\eta_t) \\
&\leq \frac{HD^2}{T} + \sum_{i=1}^{T-1} \widetilde{O}\left((HD+G)\, D \left(\frac{1}{(i+1)^{1/2}} + \frac{\sqrt{d}}{(i+1)\rho} \right) + \frac{HD^2}{(i+1)} \right) \frac{1}{T} \\
&\leq \widetilde{O}\left((HD+G)\, D \left(\frac{1}{\sqrt{T}} + \frac{\sqrt{d}}{T\rho} \right) + \frac{HD^2}{T} \right) \\
&\leq \widetilde{O}\left((HD+G)\, D \left(\frac{1}{\sqrt{T}} + \frac{\sqrt{d}}{T\rho} \right) \right)
\end{aligned}
$$

where the second inequality follows from smoothness and substituting $\prod_{t=i+1}^{T-1}(1-\eta_t) = \frac{i+1}{T-1}$. Substituting number of iterations $T = n$ completes the accuracy proof.

For the unlearning part, we start by showing that the algorithm falls into the template of bounded sensitivity prefix-sum query release. Recall that the update $u_t = \sum_{i=1}^{t}\left((i+1)\nabla\ell(w_i; z_i) - i\nabla\ell(w_{i-1}; z_i)\right)$.

The sensitivity is then bounded as,

$$\begin{aligned}
&\|((i+1)\nabla\ell(w_i; z) - i\nabla\ell(w_{i-1}; z)) - ((i+1)\nabla\ell(w_i; z') - i\nabla\ell(w_{i-1}; z'))\| \\
&\leq iH\|w_i - w_{i-i}\| + 2G \\
&\leq iH\eta_{i-1}\|v_{i-1} - w_{i-1}\| + 2G \\
&\leq 2(HD + G)
\end{aligned}$$

where the first inequality follows from smoothness and Lipschitzness of the loss. The second inequality follows from the update in Algorithm 33 and the last inequality follows from the fact that the iterates remain in the set of diameter D. Hence the correctness of the unlearning algorithm follows from Theorem 25. For runtime, the training time, in terms of gradient computations is $\Theta(n)$. Therefor, using the fact that the relative unlearning complexity, from Theorem 25, is $\widetilde{O}(\rho)$, we have $\widetilde{O}(\rho n)$ bound on expected unlearning runtime. □

Lemma 40. *The gradient estimation error is,*

$$\mathbb{E}\left\|\frac{r_t}{t+1} - \nabla L(w_t; \mathcal{D})\right\|^2 \leq \widetilde{O}\left((HD+G)^2\left(\frac{1}{t+1} + \frac{d}{(t+1)^2\rho^2}\right)\right)$$

Proof. Note that $d_t := \frac{r_t}{t+1}$ comprises of the original gradient estimate from [ZSM+20], say $\widetilde{d}_t$ and the noise added by the binary tree mechanism, say ξ_t. Hence,

$$\begin{aligned}
\mathbb{E}\|d_t - \nabla L(w_t; \mathrm{D})\|^2 &= \mathbb{E}\left\|\widetilde{d}_t - \nabla L(w_t; \mathrm{D})\right\|^2 + \mathbb{E}\|\xi_t\|^2 \\
&\leq \widetilde{O}\left(\frac{(HD+G)^2}{t+1}\right) + \sum_{i=1}^{\log(n)}\frac{d\sigma^2}{(t+1)^2\rho^2} \\
&= \widetilde{O}\left((HD+G)^2\left(\frac{1}{t+1} + \frac{d}{(t+1)^2\rho^2}\right)\right)
\end{aligned}$$

where the first inequality follows from Lemma 2 in [ZSM+20] with $\alpha = 1$, and the fact that in the binary tree mechanism we add noise of variance σ at most $\log(n)$ times; the factor $1/(t+1)^2$ comes because the gradient estimate is $r_t/(t+1)$ and r_t is the binary tree response. The final equality follows by plugging in the value of σ. □

D.5.2 Dual Averaging

Algorithm 34 Dual averaging$(t_0; \mathcal{T})$

Input: Dataset S, loss function $(w, z) \mapsto \ell(w, z)$, steps T, $\{\eta_t\}_t$,

1: **if** $t_0 = 1$ **then** Permute dataset, initialize $\mathcal{T}$, set $w_{t_0} = 0$ **end if**
2: **for** $t = 1$ to $T - 1$ **do**
3: $\quad u_t = \sum_{i=1}^{t} \nabla \ell(w_i; z_i)$
4: $\quad \mathsf{Append}(u_t, \sigma; \mathcal{T})$
5: $\quad r_t = \mathsf{GetPrefixSum}(t; \mathcal{T})$
6: $\quad w_{t+1} = \Pi_{\mathcal{W}}(w_0 - \eta_t p_t)$
7: $\quad \mathsf{Set}(\mathsf{leaf}(t), (u_t, r_t, w_t, z_t); \mathcal{T})$
8: **end for**

Output: $\widehat{w} = w_T$

Proof of Theorem 27. The accuracy guarantee directly follows from Theorem 5.1 in [KMS+21], replacing $\epsilon/\log^2(1/\delta)^2$ therein by ρ. To elaborate, we set $\sigma = \widetilde{O}\left(\frac{G^2}{\rho^2}\right)$ as opposed to $\widetilde{O}\left(\frac{G^2 \log^4(1/\delta)}{\epsilon^2}\right)$, hence substituting it in the accuracy proof of Theorem 5.1 in [KMS+21] gives the claimed guarantee.

For the unlearning part, we start by showing that the algorithm falls into the template of bounded sensitivity prefix query release.

Recall that the update $u_t = \sum_{i=1}^{t} \nabla \ell(w_t; z_i)$. The sensitivity is simply bounded by Lipschitznes as,

$$\|\nabla \ell(w_t; z) - \nabla \ell(w_t; z')\| \leq 2G$$

Hence the correctness of the unlearning algorithm follows from Theorem 25. For runtime, the training time, in terms of gradient computations is $\Theta(n)$. Therefor, using the fact that the relative unlearning complexity, from Theorem 25, is $\widetilde{O}(\rho)$, we have

$\widetilde{O}(\rho n)$ bound on expected unlearning runtime. □

D.5.3 Convex GLMs with the JL method

Proof of Theorem 28. We start with the accuracy guarantee. Let $\alpha \leq 1$ be a parameter to be set later. From the JL property, with $k = O\left(\log\left(n/\beta\right)/\alpha^2\right)$, with probability at least $1-\beta$, the norm of all data-points in S, $\|\Phi x_i\| \leq (1+\alpha)\|\mathrm{x}_i\| \leq 2\|\mathcal{X}\|$. Hence, conditioned on the above event, the GLM loss function function is $\widetilde{G} = 2G\|\mathcal{X}\|$-Lipschitz and $\widetilde{H} = 4H\|\mathcal{X}\|^2$-smooth. Let $\Phi\mathcal{D}$ denote the push-forward measure of $\mathcal{D}$ under the map $(x, y) \mapsto (\Phi x, y)$. With probability at least $1-\beta$, the excess risk is,

$$
\begin{aligned}
&\mathbb{E}[L(\widehat{w}; \mathcal{D}) - L(w^*; \mathcal{D})] \\
&= \mathbb{E}[L(\Phi^\top \widetilde{w}; \mathcal{D}) - L(\Phi w^*; \Phi\mathcal{D})] + \mathbb{E}[L(\Phi w^*; \Phi\mathcal{D}) - L(w^*; \mathcal{D})] \\
&= \mathbb{E}[L(\widetilde{w}; \Phi\mathcal{D}) - L(\Phi w^*; \Phi\mathcal{D})] + \mathbb{E}[\phi_y(\langle \Phi w^*, \Phi x\rangle) - \phi_y\left(\langle w^*, x\rangle\right)] \\
&\leq \widetilde{O}\left(\left(\widetilde{G} + \widetilde{H}\|w^*\|\right)\|w^*\|\left(\frac{1}{\sqrt{n}} + \frac{\sqrt{k}}{n\rho}\right)\right) + \frac{H}{2}\mathbb{E}\left|\langle \Phi x, \Phi w^*\rangle - \langle x, w^*\rangle\right|^2 \\
&\leq \widetilde{O}\left(\left(\widetilde{G} + \widetilde{H}\|w^*\|\right)\|w^*\|\left(\frac{1}{\sqrt{n}} + \frac{\sqrt{k}}{n\rho}\right) + \frac{\widetilde{H}\|w^*\|^2}{k}\right) \\
&= \widetilde{O}\left(\frac{\left(\widetilde{G} + \widetilde{H}\|w^*\|\right)\|w^*\|}{\sqrt{n}} + \frac{\widetilde{H}^{1/3}\widetilde{G}^{2/3}\|w^*\|^{4/3} + \widetilde{H}\|w^*\|^2}{(n\rho)^{2/3}}\right)
\end{aligned}
$$

where in the first inequality, we use the accuracy guarantee of VR-Frank Wolfe (Theorem 26) and smoothness of ϕ_y together with the fact that w^* is globally optimal. The second inequality follows from JL property and the last inequality follows by the setting of k.

For the in-expectation (over the JL matrix) bound, note that in the worst-case, $L(\widehat{w}; \mathcal{D}) - L(w^*; \mathcal{D}) \leq G\|\widehat{w} - w^*\|$. From boundedness of the range of (typical) JL maps, $\|\widehat{w} - w^*\| = \mathrm{poly}(n, d)$ w.p. 1. Hence, taking the failure probability β to be small enough suffices to be give an expectation bound which is same as above upto polylogarithmic factors.

We now proceed to the unlearning guarantee. We first remark that the correctness of the unlearning algorithm (see Lemma 36) holds as long as the learning algorithm uses prefix-sum queries, even with *unbounded* sensitivity. Hence, the correctness follows. We now proceed to bound the unlearning runtime. We first bound the TV stability parameter of the learning algorithm using Lemma 41. The setting of noise variance σ in Algorithm 14 together with the stability guarantee of Theorem 26 ensures that $\gamma(\widetilde{H}, \widetilde{G}) \leq \frac{\tau}{2}$. Hence the JL method satisfies ρ-TV stability. Now, Lemma 38 gives us that the probability of retraining is at most $\widetilde{O}(\rho)$. Since the training time, in terms of gradient computations is $\Theta(n)$, we have $\widetilde{O}(\rho n)$ bound on expected unlearning runtime. □

Proof of Theorem 29. We start with the accuracy guarantee; let $\alpha \leq 1$ be a parameter to be set later. From the JL property, with $k = O\left(\log\left(n/\beta\right)/\alpha^2\right)$, with probability at least $1-\beta$, the norm of all data-points in S, $\|\Phi x_i\| \leq (1+\alpha)\|\mathrm{x}_i\| \leq 2\|\mathcal{X}\|$. Hence, conditioned on the above event, the GLM loss function function is $\widetilde{G} = 2G\|\mathcal{X}\|$-Lipschitz. Let $\Phi\mathcal{D}$ denote the push-forward measure of $\mathcal{D}$ under the map $(x, y) \mapsto (\Phi x, y)$. With probability at least $1-\beta$, the excess risk is,

$$
\begin{aligned}
&\mathbb{E}[L(\widehat{w}; \mathcal{D}) - L(w^*; \mathcal{D})] \\
&= \mathbb{E}[L(\Phi^\top \widetilde{w}; \mathcal{D}) - L(\Phi w^*; \Phi\mathcal{D})] + \mathbb{E}[L(\Phi w^*; \Phi\mathcal{D}) - L(w^*; \mathcal{D})] \\
&= \mathbb{E}[L(\widetilde{w}; \Phi\mathcal{D}) - L(\Phi w^*; \Phi\mathcal{D})] + \mathbb{E}[\phi_y(\langle \Phi w^*, \Phi x\rangle) - \phi_y\left(\langle w^*, x\rangle\right)] \\
&\leq \widetilde{O}\left(\widetilde{G}\|w^*\|\left(\frac{1}{\sqrt{n}} + \sqrt{\frac{\sqrt{k}}{n\rho}}\right)\right) + G\mathbb{E}\left|\langle \Phi x, \Phi w^*\rangle - \langle x, w^*\rangle\right| \\
&\leq \widetilde{O}\left(\widetilde{G}\|w^*\|\left(\frac{1}{\sqrt{n}} + \sqrt{\frac{\sqrt{k}}{n\rho}}\right) + \frac{\widetilde{G}\|w^*\|}{\sqrt{k}}\right) \\
&\leq \widetilde{O}\left(\widetilde{G}\|w^*\|\left(\frac{1}{\sqrt{n}} + \frac{1}{(n\rho)^{1/3}}\right)\right)
\end{aligned}
$$

where in the first inequality, we use the accuracy guarantee of Dual Averaging (Theorem 27) and Lipschitzness of ϕ_y together. The second inequality follows from JL

property and the last inequality follows by the setting of k. As in Theorem 28, the same bound as above for in-expectation (over the JL matrix) holds follows by taking the failure probability β to be small enough.

The correctness and runtime of the unlearning algorithm follows as in the proof of Theorem 28. □

Lemma 41. *Suppose $\mathcal{A}$ is an algorithm which when run on $\widetilde{H}$-smooth and $\widetilde{G}$-Lipschitz functions is $\gamma(\widetilde{H}, \widetilde{G})$-TV stable, then the JL method with with $k = O(\log(2n/\tau))$ and $\mathcal{A}$ as input, run on H-smooth and G-Lipschitz GLMs, satisfies $\frac{\tau}{2} + \gamma\left(2G\|\mathcal{X}\|, 4H\|\mathcal{X}\|^2\right)$-TV stability.*

Proof. Given a dataset S let G_S be the uniform bound on Lipschitzness parameter of the class of loss functions $\{w \mapsto \ell(w; z)\}_{z \in S}$. We define H_S similarly. Let $\alpha \leq 1$ be a parameter to be set later. From the JL property, with $k = O(\log(n/\beta))$, with probability at least $1 - \beta$, the norm of all data-points in S, $\|\Phi x_i\| \leq 2\|\mathcal{X}\|$ - we denote this event as E_{JL}. Since the loss function is a GLM, we have that conditioned on E_{JL}, the Lipschitzness and smoothness parameters G_S and H_S are bounded by $2G\|\mathcal{X}\|$ and $2H\|\mathcal{X}\|^2$ respectively. We therefore get a stability parameter $\widetilde{\gamma} := \gamma\left(2G\|\mathcal{X}\|, 4H\|\mathcal{X}\|^2\right)$.

We set $\beta = \rho/2$. We now incorporate the failure probability in the failure guarantee. Let P_Φ and Q_Φ denote the probability distributions of the output on datasets S and S'. By definition of TV distance,

$$
\begin{aligned}
\mathrm{TV}(P_\Phi, Q_\Phi) &= \sup_E \mathbb{P}_{w\sim P}\left(w \in E\right) - \mathbb{P}_{w\sim Q}\left(w \in E\right) \\
&= \sup_E \Big(\mathbb{P}_{w\sim P}\left(w \in E | E_{\mathrm{JL}}\right)\mathbb{P}(E_{\mathrm{JL}}) + \mathbb{P}_{w\sim P}\left(w \in E | E'_{\mathrm{JL}}\right)\mathbb{P}(E'_{\mathrm{JL}}) \\
&\quad - \mathbb{P}_{w\sim Q}\left(w \in E | E_{\mathrm{JL}}\right)\mathbb{P}(E_{\mathrm{JL}}) - \mathbb{P}_{w\sim Q}\left(w \in E | E'_{\mathrm{JL}}\right)\mathbb{P}(E'_{\mathrm{JL}})\Big) \\
&\leq \left(\sup_E \mathbb{P}_{w\sim P}\left(w \in E | E_{\mathrm{JL}}\right) - \mathbb{P}_{w\sim Q}\left(w \in E | E_{\mathrm{JL}}\right)\right)\mathbb{P}(E_{\mathrm{JL}}) \\
&\quad + \left(\sup_E \mathbb{P}_{w\sim P}\left(w \in E | E'_{\mathrm{JL}}\right) - \mathbb{P}_{w\sim Q}\left(w \in E | E'_{\mathrm{JL}}\right)\right)\mathbb{P}(E'_{\mathrm{JL}}) \\
&\leq \left(\sup_E \mathbb{P}_{w\sim P}\left(w \in E | E_{\mathrm{JL}}\right) - \mathbb{P}_{w\sim Q}\left(w \in E | E_{\mathrm{JL}}\right)\right) + \rho/2 \\
&\leq \widetilde{\gamma} + \rho/2
\end{aligned}
$$

which completes the proof. □

D.6 Missing Details from Section 5.6

In this section, we present additional details and proofs of results in Section 5.6.

D.6.1 Weak Unlearning

Proof of Theorem 30. The first claim, weak unlearning guarantee of the unlearning algorithm, follows mainly from Lemma 36. Specifically, it shows that conditioned on the permutation of the dataset (in this case, since the dataset is not permuted, the permutation is simply identity), the distribution over the responses $(r_b)_b$ in the tree after unlearning, is transported to the distribution of the output under S'. Since the model output is a deterministic function of the responses, (weak unlearning) correctness follows for one request. For the streaming setting, we simply apply the above inductively over the requests.

The second claim follows since, at every time point, the executed algorithm is indistinguishable from the base algorithm executed over the current dataset. Moreover,

by assumption, the base algorithm, is *anytime*, i.e. no parameter is set which depends on the size of the dataset. Hence, the accuracy guarantee follows. For the last claim about the number of retraining, firstly, as motivated, by the assumption that the algorithm is incremental, the insertions are handled in $O(1)$ time. For the unlearning requests, note that from ρ-TV stability at every point, using Lemma 38, we have a $\widetilde{O}(\rho)$ probability of retraining. We now apply Proposition 8 from [UMR+21] which converts this to a bound on the expected number of times a retraining is triggered. For V unlearning requests, this gives us a $\widetilde{O}(\rho V)$ bound on the number of retraining triggers. □

D.6.2 Exact Unlearning

Another way to extend the results for one unlearning request to dynamic streams is to modify the definition of unlearning (Definition 11) to also hold for insertions, as is done in [UMR+21]. This allows us to apply the same tree based unlearning technique when handing insertions. Specifically, upon inserting a new point, we randomly choose a leaf and replace the leaf with the inserted point, and then insert the chosen leaf as the last leaf in the tree. We have the following guarantee for this method.

Theorem 52. *In the dynamic streaming setting with R requests, using anytime learning and unlearning algorithms, Algorithm 12 and 13, the following are true.*

1. *Exact unlearning at every time point in the stream.*
2. *The accuracy of the output $\widehat{w}_i$ at time point i, with corresponding dataset S_i, is*

$$\mathbb{E}[L(\widehat{w}_i; \mathcal{D})] - \min_w L(w; \mathcal{D}) = \alpha(\rho, |S_i|; \mathcal{P})$$

3. *The total number of times, a retraining is triggered, for R requests is at most $O(\rho R)$*

Proof. The arguments are similar to that of the proof of Theorem 30. The first part follows by applying the correctness of the unlearning algorithm, Theorem 25, inductively over the stream. We remark that the handling the insertions in the same way as deletions hardly changes anything in the proofs. The second claim follows from the anytime nature of the algorithm and by assumption on the accuracy guarantee. Finally, using the probability of retraining in Lemma 38 and Proposition 8 in [UMR+21] gives us the stated number of retraining triggers. □