Machine Unlearning via Algorithmic Stability

Enayat Ullah ENAYAT@JHU.EDU
Johns Hopkins University

Tung Mai TUMAI@ADOBE.COM
Adobe Research

Anup B. Rao ANUPRAO@ADOBE.COM
Adobe Research

Ryan Rossi RROSSI@ADOBE.COM
Adobe Research

Raman Arora ARORA@CS.JHU.EDU
Johns Hopkins University

Editors: Mikhail Belkin and Samory Kpotufe

Abstract

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 convex 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.

1. Introduction

User data is employed in data analysis for various tasks such as drug discovery, as well as in third-party services for tasks such as recommendations. With such practices becoming ubiquitous, there is a growing concern that sensitive personal data can get compromised. This has resulted in a push for broader awareness of data privacy and ownership. These efforts have led to several regulatory bodies enacting laws such as the European Union General Data Protection Regulation (GDPR) and California Consumer Act, which empowers the user with (among other provisions) the right to request to have their personal data be *deleted* (see Right to be forgotten, Wikipedia (2021)). However, currently it is unclear what it means to be *forgotten* or have the data *deleted* in a rigorous sense. Nonetheless, there is a reasonable expectation that merely deleting a user's data from the database without undoing the computations derived from the said data is insufficient. In the settings where user data are directly utilized to build machine learning models for, say prediction tasks, a reasonable criterion is that the system's state is *adjusted* to what it would have been if the user data were absent to begin with – this is the criteria we adopt in our work. We refer to it as *exact unlearning* (see Definition 1 for a formal definition).

A straightforward way to comply with the requirement of exact unlearning is to recompute (or retrain, in context of machine learning). This, however, is often computationally expensive, and hence the goal is to design *efficient* unlearning algorithms - herein and afterwards, we use efficient, in the context of unlearning, to mean that the unlearning runtime is smaller than recompute time. Most of the prior work focuses on specific *structured* problems (eg: linear regression, clustering etc.) and the proposed unlearning algorithms carefully leverage this structure for efficiency - see a discussion of useful algorithmic principles like linearity, modularity etc. which enable efficient unlearning, in Ginart et al. (2019). In this work, we consider a *larger* class of problems: smooth convex empirical risk minimization (ERM), which includes many machine learning problems, eg: linear and logistic regression, as special cases (see Section 2 for definitions). This class of smooth convex ERM is sufficiently rich and arguably lacks structure useful in prior works. Gradient-based optimization is a key algorithmic technique used for smooth convex ERM (and most of machine learning and even beyond). 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. In particular, the sequential nature of gradient descent makes it challenging to design non-trivial unlearning algorithms, at least those which satisfy an *exact* unlearning criterion. To elaborate, if the point to be deleted participates in some iteration, then the subsequent steps are dependent on the to-be-deleted point, and there is no known way but to redo the computations. It is then natural to ask whether we can design unlearning algorithms with non-trivial guarantees for this class of smooth convex ERM problems.

Problem statement (informal). We consider a smooth convex ERM problem over a given initial dataset, in a setting wherein we 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: 1) 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); 2) the unlearning runtime should be *small*; 3) the output models should be sufficiently accurate (measured in empirical risk).

1.1. Our contributions

Total variation stability. We develop new algorithmic principles which *enable* exact unlearning in very 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 Appendix H.2 how the previous work of Ginart et al. (2019) 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 Bassily et al. (2016), although in the seemingly unrelated context of adaptive data analysis.

Convex risk minimization. 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 1 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 1 for details. Furthermore, we also give excess population risk bounds by leveraging known connections between generalization and algorithmic stability (see Appendix J). 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 1, 5) are (ϵ, δ)-differentially private (see Definition 3) 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 3, 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 verified empirically.

1.2. Related work

The problem of exact unlearning in smooth 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 Ginart et al. (2019) and Neel et al. (2020), which we discuss as follows. Ginart et al. (2019) 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 Ginart et al. (2019), 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 Ginart et al. (2019) as a special case (see Appendix H.2 for details).

The work of Neel et al. (2020) 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 Neel et al. (2020) is approximate, based on (ϵ, δ)-differential privacy, whereas we focus on exact unlearning. Second, the unlearning runtime in Neel et al. (2020) 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 Neel et al. (2020) in Appendix A. We show therein that with the same unlearning time, the accuracy guarantees of Neel et al. (2020) are better than us only in regimes where their approximate unlearning parameters and hence the notion, is very weak.

2. Problem setup and preliminaries

2.1. Streaming edit requests and exact unlearning

We describe the setup very generally. Let $\mathcal{Z}$ be the data space, Θ the output/parameter space, and $\mathcal{M}$ be the *meta-data*/state space. A procedure is a tuple $(\mathbf{A}(\cdot), \mathbf{U}(\cdot))$, where $\mathbf{A} : \mathcal{Z}^* \to \Theta \times \mathcal{M}$ is the *batch* learning algorithm, and $\mathbf{U} : \Theta \times \mathcal{M} \times \mathcal{Z} \to \Theta \times \mathcal{M}$ is the unlearning algorithm which updates the current model (first argument) and meta-data (second argument) given an edit request (third argument). Examples of meta-data could be a compressed *sketch* of data points, or intermediate computations/state, which could be used upon edit time. Let $\mathcal{A}(\cdot)$ denote the first output of $\mathbf{A}$ i.e. $\mathcal{A}(\cdot) = \mathbf{A}_1(\cdot)$. Similarly, let $\mathcal{U}(\cdot)$ denote the first output of $\mathbf{U}$. We remark that when we refer to the algorithm's output, we mean the model output and does not include the metadata. Finally, given two sets S and S', we define $\Delta(S, S')$ to be the symmetric difference between these sets i.e. $\Delta(S, S') = |S \backslash S'| + |S' \backslash S|$. We now define exact unlearning.

Definition 1 (Exact unlearning) *We say a procedure* $(\mathbf{A}, \mathbf{U})$ *satisfies exact unlearning if for any* $S, S' \subset \mathcal{Z}^*$ *such that* $\Delta(S, S') = 1$, $\mathbf{A}(S') = \mathbf{U}(\mathbf{A}(S), S' \backslash S \cup S \backslash S')$. *For randomized procedures, we want that for any measurable event* $\mathcal{E} \subseteq \Theta \times \mathcal{M}$, *we have*

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

Remark 1

1. *We emphasize that the above definition not only considers the (useful) output but the meta-data as well. This, with a slight difference, is referred to as* perfect *unlearning in Neel et al. (2020).*

2. *Even though the above definition is for one edit request, it can be generalized for a stream of k edit requests, by having that this condition holds inductively for every point in the stream.*

Let $S = S^0 = \{\mathrm{z}_1, \mathrm{z}_2, \ldots, \mathrm{z}_n\}, \mathrm{z}_i \in \mathcal{Z}$ 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, as in Neel et al. (2020), we assume that at any point in the stream, the number of available data points is at least $n/2$ and at most $2n$.

2.2. Convex risk minimization

We recall some basics from convex optimization. Let $\mathcal{W} \subset \mathbb{R}^d$ be a closed convex set such that diameter$(\mathcal{W}) \leq D$ where the diameter is measured in Euclidean distance. Let $\mathcal{Z}$ be the instance space and let $f : \mathcal{W} \times \mathcal{Z} \to \mathbb{R}$ be an G-Lipschitz, L-smooth convex function in its first argument. Recall that function f is L-smooth (in its first argument) if $\|\nabla_{\mathrm{w}} f(\mathrm{w}_1, \mathrm{z}) - \nabla_{\mathrm{w}} f(\mathrm{w}_2, \mathrm{z})\| \leq L\|\mathrm{w}_1 - \mathrm{w}_2\| \;\; \forall \mathrm{w}_1, \mathrm{w}_2 \in \mathcal{W}, \mathrm{z} \in \mathcal{Z}$. We will drop the subscript w in ∇ from here on. For the constraint set $\mathcal{W}$, given a point w, a projection function $\mathcal{P} : \mathbb{R}^d \to \mathbb{R}^d$ returns $\mathcal{P}(\mathrm{w}) \in \arg\min_{\mathrm{v} \in \mathcal{W}} \|\mathrm{w} - \mathrm{v}\|$.

Empirical Risk Minimization (ERM). Given data points $S = \{\mathrm{z}_1, \mathrm{z}_2, \ldots, \mathrm{z}_n\}$, we consider the following class of problems, known as empirical risk minimization (ERM).

$$\min_{\mathrm{w} \in \mathcal{W}} \left\{ \widehat{F}_S(\mathrm{w}) := \frac{1}{n} \sum_{j=1}^{n} f(\mathrm{w}, \mathrm{z}_i) \right\} \tag{1}$$

Let $\mathcal{A}(S)$ be the output of algorithm $\mathcal{A}$ on dataset S. For measuring accuracy, we will give guarantees on *expected excess empirical risk*, which is $\mathbb{E}\widehat{F}(\mathcal{A}(S)) - F(\mathrm{w}_S^*)$, where w_S^* is the minimizer: $\mathrm{w}_S^* \in \arg\min_{\mathrm{w} \in \mathcal{W}} \widehat{F}_S(\mathrm{w})$ and the expectation is taken with respect to the randomness in algorithm $\mathcal{A}$.

We discuss the related notion of population risk and the problem of risk minimization along with our results in Appendix J.

2.3. Total variation stability and maximal coupling

We first state the definition of Total Variation (TV) distance between two distributions P and Q.

$$\mathrm{TV}(P, Q) = \sup_{\text{measurable sets } R} |P(R) - Q(R)| = \frac{1}{2} \left\| \phi_P - \phi_Q \right\|_1$$

where the second equality holds if both distributions have probability densities with respect to a base measure which are denoted by ϕ_P an ϕ_Q respectively. We now define total variation stability (TV-stability), which is the notion of algorithmic stability we will use.

Definition 2 (ρ-TV-stability) *An algorithm $\mathcal{A}$ is said to be ρ-TV-stable if*

$$\sup_{S, S' : \Delta(S, S') = 1} TV(\mathcal{A}(S), \mathcal{A}(S')) \leq \rho$$

Remark 2

1. *The above definition considers the marginals of output and does 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 and repeated applications of the above definition, we have that TV$(\mathcal{A}(S), \mathcal{A}(S')) \leq (2k_1 + k_2)\rho$*

We discuss the maximal coupling characterization of total variation distance, which is a key ingredient in the design of our unlearning algorithms.

Coupling and total variation distance: A coupling between two probability distributions P and Q over a common measurable space $(\mathcal{X}, \mathcal{B})$, where $\mathcal{B}$ denotes the sigma-algebra on $\mathcal{X}$, is a distribution $\pi \in \mathbb{P}(\mathcal{X} \times \mathcal{X}, \mathcal{B} \otimes \mathcal{B})$ such that the marginals along the projections $(x, y) \mapsto x$ and $(x, y) \mapsto y$ are P and Q respectively. Let $\Pi(P, Q)$ denotes the set of couplings between P and Q. The following describes the maximal coupling characterization of total variation distance.

1. For any coupling $\pi \in \Pi(P, Q)$, if the random variable $(p, q) \sim \pi$, then $\mathrm{TV}(P, Q) \leq \mathbb{P}[p \neq q]$.

2. There exists a "maximal" coupling π^* such that if $(p, q) \sim \pi^*$, then $\mathrm{TV}(P, Q) = \mathbb{P}[p \neq q]$

The above establishes that $\text{TV}(P, Q) = \inf_{\pi \in \Pi(P,Q)} \mathbb{P}_{(p,q)\sim\pi}[p \neq q]$.

As a final remark, in this work, we routinely deal with distances and divergence between probability distributions. In some cases, we abuse notation and write a divergence between random variables instead of probability distributions - these should be interpreted as the law of the random variables.

3. Main results

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

Theorem 1 (Main Theorem) *For any $\frac{1}{n} \leq \rho < \infty$, there exists a learning and a corresponding unlearning algorithm such that for any $f(\cdot, \mathrm{z})$, which is L-smooth and G-Lipschitz convex function $\forall\, \mathrm{z}$, 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 $\widehat{\mathrm{w}}_{S^i}$ with excess empirical risk bounded as,*

$$\mathbb{E}\widehat{F}_{S^i}(\widehat{\mathrm{w}}_{S^i}) - \widehat{F}_{S^i}(\mathrm{w}^*_{S^i}) \lesssim \min\left\{\frac{GD}{\sqrt{\rho n}}, \left(\frac{L^{1/4}GD^{3/2}\sqrt{d}}{(\rho n)}\right)^{4/5}\right\}$$

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 Appendix G.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 $\rho \gtrsim \frac{1}{n}$.

Strict improvement: 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 $\alpha > \alpha_0 = \min\left\{\frac{GD}{\sqrt{n}}, \left(\frac{L^{1/4}GD^{3/2}\sqrt{d}}{n}\right)^{4/5}\right\}$. For any such α, from the above result, there exists a $\rho < 1$, such that our algorithms have $\rho k \cdot \text{Training time}(\alpha)$ expected unlearning time, which is smaller than $k \cdot \text{Training time}(\alpha)$ - 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 Appendix J 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 1 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 5.

Sub-optimality within the TV stability framework: If we consider $L, G, D = O(1)$, 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 Appendix B.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 Appendix H.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 1 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 2 (Upper bound) *For any $0 < \rho < \infty$, there exists an algorithm which is $\min\{\rho, 1\}$-TV stable, such that for any function $f(\cdot, \mathrm{z})$ which is L-smooth and G-Lipschitz convex $\forall$ z, and any dataset S of n points, outputs $\widehat{\mathrm{w}}_S$ which satisfies the following.*

$$\mathbb{E}\widehat{F}_S(\widehat{\mathrm{w}}_S) - \widehat{F}_S(\mathrm{w}_S^*) \lesssim GD \min\left\{\frac{1}{\sqrt{\rho n}}, \frac{\sqrt{d}}{\rho n}\right\}$$

We show that the condition $\rho \geq \frac{1}{n}$ in Theorem 2 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 3 (Lower bound) *For any ρ-TV-stable algorithm $\mathcal{A}$, there exists a G-Lipschitz convex function f 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, $\mathbb{E}\widehat{F}_S(\mathcal{A}(S)) - \widehat{F}_S(\mathrm{w}_S^*) \gtrsim GD \min\left\{1, \frac{1}{\rho n}\right\}$*
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, $\mathbb{E}\widehat{F}_S(\mathcal{A}(S)) - \widehat{F}_S(\mathrm{w}_S^*) \gtrsim GD \min\left\{1, \frac{1}{\sqrt{\rho n}}\right\}$*

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 2 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 Appendix I for derivations of the lower bounds.

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 Appendix B.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.

4.1. TV-stable learning algorithms and differential privacy

In this section, we discuss the ideas underlying the design of TV-stable learning algorithms. We first give the definition of differential privacy (DP), which will be a key tool. Differential privacy is a notion of data privacy, introduced in Dwork et al. (2006), defined as follows.

Definition 3 (Differential privacy (DP)) *An algorithm $\mathcal{A}$ satisfies (ϵ, δ)-differential privacy if for any two datasets S and S', differing in one sample, for any measurable event $\mathcal{E} \in Range(\mathcal{A})$,*

$$\mathbb{P}(\mathcal{A}(S) \in \mathcal{E}) \leq e^{\epsilon}\mathbb{P}(\mathcal{A}(S') \in \mathcal{E}) + \delta$$

Intuitively, 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 (Dwork et al., 2014). 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 Bassily et al. (2014) 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 $\mathring{\mathrm{w}}_t = (1-\alpha_t)\mathrm{w}_t + \alpha_t \mathrm{w}_{t-1}$ denoted as $\nabla \widehat{F}_S(\mathring{\mathrm{w}}_t, \mathrm{z}_{b_t})$ and update as follows:

$$\mathrm{w}_{t+1} = \mathring{\mathrm{w}}_j - \eta\left(\nabla \widehat{F}_S(\mathring{\mathrm{w}}_j, \mathrm{z}_{b_j}) + \theta_t\right)$$

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 1.

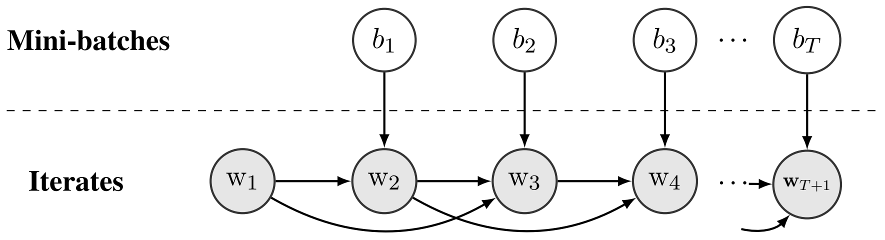

Figure 1: Markov chain for *noisy-m-A-SGD* Algorithm

A detailed discussion of challenges and certain subtleties in applying DP algorithms to our problem is provided in Appendix B.2

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 $\mathbb{P}_{(p,q)\sim\pi^*} \mathbb{1}\{(p,q) : p \neq q\} = \text{TV}(P,Q)$. 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 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 Völlering (2016) and Ernst et al. (2019), 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.

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 Appendix G, we explain what data structures to use for computational efficiency. The main result Theorem 1 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 Appendix C and focus here on *noisy-m-A-SGD*. The proofs of results in this section are deferred to Appendix E.

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 (Lan, 2012) 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 1 *noisy-m-A-SGD*(w_{t_0}, t_0)

```
Input: Initial model $\mathrm{w}_{t_0}$, data points $\{\mathrm{z}_1, \ldots, \mathrm{z}_n\}$, $T$, $\eta$, $m$
1: $\mathrm{w}_0 = 0$
2: for $t = t_0, t_0 + 1 \ldots, T$ do
3:     Sample mini-batch $b_t$ of size $m$ uniformly randomly
4:     Sample $\theta_t \sim \mathcal{N}(0, \sigma^2 \mathbb{I}_d)$
5:     $\mathring{\mathrm{w}}_t = (1 - \alpha_t)\mathrm{w}_t + \alpha_t \mathrm{w}_{t-1}$
6:     $\mathrm{g}_t = \frac{1}{m}\sum_{j \in b_t} \nabla f(\mathring{\mathrm{w}}_t, \mathrm{z}_j)$
7:     $\mathrm{w}_{t+1} = \mathcal{P}\left(\mathring{\mathrm{w}}_t - \eta\left(\mathrm{g}_t + \theta_t\right)\right)$
8:     Save($b_t, \theta_t, \mathrm{w}_t, \mathring{\mathrm{w}}_t, \mathrm{g}_t$)
9: end for
Output: $\widehat{\mathrm{w}}_S = \mathrm{w}_{T+1}$
```

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

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

$$\mathbb{E}\widehat{F}_S(\widehat{\mathrm{w}}_S) - \widehat{F}_S(\mathrm{w}_S^*) \lesssim \frac{LD^2}{T^2} + \frac{GD}{\sqrt{Tm}} + \frac{GD\sqrt{d}}{n\rho}.$$

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 1.

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 coupling of Markov chain describing *noisy-m-A-SGD*, with large mass on its diagonal. The proof of result in this

Algorithm 2 Unlearning for *noisy-m-A-SGD*

```
Input: Data point index $j$ to delete or data point $\mathrm{z}$ to insert (index $n+1$)
 1: for $t = 1, 2 \ldots, T$ do
 2:     Load$(\theta_t, \mathrm{w}_t, \mathring{\mathrm{w}}_t, b_t, \mathrm{g}_t)$
 3:     if deletion and $j \in b_t$ then
 4:         Sample $i \sim$ Uniform$([n] \backslash b_t)$
 5:         $\mathrm{g}'_t = \mathrm{g}_t - \frac{1}{m}\left(\nabla f(\mathring{\mathrm{w}}_t, \mathrm{z}_j) - \nabla f(\mathring{\mathrm{w}}_t, \mathrm{z}_i)\right)$
 6:         Save$(\mathrm{g}'_t, b_t \backslash \{j\} \cup \{i\})$
 7:     else if insertion and Bernoulli$\left(\frac{m}{n+1}\right)$ then
 8:         Sample $i \sim$ Uniform$(b_t)$
 9:         $\mathrm{g}'_t = \mathrm{g}_t - \frac{1}{m}\left(\nabla f(\mathring{\mathrm{w}}_t, \mathrm{z}_i) - \nabla f(\mathring{\mathrm{w}}_t, \mathrm{z})\right)$
10:         Save$(\mathrm{g}'_t, b_t \backslash \{i\} \cup \{n+1\})$
11:     else
12:         continue
13:     end if
14:     $\xi_t = \mathrm{g}_t + \theta_t$
15:     if Uniform$(0,1) \geq \frac{\phi_{\mathcal{N}(\mathrm{g}'_t, \sigma^2 \mathbb{I})}(\xi_t)}{\phi_{\mathcal{N}(\mathrm{g}_t, \sigma^2 \mathbb{I})}(\xi_t)}$ then
16:         $\xi'_t =$ reflect$(\xi_t, \mathrm{g}'_t, \mathrm{g}_t)$
17:         $\mathrm{w}_{t+1} = \mathrm{w}_t - \eta \xi'_t$
18:         Save$(\xi'_t)$
19:         noisy-m-A-SGD$(\mathrm{w}_{t+1}, t+1)$          // Continue retraining on current dataset
20:         break
21:     end if
22: end for
```

section is deferred to Appendix F. We first describe how Algorithm 2 couples mini-batch indices while handling edit request.

Coupling mini-batch indices: After observing a deletion request, in Algorithm 2, 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 maps: We define the notion of *reflection map*, which will be used in our coupling construction.

Definition 4 (Reflection map) *Given a vector* u *and two vectors* x *and* y*, the reflection of* u *under* (x, y)*, denoted as reflect*$(\mathrm{u}, \mathrm{x}, \mathrm{y})$*, is defined as*

$$\textit{reflect}(\mathrm{u}, \mathrm{x}, \mathrm{y}) = \mathrm{x} + (\mathrm{y} - \mathrm{u})$$

Reflection coupling is a classical idea in probability, used to construct couplings between symmetric probability distributions (Lindvall et al., 1986). The reflection map, given $\mathrm{u}, \mathrm{x}, \mathrm{y}$, reflects u about the mid-point of x and y. The context in which we will use it is u would be a sampled point from a Gaussian under old dataset S (on which the model was trained on), and x and y

being the means of the Gaussian under new 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 F.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 $\bar{\mathrm{w}}_{t+1} = \mathring{\mathrm{w}}_t - \eta(\mathrm{g}_t + \theta_t)$ is a *good* sample for the dataset S', where g_t is the gradient computed on $\mathring{\mathrm{w}}_t$ 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 2) 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 $O_d\left(Tm\right)$ 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 Appendix G.1, we also remark that using even faster variance-reduction based algorithms like Katyusha (Allen-Zhu, 2017) 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 $\bar{\mathrm{w}}_{t+1} = \mathring{\mathrm{w}}_t - \eta(\mathrm{g}_t + \theta_t)$. If we also fix the sampled mini-batch b_t, then $\bar{\mathrm{w}}_{t+1}$ is distributed as $\mathcal{N}(\mathring{\mathrm{w}}_t - \eta\mathrm{g}_t, \eta^2\sigma^2\mathbb{I})$. 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. $\frac{\phi_{S'}(\bar{\mathrm{w}}_t|b_t^S)}{\phi_S(\bar{\mathrm{w}}_t|b_t^S)}$ – this is done in line 15 of Algorithm 2, 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

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

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 F.2.3 for a more formal treatment of the coupling procedure. We now state the main result for this section.

Proposition 2 *(Algorithm 1, Algorithm 2) satisfies exact unlearning. Moreover, for k edits, Algorithm 2 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 Appendix G.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.

6. Conclusion

In this work, we presented the TV stability framework for machine unlearning and instantiated it to develop unlearning algorithms for convex risk minimization problems. Currently, our results indicate two gaps, and motivate the following future directions.

1. **Optimal TV-stable algorithm:** Our upper and lower bound on excess empirical risk of TV stable algorithms don't match. Hence, we either need to establish stronger lower bounds or search for better algorithms.

2. **Maximal coupling for unlearning:** Our coupling procedure for unlearning for *noisy-m-A-SGD* is sub-optimal, in measure of its diagonal, by a $\sqrt{T}$ factor. A natural question is whether we can design an efficient (i.e. unlearning time smaller than re-computation time) maximal coupling. We note that if efficiency were not a criterion, then this can be done - briefly, do a one step rejection sampling by computing the ratio of joint distribution of *all* iterates, if it fails, keep retraining, until the iterates generated is accepted by a rejection sampling. However, in this case, the expected number of retrains can be shown to be one, and so is trivial. The challenge in this case is to give an efficient procedure when the first rejection sampling fails.

3. **Beyond smooth convex functions**: The focus of this work was on smooth convex (loss) functions, but our techniques, and results for unlearning, extend to general non-convex functions. However, a careful investigation of trade-offs between accuracy and unlearning efficiency, in classes of, say strongly-convex, non-smooth or even some non-convex functions, is an interesting future direction.

Acknowledgements

This research was supported in part by NSF BIGDATA award IIS-1838139 and NSF CAREER award IIS-1943251.

References

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.

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

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.

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.

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.

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.

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.

Lucas Bourtoule, Varun Chandrasekaran, Christopher Choquette-Choo, Hengrui Jia, Adelin Travers, Baiwu Zhang, David Lie, and Nicolas Papernot. Machine unlearning. *arXiv preprint arXiv:1912.03817*, 2019.

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

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.

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

Luc Devroye, Abbas Mehrabian, and Tommy Reddad. The total variation distance between high-dimensional gaussians. *arXiv preprint arXiv:1810.08693*, 2018.

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.

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.

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.

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.

Chuan Guo, Tom Goldstein, Awni Hannun, and Laurens van der Maaten. Certified data removal from machine learning models. *arXiv preprint arXiv:1911.03030*, 2019.

Zachary Izzo, Mary Anne Smart, Kamalika Chaudhuri, and James Zou. Approximate data deletion from machine learning models: Algorithms and evaluations. *arXiv preprint arXiv:2002.10077*, 2020.

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

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

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

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

Seth Neel, Aaron Roth, and Saeed Sharifi-Malvajerdi. Descent-to-delete: Gradient-based methods for machine unlearning. *arXiv preprint arXiv:2007.02923*, 2020.

Arkadij Semenovich Nemirovskij and David Borisovich Yudin. *Problem complexity and method efficiency in optimization*. Wiley-Interscience, 1983.

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.

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

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

Cédric Villani. *Optimal transport: old and new*, volume 338. Springer Science & Business Media, 2008.

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

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.

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].

Blake E Woodworth and Nati Srebro. Tight complexity bounds for optimizing composite objectives. 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. URL `https://proceedings.neurips.cc/paper/2016/file/645098b086d2f9e1e0e939c27f9f2d6f-Paper.pdf`.