On Graduated Optimization for Stochastic Non-Convex Problems

Elad Hazan EHAZAN@CS.PRINCETON.EDU
Princeton University

Kfir Y. Levy KFIRYL@TX.TECHNION.AC.IL
Technion - Israel Institute of Technology

Shai Shalev-Shwartz SHAIS@CS.HUJI.AC.IL
The Hebrew University of Jerusalem, Israel

Abstract

The graduated optimization approach, also known as the continuation method, is a popular heuristic to solving non-convex problems that has received renewed interest over the last decade. Despite being popular, very little is known in terms of its theoretical convergence analysis.

In this paper we describe a new first-order algorithm based on graduated optimization and analyze its performance. We characterize a family of non-convex functions for which this algorithm provably converges to a **global** optimum. In particular, we prove that the algorithm converges to an ε-approximate solution within $O(1/\varepsilon^2)$ gradient-based steps. We extend our algorithm and analysis to the setting of stochastic non-convex optimization with noisy gradient feedback, attaining the same convergence rate. Additionally, we discuss the setting of "zero-order optimization", and devise a variant of our algorithm which converges at rate of $O(d^2/\varepsilon^4)$.

1. Introduction

Non-convex optimization programs are ubiquitous in machine learning and computer vision. Of particular interest are non-convex optimization problem that arise in the training of deep neural networks (Bengio, 2009). Often, such problems admit a multimodal structure, and therefore, the use of convex optimization machinery may lead to poor local optima.

Graduated optimization (a.k.a. continuation), (Blake & Zisserman, 1987), is a methodology that attempts to overcome such numerous local optima. Initially, a simpler coarse-grained version of the objective is generated and minimized. Then, the method progresses in stages, gradually refining the versions of the objective, and using the solution of the previous stage as an initial point for the optimization in the next stage.

Despite its popularity, there are still many gaps concerning both theoretical and practical aspects of graduated optimization, and in particular we are not aware of a rigorous running time analysis to find a global optimum, or even conditions in which a global optimum is reached. Nor are we familiar with graduated optimization in the stochastic setting, in which only a noisy gradient or value oracle to the objective is given. Moreover, any practical application of graduated optimization requires to efficiently construct coarse-grained versions of the original objective. For some special cases this construction can be made analytically (Chapelle et al., 2006; Chaudhuri & Solar-Lezama, 2011) . However, in the general case, it is commonly suggested in the literature to convolve the original function with a gaussian kernel (Wu, 1996). Yet, this operation is prohibitively inefficient in high dimensions.

Here we take an algorithmic / analytic approach to graduated optimization and show the following:

- We characterise *σ-niceness* (Def. 3.2), a property of non-convex multimodal functions which captures non-convex structure that appears in challenging optimization problems.
- We provide a stochastic algorithm inspired by graduated optimization, that performs only gradient updates and is ensured to find an ε-optimal solution of σ-nice functions within $O(1/\sigma^2\varepsilon^2)$ iterations. Our algorithm

Proceedings of the 33rd International Conference on Machine Learning, New York, NY, USA, 2016. JMLR: W&CP volume 48. Copyright 2016 by the author(s).

does not require expensive convolutions and gains access to the smoothed versions of any function using random sampling. The algorithm only requires access to the objective function through a *noisy* gradient oracle.

- We extend our method to the "zero-order optimization" model (a.k.a. "bandit feedback" model), where the objective is only accessible through a noisy value oracle. We devise a variant of our algorithm that obtains an ε-optimal solution within $O(d^2/\sigma^2\varepsilon^4)$ iterations.

Interestingly, the next question is raised in (Bengio, 2009) which reviews recent developments in the field of deep learning: **"Can optimization strategies based on continuation methods deliver significantly improved training of deep architectures?"**

As an initial attempt to establish the effectiveness of graduated optimization, we examine the task of training a NN (Neural Network) over the MNIST data set. Our experiments support the theoretical guarantees, substantiating an accelerated convergence in training the NN. Moreover, we demonstrate a non-convex phenomena that exists in natural data, and is captured by the σ-nice property.

1.1. Related Work

Among the machine vision community, the idea of graduated optimization was known since the 80's. The term "Graduated Non-Convexity" (GNC) was coined by (Blake & Zisserman, 1987), who were the first to establish this idea explicitly. Similar attitudes in the machine vision literature appeared later in (Yuille, 1989; Yuille et al., 1990), and (Terzopoulos, 1988). Concepts of the same nature appeared in the optimization literature (Wu, 1996), and in the field of numerical analysis (Allgower & Georg, 1990).

Over the last two decades, this concept was successfully applied to numerous problems in computer vision; among are: image deblurring (Boccuto et al., 2002) , image restoration (Nikolova et al., 2010), and optical flow (Brox & Malik, 2011). The method was also adopted by the machine learning community, demonstrating effective performance in tasks such as semi-supervised learning (Chapelle et al., 2006), graph matching (Zaslavskiy et al., 2009), and ranking (Chapelle & Wu, 2010). In (Bengio, 2009), it is suggested to consider some developments in deep belief architectures (Hinton et al., 2006; Erhan et al., 2009) as a kind of continuation. These approaches, in the spirit of the continuation method, offer no guarantees on the quality of the obtained solution, and are tailored to specific applications. A comprehensive survey of the graduated optimization literature can be found in (Mobahi & Fisher III, 2015a).

A recent work (Mobahi & Fisher III, 2015b) advances our theoretical understanding, by analyzing a continuation algorithm in the general setting. Yet, they offer no way to perform the smoothing efficiently, nor a way to optimize the smoothed versions; but rather assume that these are possible. Moreover, their guarantee is limited to a fixed precision that depends on the objective function and does not approach zero. In contrast, our approach can generate arbitrarily precise solutions.

2. Setting and Background

Notation and Preliminaries: During this paper we use $\mathbb{B}, \mathbb{S}$ to denote the unit Euclidean ball/sphere in $\mathbb{R}^d$, and also $\mathbb{B}_r(\mathbf{x}), \mathbb{S}_r(\mathbf{x})$ as the Euclidean r-ball/sphere in $\mathbb{R}^d$ centered at $\mathbf{x}$. For a set $A \subset \mathbb{R}^d$, $\mathbf{u} \sim A$ denotes a random variable distributed uniformly over A.

Recall the definition of strongly-convex functions,

Definition 2.1. *A function $F : \mathbb{R}^n \to \mathbb{R}$ is said to be σ-strongly convex over a set $\mathcal{K}$ if for any $\mathbf{x}, \mathbf{y} \in \mathcal{K}$ the following holds,*

$$F(\mathbf{y}) \geq F(\mathbf{x}) + \nabla F(\mathbf{x})^\top(\mathbf{y}-\mathbf{x}) + \frac{\sigma}{2}\|\mathbf{x}-\mathbf{y}\|^2 .$$

Let F be a σ-strongly convex over convex set $\mathcal{K}$, and let $\mathbf{x}^*$ be a point in $\mathcal{K}$ where F is minimized, then the following inequality is satisfied:

$$\frac{\sigma}{2}\|\mathbf{x}-\mathbf{x}^*\|^2 \leq F(\mathbf{x}) - F(\mathbf{x}^*) \qquad (1)$$

This is immediate by the definition of strong convexity combined with $\nabla F(\mathbf{x}^*)^\top(\mathbf{x}-\mathbf{x}^*) \geq 0,\ \forall \mathbf{x} \in \mathcal{K}$.

2.1. Stochastic Optimization with Gradient/Value Feedback

We discuss an optimization of a loss function $f : \mathcal{K} \mapsto \mathbb{R}$, where $\mathcal{K} \subseteq \mathbb{R}^d$ is a convex set. We assume that optimization lasts for T rounds; on each round $t = 1, \ldots, T$, we may query a point $\mathbf{x}_t \in \mathcal{K}$, and receive a *feedback*. After the last round, we choose $\bar{\mathbf{x}}_T \in \mathcal{K}$, and our performance measure is the excess loss, defined as,

$$f(\bar{\mathbf{x}}_T) - \min_{\mathbf{x}\in\mathcal{K}} f(\mathbf{x}) .$$

In Section 3.2 we characterize a family of non-convex functions we denote by σ-nice. Given such a function, we are interested in algorithms that obtain an ε-excess loss within $\text{poly}(1/\varepsilon)$ rounds.

We consider two kinds of feedback:

1. **Noisy Gradient feedback:** Upon querying $\mathbf{x}_t$ we receive $\nabla f(\mathbf{x}_t) + \xi_t$, where $\{\xi_\tau\}_{\tau=1}^T$ are independent zero mean and bounded random variables.

Oracle 1: SGO$_G$
Input: $\mathbf{x} \in \mathbb{R}^d$, smoothing parameter δ
Return: $\nabla f(\mathbf{x} + \delta\mathbf{u})$, where $\mathbf{u} \sim \mathbb{B}$

Figure 1. Smoothed gradient oracle given gradient feedback.

Oracle 2: SGO$_V$
Input: $\mathbf{x} \in \mathbb{R}^d$, smoothing parameter δ
Return: $\frac{d}{\delta} f(\mathbf{x} + \delta\mathbf{v})\mathbf{v}$, where $\mathbf{v} \sim \mathbb{S}$

Figure 2. Smoothed gradient oracle given value feedback.

2. **Noisy Value feedback (Bandit feedback):** Upon querying $\mathbf{x}_t$ we receive $f(\mathbf{x}_t) + \xi_t$, where $\{\xi_\tau\}_{\tau=1}^T$ are independent zero mean and bounded random variables.

3. Smoothing and σ-Nice functions

Constructing finer and finer approximations to the original objective function is at the heart of the continuation approach. In Section 3.1 we define the smoothed versions that we employ. Next, in Section 3.1.1 we describe an efficient way to implicitly access the smoothed versions, which will enable us to perform optimization. In Section 3.2 we define the class of *σ-nice* functions, and show in Section 3.2.1 that it captures the "valley" phenomenon. "Valley" is a non-convex structure that might prevent gradient descent methods from approaching the global minimum, and appears in challenging optimization problems as we describe in Section 7.

3.1. Smoothing

Smoothing by local averaging is formally defined next.

Definition 3.1. *Given an L-Lipschitz function $f : \mathbb{R}^d \mapsto \mathbb{R}$ define its δ-smooth version to be*

$$\hat{f}_\delta(\mathbf{x}) = \mathbf{E}_{\mathbf{u}\sim\mathbb{B}}[f(\mathbf{x} + \delta\mathbf{u})].$$

The next lemma bounds the bias between $\hat{f}_\delta$ and f.

Lemma 3.1. *Let $\hat{f}_\delta$ be the δ-smoothed version of f, then, $\forall\mathbf{x} \in \mathbb{R}^d$, $|\hat{f}_\delta(\mathbf{x}) - f(\mathbf{x})| \le \delta L$.*

3.1.1. Implicit Smoothing using Sampling

A direct way to optimize a smoothed version is by an explicit calculation of its gradients, nevertheless this might be very costly in high dimensions. A much more efficient approach is to produce an unbiased estimate for the gradients of the smoothed version by sampling the gradients/values of the function. These estimates could then be used by a stochastic optimization algorithms such as SGD (Stochastic Gradient Descent). This sampling approach is outlined in Figures 1,2.

The following two Lemmas state that the resulting estimates are unbiased and bounded [1]:

Lemma 3.2. *Let $\mathbf{x} \in \mathbb{R}^d$, $\delta \ge 0$, and suppose that f is L-Lipschitz, then the output of SGO$_G$ (Figure 1) is bounded by L and is an unbiased estimate for $\nabla\hat{f}_\delta(\mathbf{x})$.*

Lemma 3.3. *Let $\mathbf{x} \in \mathcal{K} \subseteq \mathbb{R}^d$, $\delta \ge 0$, and suppose that $\max_\mathbf{x} |f(\mathbf{x})| \le C$, then the output of SGO$_V$ (Figure 2) is bounded by $\frac{dC}{\delta}$ and is an unbiased estimate for $\nabla\hat{f}_\delta(\mathbf{x})$.*

Extensions to the *noisy* feedback settings: Note that for ease of notation, the oracles that appear in Figures 1, 2, assume we can access *exact* gradients/values of f. Given that we may only access *noisy* and bounded gradient/value estimates of f (Sec. 2.1), we could use these instead of the exact ones that appear in Figures 1,2, and still produce unbiased and bounded estimates of $\nabla\hat{f}_\delta(x)$.

3.2. σ-Nice Functions

Following is our main definition

Definition 3.2 ((a,σ)-Nice)**.** *Let $a, \sigma > 0$. A function $f : \mathcal{K} \mapsto \mathbb{R}$ is said to be (a,σ)-nice if the following holds:*

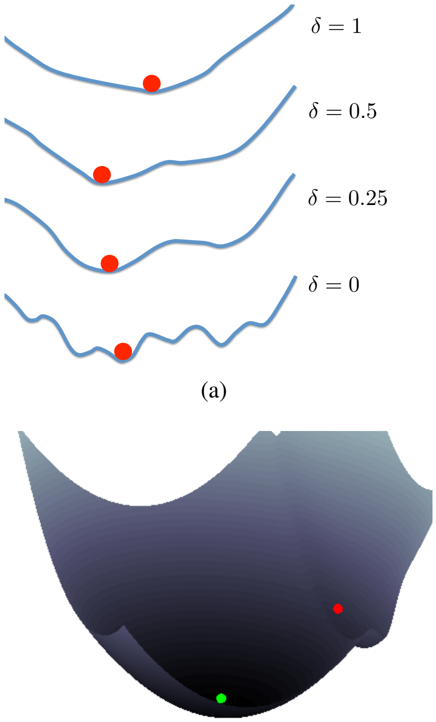
1. ***Centering property:*** *For every $\delta > 0$, and every $\mathbf{x}^*_\delta \in \arg\min_{\mathbf{x}\in\mathcal{K}} \hat{f}_{a\delta}(\mathbf{x})$, there exists $\mathbf{x}^*_{\delta/2} \in \arg\min_{x\in\mathcal{K}} \hat{f}_{a\delta/2}(\mathbf{x})$, such that $\|\mathbf{x}^*_\delta - \mathbf{x}^*_{\delta/2}\| \le \delta/2$.*
2. ***Local strong convexity of the smoothed versions:*** *For every $\delta > 0$, let $r_\delta = 3\delta$, and denote $\mathbf{x}^*_\delta = \arg\min_{\mathbf{x}\in\mathcal{K}} \hat{f}_{a\delta}(\mathbf{x})$, then over $B_{r_\delta}(\mathbf{x}^*_\delta)$ the function $\hat{f}_{a\delta}(\mathbf{x})$ is σ-strongly-convex.*

In case that $a = 1$ we say that f is σ-nice.

Hence, (a,σ)-nice is a combination of two properties. Both together imply that optimizing the smoothed version on a scale $a\delta$ is a good start for optimizing a finer version on a scale of $a\delta/2$.

For simplicity we will only analyze the case of σ-nice functions. The analysis for the case when $a \ne 1$ is similar. In Section 3.2.1 we discuss a non-convex phenomenon that admits the (a,σ)-nice property. In Section 7 we show this phenomenon to arise naturally in data. An illustration of a 1-dim, σ-nice function appears in Fig. 3(a).

[1] Note that the oracles depicted in Figures 1,2 may require sampling function gradients/values outside $\mathcal{K}$, (specifically in $\mathcal{K}+\delta\mathbb{B}$). We assume that this is possible, and that the bounds over the function gradients/values inside $\mathcal{K}$, also applies in $\mathcal{K}+\delta\mathbb{B}$.



(a)

(b)

Figure 3. Top: a 1-dim σ-nice function ($\delta = 0$), and its smoothed versions. Bottom: the valley phenomenon, green marker–global optimum, red marker– a point inside the valley.

3.2.1. The Valley Phenomenon

In this section we discuss the *valley*, a non-convex phenomenon in which a local minima is present. The valley might "entrap" the gradient descent algorithm, preventing it from approaching the global optimum. We show that a quadratic function with a local valley admits the (a, σ)-nice structure, implying that our graduated optimization algorithm (Alg. 1), overcomes the non-convex pitfall and converges to the global minima. Section 7 demonstrates this phenomenon in a task of training a NN over the MNIST dataset (Fig 4). We show that SGD stalls due to the valley, however smoothing enables to escape the local hurdle and reach a better solution.

Let $\mathbf{x} = (x_1, \ldots, x_d) \in \mathbb{R}^d$. A quadratic function with a one dimensional valley is defined by,

$$f(\mathbf{x}) = 0.5\|\mathbf{x}\|^2 + g(\mathbf{x}) = 0.5\|\mathbf{x}\|^2 - \alpha e^{-\frac{(x_1-1)^2}{2\lambda^2}} \ , \quad (2)$$

where we refer to $g(\mathbf{x}) = -\alpha e^{-\frac{(x_1-1)^2}{2\lambda^2}}$ as the valley function. Note that the valley is centered around $x_1 = 1$, and that the width of valley is controlled by λ. In Figure 3(b) we present a two dimensional graph of the function described in Equation (2). The following Lemma states that the above function is either strongly-convex or (a, σ)-nice:

Algorithm 1 GradOpt$_G$

```
Input: target error $\varepsilon$, maximal failure probability $p$, decision set $\mathcal{K}$
Choose $\bar{\mathbf{x}}_1 \in \mathcal{K}$ uniformly at random.
Set $\delta_1 = \text{diam}(\mathcal{K})$, $\tilde{p} = p/M$, and $M = \log_2 \frac{1}{\alpha_0 \varepsilon}$ where
$\alpha_0 = \min\left\{\frac{1}{2L\text{diam}(\mathcal{K})}, \frac{4}{\sqrt{\sigma}\text{diam}(\mathcal{K})}\right\}$
for $m = 1$ to $M$ do
    // Perform SGD over $\hat{f}_{\delta_m}$
    Set $\varepsilon_m := \sigma\delta_m^2/32$, and
        $T_F = \frac{12480L^2}{\sigma\varepsilon_m} \log\left(\frac{2}{\tilde{p}} + 2\log\frac{12480L^2}{\sigma\varepsilon_m}\right)$
    Set shrinked decision set,
        $\mathcal{K}_m := \mathcal{K} \cap B(\bar{x}_m, 1.5\delta_m)$
    Set gradient oracle for $\hat{f}_{\delta_m}$,
        $\text{GradOracle}(\cdot) = \text{SGO}_G(\cdot, \delta_m)$
    Update:
        $\bar{\mathbf{x}}_{m+1} \leftarrow \text{Suffix-SGD}(T_F, \mathcal{K}_m, \bar{\mathbf{x}}_m, \text{GradOracle})$
    $\delta_{m+1} = \delta_m/2$
end for
Return: $\bar{\mathbf{x}}_{M+1}$
```

Lemma 3.4. *Let f be the function described in Equation (2), and assume $\alpha \in [0, \frac{1}{200}]$. If $\lambda \leq 0.1$ then f is $(\sqrt{d}, 0.5)$-nice; Otherwise, f is 0.5-strongly-convex.*

4. Graduated Optimization with a Gradient Oracle

Here we assume that we may access a noisy gradient oracle for f. Thus, given $\mathbf{x} \in \mathbb{R}^d, \delta \geq 0$ we can use SGO$_G$ (Figure 1) to obtain an unbiased and bounded estimate for $\nabla\hat{f}_\delta(\mathbf{x})$.

Algorithm 2 Suffix-SGD

```
Input: total time $T_F$, decision set $\mathcal{K}$, initial point $\mathbf{x}_1 \in \mathcal{K}$, gradient oracle GradOracle$(\cdot)$
for $t = 1$ to $T_F$ do
    Set $\eta_t = 1/\sigma t$
    Query $g_t \leftarrow \text{GradOracle}(\mathbf{x}_t)$
    Update $\mathbf{x}_{t+1} \leftarrow \Pi_{\mathcal{K}}(\mathbf{x}_t - \eta_t g_t)$
end for
Return: $\bar{\mathbf{x}}_{T_F} := \frac{2}{T_F}\left(\mathbf{x}_{T_F/2+1} + \ldots + \mathbf{x}_{T_F}\right)$
```

Following is our main Theorem:

Theorem 4.1. *Let $\varepsilon \in (0,1)$ and $p \in (0,1/e)$, also let $\mathcal{K}$ be a convex set, and f be an L-Lipschitz σ-nice function. Suppose that we apply Algorithm 1, then after $\tilde{O}(1/\sigma^2\varepsilon^2)$ optimization steps, Algorithm 1 outputs a point $\bar{\mathbf{x}}_{M+1}$ which is ε optimal with a probability greater than $1-p$.*

Remark: Note that for (a,σ)-nice functions, we can prove a bound of $\tilde{O}(a^2/\sigma^2\varepsilon^2)$ on the number of steps required to attain an ε-optimal solution. The proof is similar to the one of Theorem 4.1 and we therefore omit the details.

Algorithm 1 is divided into epochs, at epoch m it uses SGO_G to obtain unbiased estimates for the gradients of $\hat{f}_{\delta_m}$ which are then employed by Suffix-SGD (Algorithm 2), to optimize this smoothed version. This optimization over $\hat{f}_{\delta_m}$ is performed until we are ensured to reach a point close enough to $\mathbf{x}^*_{m+1} := \arg\min_{\mathbf{x}\in\mathcal{K}} \hat{f}_{\delta_{m+1}}(\mathbf{x})$, i.e., the minimum of $\hat{f}_{\delta_{m+1}}$. Also note that at epoch m the optimization over $\hat{f}_{\delta_m}$ is initialized at $\bar{\mathbf{x}}_m$ which is the point reached at the previous epoch. Suffix-SGD (Algorithm 2), is a stochastic optimization algorithm for strongly convex functions. Its guarantees are presented in Section 4.1.

4.1. Analysis

First, note that Suffix-SGD performs projected gradient descent using the gradients received by GradOracle$(\cdot)$. The projection operator $\Pi_{\mathcal{K}}$, is defined $\forall \mathbf{y} \in \mathbb{R}^d$ as $\Pi_{\mathcal{K}}(\mathbf{y}) := \arg\min_{\mathbf{x}\in\mathcal{K}} \|\mathbf{x}-\mathbf{y}\|$.

The following lemma from (Rakhlin et al., 2011), states the performance guarantees of Suffix-SGD (Algorithm 2):

Theorem 4.2. *Let $p \in (0,1/e)$, and F be a σ-strongly convex function. Suppose that GradOracle$(\cdot)$ produces G-bounded, and unbiased estimates of ∇F. Then after no more than T_F rounds, the final point $\bar{\mathbf{x}}_{T_F}$ returned by Suffix-SGD (Algorithm 2) ensures that with a probability $\geq 1-p$, we have:*

$$F(\bar{\mathbf{x}}_{T_F}) - \min_{\mathbf{x}\in\mathcal{K}} F(\mathbf{x}) \leq \frac{6240\log\left(2\log(T_F)/p\right)G^2}{\sigma T_F} .$$

Corollary 4.1. *The latter means that for $T_F \geq \frac{12480G^2}{\sigma\varepsilon}\log\left(2/p + 2\log(12480G^2/\sigma\varepsilon)\right)$ we will have an excess loss smaller than ε.*

Notice that at each epoch, m, of GradOpt_G, it initiates Suffix-SGD with a gradient oracle $\text{SGO}_G(\cdot,\delta_m)$ which produces an unbiased and L-bounded estimates of $\hat{f}_{\delta_m}$ (Lemma 3.2). Thus in the analysis of each epoch we can use Theorem 4.2 for $\hat{f}_{\delta_m}$, taking $G=L$.

Following is our key Lemma:

Lemma 4.1. *Consider M, $\mathcal{K}_m$ and $\bar{\mathbf{x}}_{m+1}$ as defined in Algorithm 1. Also denote by $\mathbf{x}^*_m$ the minimizer of $\hat{f}_{\delta_m}$ in $\mathcal{K}$. Then the following holds for all $1 \leq m \leq M$ w.p.$\geq 1-p$:*

1. *The smoothed version $\hat{f}_{\delta_m}$is σ-strongly convex over $\mathcal{K}_m$, and $\mathbf{x}^*_m \in \mathcal{K}_m$.*
2. *Also, $\hat{f}_{\delta_m}(\bar{\mathbf{x}}_{m+1}) - \hat{f}_{\delta_m}(\mathbf{x}^*_m) \leq \sigma\delta^2_{m+1}/8$.*

Proof. We prove by induction. Let us prove that the lemma holds for $m=1$. Note that $\delta_1 = \text{diam}(\mathcal{K})$, therefore $\mathcal{K}_1 = \mathcal{K}$, and also $\mathbf{x}^*_1 \in \mathcal{K}_1$. Also recall that σ-niceness of f implies that $\hat{f}_{\delta_1}$ is σ-strongly convex in $\mathcal{K}$, thus by Corollary 4.1, after less than $T_F = \tilde{O}(\frac{12480L^2}{\sigma(\sigma\delta_1^2/32)})$ optimization steps of Suffix-SGD with a probability greater than $1-p/M$, we will have:

$$\hat{f}_{\delta_1}(\bar{\mathbf{x}}_2) - \hat{f}_{\delta_1}(\mathbf{x}^*_1) \leq \sigma\delta_1^2/32 = \sigma\delta_2^2/8 .$$

which establishes the case of $m=1$. Now assume that lemma holds for $m>1$. By this assumption, $\hat{f}_{\delta_m}(\bar{\mathbf{x}}_{m+1}) - \hat{f}_{\delta_m}(\mathbf{x}^*_m) \leq \sigma\delta^2_{m+1}/8$, $\hat{f}_{\delta_m}$ is σ-strongly convex in $\mathcal{K}_m$, and also $\mathbf{x}^*_m \in \mathcal{K}_m$. The σ-strong-convexity in $\mathcal{K}_m$ implies,

$$\|\bar{\mathbf{x}}_{m+1} - \mathbf{x}^*_m\| \leq \sqrt{\frac{2}{\sigma}}\sqrt{\hat{f}_{\delta_m}(\bar{\mathbf{x}}_{m+1}) - \hat{f}_{\delta_m}(\mathbf{x}^*_m)} \leq \frac{\delta_{m+1}}{2} .$$

Combining the latter with the centering property of σ-niceness yields:

$$\begin{aligned}\|\bar{\mathbf{x}}_{m+1} - \mathbf{x}^*_{m+1}\| &\leq \|\bar{\mathbf{x}}_{m+1} - \mathbf{x}^*_m\| + \|\mathbf{x}^*_m - \mathbf{x}^*_{m+1}\| \\ &\leq 1.5\delta_{m+1} ,\end{aligned}$$

and it follows that,

$$\mathbf{x}^*_{m+1} \in B(\bar{\mathbf{x}}_{m+1}, 1.5\delta_{m+1}) \subset B(\mathbf{x}^*_{m+1}, 3\delta_{m+1}) .$$

Recalling that $\mathcal{K}_{m+1} := B(\bar{\mathbf{x}}_{m+1}, 1.5\delta_{m+1})$, and the local strong convexity property of f (which is σ-nice), then the induction step for first part of the lemma holds. Now, by Corollary 4.1, after less than $T_F = \tilde{O}(\frac{12480L^2}{\sigma(\sigma\delta^2_{m+1}/32)})$ optimization steps of Suffix-SGD over $\hat{f}_{\delta_{m+1}}$, we will have:

$$\hat{f}_{\delta_{m+1}}(\bar{\mathbf{x}}_{m+2}) - \hat{f}_{\delta_{m+1}}(\mathbf{x}^*_{m+1}) \leq \sigma\delta^2_{m+2}/8 .$$

which establishes the induction step for the second part of the lemma.

An analysis of fail probability: since we have M epochs in total and at each epoch the fail probability is smaller than p/M, then the total fail probability of our algorithm is smaller than p. □

We are now ready to prove Theorem 4.1

Proof of Theorem 4.1. Algorithm 1 terminates after $M = \log_2\frac{1}{\alpha_0\varepsilon}$ epochs meaning, $\delta_M = \text{diam}(\mathcal{K})\alpha_0\varepsilon$. According to Lemma 4.1 the following holds w.p.$\geq 1-p$, for every $\mathbf{x} \in \mathcal{K}$,

$$\hat{f}_{\delta_M}(\bar{\mathbf{x}}_{M+1}) - \hat{f}_{\delta_M}(\mathbf{x}) \leq \sigma\delta_{M+1}^2/8 = \left(\frac{\sqrt{\sigma}\mathrm{diam}(\mathcal{K})\alpha_0\varepsilon}{4\sqrt{2}}\right)^2 .$$

Due to Lemma 3.1, $\hat{f}_{\delta_M}$ is $L\delta_M$ biased from f, using the definition of α_0, we conclude that $\forall \mathbf{x} \in \mathcal{K}$,

$$f(\bar{\mathbf{x}}_{M+1}) - f(\mathbf{x}) \leq L\mathrm{diam}(\mathcal{K})\alpha_0\varepsilon + \left(\frac{\sqrt{\sigma}\mathrm{diam}(\mathcal{K})\alpha_0\varepsilon}{4\sqrt{2}}\right)^2 \leq \varepsilon .$$

The series of smoothing parameters $\{\delta_m\}_{m=1}^M$ decays as geometric series with a decay factor of 2, it is therefore possible to show that the total number of optimization steps made by Algorithm 1 is $\tilde{O}(1/\sigma^2\varepsilon^2)$. Indeed, let T_{total} be the total number of queries made by by Algorithm 1, then we have:

$$\begin{aligned}
T_{\text{total}} &\leq \sum_{m=1}^{M} \frac{12480L^2}{\sigma\varepsilon_m}\log\Gamma \\
&\leq \sum_{m=1}^{M} \frac{12480L^2}{\sigma(\sigma\delta_m^2/32)}\log\Gamma \\
&\leq \frac{4\cdot 10^5 L^2\log\Gamma}{\sigma^2}\sum_{i=1}^{M}\frac{4^{i-1}}{\delta_1^2} \\
&\leq \frac{14\cdot 10^4 L^2\log\Gamma}{\sigma^2}\frac{4^M}{\delta_1^2} \\
&\leq \frac{14\cdot 10^4 L^2\log\Gamma}{\sigma^2}\max\{16L^2, \sigma/2\}\frac{1}{\varepsilon^2} ,
\end{aligned}$$

here we used the notation:

$$\begin{aligned}
\Gamma &:= \frac{2M}{p} + 2\log(12480L^2/\sigma\varepsilon_M) \\
&\leq \frac{2M}{p} + 2\log(4\cdot 10^5 L^2\max\{16L^2, \frac{\sigma}{2}\}/\sigma^2\varepsilon^2) .
\end{aligned}$$

□

5. Graduated Optimization with a Value Oracle

In this section we assume that we can access a noisy value oracle for f. Thus, given $\mathbf{x} \in \mathbb{R}^d, \delta \geq 0$ we can use SGO_V (Figure 2) as an oracle that outputs an unbiased and bounded estimates for $\nabla\hat{f}_\delta(\mathbf{x})$, as ensured by Lemma 3.3.

Following is our main Theorem:

Algorithm 3 $\mathrm{GradOpt}_V$

```
Input: target error $\varepsilon$, maximal failure probability $p$, decision set $\mathcal{K}$
Choose $\bar{\mathbf{x}}_1 \in \mathcal{K}$ uniformly at random.
Set $\delta_1 = \mathrm{diam}(\mathcal{K})/2$, $\tilde{p} = p/M$, and $M = \log_2\frac{1}{\alpha_0\varepsilon}$
where $\alpha_0 = \min\{\frac{1}{2L\mathrm{diam}(\mathcal{K})}, \frac{2\sqrt{2}}{\sqrt{\sigma}\mathrm{diam}(\mathcal{K})}\}$
for $m = 1$ to $M$ do
    // Perform SGD over $\hat{f}_{\delta_m}$
    Set $\varepsilon_m := \sigma\delta_m^2/32$, and
        $T_F = \frac{12480}{\sigma\varepsilon_m}\frac{d^2C^2}{\delta_m^2}\log\left(\frac{2}{\tilde{p}} + 2\log\frac{12480d^2C^2}{\sigma\varepsilon_m\delta_m^2}\right)$
    Set shrinked decision set,
        $\mathcal{K}_m := \mathcal{K} \cap B(\bar{x}_m, 1.5\delta_m)$
    Set gradient oracle for $\hat{f}_{\delta_m}$,
        $\mathrm{GradOracle}(\cdot) = \mathrm{SGO}_V(\cdot, \delta_m)$
    Update:
        $\bar{\mathbf{x}}_{m+1} \leftarrow \text{Suffix-SGD}(T_F, \mathcal{K}_m, \bar{\mathbf{x}}_m, \mathrm{GradOracle})$
    $\delta_{m+1} = \delta_m/2$
end for
Return: $\bar{\mathbf{x}}_{M+1}$
```

Theorem 5.1. *Let $\varepsilon > 0$ and $p \in (0, 1/e)$, also let $\mathcal{K}$ be a convex set, and f be an L-Lipschitz σ-nice function. Assume also that* $\max_{\mathbf{x}} |f(\mathbf{x})| \leq C$*. Suppose that we apply Algorithm 3, then after after $\tilde{O}(d^2/\sigma^2\varepsilon^4)$ rounds Algorithm 3 outputs a point $\bar{\mathbf{x}}_{M+1}$ which is ε optimal with a probability greater than $1-p$.*

Note that Algorithm 3 and its analysis are similar to the setting presented in Section 4, where a gradient oracle is available. The key difference is the use of SGO_V (Figure 2), instead of SGO_G, in order to obtain smoothed gradient estimates. We therefore defer the proofs to the full version of the paper.

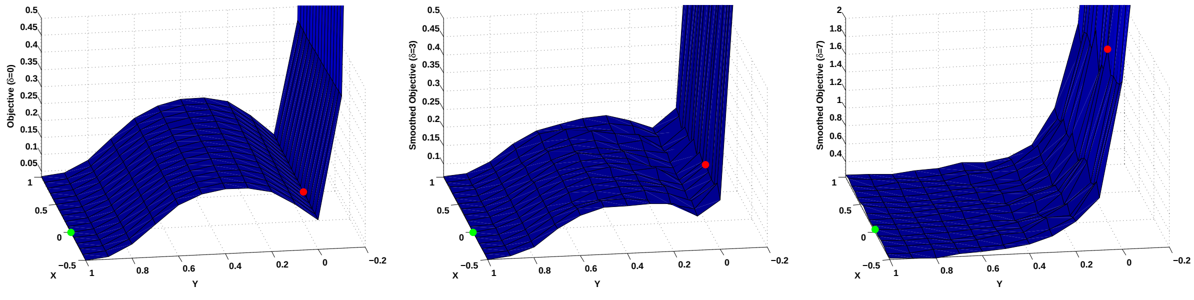
6. Omitted Proofs

6.1. Proof of Lemma 3.1

Proof.

$$\begin{aligned}
|\hat{f}_\delta(\mathbf{x}) - f(\mathbf{x})| &= |\mathbf{E}_{u\sim\mathbb{B}}\left[f(\mathbf{x}+\delta\mathbf{u})\right] - f(\mathbf{x})| \\
&\leq \mathbf{E}_{\mathbf{u}\sim\mathbb{B}}\left[|f(\mathbf{x}+\delta\mathbf{u}) - f(\mathbf{x})|\right] \\
&\leq \mathbf{E}_{\mathbf{u}\sim\mathbb{B}}\left[L\|\delta\mathbf{u}\|\right] \\
&\leq L\delta
\end{aligned}$$

in the first inequality we used Jensen's inequality, and in the last inequality we used $\|\mathbf{u}\| \leq 1$, since $\mathbf{u} \in \mathbb{B}$. □



(a) (b) (c)

Figure 4. The objective near a stall point. Left: $\delta = 0$. Middle: $\delta = 3$. Right: $\delta = 7$.

6.2. Proof of Lemma 3.2

Proof. SGO$_G$ outputs $\nabla f(\mathbf{x} + \delta\mathbf{u})$ for some $\mathbf{u} \in \mathbb{B}$, so the first part is immediate by the Lipschitzness of f. Now, by definition, $\hat{f}_\delta(\mathbf{x}) = \mathbf{E}_{\mathbf{u}\sim\mathbb{B}}[f(\mathbf{x} + \delta\mathbf{u})]$, deriving both sides we get the second part of the Lemma. □

6.3. Proof of Lemma 3.3

Proof. SGO$_V$ outputs $\frac{d}{\delta} f(\mathbf{x} + \delta\mathbf{v})\mathbf{v}$ for some $\mathbf{v} \in \mathbb{S}$, since f is C-Bounded over $\mathcal{K}$ the first part of the lemma is immediate. In order to prove the second part, we can use Stokes theorem to show that if $\mathbf{v} \sim \mathbb{S}$, then:

$$\forall \mathbf{x} \in \mathbb{R}^d \;.\; \mathbf{E}_{\mathbf{v}\sim\mathbb{S}}[f(\mathbf{x} + \delta\mathbf{v})\mathbf{v}] = \frac{\delta}{d}\nabla\hat{f}_\delta(\mathbf{x}) \qquad (3)$$

A proof of Equation (3) is found in (Flaxman et al., 2005). □

7. Experiments

In the last two decades, performing complex learning tasks using Neural-Network (NN) architectures has become an active and promising line of research. Since learning NN architectures essentially requires to solve a hard non-convex program, we have decided to focus our empirical study on this type of tasks. As a test case, we train a NN with a single hidden layer of 30 units over the MNIST data set. We adopt the experimental setup of (Dauphin et al., 2014) and train over a down-scaled version of the data, i.e., the original 28×28 images of MNIST were down-sampled to the size of 10×10. We use a ReLU activation function, and minimize the square loss.

7.1. Smoothing the NN

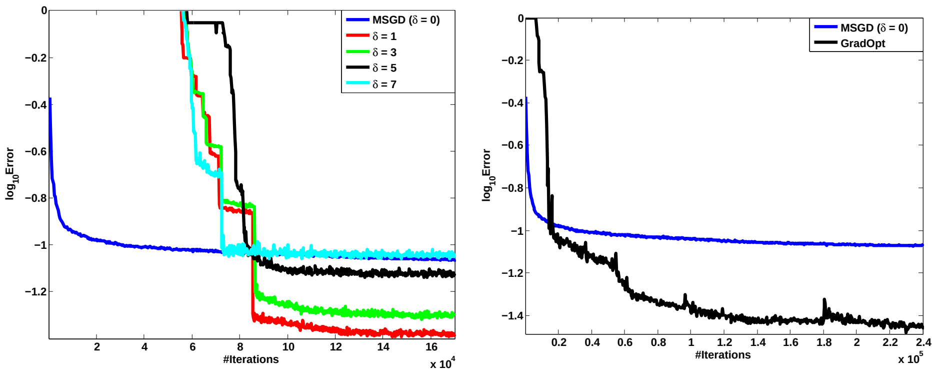
At first, we were interested in exploring the non-convex structure of the above NN learning task, and check whether our definition of σ-nice complies with this structure. We started by running MSGD (Minibatch Stochastic Gradient Descent) on the problem, using a batch size of 100, and a step size rule of $\eta_t = \eta_0(1 + \gamma t)^{-3/4}$, where $\eta_0 = 0.01$, $\gamma = 10^{-4}$. This choice of step size rule was the most effective among a grid of rules that we examined. We have found out that MSGD frequently "stalls" in areas with a relatively high loss, here we relate to points at the end of such run as stall-points.

Later, we examined the objective values along two directions around stall-points. The first direction was the gradient at the stall point, and the second direction was the line connecting the stall-point to $\mathbf{x}^*$, where $\mathbf{x}^*$ is the best weights configuration of the NN that we were able to find. An illustration depicting typical results appears in Figure 4(a). The stall-point appears in red, and $\mathbf{x}^*$ in green; also the axis marked as X is the gradient direction, and one marked Y is the direction between stall-point and $\mathbf{x}^*$. Note that the stall-point is inside a narrow "valley", which prevents MSGD from "perceiving" $\mathbf{x}^*$, and so it seems that MSGD slowly progresses downstream. Note that this resembles the phenomenon depicted in Section 3.2.1.

In Figure 4(b), we present the $\delta = 3$ smoothed version of the same objective that appears in Figure 4(a). We can see that the "valley" has not vanished, but the gradient of the smoothed version leads us slightly towards $\mathbf{x}^*$ and out of the original "valley". Figure 4(c) presents the $\delta = 7$ smoothed version of the objective. We can see that due to the coarse smoothing, the "valley" in which MSGD was stalled, has completely dissolved, and the gradient of this version leads us towards $\mathbf{x}^*$.

7.2. Graduated Optimization of NN

Here we present experiments that demonstrate the effectiveness of GradOpt$_G$ (Algorithm 1) in training the NN mentioned above. First, we wanted to learn if smoothing can help us escape points where MSGD stalls. We used MSGD ($\delta = 0$) to train the NN, and as before we found that its progress slows down, yielding relatively high error. We then took the point that MSGD reached after $5 \cdot 10^4$



(a) (b)

Figure 5. Left: running optimization with fixed smoothing values, starting at the point where MSGD stuck after $5 \cdot 10^4$ iterations. Right: comparison between MSGD and GradOpt$_G$.

iteration and initialized an optimization over the smoothed versions of the loss; this was done using smoothing values of $\{1, 3, 5, 7\}$. In Figure 5(a) we present the results of the above experiment.

As seen in Figure 5(a), small δ's converge slower than large δ's, but produce a much better solution. Furthermore, the initial optimization progresses in leaps, for large δ's the leaps are sharper, and lower δ's demonstrate smaller leaps. We believe that these leaps are associated with the advance of the optimization from one local "valley" to another; Larger values of δ dissolve the "valleys" much easily, but converge to points with higher errors, due to the increase of the bias with smoothing.

In Figure 5(b) we compare our complete graduated optimization algorithm, namely GradOpt$_G$ (Alg. 1) to MSGD. We started with an initial smoothing of $\delta = 7$, which decayed according to GradOpt$_G$. Note that GradOpt$_G$ progresses very fast and yields a much better solution than MSGD.

8. Discussion

We have described a family of non-convex functions which admit efficient optimization via the graduated optimization methodology, and gave the first rigorous analysis of a first-order algorithm in the stochastic setting.

We view it as only a first glimpse of the potential of graduated optimization to provable non-convex optimization, and amongst the interesting questions that remain we find

- Is σ-niceness necessary for convergence of first-order methods to a global optimum? Is there a more lenient property that better captures the power of graduated optimization?
- Can second-order/other methods give rise to better convergence rates / faster algorithms for stochastic or offline σ-nice non-convex optimization?

Acknowledgements

E. Hazan acknowledges support from the National Science Foundation grant IIS-1523815 and a Google research award. S^3 is supported by ICRI-CI and by the European Research Council (TheoryDL project).

References

Allgower, Eugene L and Georg, Kurt. *Numerical continuation methods*, volume 13. Springer-Verlag Berlin, 1990.

Bengio, Yoshua. Learning deep architectures for ai. *Foundations and trends in Machine Learning*, 2(1):1–127, 2009.

Blake, Andrew and Zisserman, Andrew. *Visual reconstruction*, volume 2. MIT press Cambridge, 1987.

Boccuto, A, Discepoli, M, Gerace, I, Pandolfi, R, and Pucci, P. A gnc algorithm for deblurring images with interacting discontinuities. *Proc. VI SIMAI*, pp. 296–310, 2002.

Brox, Thomas and Malik, Jitendra. Large displacement optical flow: descriptor matching in variational motion estimation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 33(3):500–513, 2011.

Chapelle, Olivier and Wu, Mingrui. Gradient descent optimization of smoothed information retrieval metrics. *Information retrieval*, 13(3):216–235, 2010.

Chapelle, Olivier, Chi, Mingmin, and Zien, Alexander. A continuation method for semi-supervised svms. In *Pro-*

ceedings of the 23rd international conference on Machine learning, pp. 185–192. ACM, 2006.

Chaudhuri, Swarat and Solar-Lezama, Armando. Smoothing a program soundly and robustly. In *Computer Aided Verification*, pp. 277–292. Springer, 2011.

Dauphin, Yann N, Pascanu, Razvan, Gulcehre, Caglar, Cho, Kyunghyun, Ganguli, Surya, and Bengio, Yoshua. Identifying and attacking the saddle point problem in high-dimensional non-convex optimization. In *Advances in Neural Information Processing Systems*, pp. 2933–2941, 2014.

Erhan, Dumitru, Manzagol, Pierre-Antoine, Bengio, Yoshua, Bengio, Samy, and Vincent, Pascal. The difficulty of training deep architectures and the effect of unsupervised pre-training. In *International Conference on Artificial Intelligence and Statistics*, pp. 153–160, 2009.

Flaxman, Abraham, Kalai, Adam Tauman, and McMahan, H. Brendan. Online convex optimization in the bandit setting: gradient descent without a gradient. In *SODA*, pp. 385–394, 2005.

Hinton, Geoffrey, Osindero, Simon, and Teh, Yee-Whye. A fast learning algorithm for deep belief nets. *Neural computation*, 18(7):1527–1554, 2006.

Mobahi, Hossein and Fisher III, John W. On the link between gaussian homotopy continuation and convex envelopes. In *Energy Minimization Methods in Computer Vision and Pattern Recognition*, pp. 43–56. Springer, 2015a.

Mobahi, Hossein and Fisher III, John W. A theoretical analysis of optimization by gaussian continuation. 2015b.

Nikolova, Mila, Ng, Michael K, and Tam, Chi-Pan. Fast nonconvex nonsmooth minimization methods for image restoration and reconstruction. *IEEE Transactions on Image Processing*, 19(12):3073, 2010.

Rakhlin, Alexander, Shamir, Ohad, and Sridharan, Karthik. Making gradient descent optimal for strongly convex stochastic optimization. *arXiv preprint arXiv:1109.5647*, 2011.

Terzopoulos, Demetri. The computation of visible-surface representations. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 10(4):417–438, 1988.

Wu, Zhijun. The effective energy transformation scheme as a special continuation approach to global optimization with application to molecular conformation. *SIAM Journal on Optimization*, 6(3):748–768, 1996.

Yuille, AL. Energy functions for early vision and analog networks. *Biological Cybernetics*, 61(2):115–123, 1989.

Yuille, Alan L, Geiger, Davi, and Bülthoff, H. Stereo integration, mean field theory and psychophysics. In *Computer Vision ECCV 90*, pp. 71–82. Springer, 1990.

Zaslavskiy, Mikhail, Bach, Francis, and Vert, J-P. A path following algorithm for the graph matching problem. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 31(12):2227–2242, 2009.