Never Go Full Batch (in Stochastic Convex Optimization)

Anonymous Author(s)
Affiliation
Address
email

Abstract

We study the generalization performance of *full-batch* optimization algorithms for stochastic convex optimization: these are first-order methods that only access the exact gradient of the empirical risk (rather than gradients with respect to individual data points), that include a wide range of algorithms such as gradient descent, mirror descent, and their regularized and/or accelerated variants. We provide a new separation result showing that, while algorithms such as stochastic gradient descent can generalize and optimize the population risk to within ε after $O(1/\varepsilon^2)$ iterations, full-batch methods either need at least $\Omega(1/\varepsilon^4)$ iterations or exhibit a dimension-dependent sample complexity.

1 Introduction

Stochastic Convex Optimization (SCO) is a fundamental problem in machine learning that received considerable attention in recent years [1, 10, 25, 14, 2]. In this problem, we assume a learner that is provided with a finite sample of convex functions drawn i.i.d. from an unknown distribution. The learner's goal is to minimize the expected function. Owing to its simplicity, it serves as an almost ideal theoretical model for studying and analyzing generalization properties of optimization algorithms that are arguably the most used in practice, particularly *first-order methods*.

One prominent approach for learning in SCO–and in fact, prominent in learning theory, as well as in practice–is to consider the *empirical risk* (the average objective over the sample) and apply a first-order optimization algorithm to minimize it. The problem of learning is then decoupled into controlling the optimization error over the empirical risk (training error) and bounding the difference between the empirical error and the expected error (generalization error).

In convex optimization, the convergence of different first-order methods have been researched extensively for many years [23, 3, 22], and we currently have a very good understanding of this setting in terms of upper as well as worst-case lower bounds. However, in SCO, where the generalization error needs also to be taken into account, it turns out that our understanding is still lacking. In fact, this is one of the few theoretical models where the optimization method affects not only the optimization error but also the generalization error (distinctively from models such as PAC learning and generalized linear models). In more detail, [25, 14] showed that some minima of the empirical risk may obtain large generalization error, even when the sample is sufficient for learning. To put it differently, it is not only a question of minimizing the empirical risk, but also a question of *how* one minimizes it. But the implication of this phenomenon in the context of tractable optimization methods is yet to be determined.

Towards better understanding, Bassily et al. [2] recently provided stability bounds for Gradient Descent (GD), which led to the first (dimension independent) upper bounds on its generalization, albeit with suboptimal rates. Subsequently, Amir et al. [1] matched their lower bound and demonstrated

Submitted to 35th Conference on Neural Information Processing Systems (NeurIPS 2021). Do not distribute.

that these suboptimal rates are in fact inherent in GD. This was the first evidence that not only abstract Empirical Risk Minimizers (ERMs) but, in fact, even first-order methods such as GD may overfit in SCO. Interestingly, these results show separation between the iteration complexity of Stochastic Gradient Descent (SGD) and that of GD. A natural question is then, whether overfitting is inherent to *full-batch* algorithms, that minimize the objective only through access to the empirical risk, such as GD, or whether this suboptimality can be remedied by, say, adding regularization, noise or smoothing, or any other mechanism for improving generalization.

In this work, we present and analyze a model of *full-batch* optimization algorithms. We study algorithms that access the empirical risk only via a first-order oracle that computes the *exact* gradient of the empirical loss, and do not access directly gradients with respect to individual samples. We provide a negative answer to the question above by significantly generalizing and extending the result of Amir et al. [1]. We show that any optimization method that uses full-batch gradients needs at least $\Omega(1/\varepsilon^4)$ iterations to minimize the expected loss to within ε error. This is in contrast with the empirical loss, which can be minimized with only $O(1/\varepsilon^2)$.

Comparing SGD and GD in terms of the sample size n, we get that SGD converges to an optimal generalization error of $O(1/\sqrt{n})$ after $O(n)$ iterations. On the other hand, to achieve the same $O(1/\sqrt{n})$ error, a full-batch method must perform $\Omega(n^2)$ iterations. We stress that we account here for the *oracle complexity*, which coincides with the iteration complexity in the case of gradient methods. In terms of individual gradients calculations, while SGD uses at most $O(n)$ gradient calculations (one sample per iteration), a full-batch method will perform $\Omega(n^3)$ calculations (n samples per iteration).

The above result is applicable to a wide family of learning algorithms which interact with the empirical risk only through full-batch gradient queries: regularized GD (with any *data-independent* regularization function), noisy GD, GD with line-search or adaptive step sizes, GD with momentum, proximal methods, coordinate methods, and more. Taken together with the work of Bassily et al. [2], that gave upper bound for GD, we obtain sharp rates for full-batch methods of $\Theta(1/\varepsilon^4)$ iterations. Surprisingly, this rate is achieved by standard GD (with an unusual choice of $\eta = \Theta(\varepsilon^3)$ as step-size), and it cannot be improved by adding regularization of any sort, nor by adding noise or any other form of implicit/explicit bias.

1.1 Related work

As discussed above, this work extends and generalizes the results of Amir et al. [1] who proved generalization lower bounds for GD (and a specific instance of regularized GD). Our work shows that in fact *any* full-batch method will suffer from similar lower bounds. Our construction builds upon the one used in [1] (which in turn builds upon previous constructions [2, 25]). However, our arguments and proofs here are more challenging, as we need to reason about a general family of algorithms, and not about a specific algorithm whose dynamics can be analyzed directly.

Empirical studies (outside of the scope of SCO) support the claim that generalization capabilities degrade with the increase of the batch size. Specifically, Zhu et al. [30] indicates that SGD outperforms GD in terms of generalization. The works of Keskar et al. [19] and Hoffer et al. [17] exhibit a similar phenomenon in which small-batch SGD generalizes better than large-batch SGD. We provide the first *theoretical* evidence for this phenomenon for *convex* losses. Several theoretical studies explore the convergence of stochastic methods that use mini-batches [9, 20, 28]. Note that this setting differs from ours, as they assume access to mini-batches sampled *without replacement* whereas full-batch means we reuse the same (full) batch with each gradient step.

Several other works study the generalization performance of GD [26, 15, 18, 21]. The work of Soudry et al. [26], for example, examines GD on unregularized logistic regression problems. They show that, in the limit, GD converges to a well-generalizing solution by arguing about the bias of the algorithm. Interestingly, both our and their results require slow-training, beyond what is required for empirical error optimization.

Another work that highlights the slow convergence of GD is that of Bassily et al. [2]. They were the first to address uniform stability of (non-smooth) GD and SGD, and provided tight bounds. While our work focuses on generalization, since stability entails generalization, we obtain stability lower bounds for any full-batch method. Consequently, we extend the lower bounds for GD in the work of Bassily et al. [2] to a wider class. It might be thought that the instability argument of Bassily et al. [2]

can be used to obtain similar generalization lower bounds—however, we note that their techniques also prove instability of SGD (which does generalize). Hence, instability does not immediately imply, in this setting, lack of generalization.

Several other works study different methods that aim to improve (full-batch) GD. The work of Wu et al. [29] interprets mini-batch SGD as a noisy version of GD. They propose a modified algorithm with noise injected to the full-batch gradients. Interestingly, the noise production requires access to the sample-points. Our work shows that (in SCO) this is unavoidable: namely, no sample-point independent noise can be used to improve generalization.

From a technical perspective, our developments build on ideas from the literature on oracle complexity lower bounds in optimization [22, 23, 27, 7, 11, 8]. In particular, we first prove our result in the simplified setting of algorithms constrained to the span of observed gradients [22, 23] and subsequently lift it to general algorithms using a random high-dimensional embedding technique proposed by Woodworth and Srebro [27] and later refined in [7, 11]. However, while these works lower bound what we call the empirical risk, we lower bound the generalization error. This requires us to develop a somewhat different argument for how the span of the gradients evolve during the optimization: in prior work, the algorithm learns the component of the solution coordinate by coordinate, whereas in our work the true (generalizing) solution is present in the observed gradients from the first query, but spurious sampling artifacts drown it out.

Finally, we note that under smoothness and strong convexity, it is well known that improved rates can be obtained. Specifically, using the stability bound of Bousquet and Elisseeff [4] one can show that (with the right sample size) we can achieve generalization error of ε after $O(1/\varepsilon^2)$ iterations if F is $O(1)$-strongly convex. The arguments of Hardt et al. [16] imply generalization bound to *smooth* instances. Our result implies that, even though these special families of functions enjoy appealing learning rates, in general there is no method to strong-convexify nor to smooth a general problems using black-box full-batch gradient access without deteriorating the oracle complexity significantly.

2 Problem Setup and Preliminaries

We study the standard setting of stochastic convex optimization. In this setting, a learning problem is specified by a fixed domain $\mathcal{W} \subseteq \mathbb{R}^d$ in d-dimensional Euclidean space, and a loss function $f : \mathcal{W} \times \mathcal{Z} \to \mathbb{R}$, which is both convex and L-Lipschitz with respect to its first argument. That is, for any $z \in \mathcal{Z}$ we assume the function $f(w; z)$ as a function of w is L-Lipschitz and convex. In particular, throughout the paper, our construction consists of 5-Lipschitz functions and we will focus on a fixed domain $\mathcal{W}$ defined to be the unit-ball in $\mathbb{R}^d$. Namely, $\mathcal{W} = \{w : \|w\|_2 \leq 1\}$.

We also assume that there exists an unknown distribution D and the goal of the learner is to optimize the *true risk* (or *true loss*, or *population risk*) defined as follows:

$$F(w) := \mathop{\mathbb{E}}_{z \sim D}[f(w; z)], \tag{1}$$

We assume that a sample $S = \{z_1, \ldots, z_n\}$ is drawn from the distribution D, and the learner has to provide a parameter w_S (the exact access the learner has to the sample, and how w_S may depend on S is discussed below). We require that, in expectation, the solution is ε-optimal for some predefined parameter $\varepsilon > 0$:

$$\mathop{\mathbb{E}}_{S \sim D^n}[F(w_S)] - \min_{w^\star \in \mathcal{W}} F(w^\star) \leq \varepsilon.$$

As discussed, the standard setting assumes that the learner has direct access to the i.i.d. sample, as well as to the gradients of the loss function (i.e., a first-order oracle). In this work, though, we focus on a specific family of algorithms called *full-batch* methods, which do not necessarily utilize this unrestricted access. Hence, the optimization process is described as follows: First, an i.i.d. sample $S = (z_1, \ldots, z_n)$ is drawn from D. Then, the learner is provided with access only to the *empirical risk* via a first-order oracle, called *full-batch* which we define next

Full-batch first-order oracle. Consider a *fixed* sample $S = (z_1, \ldots, z_n)$ of size n, drawn i.i.d. The *empirical risk* over the sample S is defined as

$$F_S(w) = \frac{1}{n} \sum_{i=1}^{n} f(w; z_i).$$

Then, a *full-batch first-order oracle* is a procedure that holds an i.i.d sample S drawn from D, and is given as input a parameter w. The procedure then computes the sub-gradients $\{\nabla f(w, z_1), \ldots, \nabla f(w, z_n)\}$, where each subgradient $\nabla f(w, z_i)$ depends *solely* on w and on z_i and the oracle returns, as output, the subgradient of the empirical risk:

$$\nabla F_S(w) = \frac{1}{n} \sum_{i=1}^{n} \nabla f(w; z_i).$$

We stress that the sample is fixed throughout the optimization. Namely, the first-order oracle computes the gradient of the *same* empirical risk function at every call, hence the name *full-batch*. Note that the subgradient with respect to a single data point, i.e., $\nabla f(w; z_i)$, is not accessible through this oracle, which only returns the average gradient over the sample S.

Notice that a full-batch first order oracle is, formally, stronger than a first-order oracle to the empirical risk. Specifically, our requirement that the gradient be computed by measuring the gradient at each point separately, restricts the oracle – since we provide here with a lower bound, this restriction strengthens our result. We make this restriction to avoid some degenerate constructions (that in fact can even be used to fail SGD if the gradient at z_i may depend on the whole sample), which are of no practical implications.

Full-batch first-order algorithm. A *full-batch* (first-order) method is naturally defined as any algorithm that has access to the optimization objective—namely the empirical risk F_S—only via the full-batch first order oracle. In particular, if w_t is the t'th query of the algorithm to the full-batch oracle then w_t has to be of the form

$$w_t = Q_t(\nabla F_S(w_0), \ldots, \nabla F_S(w_{t-1})), \tag{2}$$

where $Q_t : (\mathbb{R}^d)^t \to \mathcal{W}$ is a fixed (perhaps random – but independent of the sample) mapping. We will be mainly concerned with the algorithm's oracle complexity, which is the number of iterations T required before querying a point with ε-optimal true loss.

Examples. Perhaps the most simple example of a full batch method is (projected) GD: GD is an iterative algorithm that at each iteration performs an update step

$$w_t = \Pi_{\mathcal{W}}[w_{t-1} - \eta \nabla F_S(w_t)],$$

where $\mathcal{W}$ is a convex set on which we project the iterated step. Each step requires one call to a full batch oracle, and a single projection operation. The convergence analysis of GD to the optimal solution of the *empirical risk* has been widely studied. Specifically, it is known that with $\eta = O(\varepsilon)$ and $T = O(1/\varepsilon^2)$, GD converges to a minimizer of F_S; however, as discussed in [1], the solution may overfit.

Another important family of algorithms in this context are *regularized* variants of GD. Regularization is often perceived as a method to avoid overfitting. Hence, normally one wishes to optimize a regularized objective of the form

$$F^{\lambda \cdot r}(w) = \lambda r(w) + F_S(w).$$

Optimizing, via GD on $F^{\lambda \cdot r}$, with correct choice of λ, leads to an ε-suboptimal solution for F_S. For example, if $r(w) = \frac{1}{2}\|w\|^2$, we obtain an update rule:

$$w_{t+1} = \Pi_{\mathcal{W}}\left[(1 - \lambda\eta_t)w_t - \eta_t \nabla F_S(w)\right],$$

Again, these updates can be implemented using access to a full-batch oracle, whenever r is known in advance (in particular, independent of the sample).

3 Main Result

In this section we establish our main result, which provides a generalization lower-bound for full-batch first order algorithms. The complete proof is provided in the supplementary material.

Theorem 3.1. *Let $\mathcal{W}$ be the unit-ball in $\mathbb{R}^d$ and $d = T \cdot 2^{\Omega(n)}$. For any full-batch first-order algorithm, and any $\varepsilon > 0$, there exists a 1-Lipschitz convex function $f(w; z)$ in $\mathbb{R}^d$ and a distribution D over $\mathcal{Z}$ such that, for some universal constant $c > 0$:*

$$\mathop{\mathbb{E}}_{S \sim D^n}[F(w_S)] \geq \min_{w^\star \in \mathcal{W}} F(w^\star) + \varepsilon + \Omega\big(\min\big\{1 - c\varepsilon^2\sqrt{T}, 0\big\}\big).$$

An immediate consequence of Theorem 3.1 is that in order to obtain ε true risk we need at least $T = \Omega(1/\varepsilon^4)$ iterations. This, then, suggests that full-batch first-order algorithms are inferior to other forms of first-order algorithms that operate with access to individual examples, such as SGD.

4 Technical Overview

For the sake of exposition, we begin by describing a slightly simpler construction that is applicable for algorithms that remain in the span of the gradients. In more detail, let us examine the family of iterative algorithms of the form

$$w_t \in \operatorname{span}\big\{\nabla F_S(w_0), \nabla F_S(w_1), \ldots, \nabla F_S(w_{t-1})\big\} \cap B(0), \tag{3}$$

where $B(0)$ is the unit ball, and, $\nabla F_S(w_t)$ is the received sub-gradient at $w = w_t$. Well-studied algorithms such as GD and GD with standard ℓ_2 norm regularization fall into this category of algorithms. Though, our main result extends this to an even wider family of algorithms.

The more elaborate construction is provided in the supplementary material, and relies on similar ideas, together with well established techniques of random embedding in high dimensional space for optimization lower bounds. We discuss this briefly at the end of this section.

4.1 Construction

Let us fix $n, d \geq 1$ and parameters $z = (\alpha, \varepsilon, \gamma) \in \{0,1\}^d \times \mathbb{R} \times \mathbb{R}^2$, such that $\alpha \in \{0,1\}^d$, $\varepsilon > 0$ and $\gamma_1, \gamma_2 > 0$. Define a family of convex functions $f_{(4)} : \mathbb{R}^{d+2} \to \mathbb{R}$ as follows:

$$f_{(4)}(w; (\alpha, \varepsilon, \gamma)) = g_\gamma(w; \alpha) + \gamma_1 v_\alpha \cdot w + \varepsilon w \cdot e_{d+2} + r(w), \tag{4}$$

where g_γ, v_α and r are defined as follows:

- $g_\gamma(w; \alpha) := \sqrt{\sum_{i\in[d]} \alpha(i) h_\gamma^2(w(i))}$ with $h_\gamma(a) := \begin{cases} 0 & a \geq -\gamma_2; \\ a + \gamma_2 & a < -\gamma_2, \end{cases}$
- $r(w) := \max\{0, \max_{i\in[d+1]}\{w(i)\}\}$
- $v_\alpha(i) := \begin{cases} -\frac{1}{2n} & \text{if } \alpha(i) = 0; \\ +1 & \text{if } \alpha(i) = 1; \\ 0 & \text{if } i \in \{d+1, d+2\}, \end{cases}$

where e_i are the standard basis vectors. The distribution we will consider is uniform over α. That is, we draw $\alpha \in \{0,1\}^d$ uniformly at random and pick the function $f_{(4)}(w; (\alpha, \varepsilon, \gamma))$.

For the intuition behind the construction, we first note that γ_1 and γ_2 should be thought of as arbitrarily small parameters. In particular, the second term in Eq. (4) should be thought of as negligible, and the first term, g_γ, is roughly

$$g_\gamma(w; \alpha) \approx \sqrt{\sum_{i\in d} \alpha(i)(\max\{-w(i), 0\})^2}.$$

Another useful observation is that the expected function is minimized at $w^\star \approx -e_{d+2}$, with expected loss $F(w^\star) \approx -\varepsilon$. However, as we will see, the choice of the perturbation vector v_α as well as the last term, inhibit the learner from observing this coordinate and she is restricted to a span of coordinates that all have high generalization error due to the term $\mathbb{E}[g_\gamma]$.

4.2 Analysis

We next state the main lemma we use. Given a sample S, let us denote $\bar{v} = \frac{1}{n}\sum_{\alpha\in S} v_\alpha$, and $\operatorname{span}^1\{u_1, u_2, \ldots\} \triangleq \operatorname{span}\{u_1, u_2, \ldots\} \cap B(0)$, and $\mathfrak{I} = \{i : \forall \alpha \in S, \alpha(i) = 0\} \bigcup \{d+1\}$.

Lemma 4.1. *Suppose that $w_0 = 0$, and $\gamma_1 \leq \frac{1}{2T}$, $\gamma_2 = \frac{2\gamma_1}{\varepsilon}$. Suppose S is a sample such that $|\mathfrak{I}| > T$. Then for any algorithm that adheres to*

$$w_t \in \operatorname{span}^1\big\{\nabla F_S(w_0), \nabla F_S(w_1), \ldots, \nabla F_S(w_{t-1})\big\},$$

with respect to $f(w;(\alpha,\varepsilon,\gamma))$ defined in Eq. (4), there exists a first order oracle(independent of the algorithm) and a subset $\mathfrak{I}_t \subseteq \mathfrak{I}$ of maximum size t such that:

$$\forall t \in [T]: \qquad w_t \in \operatorname*{span}_{i\in\mathfrak{I}_t}{}^{1}\Big\{\gamma_1\bar{v} + \varepsilon e_{d+2} + e_i\Big\},$$

where $\mathfrak{I}_1 = \{d+1\} \subseteq \mathfrak{I}_2 \subseteq \ldots \subseteq \mathfrak{I}_t \subseteq \mathfrak{I}$.

We next observe that in any span of the form $\{\gamma_1\bar{v} + \varepsilon e_{d+2} + e_i\}_{i\in\mathfrak{I}_T}$ such that $|\mathfrak{I}_T| \le T$, we cannot find a solution with better risk than 0. On the other hand, note that for $\bar{w} = -e_{d+2}$, we have that

$$f_{(4)}(\bar{w};(\alpha,\varepsilon,\gamma)) = -\varepsilon.$$

In other words, our lower bound stems from the following result:

Lemma 4.2. *For sufficiently small γ_1,γ_2, and any vector $\|\bar{v}\| \le \sqrt{d}$, for any solution:*

$$w_S \in \operatorname*{span}_{i\in\mathfrak{I}_T}{}^{1}\{\gamma_1\bar{v} + \varepsilon e_{d+2} + e_i\},$$

we have that:

$$\frac{1}{2}\sqrt{\sum_{i\in[d]} h_\gamma^2(w_S(i))} + \varepsilon w_S\cdot e_{d+2} \ge \min\Big\{\frac{1}{4} - \varepsilon^2\sqrt{T}, 0\Big\} - \frac{1}{2}\varepsilon. \tag{5}$$

Proof sketch (for the simplified form in Eq. (3)) First, observe that the probability of an arbitrary index i to hold $\alpha(i) = 0$ for all $\alpha \in S$ is $(1/2)^n$. Therefore, the number of indexes that hold this from the possible d - behaves like the binomial distribution with $p = (1/2)^n$ and d experiments. Using elementary probability arguments one can show that for sufficiently large d then w.h.p. we have $|\mathfrak{I}| > T$. For more information we refer the reader to the formal claim in the supplementary material. This implies that the conditions of Lemma 4.1 are met w.h.p and therefore also the conditions of Lemma 4.2. To conclude, we relate the expected risk to the term in Eq. (5):

$$\mathop{\mathbb{E}}_{\alpha\sim D}[f_{(4)}(w;(\alpha,\varepsilon,\gamma))] = \mathop{\mathbb{E}}_{\alpha\sim D}[g_\gamma(w;\alpha)] + \gamma_1\cdot\mathop{\mathbb{E}}_{\alpha\sim D}[v_\alpha]\cdot w + \varepsilon w\cdot e_{d+2} + r(w).$$

As $g_\gamma(w;\alpha)$ is convex w.r.t. α (since $\alpha(i) = \alpha^2(i)$) we can apply Jensen's inequality with $\mathbb{E}_{\alpha\sim D}[\alpha(i)] = \frac{1}{2}$ to obtain:

$$\mathop{\mathbb{E}}_{\alpha\sim D}[g_\gamma(w_S;\alpha)] \ge \frac{1}{2}\sqrt{\sum_{i\in[d]} h_\gamma^2(w_S(i))}.$$

Applying Cauchy-Schwarz inequality to the second term, while also using the fact that $\|v_\alpha\| \le \sqrt{d}$ and that w_S remains in the unit ball, we get:

$$\gamma_1\cdot\mathop{\mathbb{E}}_{\alpha\sim D}[v_\alpha]\cdot w \ge -\gamma_1\|v_\alpha\|\cdot\|w\| \ge -\gamma_1\sqrt{d}$$

Then for sufficiently small γ_1 this term is negligible and since $r(w) \ge 0$, we get that the expected risk is approximately the LHS term in Eq. (5). Lastly, since $F(w) := \mathbb{E}_{\alpha\sim D}[f_{(4)}(w;(\alpha,\varepsilon,\gamma))]$. By using the observation that $F(-e_{d+2}) = -\varepsilon$ we get that w.h.p,

$$F(w_S) - \min_{w\in\mathcal{W}} F(w) \ge \frac{1}{2}\varepsilon + \min\Big\{\frac{1}{4} - \varepsilon^2\sqrt{T}, 0\Big\}.$$

Transforming the lower bound to hold in expectation, immediately follows by the law of total expectation. Our distribution is supported on 5-Lipschitz convex functions, hence ultimately, by reparameterization of $\frac{1}{10}\varepsilon \to \varepsilon$ we achieve the declared lower bound.

4.3 Handling general full-batch algorithms

The above construction establishes an $\Omega(1/\varepsilon^4)$ oracle complexity lower bound on any algorithm whose iterates lie in the span of the previous gradients. While this covers a large class of algorithms, techniques like preconditioning [12], coordinate methods [24] and randomized smoothing [13] do

not satisfy this assumption. In fact, a trivial algorithm that always outputs $-e_{d+2}$ will solve the hard instance (4) in a single iteration.

To address general algorithms, we employ a well-established technique in optimization lower bounds [27, 7, 11] wherein we embed a hard instance $f(w;z)$ for span-constrained algorithms in a random high-dimensional space. More concretely, we draw a random orthogonal matrix $U \in \mathbb{R}^{d' \times d}$ ($U^\top U = I_{d\times d}$) and consider the $d' > d$-dimensional instance $f_U(w;z) = f(U^\top w; z)$ along with its corresponding empirical objective $F_{S,U}(w) = \frac{1}{n}\sum_{i\in[n]} f_U(w;z_i)$. Roughly speaking, we show that for a general algorithm operating with the appropriate subgradient oracle for $F_{S,U}$ the iterate w_t is approximately in the span of $\{\nabla F_{S,U}(w_0), \ldots, \nabla F_{S,U}(w_{t-1})\}$ in the sense that the component of w_t outside that span is nearly orthogonal to the columns of U. Consequently, the response of the oracle to the query w_t at iteration t is, with high probability, identical to the information it would return if queried with the projection of w_t to the span of the previously observed gradients. This reduces, in a sense, the problem back to the span-restricted setting described above.

For the embedding technique to work, we must robustify the hard instance construction so that small perturbations around points in the span of previous gradients do not "leak" additional information about the embedding U. To do that we make a fairly standard modification to the component $r(w)$ in (4) (known as Nemirovski's function [11, 6]), replacing it with $\max\{0, \max_{i\in[d]}\{w(i) + i\gamma'\}, w(d+1) + \gamma''\}$, where γ', γ'' are small offset coefficients that go to zero as the embedding dimension d' tends to infinity. We provide the full construction and the proof of the general lower bound Theorem 3.1 in the supplementary material.

5 Proofs for Section 4

5.1 Proof of Lemma 4.1

We will prove this by induction on t. For $t = 1$ we get,

$$\begin{aligned}
\nabla F_S(w_0) = \nabla F_S(0) &= \frac{1}{n}\sum_{\alpha\in S} \nabla g_\gamma(0;\alpha) + \gamma_1\bar{v} + \varepsilon e_{d+2} + \nabla r(0) \\
&= \gamma_1\bar{v} + \varepsilon e_{d+2} + \nabla r(0) && (\nabla g_\gamma(0;\alpha) = 0) \\
&= \gamma_1\bar{v} + \varepsilon e_{d+2} + e_{d+1}, && (\nabla r(0) = e_{d+1})
\end{aligned}$$

Where we take a first order oracle that returns the subgradient e_{d+1} for the function r at zero. For the induction step we assume that $w_t \in \operatorname{span}^1{}_{i\in\mathfrak{I}_t}\left\{\gamma_1\bar{v} + \varepsilon e_{d+2} + e_i\right\}$. Therefore w_t takes the form:

$$w_t = \sum_{i\in\mathfrak{I}_t} \beta_i\left(\gamma_1\bar{v} + \varepsilon e_{d+2} + e_i\right).$$

for some $\beta_i \in \mathbb{R}$. Observe that $\bar{v}(i) = 0$ for $i \in \{d+1, d+2\}$, and $\bar{v}(i) = -\frac{1}{2n}$ for any $i \in \mathfrak{I}$, and also that $\bar{v}(i) \in \left(\frac{1}{2n}, 1\right]$ for $i \notin \{\mathfrak{I} \bigcup \{d+2\}\}$. Denote $\bar{\beta} = \sum_{i\in\mathfrak{I}_t}\beta_i$, therefore

$$w_t(i) = \begin{cases} -\frac{\gamma_1}{2n}\bar{\beta} + \beta_i & \text{if } i \in \mathfrak{I}_t \setminus \{d+1\}; \\ -\frac{\gamma_1}{2n}\bar{\beta} & \text{if } i \in \mathfrak{I} \setminus \mathfrak{I}_t; \\ \gamma_1\bar{v}(i)\bar{\beta} & \text{if } i \notin \{\mathfrak{I} \bigcup \{d+2\}\}; \\ \beta_i & \text{if } i = d+1; \\ \varepsilon\bar{\beta} & \text{if } i = d+2. \end{cases} \tag{6}$$

Consider then the gradient at w_t,

$$\nabla F_S(w_t) = \frac{1}{n}\sum_{\alpha\in S} \nabla g_\gamma(w_t;\alpha) + \gamma_1\bar{v} + \varepsilon e_{d+2} + \nabla r(w_t). \tag{7}$$

We examine the first and last terms in Eq. (7). For the first term we present the following claim:

Claim 5.1. *For sufficiently small $\gamma_1 \le \frac{\varepsilon\gamma_2}{2}$ we have*

$$\frac{1}{n}\sum_{\alpha\in S} \nabla g_\gamma(w_t;\alpha) = 0.$$

Proof. Since $\alpha(i) = 0$ for $i \in \mathfrak{I}$ we get,

$$\frac{1}{n}\sum_{\alpha\in S}\nabla g_\gamma(w_t;\alpha) = \frac{1}{n}\sum_{\alpha\in S}\nabla\left(\sqrt{\sum_{i\notin\{\mathfrak{I}\bigcup d+2\}}\alpha(i)h_\gamma^2(w_t(i))}\right). \tag{8}$$

We will show that for $\gamma_1 \le \frac{\varepsilon\gamma_2}{2}$, it holds that $w_t(i) > -\gamma_2$ for $i \notin \{\mathfrak{I}\bigcup\{d+2\}\}$. Assuming it holds, note that $h_\gamma(w_t(i)) = 0$ for any $w_t(i) \ge -\gamma_2$, then together with Eq. (8) we conclude the claim. We proceed by proving that this assumption holds.

Using Eq. (6) implies that $w_t(i) = \gamma_1\bar{v}(i)\bar{\beta}$ for $i \notin \{\mathfrak{I}\bigcup\{d+2\}\}$. Observe that for $\bar{\beta} \ge 0$ it follows immediately that $w_t(i) \ge 0 > -\gamma_2$. Then, for $\bar{\beta} < 0$, we assume by contradiction that $w_t(i) \le -\gamma_2$ for some $i \notin \{\mathfrak{I}\bigcup\{d+2\}\}$. Therefore, we get that $\gamma_1\bar{v}(i)\bar{\beta} \le -\gamma_2$, or alternatively that

$$\bar{\beta} \le -\frac{\gamma_2}{\gamma_1\bar{v}(i)}. \tag{9}$$

In addition, from Eq. (6) it is clear that

$$\begin{aligned}
\|w_t\| &\ge \|\varepsilon\bar{\beta}e_{d+2}\| && \text{(Eq. (6))}\\
&= \varepsilon(-\bar{\beta}) && (\bar{\beta} < 0)\\
&\ge \frac{\varepsilon\gamma_2}{\gamma_1\bar{v}(i)} && \text{(Eq. (9))}\\
&\ge \frac{\varepsilon\gamma_2}{\gamma_1} && (\bar{v}(i) \in (\tfrac{1}{2n}, 1] \text{ for } i \notin \{\mathfrak{I}\bigcup\{d+2\}\})\\
&\ge 2. && (\gamma_1 \le \tfrac{\varepsilon\gamma_2}{2})
\end{aligned}$$

However, we assume $\|w\| \le 1$. ■

For the last term in Eq. (7), we use the claim below:

Claim 5.2. *For $|\mathfrak{I}| > T$ and $\gamma_1 \le \frac{1}{2T}$ we get that*

$$\nabla r(w_t) = e_i,$$

for some $i \in \mathfrak{I}$.

Proof. To prove this we split the analysis to two cases:

Case 1 - Assume $\bar{\beta} \ge 0$: Observe that if $\beta_i = 0$ for all $i \in \mathfrak{I}_t$, then $w_t = 0$ which coincides with the base of the induction. Assume that not all $\beta_i = 0$. Then, we next show that for sufficiently small $\gamma_1 \le \frac{1}{2T}$ we get that

$$i^\star := \arg\max_{i\in[d+1]}\{w_t(i)\} \in \mathfrak{I}. \tag{10}$$

To see that Eq. (10) holds, note that since $\bar{\beta} \ge 0$, then for $i' = \arg\max_{i\in\mathfrak{I}_t}\beta_i$ we have that $\beta_{i'} > 0$. Observe that $\gamma_1\bar{v}(i)\bar{\beta} \le \gamma_1\bar{\beta}$ for $i \notin \{\mathfrak{I}\bigcup\{d+2\}\}$ due to the fact that $\bar{v}(i) \le 1$. Suppose $\bar{\beta} > 0$, as such for any $i \notin \{\mathfrak{I}\bigcup\{d+2\}\}$, again by Eq. (6)

$$\begin{aligned}
w_t(i) = \gamma_1\bar{v}(i)\bar{\beta} \le \gamma_1\bar{\beta} &= 2\gamma_1\bar{\beta} - \gamma_1\bar{\beta}\\
&< \frac{1}{T}\bar{\beta} - \frac{\gamma_1}{2n}\bar{\beta} && (\gamma_1 \le \tfrac{1}{2T})\\
&\le \frac{1}{|\mathfrak{I}_t|}\bar{\beta} - \frac{\gamma_1}{2n}\bar{\beta} && (|\mathfrak{I}_t| \le t \le T)\\
&= \frac{1}{|\mathfrak{I}_t|}\sum_{i\in\mathfrak{I}_t}\beta_i - \frac{\gamma_1}{2n}\bar{\beta}\\
&\le \max_{i\in\mathfrak{I}_t}\beta_i - \frac{\gamma_1}{2n}\bar{\beta}\\
&= \beta_{i'} - \frac{\gamma_1}{2n}\bar{\beta}\\
&= w_t(i').
\end{aligned}$$

And therefore $i^\star \in \mathfrak{I}$. Similarly, when $\bar{\beta} = 0$ it follows immediately that for any $i \notin \{\mathfrak{I} \bigcup \{d+2\}\}$ we get

$$w_t(i) = 0 < \beta_{i'} = w_t(i'),$$

which implies again that $i^\star \in \mathfrak{I}$. Thus,

$$\nabla r(w_t) = \nabla\Big(\max\{0, \max_{i\in[d+1]}\{w_t(i)\}\}\Big) = e_i,$$

for some $i \in \mathfrak{I}$.

Case 2 - Assume $\bar{\beta} < 0$: From Eq. (6), note that since $w_t(i) < 0$ for $i \notin \{\mathfrak{I} \bigcup \{d+2\}\}$, then

$$i^\star := \arg\max_{i\in[d+1]}\{w_t(i)\} \in \mathfrak{I}, \tag{11}$$

as long as $\mathfrak{I} \setminus \mathfrak{I}_t \neq \emptyset$ (Namely, $t < |\mathfrak{I}|$). This is justified by the assumption that $|\mathfrak{I}| > T$. Therefore, $\nabla r(w_t) = e_i$ for some $i \in \mathfrak{I}$. ■

Combining both observations in Claims 5.1 and 5.2 and plugging it into Eq. (7) we obtain,

$$\nabla F_S(w_t) = \gamma_1 \bar{v} + \varepsilon e_{d+2} + e_i,$$

for some $i \in \mathfrak{I}$. Denoting $\mathfrak{I}_{t+1} = \mathfrak{I}_t \bigcup \{i\}$ we conclude that,

$$\begin{aligned} w_{t+1} &\in \operatorname{span}\big\{\nabla F_S(w_0), \nabla F_S(w_1), \ldots, \nabla F_S(w_t)\big\} \\ &= \operatorname*{span}_{i\in\mathfrak{I}_{t+1}}\big\{\gamma_1\bar{v} + \varepsilon e_{d+2} + e_i\big\}, \end{aligned}$$

as $w_t \in \operatorname{span}\big\{\nabla F_S(w_0), \nabla F_S(w_1), \ldots, \nabla F_S(w_{t-1})\big\} = \operatorname{span}_{i\in\mathfrak{I}_t}\big\{\gamma_1\bar{v} + \varepsilon e_{d+2} + e_i\big\}$. An immediate consequence is that indeed $\mathfrak{I}_t \subseteq \mathfrak{I}_{t+1} \subseteq \mathfrak{I}$.

5.2 Proof of Lemma 4.2

By assumption, for some sequence $\{\beta_i\}_{i\in\mathfrak{I}_T}$ where $\beta_i \in \mathbb{R}$, then w_S takes the form:

$$w_S = \sum_{i\in\mathfrak{I}_T} \beta_i\big(\gamma_1\bar{v} + \varepsilon e_{d+2} + e_i\big).$$

Denote $\mathfrak{I}_T^- = \{i : \beta_i \leq 0\}$ as the non-positive subset in $\mathfrak{I}_T$. Therefore,

$$\begin{aligned} \frac{1}{2}\sqrt{\sum_{i\in[d]} h_\gamma^2(w_S(i))} + \varepsilon w_S \cdot e_{d+2} &\geq \frac{1}{2}\sqrt{\sum_{i\in\mathfrak{I}_T^-} h_\gamma^2(w_S(i))} + \varepsilon^2 \sum_{i\in\mathfrak{I}_T^-}\beta_i \\ &\geq \frac{1}{2}\sqrt{\sum_{i\in\mathfrak{I}_T^-} h_\gamma^2(\beta_i/2)} + \varepsilon^2 \sum_{i\in\mathfrak{I}_T^-}\beta_i. \end{aligned} \tag{12}$$

where the last inequality holds since $\beta_i(1+\gamma_1\bar{v}(i)) \leq \beta_i/2$ for $\beta_i \leq 0$ and $\gamma_1 \leq n$. Using the reverse triangle inequality, and the observations that $|h_\gamma(a) - a| \leq \gamma_2$ for any $a \leq 0$ and $|\mathfrak{I}_T^-| \leq T$ we get,

$$\sqrt{\sum_{i\in\mathfrak{I}_T^-} h_\gamma^2(\beta_i/2)} \geq \sqrt{\sum_{i\in\mathfrak{I}_T^-}(\beta_i/2)^2} - \sqrt{\sum_{i\in\mathfrak{I}_T^-}(h_\gamma(\beta_i/2) - \beta_i/2)^2} \geq \frac{1}{2}\sqrt{\sum_{i\in\mathfrak{I}_T^-}\beta_i^2} - \gamma_2\sqrt{T}.$$

Plugging this into Eq. (12),

$$\begin{aligned} \frac{1}{2}\sqrt{\sum_{i\in[d]} h_\gamma^2(w_S(i))} + \varepsilon w_S \cdot e_{d+2} &\geq \frac{1}{4}\sqrt{\sum_{i\in\mathfrak{I}_T^-}\beta_i^2} + \varepsilon^2\sum_{i\in\mathfrak{I}_T^-}\beta_i - \frac{1}{2}\gamma_2\sqrt{T} \\ &\geq \Big(\frac{1}{4} - \varepsilon^2\sqrt{T}\Big)\sqrt{\sum_{i\in\mathfrak{I}_T^-}\beta_i^2} - \frac{1}{2}\varepsilon, \qquad (\gamma_2 \leq \varepsilon/\sqrt{T}) \end{aligned}$$

where the second inequality follows from the known norm inequality - $\|w\|_1 \leq \sqrt{d}\|w\|_2$ for $w \in \mathbb{R}^d$, and again since $|\mathfrak{I}_T^-| \leq T$. To conclude, we note that the solution w_S must remain in the unit ball, thus

$$\sqrt{\sum_{i\in\mathfrak{I}_T}\beta_i^2} \leq \|w_S\| \leq 1,$$

and we obtain the required result.

References

[1] I. Amir, T. Koren, and R. Livni. SGD generalizes better than GD (and regularization doesn't help). *CoRR*, abs/2102.01117, 2021.

[2] R. Bassily, V. Feldman, C. Guzmán, and K. Talwar. Stability of stochastic gradient descent on nonsmooth convex losses. In *Advances in Neural Information Processing Systems 33*, 2020.

[3] L. Bottou, F. E. Curtis, and J. Nocedal. Optimization methods for large-scale machine learning. *Siam Review*, 60(2):223–311, 2018.

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

[5] S. P. Boyd and L. Vandenberghe. *Convex Optimization*. Cambridge University Press, 2014.

[6] S. Bubeck, Q. Jiang, Y. T. Lee, Y. Li, A. Sidford, et al. Complexity of highly parallel non-smooth convex optimization. *Advances in neural information processing systems*, 2019.

[7] Y. Carmon, J. C. Duchi, O. Hinder, and A. Sidford. Lower bounds for finding stationary points I. *Mathematical Programming*, pages 1–50, 2019.

[8] Y. Carmon, A. Jambulapati, Q. Jiang, Y. Jin, Y. T. Lee, A. Sidford, and K. Tian. Acceleration with a ball optimization oracle. In *Advances in Neural Information Processing Systems*, volume 33, 2020.

[9] A. Cotter, O. Shamir, N. Srebro, and K. Sridharan. Better mini-batch algorithms via accelerated gradient methods. *arXiv preprint arXiv:1106.4574*, 2011.

[10] A. Dauber, M. Feder, T. Koren, and R. Livni. Can implicit bias explain generalization? stochastic convex optimization as a case study. In *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems*, 2020.

[11] J. Diakonikolas and C. Guzmán. Lower bounds for parallel and randomized convex optimization. In *Conference on Learning Theory*, 2019.

[12] J. Duchi, E. Hazan, and Y. Singer. Adaptive subgradient methods for online learning and stochastic optimization. *Journal of machine learning research*, 12(7), 2011.

[13] J. C. Duchi, P. L. Bartlett, and M. J. Wainwright. Randomized smoothing for stochastic optimization. *SIAM Journal on Optimization*, 22(2):674–701, 2012.

[14] V. Feldman. Generalization of ERM in stochastic convex optimization: The dimension strikes back. In *Advances in Neural Information Processing Systems 29: Annual Conference on Neural Information Processing Systems 2016, December 5-10, 2016, Barcelona, Spain*, pages 3576–3584, 2016.

[15] S. Gunasekar, J. D. Lee, N. Srebro, and D. Soudry. Implicit bias of gradient descent on linear convolutional networks. *Advances in Neural Information Processing Systems*, 2018:9461–9471, 2018.

[16] M. Hardt, B. Recht, and Y. Singer. Train faster, generalize better: Stability of stochastic gradient descent. In *International Conference on Machine Learning*, pages 1225–1234. PMLR, 2016.

[17] E. Hoffer, I. Hubara, and D. Soudry. Train longer, generalize better: closing the generalization gap in large batch training of neural networks. In I. Guyon, U. von Luxburg, S. Bengio, H. M. Wallach, R. Fergus, S. V. N. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pages 1731–1741, 2017.

[18] Z. Ji and M. Telgarsky. The implicit bias of gradient descent on nonseparable data. In *Conference on Learning Theory*, pages 1772–1798, 2019.

[19] N. S. Keskar, D. Mudigere, J. Nocedal, M. Smelyanskiy, and P. T. P. Tang. On large-batch training for deep learning: Generalization gap and sharp minima. In *5th International Conference on Learning Representations, ICLR 2017, Toulon, France, April 24-26, 2017, Conference Track Proceedings*. OpenReview.net, 2017.

[20] M. Li, T. Zhang, Y. Chen, and A. J. Smola. Efficient mini-batch training for stochastic optimization. In S. A. Macskassy, C. Perlich, J. Leskovec, W. Wang, and R. Ghani, editors, *The 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD '14, New York, NY, USA - August 24 - 27, 2014*, pages 661–670. ACM, 2014.

[21] Y. Li, E. X. Fang, H. Xu, and T. Zhao. Implicit bias of gradient descent based adversarial training on separable data. In *International Conference on Learning Representations*, 2019.

[22] A. S. Nemirovskij and D. B. Yudin. Problem complexity and method efficiency in optimization. 1983.

[23] Y. Nesterov. *Introductory lectures on convex optimization: A basic course*, volume 87. Springer Science & Business Media, 2003.

[24] Y. Nesterov. Efficiency of coordinate descent methods on huge-scale optimization problems. *SIAM Journal on Optimization*, 22(2):341–362, 2012.

[25] S. Shalev-Shwartz, O. Shamir, N. Srebro, and K. Sridharan. Stochastic convex optimization. In *COLT*, 2009.

[26] D. Soudry, E. Hoffer, M. S. Nacson, S. Gunasekar, and N. Srebro. The implicit bias of gradient descent on separable data. *The Journal of Machine Learning Research*, 19(1):2822–2878, 2018.

[27] B. Woodworth and N. Srebro. Tight complexity bounds for optimizing composite objectives. In *Advances in Neural Information Processing Systems*, 2016.

[28] B. Woodworth, J. Wang, A. Smith, B. McMahan, and N. Srebro. Graph oracle models, lower bounds, and gaps for parallel stochastic optimization. *arXiv preprint arXiv:1805.10222*, 2018.

[29] J. Wu, W. Hu, H. Xiong, J. Huan, V. Braverman, and Z. Zhu. On the noisy gradient descent that generalizes as sgd. In *International Conference on Machine Learning*, pages 10367–10376. PMLR, 2020.

[30] Z. Zhu, J. Wu, B. Yu, L. Wu, and J. Ma. The anisotropic noise in stochastic gradient descent: Its behavior of escaping from sharp minima and regularization effects. In K. Chaudhuri and R. Salakhutdinov, editors, *Proceedings of the 36th International Conference on Machine Learning, ICML 2019, 9-15 June 2019, Long Beach, California, USA*, volume 97 of *Proceedings of Machine Learning Research*, pages 7654–7663. PMLR, 2019.

Checklist

1. For all authors...
   (a) Do the main claims made in the abstract and introduction accurately reflect the paper's contributions and scope? [Yes]
   (b) Did you describe the limitations of your work? [Yes]
   (c) Did you discuss any potential negative societal impacts of your work? [N/A]
   (d) Have you read the ethics review guidelines and ensured that your paper conforms to them? [Yes]
2. If you are including theoretical results...
   (a) Did you state the full set of assumptions of all theoretical results? [Yes]
   (b) Did you include complete proofs of all theoretical results? [Yes] See the supplementary material
3. If you ran experiments...

(a) Did you include the code, data, and instructions needed to reproduce the main experimental results (either in the supplemental material or as a URL)? [N/A]

(b) Did you specify all the training details (e.g., data splits, hyperparameters, how they were chosen)? [N/A]

(c) Did you report error bars (e.g., with respect to the random seed after running experiments multiple times)? [N/A]

(d) Did you include the total amount of compute and the type of resources used (e.g., type of GPUs, internal cluster, or cloud provider)? [N/A]

4. If you are using existing assets (e.g., code, data, models) or curating/releasing new assets...

(a) If your work uses existing assets, did you cite the creators? [N/A]

(b) Did you mention the license of the assets? [N/A]

(c) Did you include any new assets either in the supplemental material or as a URL? [N/A]

(d) Did you discuss whether and how consent was obtained from people whose data you're using/curating? [N/A]

(e) Did you discuss whether the data you are using/curating contains personally identifiable information or offensive content? [N/A]

5. If you used crowdsourcing or conducted research with human subjects...

(a) Did you include the full text of instructions given to participants and screenshots, if applicable? [N/A]

(b) Did you describe any potential participant risks, with links to Institutional Review Board (IRB) approvals, if applicable? [N/A]

(c) Did you include the estimated hourly wage paid to participants and the total amount spent on participant compensation? [N/A]