New Computational Guarantees for Solving Convex Optimization Problems with First Order Methods, via a Function Growth Condition Measure

Robert M. Freund* Haihao Lu†

Revised March, 2017

Abstract

Motivated by recent work of Renegar [22], we present new computational methods and associated computational guarantees for solving convex optimization problems using first-order methods. Our problem of interest is the general convex optimization problem $f^* = \min_{x \in Q} f(x)$, where we presume knowledge of a strict lower bound $f_{\text{slb}} < f^*$. [Indeed, f_{slb} is naturally known when optimizing many loss functions in statistics and machine learning (least-squares, logistic loss, exponential loss, total variation loss, etc.) as well as in Renegar's transformed version of the standard conic optimization problem [22]; in all these cases one has $f_{\text{slb}} = 0 < f^*$.] We introduce a new functional measure called the growth constant G for $f(\cdot)$, that measures how quickly the level sets of $f(\cdot)$ grow relative to the function value, and that plays a fundamental role in the complexity analysis. When $f(\cdot)$ is non-smooth, we present new computational guarantees for the Subgradient Descent Method and for smoothing methods, that can improve existing computational guarantees in several ways, most notably when the initial iterate x^0 is far from the optimal solution set. When $f(\cdot)$ is smooth, we present a scheme for periodically restarting the Accelerated Gradient Method that can also improve existing computational guarantees when x^0 is far from the optimal solution set, and in the presence of added structure we present a scheme using parametrically increased smoothing that further improves the associated computational guarantees.

1 Problem Statement and Overview of Results

1.1 Problem Statement, Strict Lower Bound, and Function Growth Constant

Motivated by recent work of Renegar [22], we present new computational methods and associated computational guarantees for solving convex optimization problems using first-order methods. Our problem of interest is the following optimization problem:

$$P: \quad f^* := \begin{array}{ll} \text{minimum}_x & f(x) \\ \text{s.t.} & x \in Q \ , \end{array} \tag{1}$$

*MIT Sloan School of Management, 77 Massachusetts Avenue, Cambridge, MA 02139 (mailto: rfreund@mit.edu). This author's research is supported by AFOSR Grant No. FA9550-15-1-0276 and the MIT-Belgium Université Catholique de Louvain Fund.

†MIT Department of Mathematics, 77 Massachusetts Avenue, Cambridge, MA 02139 (mailto: haihao@mit.edu).

where $Q \subseteq \mathbb{R}^n$ is a closed convex set and $f(\cdot) : Q \to \mathbb{R}$ is a convex function. Let the set of optimal solutions of (1) be denoted as Opt $:= \{x \in Q : f(x) = f^*\}$. For $x \in Q$, let $\text{Dist}(x, \text{Opt})$ denote the distance from x to the set of optimal solutions, namely $\text{Dist}(x, \text{Opt}) := \min_y\{\|y - x\| : y \in \text{Opt}\}$.

Strict Lower Bound f_{slb} and Function Growth Constant G. Let f_{slb} be a known and given strict lower bound on the optimal value f^* of (1), namely $f_{\text{slb}} < f^*$. Such a known strict lower bound arises naturally when optimizing many loss functions in statistics and machine learning (least-squares loss, logistic loss, exponential loss, total variation loss, etc.) perhaps with the addition of a regularization term; in all these cases $f_{\text{slb}} = 0 < f^*$. A known strict lower bound also arises in Renegar's transformed version of the standard conic optimization problem [22].

Let $\varepsilon' > 0$ be given. Given the knowledge of the strict lower bound f_{slb}, it is natural to work with the notion of a relative measure of optimality. Let us define an ε'-relative solution of (1) to be a point $\hat{x}$ that satisfies:

$$\hat{x} \in Q \quad \text{and} \quad \frac{f(\hat{x}) - f^*}{f^* - f_{\text{slb}}} \;\leq\; \varepsilon' \;. \tag{2}$$

Note that (2) is a relative error measure, relative to the optimal bound gap $f^* - f_{\text{slb}}$. We focus on an ε'-relative solution rather than on an ε-absolute solution ($f(\hat{x}) \leq f^* + \varepsilon$), as the former seems more natural in the setting where a strict lower bound is part of the problem description. Indeed, consider the context of loss functions $f(\cdot)$ in statistics and machine learning where $f_{\text{slb}} = 0$, in which case an ε'-relative solution $\hat{x}$ corresponds to $\frac{f(\hat{x})}{f^*} \leq (1 + \varepsilon')$, and hence is a multiplicative measure of optimality tolerance.

Let G denote the smallest scalar $\bar{G}$ satisfying:

$$\text{Dist}(x, \text{Opt}) \leq \bar{G} \cdot (f(x) - f_{\text{slb}}) \;\; \text{for all } x \in Q \;. \tag{3}$$

By its definition one sees that G measures how fast the distances from the optimal solution set Opt grow relative to the bound gap $f(x) - f_{\text{slb}}$. Therefore G is a measure of the growth rate of the level sets of $f(\cdot)$. We call G the "growth constant" of the function $f(\cdot)$ for the given strict lower bound f_{slb}. Note that an equivalent definition of G is given by:

$$G = \sup_{x \in Q} \left\{ \frac{\text{Dist}(x, \text{Opt})}{f(x) - f_{\text{slb}}} \right\} \;. \tag{4}$$

Unlike the strict lower bound f_{slb}, we do not assume that G is known, nor do we need any upper bounds on G. Indeed, neither knowledge of G nor the finiteness of G are needed in order to implement the computational methods presented herein; however the finiteness of G is needed for the analysis of the methods to be meaningful.

We will see in Sections 3 and 4 that the knowledge of the fixed strict lower bound f_{slb} and the concept of the function growth constant G lead to different versions of first-order methods with different computational guarantees than the traditional analysis of first-order methods would dictate. Furthermore, these different computational guarantees can dominate the traditional guarantees in many cases but most notably when the initial iterate x^0 is far from the optimal solution. Roughly speaking, for several of the algorithms developed herein our computational guarantees grow like

$\ln(1 + \text{Dist}(x^0, \text{Opt}))$ in contrast to traditional guarantees where the growth is proportional to $\text{Dist}(x^0, \text{Opt})$ and $\text{Dist}(x^0, \text{Opt})^2$ (in the smooth and nonsmooth settings, respectively).

In a departure from typical optimization approaches to lower bounds such as those arising from duality theory wherein one desires as tight a lower bound as possible, herein the lower bound f_{slb} is strict, namely $f_{\text{slb}} < f^*$, and it is fixed, i.e., it is not updated as part of a computational procedure. It is best to think of this lower bound as a *structural* lower bound that is easily connected to known properties of the function $f(\cdot)$. Such a strict lower bound on $f(\cdot)$ arises naturally in the settings of statistics and machine learning in the case of loss functions and/or regularization functions, see for example [11]. Consider when $f(\cdot)$ is the logistic loss function $f(x) = \frac{1}{m}\sum_{i=1}^m \ln\left(1 + e^{-A_i x}\right)$ or the exponential loss function $f(x) = \ln\left(\sum_{i=1}^m e^{-A_i x}\right)$, perhaps with the addition of a regularization term $\lambda\|x\|_p^r$ for some $p \geq 1$, $r \geq 1$, and $\lambda \geq 0$. If the sample data is not strictly separable, which translated herein means that there is no x satisfying $Ax \geq 0$ unless $Ax = 0$, then it follows that $f^* > 0$ and so $f_{\text{slb}} := 0$ is a strict lower bound and is quite natural in this setting. Another example is regularized least-squares regression such as the LASSO and its cousins, wherein $f(\beta) = \frac{1}{2}\|\mathbf{y} - \mathbf{X}\beta\|^2 + \lambda\|\beta\|_p^r$; it follows that $f^* \geq 0$ and one can assert that $f^* > 0 =: f_{\text{slb}}$ under a variety of mild assumptions involving either λ or the data matrix $\mathbf{X}$. Other classes of examples for which $f_{\text{slb}} = 0$ is a strict lower bound on f^* include total variation (TV) loss functions which are used in image de-noising, as well as the broad class of minimum norm problems in general, under mild assumptions. Another class of problems for which there is a natural strict lower bound on f^* is the class of projectively transformed conic convex optimization problems under a particularly clever projective transformation, as developed by Renegar [22]; indeed it was this problem class and the results in [22] that gave rise to the line of research described herein.

We can interpret G as connected to a lower estimator of $f(\cdot)$: rearranging (3), we obtain:

$$f(x) \;\geq\; \bar{f}(x) := f_{\text{slb}} + G^{-1}\text{Dist}(x, \text{Opt}) \quad \text{for all } x \in Q\ . \tag{5}$$

Therefore the convex function $\bar{f}(x) = f_{\text{slb}} + G^{-1}\text{Dist}(x, \text{Opt})$ is a lower estimator of the function $f(\cdot)$ on Q. This interpretation is illustrated in Figure 1. As Figure 1 illustrates, the concept of the growth constant G is somewhat related to the notion of the modulus of weak sharp minima for (1), see Polyak [18] and Burke and Ferris [6]; this relationship is discussed further in Appendix A.1.

A natural question to ask is under what circumstances is the growth constant G finite? Roughly speaking, it holds that G is finite except when the objective function level sets are ill-behaved relative to their recession cone. This is made precise in the following theorem, whose proof is given in Appendix A.2. For $\varepsilon > 0$, let $\text{Opt}_\varepsilon := \{x \in Q :\ f(x) \leq f^* + \varepsilon\}$ denote the ε-optimal level set of $f(\cdot)$ on Q, and let S denote the recession cone of Opt_ε, namely $S := \{d \in \mathbb{R}^n : x + \theta d \in Q \text{ and } f(x + \theta d) \leq f^* + \varepsilon, \text{ for all } x \in \text{Opt}_\varepsilon \text{ and } \theta \geq 0\}$. Note that S is the (common) recession cone of Opt_ε for all $\varepsilon \geq 0$.

Theorem 1.1. *Suppose that for some $\varepsilon > 0$ there exists a bounded set E_ε for which $\text{Opt}_\varepsilon \subset E_\varepsilon + S$ where S is the recession cone of Opt_ε. Then for any given strict lower bound $f_{\text{slb}} < f^*$, the growth constant G is finite.* □

Let us briefly examine special cases of Theorem 1.1. Consider the case when $\text{Opt} = E + T$ where E is a bounded convex set and T is a subspace. Then for any $\varepsilon > 0$ it is easy to show that

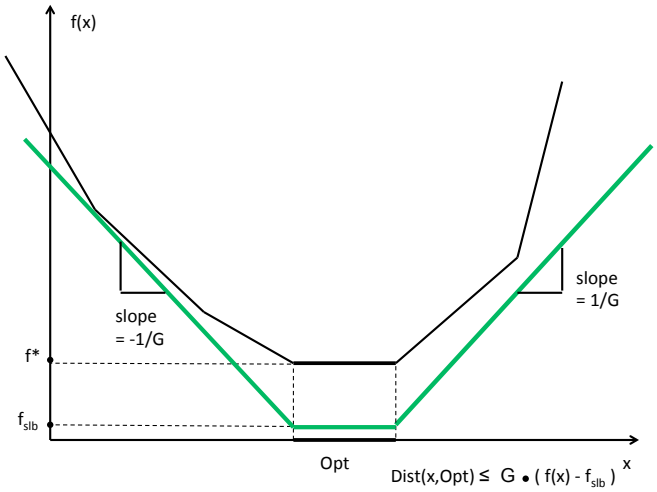

Figure 1: Illustration of G and $f_{\rm slb}$ for a function with multiple optimal solutions.

$\text{Opt}_\varepsilon = E_\varepsilon + T$ for some bounded set E_ε, in which case Theorem 1.1 implies that G is finite. In particular, when Opt itself is a bounded set, then we can set $T = \{0\}$, and so Theorem 1.1 implies that G is finite.

For an example wherein $G = \infty$, consider the function $f(x_1, x_2) := \frac{x_2^2}{x_1}$ on $Q := \{(x_1, x_2) : x_1 \geq 1\}$. It is straightforward to check that the Hessian matrix $\nabla^2 f(x)$ is positive semidefinite on Q and hence $f(\cdot)$ is convex on Q. We have $f^* = 0$ and $\text{Opt} = \{(x_1, 0) : x_1 \geq 1\}$. However, the growth constant $G = \infty$ for any strict lower bound $f_{\rm slb}$, since by letting $(x_1, x_2) = (\beta^2, \beta)$ for any $\beta \geq 1$ we obtain using (4) that

$$G \geq \lim_{\beta \to +\infty} \frac{\text{Dist}\left((\beta^2, \beta), \text{Opt}\right)}{f\left(\beta^2, \beta\right) - f_{\rm slb}} = \lim_{\beta \to +\infty} \frac{\beta}{1 - f_{\rm slb}} = +\infty \ .$$

1.2 Overview of Results

We use the knowledge of the fixed strict lower bound $f_{\rm slb}$ and the concept of the function growth constant G to design and develop computational guarantees for new versions of first-order methods for solving the optimization problem (1). In Section 3 we present such methods when $f(\cdot)$ is non-smooth and Lipschitz continuous with Lipschitz constant M. In Theorem 3.1 we present an iteration complexity of $O\left(M^2 G^2 \left[\ln\left(1 + \frac{f(x^0) - f^*}{f^* - f_{\rm slb}}\right) + \frac{1}{(\varepsilon')^2}\right]\right)$ for a version of Subgradient Descent that simultaneously runs with two step-sizes and occasional re-starting, which strictly improves the standard computational complexity bound for Subgradient Descent when x^0 is a "cold start," i.e., $\text{Dist}(x^0, \text{Opt})$ is large. In the special case when the optimal objective function value f^* is known, Theorem 3.2 shows that the standard step-size rule for Subgradient Descent yields the same result. And when $f(\cdot)$ can be smoothed, we present further improved computational guarantees for a new method (Algorithm 4) that successively smooths and restarts the Accelerated Gradient Method, see Theorem 3.3 herein.

In Section 4 we present computational guarantees for new first-order methods when $f(\cdot)$ is smooth and has Lipschitz gradient with Lipschitz constant L. We present a new first-order method (Algorithm 5) based on periodically restarting the Accelerated Gradient Method, that leads to an iteration complexity of $O\left(G\sqrt{L}\left[\sqrt{f(x^0)-f_{\rm slb}}+\frac{\sqrt{f^*-f_{\rm slb}}}{\sqrt{\varepsilon'}}\right]\right)$ (Theorem 4.1), which in many cases can improve the standard computational complexity bound for the Accelerated Gradient Method, most notably when $f(x^0)$ is far from the optimal value f^* and ε' is small. And when $f(\cdot)$ has appropriate adjoint structure, we use parametric increased smoothing and restarting of the Accelerated Gradient Method to achieve a further improvement in the above computational guarantee (Theorem 4.2).

Algorithm A in Renegar [23] provides an interesting approach to the general convex optimization setting, that bears comparison to the approach and results contained herein – which are also designed for the general convex optimization setting. Both Algorithm A in [23] and the algorithms herein generalize the methodology for conic optimization developed in Renegar [20, 22] to the general convex optimization problems, but they do so in different ways. Herein the generalization is obtained by introducing the new function measure G based on the strict lower bound $f_{\rm slb}$, while in Algorithm A in [23] the original problem is transformed (implicitly or explicitly) to a conic optimization problem in a slightly lifted space. The resulting algorithms appear to be very different, and have different computational requirements and convergence bounds – Algorithm A in [23] requires a 1-dimensional root finding procedure each iteration, whereas Algorithm 3 herein requires orthogonal projection onto the feasible region. (And indeed it is rather remarkable that Algorithm A of [23] does not require such projection.) Algorithm A does not need a Lipschitz constant; however in the case of a smooth objective function Algorithm A cannot take advantage of such smoothness, unlike Algorithm 5 (and also Algorithm 4) herein.

The paper is organized as follows. Section 2 contains a brief review of the Subgradient Descent and an Accelerated Gradient Method. Section 3 contains first-order methods and computational guarantees when $f(\cdot)$ is non-smooth. Section 4 contains first-order methods and computational guarantees when $f(\cdot)$ is smooth.

Notation. Unless otherwise specified, the norm is the Euclidean (inner product) norm $\|x\| := \sqrt{x^T x}$. We occasionally refer to the ℓ_p norm of a vector v, which is denoted by $\|v\|_p$. For $Q \subset \mathbb{R}^n$, let $\Pi_Q(\cdot)$ denote the Euclidean projection operator onto Q, namely $\Pi_Q(x) := \arg\min_{y\in Q} \|y - x\|$. We define $\mathrm{Dist}(x, S) := \min_y\{\|x - y\| : y \in S\}$. The set of optimal solutions of (1) is denoted by $\mathrm{Opt} := \{x \in Q : f(x) = f^*\}$.

2 Review of Subgradient Descent and an Accelerated Gradient Method

We briefly review the Subgradient Descent Method and an Accelerated Gradient Method (as analyzed in Tseng [27]) for solving the convex optimization problem (1).

2.1 Subgradient Descent

Recall that g is a subgradient of $f(\cdot)$ at x if the following subgradient inequality holds:

$$f(y) \geq f(x) + g^T(y - x) \quad \text{for all } y \in Q \ .$$

Let $\partial f(x)$ denote the set of subgradients of $f(\cdot)$ at x. Here we assume that $f(\cdot)$ is Lipschitz continuous on a relatively open set $\hat{Q}$ containing Q, namely, there is a scalar M for which

$$|f(y) - f(x)| \leq M\|y - x\| \quad \text{for all } x, y \in \hat{Q} \ . \tag{6}$$

It follows from (6) that for all $x \in Q$ and $g \in \partial f(x)$ it holds that $\|g\| \leq M$.

Algorithm 1 presents the standard subgradient scheme. In this method x^k is the iterate at iteration k, the best objective value among the first k iterates is f_b^k, and the best iterate among the first k iterates is x_b^k.

Algorithm 1 Subgradient Method for Non-Smooth Optimization

Initialize. Initialize with $x^0 \in Q$, $f_b^0 \leftarrow f(x^0)$, $x_b^0 \leftarrow x^0$. $i \leftarrow 0$.

At iteration i :
1. **Compute Subgradient.** Compute $g_i \in \partial f(x^i)$.
2. **Determine Step-size.** Determine $\alpha_i \geq 0$.
3. **Perform Updates.** Compute $x^{i+1} \leftarrow \Pi_Q(x^i - \alpha_i g_i)$,

$$f_b^{i+1} \leftarrow \min\{f_b^i, f(x^{i+1})\} \ ,$$
$$x_b^{i+1} \leftarrow \underset{x \in \{x_b^i,\ x^{i+1}\}}{\arg\min} \{f(x)\} \ .$$

The following theorem summarize well-known computational guarantees associated with the subgradient descent method.

Theorem 2.1. (Convergence Bounds for Subgradient Descent [19, 14])
(i) Consider the subgradient descent method (Algorithm 1). Then for all $k \geq i \geq 0$, the following inequality holds:

$$f_b^k \ \leq \ f^* \ + \ \frac{\mathrm{Dist}(x^i, \mathrm{Opt})^2 + \sum_{l=i}^k \|g_l\|^2 \alpha_l^2}{2\sum_{l=i}^k \alpha_l} \ \leq \ f^* \ + \ \frac{\mathrm{Dist}(x^i, \mathrm{Opt})^2 + M^2 \sum_{l=i}^k \alpha_l^2}{2\sum_{l=i}^k \alpha_l} \ .$$

(ii) Suppose that f^* is known, and let the step-sizes for Algorithm 1 be $\alpha_i = (f(x^i) - f^*)/\|g_i\|^2$. Then for all $k \geq i \geq 0$, the following inequality holds:

$$f_b^k \ \leq \ f^* \ + \ \frac{M\mathrm{Dist}(x^i, \mathrm{Opt})}{\sqrt{k - i + 1}} \ . \qquad \square$$

Suppose that we seek to bound the number of iterations N of the Subgradient Descent method required to compute an (absolute) ε-optimal solution of (1), which is a point $\hat{x} \in Q$ that satisfies $f(\hat{x}) \leq f^* + \varepsilon$. If $\varepsilon > 0$ is given, and the step-sizes are chosen as $\alpha_i = \varepsilon/\|g_i\|^2$, then it follows from part (i) of Theorem 2.1 that $f_b^N \leq f^* + \varepsilon$ for all

$$N \geq \bar{N} := \frac{M^2 \mathrm{Dist}(x^0, \mathrm{Opt})^2}{\varepsilon^2} - 1 \ . \tag{7}$$

If instead we know (or can bound from above) $\text{Dist}(x^0, \text{Opt})$, and the step-sizes are chosen as $\alpha_i = \text{Dist}(x^0, \text{Opt})/(\sqrt{N+1}\|g_i\|)$ where N satisfies (7), then it also follows from part (i) of Theorem 2.1 that $f_b^N \leq f^* + \varepsilon$. And if f^* is known, then the bound (7) is also sufficient to guarantee $f_b^N \leq f^* + \varepsilon$ if the steps-sizes are chosen as in part (ii) of Theorem 2.1. Furthermore, it follows from [13] that the bound (7) cannot in general be improved in the black-box oracle model of computation with complexity bounds depending only on M, $\text{Dist}(x^0, \text{Opt})$, and ε. In this regard, we note that the dependence on additional parameters, namely the strict lower bound f_{slb} and the function growth constant G, which are used throughout this paper, shows how we can achieve different (and better in many cases) complexity bounds by including additional parameters and appropriately amending algorithms and their analysis.

2.2 Accelerated Gradient Method for Smooth Optimization

Here we assume that $f(\cdot)$ is differentiable on an open set containing Q, and that $\nabla f(\cdot)$ is Lipschitz on Q with scalar L, namely:

$$\|\nabla f(y) - \nabla f(x)\| \leq L\|y - x\| \quad \text{for all } x, y \in Q \ . \tag{8}$$

Algorithm 2 presents a standard Accelerated Gradient Method as in Tseng [27].

Algorithm 2 Accelerated Gradient Method

Initialize. Initialize with $x^0 \in Q$ and $z^0 := x^0$, and $i \leftarrow 0$. Define step-size parameters $\theta_i \in (0, 1]$ recursively by $\theta_0 := 1$ and θ_{i+1} satisfies $\frac{1}{\theta_{i+1}^2} - \frac{1}{\theta_{i+1}} = \frac{1}{\theta_i^2}$.

At iteration i :

1. **Perform Updates.** Define $y^i \leftarrow (1 - \theta_i)x^i + \theta_i z^i$, and compute $\nabla f(y^i)$,
$$z^{i+1} \leftarrow \arg\min_{x \in Q}\{f(y^i) + \nabla f(y^i)^T(x - z^i) + \tfrac{1}{2}\theta_i L(x - z^i)^T(x - z^i)\} \ ,$$
$$x^{i+1} \leftarrow (1 - \theta_i)x^i + \theta_i z^{i+1} \ .$$

For $\delta \geq f^*$ define the level set $S_\delta := \{x \in Q : f(x) \leq \delta\}$. For $x \in Q$, let $\text{Dist}(x, S_\delta)$ denote the distance from x to the level set S_δ, namely $\text{Dist}(x, S_\delta) := \min_y\{\|y - x\| : y \in S_\delta\}$. The following theorem is a computational guarantee for the Accelerated Gradient Method due to Tseng [27].

Theorem 2.2. (Convergence Bound for Accelerated Gradient Method [27]) *Consider the Accelerated Gradient Method (Algorithm 2). Let $\delta \geq f^*$ and $S_\delta := \{x \in Q : f(x) \leq \delta\}$. Then for all $k \geq 0$, the following inequality holds:*

$$f(x^k) - \delta \quad \leq \quad \frac{2L\text{Dist}(x^0, S_\delta)^2}{(k+1)^2} \ . \qquad \square$$

Note that in the case when $\delta = f^*$, then $S_\delta = \text{Opt}$ whereby Theorem 2.2 specializes to the standard result for the Accelerated Gradient Method. We will utilize the more general result in Theorem 2.2 in the context of smoothing of a non-smooth function, in Sections 3.3 and 4 herein.

3 Computational Guarantees when $f(\cdot)$ is Non-Smooth

Let $\varepsilon' > 0$ be given. We aspire to compute an ε'-relative solution of (1), which recall from (2) is a point $\hat{x} \in Q$ satisfying: $\frac{f(\hat{x})-f^*}{f^*-f_{\rm slb}} \leq \varepsilon'$. In this section we present three new computational guarantees for first-order methods applied to computing a ε'-relative solution of problem (1) that are based on the strict lower bound $f_{\rm slb}$ and growth constant G. The first guarantee is for a new algorithm based on Subgradient Descent that runs two different step-sizes simultaneously with occasional re-starts. The second guarantee is for the standard Subgradient Method using a standard step-size rule in the case when the optimal value f^* is known. The third guarantee is for the case when the function $f(\cdot)$ can be smoothed and then solved using an algorithm based on the Accelerated Gradient Method.

3.1 Subgradient Descent using Two Step-Size Rules Running Simultaneously

We consider solving (1) using a version of subgradient descent that simultaneously runs two versions of the Subgradient Descent Method – each with a different step-size rule – with occasional simultaneous re-starts of both versions. The formal description of our method is given in Algorithm 3. In the algorithm, the notation "$(x_{i,j+1}, f_b^{i,j+1}, x_b^{i,j+1}) \leftarrow \text{SDM}(f(\cdot), x_{i,j}, \alpha_{ij}, g_{ij})$" denotes assigning to $x_{i,j+1}$ the next value of the Subgradient Descent Method applied to the optimization problem (1) with objective function $f(\cdot)$ with current point $x_{i,j} \in Q$ using the step-size α_{ij} and the subgradient g_{ij}, along with updates of the best objective function value obtained thus far $f_b^{i,j+1}$ with the corresponding best iterate computed $x_b^{i,j+1}$.

We now walk through the structure of Algorithm 3. The algorithm requires as input the starting point x^0 and the desired relative accuracy value ε' used to define an ε'-relative solution, see (2). The algorithm then defines an absolute constant $\bar{\varepsilon}' := 0.9$. The two values ε' and $\bar{\varepsilon}'$ are then used as aspirational goals for simultaneously running the standard Subgradient Descent Method in search of either an ε'-relative solution of (1) or an $\bar{\varepsilon}'$-relative solution of (1). For notational ease, both ε' and $\bar{\varepsilon}'$ are converted to a slightly different form by defining ε and $\bar{\varepsilon}$. At the start of the $i^{\rm th}$ outer iteration, Algorithm 3 runs the Subgradient Descent Method simultaneously using two different step-size rules (but starting at the same point $x_{i,0} = \bar{x}_{i,0}$), and so generates inner iterations $\{x_{i,j}\}$ and $\{\bar{x}_{i,j}\}$ for $j = 0, 1, \ldots$ based on computed subgradients $\{g_{ij}\}$ and $\{\bar{g}_{ij}\}$ and step-sizes $\{\alpha_{ij}\}$ and $\{\bar{\alpha}_{ij}\}$, respectively. The only structural difference between the two instantiations of Subgradient Descent is that the steps-sizes $\{\alpha_{ij}\}$ use ε in their definition whereas $\{\bar{\alpha}_{ij}\}$ use $\bar{\varepsilon}$ in their definition. The number of inner iterations j that are run in the $i^{\rm th}$ outer iteration is initially set to be $K_i \leftarrow +\infty$. If either $f(x_{i,j})$ or $f(\bar{x}_{i,j})$ makes sufficient progress relative to the starting value $f(x_{i,0})(= f(\bar{x}_{i,0}))$ as determined in the ratio test at the start of Step (2.), then the outer iteration i is concluded and K_i, which counts the number of inner iterations therein, is updated to $K_i \leftarrow j$. Finally, the next outer iteration starting values $x_{i+1,0} = \bar{x}_{i+1,0}$ are re-set to either $x_{i,j}$ or $\bar{x}_{i,j}$, depending on which of $x_{i,j}$ or $\bar{x}_{i,j}$ satisfies the ratio test.

Many of the ideas used in the construction of Algorithm 3 were motivated from similar notions developed in Algorithm 2 of [22] as well as the algorithm "MainAlgo" in [21] (which uses the construct of running two algorithms simultaneously with different parameters).

Regarding counting of iterates $x_{i,j}$, $\bar{x}_{i,j}$ that are computed by Algorithm 3, we will say that the

Algorithm 3 Non-Smooth Method with Two Step-Size Rules Running Simultaneously

Initialize. Initialize with $x^0 \in Q$ and $\varepsilon' > 0$.
Define constants $\bar{\varepsilon}' := 0.9$, $\varepsilon := \frac{\varepsilon'}{1+\varepsilon'}$, $\bar{\varepsilon} := \frac{\bar{\varepsilon}'}{1+\bar{\varepsilon}'}$, $B := 1/\sqrt{e}$, $F := \sqrt{e}$.
Set $x_{1,0} \leftarrow x^0$, $\bar{x}_{1,0} \leftarrow x^0$, $i \leftarrow 1$.

At outer iteration i :

1. **Initialize inner iterations.** $f_b^{i,0} \leftarrow f(x_{i,0})$, $\bar{f}_b^{i,0} \leftarrow f(\bar{x}_{i,0})$
$x_b^{i,0} \leftarrow x_{i,0}$, $\bar{x}_b^{i,0} \leftarrow \bar{x}_{i,0}$
$K_i \leftarrow +\infty$, $j \leftarrow 0$.

2. **Test/update current iterates.** At inner iteration j:

If $\dfrac{f(x_{i,j}) - f_{\text{slb}}}{f(x_{i,0}) - f_{\text{slb}}} > B$ and $\dfrac{f(\bar{x}_{i,j}) - f_{\text{slb}}}{f(x_{i,0}) - f_{\text{slb}}} > B$, then

Compute subgradients. Compute $g_{ij} \in \partial f(x_{i,j})$, $\bar{g}_{ij} \in \partial f(\bar{x}_{i,j})$

Set step-sizes. $\alpha_{ij} \leftarrow \frac{\varepsilon(f(x_{i,0}) - f_{\text{slb}})}{F\|g_{ij}\|^2}$, $\bar{\alpha}_{ij} \leftarrow \frac{\bar{\varepsilon}(f(\bar{x}_{i,0}) - f_{\text{slb}})}{F\|\bar{g}_{ij}\|^2}$

Update: $(x_{i,j+1}, f_b^{i,j+1}, x_b^{i,j+1}) \leftarrow \text{SDM}(f(\cdot), x_{i,j}, \alpha_{ij}, g_{ij})$
$(\bar{x}_{i,j+1}, \bar{f}_b^{i,j+1}, \bar{x}_b^{i,j+1}) \leftarrow \text{SDM}(f(\cdot), \bar{x}_{i,j}, \bar{\alpha}_{ij}, \bar{g}_{ij})$

Else if $\dfrac{f(x_{i,j}) - f_{\text{slb}}}{f(x_{i,0}) - f_{\text{slb}}} \leq B$, then:

$K_i \leftarrow j$, $x_{i+1,0} \leftarrow x_{i,j}$, $\bar{x}_{i+1,0} \leftarrow x_{i,j}$, $i \leftarrow i+1$, and Goto Step 1.

Else $\dfrac{f(\bar{x}_{i,j}) - f_{\text{slb}}}{f(\bar{x}_{i,0}) - f_{\text{slb}}} \leq B$, and:

$K_i \leftarrow j$, $x_{i+1,0} \leftarrow \bar{x}_{i,j}$, $\bar{x}_{i+1,0} \leftarrow \bar{x}_{i,j}$, $i \leftarrow i+1$, and Goto Step 1.

---

algorithm has computed an iterate whenever it computes a subgradient and then calls $\text{SDM}(\cdot,\cdot,\cdot)$. There are therefore two iterates computed at each inner iteration. We have:

Theorem 3.1. (Complexity Bound for Algorithm 3) *Within a total number of iterates computed that does not exceed*

$$18M^2G^2\left(2.7\ln\left(1 + \frac{f(x^0) - f^*}{f^* - f_{\text{slb}}}\right) + \left(\frac{1+\varepsilon'}{\varepsilon'}\right)^2\right) ,$$

Algorithm 3 will compute an iterate $x_{i,j}$ for which

$$\frac{f(x_{i,j}) - f^*}{f^* - f_{\text{slb}}} \leq \varepsilon' . \quad \square$$

Since $f(x^0) \leq f^* + M\text{Dist}(x^0, \text{Opt})$, the computational guarantee in Theorem 3.1 can itself be

bounded by:

$$18M^2G^2\left(2.7\ln\left(1+\frac{M\mathrm{Dist}(x^0,\mathrm{Opt})}{f^*-f_{\mathrm{slb}}}\right)\;+\;\left(\frac{1+\varepsilon'}{\varepsilon'}\right)^2\right)\;, \tag{9}$$

which is qualitatively different from the guarantee of the standard Subgradient Descent Method (Algorithm 1) in (7) in two interesting ways. First, the dependence in (7) on $\mathrm{Dist}(x^0,\mathrm{Opt})$ is quadratic, whereas in (9) it is logarithmic. Second, although both guarantees are linear in the inverse square of the desired relative accuracy ε' (from (2) an ε'-relative solution corresponds to an absolute $\varepsilon'\cdot(f^*-f_{\mathrm{slb}})$ solution of (1)), however x^0 affects this factor multiplicatively through $\mathrm{Dist}(x^0,\mathrm{Opt})^2$ in (7), whereas the factor is independent of x^0 in (9).

Let us also quantitatively compare the computational guarantee of Theorem 3.1 with the standard guarantee for Subgradient Descent given by (7). The standard computational guarantee (7) can be written as:

$$\frac{M^2\mathrm{Dist}(x^0,\mathrm{Opt})^2}{\varepsilon'^2\left(f^*-f_{\mathrm{slb}}\right)^2}\;.$$

Let us presume that ε' is small, whereby $\frac{1+\varepsilon'}{\varepsilon'}\approx\frac{1}{\varepsilon'}$. Then the ratio of the new guarantee (9) from Theorem 3.1 to the standard guarantee (7) is at most

$$\frac{\text{Guarantee of Theorem 3.1}}{\text{Standard Guarantee (7)}}\;\leq\;18(f^*-f_{\mathrm{slb}})^2G^2\left(\frac{2.7(\varepsilon')^2\ln\left(1+\frac{M\mathrm{Dist}(x^0,\mathrm{Opt})}{f^*-f_{\mathrm{slb}}}\right)+1}{\mathrm{Dist}(x^0,\mathrm{Opt})^2}\right)\;. \tag{10}$$

Notice from (10) that for any instance of (1), when $\mathrm{Dist}(x^0,\mathrm{Opt})$ is sufficiently large the right-hand side of (10) can be made arbitrarily small, thereby showing that in these cases the computational guarantee in Theorem 3.1 can be made arbitrarily better than the standard guarantee (7) for Subgradient Descent.

We will prove Theorem 3.1 by first establishing eight propositions. The reader familiar with [22] will notice certain resemblances between aspects of the proof constructs below and the proof of Theorem 3.8 of [22], see also [23]. Throughout, for notational convenience, we will work with three constants B, F, and $\bar{\varepsilon}'$ that must be chosen to satisfy the conditions:

$$B\in(0,1)\;\;,\;\;F>\frac{1}{2B}\;\;,\;\;\text{and}\;\;\bar{\varepsilon}'>0\;,$$

and whose specific values in Algorithm 3 are set to $B=1/\sqrt{e}$, $F=\sqrt{e}$, and $\bar{\varepsilon}'=0.9$, where e is the base of the natural logarithm.

Let $\delta'>0$ play the role of either ε' or $\bar{\varepsilon}'$, and also define $\delta:=\frac{\delta'}{1+\delta'}$ (analogous to the definitions of ε and $\bar{\varepsilon}$).

The first two propositions below apply to the generic setting of the Subgradient Descent Method.

Proposition 3.1. *Let $\delta\in(0,1)$ be given, and suppose we run the Subgradient Descent Method (Algorithm 1) with starting iterate $\hat{x}^0$, using step-sizes:*

$$\alpha_j:=\frac{\delta(f(\hat{x}^0)-f_{\mathrm{slb}})}{F\|g_j\|^2}$$

for all iterations j. Then for all $j \geq 0$ it holds that

$$f_b^j - f_{\text{slb}} \leq f^* - f_{\text{slb}} + \left[\frac{G^2M^2F}{2\delta(j+1)} + \frac{\delta}{2F}\right](f(\hat{x}^0) - f_{\text{slb}}) \ .$$

Proof: Define $\alpha := \frac{\delta(f(\hat{x}^0)-f_{\text{slb}})}{F}$. Then $\alpha_j = \frac{\alpha}{\|g_j\|^2} \geq \frac{\alpha}{M^2}$. It follows from part (i) of Theorem 2.1 that

$$\begin{aligned} f_b^j - f_{\text{slb}} &\leq f^* - f_{\text{slb}} + \frac{\text{Dist}(\hat{x}^0, \text{Opt})^2}{2\sum_{l=0}^j \alpha_l} + \frac{\sum_{l=0}^j \|g_l\|^2\alpha_l^2}{2\sum_{l=0}^j \alpha_l} \\ &\leq f^* - f_{\text{slb}} + \frac{M^2\text{Dist}(\hat{x}^0, \text{Opt})^2}{2\alpha(j+1)} + \frac{\alpha}{2} \\ &\leq f^* - f_{\text{slb}} + \frac{M^2G^2F(f(\hat{x}^0) - f_{\text{slb}})^2}{2(j+1)\delta(f(\hat{x}^0) - f_{\text{slb}})} + \frac{\delta(f(\hat{x}^0) - f_{\text{slb}})}{2F} \ , \end{aligned}$$

where the second inequality uses the definition of α_l and the inequality $\|g_j\| \leq M$, and the third inequality uses the definitions of α and G. Simplifying the last expression completes the proof. □

Proposition 3.2. *Under the identical set-up as Proposition 3.1, let $\delta' := \delta/(1-\delta)$, and define:*

$$W := \left\lfloor \frac{FM^2G^2}{2\delta^2\left[B - \frac{1}{2F}\right]} \right\rfloor \ .$$

Then either $\frac{f_b^W - f^*}{f^* - f_{\text{slb}}} \leq \delta'$, or $f_b^W - f_{\text{slb}} \leq B(f(\hat{x}^0) - f_{\text{slb}})$, or both.

Proof: Suppose that $\frac{f_b^W - f^*}{f^* - f_{\text{slb}}} > \delta'$. This rearranges to: $\delta' < \frac{f_b^W - f_{\text{slb}}}{f^* - f_{\text{slb}}} - 1$, whereby

$$\frac{f^* - f_{\text{slb}}}{f_b^W - f_{\text{slb}}} < \frac{1}{1+\delta'} = 1 - \delta \ . \tag{11}$$

Invoking Proposition 3.1 we have:

$$\begin{aligned} f_b^W - f_{\text{slb}} &\leq f^* - f_{\text{slb}} + \left[\tfrac{G^2M^2F}{2\delta(W+1)} + \tfrac{\delta}{2F}\right](f(\hat{x}^0) - f_{\text{slb}}) \\ &< f^* - f_{\text{slb}} + \left[\delta\left(B - \tfrac{1}{2F}\right) + \tfrac{\delta}{2F}\right](f(\hat{x}^0) - f_{\text{slb}}) \\ &= f^* - f_{\text{slb}} + \delta B(f(\hat{x}^0) - f_{\text{slb}}) \\ &< (1-\delta)(f_b^W - f_{\text{slb}}) + \delta B(f(\hat{x}^0) - f_{\text{slb}}) \ , \end{aligned}$$

where the second inequality follows since $W + 1 > \frac{FM^2G^2}{2\delta^2[B-1/(2F)]}$, and the last inequality uses (11). Rearranging the final inequality and dividing by δ then yields $f_b^W - f_{\text{slb}} \leq B(f(\hat{x}^0) - f_{\text{slb}})$, which completes the proof. □

In the next two propositions we apply Proposition 3.2 directly to the setting of Algorithm 3.

Proposition 3.3. *Consider outer iteration i of Algorithm 3. Define:*

$$U := \left\lfloor \frac{FM^2G^2}{2\varepsilon^2\left[B - \frac{1}{2F}\right]} \right\rfloor .$$

If $K_i > U$, then $\frac{f_b^{ij} - f^*}{f^* - f_{\rm slb}} \leq \varepsilon'$ for all $j = U, \ldots, K_i$.

Proof: Let us apply Proposition 3.2 with $\delta' := \varepsilon'$, $W := U$, and $\hat{x}^0 := x_{i,0}$. If $K_i > U$, then by definition of K_i it holds that $f_b^{i,U} - f_{\rm slb} > B(f_b^{i,0} - f_{\rm slb})$. Thus, from Proposition 3.2 it holds that $\frac{f_b^{i,U} - f^*}{f^* - f_{\rm slb}} \leq \varepsilon'$. Therefore $\frac{f_b^{ij} - f^*}{f^* - f_{\rm slb}} \leq \varepsilon'$ for all $j = U, \ldots, K_i$, since f_b^{ij} is by definition monotonically nonincreasing in j. □

Proposition 3.4. *Consider outer iteration i of Algorithm 3. Define:*

$$V := \left\lfloor \frac{FM^2G^2}{2\bar{\varepsilon}^2\left[B - \frac{1}{2F}\right]} \right\rfloor .$$

If $K_i > V$, then $\frac{f(x_{i,0}) - f_{\rm slb}}{f^* - f_{\rm slb}} \leq \frac{1+\bar{\varepsilon}'}{B}$.

Proof: Let us similarly apply Proposition 3.2 with $\delta' := \bar{\varepsilon}'$, $W := V$, and $\hat{x}^0 := x_{i,0} = \bar{x}_{i,0}$. Let us suppose $K_i > V$. First, notice that for $0 \leq j < K_i$, it holds that

$$f(\bar{x}_{i,j}) - f_{\rm slb} > B(f(\bar{x}_{i,0}) - f_{\rm slb}) = B(f(x_{i,0}) - f_{\rm slb}) .$$

Therefore the left-hand term above can be replaced by $\bar{f}_b^{ij} - f_{\rm slb}$, and setting $j = V$ we obtain $\bar{f}_b^{i,V} - f_{\rm slb} > B(f(x_{i,0}) - f_{\rm slb})$. Therefore from Proposition 3.2 it holds that $\frac{\bar{f}_b^{i,V} - f^*}{f^* - f_{\rm slb}} \leq \bar{\varepsilon}'$. Combining these inequalities we obtain:

$$B(f(x_{i,0}) - f_{\rm slb}) < \bar{f}_b^{i,V} - f_{\rm slb} = \bar{f}_b^{i,V} - f^* + f^* - f_{\rm slb} \leq \bar{\varepsilon}'(f^* - f_{\rm slb}) + f^* - f_{\rm slb} ,$$

and rearranging yields the result. □

In the next proposition we use the standard notation a^+ for the nonnegative part of a scalar a.

Proposition 3.5. *Let m denote the number of outer iterations i of Algorithm 3 for which*

$$\frac{f(x_{i,0}) - f_{\rm slb}}{f^* - f_{\rm slb}} > \frac{1 + \bar{\varepsilon}'}{B} .$$

Then

$$m \leq \left\lceil \frac{\ln\left(1 + \frac{f(x^0) - f^*}{f^* - f_{\rm slb}}\right) - \ln\left(\frac{1+\bar{\varepsilon}'}{B}\right)}{\ln(1/B)} \right\rceil^+ .$$

Proof: If $m = 0$ then the result holds trivially, so let us suppose that $m \geq 1$. It then follows using induction on $f(x_{i+1,0}) - f_{\rm slb} \leq B(f(x_{i,0}) - f_{\rm slb})$ that

$$\frac{1 + \bar{\varepsilon}'}{B} < \frac{f(x_{m,0}) - f_{\rm slb}}{f^* - f_{\rm slb}} \leq \frac{B^{m-1}(f(x_{1,0}) - f_{\rm slb})}{f^* - f_{\rm slb}} ,$$

and taking logarithms yields

$$m-1 \;<\; \frac{\ln\left(\frac{f(x_{1,0})-f_{\rm slb}}{f^*-f_{\rm slb}}\right)-\ln\left(\frac{1+\bar\varepsilon'}{B}\right)}{\ln(1/B)} \;=\; \frac{\ln\left(1+\frac{f(x^0)-f^*}{f^*-f_{\rm slb}}\right)-\ln\left(\frac{1+\bar\varepsilon'}{B}\right)}{\ln(1/B)}\;,$$

from which the result follows. □

In the following proposition, as well as others later on, we use the standard notational convention that $\sum_{i=1}^{n}\cdot := 0$ for $n \le 0$.

Proposition 3.6. *Let V and m be as defined in Propositions 3.4 and 3.5. Then $x_{m+1,0}$ exists, and and let T_m denote the total number of iterates computed prior to and including $x_{m+1,0}$. It holds that:*

$$\frac{f(x_{m+1,0})-f_{\rm slb}}{f^*-f_{\rm slb}} \le \frac{1+\bar\varepsilon'}{B}\;,$$

and furthermore $T_m \le 2mV$.

Proof: If $m = 0$ then the results holds trivially from the definition of m.

Next suppose that $m \ge 1$, and consider any outer iteration $i \le m$. Then since

$$\frac{f(x_{i,0})-f_{\rm slb}}{f^*-f_{\rm slb}} \;>\; \frac{1+\bar\varepsilon'}{B}\;,$$

it follows from Proposition 3.4 that $K_i \le V$. This also implies that $x_{m+1,0}$ exists and therefore must satisfy

$$\frac{f(x_{m+1,0})-f_{\rm slb}}{f^*-f_{\rm slb}} \le \frac{1+\bar\varepsilon'}{B}\;.$$

Finally, since $T_m = 2\sum_{i=1}^{m} K_i$, it therefore follows that $T_m \le 2mV$. □

Proposition 3.7. *Let p denote the number of outer iterations i for which K_i is finite. Then*

$$p \le m + \left\lfloor \frac{\ln\left(\frac{1+\bar\varepsilon'}{B}\right)}{\ln(1/B)} \right\rfloor\;,$$

where m is as defined in Proposition 3.5.

Proof: It follows from Proposition 3.6 that $p \ge m$. Therefore $f^* - f_{\rm slb} \le f(x_{p,K_p}) - f_{\rm slb} = f(x_{p+1,0}) - f_{\rm slb} \le B^{p-m}(f(x_{m+1,0}) - f_{\rm slb}) \le B^{p-m}\left(\frac{1+\bar\varepsilon'}{B}\right)(f^* - f_{\rm slb})$, where we have used the properties of $x_{m+1,0}$ in Proposition 3.6. Taking logarithms yields

$$p - m \;\le\; \frac{\ln\left(\frac{1+\bar\varepsilon'}{B}\right)}{\ln(1/B)}\;,$$

from which the result follows. □

Proposition 3.8. *Let U, m, and p be as defined in Propositions 3.3, 3.5, and 3.7. Within a total number of computed iterates after $x_{m+1,0}$ that does not exceed $2(p-m+1)U$, Algorithm 3 will compute an iterate $x_{\hat i,\hat j}$ for which*

$$\frac{f(x_{\hat i,\hat j})-f^*}{f^*-f_{\rm slb}} \le \varepsilon'\;.$$

Proof: Let $\hat{i}$ denote the index of the first outer iteration $i \in \{m+1, \ldots, p+1\}$ for which $K_i > U$. Notice that since $K_{p+1} = +\infty$ it must hold that $\hat{i} \leq p+1$. It follows from Proposition 3.3 that $\frac{f_b^{\hat{i},U} - f^*}{f^* - f_{\rm slb}} \leq \varepsilon'$ and hence for some $\hat{j} \leq U$ it holds that $\frac{f(x_{\hat{i},\hat{j}}) - f^*}{f^* - f_{\rm slb}} \leq \varepsilon'$. Let us now count the number of iterates computed after $x_{m+1,0}$ and prior to and including $x_{\hat{i},\hat{j}}$. This number is bounded above by:

$$2\left(\sum_{i=m+1}^{\hat{i}-1} K_i + U\right) \leq 2\Big((\hat{i}-m-1)U + U\Big) = 2(\hat{i}-m)U \leq 2(p-m+1)U \ ,$$

where the first inequality follows since $K_i \leq U$ for $i < \hat{i}$, and the last inequality uses $\hat{i} \leq p+1$. □

We now use these propositions to prove Theorem 3.1.

Proof of Theorem 3.1: Utilizing the definitions of U, V, m, p, and $x_{\hat{i},\hat{j}}$ in Propositions 3.3, 3.4, 3.5, 3.7, and 3.8, it follows from Propositions 3.6 and 3.8 that the total number of iterates computed prior to and including $x_{\hat{i},\hat{j}}$ is at most $2[mV + (p-m+1)U]$. Substituting the values of U and V and using the bounds on m and p in Propositions 3.5 and 3.7 yields:

$$\begin{aligned}
& 2\left\lceil \frac{\ln\left(1+\frac{f(x^0)-f^*}{f^*-f_{\rm slb}}\right) - \ln\left(\frac{1+\bar{\varepsilon}'}{B}\right)}{\ln(1/B)} \right\rceil^+ \left\lfloor \frac{FM^2G^2}{2\bar{\varepsilon}^2\left[B-\frac{1}{2F}\right]} \right\rfloor + 2\left\lfloor 1 + \frac{\ln\left(\frac{1+\bar{\varepsilon}'}{B}\right)}{\ln(1/B)} \right\rfloor \left\lfloor \frac{FM^2G^2}{2\varepsilon^2\left[B-\frac{1}{2F}\right]} \right\rfloor \\
\leq \ & 2\left\lceil \frac{\ln\left(1+\frac{f(x^0)-f^*}{f^*-f_{\rm slb}}\right)}{\ln(1/B)} \right\rceil \left\lceil \frac{FM^2G^2}{2\bar{\varepsilon}^2\left[B-\frac{1}{2F}\right]} \right\rceil + 2\left[2 + \frac{\ln(1+\bar{\varepsilon}')}{\ln(1/B)}\right] \left\lceil \frac{FM^2G^2}{2\varepsilon^2\left[B-\frac{1}{2F}\right]} \right\rceil \\
\leq \ & M^2G^2\left(48.5\ln\left(1+\frac{f(x^0)-f^*}{f^*-f_{\rm slb}}\right) \ + \ 18\left(\frac{1+\varepsilon'}{\varepsilon'}\right)^2\right) \ ,
\end{aligned}$$

where the second inequality follows from substituting in the values $B = 1/\sqrt{e}$, $F = \sqrt{e}$, and $\bar{\varepsilon}' = 0.9$, and rounding terms upward. This last expression then is rounded upward to yield the desired iteration bound. □

3.2 Subgradient Descent when f^* is known

In the special case when f^* is known, we can obtain a computational guarantee that is of the same order as that of Theorem 3.1 by directly using the standard Subgradient Descent Method (Algorithm 1) with the (standard) step-size rule $\alpha_i := (f(x^i) - f^*)/\|g_i\|^2$. This is shown in the following theorem.

Theorem 3.2. (Complexity Bound for standard Subgradient Descent when f^* is known) *Let the step-sizes for the Subgradient Descent Method (Algorithm 1) applied to solve problem* (1) *be chosen as:*

$$\alpha_i := \frac{f(x^i) - f^*}{\|g_i\|^2} \ ,$$

and suppose that $N \geq 0$ *and satisfies*

$$N \;\geq\; 2M^2G^2\left(1 + 2.9\ln\left(\frac{f(x^0)-f^*}{f^*-f_{\rm slb}}\right) + 2.9\ln\left(\frac{1}{\varepsilon'}\right) + 6.8\left(\frac{1}{\varepsilon'}\right) + 2\left(\frac{1}{\varepsilon'}\right)^2\right) \;.$$

Then:

$$\frac{f(x_b^N)-f^*}{f^*-f_{\rm slb}} \;\leq\; \varepsilon' \;. \qquad \square \tag{12}$$

The computational guarantee above is an almost-exact generalization of Theorem 3.7 of Renegar [22], which therein pertains to a specific transformed conic optimization problem. The proof of this theorem follows the logic for the proof of Theorem 3.7 of [22] in many respects as well.

Notice that up to an absolute constant, the computational guarantee of Theorem 3.2 is essentially the same as that of Theorem 3.1 in the worst case.

Proof of Theorem 3.2: We will presume that $\frac{f(x^0)-f^*}{f^*-f_{\rm slb}} > \varepsilon'$, since otherwise (12) is satisfied trivially for all $N \geq 0$. Let $B \in (0,1)$ be a given fractional quantity. Define $K_0 := 0$, and for all i such that $f_b^{K_i} - f^* > 0$ define K_{i+1} inductively as the smallest iteration index of Subgradient Descent for which $f_b^{K_{i+1}} - f^* \leq B(f_b^{K_i} - f^*)$. Notice that so long as $f_b^{K_i} - f^* > 0$ it follows using part (ii) of Theorem 2.1 that K_{i+1} exists (i.e., is finite). Let i' be the smallest sub-index i for which $\frac{f_b^{K_{i'}}-f^*}{f^*-f_{\rm slb}} \leq \varepsilon'$. It follows from the initial presumption above that $i' \geq 1$, and it holds for any $i \geq 0$ satisfying $i < i'$ that $\varepsilon'(f^* - f_{\rm slb}) < f(x^{K_i}) - f^* \leq B^i(f(x^{K_0}) - f^*) = B^i(f(x^0) - f^*)$, from which it follows that i, and hence also i', is finite. Furthermore, it holds for any $i \geq 0$ satisfying $i < i'$ that:

$$\varepsilon'(f^* - f_{\rm slb}) < f(x^{K_{i'-1}}) - f^* \leq B^{i'-1-i}(f(x^{K_i}) - f^*) \;, \tag{13}$$

since $x^{K_i} = x_b^{K_i}$ by the definition of K_i. Using $i = 0$ in (13) and taking logarithms yields:

$$i' < 1 + \frac{\ln\left(\frac{1}{\varepsilon'}\right) + \ln\left(\frac{f(x^0)-f^*}{f^*-f_{\rm slb}}\right)}{\ln(B^{-1})} \;. \tag{14}$$

If K_{i+1} exists (i.e., is finite), then it follows from part (ii) of Theorem 2.1 that:

$$f(x_b^{K_{i+1}-1}) - f^* \leq \frac{M\mathrm{Dist}(x^{K_i}, \mathrm{Opt})}{\sqrt{K_{i+1} - 1 - K_i + 1}} \;.$$

This last inequality can be rearranged to yield:

$$K_{i+1} - K_i \leq \frac{M^2\mathrm{Dist}(x^{K_i}, \mathrm{Opt})^2}{\left(f(x_b^{K_{i+1}-1}) - f^*\right)^2} < \frac{B^{-2}M^2G^2\left(f(x^{K_i}) - f_{\rm slb}\right)^2}{\left(f(x^{K_i}) - f^*\right)^2} = B^{-2}M^2G^2\left(1 + \frac{f^* - f_{\rm slb}}{f(x^{K_i}) - f^*}\right)^2 \tag{15}$$

where the second inequality uses the definition of the growth constant G as well as the fact that $f(x_b^{K_{i+1}-1}) - f^* > B(f(x^{K_i}) - f^*)$. Now putting all of this together we obtain:

$$K_{i'} = \sum_{i=0}^{i'-1}(K_{i+1} - K_i)$$

$$
\begin{aligned}
&\leq B^{-2}M^2G^2 \sum_{i=0}^{i'-1} \left(1 + \frac{f^* - f_{\text{slb}}}{f(x^{K_i}) - f^*}\right)^2 \\
&\leq B^{-2}M^2G^2 \sum_{i=0}^{i'-1} \left(1 + \frac{1}{\varepsilon'} B^{i'-1-i}\right)^2 \\
&= B^{-2}M^2G^2 \sum_{j=0}^{i'-1} \left(1 + \left(\frac{2}{\varepsilon'}\right) B^j + \left(\frac{1}{\varepsilon'}\right)^2 (B^2)^j\right) \\
&\leq B^{-2}M^2G^2 \left(i' + \left(\frac{2}{\varepsilon'}\right) \frac{1}{1-B} + \left(\frac{1}{\varepsilon'}\right)^2 \frac{1}{1-B^2}\right) \\
&\leq B^{-2}M^2G^2 \left(1 + \frac{\ln\left(\frac{1}{\varepsilon'}\right) + \ln\left(\frac{f(x^0) - f^*}{f^* - f_{\text{slb}}}\right)}{\ln(B^{-1})} + \left(\frac{2}{\varepsilon'}\right) \frac{1}{1-B} + \left(\frac{1}{\varepsilon'}\right)^2 \frac{1}{1-B^2}\right),
\end{aligned}
$$

where the first inequality is from (15), the second inequality uses (13), the third inequality replaces the two finite geometric series with corresponding infinite series, and the fourth inequality uses (14). Finally, using the value of $B = 1/\sqrt{2}$ and substituting into the above yields the result. □

We remark that one obtains the precise constants of Theorem 3.7 of [22] by using $B = 1/2$. Choosing B to optimize the absolute constant of the $(1/\varepsilon')^2$ term yields $B = 1/\sqrt{2}$ and the absolute constants as presented in the statement of the threorem. Choosing B to optimize the absolute constant of the $\ln\left(\frac{f(x^0)-f^*}{f^*-f_{\text{slb}}}\right)$ term would yield $B = 1/\sqrt{e}$ with the coefficient of 2 in the $\ln(\cdot)$ terms.

To conclude this subsection, consider the case when f^* is known and G can be upper-bounded by a constant for any f_{slb} (as is the case when $f(\cdot)$ is piecewise-linear or, more generally, when $f(\cdot)$ has weak sharp minima (30)). Then given an absolute tolerance ε, we can set $\varepsilon' = 1$ and $f_{\text{slb}} = f^* - \varepsilon$, whereby Theorem 3.2 implies linear convergence of Algorithm 1 in term of the absolute tolerance ε. However, this is not as favorable a result as that in Yang and Lin [28], which obtains linear convergence without requiring that f^* is known. Also, this result can be considered a slight variation of Gilpin, Peña and Soheli [10], which assumes $f(\cdot)$ is piecewise-linear but does not require that f^* is known.

3.3 Non-Smooth Optimization using a New Smooth Approximations Method

As first proposed by Nesterov [15], there are many practical settings wherein one can approximate the non-smooth convex function $f(\cdot)$ by a smooth convex function $f_\mu(\cdot)$, where the sense of the approximation depends on the parameter μ. If the smooth approximation $f_\mu(\cdot)$ is computationally easy to work with, one can then use the Accelerated Gradient Method (Algorithm 2) to approximately optimize $f_\mu(\cdot)$ (thereby also approximately optimizing $f(\cdot)$) on the feasible set Q. There are a variety of techniques that can be used to construct a parametric family of smooth functions $f_\mu(\cdot)$ depending on the known structure of $f(\cdot)$ and Q, see [15] as well as [16] and Beck and Teboulle [1]

among others. For our purposes herein, we will suppose that there is a smoothing technique with the following two properties:

(i) there is a known constant $\bar{D} > 0$ such that for any given $\mu > 0$ we can construct a smooth convex function $f_\mu(\cdot) : Q \to \mathbb{R}$ which is not far from $f(\cdot)$, namely:

$$f(x) - \bar{D}\mu \;\leq\; f_\mu(x) \;\leq\; f(x) \quad \text{for all } x \in Q \text{ , and} \tag{16}$$

(ii) $f_\mu(\cdot)$ has Lipschitz continuous gradient on Q with Lipschitz constant L_μ satisfying

$$L_\mu \leq \bar{A}/\mu \tag{17}$$

for some known positive constant $\bar{A}$.

These properties can be used to design an implementation of the Accelerated Gradient Method (Algorithm 2) applied to $f_\mu(\cdot)$, that can be used to compute an absolute ε-optimal solution of the original optimization problem (1). The scheme developed in [15] in conjunction with the Accelerated Gradient Method (Algorithm 2) yields an iteration complexity bound of

$$\check{N} := \left\lceil \frac{\sqrt{8\bar{A}\bar{D}}\text{Dist}(x^0, \text{Opt})}{\varepsilon} - 1 \right\rceil \tag{18}$$

to obtain an (absolute) ε-optimal solution of (1) for a suitably designed version of the basic method.

Herein we develop a variant of the basic smoothing method to solve the optimization problem (1) that yields a new computational guarantee that can improve on (18) in many cases. Algorithm 4 presents parametric smoothing and restarting method for computing an ε'-relative solution of the optimization problem (1) for the non-smooth objective function $f(\cdot)$ based on successive smooth approximations and re-starting of the Accelerated Gradient Method (Algorithm 2). In the description of Algorithm 4 the general notation "$x_{i,j} \leftarrow \text{AGM}(f_\mu(\cdot),\ x_{i,0},\ j)$" denotes assigning to $x_{i,j}$ the j^{th} iterate of the Accelerated Gradient Method applied to the optimization problem (1) with objective function $f_\mu(\cdot)$ using the initial point $x_{i,0} \in Q$.

At the i^{th} outer iteration of Algorithm 4, the algorithm sets two different smoothing parameters in Step (1.), namely μ_i^1 and μ_i^2, where μ_i^2 differs from μ_i^1 by the relative accuracy input value ε'. The algorithm then runs the Accelerated Gradient Method with starting point $x_{i,0}$ simultaneously on the two smoothed functions $f_{\mu_i^1}(\cdot)$ and $f_{\mu_i^2}(\cdot)$, using the double indexing notation of $x_{i,j}$ and $y_{i,j}$ to denote iteration j of the Accelerated Gradient Method initialized at the point $x_{i,0}$ for optimizing $f_{\mu_i^1}(\cdot)$ and $f_{\mu_i^2}(\cdot)$ on Q, respectively. Notice that the smoothing parameters μ_i^1 and μ_i^2 decrease over the course of the outer iterations, as it makes more sense to set these values higher at first and then decrease them as the solution is approached. The outer iteration i runs until the ratio test in Step (3a.) fails, at which point the current point $x_{i,j}$ becomes the starting point of the next outer iteration, namely $x_{i+1,0} \leftarrow x_{i,j}$. The counter K_i records the number of inner iterations j of outer iteration i. Regarding counting of iterates computed in Algorithm 4, we will say that the algorithm has computed an iterate whenever it calls $\text{AGM}(\cdot,\cdot,\cdot)$. There are therefore two computed iterates at each inner iteration.

Algorithm 4 Parametric Smoothing/Restarting Method using $f_\mu(\cdot)$

Initialize. Initialize with $x^0 \in Q$ and $\varepsilon' > 0$.
Define $B := \frac{1}{2}$, $t := \frac{1}{8}$.
Set $x_{1,0} \leftarrow x^0$, $i \leftarrow 1$.

At outer iteration i :

1. **Set smoothing parameters.** $\mu_i^1 \leftarrow \dfrac{t \cdot (f(x_{i,0}) - f_{\rm slb})}{\bar{D}}$, $\mu_i^2 \leftarrow \dfrac{t\varepsilon' \cdot (f(x_{i,0}) - f_{\rm slb})}{\bar{D}}$.
2. **Initialize inner iteration.** $K_i \leftarrow +\infty$, $j \leftarrow 0$
3. **Run inner iterations.** At inner iteration j:

(3a.) If $\dfrac{f(x_{i,j}) - f_{\rm slb}}{f(x_{i,0}) - f_{\rm slb}} > B$, then

$x_{i,j+1} \leftarrow {\rm AGM}(f_{\mu_i^1}(\cdot),\ x_{i,0},\ j+1)$,

$y_{i,j+1} \leftarrow {\rm AGM}(f_{\mu_i^2}(\cdot),\ x_{i,0},\ j+1)$,

$j \leftarrow j+1$, and Goto (3a.)

Else $K_i \leftarrow j$, $x_{i+1,0} \leftarrow x_{i,j}$, $i \leftarrow i+1$, and Goto Step 1.

Restarting for accelerated gradient methods for strongly convex functions has been studied in [17] and [26]. To the best of our knowledge, restarting of accelerated methods in the absence of strong convexity was first used in Renegar [20], and Algorithm 4 exploits this and other ideas from [20] and [21] as well. We have the following computational guarantee associated with Algorithm 4.

Theorem 3.3. (Complexity Bound for Parametric Smoothing/Restarting Method (Algorithm 4) for Non-smooth Optimization) *Suppose that $f_\mu(\cdot)$ satisfies the smoothing conditions* (16) *and* (17)*. Within a total number of computed iterates that does not exceed*

$$23G\sqrt{\bar{A}\bar{D}}\left(1 + 1.42\ln\left(1 + \frac{f(x^0) - f^*}{f^* - f_{\rm slb}}\right) + 2\left(\frac{1}{\varepsilon'}\right)\right) ,$$

Algorithm 4 will compute an iterate $y_{i,j}$ for which

$$\frac{f(y_{i,j}) - f^*}{f^* - f_{\rm slb}} \le \varepsilon' . \quad \square$$

Similar to Theorem 3.1, the dependence in Theorem 3.3 on the quality of the initial iterate is logarithmic in the initial optimality gap $f(x^0) - f^*$. Also, the factor involving $1/\varepsilon'$ in Theorem 3.3 is independent of the quality of the initial iterate, unlike that of the standard bound for the smoothing method given in (18). We will prove Theorem 3.3 by first establishing several propositions. Throughout, for notational convenience, we will work with two constants B and t that must be chosen to satisfy

$$B > 0\ ,\ t > 0\ ,\ B - B^2 \ge 2t\ ,\ \text{and}\ B \ge 4t\ ,$$

and whose specific values are set to $B = 1/2$ and $t = 1/8$ in Algorithm 4.

The following proposition applies to the generic setting of the Accelerated Gradient Method applied to the smoothed function $f_\mu(\cdot)$. Recall that L_μ denotes the Lipschitz constant of the gradient of $f_\mu(\cdot)$ on Q.

Proposition 3.9. *Given the smoothing parameter $\mu > 0$ and a given constant $\beta > 0$, define $Y := \left\lceil G\sqrt{2\beta} - 1 \right\rceil$. Let $x_k \leftarrow \mathrm{AGM}(f_\mu(\cdot), \hat{x}^0, k)$ denote the k^{th} iterate of the Accelerated Gradient Method applied to the function $f_\mu(\cdot)$ with starting point $\hat{x}^0$. For $k \geq Y$ it holds that:*

$$f(x_k) - f^* \leq \frac{L_\mu}{\beta}(f(\hat{x}^0) - f_{\mathrm{slb}})^2 + \mu\bar{D} \ . \tag{19}$$

Proof: Note that for any $x \in \mathrm{Opt}$ it holds that $f_\mu(x) \leq f^*$, whereby $\mathrm{Opt} \subset S := \{x \in Q : f_\mu(x) \leq f^*\}$. It then follows from Theorem 2.2 applied to the function $f_\mu(\cdot)$ and using $\delta = f^*$ that for any $k \geq Y$ we have:

$$\begin{aligned} f_\mu(x_k) - f^* &\leq \tfrac{2L_\mu}{(Y+1)^2}\mathrm{Dist}(\hat{x}^0, S)^2 \\ &\leq \tfrac{2L_\mu}{(Y+1)^2}\mathrm{Dist}(\hat{x}^0, \mathrm{Opt})^2 \leq \tfrac{2L_\mu}{(Y+1)^2}G^2\left(f(\hat{x}^0) - f_{\mathrm{slb}}\right)^2 \leq \tfrac{L_\mu}{\beta}\left(f(\hat{x}^0) - f_{\mathrm{slb}}\right)^2 \ , \end{aligned}$$

where the second inequality uses the fact that $\mathrm{Opt} \subset S$, the third inequality uses the definition of G, and the last inequality uses the value of Y.

Note from (16) that $f(x) \leq f_\mu(x) + \mu\bar{D}$, whereby:

$$f(x_k) - f^* \leq f_\mu(x_k) - f^* + \mu\bar{D} \leq \frac{L_\mu}{\beta}\left(f(\hat{x}^0) - f_{\mathrm{slb}}\right)^2 + \mu\bar{D} \ . \quad \square$$

We now apply Proposition 3.9 to the setting of the Parametric Smoothing/Restarting Method (Algorithm 4).

Proposition 3.10. *Let i be the index of an outer iteration of Algorithm 4. Define $T := \left\lceil \frac{G\sqrt{2\bar{A}\bar{D}}}{t} - 1 \right\rceil$. If $k \geq T$ and $x_{i,k}$ exists, then it holds that:*

$$f(x_{i,k}) - f^* \leq 2t(f(x_{i,0}) - f_{\mathrm{slb}}) \ .$$

Proof: The proof follows by applying Proposition 3.9 with $\mu = \mu_i^1 = \frac{t\cdot(f(x_{i,0}) - f_{\mathrm{slb}})}{\bar{D}}$, $\beta = \frac{\bar{A}\bar{D}}{t^2}$, $Y = T$, and $\hat{x}^0 = x_{i,0}$. It then follows that

$$f(x_{i,k}) - f^* \leq \frac{L_\mu}{\beta}\left(f(\hat{x}^0) - f_{\mathrm{slb}}\right)^2 + \mu\bar{D} \leq \frac{\bar{A}}{\mu\beta}\left(f(\hat{x}^0) - f_{\mathrm{slb}}\right)^2 + \mu\bar{D} = 2t(f(x_{i,0}) - f_{\mathrm{slb}}) \ ,$$

where the second inequality uses $L_\mu \leq \bar{A}/\mu$ from (17) and the last equality uses the values of μ and β. $\square$

Proposition 3.11. *Let i be the index of an outer iteration of Algorithm 4. Define $U := \left\lceil \frac{G\sqrt{2\bar{A}\bar{D}}}{\varepsilon' t} - 1 \right\rceil$. If $k \geq U$ and $y_{i,k}$ exists, then it holds that:*

$$f(y_{i,k}) - f^* \leq 2\varepsilon' t(f(x_{i,0}) - f_{\mathrm{slb}}) \ .$$

Proof: The proof follows by applying Proposition 3.9 with $\mu = \mu_i^2 = \frac{t\varepsilon' \cdot (f(x_{i,0}) - f_{\text{slb}})}{\bar{D}}$, $\beta = \frac{\bar{A}\bar{D}}{t^2(\varepsilon')^2}$, $Y = U$, and $\hat{x}^0 = x_{i,0}$. It then follows that

$$f(y_{i,k}) - f^* \leq \frac{L_\mu}{\beta}\left(f(\hat{x}^0) - f_{\text{slb}}\right)^2 + \mu\bar{D} \leq \frac{\bar{A}}{\mu\beta}\left(f(\hat{x}^0) - f_{\text{slb}}\right)^2 + \mu\bar{D} = 2t\varepsilon'(f(x_{i,0}) - f_{\text{slb}}) \ ,$$

where the second inequality uses $L_\mu \leq \bar{A}/\mu$ from (17) and the final equality derives from substituting in the values of μ and β. □

The next three propositions pertain to Algorithm 4 as well as to a more general setting which will be used in Section 4 to prove computational guarantees for algorithms when $f(\cdot)$ is smooth. The more general setting is described in the body of the following proposition.

Proposition 3.12. *Let $B, v > 0$ be constants satisfying $B - B^2 \geq v$, $B \geq 2v$. Consider an algorithm with outer and inner iterations indexed with counters i and j, respectively (such as Algorithm 4), with initial iterate x^0 that is used to set $x_{1,0} = y_{1,0} \leftarrow x^0$ in simultaneous running of the Accelerated Gradient Method using the same indexing notation as in Algorithm 4, and where $x_{i+1,0} = y_{i+1,0} \leftarrow x_{i,K_i}$ where $K_i \leftarrow j$ denotes the first index j for which $\frac{f(x_{i,j}) - f_{\text{slb}}}{f(x_{i,0}) - f_{\text{slb}}} \leq B$. Suppose that there are nonnegative sequences $\{J_i\}$ and $\{I_i\}$ indexed over the outer iteration counter i such that the following conditions are satisfied:*

(i) for all $k \geq J_i$ it holds that $f(x_{i,k}) - f^* \leq v\,(f(x_{i,0}) - f_{\text{slb}})$, and

(ii) for all $k \geq I_i$ it holds that $f(y_{i,k}) - f^* \leq v\varepsilon'\,(f(x_{i,0}) - f_{\text{slb}})$.

Let p denote the number of outer iterations i for which K_i is finite. Then

$$p \leq \left\lfloor \frac{\ln\left(1 + \frac{f(x^0) - f^*}{f^* - f_{\text{slb}}}\right)}{\ln(1/B)} \right\rfloor \ .$$

Furthermore, if $i \geq 1$ and $i \leq p - 1$, then $K_i \leq J_i$.

Proof: If $p = 0$ the results follow trivially, so let us suppose that $p \geq 1$, whereby K_p is finite and $x_{p+1,0}$ exists. It then follows that $f^* - f_{\text{slb}} \leq f(x_{p+1,0}) - f_{\text{slb}} \leq B^p(f(x_{1,0}) - f_{\text{slb}})$, and taking logarithms yields the proof of the bound on p.

Suppose additionally that $i \geq 1$ and $i \leq p-1$. Let us assume that $K_i \geq J_i + 1$, from which we will derive a contradiction. We have

$$f(x_{i,K_i-1}) - f^* \leq v(f(x_{i,0}) - f_{\text{slb}}) \leq (B - B^2)(f(x_{i,0}) - f_{\text{slb}}) \ ,$$

where the first inequality uses condition (i) and the second inequality uses $B - B^2 \geq v$. Also, $i+2 \leq p+1$, whereby $x_{i+2,0}$ exists and therefore satisfies $f^* - f_{\text{slb}} \leq f(x_{i+2,0}) - f_{\text{slb}} \leq B^2(f(x_{i,0}) - f_{\text{slb}})$. Combining this inequality with that above yields $f(x_{i,K_i-1}) - f^* \leq B(f(x_{i,0}) - f_{\text{slb}}) - f^* + f_{\text{slb}}$, which rearranges to yield:

$$\frac{f(x_{i,K_i-1}) - f_{\text{slb}}}{f(x_{i,0}) - f_{\text{slb}}} \leq B \ ,$$

and which contradicts the definition of K_i. Therefore $K_i \leq J_i$. □

Proposition 3.13. *Under the same setting, notation, and conditions* (i) *and* (ii) *of Proposition 3.12, let* i *be the index of an outer iteration. If* $j \geq J_i$ *and* $x_{i,j+1}$ *exists, then:*

$$\frac{f(x_{i,0}) - f_{\text{slb}}}{f^* - f_{\text{slb}}} \leq \frac{1}{B - v} \ .$$

Furthermore, if also $j \geq \max\{J_i, I_i\}$*, then*

$$\frac{f(y_{i,j}) - f^*}{f^* - f_{\text{slb}}} \leq \varepsilon' \ .$$

Proof: Since $j \geq J_i$ it follows from condition (i) that

$$f(x_{i,j}) - f^* \leq v(f(x_{i,0}) - f_{\text{slb}}) \ ,$$

and also since $x_{i,j+1}$ exists then $K_i \geq j + 1$, whereby:

$$\frac{f(x_{i,j}) - f_{\text{slb}}}{f(x_{i,0}) - f_{\text{slb}}} > B \ .$$

It then follows from these two inequalities that

$$\frac{f(x_{i,0}) - f_{\text{slb}}}{f^* - f_{\text{slb}}} = \frac{1}{\frac{f(x_{i,j}) - f_{\text{slb}}}{f(x_{i,0}) - f_{\text{slb}}} - \frac{f(x_{i,j}) - f^*}{f(x_{i,0}) - f_{\text{slb}}}} \leq \frac{1}{B - v} \ . \tag{20}$$

If also $j \geq I_i$, then we have from condition (ii) that

$$f(y_{i,j}) - f^* \leq v\varepsilon'(f(x_{i,0}) - f_{\text{slb}}) \leq \frac{v\varepsilon'}{B - v}(f^* - f_{\text{slb}}) \leq (f^* - f_{\text{slb}})\varepsilon' \ ,$$

where the first inequality is from condition (ii), the second inequality uses (20), and the third inequality uses $B \geq 2v$. □

Proposition 3.14. *Under the same setting, notation, and conditions* (i) *and* (ii) *of Proposition 3.12, let* $\hat{N}$ *count the total number of inner iterations prior to and including the first iteration for which* $y_{i,j}$ *is an* ε'*-relative solution* (2)*. Then*

$$\textit{either} \quad (i)\ \hat{N} \leq \sum_{i=1}^{p+1} J_i + I_{p+1} \ , \qquad \textit{or} \qquad (ii)\ \hat{N} \leq \sum_{i=1}^{p+1} J_i + I_p + I_{p+1} \ \textit{and}\ K_p \geq J_p + 1 \ . \tag{21}$$

Proof: First consider the case when $p = 0$. Then $K_1 = +\infty$ and therefore with $i = 1$ we have $x_{i,j+1}$ exists for $j = \max\{J_1, I_1\}$, whereby from Proposition 3.13 it holds that $y_{1,j}$ satisfies (2). In this case $\hat{N} \leq j = \max\{J_1, I_1\} \leq J_1 + I_1 = \sum_{i=1}^{p+1} J_i + I_{p+1}$ and therefore (i) of (21) is satisfied.

Next consider the case where $p \geq 1$ and $K_p \geq \max\{J_p, I_p\}+1$. Let i be the index of an outer iterate. If $i \leq p - 1$ it follows from Proposition 3.12 that $K_i \leq J_i$. For $i = p$ it holds for this case that $K_p \geq \max\{J_p, I_p\} + 1$, and it follows from Proposition 3.13 that $x_{p,j+1}$ exists for $j = \max\{J_p, I_p\}$ and therefore $y_{p,j}$ satisfies (2). In this case $\hat{N} \leq \sum_{i=1}^{p-1} K_i + \max\{J_p, I_p\} \leq \sum_{i=1}^{p-1} J_i + \max\{J_p, I_p\} \leq \sum_{i=1}^{p} J_i + I_p$ and $K_p \geq J_p + 1$ whereby (ii) of (21) is satisfied.

Next consider the case where $p \geq 1$ and $K_p \leq \max\{J_p, I_p\}$ and also $K_p \leq J_p$. Let i be the index of an outer iterate. If $i \leq p-1$ it follows from Proposition 3.12 that $K_i \leq J_i$. Since $K_{p+1} = +\infty$ it follows that $x_{p+1,j+1}$ exists for $j = \max\{J_{p+1}, I_{p+1}\}$, whereby from Proposition 3.13 we have $y_{p+1,j}$ satisfies (2). And since $K_p \leq J_p$ in this case, it follows that $\hat{N} \leq \sum_{i=1}^{p-1} K_i + J_p + \max\{J_{p+1}, I_{p+1}\} \leq \sum_{i=1}^{p+1} J_i + I_{p+1}$, and therefore (i) of (21) is satisfied.

The last case is where $p \geq 1$ and $K_p \leq \max\{J_p, I_p\}$ and also $K_p \geq J_p + 1$. Then just as in the third case above, we arrive at $\hat{N} \leq \sum_{i=1}^{p-1} K_i + \max\{J_p, I_p\} + \max\{J_{p+1}, I_{p+1}\} \leq \sum_{i=1}^{p+1} J_i + I_p + I_{p+1}$, and thus (ii) of (21) is satisfied, thereby proving (21). □

Proof of Theorem 3.3: Algorithm 4 satisfies the setting of Proposition 3.12, and it follows from Propositions 3.10 and 3.11 that Algorithm 4 satisfies conditions (i) and (ii) of Proposition 3.12 by letting $v = 2t$, $J_i = T$, and $I_i = U$ for all outer iterations i. Therefore the conclusions of Propositions 3.12, 3.13, and 3.14 all hold true. Let N denote the total number of iterates of Algorithm 4 computed prior to and including the first iterate $y_{i,j}$ that is an ε'-relative solution (2). Since two iterates are computed at each iteration, we have $N = 2\hat{N}$ (where $\hat{N}$ is defined in Proposition 3.14) and it follows from Proposition 3.14 that $N = 2\hat{N} \leq 2\sum_{i=1}^{p+1} J_i + 2I_p + 2I_{p+1}$, since the right-side of this inequality dominates both bounds (i) and (ii) of (21). Substituting in the values of T and U and the bound on p from Proposition 3.12 we obtain:

$$\begin{aligned} N \;\leq\; 2(p+1)T + 4U \quad &\leq \quad 2\left\lfloor 1 + \frac{\ln\left(1+\frac{f(x^0)-f^*}{f^*-f_{\rm slb}}\right)}{\ln(1/B)} \right\rfloor \left\lceil \frac{G\sqrt{2\bar{A}\bar{D}}}{t} - 1 \right\rceil + 4\left\lceil \frac{G\sqrt{2\bar{A}\bar{D}}}{\varepsilon' t} - 1 \right\rceil \\ &\leq \quad G\sqrt{\bar{A}\bar{D}}\left(22.63 + 32.65\ln\left(1+\tfrac{f(x^0)-f^*}{f^*-f_{\rm slb}}\right) \;+\; 45.26\left(\frac{1}{\varepsilon'}\right)\right), \end{aligned}$$

where the third inequality follows from substituting in the values $B = \frac{1}{2}$ and $t = \frac{1}{8}$, which then rounds up to the desired bound in the theorem. □

4 Computational Guarantees when $f(\cdot)$ is Smooth

In this section we study the computational complexity of solving (1) in the case when $f(\cdot)$ is convex and differentiable on an open set containing Q. We assume that $\nabla f(\cdot)$ is Lipschitz on Q as defined in (8).

Let us first consider directly applying the Accelerated Gradient Method (Algorithm 2) to solve (1), and let us apply Theorem 2.2. Let $\varepsilon' > 0$ denote the relative accuracy, and note again that an ε'-relative solution of (1) corresponds to an absolute ε-solution for $\varepsilon := \varepsilon' \cdot (f^* - f_{\rm slb})$. Let $x^0 \in Q$ be the initial point. It then follows from Theorem 2.2 using $\delta = f^*$ (whereby $S_\delta = \{x \in Q : f(x) \leq f^*\} = \text{Opt}$) that if

$$N \;\geq\; \frac{\sqrt{2}\sqrt{L}\text{Dist}(x^0, \text{Opt})}{\sqrt{\varepsilon'}\sqrt{f^* - f_{\rm slb}}} \;-1\;, \tag{22}$$

then

$$\frac{f(x^N) - f^*}{f^* - f_{\rm slb}} \;\leq\; \varepsilon'\;.$$

Herein we will derive a new computational guarantee for a version of the Accelerated Gradient Method that can improve on (22) in many cases. Our new version of the Accelerated Gradient Method periodically restarts the method with an appropriate rule for deciding when to do the restarts, and is presented in Algorithm 5. At the i^{th} outer iteration of Algorithm 5 the algorithm starts the Accelerated Gradient Method at the point $x_{i,0}$ for optimizing $f(\cdot)$ on Q. The outer iteration i runs until the ratio test in Step (2a.) fails, at which point the current point $x_{i,j}$ becomes the starting point of the next outer iteration, namely $x_{i+1,0} \leftarrow x_{i,j}$. The counter K_i records the number of inner iterations computed in outer iteration i. Similar to the notation in Algorithm 4, the notation "$x_{i,j} \leftarrow \text{AGM}(f(\cdot), x_{i,0}, j)$" in Algorithm 5 denotes assigning to $x_{i,j}$ the j^{th} iterate of the Accelerated Gradient Method applied to the optimization problem (1) with objective function $f(\cdot)$ using the initial point $x_{i,0} \in Q$.

Algorithm 5 Accelerated Gradient Method with Simple Restarting

Initialize. Initialize with $x^0 \in Q$.
Define $B := 0.5$
Set $x_{1,0} \leftarrow x^0$, $i \leftarrow 1$.

At outer iteration i :

1. **Initialize inner iteration.** $K_i \leftarrow +\infty$, $j \leftarrow 0$
2. **Run inner iterations.** At inner iteration j:

(2a.) If $\dfrac{f(x_{i,j}) - f_{\text{slb}}}{f(x_{i,0}) - f_{\text{slb}}} > B$, then

$$x_{i,j+1} \leftarrow \text{AGM}(f(\cdot),\ x_{i,0},\ j+1) ,$$

$j \leftarrow j+1$, and Goto (2a.).

Else $K_i \leftarrow j$, $x_{i+1,0} \leftarrow x_{i,j}$, $i \leftarrow i+1$, and Goto step 1.

We have the following computational guarantee associated with Algorithm 5.

Theorem 4.1. (Complexity Bound for Accelerated Gradient Method with Simple Restarting) *Within a total number of computed iterates that does not exceed*

$$G\sqrt{L}\left(10\sqrt{f(x^0) - f_{\text{slb}}} \ +\ 12\left[\frac{\sqrt{f^* - f_{\text{slb}}}}{\sqrt{\varepsilon'}}\right]\right) ,$$

the Accelerated Gradient Method with Simple Restarting (Algorithm 5) will compute an iterate $x_{i,j}$ for which

$$\frac{f(x_{i,j}) - f^*}{f^* - f_{\text{slb}}} \ \leq\ \varepsilon' . \qquad \square$$

The computational guarantee in Theorem 4.1 can itself be bounded by:

$$G\sqrt{L}\left(10\sqrt{f^* - f_{\text{slb}}} \ + 10\sqrt{\tfrac{L}{2}}\text{Dist}(x^0, \text{Opt}) \ +\ 12\left[\frac{\sqrt{f^* - f_{\text{slb}}}}{\sqrt{\varepsilon'}}\right]\right) , \tag{23}$$

which follows from the chain of inequalities:

$$\begin{aligned}\sqrt{f(x^0)-f_{\text{slb}}} &= \sqrt{(f^*-f_{\text{slb}})+(f(x^0)-f^*)} \\ &\leq \sqrt{(f^*-f_{\text{slb}})+\tfrac{L}{2}\text{Dist}(x^0,\text{Opt})^2} \;\leq\; \sqrt{(f^*-f_{\text{slb}})}+\sqrt{\tfrac{L}{2}}\text{Dist}(x^0,\text{Opt}) \;.\end{aligned}$$

Comparing (23) with the standard bound for the Accelerated Gradient Method given in (22), we see that the factor involving $1/\sqrt{\varepsilon'}$ in (23) is independent of $\text{Dist}(x^0,\text{Opt})$, unlike the standard bound (22).

Towards the proof of Theorem 4.1, for notational convenience we will work with two constants B and v that must be chosen to satisfy

$$B>0 \;,\; v>0 \;,\; B-B^2\geq v \;,\; \text{and } B\geq 2v \;, \tag{24}$$

and whose specific values are set to $B=0.5$ in Algorithm 4, and $v=0.25$.

Proposition 4.1. *Let i be the index of an outer iteration of Algorithm 5. Define $J_i := \left\lceil G\sqrt{\frac{2L(f(x_{i,0})-f_{\text{slb}})}{v}}-1\right\rceil$. If $k\geq J_i$ and $x_{i,k}$ exists, then it holds that:*

$$f(x_{i,k})-f^*\leq v(f(x_{i,0})-f_{\text{slb}}) \;. \tag{25}$$

Proof: It follows from Theorem 2.2 applied to the function $f(\cdot)$ and using $\delta=f^*$ that for any $k\geq J_i$ we have:

$$f(x_{i,k})-f^*\leq \frac{2L}{(J_i+1)^2}\text{Dist}(x_{i,0},\text{Opt})^2\leq \frac{2L}{(J_i+1)^2}G^2\left(f(x_{i,0})-f_{\text{slb}}\right)^2\leq v\left(f(x_{i,0})-f_{\text{slb}}\right) \;,$$

where the second inequality uses the definition of G, and the last inequality uses the value of J_i. □

Proposition 4.2. *Let i be the index of an outer iteration of Algorithm 5. Define $I_i := \left\lceil G\sqrt{\frac{2L(f(x_{i,0})-f_{\text{slb}})}{v\varepsilon'}}-1\right\rceil$. If $k\geq I_i$ and $x_{i,k}$ exists, then it holds that:*

$$f(x_{i,k})-f^*\leq v\varepsilon'(f(x_{i,0})-f_{\text{slb}}) \;.$$

Proof: The proof follows using identical logic as in Proposition 4.1. □

Proof of Theorem 4.1: Even though Algorithm 5 does not simultaneously run two versions of the Accelerated Gradient Method, we can still view Algorithm 5 as an instance of the general algorithm setting of Proposition 3.12 by simply defining $y_{i,j}:=x_{i,j}$ for all i,j. It follows from Propositions 4.1 and 4.2 that Algorithm 5 satisfies conditions (i) and (ii) of Proposition 3.12, and therefore Propositions 3.12, 3.13, and 3.14 hold for Algorithm 5. Substituting in the values of J_i and using

the fact that $f(x_{i,0}) - f_{\rm slb} \leq B^{i-1}(f(x_{1,0}) - f_{\rm slb})$ for all iteration counters i, we obtain:

$$\begin{aligned}
\sum_{i=1}^{p+1} J_i &\leq \sum_{i=1}^{p+1} G\sqrt{\frac{2L(f(x_{i,0}) - f_{\rm slb})}{v}} \\
&\leq \left(\sum_{i=0}^{p} B^{\frac{i}{2}}\right) G\sqrt{\frac{2L(f(x_{1,0}) - f_{\rm slb})}{v}} \\
&< \left(\sum_{i=0}^{\infty} B^{\frac{i}{2}}\right) G\sqrt{\frac{2L(f(x_{1,0}) - f_{\rm slb})}{v}} \;=\; \frac{G}{1-\sqrt{B}}\sqrt{\frac{2L(f(x^0) - f_{\rm slb})}{v}} \;.
\end{aligned}$$

Next observe that $K_{p+1} = \infty \geq J_p$, whereby it follows from Proposition 3.13 with $i = p+1$ that $f(x_{p+1,0}) - f_{\rm slb} \leq \frac{1}{B-v}(f^* - f_{\rm slb})$, and therefore it holds that:

$$I_{p+1} \leq G\sqrt{\frac{2L(f(x_{p+1,0}) - f_{\rm slb})}{v\varepsilon'}} \leq G\sqrt{\frac{2L(f^* - f_{\rm slb})}{(B-v)v\varepsilon'}} \;.$$

Also, if $K_p \geq J_p + 1$, then similarly applying Proposition 3.13 with $i = p$ using the logic above implies that $I_p \leq G\sqrt{\frac{2L(f^* - f_{\rm slb})}{(B-v)v\varepsilon'}}$.

Let N denote the total number of iterates of Algorithm 5 computed prior to and including the first iterate $x_{i,j}$ that is an ε'-relative solution (2). Then $N = \hat{N}$ where $\hat{N}$ is defined in Proposition 3.14. In either case *(i)* or *(ii)* of (21), it follows from Proposition 3.14 that:

$$\begin{aligned}
N \;=\; \hat{N} &\leq \sum_{i=1}^{p+1} J_i + I_{p+1} + G\sqrt{\frac{2L(f^* - f_{\rm slb})}{(B-v)v\varepsilon'}} \\
&\leq \frac{G}{1-\sqrt{B}}\sqrt{\frac{2L(f(x^0) - f_{\rm slb})}{v}} + 2G\sqrt{\frac{2L(f^* - f_{\rm slb})}{(B-v)v\varepsilon'}} \\
&\leq G\sqrt{L}\left(9.66\sqrt{f(x^0) - f_{\rm slb}} \;+\; \frac{11.32\sqrt{f^* - f_{\rm slb}}}{\sqrt{\varepsilon'}}\right) ,
\end{aligned}$$

where the third inequality follows from substituting in the values $B = \frac{1}{2}$ and $v = \frac{1}{4}$, which then rounds up to the bound stated in the theorem. □

It turns out that we can further improve the computational guarantee of Theorem 4.1 by further modifying the Accelerated Gradient Method with Simple Restarting (Algorithm 5), if we know and can easily work with an adjoint representation of $f(\cdot)$ to do "extra smoothing." Let us see how this can be done. We will assume that $f(\cdot)$ has the representation:

$$f(x) = \max_{\lambda \in P}\{\lambda^T A x - d(\lambda)\} \;, \tag{26}$$

where P is a convex set and $d(\cdot)$ is a strongly convex function on P with strong convexity parameter σ and for which $\min_{\lambda \in P} d(\lambda) \geq 0$. (See [14] for properties of strongly convex functions.) It then

follows that $f(\cdot)$ is a globally smooth convex function with Lipschitz constant at most $L := \|A\|^2/\sigma$, see Nesterov [15]. We presume further that A, $d(\cdot)$, and P are given and that the optimization problem in (26) is simple to solve. That being the case, for a given $x \in Q$, if $\tilde{\lambda}$ solves the optimization problem (26), then it holds that $f(x) = \tilde{\lambda}^T Ax - d(\tilde{\lambda})$ and $\nabla f(x) = A^T\tilde{\lambda}$.

In a similar spirit as the smoothing technique employed in Section 3.3, we will consider parametrically working with a modification $f_\mu(\cdot)$ of $f(\cdot)$ that is more smooth than $f(\cdot)$ by increasing the weight on the the strongly convex function $d(\cdot)$ in (26). For any $\mu \geq 0$ define the function $f_\mu(\cdot)$ by:

$$f_\mu(x) = \max_{\lambda \in P}\{\lambda^T Ax - (1+\mu)d(\lambda)\} \ . \tag{27}$$

If P is bounded, then $\bar{D} := \max_{\lambda\in P}\{d(\lambda)\}$ is finite, and the above smoothing technique has the following two properties:

(i) $f_\mu(\cdot)$ is not far from $f(\cdot)$,

$$f(x) - \bar{D}\mu \ \leq \ f_\mu(x) \ \leq \ f(x) \quad \text{for all } x \in Q \ , \text{ and} \tag{28}$$

(ii) $f_\mu(\cdot)$ has Lipschitz continuous gradient on Q with Lipschitz constant L_μ satisfying

$$L_\mu \leq L/(1+\mu) \ . \tag{29}$$

This setting is very similar to the properties we have for smoothing of a non-smooth function $f(\cdot)$ in Section 3.3, and the only difference is that the Lipschitz constant L_μ here is bounded above by $L/(1+\mu)$ instead of by $\bar{A}/\mu$ as was the case in (17).

Let $\varepsilon' > 0$ be given. As before, we aspire to compute an ε'-relative solution of (1) as defined in (2). We will use and analyze the Parametric Smoothing/Rescaling Method (Algorithm 4) but with $f_\mu(\cdot)$ defined by (27) and hence satisfying (28) and (29). We have the following computational guarantee associated with Algorithm 4 applied to the case when $f(\cdot)$ is smooth and $f_\mu(\cdot)$ is given by (27).

Theorem 4.2. (Complexity Bound for Parametric Smoothing/Restarting Method (Algorithm 4) for Smooth Optimization) *Suppose that $f_\mu(\cdot)$ is given by* (27) *and hence satisfies* (28) *and* (29)*. Within a total number of computed iterates that does not exceed*

$$G\sqrt{L\bar{D}}\left(22.7 + 32.7\ln\left(1 + \frac{f(x^0) - f^*}{f^* - f_{\text{slb}}}\right) + 32\sqrt{\frac{f^* - f_{\text{slb}}}{\varepsilon'}}\right) \ ,$$

Algorithm 4 will compute an iterate $y_{i,j}$ for which

$$\frac{f(y_{i,j}) - f^*}{f^* - f_{\text{slb}}} \leq \varepsilon' \ . \qquad \square$$

The dependence in Theorem 4.2 on the quality of the initial point is logarithmic in the optimality gap $f(x^0) - f^*$, while it is the square root of the optimality gap in Theorem 4.1. We will prove Theorem 4.2 by first proving two propositions. For notational convenience we will work with two constants B and t, whose specific values are $B = \frac{1}{2}$ and $t = \frac{1}{8}$.

Proposition 4.3. *Let i be the index of an outer iteration of Algorithm 4. Define $T := \left\lceil G\sqrt{\frac{2L\bar{D}}{t^2}} - 1 \right\rceil$. If $k \geq T$ and $x_{i,k}$ exists, then:*

$$f(x_{i,k}) - f^* \leq 2t(f(x_{i,0}) - f_{\text{slb}}) \ .$$

Proof: The proof follows by applying Proposition 3.9 with $\mu = \mu_i^1 = \frac{t\cdot(f(x_{i,0})-f_{\text{slb}})}{\bar{D}}$, $\beta = \frac{L\bar{D}}{t^2}$, $Y = T$, and $\hat{x}^0 = x_{i,0}$. It then follows that

$$f(x_{i,k}) - f^* \leq \frac{L_\mu}{\beta}\left(f(\hat{x}^0) - f_{\text{slb}}\right)^2 + \mu\bar{D} \leq \frac{L}{\mu\beta}\left(f(x_{i,0}) - f_{\text{slb}}\right)^2 + \mu\bar{D} = 2t(f(x_{i,0}) - f_{\text{slb}}) \ ,$$

where the second inequality uses $L_\mu \leq L/(1+\mu) \leq L/\mu$ from (29) and the final equality derives from substituting in the values of μ and β. □

Proposition 4.4. *Let i be the index of an outer iteration of Algorithm 4. Define $I_i := \left\lceil G\sqrt{\frac{2L(f(x_{i,0})-f_{\text{slb}})}{t\varepsilon'}} - 1 \right\rceil$. If $k \geq I_i$ and $y_{i,k}$ exists, then:*

$$f(y_{i,k}) - f^* \leq 2t\varepsilon'(f(x_{i,0}) - f_{\text{slb}}) \ .$$

Proof: The proof follows by applying Proposition 3.9 with $\mu = \mu_i^2 = \frac{t\varepsilon'\cdot(f(x_{i,0})-f_{\text{slb}})}{\bar{D}}$, $\beta = \frac{L(f(x_{i,0})-f_{\text{slb}})}{t\varepsilon'}$, $Y = I_i$, and $\hat{x}^0 = x_{i,0}$. It then follows that

$$f(y_{i,k}) - f^* \leq \frac{L_\mu}{\beta}\left(f(\hat{x}^0) - f_{\text{slb}}\right)^2 + \mu\bar{D} \leq \frac{L}{\beta}\left(f(x_{i,0}) - f_{\text{slb}}\right)^2 + \mu\bar{D} = 2t\varepsilon'(f(x_{i,0}) - f_{\text{slb}}) \ ,$$

where the second inequality uses $L_\mu \leq L/(1+\mu) \leq L$ from (29) and the final equality derives from substituting in the values of μ and β. □

Proof of Theorem 4.2: Algorithm 4 satisfies the setting of Proposition 3.12, and it follows from Propositions 4.3 and 4.4 that Algorithm 4 satisfies conditions (i) and (ii) of Proposition 3.12 by letting $v = 2t$ and $J_i = T$ for all outer iterations i. Therefore the conclusions of Propositions 3.12, 3.13, and 3.14 all hold true. Let N denote the total number of iterates of Algorithm 4 computed prior to and including the first iterate $y_{i,j}$ that is an ε'-relative solution (2). Since two iterates are computed at each iteration, we have $N = 2\hat{N}$, where $\hat{N}$ is defined in Proposition 3.14 and is bounded by either (i) or (ii) of (21).

Note that $K_{p+1} = \infty \geq T = J_i$, whereby it follows from Proposition 3.13 that $f(x_{p+1,0}) - f_{\text{slb}} \leq \frac{1}{B-2t}(f^* - f_{\text{slb}})$, and therefore

$$I_{p+1} \leq G\sqrt{\frac{2L(f(x_{p+1,0}) - f_{\text{slb}})}{t\varepsilon'}} \leq G\sqrt{\frac{2L(f^* - f_{\text{slb}})}{(B-2t)t\varepsilon'}} \ .$$

Similarly, if $K_p \geq T + 1 = J_i + 1$, similar logic demonstrates that $I_p \leq G\sqrt{\frac{2L(f^*-f_{\text{slb}})}{(B-2t)t\varepsilon'}}$. Therefore, in either case (i) or (ii) of (21) it holds that:

$$
\begin{aligned}
N \;=\; 2\hat{N} \quad &\leq \quad 2\sum_{i=1}^{p+1} J_i + 2I_{p+1} + 2G\sqrt{\tfrac{2L(f^*-f_{\rm slb})}{(B-2t)t\varepsilon'}} \\
&\leq \quad 2(p+1)\left\lceil G\sqrt{\tfrac{2L\bar{D}}{t^2}} - 1\right\rceil + 2I_{p+1} + 2G\sqrt{\tfrac{2L(f^*-f_{\rm slb})}{(B-2t)t\varepsilon'}} \\
&\leq \quad 2\left(1 + \frac{\ln\left(1+\frac{f(x^0)-f^*}{f^*-f_{\rm slb}}\right)}{\ln(1/B)}\right) G\sqrt{\tfrac{2L\bar{D}}{t^2}} + 4G\sqrt{\tfrac{2L(f^*-f_{\rm slb})}{(B-2t)t\varepsilon'}} \\
&\leq \quad G\sqrt{L\bar{D}}\left(22.7 + 32.7\ln\left(1+\tfrac{f(x^0)-f^*}{f^*-f_{\rm slb}}\right) + 32\sqrt{\tfrac{f^*-f_{\rm slb}}{\varepsilon'}}\right) \;,
\end{aligned}
$$

where the third inequality follows from substituting in the values $B = \frac{1}{2}$ and $t = \frac{1}{8}$, which then rounds up to the desired bound in the theorem. □

Acknowledgement

The authors thank James Renegar for helpful discussions and encouragement. The authors are also grateful to the two referees for their comprehensive efforts and their suggestions on ways to improve the readability of the paper.

A Appendix

A.1 Growth Constant G and the Modulus of Weak Sharp Minima

The optimal solution set Opt of (1) is called a set of *weak sharp minima* with modulus α if it holds that:

$$
f(x) \geq f^* + \alpha \cdot \mathrm{Dist}(x, \mathrm{Opt}) \;\; \text{for all } x \in Q \;. \tag{30}
$$

This concept was first developed by Polyak [18] when Opt is a singleton, and generalized by Burke and Ferris [6] to include the possibility of multiple optima. The modulus of weak sharp minima has been a useful tool in sensitivity analysis [8, 12], convergence analysis for certain problem classes [7, 6], linear regularity and error bounds [3, 4, 5], perturbation properties of nonlinear optimization [24, 25, 2], as well as in the finite termination of certain algorithms [19], [9], and [7].

Comparing (30) to (5), we see that the modulus α of weak sharp minima is a close cousin of the growth constant G. Indeed, if we were to loosen the restriction that $f_{\rm slb}$ be a strict lower bound and instead allow it to take the value $f_{\rm slb} = f^*$ in the definition of G in (3), then we would obtain precisely that $G = \alpha^{-1}$. However, the notion of $f_{\rm slb}$ being a strict lower bound is fundamental for the results herein.

Note that (30) specifies the exact local growth of $f(\cdot)$ away from the set of optimal solutions. And although as defined in (30) the weak sharp minima is a global property, due to convexity it is essentially a local property and indeed its usefulness derives from the local nature of the weak sharp minima in a neighborhood of the optimal solution set. This is in contrast to the growth constant G as defined in (3), which by its nature is a global property as illustrated in the constructions in

Figure 1. Last of all we point out that while one can easily have $\alpha = 0$ for weak sharp minima (just let $f(x)$ be a differentiable convex function whose optimum is attained in the relative interior of Q), Theorem 1.1 shows that G is finite for all reasonably-behaved convex functions.

A.2 Proof of Theorem 1.1

Proof of Theorem 1.1: Let us fix an optimal solution $x^* \in \mathrm{Opt}$, and define $\delta := \max_{v \in E_\varepsilon} \|v - x^*\|$ and define $\bar{G} := \max\{\frac{\delta}{\varepsilon}, \frac{\delta}{f^* - f_{\mathrm{slb}}}\}$. We will prove that for any $x \in Q$, the following inequality holds:

$$\mathrm{Dist}\,(x, \mathrm{Opt}) \leq \bar{G}\,(f(x) - f_{\mathrm{slb}}) \ , \tag{31}$$

which then implies that $G \leq \bar{G}$ is finite. We consider two cases as follows:

Case (i): $x \in \mathrm{Opt}_\varepsilon$. In this case we have $x = v + s$ where $v \in E_\varepsilon$ and $s \in S$. Since s is in the recession cone of Opt_ε it holds that $x^* + s \in \mathrm{Opt}$, whereby

$$\mathrm{Dist}(x, \mathrm{Opt}) \leq \|x - (x^* + s)\| = \|v - x^*\| \leq \delta \ , \tag{32}$$

and therefore

$$f(x) - f_{\mathrm{slb}} \geq f^* - f_{\mathrm{slb}} \geq \frac{(f^* - f_{\mathrm{slb}})(\mathrm{Dist}(x, \mathrm{Opt}))}{\delta} \geq \bar{G}^{-1}\mathrm{Dist}(x, \mathrm{Opt}) \ ,$$

which shows (31) in this case.

Case (ii): $x \notin \mathrm{Opt}_\varepsilon$. Let x^1 be the projection of x onto Opt and let x^2 be the point on the line segment from x^1 to x that satisfies $f(x^2) = f^* + \varepsilon$. (Existence of x^2 is guaranteed by continuity of $f(\cdot)$.) Then

$$f(x) - f_{\mathrm{slb}} \geq f(x) - f^* \geq (f(x^2) - f^*)\frac{\|x - x^1\|}{\|x^2 - x^1\|} \geq \frac{\varepsilon\|x - x^1\|}{\delta} = \frac{\varepsilon \mathrm{Dist}(x, \mathrm{Opt})}{\delta} \geq \bar{G}^{-1}\mathrm{Dist}(x, \mathrm{Opt}) \ ,$$

where the second inequality is from the convexity of $f(\cdot)$ which implies the chordal inequality $\frac{f(x) - f^*}{\|x - x^1\|} \geq \frac{f(x^2) - f^*}{\|x^2 - x^1\|}$, and the third inequality uses $\|x^2 - x^1\| = \mathrm{Dist}(x^2, \mathrm{Opt}) \leq \delta$ (from (32)). The last equality above uses the fact that $\mathrm{Dist}(x, \mathrm{Opt}) = \|x - x^1\|$. This proves (31) in this case. □

References

[1] Amir Beck and Marc Teboulle, *Smoothing and first order methods: A unified framework*, SIAM Journal on Optimization **22** (2012), no. 2, 557–580.

[2] J. Frederic Bonnans and Alexander D. Ioffe, *Quadratic growth and stability in convex programming problems with multiple solutions*, Tech. report, INRIA Research Report RR-2403, 1994.

[3] James V. Burke and Sien Deng, *Weak sharp minima revisited, part I: basic theory*, Control and Cybernetics **31** (2002), no. 3, 439–469.

[4] ———, *Weak sharp minima revisited, part II: application to linear regularity and error bounds*, Mathematical programming **104** (2005), no. 2-3, 235–261.

[5] ———, *Weak sharp minima revisited, part III: Error bounds for differentiable convex inclusions*, Mathematical Programming **116** (2009), no. 1-2, 37–56.

[6] James V. Burke and Michael C. Ferris, *Weak sharp minima in mathematical programming*, SIAM Journal on Control and Optimization **31** (1993), no. 5, 1340–1359.

[7] James V. Burke and Michael C. Ferris, *A Gauss-Newton method for convex composite optimization*, Mathematical Programming **71** (1995), no. 2, 179–194.

[8] James V. Burke, Adrian S. Lewis, and Michael L. Overton, *Optimal stability and eigenvalue multiplicity*, Foundations of Computational Mathematics **1** (2001), no. 2, 205–225.

[9] Michael C. Ferris, *Finite termination of the proximal point algorithm*, Mathematical Programming **50** (1991), no. 1-3, 359–366.

[10] Andrew Gilpin, Javier Peña, and Tuomas Sandholm, *First-order algorithm with $O(\ln(1/\epsilon))$ convergence for ϵ-equilibrium in two-person zero-sum games*, Mathematical programming **133** (2012), no. 1-2, 279–298.

[11] Trevor Hastie, Robert Tibshirani, and Jerome Friedman, *Elements of statistical learning*, second ed., Springer Series in Statistics, Springer, New York, 2009.

[12] Abderrahim Jourani, *Hoffman's error bound, local controllability, and sensitivity analysis*, SIAM Journal on Control and Optimization **38** (2000), no. 3, 947–970.

[13] Arkadi S Nemirovsky and David B Yudin, *Problem complexity and method efficiency in optimization*, Wiley, New York, 1983.

[14] Yurii Nesterov, *Introductory lectures on convex optimization: a basic course*, Kluwer Academic Publishers, Boston, 2003.

[15] ———, *Smooth minimization of non-smooth functions*, Mathematical Programming **103** (2005), no. 1, 127–152.

[16] ———, *Smoothing technique and its applications in semidefinite optimization*, Mathematical Programming **110** (2007), no. 2, 245–259.

[17] Brendan O'Donoghue and Emmanuel Candes, *Adaptive restart for accelerated gradient schemes*, Foundations of Computational Mathematics **15** (2013), no. 3, 715–732.

[18] Boris Polyak, *Sharp minima*, Proceedings of the IIASA Workshop on Generalized Lagrangians and Their Applications, Laxenburg, Austria. Institute of Control Sciences Lecture Notes, Moscow, 1979.

[19] ———, *Introduction to optimization*, Optimization Software, Inc., New York, 1987.

[20] James Renegar, *Efficient first-order methods for linear programming and semidefinite programming*, preprint, arXiv:1409.5832 (2014).

[21] ———, *Accelerated first-order methods for hyperbolic programming*, preprint, arXiv:1512.07569 (2015).

[22] ———, *A framework for applying subgradient methods to conic optimization problems, version 2*, preprint, arXiv:1503.02611 (2015).

[23] ———, *"Efficient" subgradient methods for general convex optimization*, SIAM Journal on Optimization **26** (2016), no. 4, 2649–2676.

[24] Alexander Shapiro, *Perturbation theory of nonlinear programs when the set of optimal solutions is not a singleton*, Applied Mathematics and Optimization **18** (1988), no. 1, 215–229.

[25] ———, *Perturbation analysis of optimization problems in Banach spaces*, Numerical Functional Analysis and Optimization **13** (1992), no. 1-2, 97–116.

[26] Weijie Su, Stephen Boyd, and Emmanuel Candes, *A differential equation for modeling Nesterov's accelerated gradient method: theory and insights*, Advances in Neural Information Processing Systems, 2014, pp. 2510–2518.

[27] Paul Tseng, *On accelerated proximal gradient methods for convex-concave optimization*, Tech. report, May 21, 2008.

[28] Tianbao Yang and Qihang Lin, *RSG: Beating subgradient method without smoothness and strong convexity*, arXiv preprint arXiv:1512.03107 (2015).